```
Dec 16 15:43:43 helion-cp1-c1-m1-mgmt Keepalived_vrrp[2218]: bogus VRRP packet received on br-bond0 !!!
Dec 16 15:43:43 helion-cp1-c1-m1-mgmt Keepalived_vrrp[2218]: VRRP_Instance(VI_2) ignoring received advertisment...
```

To resolve this issue, our recommendation is to install your separate HPE Helion OpenStack 5.0 systems with VRRP traffic on different subnets.

If this is not possible, you may also assign a unique routerid to your separate HPE Helion OpenStack 5.0 system by changing the `keepalived_vrrp_offset` service configurable. The routerid is currently derived using the `keepalived_vrrp_index` which comes from a configuration processor variable and the `keepalived_vrrp_offset`.

For example,

1. Log in to your lifecycle manager.
2. Edit your `~/helion/my_cloud/config/keepalived/defaults.yml` file and change the value of the following line:

```
keepalived_vrrp_offset: 0
```

Change the off value to a number that uniquely identifies a separate vrrp cluster. For example:

`keepalived_vrrp_offset: 0` for the 1st vrrp cluster on this subnet.

`keepalived_vrrp_offset: 1` for the 2nd vrrp cluster on this subnet.

`keepalived_vrrp_offset: 2` for the 3rd vrrp cluster on this subnet.

> ⚠ **Important:** You should be aware that the files in the `~/helion/my_cloud/config/` directory are symlinks to the `~/helion/hos/ansible/` directory. For example, `~/helion/my_cloud/config/keepalived/defaults.yml` is a symbolic link to `~/helion/hos/ansible/roles/keepalived/defaults/main.yml`
>
> ```
> ls -al ~/helion/my_cloud/config/keepalived/defaults.yml
> lrwxrwxrwx 1 stack stack 55 May 24 20:38 /home/stack/helion/my_cloud/config/keepalived/defaults.yml -> ../../../hos/ansible/roles/keepalived/defaults/main.yml
> ```
>
> If you are using a tool like `sed` to make edits to files in this directory, you might break the symbolic link and create a new copy of the file. To maintain the link, you will need to force `sed` to follow the link:
>
> ```
> sed -i --follow-symlinks 's$keepalived_vrrp_offset: 0$keepalived_vrrp_offset: 2$' ~/helion/my_cloud/config/keepalived/defaults.yml
> ```
>
> Alternatively, you could directly edit the target of the link `~/helion/hos/ansible/roles/keepalived/defaults/main.yml`.

3. Commit your configuration to the *local git repo*, as follows:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "changing Admin password"
```

4. Run the configuration processor with this command:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

5. Use the playbook below to create a deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

6. If you are making this change after your initial install, run the following reconfigure playbook to make this change in your environment:

```
cd ~/scratch/ansible/next/hos/ansible/
ansible-playbook -i hosts/verb_hosts FND-CLU-reconfigure.yml
```

HPE Helion OpenStack® 5.0: Installation Overview

Before jumping into your installation, we recommend taking the time to read through our *Example Configurations* documentation to get an overview of the sample configurations HPE Helion OpenStack 5.0 offers. We have highly tuned example configurations for each of these Cloud models:

Name	Location
Entry-scale KVM with VSA model	`~/helion/examples/entry-scale-kvm-vsa`
Entry-scale KVM with VSA model with Dedicated Cluster for Metering, Monitoring, and Logging	`~/helion/examples/entry-scale-kvm-vsa-mml`
Entry-scale KVM with Ceph model	`~/helion/examples/entry-scale-kvm-ceph`
Mid-scale KVM with VSA model	`~/helion/examples/mid-scale-kvm-vsa`
Entry-scale ESX, KVM and VSA model	`~/helion/examples/entry-scale-esx-kvm-vsa`
Entry-scale ESX, KVM and VSA model with Dedicated Cluster for Metering, Monitoring, and Logging	`~/helion/examples/entry-scale-esx-kvm-vsa-mml`
Entry-scale Swift-only model	`~/helion/examples/entry-scale-swift`
Entry-scale Cloud with Ironic Flat Network	`~/helion/examples/entry-scale-ironic-flat-network`
Entry-scale Cloud with Ironic Multi-Tenancy	`~/helion/examples/entry-scale-ironic-multi-tenancy`

Using the Command-line

You should use the command-line if:

- You are installing a more complex or large-scale cloud.
- You have more than 15 nodes.
- You need to use availability zones or the server groups functionality of the cloud model. See the *Input Model page* for more information.
- You want to customize the cloud configuration beyond the tuned defaults that HPE provides out of the box.
- You need deeper customizations than are possible to express using the GUI.

Instructions for installing via the command-line are here:

- *Installing Mid-scale and Entry-scale KVM*
- *Installing ESX, KVM with VSA*

For More Information

Other useful documents that will help you understand, plan, configure, and install your cloud are listed below.

- *Using Git for Configuration Management*
- *Troubleshooting Your Installation*
- *Verifying Your Installation*
- *Common Post-Installation Steps*

HPE Helion OpenStack® 5.0: Pre-installation Overview

We have a *Pre-Installation Checklist* we recommend going through to ensure your environment meets the requirements of the Cloud model you choose.

HPE Helion OpenStack 5.0 Installation

Once you have an idea of the configuration you want to choose for your cloud and you have gone through the pre-installation steps, you have two options for installation. You can use a GUI that runs in your web browser or you can install via the command-line that exposes the full power and flexibility of HPE Helion OpenStack 5.0.

Using the GUI

You should use the GUI if:

- You are not planning to deploy availability zones or use L3 segmentation in your initial deployment.
- You are happy with the tuned HPE default OpenStack configurations.

Instructions for installing via the GUI are here.

- *Installing via the GUI*

Note that reconfiguration of your cloud, at the moment, can only be done via the command-line. The GUI installer is for initial installation only.

HPE Helion OpenStack® 5.0: Pre-Installation Checklist

Attention: The formatting of this page facilitates printing it out and using it to record details of your setup.

This checklist is focused on the Entry-scale KVM with VSA model but you can alter it to fit the example configuration you chose for your cloud.

BIOS and iLO Settings

Ensure that the following BIOS and iLO settings are applied to each baremetal server:

#	Item
	Setup the iLO Advanced license in the iLO configuration
	Choose either UEFI or Legacy BIOS in the BIOS settings
	Verify the Date and Time settings in the BIOS. **Note:** It is important that all of your systems have the correct date/time because the HPE StoreVirtual VSA license has a start date. If the start date is later than the system time then the installation will fail. Note also that Helion installs and runs with UTC, not local time.
	Ensure that Wake-on-LAN is disabled in the BIOS
	Ensure that the NIC port to be used for PXE installation has PXE enabled in the BIOS
	Ensure that all other NIC ports have PXE disabled in the BIOS
	Ensure all hardware in the server not directly used by Helion is disabled

Network Setup and Configuration

Before installing HPE Helion OpenStack, the following networks must be provisioned and tested. The networks are not installed or managed by the Cloud. You must install and manage the networks as documented in the *Example Configurations*.

Note that if you want a pluggable IPAM driver, it must be specified at install time. Only with a clean install of HPE Helion OpenStack 5.0 can you specify a different IPAM driver. If upgrading, you must use the default driver. More information can be found in *HPE Helion OpenStack 5.0: Using IPAM Drivers in the Networking Service*.

Use these checklists to confirm and record your network configuration information.

Router

The IP router used with HPE Helion OpenStack must support the updated of its ARP table through gratuitous ARP packets.

PXE Installation Network

When provisioning the IP range, allocate sufficient IP addresses to cover both the current number of servers and any planned expansion. Use the following table to help calculate the requirements:

Instance	Description	IPs
Deployer O/S		1
Controller server O/S (x3)		3
Compute servers (2nd thru 100th)	single IP per server	
VSA host servers	single IP per server	

#	Item	Value
	Network is untagged	
	No DHCP servers other than Helion are on the network	
	Switch PVID used to map any "internal" VLANs to untagged	
	Routable to the IPMI network	
	IP CIDR	
	IP Range (Usable IPs)	begin: end:
	Default IP Gateway	

Management Network

The management network is the backbone used for the majority of HPE Helion OpenStack management communications. Control messages are exchanged between the Controllers, Compute hosts, and Cinder backends through this network. In addition to the control flows, the management network is also used to transport Swift and iSCSI based Cinder block storage traffic between servers.

When provisioning the IP Range, allocate sufficient IP addresses to cover both the current number of servers and any planned expansion. Use the following table to help calculate the requirements:

Instance	Description	IPs
Controller server O/S (x3)		3
Controller VIP		1
Compute servers (2nd thru 100th)	single IP per server	
VSA VM servers	single IP per server	
VSA VIP per cluster		

#	Item	Value
	Network is untagged	
	No DHCP servers other than Helion are on the network	
	Switch PVID used to map any "internal" VLANs to untagged	
	IP CIDR	
	IP Range (Usable IPs)	begin: end:
	Default IP Gateway	
	VLAN ID	

IPMI Network

The IPMI network is used to connect the IPMI interfaces on the servers that are assigned for use with implementing the cloud. For HPE ProLiant servers, the IPMI network connects to the HPE iLO management device port for the server. This network is used by Cobbler to control the state of the servers during baremetal deployments.

#	Item	Value
	Network is untagged	
	Routable to the Management Network	

#	Item	Value
	IP Subnet	
	Default IP Gateway	

External API Network

The External network is used to connect OpenStack endpoints to an external public network such as a company's intranet or the public internet in the case of a public cloud provider.

When provisioning the IP Range, allocate sufficient IP addresses to cover both the current number of servers and any planned expansion. Use the following table to help calculate the requirements.

Instance	Description	IPs
Controller server O/S (x3)		3
Controller VIP		1

#	Item	Value
	VLAN Tag assigned:	
	IP CIDR	
	IP Range (Usable IPs)	begin: end:
	Default IP Gateway	
	VLAN ID	

External VM Network

The External VM network is used to connect cloud instances to an external public network such as a company's intranet or the public internet in the case of a public cloud provider. The external network has a predefined range of Floating IPs which are assigned to individual instances to enable communications to and from the instance to the assigned corporate intranet/internet. There should be a route between the External VM and External API networks so that instances provisioned in the cloud, may access the Cloud API endpoints, using the instance floating IPs.

#	Item	Value
	VLAN Tag assigned:	
	IP CIDR	
	IP Range (Usable IPs)	begin: end:
	Default IP Gateway	
	VLAN ID	

Lifecycle Manager

This server contains the HPE Helion OpenStack installer, which is based on Git, Ansible, and Cobbler.

#	Item	Value
	Disk Requirement: Single 8GB disk needed per the *Minimum Hardware Requirements*	
	Set up the Deployer using the hLinux ISO	
	Ensure your local DNS nameserver is placed into your /etc/resolv.conf file	

#	Item	Value
	Install and configure NTP for your environment	
	Ensure your NTP server(s) is placed into your /etc/ntp.conf file	
	NTP time source:	

Information for the nic_mappings.yml Input File

Log onto each type of physical server you have and issue platform-appropriate commands to identify the bus-address and port-num values that may be required. For example, run the following command:

```
sudo lspci -D | grep -i net
```

and use this information for the bus-address value in your nic_mappings.yml file.

NIC Adapter PCI Bus Address Output

To find the port-num use:

```
cat /sys/class/net/<device name>/dev_port
```

where the 'device-name' is the name of the device **currently mapped** to this address, not necessarily the name of the device **to be mapped**.

Network Device Port Number Output

Control Plane

The Control Plane consists of three servers, in a highly available cluster, that host the core HPE Helion OpenStack services including Nova, Keystone, Glance, Cinder, Heat, Neutron, Swift, Ceilometer, and Horizon. Additional services include mysql, ip-cluster, apache2, rabbitmq, memcached, zookeeper, kafka, influxDB, storm, monasca, logging, and cmc.

> **Note:** To mitigate the "split-brain" situation described in *Network Troubleshooting* it is recommended that you have HA network configuration with Multi-Chassis Link Aggregation (MLAG) and NIC bonding configured for all the controllers to deliver system-level redundancy as well network-level resiliency. Also reducing the ARP timeout on the TOR switches will help.

Control Plane 1

#	Item	Value
	Disk Requirement: 3x 512 GB disks (or enough space to create three logical drives with that amount of space)	
	Ensure the disks are wiped	
	MAC address of first NIC	
	A second NIC, or a set of bonded NICs are required	
	iLO IP address	
	iLO Username/Password	

Control Plane 2

#	Item	Value
	Disk Requirement: 3x 512 GB disks (or enough space to create three logical drives with that amount of space)	
	Ensure the disks are wiped	
	MAC address of first NIC	
	A second NIC, or a set of bonded NICs are required	
	iLO IP address	
	iLO Username/Password	

Control Plane 3

#	Item	Value
	Disk Requirement: 3x 512 GB disks (or enough space to create three logical drives with that amount of space)	
	Ensure the disks are wiped	
	MAC address of first NIC	
	A second NIC, or a set of bonded NICs are required	
	iLO IP address	
	iLO Username/Password	

Compute Hosts

One or more KVM Compute servers will be used as the compute host targets for instances.

#	Item
	Disk Requirement: 2x 512 GB disks (or enough space to create three logical drives with that amount of space)
	A NIC for PXE boot and a second NIC, or a NIC for PXE and a set of bonded NICs are required
	Ensure the disks are wiped

Table to record your Compute host details:

ID	NIC MAC Address	iLO Username/Password	iLO IP Address	CPU/Mem/Disk

VSA Hosts

Three or more servers with local disk volumes running HPE StoreVirtual VSA to provide Cinder block storage resources.

Note: The cluster created from VSA nodes should be quorum. In otherwords, the cluster must be formed with either 3,5,7 and so on.

#	Item
	Disk Requirement: 3x 512 GB disks (or enough space to create three logical drives with that amount of space)
	- The VSA appliance deployed on a host is expected to consume ~40 GB of disk space from the host root disk for ephemeral storage to run the VSA virtual machine.
	A NIC for PXE boot and a second NIC, or a NIC for PXE ad a set of bonded NICs are required
	Ensure the disks are wiped

Table to record your VSA host details:

ID	NIC MAC Address	iLO Username/Password	iLO IP Address	CPU/Mem/Disk	VSA Data Volume

Additional Comments

This section is so you can add any additional information that you deem necessary

HPE Helion OpenStack® 5.0: SLES Pre-Installation Checks

Introduction
This document needs review for SLES content.

As part of the preparation to provision a SLES compute node you need to be aware of the issues raised in this article and take any remedial action required.

Sample iptables rules must be removed on SLES

HPE Helion OpenStack 5.0 uses iptables to secure access to lifecycle manager network interfaces and on Red Hat this requires the package `iptables-services` to be installed. The package will provide sample iptables configurations for IPv4 and IPv6 if none existed before. This sample configuration is inappropriate for HPE Helion OpenStack operation and the node will not be able to run HOS with these rules installed.

The files installed are:

- `/etc/sysconfig/iptables`
- `/etc/sysconfig/ip6tables`

If these files **do not exist** on the candidate Red Hat systems the rest of this note may be skipped as the HPE Helion OpenStack 5.0 install will prevent the installation of the sample files. However, if these files do exist, there are a number of steps that you must follow before you install HPE Helion OpenStack 5.0.

The contents of these two files are displayed here for reference:

/etc/sysconfig/iptables

```
# sample configuration for iptables service
# you can edit this manually or use system-config-firewall
# please do not ask us to add additional ports/services to this default configuration
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

/etc/sysconfig/ip6tables

```
# sample configuration for ip6tables service
# you can edit this manually or use system-config-firewall
# please do not ask us to add additional ports/services to this default configuration
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p ipv6-icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -d fe80::/64 -p udp -m udp --dport 546 -m state --state NEW -j ACCEPT
-A INPUT -j REJECT --reject-with icmp6-adm-prohibited
-A FORWARD -j REJECT --reject-with icmp6-adm-prohibited
COMMIT
```

These rules are applied to network traffic on *any interface* on the system whereas HPE Helion OpenStack components and OpenStack components manage *interface specific* rules. With reference to the security policies that you may require for your environment, it is essential that you either:

- Delete the files /etc/sysconfig/iptables and /etc/sysconfig/ip6tables as none of the contained rules are required.
- Ensure that any remaining rules are limited to interfaces not used by HPE Helion OpenStack.

⚠ **Important:** If these files existed on your system, and did contain content, the corresponding rules are currently installed and active on the system. Once you delete these two files (or edit them to limit the rules to interfaces not used by HPE Helion OpenStack), you will need to reboot the system to activate the new settings.

HPE Helion OpenStack® 5.0: RHEL Pre-Installation Checks

Introduction

As part of the preparation to provision a RedHat compute node you need to be aware of the issues raised in this article and take any remedial action required.

Sample iptables rules must be removed on RedHat

HPE Helion OpenStack 5.0 uses iptables to secure access to lifecycle manager network interfaces and on Red Hat this requires the package `iptables-services` to be installed. The package will provide sample iptables configurations for IPv4 and IPv6 if none existed before. This sample configuration is inappropriate for HPE Helion OpenStack operation and the node will not be able to run HOS with these rules installed.

The files installed are:

- `/etc/sysconfig/iptables`
- `/etc/sysconfig/ip6tables`

If these files **do not exist** on the candidate Red Hat systems the rest of this note may be skipped as the HPE Helion OpenStack 5.0 install will prevent the installation of the sample files. However, if these files do exist, there are a number of steps that you must follow before you install HPE Helion OpenStack 5.0.

The contents of these two files are displayed here for reference:

/etc/sysconfig/iptables

```
# sample configuration for iptables service
# you can edit this manually or use system-config-firewall
# please do not ask us to add additional ports/services to this default configuration
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

/etc/sysconfig/ip6tables

```
# sample configuration for ip6tables service
# you can edit this manually or use system-config-firewall
# please do not ask us to add additional ports/services to this default configuration
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p ipv6-icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -d fe80::/64 -p udp -m udp --dport 546 -m state --state NEW -j ACCEPT
-A INPUT -j REJECT --reject-with icmp6-adm-prohibited
-A FORWARD -j REJECT --reject-with icmp6-adm-prohibited
COMMIT
```

These rules are applied to network traffic on *any interface* on the system whereas HPE Helion OpenStack components and OpenStack components manage *interface specific* rules. With reference to the security policies that you may require for your environment, it is essential that you either:

- Delete the files `/etc/sysconfig/iptables` and `/etc/sysconfig/ip6tables` as none of the contained rules are required.
- Ensure that any remaining rules are limited to interfaces not used by HPE Helion OpenStack.

⚠ **Important:** If these files existed on your system, and did contain content, the corresponding rules are currently installed and active on the system. Once you delete these two files (or edit them to limit the rules to interfaces not used by HPE Helion OpenStack), you will need to reboot the system to activate the new settings.

HPE Helion OpenStack® 5.0: Using Git for Configuration Management

In HPE Helion OpenStack 5.0, a local git repository is used to track configuration changes and the configuration processor (CP) uses this repository. The introduction of a git workflow also means that your configuration history is maintained, making rollbacks easier as well as keeping a record of previous configuration settings.

The git repository is installed by the lifecycle manager on the lifecycle manager node.

The git repository provides a way for you to merge changes that you pull down as "upstream" (i.e. from HP) updates, and allows you to manage your own configuration changes.

Initialization on a new deployment

On a system new to HPE Helion OpenStack 5.0, the lifecycle manager will prepare a git repository under `~/helion`. The lifecycle manager provisioning runs the `hlm-init-deployer` script automatically - this calls `ansible-playbook -i hosts/localhost git-00-initialise.yml`.

As a result, the `~/helion` directory is initialized as a git repo (if it's empty). It is initialized with four empty branches:

hos	This holds the upstream source code corresponding to the contents of the `~/helion` directory on a pristine fresh installation. Every source code release that is downloaded from HPE is applied as a fresh commit to this branch. This branch contains no customization by the end user.
site	This branch begins life as a copy of the first 'hos' drop. It is onto this branch that you commit your configuration changes. It's the branch most visible to the end user.
ansible	This branch contains the variable definitions generated by the CP that our main ansible playbooks need. This includes the `verb_hosts` file that describes to ansible what servers are playing what roles. The `ready-deployment` playbook takes this output and assembles a `~/scratch` directory containing the ansible playbooks together with the variable definitions in this branch. The result is a working ansible directory `~/scratch/ansible/next/hos/ansible` from which the main deployment playbooks may be successfully run.
cp-persistent	This branch contains the persistent state that the CP needs to maintain. That state is mostly the assignment of IP addresses and roles to particular servers. Some operational procedures may involve editing the contents of this branch: for example, retiring a machine from service or repurposing it.

Two temporary branches are created and populated at run time:

staging-ansible	This branch hosts the most recent commit that will be appended to the ansible branch.
staging-cp-persistent	This branch hosts the most recent commit that will be appended to the cp-persistent branch.

Note: The information above provides insight into the workings of the configuration processor and the git repository. However, in practice you can simply follow the steps below to make configuration changes.

Updating any configuration, including the default configuration

When you need to make updates to a configuration you must

1. Check out the **site** branch. You may already be on that branch. If so, git will tell you that and the command will leave you there.

```
git checkout site
```

2. Edit the YAML file or files that contain the configuration you want to change.
3. Commit the changes to the **site** branch.

```
git add -A
git commit -m "your commit message goes here in quotes"
```

If you want to add a single file to your git repository, you can use the command below, as opposed to using `git add -A`:

```
git add PATH_TO_FILE
```

For example, if you made a change to your `servers.yml` file and wanted to only commit that change, you would use this command:

```
git add ~/helion/my_cloud/definition/data/servers.yml
```

4. Run the configuration processor:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

to produce the required configuration processor output from those changes. Review the output files manually if required.

5. Ready the deployment area

```
ansible-playbook -i hosts/localhost ready-deployment.yml
```

6. Run the deployment playbooks from the resulting scratch directory.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts site.yml
```

HPE Helion OpenStack® 5.0: Boot from SAN and Multipath Configuration

Introduction

HPE Helion OpenStack 5.0 supports Fibre Channel and FCoE boot from SAN in multipath environments. The following list outlines the current limitations based on testing:

- **Emulex based LPE1605 Native Fibre Channel** - Up to 1024 paths during boot
- **Qlogic based SN100Q Native Fibre Channel** - Up to 1024 paths during boot
- **Emulex Flex Fabric 650 series** - Up to 1024 paths during boot
- **Emulex Flex Fabric 554FLB** - Up to 1024 paths during boot
- **Qlogic Flex Fabric 536 and 630 series** - Up to 1024 paths during boot

⚠ **Important:** When exporting a LUN to a node for boot from san, you should ensure that "LUN 0" is assigned to the LUN and configure any setup dialog that is necessary in the firmware to consume this LUN0 for OS boot.

⚠ **Important:** Any hosts that are connected to 3PAR storage must have a `host persona` of `2-generic-alua` set on the 3PAR. Refer to the 3PAR documentation for the steps necessary to check this and change if necessary.

iSCSI boot from SAN is not supported. For more information on the use of Cinder with multipath, see *3PAR Configuration - Multipath*.

In order to allow HPE Helion OpenStack 5.0 use volumes from a SAN, a number of configuration options need to be specified for both the install phases and for the osconfig phases. In all cases the devices that are utilised are devices for which multipath is configured.

Install Phase Configuration

For FC connected nodes and for FCoE nodes where the network processor used is from the Emulex family such as for the 650FLB, the following changes need to be made.

1. In each stanza of the servers.yml insert a line stating **boot-from-san: true**

```
  - id: controller2
    ip-addr: 192.168.10.4
    role: CONTROLLER-ROLE
    server-group: RACK2
    nic-mapping: HP-DL360-4PORT
    mac-addr: "8a:8e:64:55:43:76"
    ilo-ip: 192.168.9.4
    ilo-password: password
    ilo-user: admin
    boot-from-san: true
```

This uses the disk `/dev/mapper/mpatha` as the default device on which to install the OS.

2. In the disk input models you also need to specify the devices that will be used via their multipath names (which will be of the form `/dev/mapper/mpatha`, `/dev/mapper/mpathb` etc).

```
    volume-groups:
      - name: hlm-vg
        physical-volumes:

          # NOTE: 'sda_root' is a templated value. This value is checked in
          # os-config and replaced by the partition actually used on sda
          #e.g. sda1 or sda5
          - /dev/mapper/mpatha_root
```

```
...
      - name: vg-comp
        physical-volumes:
          - /dev/mapper/mpathb
```

QLogic FCoE restrictions and additional configurations

Applies to HPE Helion OpenStack® 3.0.1.

If you are using network cards such as Qlogic Flex Fabric 536 and 630 series then there are additional OS configuration steps to support the importation of LUNs as well as some restrictions on supported configurations.

The restrictions are:

1. Only one network card can be enabled in the system.
2. The FCoE interfaces on this card are dedicated to FCoE traffic - these cannot have ip addresses associated with them.
3. Nic mapping cannot be used.

In addition to the configuration options above, you also need to specify the FCoE interfaces for install and for os configuration. There are 3 places where you need to add additional configuration options for fcoe-support:

- In `servers.yml`, which is used for configuration of the system during OS install, FCoE interfaces need to be specified for each server. In particular, the mac addresses of the FCoE interfaces need to be given, **not** the symbolic name (e.g. eth2).

```
      - id: compute1
        ip-addr: 10.245.224.201
        role: COMPUTE-ROLE
        server-group: RACK2
        mac-addr: 6c:c2:17:33:4c:a0
        ilo-ip: 10.1.66.26
        ilo-user: hlinux
        ilo-password: hlinux123
        boot-from-san: True
        fcoe-interfaces:
          - 6c:c2:17:33:4c:a1
          - 6c:c2:17:33:4c:a9
```

⚠ **Important:** Nic mapping can not be used.

- For the osconfig phase, you will need to specify the `fcoe-interfaces` as a peer of `network-interfaces` in the `net_interfaces.yml` file:

```
      - name: CONTROLLER-INTERFACES
        fcoe-interfaces:
          - name: fcoe
            devices:
              - eth2
              - eth3
        network-interfaces:
          - name: eth0
            device:
                name: eth0
            network-groups:
              - EXTERNAL-API
              - EXTERNAL-VM
              - GUEST
              - MANAGEMENT
```

⚠ **Important:**

The mac addresses specified in the `fcoe-interfaces` stanza in `servers.yml` must correspond to the symbolic names used in the `fcoe-interfaces` stanza in `net_interfaces.yml`.

Also, to satisfy the restriction outlined in *2* above, there can be no overlap between the devices in `fcoe-interfaces` and those in `network-interfaces` in the `net_interfaces.yml` file. In the example, `eth2` and `eth3` are `fcoe-interfaces` while `eth0` is in `network-interfaces`.

- As part of the initial install from an iso, additional parameters need to be supplied on the kernel command line:

```
multipath=true partman-fcoe/interfaces=<mac address1>,<mac address2> disk-detect/fcoe/enable=true --- quiet
```

See the sections of installing from an iso using *UEFI* and *Legacy Bios*.

Since nic mapping is not used to guarantee order of the networks across the system the installer will remap the network interfaces in a deterministic fashion as part of the install. As part of the installer dialogue, if DHCP is not configured for the interface, it is necessary to confirm that the appropriate interface is assigned the ip address. The network interfaces may not be at the names expected when installing via an iso. When you are asked to apply an ip address to an interface, enter `Alt-F2` and in the console window issue an `ip a` command to examine the interfaces and their associated mac addresses. Take a note of the interface name with the expected mac address and use this in the subsequent dialogue. Enter `Alt-F1` to return to the installation screen. You should note that the names of the interfaces may have changed after the installation completes. These names are used consistently in any subsequent operations.

Therefore, even if FCoE is not used for boot from SAN (e.g. for cinder), then it is recommended that `fcoe-interfaces` be specified as part of install (without the multipath or disk detect options). Alternatively, you need to run `osconfig-fcoe-reorder.yml` before `site.yml` or `osconfig-run.yml` is invoked to reorder the networks in a similar manner to the installer. In this case, the nodes will need to be manually rebooted for the network reorder to take effect. You will run `osconfig-fcoe-reorder.yml` in the following scenarios:

- If you have used a third-party installer to provision your baremetal nodes
- If you are booting from a local disk (i.e. one that is not presented from the SAN) but you want to use FCoE later, for example, for cinder.

To run the command:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts osconfig-fcoe-reorder.yml
```

If you do not run `osconfig-fcoe-reorder.yml`, you will encounter a failure in `osconfig-run.yml`.

If you are booting from a local disk, the LUNs that will be imported over FCoE will not be visible before `site.yml` or `osconfig-run.yml` has been run. However, if you need to import the LUNs before this, for instance, in scenarios where you need to run `wipe_disks.yml`, then you can run the `fcoe-enable` playbook across the nodes in question. This will configure FCoE and import the LUNs presented to the nodes.

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/verb_hosts fcoe-enable.yml
```

Red Hat Compute Host for FCoE

When installing a Red Hat compute host, the names of the network interfaces will have names like `ens1f2` rather than `eth2` therefore a separate role and associated `network-interfaces` and `fcoe-interfaces` descriptions will have to be provided in the input model for the Red Hat compute hosts. Here are some excerpts highlighting the changes required:

net_interfaces.yml

```
    - name: RHEL-COMPUTE-INTERFACES
      fcoe-interfaces:
        - name: fcoe
          devices:
            - ens1f2
            - ens1f3
      network-interfaces:
        - name: ens1f0
          device:
            name: ens1f0
          network-groups:
            - EXTERNAL-VM
            - GUEST
            - MANAGEMENT
```

control_plane.yml

```
      - name: rhel-compute
        resource-prefix: rhcomp
        server-role: RHEL-COMPUTE-ROLE
        allocation-policy: any
        min-count: 0
        service-components:
          - ntp-client
          - nova-compute
          - nova-compute-kvm
          - neutron-l3-agent
          - neutron-metadata-agent
          - neutron-openvswitch-agent
          - neutron-lbaasv2-agent
```

server_roles.yml

```
      - name: RHEL-COMPUTE-ROLE
      interface-model: RHEL-COMPUTE-INTERFACES
      disk-model: COMPUTE-DISKS
```

servers.yml

```
   - id: QlogicFCoE-Cmp2
     ip-addr: 10.245.224.204
     role: RHEL-COMPUTE-ROLE
     server-group: RACK2
     mac-addr: "6c:c2:17:33:4c:a0"
     ilo-ip: 10.1.66.26
     ilo-password: hlinux123
     ilo-user: hlinux
     boot-from-san: True
     distro-id: rhel72-x86_64-multipathfcoe-interfaces:
        - 6c:c2:17:33:4c:a1
        - 6c:c2:17:33:4c:a9
```

⚠ **Important:** Note that you need to add a -multipath suffix to the distro-id value when using multipath with RHEL.

If you are installing a Red Hat compute node with QLogic FCoE boot from SAN, you will need to edit the /var/lib/cobbler/kickstarts/rhel72-anaconda-ks-multipath.cfg file. and uncomment the **two** sections that are prefaced by:

```
#Qlogic-FCOE: Uncomment the below lines if using qlogic fcoe boot from san
```

The values will be overwritten on running the cobbler-deploy play.

📄 **Note:** It is highly recommended as part of a Red Hat install that only one disk is presented to the node during the install phase. **You should ensure that any additional LUNs that are required are attached and visible on the compute hosts before running the site.yml playbook.**

Installing the HPE Helion OpenStack 5.0 iso for nodes that support Boot from SAN

This sections describes how to install the iso on the Lifecycle Manager to support the following configurations:

UEFI Boot Mode
UEFI Boot Mode with QLogic FCoE
Legacy BIOS Boot Mode
Legacy BIOS Boot Mode with QLogic FCoE

The lifecycle manager will then automatically handle the installation on nodes that supports Boot from SAN based on the configuration information in `servers.yml` and the disk models, as described in the preceding section.

UEFI Boot Mode

On the installer boot menu, type a lower-case "e" to enter the `Edit Selection` screen.
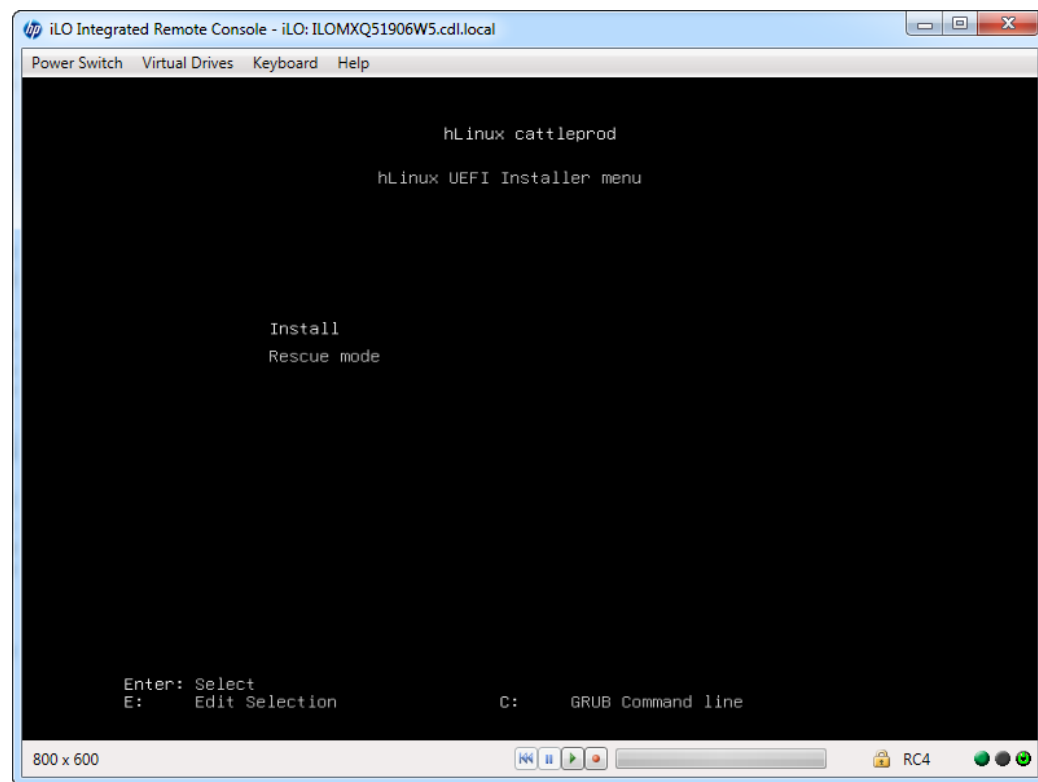
This brings up the `Edit Selection` screen.
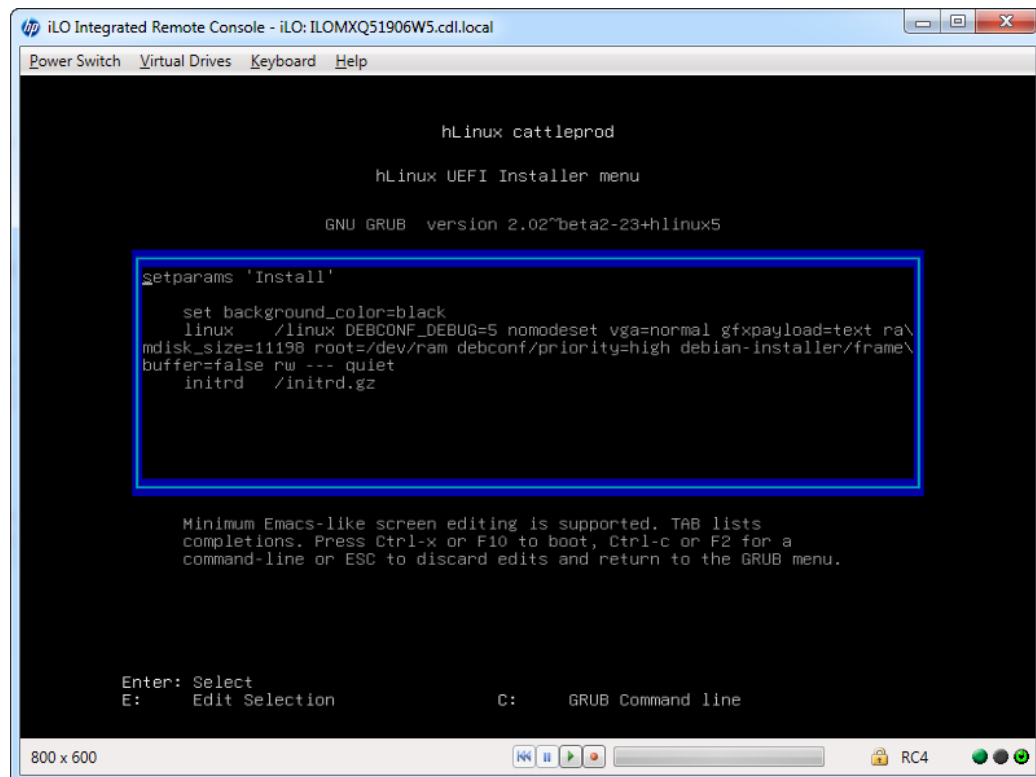
Enter the text `multipath=true` before `--- quiet`:
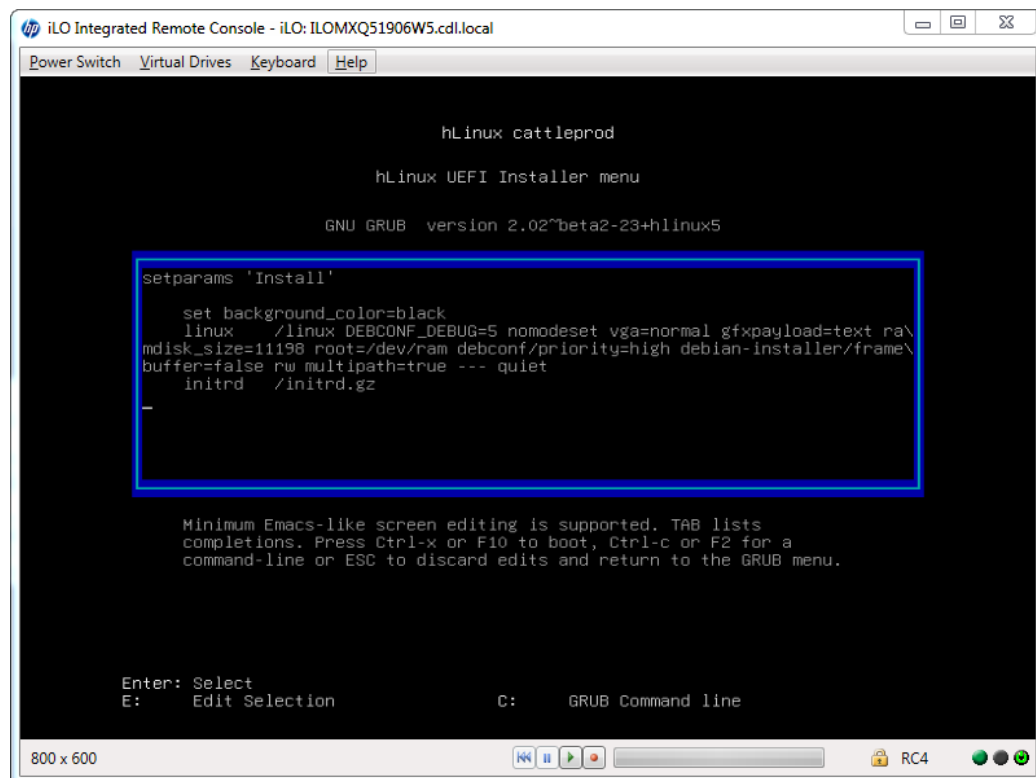
Press `Ctrl-X` or `F10` to proceed with the install.

UEFI Boot Mode with QLogic FCoE

Applies to HPE Helion OpenStack® 3.0.1.

At the installer boot menu, type a lower-case "e" to enter the `Edit Selection` screen as described in the preceding section. In addition to inserting `multipath=true`, it is necessary to supply details of the FCoE network interfaces. In the example below, the interfaces are specified as:

```
partman-fcoe/interfaces=6c:c2:17:33:4c:a1,6c:c2:17:33:4c:a9  disk-detect/fcoe/enable=true
```
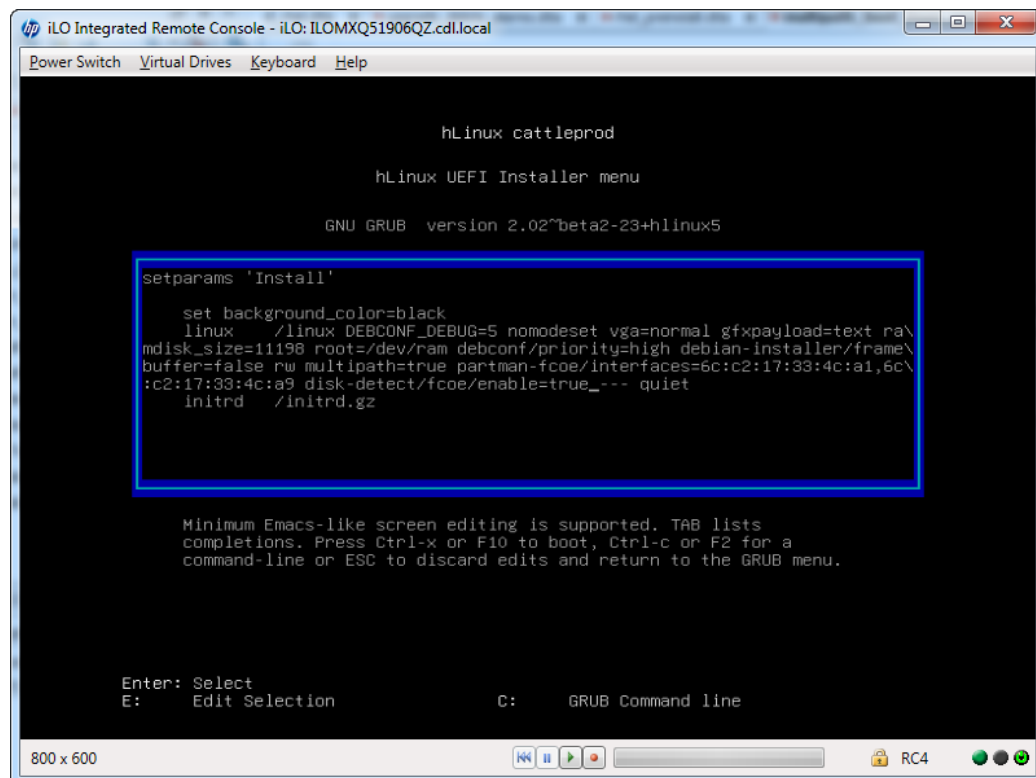
Press `Ctrl-X` or `F10` to proceed with the install.

Legacy BIOS Boot Mode

On the installer boot menu, navigate (using the Down Arrow) to the `Advanced options` entry and then press Enter
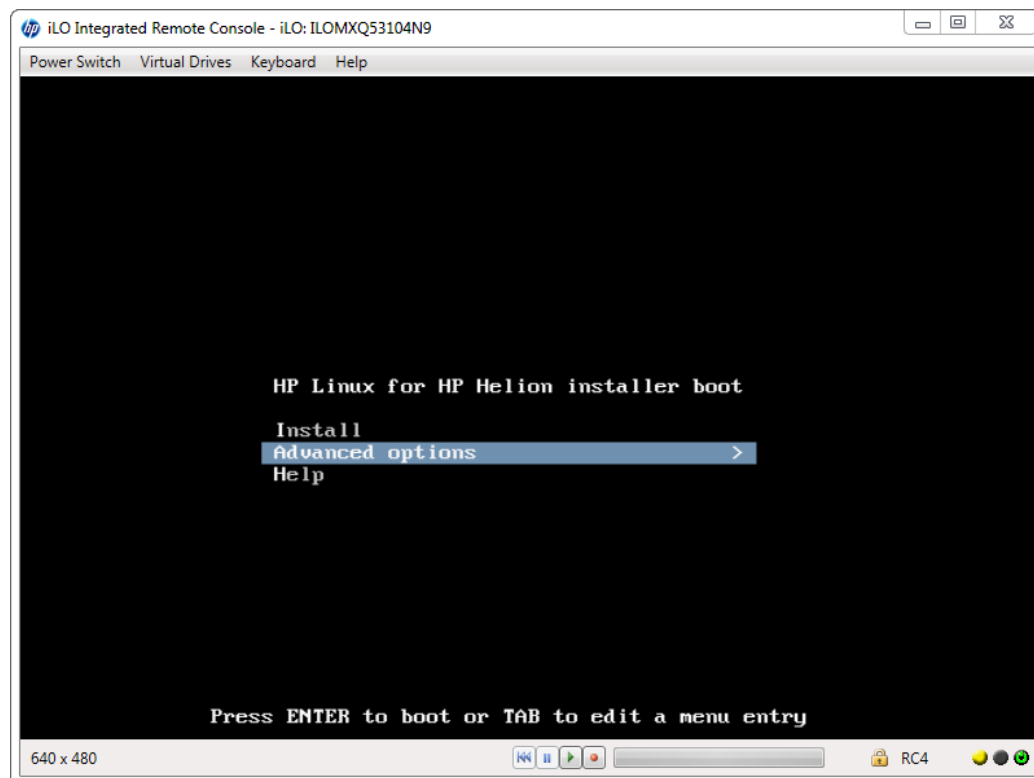
This will bring up the Advanced options menu:
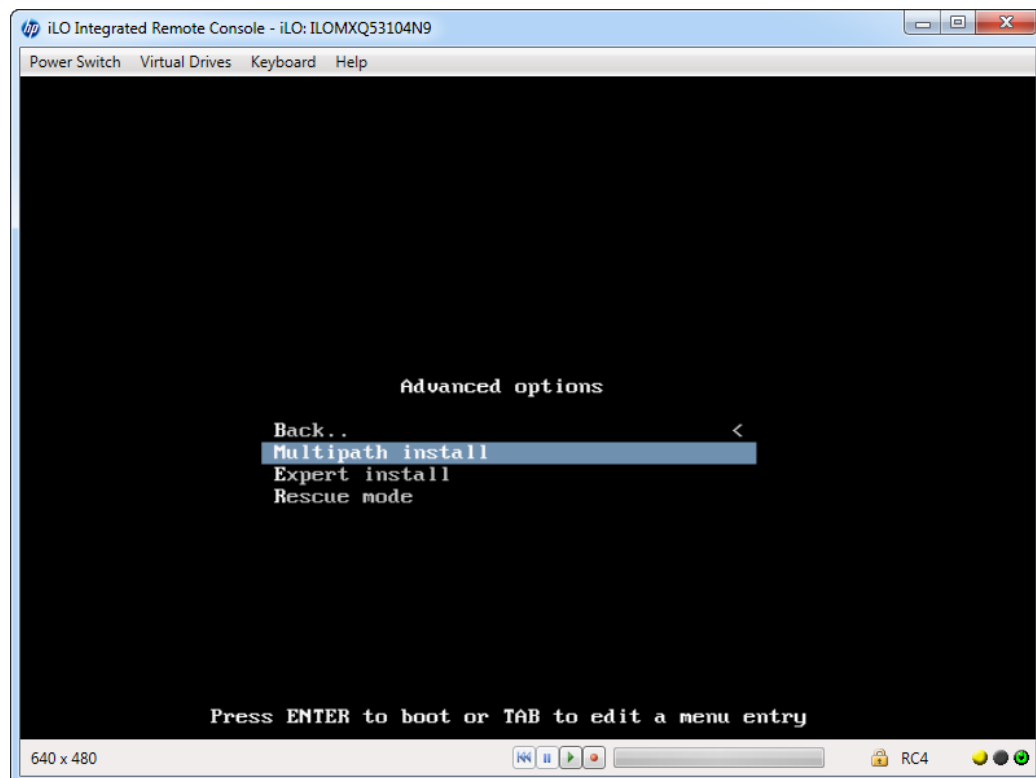
Navigate to the `Multipath install` entry and press Enter to start the installation.

Legacy BIOS Boot Mode with QLogic FCoE

Applies to HPE Helion OpenStack® 3.0.1.

On the installer boot menu, navigate (using the Down Arrow) to the `Advanced options` entry and then press Enter
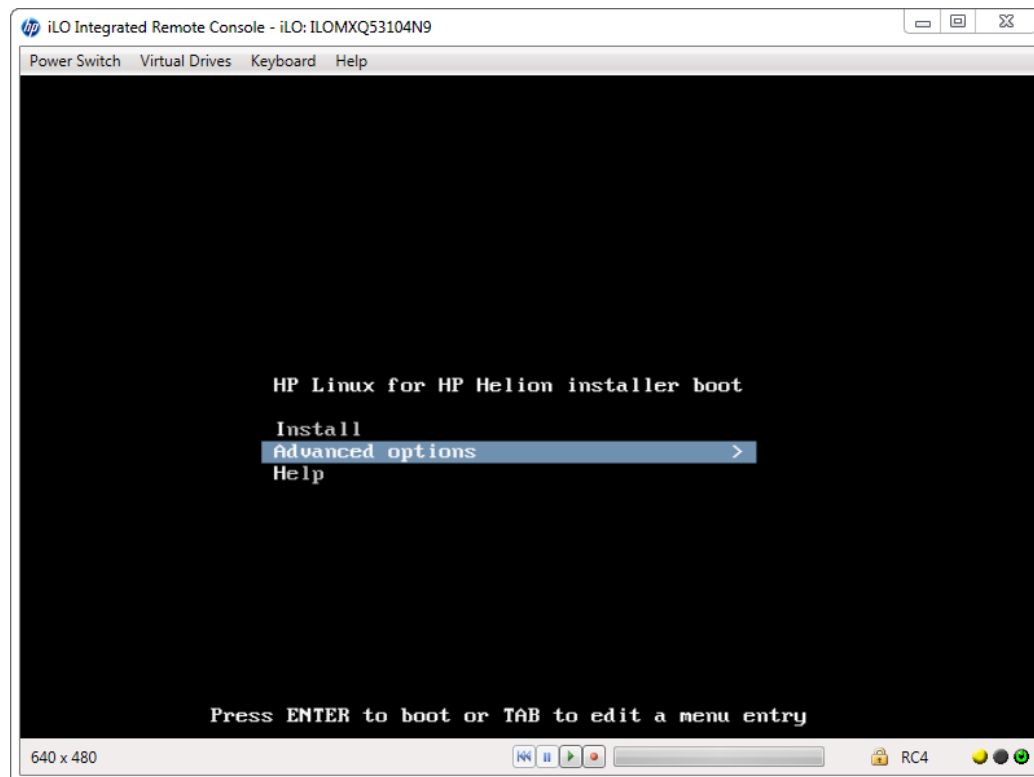
This will bring up the Advanced options menu:
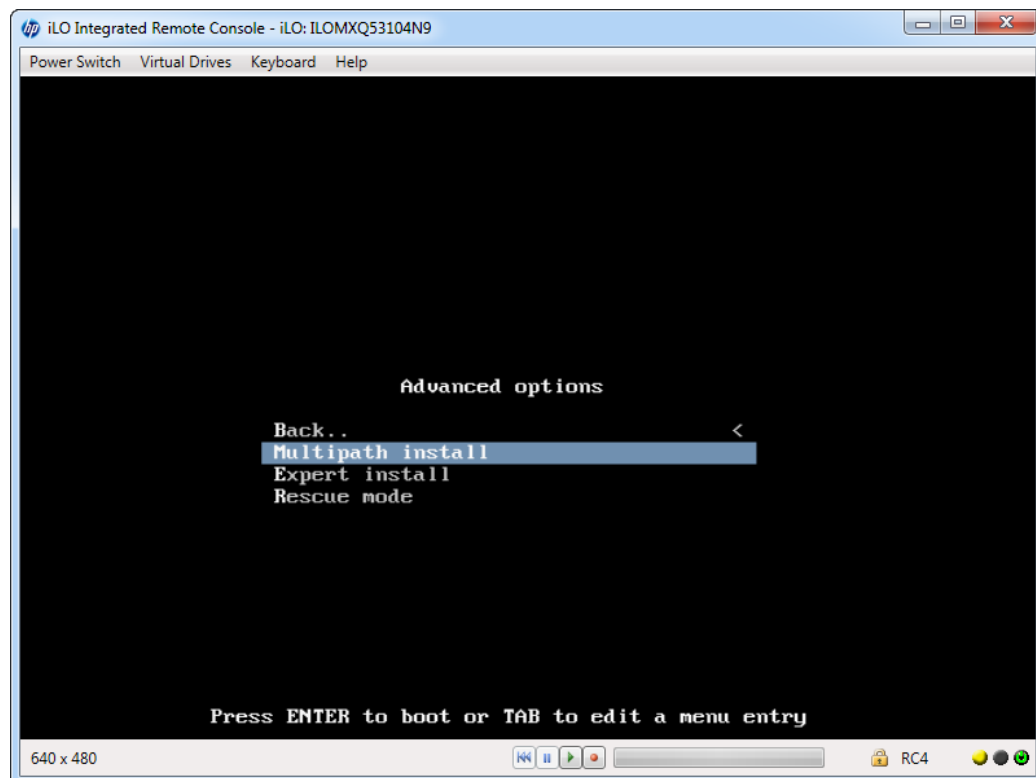
Navigate to the `Multipath install` entry and press TAB to edit the entry details. Notice that the kernel command line is now displayed at the bottom of the screen and that `multipath=true` is already specified.
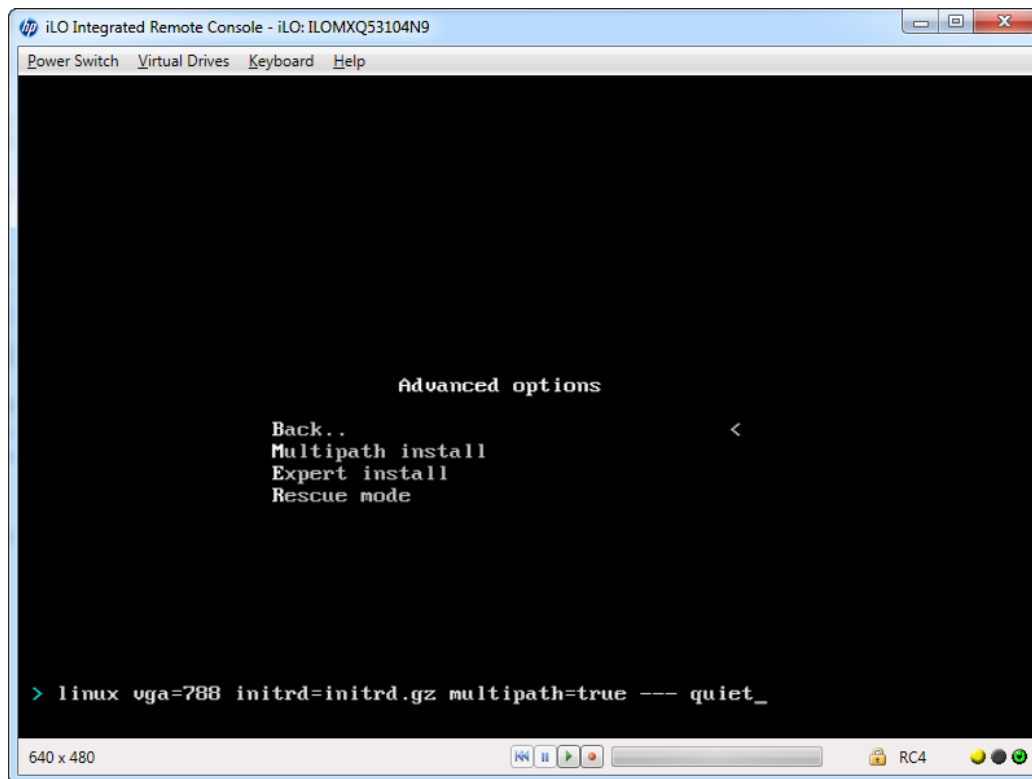
Edit the kernel command line to add the FCoE network interfaces before the - - -. In the example below, the interfaces are specified as:

```
partman-fcoe/interfaces=6c:c2:17:33:4c:a1,6c:c2:17:33:4c:a9  disk-detect/fcoe/enable=true
```
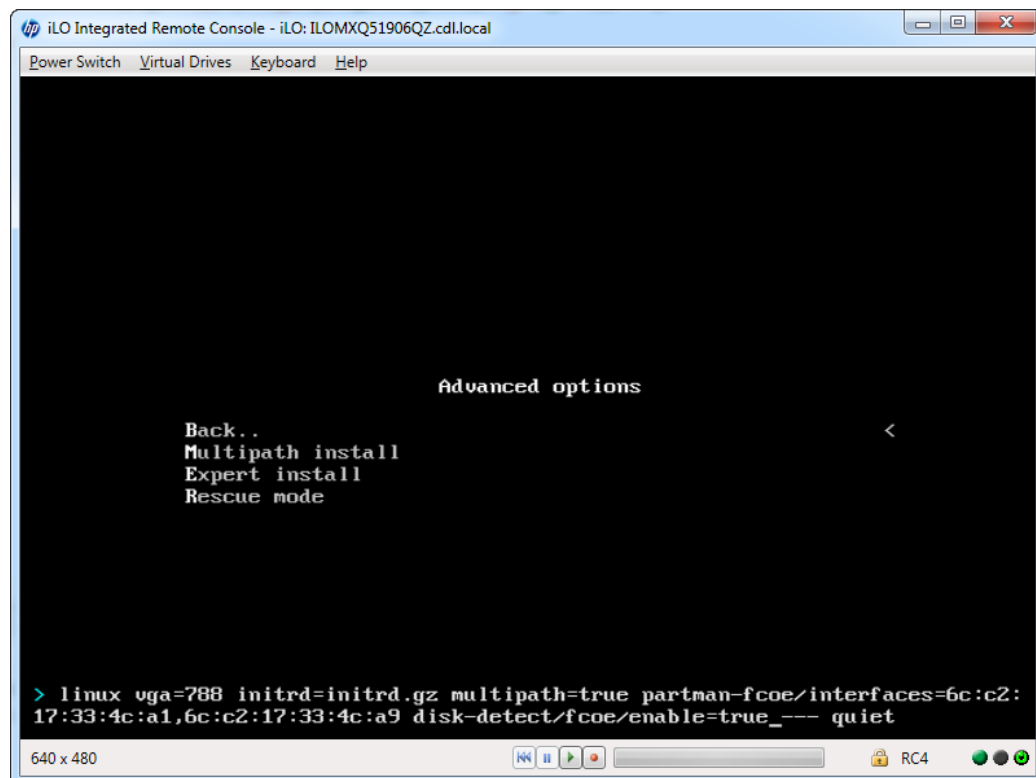
Now press Enter to start the installation.

HPE Helion OpenStack® 5.0: Installing the L2 Gateway Agent for the Networking Service

Introduction

The L2 gateway is a service plug-in to the Neutron networking service that allows two L2 networks to be seamlessly connected to create a single L2 broadcast domain. The initial implementation provides for the ability to connect a virtual Neutron VxLAN network to a physical VLAN using a VTEP-capable HPE 5930 switch. The L2 gateway is to be enabled only for VxLAN deployments.

To begin L2 gateway agent setup, you need to configure your switch. These instructions use an *HPE FlexFabric 5930 Switch Series* switch.

- Use case
- *Sample network topology*
- *Networks*
- *HPE 5930 Switch Configuration*
- *Enabling and configuring L2 gateway agent*
- *Routing between software and hardware VTEP networks*
- *Connecting baremetal server to HPE 5930 switch*

- *Configuration on baremetal server to receive tagged packets from the switch*
- *NIC bonding and IRF configuration*
- *Scale numbers tested*
- *L2 gateway commands*

Sample network topology (for illustration purposes)

When viewing the following network diagram, assume that the blue VNET has been created by the tenant and has been assigned a segmentation ID of 1000 (VNI 1000). The Cloud Admin is now connecting physical servers to this VNET.

Assume also that the blue L2 Gateway is created and points to the HW VTEPS, the physical ports, the VLAN, and if it is an access or trunk port (tagged or untagged)

Sample Network Topology

L2 GW Logical Interface

L2 GW Logical Interface

L2 GW Logical Interface

Physical Ports

L2 GW Device
(VTEP IP, UUD)

5930 HW VTEP

VSI
(VNI 1000)

HW VTEP IP

**Blue L2
Gateway**

VLAN, Pport,
Untagged

VLAN, Pport,
Tagged

VLAN, Pport,
Untagged

| P0 | P1 | P2 | P3 | P4 | ... | Pn |

**Blue VNET
(Assigned VNI 1000)**

Accecss
Untagged

Trunk
Tagged 20

Accecss
Untagged

Desktop

Switch

Desktop

VTEP IP

VTEP IP

Compute Node

Compute Node

VM

VM

Desktop

Desktop

Desktop

Figure 1

Networks

The following diagram illustrates an example network configuration.

Setting up L2 gateway with HPE 5930 switch

Management Network Switch

Interface: M-GE0/0/0
10.10.85.10

HPE 5930 Switch

Data Network
VLAN 103   VLAN 83
10.1.1.10   10.10.83.3

Bare Metal Server (Ubuntu 14.04)

eth0
10.10.85.129

eth1.183

10.10.10.129

10.10.85.130
Controller 1 eth0 eth1
10.1.1.1

10.10.85.131
Controller 2 eth0 eth1
10.1.1.2

10.10.85.132
Controller 3 eth0 eth1
10.1.1.3

1/0/6:2 1/0/6:1 1/0/5:4 1/0/5:2 1/0/5:1 1/0/5:3

VLAN 103 (untagged) ×4
VLAN 83 (tagged)
VTEP access port
VLAN 183 (untagged)

10.1.1.4
Compute Node eth0
OVS
10.10.85.133

VTEP IP Addresses

Controllers: 10.10.87.1, 10.10.87.2, 10.10.87.3
Compute Node: 10.10.87.4

rise Development Company, L.P.

Figure 2

An L2 gateway is useful in extending virtual networks (VxLAN) in a cloud onto physical VLAN networks. The L2 gateway switch converts VxLAN packets into VLAN packets and back again, as shown in the following diagram. This topic assumes a VxLAN deployment.
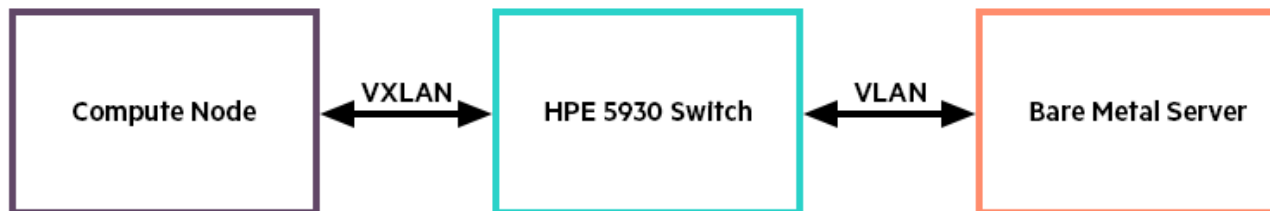
Figure 3

- Management Network: 10.10.85.0/24
- Data Network: 10.1.1.0/24
- Tenant VM Network: 10.10.10.0/24

Note: These IP ranges are used in the topology shown in the diagram for illustration only.

HPE 5930 switch configuration

1. Telnet to the 5930 switch and provide your username and password.
2. Go into system view:

```
system-view
```

3. Create the required VLANs and VLAN ranges:

```
vlan 103
vlan 83
vlan 183
vlan 1261 to 1270
```

4. Assign an IP address to VLAN 103. This is used as a data path network for VxLAN traffic.

```
interface vlan 103
ip address 10.1.1.10 255.255.255.0
```

5. Assign an IP address to VLAN 83. This is used as a hardware VTEP network.

```
interface vlan 83
ip address 10.10.83.3 255.255.255.0
```

The 5930 switch has a fortygigE1/0/5 interface to which a splitter cable is connected that splits the network it into four tengigEthernet (tengigEthernet1/0/5:1 to tengigEthernet1/0/5:4) interfaces:

- engigEthernet1/0/5:1 and tengigEthernet1/0/5:2 are connected to the compute node. This is required just to bring the interface up. In other words, in order to have the HPE 5930 switch work as a router, there should be at least one interface of that particular VLAN up. Alternatively, the interface can be connected to any host or network element.
- tengigEthernet1/0/5:3 is connected to a baremetal server.

- tengigEthernet1/0/5:4 is connected to controller 3, as shown in *Figure 2*.

The switch's fortygigE1/0/6 interface to which the splitter cable is connected splits it into four tengigEthernet (tengigEthernet1/0/6:1 to tengigEthernet1/0/6:4) interfaces:

- tengigEthernet1/0/6:1 is connected to controller 2
- tengigEthernet1/0/6:2 is connected to controller 1

Note: 6:3 and 6:4 are not used although they are available.

6. Split the fortygigE 1/0/5 interface into tengig interfaces:

```
interface fortygigE 1/0/5
using tengig
The interface FortyGigE1/0/5 will be deleted. Continue? [Y/N]: y
```

7. Configure the Ten-GigabitEthernet1/0/5:1 interface:

```
interface Ten-GigabitEthernet1/0/5:1
port link-type trunk
port trunk permit vlan 83
```

Configuring the provider data path network

1. Configure the Ten-GigabitEthernet1/0/5:2 interface:

```
interface Ten-GigabitEthernet1/0/5:2
port link-type trunk
port trunk permit vlan 103
port trunk permit vlan 1261 to 1270
port trunk pvid vlan 103
```

2. Configure the Ten-GigabitEthernet1/0/5:4 interface:

```
interface Ten-GigabitEthernet1/0/5:4
port link-type trunk
port trunk permit vlan 103
port trunk permit vlan 1261 to 1270
port trunk pvid vlan 103
```

3. Configure the Ten-GigabitEthernet1/0/5:3 interface:

```
interface Ten-GigabitEthernet1/0/5:3
port link-type trunk
port trunk permit vlan 183
vtep access port
```

4. Split the fortygigE 1/0/6 interface into tengig interfaces:

```
interface fortygigE 1/0/6
using tengig
The interface FortyGigE1/0/6 will be deleted. Continue? [Y/N]: y
```

5. Configure the Ten-GigabitEthernet1/0/6:1 interface:

```
interface Ten-GigabitEthernet1/0/6:1
port link-type trunk
port trunk permit vlan 103
port trunk permit vlan 1261 to 1270
port trunk pvid vlan 103
```

6. Configure the Ten-GigabitEthernet1/0/6:2 interface:

```
interface Ten-GigabitEthernet1/0/6:2
port link-type trunk
port trunk permit vlan 103
port trunk permit vlan 1261 to 1270
port trunk pvid vlan 103
```

7. Enable l2vpn:

```
l2vpn enable
```

8. Configure a passive TCP connection for OVSDB on port 6632:

```
ovsdb server ptcp port 6632
```

9. Enable OVSDB server:

```
ovsdb server enable
```

10. Enable a VTEP process:

```
vtep enable
```

11. Configure 10.10.83.3 as the VTEP source IP. This acts as a hardware VTEP IP.

```
tunnel global source-address 10.10.83.3
```

12. Configure the VTEP access port:

```
interface Ten-GigabitEthernet1/0/5:3
vtep access port
```

13. Disable VxLAN tunnel mac-learning:

```
vxlan tunnel mac-learning disable
```

14. Display the current configuration of the 5930 switch and verify the configuration:

```
display current-configuration
```

After switch configuration is complete, you can dump OVSDB to see the entries.

1. Run the ovsdb-client from any Linux machine reachable by the switch:

```
ovsdb-client dump --pretty tcp:10.10.85.10:6632

sdn@small-hLinux:~$ ovsdb-client dump --pretty tcp:10.10.85.10:6632
Arp_Sources_Local table
_uuid locator src_mac
----- ------- -------

Arp_Sources_Remote table
_uuid locator src_mac
----- ------- -------

Global table
_uuid                                managers switches
------------------------------------ -------- -------------------------------------
2c891edc-439b-4144-84d9-fa4cd88092bf []       [f5f4b43b-40bc-4640-b580-d4b12011f688]
```

```
Logical_Binding_Stats table
_uuid bytes_from_local bytes_to_local packets_from_local packets_to_local
----- ---------------- -------------- ----------------- ----------------

Logical_Router table
_uuid description name static_routes switch_binding
----- ----------- ---- ------------- --------------

Logical_Switch table
_uuid description name tunnel_key
----- ----------- ---- ----------

Manager table
_uuid inactivity_probe is_connected max_backoff other_config status target
----- ---------------- ------------ ----------- ------------ ------ ------

Mcast_Macs_Local table
MAC _uuid ipaddr locator_set logical_switch
--- ----- ------ ----------- --------------

Mcast_Macs_Remote table
MAC _uuid ipaddr locator_set logical_switch
--- ----- ------ ----------- --------------

Physical_Locator table
_uuid dst_ip encapsulation_type
----- ------ ------------------

Physical_Locator_Set table
_uuid locators
----- --------

Physical_Port table
_uuid                                description name                           port_fault_status vlan_bindings vlan_stats
------------------------------------ ----------- ------------------------------ ----------------- ------------- ----------
fda90870-656c-479b-91ee-852b70e2007e ""          "Ten-GigabitEthernet1/0/5:3" [UP]                {}            {}

Physical_Switch table
_uuid                                description management_ips name       ports                                           switch_fault_status tunnel_ips     tunnels
------------------------------------ ----------- -------------- ---------- ----------------------------------------------- ------------------- -------------- -------
f5f4b43b-40bc-4640-b580-d4b12011f688 ""          []             "L2GTWY02" [fda90870-656c-479b-91ee-852b70e2007e] []                           ["10.10.83.3"] []

Tunnel table
_uuid bfd_config_local bfd_config_remote bfd_params bfd_status local remote
----- ---------------- ----------------- ---------- ---------- ----- ------

Ucast_Macs_Local table
MAC _uuid ipaddr locator logical_switch
--- ----- ------ ------- --------------

Ucast_Macs_Remote table
MAC _uuid ipaddr locator logical_switch
--- ----- ------ ------- --------------
```

Enabling and configuring the L2 gateway agent

1. Update the input model (in `control_plane.yml`) to specify where you want to run the neutron-l2gateway-agent. For example, see the line in bold in the following yml:

```
---
 product:
  version: 2
  control-planes:
```

```
    - name: cp
      region-name: region1
    failure-zones:
    - AZ1
      common-service-components:
          - logging-producer
          - monasca-agent
          - freezer-agent
          - stunnel
          - lifecycle-manager-target
    clusters:
    - name: cluster1
      cluster-prefix: c1
      server-role: ROLE-CONTROLLER
      member-count: 2
      allocation-policy: strict
    service-components:

    ...
    - neutron-l2gateway-agent
    ... )
```

2. Update `l2gateway_agent.ini.j2`. For example, here the IP address (10.10.85.10) must be the management IP address of your 5930 switch. Open the file in vi:

```
$vi /home/stack/my_cloud/config/neutron/l2gateway_agent.ini.j2
```

3. Then make the changes:

```
[ovsdb]
# (StrOpt) OVSDB server tuples in the format
# <ovsdb_name>:<ip address>:<port>[,<ovsdb_name>:<ip address>:<port>]
# - ovsdb_name: a symbolic name that helps identifies keys and certificate files
# - ip address: the address or dns name for the ovsdb server
# - port: the port (ssl is supported)
ovsdb_hosts = hardware_vtep:10.10.85.10:6632
```

4. By default, the L2 gateway agent initiates a connection to OVSDB servers running on the L2 gateway switches. Set the attribute `enable_manager` to `True` if you want to change this behavior (to make L2 gateway switches initiate a connection to the L2 gateway agent). In this case, it is assumed that the Manager table in the OVSDB hardware_vtep schema on the L2 gateway switch has been populated with the management IP address of the L2 gateway agent and the port.

```
#enable_manager = False
#connection can be initiated by the ovsdb server.
#By default 'enable_manager' value is False, turn on the variable to True
#to initiate the connection from ovsdb server to l2gw agent.
```

5. If the port that is configured with `enable_manager = True` is any port other than 6632, update the `2.0/services/neutron/l2gateway-agent.yml` input model file with that port number:

```
endpoints:
    - port: '6632'
      roles:
        - ovsdb-server
```

6. Note: The following command can be used to set the Manager table on the switch from a remote system:

```
sudo vtep-ctl --db=tcp:10.10.85.10:6632 set-manager tcp:10.10.85.130:6632
```

7. For SSL communication, the command is:

```
sudo vtep-ctl --db=tcp:10.10.85.10:6632 set-manager ssl:10.10.85.130:6632
```

where **10.10.85.10** is the management IP address of the L2 gateway switch and **10.10.85.130** is the management IP of the host on which the L2 gateway agent runs.

Therefore, in the above topology, this command has to be repeated for **10.10.85.131** and **10.10.85.132**.

8. If you are not using SSL, comment out the following:

```
#l2_gw_agent_priv_key_base_path={{ neutron_l2gateway_agent_creds_dir }}/keys
#l2_gw_agent_cert_base_path={{ neutron_l2gateway_agent_creds_dir }}/certs
#l2_gw_agent_ca_cert_base_path={{ neutron_l2gateway_agent_creds_dir }}/ca_certs
```

9. If you are using SSL, then rather than commenting out the attributes, specify the directory path of the private key, the certificate, and the CA cert that the agent should use to communicate with the L2 gateway switch which has the OVSDB server enabled for SSL communication.

Make sure that the directory path of the files is given permissions 755, and the files' owner is root and the group is root with 644 permissions.

Private key: The name should be the same as the symbolic name used above in ovsdb_hosts attribute. The extension of the file should be ".key". With respect to the above example, the filename will be hardware_vtep.key

Certificate The name should be the same as the symbolic name used above in ovsdb_hosts attribute. The extension of the file should be ".cert". With respect to the above example, the filename will be hardware_vtep.cert

CA certificate The name should be the same as the symbolic name used above in ovsdb_hosts attribute. The extension of the file should be ".ca_cert". With respect to the above example, the filename will be hardware_vtep.ca_cert

10. To enable the HPE 5930 switch for SSL communication, execute the following commands:

```
undo ovsdb server ptcp
undo ovsdb server enable
ovsdb server ca-certificate flash:/cacert.pem bootstrap
ovsdb server certificate flash:/sc-cert.pem
ovsdb server private-key flash:/sc-privkey.pem
ovsdb server pssl port 6632
ovsdb server enable
```

11. Data from the OVSDB sever with SSL can be viewed using the following command:

```
ovsdb-client -C <ca-cert.pem> -p <client-private-key.pem> -c <client-cert.pem> dump ssl:10.10.85.10:6632
```

Routing between software and hardware

VTEP networks

In order to allow L2 gateway switches to send VxLAN packets over the correct tunnels destined for the compute node and controller node VTEPs, you must ensure that the cloud VTEP (compute and controller) IP addresses are in different network/subnet from that of the L2 gateway switches. You must also create a route between these two networks. This is explained below.

1. In the following example of the input model file `networks.yml`, the GUEST-NET represents the cloud data VxLAN network. REMOTE-NET is the network that represents the hardware VTEP network.

```
#networks.yml
 networks:
    - name: GUEST-NET
      vlanid: 103
      tagged-vlan: false
      cidr: 10.1.1.0/24
      gateway-ip: 10.1.1.10
      network-group: GUEST

    - name: REMOTE-NET
      vlanid: 183
      tagged-vlan: false
      cidr: 10.10.83.0/24
      gateway-ip: 10.10.83.3
      network-group: REMOTE
```

2. The route must be configured between the two networks in the `network-groups.yml` input model file:

```
#network_groups.yml
 network-groups:
    - name: REMOTE
      routes:
        - GUEST
```

```
      - name: GUEST
        hostname-suffix: guest
        tags:
          - neutron.networks.vxlan:
              tenant-vxlan-id-range: "1:5000"
        routes:
          - REMOTE
```

Note that the IP route is configured on the compute node. Per this route, the HPE 5930 acts as a gateway that routes between the two networks.

3. On the compute node, it looks like this:

```
stack@padawan-cp1-comp0001-mgmt:~$ sudo ip route
10.10.83.0/24 via 10.1.1.10 dev eth4
```

4. Run the following Ansible playbooks to apply the changes.

config-processor-run.yml:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

ready-deployment.yml:

```
ansible-playbook -i hosts/localhost ready-deployment.yml
```

hlm-reconfigure.yml:

```
cd ~/scratch/ansible/next/hos/ansible/
ansible-playbook -i hosts/verb_hosts hlm-reconfigure.yml
```

Notes:

1. Make sure that the controller cluster is able to reach the management IP address of the L2 gateway switch. Otherwise, the L2 gateway agents running on the controllers will not be able to reach the gateway switches.

2. Make sure that the interface on the baremetal server connected to the 5930 switch is tagged (this is explained shortly).

Connecting a baremetal server to the HPE 5930 switch

As Ubuntu (baremetal server) is not aware of tagged packets, it is not possible for a virtual machine to communicate with a baremetal box.

You (the administrator) must manually perform configuration changes to the interface in the HPE 5930 switch to which the baremetal server is connected so that the switch can send untagged packets. Either one of the following command sets can be used to do so:

```
Interface <interface number>
service-instance <service-instance id>
encapsulation untagged
xconnect vsi <vsi-name>
```

Or:

```
Interface <interface number>
service-instance <service-instance id>
encapsulation s-vid <vlan-id>
xconnect vsi <vsi-name> access-mode ethernet
```

There are two ways of configuring the baremetal server to communicate with virtual machines. If the switch sends tagged traffic, then the baremetal server should be able to receive the tagged traffic.

Configuration on a baremetal server

To receive tagged packets from the switch

If the configuration changes mentioned previously are not made on the switch to send untagged traffic to the baremetal server, then the following changes are required on the baremetal server so that it can receive tagged traffic from the switch.

Bare metal Management ip: 10.10.85.129 on interface em1 Switch 5930 is connected to baremetal on eth1 Need to set the IP address into tagged interface of eth1

1. create tagged (VLAN 183) interface

```
vconfig add eth1 183
```

2. Assign the IP address (10.10.10.129) to eth1.183 tagged interface (IP from the subnet 10.10.10.0/24 as VM(10.10.10.4) spawned in Compute node belongs to this subnet).

```
ifconfig eth1.183 10.10.10.129/24
```

NIC bonding and IRF configuration

With L2 gateway in actiondeployment, NIC bonding can be enabled on compute nodes. For more details on NIC bonding, please refer to *Interface Models* In order to achieve high availability, HPE 5930 switches can be configured to form a cluster using Intelligent Resilient Framework (IRF). Please refer to the *HPE FlexFabric 5930 Switch Series configuraton guide* for details.

Scale numbers tested

*   Number of neutron port MACs tested on a single switch: 4000
*   Number of HPE 5930 switches tested: 2
*   Number of baremetal connected to a single HPE 5930 switch: 100
*   Number of L2 gateway connections to different networks: 800

L2 gateway commands

These commands are not part of the L2 gateway deployment. They are to be executed after L2 gateway is deployed.

1. Create Network

```
neutron net-create net1
```

2. Create Subnet

```
neutron subnet-create net1 10.10.10.0/24
```

3. Boot a tenant VM (nova boot –image <Image-id> --flavor 2 –nic net-id=<net_id> <VM name>)

```
nova boot --image 1f3cd49d-9239-49cf-8736-76bac5360489 --flavor 2 --nic net-id=4f6c58b6-0acc-4e93-bb4c-439b38c27a23 VM
```

Assume the VM was assigned the IP address 10.10.10.4

4. Create the L2 gateway filling in your information here:

```
neutron l2-gateway-create --device name="<switch name>",interface_names="<interface_name>"  <gateway-name>
```

For this example:

```
neutron l2-gateway-create --device name="L2GTWY02",interface_names="Ten-GigabitEthernet1/0/5:3" gw1
```

Ping from the VM (10.10.10.4) to baremetal server and from baremetal server (10.10.10.129) to the VM Ping should not work as there is no gateway connection created yet.

5. Create l2 gateway Connection

```
neutron l2-gateway-connection-create gw1 net1 --segmentation-id 183
```

6. Ping from VM (10.10.10.4) to baremetal and from baremetal (10.10.10.129) to VM Ping should work.

7. Delete l2 gateway Connection

```
neutron l2-gateway-connection-delete <gateway id/gateway_name>
```

8. Ping from the VM (10.10.10.4) to baremetal and from baremetal (10.10.10.129) to the VM. Ping should not work as l2 gateway connection was deleted.

HPE Helion OpenStack® 5.0: Cloud Installation Overview

Before jumping into your installation, we recommend taking the time to read through our *Example Configurations* documentation to get an overview of the sample configurations HPE Helion OpenStack 5.0 offers. We have highly tuned example configurations for each of these Cloud models:

Name	Location
Entry-scale KVM with VSA model	`~/helion/examples/entry-scale-kvm-vsa`
Entry-scale KVM with VSA model with Dedicated Cluster for Metering, Monitoring, and Logging	`~/helion/examples/entry-scale-kvm-vsa-mml`
Entry-scale KVM with Ceph model	`~/helion/examples/entry-scale-kvm-ceph`
Mid-scale KVM with VSA model	`~/helion/examples/mid-scale-kvm-vsa`
Entry-scale ESX, KVM and VSA model	`~/helion/examples/entry-scale-esx-kvm-vsa`
Entry-scale ESX, KVM and VSA model with Dedicated Cluster for Metering, Monitoring, and Logging	`~/helion/examples/entry-scale-esx-kvm-vsa-mml`
Entry-scale Swift-only model	`~/helion/examples/entry-scale-swift`
Entry-scale Cloud with Ironic Flat Network	`~/helion/examples/entry-scale-ironic-flat-network`
Entry-scale Cloud with Ironic Multi-Tenancy	`~/helion/examples/entry-scale-ironic-multi-tenancy`

Using the Command-line

You should use the command-line if:

- You are installing a more complex or large-scale cloud.
- You have more than 15 nodes.
- You need to use availability zones or the server groups functionality of the cloud model. See the *Input Model page* for more information.
- You want to customize the cloud configuration beyond the tuned defaults that HPE provides out of the box.
- You need deeper customizations than are possible to express using the GUI.

Instructions for installing via the command-line are here:

- *Installing Mid-scale and Entry-scale KVM*
- *Installing Entry-scale ESX, KVM and VSA*

HPE Helion OpenStack® 5.0: Installing via the GUI

HPE Helion OpenStack 5.0 comes with a UI installer for installing your cloud the first time. Rather than searching through and editing a number of YAML files to input your specific configuration data, finding the correct location, and maintaining the format of the files, the UI presents fields in which you may enter your information. In addition, this latest version allows you to populate server information via a .csv file import.

Using the installer, the series Cobbler and Ansible scripts that initiate the installation of HPE Helion OpenStack are run for you behind the scenes.

The UI app populates the configuration files that the configuration processor uses as input to its processes. Therefore you can inspect the files whenever you like, or before reconfiguring your cloud.

Note, the GUI installer will periodically save the configuration files during the process and when you click the Deploy button at the end of the process. This also means that if you were to try to change anything in the files themselves mid process, the values would get overwritten by values you have entered in the GUI.

These YAML configuration files are stored in a local git repository as well, as explained in *Using git for Configuration Management*.

When installing via the GUI, you begin with a model (or a cloud configuration template) and edit the contents to reflect your environment.

However, the full range of customizations is not possible via the GUI as some configuration items are not exposed. Below is the list of what can and cannot be changed:

Changes to the following items may be made:

- servers (including Linux for HPE Helion installation configuration)
- networks
- disk models
- interface models
- NIC mappings

- NTP servers
- Name servers
- tags in network groups

Changes to the following items cannot be made:

- server groups
- server roles
- network groups
- firewall rules
- cloudConfig.yml (i.e. DNS, SMTP, firewall settings)
- control planes

Before you begin

Before you run the GUI installer to install your cloud, there are a number of things you need to do:

1. Prepare your servers
2. Gather your information

   - Server names
   - IP addresses
   - Server Roles
   - PXE MAC addresses
   - PXE IP addresses
   - PXE interfaces
   - IPMI/iLO IP address, username, password

3. Choose your *cloud model/example* template. No action other than an understanding of your needs is necessary at this point. You will indicate which model you wish to deploy in a GUI screen. Note, however, that the only VSA options for the *Entry-scale KVM with VSA* model is to have 0 or 3 VSA nodes. Note also that you cannot run the lifecycle manager on a dedicated node using the GUI installer; it can only run in the control plane using the models exposed in the GUI.
4. Before you use the GUI to install your cloud, you may install the operating system, Linux for HPE Helion, on your nodes (servers) if you prefer. Otherwise, the installer will install it for you.

   If you are installing the operating system on all your nodes yourself, note that you must do so using the Linux for HPE Helion image that is included in the HPE Helion OpenStack 5.0 package. You may find the instructions on the following pages helpful even when using your own tools to install the operating systems:

   - *Operating system install help for KVM-based clouds*
   - *Operating system install help for ESX-based clouds*

   Note: When following the guidance for operating system installation on those pages, stop before the sections for **Running the configuration processor**. The GUI installer will run those steps, and in the case where you have the GUI also install the operating systems, it will run all the previous steps (playbooks) as well.

Creating a CSV file for import

You may create a CSV file for import into the GUI, indicating your server information rather than entering it directly into the GUI. The following table shows the fields you need and those that are optional.

Field	Optional	OS Install Only?	Aliases
Server ID	NO	NO	server_id, id
IP Address	NO	NO	ip, ip_address, address
MAC Address	NO	YES	mac, mac_address, mac_addr
IPMI IP Address	NO	YES	ipmi_ip, ilo_ip
IPMI User	NO	YES	ipmi_user, ilo_user, user
IPMI Password	NO	YES	Ipmi_password, ilo_password, password
Server Role	NO	NO	server_role, role
Server Group	YES	NO	server_group, group

NIC Mapping	NO	NO	server_nic_map, nic_map, nic_mapping

Here "OS Install Only?" indicates that this field is ONLY needed if you have indicated that you want the installer to install HPE Linux on the servers.

The aliases are all the valid names that can be used in the CSV file for the column header for a given field. Note field names are not case sensitive and that you can use either ' ' (space) or '-' (hyphen) in place of underscore for a field name.

So, an example header line in the CSV file could be:

```
Server ID, ip address, mac-address, IPMI_ip, password, user, server-role, server-GROUP, nic mapping
```

The order of the fields does NOT have to match the order in the table above.

Run the GUI to install your cloud
The GUI is found on the lifecycle manager node at: **http://<lifecycle manager_IP>:79/dayzero**

To create a secure tunnel/port forwarding into Apache from the lifecycle manager to be able to use the GUI in a browser, SSH tunnel into the lifecycle manager with the username/password set up when installing the lifecycle manager. The username and password should be what was set in "Set up the lifecycle manager" in the installation instructions. For example:

```
ssh -v -N -L 8080:<lifecycle manager_IP>:79 <Username>@<lifecycle manager_IP>
```

Then on your local machine, the one you are tunneling from, point your web browser to: **http://localhost:8080/dayzero**

As you proceed through the GUI, you can either enter your server names and IP addresses manually or import them from the CSV file using the CSV fields described above.

In the Assign network groups: External VM network fields, **do not** add entries for CIDR, Gateway, or IP Range, only VLAN ID. Neutron will assign these.

When the installation is complete, the Ansible log will be displayed on the final page.

Use the installer securely

The GUI installer is a web-based application that runs on the lifecycle manager node and is implemented in node.js. Currently this application runs behind Apache 2.0 where requests are proxied to the node.js Express Web Server. The node.js Express Server runs on port 3000 and Apache runs on port 79.

You should access the installer through Apache and not directly from the Express server. An Example URL looks like this: http://<lifecycle manager-host-ip>:79/dayzero When you finish installing, the Installer will be disabled via Ansible. It is also disabled once deployment is completed following the CLI instructions..

The GUI is used only once: during the initial installation of your cloud.

For VSA deployments, once the GUI competes its installation, you must follow the steps outlined in the *VSA Configuration*page.

After deployment, you may also continue to *Verifying Your Installation* and *Post-Installation Checklist*.

To understand a cloud configuration more thoroughly and to learn how to make any changes later, visit these documentation pages:

• *HPE Helion OpenStack 5.0 Input Model*
• *Using Git for Configuration Management*

Note:

Because the GUI installer is disabled after deployment is complete, whether you installed your cloud via the GUI or via the command line, if you want to re-enable the GUI later but are not re-installing from scratch, run these commands:

```
sudo a2ensite dayzero-apache.conf
sudo systemctl start dayzero.service
```

Final Steps

• The public Horizon Dashboard connection is over TLS, thus please use HTTPS:// to navigate to Horizon with the IP address the installer presents once install is complete.
• Visit the *Ceph configuration page* for Ceph post-install steps.

HPE Helion OpenStack® 5.0: DNS Service Installation Overview

DNS Service Installation Overview

The HPE Helion OpenStack DNS Service supports several different backends for domain name service. The choice of backend must be included in the deployment model before the HPE Helion OpenStack install is completed.

⚠️ **Warning:** By default any user in the project is allowed to manage a DNS domain. This can be changed by updating the Policy.json file for Designate.

The backends that are available within the DNS Service are separated into two categories, self-contained and external.

Table 1:

Category	Backend	Description	Recommended For
Self-contained	PowerDNS 3.4.1, BIND 9.9.5	All components necessary will be installed and configured as part of the HPE Helion OpenStack install.	POCs and customers who wish to keep cloud and traditional DNS separated.
External	InfoBlox 📝 **Note:** tested version 7.0.2	The authoritative DNS server itself is external to HPE Helion OpenStack. Management and configuration is out of scope for the lifecycle manager but remains the responsibility of the customer.	Customers who wish to integrate with their existing DNS infrastructure.

Getting Started
HPE Helion OpenStack® 5.0: Install the DNS Service with PowerDNS

Install DNS Service with PowerDNS

HPE Helion OpenStack DNS Service defaults to the PowerDNS Backend if another backend is not configured for domain name service. PowerDNS will be deployed to one or more control planes clusters. The following configuration example denotes where the PowerDNS service is installed.

Configure the Backend

To configure the backend for PowerDNS, follow these steps.

1. Ensure the DNS Service components, and the PowerDNS component have been placed on a cluster. PowerDNS may be placed on a separate cluster to the other DNS Service components.

```
control-planes:
        - name: control-plane-1
        region-name: region1

        clusters:
        - name: cluster1
        service-components:
        - lifecycle-manager
        - mysql
        - ip-cluster
        - apache2
        - ...
        - designate-api
        - designate-central
        - designate-pool-manager
        - designate-zone-manager
        - designate-mdns
        - designate-client
        - powerdns
```

2. Edit the `~/helion/my_cloud/definitions/data/network_groups.yml` file to include the powerdns-ext.

```
- name: EXTERNAL-API
hostname-suffix: extapi
component-endpoints:
  - powerdns-ext
```

© Copyright Hewlett Packard Enterprise Development Company, L.P.

```
load-balancers:
 - provider: ip-cluster
```

3. Edit the `~/helion/my_cloud/definitions/data/firewall_rules.yml` to allow UDP/TCP access.

```
    - name: DNSudp
        # network-groups is a list of all the network group names that the rules apply to
        network-groups:
        - EXTERNAL-API
        rules:
        - type: allow
          # range of remote addresses in CIDR format that this rule applies to
          remote-ip-prefix:  0.0.0.0/0
          port-range-min: 53
          port-range-max: 53
          protocol: udp

    - name: DNStcp
        # network-groups is a list of all the network group names that the rules apply to
        network-groups:
        - EXTERNAL-API
        rules:
        - type: allow
          # range of remote addresses in CIDR format that this rule applies to
          remote-ip-prefix:   0.0.0.0/0
          port-range-min: 53
          port-range-max: 53
          protocol: tcp
```

Please see *HPE Helion OpenStack 5.0: Designate Initial Configuration* for post installation DNS Service configuration.

HPE Helion OpenStack® 5.0: Install the DNS Service with BIND

Install DNS Service with BIND

HPE Helion OpenStack DNS Service and BIND can be installed together instead of the default **PowerDNS** backend. BIND will be deployed to one or more control planes clusters. The following configuration example denotes where the BIND service is installed.

Configure the Backend

Ensure the DNS Service components, and the BIND component have been placed on a cluster. BIND may be placed on a separate cluster to the other DNS Service components. Additionally, ensure the default **powerdns** component has been removed.

```
control-planes:
        - name: control-plane-1
        region-name: region1

        clusters:
        - name: cluster1
        service-components:
        - lifecycle-manager
        - mysql
        - ip-cluster
        - apache2
        - ...
        - designate-api
        - designate-central
        - designate-pool-manager
        - designate-zone-manager
        - designate-mdns
        - designate-client
        - bind
```

Update Input Model

Once the backend is configured, you will need to add `bind-ext` to the `network_groups.yml` file.

1. Edit `helion/my_cloud/definition/data/network_groups.yml`, add `bind-ext` to compnent-endpoints.

```
name: EXTERNAL-API
hostname-suffix: extapi
component-endpoints:
- bind-ext
```

2. Save file.

HPE Helion OpenStack® 5.0: Install the DNS Service with InfoBlox

Install DNS Service with InfoBlox

HPE Helion OpenStack DNS Service can be installed with the **InfoBlox** backend instead of the default **PowerDNS** backend. In order to use InfoBlox as the backend, all prerequisites must be satisfied and the configuration of InfoBlox complete.

📝 **Note:** No DNS server will be deployed onto the HPE Helion OpenStack nodes. Instead, zones will be hosted on the **InfoBlox** servers.

⚠️ **Important:** The InfoBlox integration was enabled and validated in HPE Helion OpenStack 3.x. To use InfoBlox with version [4.x / 5.x] we recommend that you review the InfoBlox Deployment guide which can be found here: *https://www.infoblox.com/wp-content/uploads/infoblox-deployment-guide-infoblox-openstack-driver.pdf* and address any questions with your InfoBlox support contact.

Prerequisites

1. An existing InfoBlox system deployed within your organization.
2. IPs and other credentials required to access the InfoBlox WS API.
3. Network connectivity to and from the cluster containing designate and the InfoBlox WS API.

This information is available from your InfoBlox system administrator.

Configure the Backend

If not already present, DNS Service components must be placed on a cluster. Additionally, ensure the default **powerdns** component has been removed, and **bind** has not been added. Replace the `designate-mdns` component with the `designate-mdns-external` component.

```
control-planes:
        - name: control-plane-1
          region-name: region1
             - lifecycle-manager-target

        clusters:
          - name: cluster1
            service-components:
            - lifecycle-manager
            - mysql
            - ip-cluster
            - apache2
            - ...
            - designate-api
            - designate-central
            - designate-pool-manager
            - designate-zone-manager
            - designate-mdns-external
            - designate-client
```

You will need to provide DNS Service information details on your InfoBlox deployment. Open the designate pool-manager configuration template:

```
$ cd ~/helion/my_cloud
$ nano config/designate/pool-manager.conf.j2
```

In the `config/designate/pool-manager.conf.j2`, find the following code block:

```
  nameservers:
    - host: <infoblox-server-ip>
      port: <infoblox-port-number>
  ...
     also_notifies:
    - host: <infoblox-server-ip>
      port: <infoblox-port-number>
```

Make the following changes:

1. Uncomment the block for infoblox and also_notifies. In Jinja2, this means replacing the {# and #} markers with {{ and }}.
2. Fill in the API URL, username, password, and all remaining fields.
3. Save the file.

Once complete, the block should look like this:

```
    - type: infoblox
      description: infoblox Cluster

      masters:
{% if DES_PMG.consumes_DES_MDN_EXT is defined %}
{% for mdn_member in DES_PMG.consumes_DES_MDN_EXT.members.private %}
        - host: {{ mdn_member.ip_address }}
          port: {{ mdn_member.port }}
{% endfor %}
{% endif %}

      options:
        wapi_url: https://<infoblox-server-ip>/wapi/v2.2.2/
        username: hos-designate
        password: MySecretPassword
        ns_group: designate
        sslverify: False
        dns_view: default
        network_view: default
```

You will need to inspect and commit the changes before proceeding with the deployment:

```
$diff --git a/hos/ansible/roles/designate-pool-manager/templates/pools.yaml.j2 b/hos/ansible/roles/designate-pool-manager/templates/pools.yaml.j2
index 291c6c9..b7fb39c 100644
--- a/hos/ansible/roles/designate-pool-manager/templates/pools.yaml.j2
+++ b/hos/ansible/roles/designate-pool-manager/templates/pools.yaml.j2
@@ -28,6 +28,8 @@
      priority: 2

   nameservers:
+    - host: <infoblox-server-ip>
+      port: <infoblox-port-number>
  {% if DES_PMG.consumes_FND_PDN is defined %}
  {% for pdn_member in DES_PMG.consumes_FND_PDN.members.private %}
    - host: {{ pdn_member.ip_address }}
@@ -40,7 +42,9 @@
      port: {{ bnd_member.port }}
  {% endfor %}
  {% endif %}
-#  also_notifies:
+  also_notifies:
+    - host: <infoblox-server-ip>
+      port: <infoblox-port-number>
   targets:
```

```
 {% if DES_PMG.consumes_FND_PDN is defined %}
    - type: powerdns
@@ -89,27 +93,27 @@
 {% endfor %}
 {% endif %}

-#
-#    - type: infoblox
-#      description: infoblox Cluster
-#
-#      masters:
+
+    - type: infoblox
+      description: infoblox Cluster
+
+      masters:
 {% if DES_PMG.consumes_DES_MDN_EXT is defined %}
 {% for mdn_member in DES_PMG.consumes_DES_MDN_EXT.members.private %}
-#        - host: {{ mdn_member.ip_address }}
-#          port: {{ mdn_member.port }}
+        - host: {{ mdn_member.ip_address }}
+          port: {{ mdn_member.port }}
 {% endfor %}
 {% endif %}
-#
-#      options:
-#        wapi_url: https://127.0.0.1/wapi/v2.2.2/
-#        username: admin
-#        password: infoblox
-#        ns_group: designate
-#        sslverify: False
-#        dns_view: default
-#        network_view: default
-#
+
+      options:
+        wapi_url: https://127.0.0.1/wapi/v2.2.2/
+        username: admin
+        password: infoblox
+        ns_group: designate
+        sslverify: False
+        dns_view: default
```

SSL and CA Certificate

To enable SSL Verify, edit the following file:

```
$ cd ~/helion/my_cloud
$ nano config/designate/pools.yaml.j2
```

In the infoblox section, set sslverify to true:

```
sslverify: True
```

To generate a CA certificate for InfoBlox, follow these steps:

1. Login to the InfoBlox User Interface
2. Generate a self signed certificate by selecting: System->Certificate->HTTP Cert->Generate Self Signed Certificate
3. Provide the hostname as the InfoBlox server-ip
4. Reload the InfoBlox User Interface. The certificate will be loaded and can be verified through the browser.

5. If you want to download the certificate, select: System->Certificate->HTTP Cert->Download the Certificate

6. Copy Certificate file to ~/helion/my_cloud/config/tls/cacerts/

Commit changes.

```
$ git commit -a -m "Configure Designate's InfoBlox Backend"
```

HPE Helion OpenStack® 5.0: Configure DNS Domain and NS Records

Configure DNS Domain and NS Records

To configure the default DNS domain and Name Server records for the default pool, follow these steps.

1. Ensure that designate_config.yml file is present in the ~/helion/my_cloud/definition/data/designate folder. If the file or folder is not present, create the folder and copy designate_config.yml file from one of the example input models (for example, ~/helion/examples/entry-scale-esx-kvm-vsa/data/designate/designate_config.yml).

2. Modify the **dns_domain** and/or **ns_records** entries in the designate_config.yml file.

```
data:
dns_domain: example.org.
ns_records:
    hostname: ns1.example.org.
    priority: 1
    hostname: ns2.example.org.
    priority: 2
```

3. Edit your input model's control_plane.yml file to include **DESIGNATE-CONFIG-CP1** in **configuration-data** section.

```
control-planes:
   - name: control-plane-1
     region-name: region1
     lifecycle-manager-target
     configuration-data:
        - DESIGNATE-CONFIG-CP1
        - NEUTRON-CONFIG-CP1
```

4. Continue your cloud deployment by reviewing and committing your changes.

```
$ git add ~/helion/my_cloud/definition/data/designate/designate_config.yml
$ git commit -m "Adding DNS Domain and NS Records"
```

Note: In an *entry-scale model*, you will have 3 ns_records since the DNS service runs on all three control planes. In a *mid-scale model* or *dedicated metering, monitoring and logging model* the above example would be correct since there are only two controller nodes.

HPE Helion OpenStack® 5.0: Magnum Overview

Magnum Overview

The HPE Helion OpenStack Magnum Service provides container orchestration engines such as Docker Swarm, Kubernetes, and Apache Mesos available as first class resources. HPE Helion OpenStack Magnum uses Heat to orchestrate an OS image which contains Docker and Kubernetes and runs that image in either virtual machines or bare metal in a cluster configuration.

HPE Helion OpenStack® 5.0: Install the Magnum Service

Install the Magnum Service

Installing the Magnum Service can be performed as part of a new HPE Helion OpenStack 5.0 environment or can be added to an existing HPE Helion OpenStack 5.0 environment. Both installations require container management services, running in Magnum cluster VMs with access to specific Openstack API endpoints. The following TCP ports need to be open in your firewall to allow access from VMs to external (public) HPE Helion OpenStack endpoints.

TCP Port	Service
5000	Idenity
8004	Heat
9511	Magnum

Additionally, Magnum is dependent on the following OpenStack services.

- Keystone
- Heat
- Nova KVM
- Neutron
- Glance
- Cinder
- Swift
- Barbican
- LBaaS v2 (Octavia) - *optional*

⚠️ **Warning:** Magnum relies on the public discovery service **https://discovery.etcd.io** during cluster bootstrapping and update. This service does not perform authentication checks. Although running a cluster can't be harmed by unauthorized changes in the public discovery registry, it can be compromised during a cluster update operation. To avoid this, it is recommended that you keep your cluster discovery URL (i.e. https://discovery.etc.io/<some random ID>) secret.

Installing Magnum as part of new HPE Helion OpenStack 5.0 environment

Magnum components are already included in example HPE Helion OpenStack models based on Nova KVM, such as **entry-scale-kvm-vsa**, **entry-scale-kvm-vsa-mml** and **mid-scale**. These models contain the Magnum dependencies (see above). You can follow generic installation instruction for Mid-Scale and Entry-Scale KM model by using this guide: *Installing Mid-scale and Entry-scale KVM Models*

📋 **Note:**

1. If you modify the cloud model to utilize a dedicated lifecycle manager, add `magnum-client` item to the list of service components for the lifecycle manager cluster.
2. Magnum needs a properly configured external endpoint. While preparing the cloud model, ensure that `external-name` setting in `data/network_groups.yml` is set to valid hostname, which can be resolved on DNS server, and a valid TLS certificate is installed for your external endpoint. For non-production test installations, you can omit `external-name`. In test installations, the HPE Helion OpenStack installer will use an IP address as a public endpoint hostname, and automatically generate a new certificate, signed by the internal CA. Please refer to *HPE Helion OpenStack 5.0: Configuring Transport Layer Security (TLS)* for more details.
3. To use LBaaS v2 (Octavia) for container management and container applications, follow the additional steps to configure LBaaS v2 in the guide *Configuring Load Balancer as a Service*.

Adding Magnum to existing HPE Helion OpenStack 5.0 environment

Adding Magnum to an already deployed HPE Helion OpenStack 5.0 installation or during and upgrade can be achieved by performing the following steps.

1. Add items listed below to the list of service components in `/home/stack/helion/my_cloud/definition/data/control_plane.yml`. Add them to clusters which have `server-role` set to `CONTROLLER-ROLE` (entry-scale models) or `CORE_ROLE` (mid-scale model).

```
 - magnum-api
 - magnum-conductor
```

2. If your environment utilizes a dedicated lifecycle manager, add `magnum-client` to the list of service components for the lifecycle manager.
3. Commit your changes to the local git repository. Run the following playbooks as described in *Using Git for Configuration Management* for your Installation.

- `config-processor-run.yml`
- `ready-deployment.yml`
- `site.yml`

4. Ensure that your external endpoint is configured correctly. The current public endpoint configuration can be verified by running the following commands on the lifecycle manager.

```
$ source service.osrc
$ openstack endpoint list --interface=public --service=identity
+----------------------------------+--------+--------------+--------------+---------+-----------+-----------------------------+
| ID                               | Region | Service Name | Service Type | Enabled | Interface | URL                         |
+----------------------------------+--------+--------------+--------------+---------+-----------+-----------------------------+
```

```
| d83e87ef474a497f8d8373357cce4aa3 | region1 | keystone      | identity     | True    | public    | https://10.245.41.168:5000/v2.0 |
+----------------------------------+---------+--------------+--------------+---------+-----------+----------------------------------+
```

Ensure that the endpoint URL is using either an IP address, or a valid hostname, which can be resolved on the DNS server. If the URL is using an invalid hostname (for example, 'myhelion.test'), follow the steps described in *HPE Helion OpenStack 5.0: Configuring Transport Layer Security (TLS)* to configure a valid external endpoint. You will need to update the `external-name` setting in the `data/network_groups.yml` to a valid hostname, which can be resolved on DNS server, and provide a valid TLS certificate for the external endpoint. For non-production test installations, you can omit the `external-name`. The HPE Helion OpenStack installer will use an IP address as public endpoint hostname, and automatically generate a new certificate, signed by the internal CA. Please refer to *HPE Helion OpenStack 5.0: Configuring Transport Layer Security (TLS)* for more information.

5. Ensure that LBaaS v2 (Octavia) is correctly configured. See *Configuring Load Balancer as a Service* for more information.

⚠️ **Warning:** By default HPE Helion OpenStack stores the private key used by Magnum and its passphrase in Barbican which provides a secure place to store such information. You can change this such that this sensitive information is stored on the file system or in the database without encryption. Making such a change exposes you to the risk of this information being exposed to others. If stored in the database then any database backups, or a database breach, could lead to the disclosure of the sensitive information. Similarly, if stored unencrypted on the file system this information is exposed more broadly than if stored in Barbican.

HPE Helion OpenStack® 5.0: Installing Mid-scale and Entry-scale KVM

- *Set up the Lifecycle-manager*
- *Configure Your Environment*
- *Provision Your Baremetal Nodes*
- *Run the Configuration Processor*
- *Configure TLS*
- *Deploy the Cloud*
- *Configure a Block Storage Backend (Optional)*
- *Post-Installation Verification and Administration*

Important Notes

- If you are looking for information about when to use the GUI installer and when to use the CLI, see the *Installation Overview*.
- Review the *recommended minimum hardware requirements* that we have listed.
- Review the *Release Notes* to make yourself aware of any known issues and limitations.
- The installation process can occur in different phases. For example, you can install the control plane only and then add Compute nodes afterwards if you would like.
- If you run into issues during installation, we have put together a list of *Installation Troubleshooting Steps* you can reference.
- Make sure all disks on the system(s) are wiped before you begin the install. (For Swift, refer to *Swift Requirements for Device Group Drives*)
- There is no requirement to have a dedicated network for OS-install and system deployment, this can be shared with the management network. More information can be found on the *Example Configuration* page.
- You may see the terms deployer and lifecycle manager used interchangeably. These are referring to the same nodes in your environment.
- When running the Ansible playbook in this installation guide, if a runbook fails you will see in the error response to use the `--limit` switch when retrying a playbook. This should be avoided. You can simply re-run any playbook without this switch.
- DVR is not supported with ESX compute.
- When you attach a Cinder volume to the VM running on the ESXi host, the volume will not get detected automatically. Make sure to set the image metadata **vmware_adaptertype=lsiLogicsas** for image before launching the instance. This will help to discover the volume change appropriately.

Before You Start

We have put together a *Pre-Installation Checklist* that should help with the recommended pre-installation tasks.

Set up the Lifecycle Manager

Installing the Lifecycle Manager

The lifecycle manager will contain the installation scripts and configuration files to deploy your cloud. You can set up the lifecycle manager on a dedicated node or you do so on your first controller node. The default choice is to use the first controller node as the lifecycle manager.

1. Sign in and download the product and signature files at the link below:

    a. *Software Entitlement Portal*

2. You can verify the download was complete via the signature verification process outlined *here*.

3. Boot your lifecycle manager from the ISO contained in the download.

4. Enter "install" to start installation.

> 📝 **Note:** "install" is all lower case

5. Select the language. Note that only the English language selection is currently supported.
6. Select the location.
7. Select the keyboard layout.
8. Select the primary network interface, if prompted:

    a. Assign IP address, subnet mask, and default gateway
9. Create new account:

    a. Enter a username.
    b. Enter a password.
    c. Enter time zone.

Once the initial installation is finished, complete the lifecycle manager setup with these steps:

1. Ensure your lifecycle manager has a valid DNS nameserver specified in `/etc/resolv.conf`.
2. Set the environment variable LC_ALL:

```
export LC_ALL=C
```

> 📝 **Note:** This can be added to `~stack/.bashrc` or `/etc/bash.bashrc`.

At the end of this section you should have a node set up with Linux for HPE Helion on it.

Configure and Run the Lifecycle Manager

> ⚠️ **Important:** It is critical that you don't run any of the commands below as the `root` user or use `sudo`, unless it is stated explicitly in the steps. Run then as the user you just created (or `stack` if you left the default of "stack").

1. Log into your lifecycle manager as the user you created and mount the install media at `/media/cdrom`. It may be necessary to use `wget` or another file transfer method to transfer the install media to the lifecycle manager before completing this step. Here is the command to mount the media:

    Example for HPE Helion OpenStack® 5.0

    ```
    sudo mount HelionOpenStack-5.0.iso /media/cdrom
    ```
2. Unpack the tarball that is in the `/media/cdrom/hos/` directory:

    Example for HPE Helion OpenStack 5.0

    ```
    tar xvf /media/cdrom/hos/hos-5.0.0-<timestamp>.tar
    ```
3. Run the hos-init.bash script which is included in the build:

    Example for HPE Helion OpenStack® 5.0

    ```
    ~/hos-5.0.0/hos-init.bash
    ```

    You will be prompted to enter an optional SSH passphrase when running `hos-init.bash`. This passphrase is used to protect the key used by Ansible when connecting to its client nodes. If you do not want to use a passphrase then just press return at the prompt.

    For automated installation (e.g. CI) it is possible to disable SSH passphrase prompting by setting the `HOS_INIT_AUTO` environment variable before running `hos-init.bash`, like this:

    ```
    export HOS_INIT_AUTO=y
    ```

If you have protected the SSH key with a passphrase then execute the following commands to avoid having to enter the passphrase on every attempt by Ansible to connect to its client nodes:

```
eval $(ssh-agent)
ssh-add ~/.ssh/id_rsa
```

At the end of this section you should have a local directory structure, as described below:

```
~/helion/                    Top level directory
~/helion/examples/           Directory contains the config input files of the example clouds
~/helion/my_cloud/definition/   Directory contains the config input files
~/helion/my_cloud/config/    Directory contains .j2 files which are symlinks to the ~/helion/hos/ansible directory
~/helion/hos/                Directory contains files used by the installer
```

⚠️ **Warning:** It's important that you do not add any extra files in the ~/helion directory on your lifecycle manager. This includes temporary save files. Doing so will cause errors during the installation.

❗ **Important:** You should be aware that the files in the ~/helion/my_cloud/config/ directory are symlinks to the ~/helion/hos/ansible/ directory. For example, ~/helion/my_cloud/config/keepalived/defaults.yml is a symbolic link to ~/helion/hos/ansible/roles/keepalived/defaults/main.yml

```
ls -al ~/helion/my_cloud/config/keepalived/defaults.yml
lrwxrwxrwx 1 stack stack 55 May 24 20:38 /home/stack/helion/my_cloud/config/keepalived/defaults.yml -> ../../../hos/ansible/roles/keepalived/defaults/main.yml
```

If you are using a tool like sed to make edits to files in this directory, you might break the symbolic link and create a new copy of the file. To maintain the link, you will need to force sed to follow the link:

```
sed -i --follow-symlinks 's$keepalived_vrrp_offset: 0$keepalived_vrrp_offset: 2$' ~/helion/my_cloud/config/keepalived/defaults.yml
```

Alternatively, you could directly edit the target of the link ~/helion/hos/ansible/roles/keepalived/defaults/main.yml.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Configure Your Environment

During the configuration phase of the installation you will be making modifications to the example configuration input files to match your cloud environment. You should use the *Example Configurations* documentation for detailed information on how to do this. There is also a README.md file included in each of the example directories on the lifecycle manager that has useful information about the models.

In the steps below we show how to set up the directory structure with the example input files as well as use the optional encryption methods for your sensitive data.

1. Setup your configuration files, as follows:

   a. Copy the example configuration files into the required setup directory and edit them to contain the details of your environment.

   For example, if you want to use the Helion Mid-scale KVM with VSA model, you can use this command to copy the files to your cloud definition directory:

   ```
   cp -r ~/helion/examples/mid-scale-kvm-vsa/* ~/helion/my_cloud/definition/
   ```

   If you want to use the Helion Entry-scale KVM with VSA model, you can use this command to copy the files to your cloud definition directory:

   ```
   cp -r ~/helion/examples/entry-scale-kvm-vsa/* ~/helion/my_cloud/definition/
   ```

   b. Begin inputting your environment information into the configuration files in the ~/helion/my_cloud/definition directory.

   If you are using the Mid-scale or Entry-scale KVM with VSA model, see *Modifying the Entry-scale KVM with VSA model for your Environment* for details.

   If you are using the Entry-scale KVM with Ceph model, see *Ceph Overview* for details.

2. [OPTIONAL] - You can use the hosencrypt.py script to encrypt your iLo passwords. This script uses OpenSSL.

   a. Change to the Ansible directory:

   ```
   cd ~/helion/hos/ansible
   ```

   b. Put the encryption key into the following environment variable:

   ```
   export HOS_USER_PASSWORD_ENCRYPT_KEY=<encryption key>
   ```

   c. Run the python script below and follow the instructions. Enter a password that you want to encrypt.

```
./hosencrypt.py
```

   d. Take the string generated and place it in the `"ilo-password"` field in your `~/helion/my_cloud/definition/data/servers.yml` file, remembering to enclose it in quotes.

   e. Repeat the above for each server.

> 📝 **Note:** Before you run any playbooks, remember that you need to export the encryption key in the following environment variable: `export HOS_USER_PASSWORD_ENCRYPT_KEY=<encryption key>`

3. Commit your configuration to the *local git repo*, as follows:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

> ⚠️ **Important:** This step needs to be repeated any time you make changes to your configuration files before you move onto the following steps. See *Using Git for Configuration Management* for more information.

Provision Your Baremetal Nodes

To provision the baremetal nodes in your cloud deployment you can either use the automated operating system installation process provided by HPE Helion OpenStack or you can use the 3rd party installation tooling of your choice. We will outline both methods below:

Using 3rd Party Baremetal Installers

If you do not wish to use the automated operating system installation tooling included with 5.0 then the requirements that have to be met using the installation tooling of your choice are:

- The operating system must be installed via the HPE Linux for HPE Helion OpenStack ISO provided on the *Software Entitlement Portal*.
- Each node must have SSH keys in place that allows the same user from the lifecycle manager node who will be doing the deployment to SSH to each node without a password.
- Passwordless sudo needs to be enabled for the user.
- There should be a LVM logical volume as `/root` on each node.
- If the LVM volume group name for the volume group holding the "root" LVM logical volume is hlm-vg then it will align with the disk input models in the examples.
- Ensure that `openssh-server`, `python`, `python-apt`, and `rsync` are installed.

If you chose this method for installing your baremetal hardware, skip forward to the *Run the Configuration Processor* step.

If you would like to use the automated operating system installation tools provided by 5.0 then complete all of the steps below.

Using the Automated Operating System Installation Provided by HPE Helion OpenStack

Part One: Deploy Cobbler

This phase of the install process takes the baremetal information that was provided in `servers.yml` and installs the Cobbler provisioning tool and loads this information into Cobbler. This sets each node to `netboot-enabled: true` in Cobbler. Each node will be automatically marked as `netboot-enabled: false` when it completes its operating system install successfully. Even if the node tries to PXE boot subsequently, Cobbler will not serve it. This is deliberate so that you can't reimage a live node by accident.

The `cobbler-deploy.yml` playbook prompts for a password - this is the password that will be encrypted and stored in Cobbler, which is associated with the user running the command on the lifecycle manager, that you will use to log in to the nodes via their consoles after install. The username is the same as the user set up in the initial dialogue when installing the lifecycle manager from the iso, and is the same user that is running the cobbler-deploy play.

1. Run the following playbook which confirms that there is iLo connectivity for each of your nodes so that they are accessible to be re-imaged in a later step:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost bm-power-status.yml
```

2. Run the following playbook to deploy Cobbler:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost cobbler-deploy.yml
```

Part Two: Image the Nodes

This phase of the install process goes through a number of distinct steps:

1. Powers down the nodes to be installed
2. Sets the nodes hardware boot order so that the first option is a network boot.

3. Powers on the nodes. (The nodes will then boot from the network and be installed using infrastructure set up in the previous phase)
4. Waits for the nodes to power themselves down (this indicates a success install). This can take some time.
5. Sets the boot order to hard disk and powers on the nodes.
6. Waits for the nodes to be ssh-able and verifies that they have the signature expected.

The reimage command is:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost bm-reimage.yml [-e nodelist=node1,node2,node3]
```

If a nodelist is not specified then the set of nodes in cobbler with `netboot-enabled: True` is selected. The playbook pauses at the start to give you a chance to review the set of nodes that it is targeting and to confirm that it's correct.

You can use the command below which will list all of your nodes with the `netboot-enabled: True` flag set:

```
sudo cobbler system find --netboot-enabled=1
```

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Run the Configuration Processor

Once you have your configuration files setup you need to run the configuration processor to complete your configuration.

When you run the configuration processor you will be prompted for two passwords. Enter the first password to make the configuration processor encrypt its sensitive data, which is comprised of the random inter-service passwords that it generates and the ansible `group_vars` and `host_vars` that it produces for subsequent deploy runs. You will need this password for subsequent ansible deploy and configuration processor runs. If you wish to change an encryption password that you have already used when running the configuration processor then enter the new password at the second prompt, otherwise just press carriage return to bypass this.

Run the configuration processor with this command:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

For automated installation (e.g. CI) you can specify the required passwords on the ansible command line. For example, the command below will disable encryption by the configuration processor

```
ansible-playbook -i hosts/localhost config-processor-run.yml -e encrypt="" -e rekey=""
```

If you receive an error during this step then there is probably an issue with one or more of your configuration files. We recommend that you verify that all of the information in each of your configuration files is correct for your environment and then commit those changes to git using the instructions in the previous section before re-running the configuration processor again.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Configuring TLS

⚠ **Important:** This section is optional, but recommended, for a HPE Helion OpenStack installation.

After you run the configuration processor the first time, the IP addresses for your environment will be generated and populated in the `~/helion/my_cloud/info/address_info.yml` file. It's at this point that you will want to take into consideration whether or not you want to configure TLS and setup a SSL certificate for your environment. Please read *Enabling TLS* before proceeding for how to achieve this.

Deploy the Cloud

1. Use the playbook below to create a deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

2. [OPTIONAL] - Run the `wipe_disks.yml` playbook to ensure all of your partitions on your nodes are completely wiped before continuing with the installation. If you are using fresh machines this step may not be necessary.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts wipe_disks.yml
```

If you have used an encryption password when running the configuration processor use the command below and enter the encryption password when prompted:

```
ansible-playbook -i hosts/verb_hosts wipe_disks.yml --ask-vault-pass
```

3. Run the `site.yml` playbook below:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts site.yml
```

If you have used an encryption password when running the configuration processor use the command below and enter the encryption password when prompted:

```
ansible-playbook -i hosts/verb_hosts site.yml --ask-vault-pass
```

📝 **Note:** The step above runs `osconfig` to configure the cloud and `hlm-deploy` to deploy the cloud. Therefore, this step may run for a while, perhaps 45 minutes or more, depending on the number of nodes in your environment.

4. Verify that the network is working correctly. Ping each IP in the `/etc/hosts` file from one of the controller nodes.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Configure a Block Storage Backend (Optional)

HPE Helion OpenStack supports VSA, 3PAR, and Ceph as block storage backend options. You can utilize one or more of these as setting up multiple block storage backends and multiple volume types is supported.

Regardless of whether you have a single or multiple block storage backends defined in your `cinder.conf.j2` file then you can create one or more volume types using the specific attributes associated with the backend. You can find details on how to do that for each of the supported backend types here:

- *VSA Configuration*
- *Ceph Configuration*
- *3PAR Configuration*

Post-Installation Verification and Administration

We recommend verifying the installation using the *Cloud Verification* page.

There are also a list of other common post-installation administrative tasks listed in the *Common Post-Installation Tasks* list.

HPE Helion OpenStack® 5.0: Installation for Helion Entry Scale ESX (with OVSvApp, KVM with VSA Model

This document describes the procedure for the deployment of an ESX cloud using input model and preparing and deploying ESX Computes and OVSvAPPs.

It contains the following topics:

- *Prerequisite*
- *Deploy ESX Cloud with OVSvAPP*
- *Procedure to Deploy ESX Cloud with OVSvAPPs*
- *Prepare and Deploy ESX Computes and OVSvAPPs*

⚠️ **Important:** Before you start your ESX cloud deployment make sure to you read the following instructions carefully.

Important Notes

- If you are looking for information about when to use the GUI installer and when to use the CLI, see the *Installation Overview*.
- Review the *recommended minimum hardware requirements* that we have listed.
- Review the *Release Notes* to make yourself aware of any known issues and limitations.
- The installation process can occur in different phases. For example, you can install the control plane only and then add Compute nodes afterwards if you would like.
- If you run into issues during installation, we have put together a list of *Installation Troubleshooting Steps* you can reference.
- Make sure all disks on the system(s) are wiped before you begin the install. (For Swift, refer to *Swift Requirements for Device Group Drives*)
- There is no requirement to have a dedicated network for OS-install and system deployment, this can be shared with the management network. More information can be found on the *Example Configuration* page.

- You may see the terms deployer and lifecycle manager used interchangeably. These are referring to the same nodes in your environment.
- When running the Ansible playbook in this installation guide, if a runbook fails you will see in the error response to use the --limit switch when retrying a playbook. This should be avoided. You can simply re-run any playbook without this switch.
- DVR is not supported with ESX compute.
- When you attach a Cinder volume to the VM running on the ESXi host, the volume will not get detected automatically. Make sure to set the image metadata **vmware_adaptertype=lsiLogicsas** for image before launching the instance. This will help to discover the volume change appropriately.

Before You Start

We have put together a *Pre-Installation Checklist* that should help with the recommended pre-installation tasks.

HPE Helion OpenStack® 5.0: Prerequisites

ESX/vCenter integration is not fully automatic, vCenter administrators are advised of the following responsibilities to ensure secure operation:

1. The VMware administrator is responsible for administration of the vCenter servers and the ESX nodes using the VMware administration tools. These responsibilities include:

   a. Installing and configuring vCenter server
   b. Installing and configuring ESX server and ESX cluster
   c. Installing and configuring shared datastores
   d. Establishing network connectivity between the ESX network and the HPE Helion OpenStack® management network
2. The VMware administration staff is responsible for the review of vCenter logs. These logs are not automatically included in HPE Helion OpenStack® centralized logging.
3. The VMware administrator is responsible for administration of the vCenter servers and the ESX nodes using the VMware administration tools.
4. Logging levels for vCenter should be set appropriately to prevent logging of the password for the HPE Helion OpenStack® message queue.
5. The vCenter cluster and ESX Compute nodes must be appropriately backed up.
6. Backup procedures for vCenter should ensure that the file containing the HPE Helion OpenStack® configuration as part of Nova and Cinder volume services is backed up and the backups are protected appropriately.
7. Since the file containing the HPE Helion OpenStack® message queue password could appear in the swap area of a vCenter server, appropriate controls should be applied to the vCenter cluster to prevent discovery of the password via snooping of the swap area or memory dumps
8. It is recommended to have a common shared storage for all the ESXi hosts in a particular cluster.
9. Ensure that you have enabled HA (High Availability) and DRS (Distributed Resource Scheduler) settings in a cluster configuration before running the installation. DRS/HA is disabled only for OVSvApp. This is done so that it does not move to a different host. If you do not enable DRS/HA prior to installation then you will not be able to disable it only for OVSvApp. As a result DRS/HA can migrate OVSvApp to different host, which will create a network loop.

   Note: No two clusters should have the same name across datacenters in a given vCenter.

10. L3 HA VRRP is enabled by default.

HPE Helion OpenStack® 5.0: Deploy ESX Cloud with OVSvApp

The following diagram depicts the procedure to configure and deploy ESX cloud.
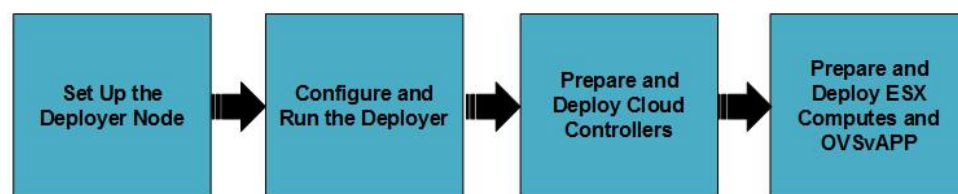
HPE Helion OpenStack® 5.0: Procedure to Deploy ESX Cloud with OVSvApp

Set up the Lifecycle Manager

Installing the Lifecycle Manager

The lifecycle manager will contain the installation scripts and configuration files to deploy your cloud. You can set up the lifecycle manager on a dedicated node or you do so on your first controller node. The default choice is to use the first controller node as the lifecycle manager.

1. Sign in and download the product and signature files at the link below:

    a. *Software Entitlement Portal*

2. You can verify the download was complete via the signature verification process outlined *here*.

3. Boot your lifecycle manager from the ISO contained in the download.

4. Enter "install" to start installation.

> 📝 **Note:** "install" is all lower case

5. Select the language. Note that only the English language selection is currently supported.

6. Select the location.

7. Select the keyboard layout.

8. Select the primary network interface, if prompted:

    a. Assign IP address, subnet mask, and default gateway

9. Create new account:

    a. Enter a username.
    b. Enter a password.
    c. Enter time zone.

Once the initial installation is finished, complete the lifecycle manager setup with these steps:

1. Ensure your lifecycle manager has a valid DNS nameserver specified in `/etc/resolv.conf`.

2. Set the environment variable LC_ALL:

```
export LC_ALL=C
```

> 📝 **Note:** This can be added to `~stack/.bashrc` or `/etc/bash.bashrc`.

At the end of this section you should have a node set up with Linux for HPE Helion on it.

Configure and Run the Lifecycle Manager

> ⚠️ **Important:** It is critical that you don't run any of the commands below as the `root` user or use `sudo`, unless it is stated explicitly in the steps. Run then as the user you just created (or `stack` if you left the default of "stack").

1. Log into your lifecycle manager as the user you created and mount the install media at `/media/cdrom`. It may be necessary to use `wget` or another file transfer method to transfer the install media to the lifecycle manager before completing this step. Here is the command to mount the media:

Example for HPE Helion OpenStack® 5.0

```
sudo mount HelionOpenStack-5.0.iso /media/cdrom
```

2. Unpack the tarball that is in the `/media/cdrom/hos/` directory:

Example for HPE Helion OpenStack 5.0

```
tar xvf /media/cdrom/hos/hos-5.0.0-<timestamp>.tar
```

3. Run the hos-init.bash script which is included in the build:

Example for HPE Helion OpenStack® 5.0

```
~/hos-5.0.0/hos-init.bash
```

You will be prompted to enter an optional SSH passphrase when running `hos-init.bash`. This passphrase is used to protect the key used by Ansible when connecting to its client nodes. If you do not want to use a passphrase then just press return at the prompt.

For automated installation (e.g. CI) it is possible to disable SSH passphrase prompting by setting the `HOS_INIT_AUTO` environment variable before running `hos-init.bash`, like this:

```
export HOS_INIT_AUTO=y
```

If you have protected the SSH key with a passphrase then execute the following commands to avoid having to enter the passphrase on every attempt by Ansible to connect to its client nodes:

```
eval $(ssh-agent)
ssh-add ~/.ssh/id_rsa
```

At the end of this section you should have a local directory structure, as described below:

```
~/helion/                    Top level directory
~/helion/examples/           Directory contains the config input files of the example clouds
~/helion/my_cloud/definition/  Directory contains the config input files
~/helion/my_cloud/config/    Directory contains .j2 files which are symlinks to the ~/helion/hos/ansible directory
~/helion/hos/                Directory contains files used by the installer
```

⚠️ **Warning:** It's important that you do not add any extra files in the `~/helion` directory on your lifecycle manager. This includes temporary save files. Doing so will cause errors during the installation.

❗ **Important:** You should be aware that the files in the `~/helion/my_cloud/config/` directory are symlinks to the `~/helion/hos/ansible/` directory. For example, `~/helion/my_cloud/config/keepalived/defaults.yml` is a symbolic link to `~/helion/hos/ansible/roles/keepalived/defaults/main.yml`

```
ls -al ~/helion/my_cloud/config/keepalived/defaults.yml
lrwxrwxrwx 1 stack stack 55 May 24 20:38 /home/stack/helion/my_cloud/config/keepalived/defaults.yml -> ../../../hos/ansible/roles/keepalived/defaults/main.yml
```

If you are using a tool like `sed` to make edits to files in this directory, you might break the symbolic link and create a new copy of the file. To maintain the link, you will need to force `sed` to follow the link:

```
sed -i --follow-symlinks 's$keepalived_vrrp_offset: 0$keepalived_vrrp_offset: 2$' ~/helion/my_cloud/config/keepalived/defaults.yml
```

Alternatively, you could directly edit the target of the link `~/helion/hos/ansible/roles/keepalived/defaults/main.yml`.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Prepare and Deploy Cloud Controllers

1. Setup your configuration files, as follows:

   a. Copy the example configuration files into the required setup directory and edit them as required:

   ```
   cp -r ~/helion/examples/entry-scale-esx-kvm-vsa/* ~/helion/my_cloud/definition/
   ```

   See a sample set of configuration files in the `~/helion/examples/entry-scale-esx-kvm-vsa` directory. The accompanying README.md file explains the contents of each of the configuration files.

   b. Begin inputting your environment information into the configuration files in the `~/helion/my_cloud/definition` directory.

   📝 **Note:** If you want to use a dedicated deployer node in your ESX deployment, add **eon-client** service-component, to manage vCenter via EON operation from the deployer node, in the `control_plane.yml` file as shown in the following example.

   ```
   clusters:
         - name: cluster0
           cluster-prefix: c0
           server-role: HLM-ROLE
           service-components:
             - lifecycle-manager
             - eon-client
             ...
   ```

2. Commit your cloud deploy configuration to the *local git repo*, as follows:

```
cd ~/helion/hos/ansible
git add -A
```

```
git commit -m "My config or other commit message"
```

📝 **Note:** This step needs to be repeated any time you make changes to your configuration files before you move onto the following steps. See *Using Git for Configuration Management* for more information.

Provision Your Baremetal Nodes

To provision the baremetal nodes in your cloud deployment you can either use the automated operating system installation process provided by HPE Helion OpenStack or you can use the 3rd party installation tooling of your choice. We will outline both methods below:

Using 3rd Party Baremetal Installers

If you do not wish to use the automated operating system installation tooling included with 5.0 then the requirements that have to be met using the installation tooling of your choice are:

- The operating system must be installed via the HPE Linux for HPE Helion OpenStack ISO provided on the *Software Entitlement Portal*.
- Each node must have SSH keys in place that allows the same user from the lifecycle manager node who will be doing the deployment to SSH to each node without a password.
- Passwordless sudo needs to be enabled for the user.
- There should be a LVM logical volume as `/root` on each node.
- If the LVM volume group name for the volume group holding the "root" LVM logical volume is hlm-vg then it will align with the disk input models in the examples.
- Ensure that `openssh-server`, `python`, `python-apt`, and `rsync` are installed.

If you chose this method for installing your baremetal hardware, skip forward to the *Run the Configuration Processor* step.

If you would like to use the automated operating system installation tools provided by 5.0 then complete all of the steps below.

Using the Automated Operating System Installation Provided by HPE Helion OpenStack

Part One: Deploy Cobbler

This phase of the install process takes the baremetal information that was provided in `servers.yml` and installs the Cobbler provisioning tool and loads this information into Cobbler. This sets each node to `netboot-enabled: true` in Cobbler. Each node will be automatically marked as `netboot-enabled: false` when it completes its operating system install successfully. Even if the node tries to PXE boot subsequently, Cobbler will not serve it. This is deliberate so that you can't reimage a live node by accident.

The `cobbler-deploy.yml` playbook prompts for a password - this is the password that will be encrypted and stored in Cobbler, which is associated with the user running the command on the lifecycle manager, that you will use to log in to the nodes via their consoles after install. The username is the same as the user set up in the initial dialogue when installing the lifecycle manager from the iso, and is the same user that is running the cobbler-deploy play.

1. Run the following playbook which confirms that there is iLo connectivity for each of your nodes so that they are accessible to be re-imaged in a later step:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost bm-power-status.yml
```

2. Run the following playbook to deploy Cobbler:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost cobbler-deploy.yml
```

Part Two: Image the Nodes

This phase of the install process goes through a number of distinct steps:

1. Powers down the nodes to be installed
2. Sets the nodes hardware boot order so that the first option is a network boot.
3. Powers on the nodes. (The nodes will then boot from the network and be installed using infrastructure set up in the previous phase)
4. Waits for the nodes to power themselves down (this indicates a success install). This can take some time.
5. Sets the boot order to hard disk and powers on the nodes.
6. Waits for the nodes to be ssh-able and verifies that they have the signature expected.

The reimage command is:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost bm-reimage.yml [-e nodelist=node1,node2,node3]
```

If a nodelist is not specified then the set of nodes in cobbler with `netboot-enabled: True` is selected. The playbook pauses at the start to give you a chance to review the set of nodes that it is targeting and to confirm that it's correct.

You can use the command below which will list all of your nodes with the `netboot-enabled: True` flag set:

```
sudo cobbler system find --netboot-enabled=1
```

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Run the Configuration Processor

Once you have your configuration files setup you need to run the configuration processor to complete your configuration.

When you run the configuration processor you will be prompted for two passwords. Enter the first password to make the configuration processor encrypt its sensitive data, which is comprised of the random inter-service passwords that it generates and the ansible `group_vars` and `host_vars` that it produces for subsequent deploy runs. You will need this password for subsequent ansible deploy and configuration processor runs. If you wish to change an encryption password that you have already used when running the configuration processor then enter the new password at the second prompt, otherwise just press carriage return to bypass this.

Run the configuration processor with this command:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

For automated installation (e.g. CI) you can specify the required passwords on the ansible command line. For example, the command below will disable encryption by the configuration processor

```
ansible-playbook -i hosts/localhost config-processor-run.yml -e encrypt="" -e rekey=""
```

If you receive an error during this step then there is probably an issue with one or more of your configuration files. We recommend that you verify that all of the information in each of your configuration files is correct for your environment and then commit those changes to git using the instructions in the previous section before re-running the configuration processor again.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Deploy the Cloud

1. Use the playbook below to create a deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

2. [OPTIONAL] - Run the `wipe_disks.yml` playbook to ensure all of your partitions on your nodes are completely wiped before continuing with the installation. If you are using fresh machines this step may not be necessary.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts wipe_disks.yml
```

If you have used an encryption password when running the configuration processor use the command below and enter the encryption password when prompted:

```
ansible-playbook -i hosts/verb_hosts wipe_disks.yml --ask-vault-pass
```

3. Run the `site.yml` playbook below:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts site.yml
```

If you have used an encryption password when running the configuration processor use the command below and enter the encryption password when prompted:

```
ansible-playbook -i hosts/verb_hosts site.yml --ask-vault-pass
```

> **Note:** The step above runs `osconfig` to configure the cloud and `hlm-deploy` to deploy the cloud. Therefore, this step may run for a while, perhaps 45 minutes or more, depending on the number of nodes in your environment.

4. Verify that the network is working correctly. Ping each IP in the `/etc/hosts` file from one of the controller nodes.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

HPE Helion OpenStack® 5.0: Prepare and Deploy ESX Computes and OVSvAPPs

The following sections describe the procedure to install and configure ESX compute and OVSvAPPs on vCenter.

Preparation for ESX Cloud Deployment

This section describes the procedures to prepare and deploy the ESX computes and OVSvAPPs for deployment.

1. Login to the lifecycle manager.
2. Source `service.osrc`.
3. *Register a vCenter Server*
4. *Activate Clusters*

   a. *Modify the Volume Configuration File*
   b. *Commit your Cloud Definition*
   c. *Deploy ESX Compute Proxy and OVSvApps*

Manage vCenters and Clusters

The following section describes the detailed procedure on managing the vCenters and clusters.

Register a vCenter Server

Note:

- If ESX related roles are added in the input model after the controller cluster is up, you must run `hlm-reconfigure.yml` before adding the eon resource-manager.
- Make sure to provide single quotes (' ') or backslash (\) for the special characters (& ! ; " ' () | \ >) that are used when setting a password for vCenter registration . For example: '2the!Moon' or 2the\!Moon. You can use either of the format to set the vCenter password.

vCenter provides centralized management of virtual host and virtual machines from a single console.

1. Add a vCenter using EON python client.

```
# eon resource-manager-add [--name <RESOURCE_MANAGER_NAME>] --ip-address <RESOURCE_MANAGER_IP_ADDR> --username <RESOURCE_MANAGER_USERNAME> --password
  <RESOURCE_MANAGER_PASSWORD> [--port <RESOURCE_MANAGER_PORT>] --type <RESOURCE_MANAGER_TYPE>
```

where:

- Resource Manager Name - the identical name of the vCenter server.
- Resource Manager IP address - the IP address of the vCenter server.
- Resource Manager Username - the admin privilege username for the vCenter.
- Resource Manager Password - the password for the above username.
- Resource Manager Port - the vCenter server port. By default it is 443.
- Resource Manager Type - specify `vcenter`.

  Important: Please do not change the vCenter Port unless you are certain it is required to do so.

Sample Output:

```
# eon resource-manager-add --name vc01 --ip-address 10.1.200.38 --username administrator@vsphere.local --password init123# --port 443 --type vcenter
+------------+-----------------------------------+
| Property   | Value                             |
+------------+-----------------------------------+
| ID         | BC9DED4E-1639-481D-B190-2B54A2BF5674 |
| IPv4Address | 10.1.200.38                      |
| Name       | vc01                              |
| Password   | <SANITIZED>                       |
| Port       | 443                               |
| State      | registered                        |
| Type       | vcenter                           |
| Username   | administrator@vsphere.local       |
+------------+-----------------------------------+
```

Show vCenter

1. Show vCenter using EON python client.

```
# eon resource-manager-show <RESOURCE_MANAGER_ID>
```

Sample Output:

```
eon resource-manager-show BC9DED4E-1639-481D-B190-2B54A2BF5674
+------------+------------------------------------+
| Property   | Value                              |
+------------+------------------------------------+
| Clusters   | Cluster1, virtClust, Cluster2      |
| Datacenters| DC1, DC2                           |
| ID         | BC9DED4E-1639-481D-B190-2B54A2BF5674 |
| IPv4Address| 10.1.200.38                        |
| Name       | vc01                               |
| Password   | <SANITIZED>                        |
| Port       | 443                                |
| State      | registered                         |
| Type       | vcenter                            |
| Username   | administrator@vsphere.local        |
+------------+------------------------------------+
```

Create and edit an activation template

This involves getting a sample network information template and modifying the details of the template. You will use the template to register the cloud network configuration for the vCenter.

1. Execute the following command to generate a sample activation template:

```
eon get-activation-template [--filename <ACTIVATION_JSON_NAME>] --type <RESOURCE_TYPE>
```

Sample Output:

```
eon get-activation-template --filename activationtemplate.json --type esxcluster
----------------------------------------------------------
Saved the sample network file in /home/user/activationtemplate.json
----------------------------------------------------------
```

2. Change to the `/home/user/` directory:

```
cd /home/user/
```

3. Modify the template (json file) as per your environment. See *Sample activationtemplate.json File for Helion Entry Scale ESX, KVM with VSA Model*.

Activate Clusters

This involves using the activation template to register the cloud network configuration for the vCenter.

This process spawns one compute proxy VM per cluster and one OVSvApp VM per host and configures the networking as defined in the JSON template. This process also updates the input model with the service VM details, executes the ansible playbooks on the new nodes, and performs a post activation check on the service VMs.

1. Activate the cluster for the selected vCenter.

```
# eon resource-activate <RESOURCE_ID> --config-json /home/user/<ACTIVATION_JSON_NAME>
```

📝 **Note:** To retrieve the resource ID, execute `eon resource-list`.

📝 **Note:** Activating the first cluster in a datacenter requires you to provide an activation template JSON. You can provide the same activation template or a new activation template to activate the subsequent clusters.

> **Note:** Minimum disk space required for a cluster activation is (number of hosts in that cluster +1)* 45 GB.

2. Execute the following command to view the status of the cluster:

```
eon resource-list
+------------------------------------+----------+-------------+------------------------------------+------------+-------+------------+-----------+
| ID                                 | Name     | Moid        | Resource Manager ID                | IP Address | Port  | Type       | State     |
+------------------------------------+----------+-------------+------------------------------------+------------+-------+------------+-----------+
| 1228fce5-df5d-445c-834e-ae633ac7e426 | Cluster2 | domain-c184 | BC9DED4E-1639-481D-B190-2B54A2BF5674 | UNSET      | UNSET | esxcluster | imported  |
| 469710f6-e9f2-48a4-aace-1f00cbd60487 | virtClust | domain-c943 | BC9DED4E-1639-481D-B190-2B54A2BF5674 | UNSET      | UNSET | esxcluster | activated |
| a3003a32-6e3a-4d89-a072-ec64a4247fb0 | Cluster1 | domain-c21  | BC9DED4E-1639-481D-B190-2B54A2BF5674 | UNSET      | UNSET | esxcluster | imported  |
+------------------------------------+----------+-------------+------------------------------------+------------+-------+------------+-----------+
```

When the state is `activated`, the input model is updated with the service VM details, a git commit has been performed, the required playbooks and the post activation checks are completed successfully.

> **Note:** Whenever an ESX compute is activated or deactivated, you must run the following playbook to reflect the changes in opsconsole.

```
cd scratch/ansible/next/hos/ansible/
ansible-playbook -i hosts/verb_hosts ops-console-reconfigure.yml
```

Modify the Volume Configuration File

Once the cluster is activated you must configure the volume.

Perform the following steps to modify the volume configuration files:

1. Change the directory. The `cinder.conf.j2` is present in following directories :

```
cd /home/stack/helion/hos/ansible/roles/_CND-CMN/templates
```

OR

```
cd /home/stack/helion/my_cloud/config/cinder
```

It is recommended to modify the `cinder.conf.j2` available in `/home/stack/helion/my_cloud/config/cinder`

2. Modify the `cinder.conf.j2` as follows:

```
# Configure the enabled backends
enabled_backends=<unique-section-name>

# Start of section for VMDK block storage
#
# If you have configured VMDK Block storage for cinder you must
# uncomment this section, and replace all strings in angle brackets
# with the correct values for vCenter you have configured. You
# must also add the section name to the list of values in the
# 'enabled_backends' variable above. You must provide unique section
# each time you configure a new backend.

#[<unique-section-name>]
#vmware_api_retry_count = 10
#vmware_tmp_dir = /tmp
#vmware_image_transfer_timeout_secs = 7200
#vmware_task_poll_interval = 0.5
#vmware_max_objects_retrieval = 100
#vmware_volume_folder = cinder-volumes
#volume_driver = cinder.volume.drivers.vmware.vmdk.VMwareVcVmdkDriver
#vmware_host_ip = <ip_address_of_vcenter>
#vmware_host_username = <vcenter_username>
#vmware_host_password = <password>
#
```

```
#volume_backend_name = <vmdk-backend-name>
#
# End of section for VMDK block storage
```

Commit your Cloud Definition

1. Add the cloud deployment definition to git :

```
cd /home/stack/helion/hos/ansible;
git add -A;
git commit -m 'Adding ESX Configurations or other commit message';
```

2. Prepare your environment for deployment:

```
ansible-playbook -i hosts/localhost config-processor-run.yml;
ansible-playbook -i hosts/localhost ready-deployment.yml;
cd /home/stack/scratch/ansible/next/hos/ansible;
```

Configuring VMDK block storage

Execute the following command to configure VMDK block storage:

```
ansible-playbook -i hosts/verb_hosts cinder-reconfigure.yml
```

If there are more backends (like VSA) defined, you must create and use specific volume type to ensure that volume is created in ESX, as shown below:

```
# cinder type-create "ESX VMDK Storage"
...
# cinder type-key "ESX VMDK Storage" set volume_backend_name=<name of VMDK backend selected during installation>
...
# cinder create --volume-type "ESX VMDK Storage" 1
...
```

Validate the block storage

You can validate that the VMDK block storage is added to the cloud successfully using the following command:

```
# cinder service-list
```

```
+------------------+-------------------------+-------+----------+-------+----------------------------+-----------------+
|      Binary      |          Host           | Zone  |  Status  | State |         Updated_at         | Disabled Reason |
+------------------+-------------------------+-------+----------+-------+----------------------------+-----------------+
|   cinder-backup  |     ha-volume-manager   | nova  | enabled  |  up   | 2016-06-14T08:44:51.000000 |        -        |
| cinder-scheduler |     ha-volume-manager   | nova  | enabled  | down  | 2016-06-13T08:47:07.000000 |        -        |
| cinder-scheduler |     ha-volume-manager   | nova  | enabled  |  up   | 2016-06-14T08:44:57.000000 |        -        |
|   cinder-volume  |     ha-volume-manager   | nova  | enabled  | down  | 2016-06-13T07:36:41.000000 |        -        |
|   cinder-volume  | ha-volume-manager@vmdk1 | nova  | enabled  |  up   | 2016-06-14T08:44:55.000000 |        -        |
|   cinder-volume  | ha-volume-manager@vsa1  | nova  | enabled  |  up   | 2016-06-14T08:44:50.000000 |        -        |
+------------------+-------------------------+-------+----------+-------+----------------------------+-----------------+
```

Validate the compute

You can validate that the ESX compute cluster is added to the cloud successfully using the following command:

```
#  nova service-list
```

```
+-----+----------------+-----------------------------+---------+--------+-------+----------------------------+-----------------+
| Id  | Binary         | Host                        | Zone    | Status | State | Updated_at                 | Disabled Reason |
```

```
+-----+---------------+------------------------------+----------+---------+------+---------------------------+----------------+
| 3   | nova-conductor  | esxhos-joh-core-m1-mgmt        | internal | enabled | up   | 2016-03-30T12:57:47.000000 | -              |
| 63  | nova-scheduler  | esxhos-joh-core-m1-mgmt        | internal | enabled | up   | 2016-03-30T12:57:43.000000 | -              |
| 66  | nova-conductor  | esxhos-joh-core-m2-mgmt        | internal | enabled | up   | 2016-03-30T12:57:48.000000 | -              |
| 111 | nova-conductor  | esxhos-joh-core-m3-mgmt        | internal | enabled | up   | 2016-03-30T12:57:41.000000 | -              |
| 129 | nova-scheduler  | esxhos-joh-core-m3-mgmt        | internal | enabled | up   | 2016-03-30T12:57:41.000000 | -              |
| 132 | nova-consoleauth| esxhos-joh-core-m1-mgmt        | internal | enabled | up   | 2016-03-30T12:57:44.000000 | -              |
| 135 | nova-scheduler  | esxhos-joh-core-m2-mgmt        | internal | enabled | up   | 2016-03-30T12:57:47.000000 | -              |
| 138 | nova-compute    | esxhos-joh-esx-comp0001-mgmt   | nova     | enabled | up   | 2016-03-30T12:57:41.000000 | -              |
+-----+---------------+------------------------------+----------+---------+------+---------------------------+----------------+
```

You can validate the hypervisor list using the following command:

```
# nova hypervisor-list
```

```
+----+--------------------------------------------+-------+---------+
| ID | Hypervisor hostname                        | State | Status  |
+----+--------------------------------------------+-------+---------+
| 9  | domain-c40.9FDCFA66-6677-42A1-83FF-16DC32448021 | up  | enabled |
+----+--------------------------------------------+-------+---------+
```

Validate the neutron

You can validate that the networking of the ESX cluster using the following command:

```
# neutron agent-list
```

```
+--------------------------------------+------------------+----------------------------------+-------+----------------+-------------------------+
| id                                   | agent_type       | host                             | alive | admin_state_up | binary                  |
+--------------------------------------+------------------+----------------------------------+-------+----------------+-------------------------+
| 097bdbc3-108c-41ca-8b52-9d249f65077f | Metadata agent   | esxhos-joh-core-m3-mgmt          | :-)   | True           | neutron-metadata-agent  |
| 2b255256-9505-489e-93bf-0d37f7ff83e4 | DHCP agent       | esxhos-joh-core-m3-mgmt          | :-)   | True           | neutron-dhcp-agent      |
| 43843bcb-f929-4a42-a1e1-207ff97dc09e | OVSvApp Agent    | esxhos-joh-esx-ovsvapp0002-mgmt  | :-)   | True           | ovsvapp-agent           |
| 4f0be657-fe9e-4bab-bef8-d8e688b6573e | L3 agent         | esxhos-joh-core-m3-mgmt          | :-)   | True           | neutron-vpn-agent       |
| 6db7d585-604e-4205-b29c-b9684bfb0cb2 | OVSvApp Agent    | esxhos-joh-esx-ovsvapp0001-mgmt  | :-)   | True           | ovsvapp-agent           |
| 700f1a18-d237-4605-bf54-145951bd12db | DHCP agent       | esxhos-joh-core-m1-mgmt          | :-)   | True           | neutron-dhcp-agent      |
| a723fb23-2c87-4f69-b272-643df870f0b6 | DHCP agent       | esxhos-joh-core-m2-mgmt          | :-)   | True           | neutron-dhcp-agent      |
| bd512ed5-9f39-4ad6-a505-148f58cf2e64 | L3 agent         | esxhos-joh-core-m1-mgmt          | :-)   | True           | neutron-vpn-agent       |
| e3d4b8be-077e-4503-bed2-031871f3829e | Open vSwitch agent | esxhos-joh-core-m2-mgmt        | :-)   | True           | neutron-openvswitch-agent |
| e5d75917-1d26-4443-99f4-45d8c574e3f0 | Open vSwitch agent | esxhos-joh-core-m1-mgmt        | :-)   | True           | neutron-openvswitch-agent |
| e6a40540-adad-4ee7-b194-db18ac7287bd | Metadata agent   | esxhos-joh-core-m1-mgmt          | :-)   | True           | neutron-metadata-agent  |
| e734a20e-159c-450b-9fb8-dc46824ef12e | Metadata agent   | esxhos-joh-core-m2-mgmt          | :-)   | True           | neutron-metadata-agent  |
| f019c268-a6bd-4f66-8a46-6762c720a5bd | L3 agent         | esxhos-joh-core-m2-mgmt          | :-)   | True           | neutron-vpn-agent       |
| fc83c8dd-ead4-4ad5-aed3-95ab41e567bc | Open vSwitch agent | esxhos-joh-core-m3-mgmt        | :-)   | True           | neutron-openvswitch-agent |
+--------------------------------------+------------------+----------------------------------+-------+----------------+-------------------------+
```

HPE Helion OpenStack® 5.0: Sample activationtemplate.json File for ESX Compute
Sample activation template file and its parameters.

The process for installing the Helion Entry Scale ESX, KVM with VSA Model requires creating a sample network information template and modifying the details of the template. You will use the template to register the cloud network configuration for the vCenter. The template is in the JSON format.

Execute the following command to get the activation template.

```
eon get-activation-template [--filename <Activate JSON>] --type <Type of the resource>
```

For example:

```
eon get-activation-template --filename activationtemplate.json --type esxcluster
```

```
------------------------------------------------------------
Saved the sample network file in /home/user/activationtemplate.json
------------------------------------------------------------
```

EON supports two network driver for ESX deployment. Based on your deployment of ESX cloud, the template (activationtemplate.json) is automatically pre-configured specific to the driver

1. OVSvApp driver: For ESX with OVSvApp driver (default)
2. NoOp driver: For ESX with DCN driver

A sample of `activationtemplate.json` file for OVSvApp driver (default) and NoOp are shown as follows.

⚠️ **Warning:** Update the sample template as per your environment.

• **OVSvApp template**

A Sample json file for OVSvApp driver where networking solution is provided by OVSvApp appliances (default).

```
{
 "input_model": {
  "server_group": "RACK1"
 },
 "network_properties": {
  "switches": [
   {
    "type": "dvSwitch",
    "name": "MGMT-DVS",
    "physical_nics": "vmnic1",
    "mtu": "1500"
   },
   {
    "type": "dvSwitch",
    "name": "TRUNK-DVS",
    "physical_nics": "",
    "mtu": "1500"
   }
  ],
  "vm_config": [
   {
    "nics": [
     {
      "device": "eth0",
      "type": "vmxnet3",
      "pci_id": "",
      "portGroup": "ESX-CONF"
     },
     {
      "device": "eth1",
      "type": "vmxnet3",
      "pci_id": "",
      "portGroup": "MGMT"
     }
    ],
    "server_role": "ESX-COMPUTE-ROLE",
    "memory_in_mb": "4096",
    "cpu": "4"
   },
   {
    "nics": [
     {
      "device": "eth0",
      "type": "vmxnet3",
      "pci_id": "",
```

```
      "portGroup": "ESX-CONF"
    },
    {
      "device": "eth1",
      "type": "vmxnet3",
      "pci_id": "",
      "portGroup": "MGMT"
    },
    {
      "device": "eth2",
      "type": "vmxnet3",
      "pci_id": "",
      "portGroup": "GUEST"
    },
    {
      "device": "eth3",
      "type": "vmxnet3",
      "pci_id": "",
      "portGroup": "TRUNK"
    }
  ],
  "server_role": "OVSVAPP-ROLE",
  "memory_in_mb": "4096",
  "cpu": "4"
 }
],
"portGroups": [
 {
  "nic_teaming": {
   "network_failover_detection": "1",
   "notify_switches": "yes",
   "load_balancing": "1",
   "active_nics": "vmnic1"
  },
  "vlan_type": "trunk",
  "vlan": "33",
  "name": "ESX-CONF",
  "switchName": "MGMT-DVS"
 },
 {
  "nic_teaming": {
   "network_failover_detection": "1",
   "notify_switches": "yes",
   "load_balancing": "1",
   "active_nics": "vmnic1"
  },
  "vlan_type": "none",
  "vlan": "",
  "name": "MGMT",
  "switchName": "MGMT-DVS"
 },
 {
  "vlan": "34",
  "name": "GUEST",
  "vlan_type": "trunk",
  "switchName": "MGMT-DVS",
  "nic_teaming": {
   "network_failover_detection": "1",
   "notify_switches": "yes",
   "load_balancing": "1",
   "active_nics": "vmnic1"
  },
  "cloud_network_type": "vxlan"
```

```
  },
  {
    "vlan_type": "trunk",
    "vlan": "1-4094",
    "name": "TRUNK",
    "switchName": "TRUNK-DVS"
  }
 ],
 "template_info": {
  "upload_to_cluster": false
 },
 "esx_conf_net": {
  "start_ip": "",
  "cidr": "10.20.18.0/23",
  "end_ip": "",
  "gateway": "10.20.18.1",
  "portGroup": "ESX-CONF"
 }
 }
}
```

- **NoOp template**

A sample json file for NoOp driver where networking solution is provided by DCN.

```
{
 "input_model": {
  "server_group": "RACK1"
 },
 "network_properties": {
  "switches": [
   {
    "type": "dvSwitch",
    "name": "MGMT-DVS",
    "physical_nics": "vmnic1",
    "mtu": "1500"
   }
  ],
  "vm_config": [
   {
    "nics": [
     {
      "device": "eth0",
      "type": "vmxnet3",
      "pci_id": "",
      "portGroup": "ESX-CONF"
     },
     {
      "device": "eth1",
      "type": "vmxnet3",
      "pci_id": "",
      "portGroup": "MGMT"
     }
    ],
    "server_role": "ESX-COMPUTE-ROLE",
    "memory_in_mb": "4096",
    "cpu": "4"
   }
  ],
  "portGroups": [
   {
    "nic_teaming": {
     "network_failover_detection": "1",
```

```
      "notify_switches": "yes",
      "load_balancing": "1",
      "active_nics": "vmnic1"
     },
     "vlan_type": "trunk",
     "vlan": "33",
     "name": "ESX-CONF",
     "switchName": "MGMT-DVS"
    },
    {
     "nic_teaming": {
      "network_failover_detection": "1",
      "notify_switches": "yes",
      "load_balancing": "1",
      "active_nics": "vmnic1"
     },
     "vlan_type": "none",
     "vlan": "",
     "name": "MGMT",
     "switchName": "MGMT-DVS"
    }
   ],
   "template_info": {
    "upload_to_cluster": false
   },
   "esx_conf_net": {
    "start_ip": "",
    "cidr": "10.20.18.0/23",
    "end_ip": "",
    "gateway": "10.20.18.1",
    "portGroup": "ESX-CONF"
   }
  }
 }
}
```

The above template consists of the following sections:

- Input model

  - Server group
- Network properties

  - Port Groups to be created or reused
  - Distributed Virtual Switches to be created or reused
  - Template information
  - Virtual machine configuration (RAM, CPU and network interfaces)
  - Ansbile network

The main purpose of the activation template is to create an environment to host nova-compute-proxy (in case of OVSvApp driver) or only nova-compute-proxy (in case of NoOp driver). The environment here refers to the required virtual networking (Distributed Virtual Switch (DVS) and port groups), VMs (RAM, CPU, NIC and associated server role) going to host the Neutron and Nova services.

- **Input model**

  It describes the configuration of each entities in the input model.

  In the following example we have specified the value of server_group as RACK1.

```
 {
  "input_model": {
   "server_group": "RACK1"
  },
```

- **Server group**

This key takes values specified in `server_groups.yml` file defined in the input model. Default server groups defined in the input model are "RACK1", "RACK2", and "RACK3" based on different network definitions. This implies that all the member hosts in the cluster are part of the same RACK.

- **Network properties**

  You must define and configure the following network properties.

  - **Port Groups**

    EON creates the portGroups (PGs) if the specified PG is unavailable. The portGroups key contains a JSON list of one or more entries.

    You can set the tenant network type as VLAN or VXLAN or both VLAN and VXLAN. The `cloud_network_type` must be set as per the portGroup.

    ⚠ **Important:** Do not modify the VLAN range for trunk PG. The value should remain as 1-4094

    In the following example there are three portGroups with a same tenant network type with an active NIC. Also, if there is any failover in the switches then it is notified as the value is set to **yes** with the failover detection set to 1.

```
"portGroups": [
    {
     "nic_teaming": {
      "network_failover_detection": "1",
      "notify_switches": "yes",
      "load_balancing": "1",
      "active_nics": "vmnic1"
     },
     "vlan_type": "trunk",
     "vlan": "33",
     "name": "ESX-CONF",
     "switchName": "MGMT-DVS"
    },
    {
     "nic_teaming": {
      "network_failover_detection": "1",
      "notify_switches": "yes",
      "load_balancing": "1",
      "active_nics": "vmnic1"
     },
     "vlan_type": "trunk",
     "vlan": "3307",
     "name": "MGMT",
     "switchName": "MGMT-DVS"
    },
    {
     "vlan": "3285",
     "name": "GUEST",
     "vlan_type": "trunk",
     "switchName": "MGMT-DVS",
     "nic_teaming": {
      "network_failover_detection": "1",
      "notify_switches": "yes",
      "load_balancing": "1",
      "active_nics": "vmnic1"
     },
     "cloud_network_type": "vxlan"
    },
```

In the following example a portGroup **GUEST** has an active NIC and **both** tenant network types. Also, if there is any failover in the switches then it is notified as the value is set to **yes** with the failover detection set to 1. you wnat only tenant vlan u can define u can configure this.

```
"portGroups": [
    {
     "name": "GUEST",
     "vlan_type": "trunk",
```

```
        "vlan": "3285,3286-3290",
        "switchName": "MGMT-DVS",
        "nic_teaming": {
        "network_failover_detection": "1",
        "notify_switches": "yes",
        "load_balancing": "1",
        "active_nics": "vmnic1"
        }
        "cloud_network_type": "vxlan, vlan"
    }
]
```

In the following example a portGroup **GUEST** has an active NIC with vlan as a tenant network type. Also, if there is any failover in the switches then it is notified as the value is set to **yes** with the failover detection set to 1.

```
"portGroups": [
    {
        "name": "GUEST",
        "vlan_type": "trunk",
        "vlan": "3285-3290",
        "switchName": "MGMT-DVS",
        "nic_teaming": {
        "network_failover_detection": "1",
        "notify_switches": "yes",
        "load_balancing": "1",
        "active_nics": "vmnic1"
        }
        "cloud_network_type": "vlan"
    }
]
```

In the following example a portGroup **GUEST** has an active NIC with vxlan as a tenant network types. Also, if there is any failover in the switches then it is notified as the value is set to **yes** with the failover detection set to 1.

```
"portGroups": [
    {
        "name": "GUEST",
        "vlan_type": "trunk",
        "vlan": "3285",
        "switchName": "MGMT-DVS",
        "nic_teaming": {
        "network_failover_detection": "1",
        "notify_switches": "yes",
        "load_balancing": "1",
        "active_nics": "vmnic1"
        }
        "cloud_network_type": "vxlan"
    }
]
```

The following table provides the list of parameters that must be configured.

Parameter	Description
name (String)	Unique name for the portGroup to be created or used.
vlan_type (String)	The following are acceptable VLAN types: • none = Untagged port • vlan = VLAN port and only the VLAN ID mentioned in "VLAN" key is allowed • trunk = Port allows multiple VLAN networks as mentioned by "VLAN" Key
vlan	Uplinks for the switch (comma separated string). The vlan is configured on the NIC. For example, "vlan" : "33" or "vlan" : "0,33-35" .

Parameter	Description
	Example: Two uplinks -- "vmnic0, vmnic1". Single uplink – 'vmnic0'. No uplinks – ''.
switchName (String)	Name of the Distributed Virtual Switch (DVS) under which this port-group should be created.
load_balancing (Int)	Specify how to choose an uplink. Acceptable values are 1 to 5. 1. Route based on the originating virtual port 2. Route based on IP hash 3. Route based on source MAC hash 4. Route based on physical NIC load 5. Use explicit failover order
notify_switches (Boolean)	Select whether to notify switches in the case of failover.
network_failover_detection (Int)	Specify the method to use for failover detection. • Link Status only • Beacon Probing
active_nics	(Comma separated string) Specify the uplinks to be used.
cloud_network_type (Srting)	(Optional) Specify the tenant network type ("vlan" or "vxlan", or both ("vlan", "vxlan"). In case of VLAN, the security policy for the port group is modified to enable 'Promiscuous Mode' and 'Forged transmits'.

Configuration of PG carrying Data in the single NIC VXLAN model.

```
+--------------------+----------------------------+
|    Input model     |      Net_conf.json Vxlan   |
+--------------------+----------------------------+
| tagged-vlan: true  | "vlan": "33",              |
| vlanid: 33         | "vlan_type":"trunk",       |
+--------------------+----------------------------+
| tagged-vlan: false | "vlan": "",                |
| vlanid:            | "vlan_type":"none",        |
+--------------------+----------------------------+
| tagged-vlan: false | If the physical uplink     |
| vlanid: 33         | switch is tagged to        |
|                    | particular vlan            |
|                    | "vlan": "33",              |
|                    | "vlan_type":"vlan",        |
+--------------------+----------------------------+
```

Configuration of PG carrying Data in the single NIC VLAN model.

```
+------------------------+----------------------------+
|    Input model         | Net_conf.json Vlan         |
+------------------------+----------------------------+
| tagged-vlan: false     | "vlan": "0, 1231-1331",    |
| vlanid:                |                            |
| tenant-vlan-id:1231-1331 | "vlan_type":"trunk",     |
+------------------------+----------------------------+
```

• **Distributed Virtual Switches**

EON creates the virtual switch if the specified switch does not exists in the vCenter. User needs to provide the same kind of input required in creating one from the vSphere web-client.

When 2 networks (PXE & CLM) are using a single dvSwitch(DVS-MGMT) and their MTU setting are different, the MTU setting of the network which have higher value should be assigned to the dvSwitch(DVS-MGMT) in the ESX activation.json.

OVSvApp: A minimum of two switch entries are required. One for OVSvApp trunk DVS with no physical_nics and other for management switch with physical_nics. In the following example we have created two DVS with a name MGMT-DVS (with vmnic1 as the uplink) and TRUNK-DVS (with no uplink)

In the following example we have created two DVS with a name MGMT-DVS and TRUNK-DVS and used `vmnic1` as the uplink for the switch.

```
"switches": [
    {
     "type": "dvSwitch",
     "name": "MGMT-DVS",
     "physical_nics": "vmnic1",
     "mtu": "1500"
    },
    {
     "type": "dvSwitch",
     "name": "TRUNK-DVS",
     "physical_nics": "",
     "mtu": "1500"
    }
```

NoOp: One management switch entry is required with physical_nics

In the following example we have created a switch with a name MGMT-DVS and used `vmnic1` as the uplink for the switch.

```
"switches": [
    {
     "type": "dvSwitch",
     "name": "MGMT-DVS",
     "physical_nics": "vmnic1",
     "mtu": "1500"
    },
 ]
```

The following table provides the list of parameters that must be configured.

Parameter	Description
type	The category of switch. It can be Virtual Standard Switch (VSS) or Distributed Virtual Switch (DVS). The supported type of switch is DVS.
name (String)	Unique name for the switch.
physical_nics	Uplinks for the switch (comma separated string). Example: Two uplinks -- "vmnic0, vmnic1". Single uplink – 'vmnic0'. No uplinks – " ".
MTU (Optional)	Maximum Transmission Unit (MTU) packet size. Provide "mtu" value within the range of 1500 to 9000.

• **Template information**

This key takes a dedicated boolen value (true or false) for the parameter `upload_to_cluster`. If the value is set to true, every activation of a cluster will upload the OVA template to the cluster. By default, it is set to false. In this case, the OVA is uploaded to the first cluster which is being activated. The format of the template name is **hlm-shell-vm-<hlm_version>-<dc-name>**, user given "template_name" will not be considered.

```
"template_info": {"upload_to_cluster": false}
```

• **Virtual machine configuration**

EON creates one `nova-compute-proxy` virtual machine and 'N' number of OVSvApp virtual machines where N represents the number of ESXi hosts participating in the cluster. In other words, a OVSvApp VM will be deployed per host.

In the following example we have mentioned the server role as **ESX-Compute-Role** and **OVSVAPP-ROLE**. These roles are defined in the `server_role.yml` file of the input model. We require a template to create a virtual machine. 4 vCPU is configured along with 4096 MB memory. The name of the network interfaces are eth0, eth1, eth2, and eth3 and interface model is vmxnet3.

```
"vm_config": [
    {
     "nics": [
        {
         "device": "eth0",
         "type": "vmxnet3",
```

```
          "pci_id": "",
          "portGroup": "ESX-CONF"
        },
        {
          "device": "eth1",
          "type": "vmxnet3",
          "pci_id": "",
          "portGroup": "MGMT"
        }
      ],
      "server_role": "ESX-COMPUTE-ROLE",
      "memory_in_mb": "4096",
      "cpu": "4"
    },
    {
      "nics": [
        {
          "device": "eth0",
          "type": "vmxnet3",
          "pci_id": "",
          "portGroup": "ESX-CONF"
        },
        {
          "device": "eth1",
          "type": "vmxnet3",
          "pci_id": "",
          "portGroup": "MGMT"
        },
        {
          "device": "eth2",
          "type": "vmxnet3",
          "pci_id": "",
          "portGroup": "GUEST"
        },
        {
          "device": "eth3",
          "type": "vmxnet3",
          "pci_id": "",
          "portGroup": "TRUNK"
        }
      ],
      "server_role": "OVSVAPP-ROLE",
      "memory_in_mb": "4096",
      "cpu": "4"
    }
  ],
```

EON creates only one `nova-compute-proxy` virtual machine for NoOp driver.

```
"vm_config": [
  {
    "nics": [
      {
        "device": "eth0",
        "type": "vmxnet3",
        "pci_id": "",
        "portGroup": "ESX-CONF"
      },
      {
        "device": "eth1",
        "type": "vmxnet3",
        "pci_id": "",
        "portGroup": "MGMT"
```

```
    }
  ],
  "server_role": "ESX-COMPUTE-ROLE",
  "memory_in_mb": "4096",
  "cpu": "4"
}
```

The following table provides the list of parameters that must be configured.

Parameter	Description
server_role (String)	Specify the role associated to this appliance in the cloud input model.
CPU (Int)	Number of vCPUs to be configured.
memory_in_mb (Int)	Amount of memory to be configured in MB.
Device (String)	Network interface name. Generally in the format eth0, eth1… ethX.
portGroup (String)	Name of the virtual network (port-group) to be attached to the interface .
type (String)	Type of interface model. Supported models are 'vmxnet3' and 'e1000'.
pci_id	Device ID of the ESXi host when passthrough for a Network Device on the host is enabled.
nics	List of network interfaces to be connected to the virtual machine.

- **Ansbile network**

In ESX cloud, you require a separate network for running ansible commands instead of sharing the management network. IPAM is managed by EON and assigns IP addresses to virtual machines from this network.

The following example provides a network configuration that is defined under `esx_conf_net`.

```
"esx_conf_net": {
    "start_ip": "",
    "cidr": "10.20.18.0/23",
    "end_ip": "",
    "gateway": "10.20.18.1",
    "portGroup": "ESX-CONF"
},
```

The following table provides the list of parameters that must be configured.

Parameter	Description
portGroup (String)	Name of the network interface for which IP address needs to be assigned.
CIDR (String)	Classless Inter-Domain Routing of ESX Configuration Network.
Start_ip (String)	First IPv4 allocation.
end_ip (String)	Last IPv4 allocation.
Gateway (String)	IPv4 address of the network gateway.

Return to installing *Helion Entry Scale ESX, KVM with VSA Model*.

Installing Baremetal (Ironic)

Overview

Bare Metal as a Service is enabled in this release for deployment of nova instances on bare metal nodes using flat networking. HPE DL and SL line of servers are supported.

HPE Helion OpenStack® 5.0: Installation for HPE Helion Entry-scale Cloud with Ironic Flat Network

This page describes the installation step requirements for the HPE Helion Entry-scale Cloud with Ironic Flat Network.

Important Notes

- If you are looking for information about when to use the GUI installer and when to use the CLI, see the *Installation Overview*.
- Review the *recommended minimum hardware requirements* that we have listed.
- Review the *Release Notes* to make yourself aware of any known issues and limitations.
- If you run into issues during installation, we have put together a list of *Installation Troubleshooting Steps* you can reference.
- Make sure all disks on the system(s) are wiped before you begin the install. (For Swift, refer to *Swift Requirements for Device Group Drives*)
- There is no requirement to have a dedicated network for OS-install and system deployment, this can be shared with the management network. More information can be found on the *Example Configuration* page.
- You may see the terms deployer and lifecycle manager used interchangeably. These are referring to the same nodes in your environment.
- When running the Ansible playbook in this installation guide, if a runbook fails you will see in the error response to use the `--limit` switch when retrying a playbook. This should be avoided. You can simply re-run any playbook without this switch.

Before You Start

We have put together a *Pre-Installation Checklist* that should help with the recommended pre-installation tasks.

Set up the Lifecycle Manager

Installing the Lifecycle Manager

The lifecycle manager will contain the installation scripts and configuration files to deploy your cloud. You can set up the lifecycle manager on a dedicated node or you do so on your first controller node. The default choice is to use the first controller node as the lifecycle manager.

1. Sign in and download the product and signature files at the link below:

   a. *Software Entitlement Portal*
2. You can verify the download was complete via the signature verification process outlined *here*.
3. Boot your lifecycle manager from the ISO contained in the download.
4. Enter "install" to start installation.

   Note: "install" is all lower case

5. Select the language. Note that only the English language selection is currently supported.
6. Select the location.
7. Select the keyboard layout.
8. Select the primary network interface, if prompted:

   a. Assign IP address, subnet mask, and default gateway
9. Create new account:

   a. Enter a username.
   b. Enter a password.
   c. Enter time zone.

Once the initial installation is finished, complete the lifecycle manager setup with these steps:

1. Ensure your lifecycle manager has a valid DNS nameserver specified in `/etc/resolv.conf`.
2. Set the environment variable LC_ALL:

```
export LC_ALL=C
```

   Note: This can be added to `~stack/.bashrc` or `/etc/bash.bashrc`.

At the end of this section you should have a node set up with Linux for HPE Helion on it.

Configure and Run the Lifecycle Manager

⚠ **Important:** It is critical that you don't run any of the commands below as the `root` user or use `sudo`, unless it is stated explicitly in the steps. Run then as the user you just created (or `stack` if you left the default of "stack").

1. Log into your lifecycle manager as the user you created and mount the install media at `/media/cdrom`. It may be necessary to use `wget` or another file transfer method to transfer the install media to the lifecycle manager before completing this step. Here is the command to mount the media:

   Example for HPE Helion OpenStack® 5.0

   ```
   sudo mount HelionOpenStack-5.0.iso /media/cdrom
   ```

2. Unpack the tarball that is in the `/media/cdrom/hos/` directory:

   Example for HPE Helion OpenStack 5.0

   ```
   tar xvf /media/cdrom/hos/hos-5.0.0-<timestamp>.tar
   ```

3. Run the hos-init.bash script which is included in the build:

   Example for HPE Helion OpenStack® 5.0

   ```
   ~/hos-5.0.0/hos-init.bash
   ```

   You will be prompted to enter an optional SSH passphrase when running `hos-init.bash`. This passphrase is used to protect the key used by Ansible when connecting to its client nodes. If you do not want to use a passphrase then just press return at the prompt.

   For automated installation (e.g. CI) it is possible to disable SSH passphrase prompting by setting the `HOS_INIT_AUTO` environment variable before running `hos-init.bash`, like this:

   ```
   export HOS_INIT_AUTO=y
   ```

If you have protected the SSH key with a passphrase then execute the following commands to avoid having to enter the passphrase on every attempt by Ansible to connect to its client nodes:

```
eval $(ssh-agent)
ssh-add ~/.ssh/id_rsa
```

At the end of this section you should have a local directory structure, as described below:

```
~/helion/                      Top level directory
~/helion/examples/             Directory contains the config input files of the example clouds
~/helion/my_cloud/definition/  Directory contains the config input files
~/helion/my_cloud/config/      Directory contains .j2 files which are symlinks to the ~/helion/hos/ansible directory
~/helion/hos/                  Directory contains files used by the installer
```

⚠ **Warning:** It's important that you do not add any extra files in the `~/helion` directory on your lifecycle manager. This includes temporary save files. Doing so will cause errors during the installation.

⚠ **Important:** You should be aware that the files in the `~/helion/my_cloud/config/` directory are symlinks to the `~/helion/hos/ansible/` directory. For example, `~/helion/my_cloud/config/keepalived/defaults.yml` is a symbolic link to `~/helion/hos/ansible/roles/keepalived/defaults/main.yml`

```
ls -al ~/helion/my_cloud/config/keepalived/defaults.yml
lrwxrwxrwx 1 stack stack 55 May 24 20:38 /home/stack/helion/my_cloud/config/keepalived/defaults.yml -> ../../../hos/ansible/roles/keepalived/defaults/main.yml
```

If you are using a tool like `sed` to make edits to files in this directory, you might break the symbolic link and create a new copy of the file. To maintain the link, you will need to force `sed` to follow the link:

```
sed -i --follow-symlinks 's$keepalived_vrrp_offset: 0$keepalived_vrrp_offset: 2$' ~/helion/my_cloud/config/keepalived/defaults.yml
```

Alternatively, you could directly edit the target of the link `~/helion/hos/ansible/roles/keepalived/defaults/main.yml`.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Configure Your Environment

Prior to deploying an operational environment with Ironic, operators need to be aware of the nature of TLS certificate authentication. As pre-built deployment agent ramdisks images are supplied, these ramdisk images will only authenticate known third-party TLS Certificate Authorities in the interest of end-to-end security. As such, uses of self-signed certificates and private certificate authorities will be unable to leverage ironic without modifying the supplied ramdisk images.

1. Setup your configuration files, as follows:

   a. See the sample sets of configuration files in the `~/helion/examples/` directory. Each set will have an accompanying README.md file that explains the contents of each of the configuration files.

   b. Copy the example configuration files into the required setup directory and edit them to contain the details of your environment:

   ```
   cp -r ~/helion/examples/entry-scale-ironic-flat-network/* ~/helion/my_cloud/definition/
   ```

2. [OPTIONAL] - You can use the `hosencrypt.py` script to encrypt your iLo passwords. This script uses OpenSSL.

   a. Change to the Ansible directory:

   ```
   cd ~/helion/hos/ansible
   ```

   b. Put the encryption key into the following environment variable:

   ```
   export HOS_USER_PASSWORD_ENCRYPT_KEY=<encryption key>
   ```

   c. Run the python script below and follow the instructions. Enter a password that you want to encrypt.

   ```
   ./hosencrypt.py
   ```

   d. Take the string generated and place it in the `"ilo-password"` field in your `~/helion/my_cloud/definition/data/servers.yml` file, remembering to enclose it in quotes.

   e. Repeat the above for each server.

   > **Note:** Before you run any playbooks, remember that you need to export the encryption key in the following environment variable: `export HOS_USER_PASSWORD_ENCRYPT_KEY=<encryption key>`

3. Commit your configuration to the *local git repo*, as follows:

   ```
   cd ~/helion/hos/ansible
   git add -A
   git commit -m "My config or other commit message"
   ```

   > **Important:** This step needs to be repeated any time you make changes to your configuration files before you move onto the following steps. See *Using Git for Configuration Management* for more information.

Provision Your Baremetal Nodes

To provision the baremetal nodes in your cloud deployment you can either use the automated operating system installation process provided by HPE Helion OpenStack or you can use the 3rd party installation tooling of your choice. We will outline both methods below:

Using 3rd Party Baremetal Installers

If you do not wish to use the automated operating system installation tooling included with 5.0 then the requirements that have to be met using the installation tooling of your choice are:

- The operating system must be installed via the HPE Linux for HPE Helion OpenStack ISO provided on the *Software Entitlement Portal*.
- Each node must have SSH keys in place that allows the same user from the lifecycle manager node who will be doing the deployment to SSH to each node without a password.
- Passwordless sudo needs to be enabled for the user.
- There should be a LVM logical volume as `/root` on each node.
- If the LVM volume group name for the volume group holding the "root" LVM logical volume is hlm-vg then it will align with the disk input models in the examples.
- Ensure that `openssh-server`, `python`, `python-apt`, and `rsync` are installed.

If you chose this method for installing your baremetal hardware, skip forward to the *Run the Configuration Processor* step.

If you would like to use the automated operating system installation tools provided by 5.0 then complete all of the steps below.

Using the Automated Operating System Installation Provided by HPE Helion OpenStack

Part One: Deploy Cobbler

This phase of the install process takes the baremetal information that was provided in `servers.yml` and installs the Cobbler provisioning tool and loads this information into Cobbler. This sets each node to `netboot-enabled: true` in Cobbler. Each node will be automatically marked as `netboot-enabled: false` when it completes its operating system install successfully. Even if the node tries to PXE boot subsequently, Cobbler will not serve it. This is deliberate so that you can't reimage a live node by accident.

The `cobbler-deploy.yml` playbook prompts for a password - this is the password that will be encrypted and stored in Cobbler, which is associated with the user running the command on the lifecycle manager, that you will use to log in to the nodes via their consoles after install. The username is the same as the user set up in the initial dialogue when installing the lifecycle manager from the iso, and is the same user that is running the cobbler-deploy play.

1. Run the following playbook which confirms that there is iLo connectivity for each of your nodes so that they are accessible to be re-imaged in a later step:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost bm-power-status.yml
```

2. Run the following playbook to deploy Cobbler:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost cobbler-deploy.yml
```

Part Two: Image the Nodes

This phase of the install process goes through a number of distinct steps:

1. Powers down the nodes to be installed
2. Sets the nodes hardware boot order so that the first option is a network boot.
3. Powers on the nodes. (The nodes will then boot from the network and be installed using infrastructure set up in the previous phase)
4. Waits for the nodes to power themselves down (this indicates a success install). This can take some time.
5. Sets the boot order to hard disk and powers on the nodes.
6. Waits for the nodes to be ssh-able and verifies that they have the signature expected.

The reimage command is:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost bm-reimage.yml [-e nodelist=node1,node2,node3]
```

If a nodelist is not specified then the set of nodes in cobbler with `netboot-enabled: True` is selected. The playbook pauses at the start to give you a chance to review the set of nodes that it is targeting and to confirm that it's correct.

You can use the command below which will list all of your nodes with the `netboot-enabled: True` flag set:

```
sudo cobbler system find --netboot-enabled=1
```

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Run the Configuration Processor

Once you have your configuration files setup you need to run the configuration processor to complete your configuration.

When you run the configuration processor you will be prompted for two passwords. Enter the first password to make the configuration processor encrypt its sensitive data, which is comprised of the random inter-service passwords that it generates and the ansible `group_vars` and `host_vars` that it produces for subsequent deploy runs. You will need this password for subsequent ansible deploy and configuration processor runs. If you wish to change an encryption password that you have already used when running the configuration processor then enter the new password at the second prompt, otherwise just press carriage return to bypass this.

Run the configuration processor with this command:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

For automated installation (e.g. CI) you can specify the required passwords on the ansible command line. For example, the command below will disable encryption by the configuration processor

```
ansible-playbook -i hosts/localhost config-processor-run.yml -e encrypt="" -e rekey=""
```

If you receive an error during this step then there is probably an issue with one or more of your configuration files. We recommend that you verify that all of the information in each of your configuration files is correct for your environment and then commit those changes to git using the instructions in the previous section before re-running the configuration processor again.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Deploy the Cloud

1. Use the playbook below to create a deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

2. [OPTIONAL] - Run the `wipe_disks.yml` playbook to ensure all of your partitions on your nodes are completely wiped before continuing with the installation. If you are using fresh machines this step may not be necessary.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts wipe_disks.yml
```

If you have used an encryption password when running the configuration processor use the command below and enter the encryption password when prompted:

```
ansible-playbook -i hosts/verb_hosts wipe_disks.yml --ask-vault-pass
```

3. Run the `site.yml` playbook below:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts site.yml
```

If you have used an encryption password when running the configuration processor use the command below and enter the encryption password when prompted:

```
ansible-playbook -i hosts/verb_hosts site.yml --ask-vault-pass
```

📝 **Note:** The step above runs `osconfig` to configure the cloud and `hlm-deploy` to deploy the cloud. Therefore, this step may run for a while, perhaps 45 minutes or more, depending on the number of nodes in your environment.

4. Verify that the network is working correctly. Ping each IP in the `/etc/hosts` file from one of the controller nodes.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Ironic configuration

Run the `ironic-cloud-configure.yml` playbook below:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts ironic-cloud-configure.yml
```

This step configures ironic flat network, uploads glance images and sets the ironic configuration

To see the images uploaded to glance, run:

```
$ source ~/service.osrc
$ glance image-list
```

This will produce output like the following example, showing three images that have been added by Ironic:

```
+--------------------------------------+------------------------+
| ID                                   | Name                   |
+--------------------------------------+------------------------+
| d4e2a0ff-9575-4bed-ac5e-5130a1553d93 | ir-deploy-iso-HOS3.0    |
| b759a1f0-3b33-4173-a6cb-be5706032124 | ir-deploy-kernel-HOS3.0 |
| ce5f4037-e368-46f2-941f-c01e9072676c | ir-deploy-ramdisk-HOS3.0 |
+--------------------------------------+------------------------+
```

To see the network created by Ironic, run:

```
$ neutron net-list
```

This returns details of the "flat-net" generated by the Ironic install:

```
+--------------------------------------+----------+-------------------------------------------------------+
| id                                   | name     | subnets                                               |
+--------------------------------------+----------+-------------------------------------------------------+
| f9474c06-6660-4a03-9644-f439e2a11010 | flat-net | ca8f8df8-12c8-4e58-b1eb-76844c4de7e8 192.168.245.0/24 |
+--------------------------------------+----------+-------------------------------------------------------+
```

HPE Helion OpenStack® 5.0: Node Configuration

DHCP

Once booted, nodes obtain network configuration via DHCP. If multiple interfaces are to be utilized, a user may wish to pre-build images with settings to execute DHCP on all interfaces. An easy utility to build custom images is the *diskimage-builder* utility which has a *dhcp-all-interfaces* element that can be utilized to initiate DHCP on all interfaces.

The diskimage-builder tool does not support building UEFI images. The following patchset can be used to build UEFI images: *https://review.openstack.org/#/c/287784*

Configuration Drives

⚠️ **Caution:** Configuration Drives are stored unencrypted and should not include any sensitive data.

Users may wish to utilize *Configuration Drives* to store metadata for initial boot setting customization. While Configuration Drives are extremely useful for initial machine configuration, as a general security practice, they should not include any sensitive data. Configuration Drives should only be trusted upon the initial boot of an instance. cloud-init utilizes a lock file for this purpose. Custom instance images should not rely upon the integrity of a Configuration Drive beyond the initial boot of a host as an administrative user with-in a deployed instance can potentially modify a configuration drive once written to disk and released for use.

HPE Helion OpenStack® 5.0: TLS Certificates with Ironic Python Agent (IPA) Images

As part of HPE Helion OpenStack 5.0, Ironic Python Agent, better known as IPA in the OpenStack community, images are supplied and loaded into glance. Two types of image exist. One is a traditional boot ramdisk which is used by the agent_ipmitool, pxe_ipmitool, and pxe_ilo drivers. The other is an ISO image that is supplied as virtual media to the host when using the agent_ilo driver.

As these images are built in advance, they are unaware of any private certificate authorities. Users attempting to utilize self-signed certificates or a private certificate authority will need to inject their signing certificate(s) into the image in order for IPA to be able to boot on a remote node, and ensure that the TLS endpoints being connected to in HPE Helion OpenStack can be trusted. This is not an issue with publicly signed certificates.

As two different types of images exist, below are instructions for disassembling the image ramdisk file or the ISO image. Once this has been done, you will need to re-upload the files to glance, and update any impacted node's driver_info, for example the deploy_ramdisk and ilo_deploy_iso settings that were set when the node was first defined. Respectively, this can be done with the

```
ironic node-update <node> replace driver_info/deploy_ramdisk=<glance_id>
```

or

```
ironic node-update <node> replace driver_info/ilo_deploy_iso=<glance_id>
```

Adding a certificate into a ramdisk image

As root, from a folder where the ramdisk image is present, perform the following steps.

1. Download the ramdisk image to /tmp/ and name the file ironic-deploy.initramfs
2. Change your working directory to /tmp/

```
cd /tmp/
```

3. Create a temporary folder that will hold the temporarily extracted image contents.

```
mkdir new_deploy_ramdisk
```

4. Change your shell working directory to the temporary folder

```
cd /tmp/new_deploy_ramdisk
```

5. Extract the original deployment archive file.

```
zcat /tmp/ironic-deploy.initramfs | cpio --extract --make-directories
```

6. Append your CA certificate to the file located at `/tmp/new_deploy_ramdisk/usr/local/lib/python2.7/dist-packages/requests/cacert.pem`, example below:

```
cat your_ca_certificate.pem >> /tmp/new_deploy_ramdisk/usr/local/lib/python2.7/dist-packages/requests/cacert.pem
```

`your_ca_certificate.pem` is `/etc/ssl/certs/ca-certificates.crt` which can be obtained through

```
cat service.osrc  | grep  CACERT
```

7. Package a new ramdisk file. From with-in the `/tmp/new_deploy_ramdisk` folder, execute the following command:

```
find . | cpio --create --format='newc' |gzip -c -9 > /tmp/updated-ironic-deploy.initramfs
```

Once completed, the new image can be found at `/tmp/updated-ironic-deploy.initramfs`, and will need to be uploaded to Glance. New Glance IDs will need to be recorded for any instances requiring this new image, as noted in the parent paragraph.

Adding a certificate into an ISO image

As root, proceed with the following steps to disassemble the ISO image, and insert a new Certificate Authority or Signing Certificate into the provided image.

1. Download the deployment ISO image from glance and place it in `/tmp/` as `ironic-deploy.iso`.
2. Make a temporary folder to mount the `ironic-deploy.iso` image to in order to extract it's contents.

```
mkdir /mnt/deploy_iso
```

3. Mount the `ironic-deploy.iso` image file to a folder you just create.

```
mount -o loop /tmp/ironic-deploy.iso /mnt/deploy_iso/
```

4. Make a temporary location to stage the new image.

```
mkdir /tmp/new_deploy_iso
```

5. Copy the contents from the previous image into the new temporary staging location.

```
cp -ra /mnt/deploy_iso/. /tmp/new_deploy_iso/
```

6. Make a folder to house the ramdisk that will need to be extracted from the ISO image.

```
mkdir /tmp/new_deploy_ramdisk
```

7. Change your working directory to `/tmp/new_deploy_ramdisk`

```
cd /tmp/new_deploy_ramdisk
```

8. Unpack the ramdisk image.

```
zcat /tmp/new_deploy_iso/initrd | cpio --extract --make-directories
```

9. Append your CA certificate to the file located at `/tmp/new_deploy_ramdisk/usr/local/lib/python2.7/dist-packages/requests/cacert.pem`, example below:

```
cat your_ca_certificate.pem >> /tmp/new_deploy_ramdisk/usr/local/lib/python2.7/dist-packages/requests/cacert.pem
```

10. Replace the pre-existing initial ramdisk in the extracted disk image.

```
find . | cpio --create --format='newc' | gzip -c -9 > /tmp/new_deploy_iso/initrd
```

11. Change directory back to the new deployment ISO image folder.

```
cd /tmp/new_deploy_iso
```

12. Ensure that the program `xorriso` is available, you may need to execute `apt-get install xorriso` to install it.

13. Create the new ISO deployment image utilizing the following command.

```
xorriso -as mkisofs -b isolinux/isolinux.bin -c boot.cat -V VMEDIA_BOOT_ISO -r -J -no-emul-boot -boot-load-size 4 -boot-info-table -eltorito-alt-boot --efi-boot boot/grub/
efi.img -isohybrid-gpt-basdat -isohybrid-apm-hfsplus -o /tmp/new_ironic_deploy.iso ./
```

Once completed, the new image can be found at `/tmp/new_ironic_deploy.iso`, and will need to be uploaded to Glance. New Glance IDs will need to be recorded for any instances requiring this new image, as noted in the parent paragraph

HPE Helion OpenStack® 5.0: Provisioning Baremetal Nodes with Flat Network Model

Warning: Providing bare metal resources to an un-trusted third party is not advised as a malicious user can potentially modify hardware firmware.

Important: The steps outlined in *TLS Certificates with Ironic Python Agent (IPA) Images***must** be performed.

Introduction

A number of drivers are available to provision and manage bare-metal machines. The drivers are named based on the deployment mode and the power management interface. HPE Helion OpenStack has been tested with the following drivers:

- agent_ilo
- agent_ipmi
- pxe_ilo
- pxe_ipmi

For HPE servers, `agent_ipmitool` and `agent_ilo` drivers are supported.

Before you start, you should be aware that:

1. Node Cleaning is enabled for all the drivers in HPE Helion OpenStack 5.0.
2. Node parameter settings must have matching flavors in terms of cpus, local_gb, and memory_mb,boot_mode and cpu_arch.
3. It is advisable that nodes enrolled for ipmitool drivers are pre-validated in terms of bios settings, in terms of boot mode, prior to setting capabilities.
4. Network cabling and interface layout should also be pre-validated in any given particular boot mode or configuration that is registered.
5. The use of `agent_` drivers is predicated upon Glance images being backed by a Swift image store, specifically the need for the temporary file access features. Use of Ceph or the filesystem as a Glance back-end image store means that the `agent_` drivers cannot be used.
6. Manual Cleaning (RAID) and Node inspection is supported by ilo drivers (`agent_ilo` and `pxe_ilo`)

Supplied Images

As part of the Helion Entry-scale Ironic Cloud installation, Ironic Python Agent (IPA) images are supplied and loaded into Glance. To see the images that have been loaded, execute the following commands on the deployer node:

```
$ source ~/service.osrc
glance image-list
```

This will produce output like the following example, showing three images that have been added by Ironic:

```
+----------------------------------------+------------------------+
| ID                                     | Name                   |
+----------------------------------------+------------------------+
| b9499494-7db3-4448-b67f-233b86489c1f   | ir-deploy-iso-HOS4.0   |
| 8bee92b7-98ae-4242-b80e-1201a475361a   | ir-deploy-kernel-HOS4.0 |
| 0c889803-469d-4aad-8cf6-3501e39c532c   | ir-deploy-ramdisk-HOS4.0 |
+----------------------------------------+------------------------+
```

The `ir-deploy-ramdisk-HOS4.0` image is a traditional boot ramdisk used by the `agent_ipmitool`, `pxe_ipmitool`, and `pxe_ilo` drivers while `ir-deploy-iso-HOS4.0` is an ISO image that is supplied as virtual media to the host when using the `agent_ilo` driver.

Provisioning a node

The information required to provision a node varies slightly depending on the driver used. In general the following details are required.

- Network access information and credentials to connect to the management interface of the node.
- Sufficient properties to allow for nova flavor matching.
- A deployment image to perform the actual deployment of the guest operating system to the baremetal node.

A combination of the `ironic node-create` and `ironic node-update` commands are used for registering a node's characteristics with the Ironic service. In particular, `ironic node-update <nodeid>` **add** and `ironic node-update <nodeid>` **replace** can be used to modify the properties of a node after it has been created while `ironic node-update <nodeid>` **remove** will remove a property.

Creating a node using `agent_ilo`

If you want to use a boot mode of BIOS as opposed to UEFI, then you need to ensure that the boot mode has been set correctly on the iLO:

While the iLO driver can automatically set a node to boot in UEFI mode via the `boot_mode` defined capability, it cannot set BIOS boot mode once UEFI mode has been set.

Use the `ironic node-create` command to specify the `agent_ilo` driver, network access and credential information for the iLO, properties of the node and the Glance ID of the supplied ISO IPA image. Note that memory size is specified in megabytes while disk size is gigabytes.

```
ironic node-create -d agent_ilo -i ilo_address=10.1.196.117 -i ilo_username=Administrator -i ilo_password=******** \
-p cpus=2 -p cpu_arch=x86_64 -p memory_mb=64000 -p local_gb=99  \
-i ilo_deploy_iso=b9499494-7db3-4448-b67f-233b86489c1f
```

This will generate output similar to the following:

```
+-------------+---------------------------------------------------------+
| Property    | Value                                                   |
+-------------+---------------------------------------------------------+
| uuid        | ea7246fd-e1d6-4637-9699-0b7c59c22e67                    |
| driver_info | {u'ilo_address': u'10.1.196.117', u'ilo_password': u'******', |
|             | u'ilo_deploy_iso': u'b9499494-7db3-4448-b67f-233b86489c1f',   |
|             | u'ilo_username': u'Administrator'}                      |
```

```
| extra          | {}                                                          |
| driver         | agent_ilo                                                   |
| chassis_uuid   |                                                             |
| properties     | {u'memory_mb': 64000, u'local_gb': 99, u'cpus': 2,          |
|                | u'cpu_arch': u'x86_64'}                                      |
| name           | None                                                        |
+----------------+-------------------------------------------------------------+
```

Now update the node with `boot_mode` and `boot_option` properties:

```
ironic node-update ea7246fd-e1d6-4637-9699-0b7c59c22e67 add  properties/capabilities="boot_mode:bios,boot_option:local"
```

The `ironic node-update` command returns details for all the node's characteristics.

```
+-----------------------+-------------------------------------------------------------+
| Property              | Value                                                       |
+-----------------------+-------------------------------------------------------------+
| target_power_state    | None                                                        |
| extra                 | {}                                                          |
| last_error            | None                                                        |
| updated_at            | None                                                        |
| maintenance_reason    | None                                                        |
| provision_state       | available                                                   |
| clean_step            | {}                                                          |
| uuid                  | ea7246fd-e1d6-4637-9699-0b7c59c22e67                        |
| console_enabled       | False                                                       |
| target_provision_state| None                                                        |
| provision_updated_at  | None                                                        |
| maintenance           | False                                                       |
| inspection_started_at | None                                                        |
| inspection_finished_at| None                                                        |
| power_state           | None                                                        |
| driver                | agent_ilo                                                   |
| reservation           | None                                                        |
| properties            | {u'memory_mb': 64000, u'cpu_arch': u'x86_64', u'local_gb': 99, u'cpus': |
|                       | 2, u'capabilities': u'boot_mode:bios,boot_option:local'}    |
| instance_uuid         | None                                                        |
| name                  | None                                                        |
| driver_info           | {u'ilo_address': u'10.1.196.117', u'ilo_password': u'******', |
|                       | u'ilo_deploy_iso': u'b9499494-7db3-4448-b67f-233b86489c1f', |
|                       | u'ilo_username': u'Administrator'}                          |
| created_at            | 2016-03-11T10:17:10+00:00                                   |
| driver_internal_info  | {}                                                          |
| chassis_uuid          |                                                             |
| instance_info         | {}                                                          |
+-----------------------+-------------------------------------------------------------+
```

Creating a node using `agent_ipmi`

Use the `ironic node-create` command to specify the `agent_ipmi` driver, network access and credential information for the iLO, properties of the node and the Glance IDs of the supplied kernel and ramdisk images. Note that memory size is specified in megabytes while disk size is gigabytes.

```
ironic node-create -d agent_ipmitool \
-i ipmi_address=10.1.196.117 -i ipmi_username=Administrator -i ipmi_password=******  \
-p cpus=2 -p memory_mb=64000 -p local_gb=99 -p cpu_arch=x86_64 \
-i deploy_kernel=8bee92b7-98ae-4242-b80e-1201a475361a  \
```

```
-i deploy_ramdisk=0c889803-469d-4aad-8cf6-3501e39c532c
```

This will generate output similar to the following:

```
+--------------+-----------------------------------------------------------------+
| Property     | Value                                                           |
+--------------+-----------------------------------------------------------------+
| uuid         | cf25b19d-098d-406c-b7a1-e0c40cc84a12                            |
| driver_info  | {u'deploy_kernel': u'8bee92b7-98ae-4242-b80e-1201a475361a',     |
|              | u'ipmi_address': u'10.1.196.117', u'ipmi_username': u'Administrator', |
|              | u'ipmi_password': u'******', u'deploy_ramdisk': u'0c889803-469d-4aad- |
|              | 8cf6-3501e39c532c'}                                             |
| extra        | {}                                                              |
| driver       | agent_ipmitool                                                 |
| chassis_uuid |                                                                |
| properties   | {u'memory_mb': 64000, u'cpu_arch': u'x86_64', u'local_gb': 99, u'cpus': |
|              | 2}                                                             |
| name         | None                                                           |
+--------------+-----------------------------------------------------------------+
```

Now update the node with `boot_mode` and `boot_option` properties:

```
ironic node-update cf25b19d-098d-406c-b7a1-e0c40cc84a12 add  properties/capabilities="boot_mode:bios,boot_option:local"
```

The `ironic node-update` command returns details for all the node's characteristics.

```
+------------------------+-----------------------------------------------------------------+
| Property               | Value                                                           |
+------------------------+-----------------------------------------------------------------+
| target_power_state     | None                                                           |
| extra                  | {}                                                             |
| last_error             | None                                                           |
| updated_at             | None                                                           |
| maintenance_reason     | None                                                           |
| provision_state        | available                                                      |
| clean_step             | {}                                                             |
| uuid                   | cf25b19d-098d-406c-b7a1-e0c40cc84a12                           |
| console_enabled        | False                                                          |
| target_provision_state | None                                                           |
| provision_updated_at   | None                                                           |
| maintenance            | False                                                          |
| inspection_started_at  | None                                                           |
| inspection_finished_at | None                                                           |
| power_state            | None                                                           |
| driver                 | agent_ipmitool                                                |
| reservation            | None                                                           |
| properties             | {u'memory_mb': 64000, u'cpu_arch': u'x86_64', u'local_gb': 99, u'cpus': |
|                        | 2, u'capabilities': u'boot_mode:bios,boot_option:local'}      |
| instance_uuid          | None                                                           |
| name                   | None                                                           |
| driver_info            | {u'ipmi_password': u'******', u'ipmi_address': u'10.1.196.117', |
|                        | u'ipmi_username': u'Administrator', u'deploy_kernel':         |
|                        | u'8bee92b7-98ae-4242-b80e-1201a475361a', u'deploy_ramdisk': u'0c889803 |
|                        | -469d-4aad-8cf6-3501e39c532c'}                                |
| created_at             | 2016-03-11T14:19:18+00:00                                      |
| driver_internal_info   | {}                                                             |
| chassis_uuid           |                                                                |
| instance_info          | {}                                                             |
```

```
+-----------------------+----------------------------------------------------------------+
```

For more information on node enrollment, see the OpenStack documentation at *http://docs.openstack.org/developer/ironic/deploy/install-guide.html#enrollment*.

Create Flavor

Nova uses flavors when fulfilling requests for bare metal nodes. The Nova scheduler attempts to match the requested flavor against the properties of the created Ironic nodes. So an administrator needs to set up flavors that correspond to the available bare metal nodes using the `nova flavor-create` command.

```
nova flavor-create bmtest auto 64000  99 2

+--------------------------------------+--------+-----------+------+-----------+------+-------+-------------+-----------+
| ID                                   | Name   | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+--------------------------------------+--------+-----------+------+-----------+------+-------+-------------+-----------+
| 645de08d-2bc6-43f1-8a5f-2315a75b1348 | bmtest | 64000     | 99   | 0         |      | 2     | 1.0         | True      |
+--------------------------------------+--------+-----------+------+-----------+------+-------+-------------+-----------+
```

To see a list of all the available flavors, run `nova flavor-list`

```
nova flavor-list

+--------------------------------------+-------------+-----------+------+-----------+------+-------+-------------+-----------+
| ID                                   | Name        | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+--------------------------------------+-------------+-----------+------+-----------+------+-------+-------------+-----------+
| 1                                    | m1.tiny     | 512       | 1    | 0         |      | 1     | 1.0         | True      |
| 2                                    | m1.small    | 2048      | 20   | 0         |      | 1     | 1.0         | True      |
| 3                                    | m1.medium   | 4096      | 40   | 0         |      | 2     | 1.0         | True      |
| 4                                    | m1.large    | 8192      | 80   | 0         |      | 4     | 1.0         | True      |
| 5                                    | m1.xlarge   | 16384     | 160  | 0         |      | 8     | 1.0         | True      |
| 6                                    | m1.baremetal| 4096      | 80   | 0         |      | 2     | 1.0         | True      |
| 645de08d-2bc6-43f1-8a5f-2315a75b1348 | bmtest      | 64000     | 99   | 0         |      | 2     | 1.0         | True      |
+--------------------------------------+-------------+-----------+------+-----------+------+-------+-------------+-----------+
```

Now set the cpu architecture and boot mode and boot option capabilities:

```
nova flavor-key 645de08d-2bc6-43f1-8a5f-2315a75b1348 set cpu_arch=x86_64
nova flavor-key 645de08d-2bc6-43f1-8a5f-2315a75b1348 set capabilities:boot_option="local"
nova flavor-key 645de08d-2bc6-43f1-8a5f-2315a75b1348 set capabilities:boot_mode="bios"
```

For more information on flavor creation, see the OpenStack documentation at *http://docs.openstack.org/developer/ironic/deploy/install-guide.html#flavor-creation*

Create network port

Register the MAC addresses of all connected physical network interfaces intended for use with the baremetal node.

```
ironic port-create -a 5c:b9:01:88:f0:a4 -n ea7246fd-e1d6-4637-9699-0b7c59c22e67
```

Create Glance Image

You can create a whole disk image with *diskimage-builder* using a command similar to the following:

```
DIB_RELEASE=trusty disk-image-create -o ubuntu-trusty.qcow2 vm ubuntu
```

Then load the image into glance:

```
glance image-create --name='ubuntu' --disk-format=qcow2 --container-format=bare   --file ~/ubuntu.qcow2


+-----------------+--------------------------------------+
| Property        | Value                                |
+-----------------+--------------------------------------+
| checksum        | 45a4a06997e64f7120795c68beeb0e3c     |
| container_format| bare                                 |
| created_at      | 2016-02-17T10:42:14Z                 |
| disk_format     | qcow2                                |
| id              | 17e4915a-ada0-4b95-bacf-ba67133f39a7 |
| min_disk        | 0                                    |
| min_ram         | 0                                    |
| name            | ubuntu                               |
| owner           | 821b7bb8148f439191d108764301af64     |
| protected       | False                                |
| size            | 372047872                            |
| status          | active                               |
| tags            | []                                   |
| updated_at      | 2016-02-17T10:42:23Z                 |
| virtual_size    | None                                 |
| visibility      | private                              |
+-----------------+--------------------------------------+
```

This image will be used subsequently to boot the baremetal node.

Generate Key Pair

Create a keypair that you will use when you login to the newly booted node.

```
nova keypair-add ironic_kp > ironic_kp.pem
```

Determine the neuron network ID

```
neutron net-list

+--------------------------------------+----------+-----------------------------------------------------+
| id                                   | name     | subnets                                             |
+--------------------------------------+----------+-----------------------------------------------------+
| c010267c-9424-45be-8c05-99d68531ca8c | flat-net | 709ee2a1-4110-4b26-ba4d-deb74553adb9 192.3.15.0/24  |
+--------------------------------------+----------+-----------------------------------------------------+
```

Boot the node

Before booting, it is advisable to power down the node:

```
ironic node-set-power-state ea7246fd-e1d6-4637-9699-0b7c59c22e67    off
```

You can now boot the baremetal node with the information complied in the preceding steps, using the neutron network id, the whole disk image id, the matching flavor and the key name.

```
nova boot --nic net-id=c010267c-9424-45be-8c05-99d68531ca8c \
```

```
--image  17e4915a-ada0-4b95-bacf-ba67133f39a7 \
--flavor 645de08d-2bc6-43f1-8a5f-2315a75b1348 \
--key-name ironic_kp ubuntu
```

This command returns information about the state of the node that is booting.

```
+-------------------------------------+------------------------------------------------+
| Property                            | Value                                          |
+-------------------------------------+------------------------------------------------+
| OS-EXT-AZ:availability_zone         |                                                |
| OS-EXT-SRV-ATTR:host                | -                                              |
| OS-EXT-SRV-ATTR:hypervisor_hostname | -                                              |
| OS-EXT-SRV-ATTR:instance_name       | instance-00000001                              |
| OS-EXT-STS:power_state              | 0                                              |
| OS-EXT-STS:task_state               | scheduling                                     |
| OS-EXT-STS:vm_state                 | building                                       |
| OS-SRV-USG:launched_at              | -                                              |
| OS-SRV-USG:terminated_at            | -                                              |
| accessIPv4                          |                                                |
| accessIPv6                          |                                                |
| adminPass                           | adpHw3KKTjHk                                   |
| config_drive                        |                                                |
| created                             | 2016-03-11T11:00:28Z                           |
| flavor                              | bmtest (645de08d-2bc6-43f1-8a5f-2315a75b1348)  |
| hostId                              |                                                |
| id                                  | a90122ce-bba8-496f-92a0-8a7cb143007e           |
| image                               | ubuntu (17e4915a-ada0-4b95-bacf-ba67133f39a7)  |
| key_name                            | ironic_kp                                      |
| metadata                            | {}                                             |
| name                                | ubuntu                                         |
| os-extended-volumes:volumes_attached | []                                            |
| progress                            | 0                                              |
| security_groups                     | default                                        |
| status                              | BUILD                                          |
| tenant_id                           | d53bcaf15afb4cb5aea3adaedbaa60dd               |
| updated                             | 2016-03-11T11:00:28Z                           |
| user_id                             | e580c645bfec4faeadef7dbd24aaf990               |
+-------------------------------------+------------------------------------------------+
```

The boot procedure can typically take up to 10 minutes to complete. You can monitor the progress with the iLO console and by using the `nova list`, `nova show <nova_node_id>` and `ironic node-show <ironic_node_id>` commands.

```
nova list

+--------------------------------------+--------+--------+------------+-------------+---------------------+
| ID                                   | Name   | Status | Task State | Power State | Networks            |
+--------------------------------------+--------+--------+------------+-------------+---------------------+
| a90122ce-bba8-496f-92a0-8a7cb143007e | ubuntu | BUILD  | spawning   | NOSTATE     | flat-net=192.3.15.12 |
+--------------------------------------+--------+--------+------------+-------------+---------------------+
```

During the boot procedure, a login prompt will appear for Linux for HPE Helion:
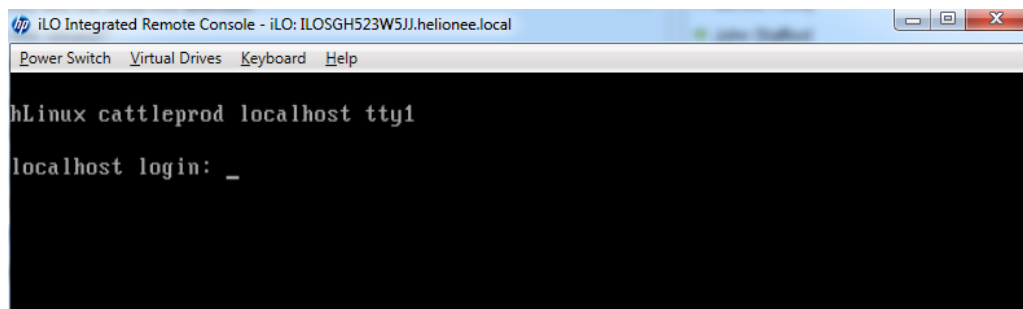
Just ignore this login and wait for the login screen of your target Operating System to appear:
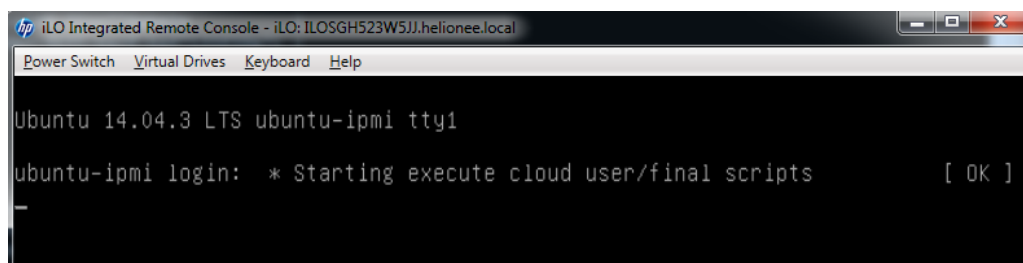
If you now run the `nova list` command, it should show the node in the running state:

```
nova list
+--------------------------------------+--------+--------+------------+-------------+--------------------+
| ID                                   | Name   | Status | Task State | Power State | Networks           |
+--------------------------------------+--------+--------+------------+-------------+--------------------+
| a90122ce-bba8-496f-92a0-8a7cb143007e | ubuntu | ACTIVE | -          | Running     | flat-net=192.3.15.14 |
+--------------------------------------+--------+--------+------------+-------------+--------------------+
```

You can now log in to the booted node using the key you generated earlier. (You may be prompted to change the permissions on your private key files so that they are not accessible by others).

```
ssh ubuntu@192.3.15.14 -i ironic_kp.pem
```

Create Glance Images using `diskimage-builder` (DIB)

The following sections show you how to create your own images using the OpenStack `diskimage-builder` tool - for information on how to install the tool see *http://docs.openstack.org/developer/diskimage-builder/user_guide/installation.html*

Creating BIOS images for RHEL

1. Install `dib-utils`:

```
pip install dib-utils
```

2. Download the qcow2 guest image from RHEL portal and define `DIB_LOCAL_IMAGE`:

```
export DIB_LOCAL_IMAGE=/home/stack/rhel-guest-image-7.2-20151102.0.x86_64.qcow2
```

3. Create a local yum repo and define DIB_YUM_REPO_CONF to be the path to the repo:

```
export DIB_YUM_REPO_CONF = <<path_to_repo_file>>
```

4. Use `disk-image-create` to create the image:

```
cd diskimage-builder/bin
./disk-image-create  -o rhel.qcow2 vm dhcp-all-interfaces  rhel7
```

Creating BIOS images for Ubuntu

```
DIB_RELEASE=trusty ./disk-image-create  -o ubuntu.qcow2 vm dhcp-all-interfaces ubuntu
```

Creating UEFI images

To create UEFI images, you will need to setup sudo to not prompt for passwords and configure web-proxy. Once you have sudo and web-proxy configured, follow these steps:

1. Clone the diskimage-builder git repository.

```
git clone https://git.gozer.hpcloud.net/openstack/diskimage-builder.git -b feature/v2
```

2. Apply the patch to your cloned `diskimage-builder` repository (*https://review.openstack.org/#/c/391850/6*) and then clone the diskimage-builder utilities (dib-utils) repository.

```
cd diskimage-builder/
git fetch git://git.openstack.org/openstack/diskimage-builder refs/changes/50/391850/6 && git cherry-pick FETCH_HEAD
cd ..
git clone https://git.gozer.hpcloud.net/openstack/dib-utils.git
```

3. Add execute permissions for `disk-image-create`.

```
chmod a+x /root/diskimage-builder/diskimage_builder/lib/disk-image-create
```

4. Export your path to include diskimage-builder utilities.

```
export PATH=/root/diskimage-builder/diskimage_builder/lib:/root/dib-utils/bin:$PATH
```

5. Verify that the diskimage-build utilities are in your cloned path by locating `dib-run-parts`.

```
which dib-run-parts
/root/dib-utils/bin/dib-run-parts
```

6. Verify that the diskimage-build is in your cloned path by locating `disk-image-create`.

```
which disk-image-create
/root/diskimage-builder/diskimage_builder/lib/disk-image-create
```

7. Install diskimage-builder.

```
cd diskimage-builder/
python setup.py install
```

8. Export the `diskimage-builder` lib files.

```
export _LIB=/root/diskimage-builder/diskimage_builder/lib
```

9. Create the UEFI Image.

- To create an Ubuntu UEFI image:

```
DIB_RELEASE=trusty DIB_UEFI_SUPPORT=true disk-image-create -o ubuntu-trusty-uefi.qcow2 vm ubuntu dhcp-all-interfaces
```

- To create a Fedora UEFI image:

```
DIB_UEFI_SUPPORT=true disk-image-create -o fedora-uefi.qcow2 vm fedora dhcp-all-interfaces
```

10. For OS support, refer to *supported_distributions.rst* in diskimage-builder user documentation.

HPE Helion OpenStack® 5.0: Provisioning Baremetal Nodes with Multi-Tenancy

Baremetal Nodes with Multi-Tenancy

1. Create a network and a subnet

```
$ neutron net-create guest-net-1
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| availability_zone_hints   |                                      |
| availability_zones        |                                      |
| created_at                | 2017-06-10T02:49:56Z                 |
| description               |                                      |
| id                        | 256d55a6-9430-4f49-8a4c-cc5192f5321e |
| ipv4_address_scope        |                                      |
| ipv6_address_scope        |                                      |
| mtu                       | 1500                                 |
| name                      | guest-net-1                          |
| project_id                | 57b792cdcdd74d16a08fc7a396ee05b6     |
| provider:network_type     | vlan                                 |
| provider:physical_network | physnet1                             |
| provider:segmentation_id  | 1152                                 |
| revision_number           | 2                                    |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tags                      |                                      |
| tenant_id                 | 57b792cdcdd74d16a08fc7a396ee05b6     |
| updated_at                | 2017-06-10T02:49:57Z                 |
+---------------------------+--------------------------------------+

$ neutron  subnet-create guest-net-1 200.0.0.0/24
Created a new subnet:
+-------------------+------------------------------------------------+
| Field             | Value                                          |
+-------------------+------------------------------------------------+
| allocation_pools  | {"start": "200.0.0.2", "end": "200.0.0.254"}   |
| cidr              | 200.0.0.0/24                                   |
| created_at        | 2017-06-10T02:53:08Z                           |
| description       |                                                |
| dns_nameservers   |                                                |
| enable_dhcp       | True                                           |
| gateway_ip        | 200.0.0.1                                      |
| host_routes       |                                                |
| id                | 53accf35-ae02-43ae-95d8-7b5efed18ae9           |
| ip_version        | 4                                              |
| ipv6_address_mode |                                                |
```

```
| ipv6_ra_mode        |                                       |
| name                |                                       |
| network_id          | 256d55a6-9430-4f49-8a4c-cc5192f5321e  |
| project_id          | 57b792cdcdd74d16a08fc7a396ee05b6      |
| revision_number     | 2                                     |
| service_types       |                                       |
| subnetpool_id       |                                       |
| tenant_id           | 57b792cdcdd74d16a08fc7a396ee05b6      |
| updated_at          | 2017-06-10T02:53:08Z                  |
+---------------------+---------------------------------------+
```

2. Review glance image list

```
$ glance image-list
+--------------------------------------+--------------------------+
| ID                                   | Name                     |
+--------------------------------------+--------------------------+
| 73205fb7-64f1-4243-8d0f-a0dd2e493c9d | cirros-0.3.4-x86_64      |
| 83eecf9c-d675-4bf9-a5d5-9cf1fe9ee9c2 | ir-deploy-iso-HOS5.0     |
| db3d131f-2fb0-4189-bb8d-424ee0886e4c | ir-deploy-kernel-HOS5.0  |
| 304cae15-3fe5-4f1c-8478-c65da5092a2c | ir-deploy-ramdisk-HOS5.0 |
+--------------------------------------+--------------------------+
```

3. Create Ironic node

```
$ ironic --ironic-api-version 1.22 node-create -d agent_ipmitool -n test-node-1 -i ipmi_address=192.168.9.69 -i ipmi_username=ilo_user -i ipmi_password=XXXXXXXX --network-
interface neutron -p  memory_mb=4096 -p cpu_arch=x86_64 -p local_gb=80 -p cpus=2 -p capabilities=boot_mode:bios,boot_option:local -p root_device='{"name":"/dev/sda"}' -i
 deploy_kernel=db3d131f-2fb0-4189-bb8d-424ee0886e4c -i deploy_ramdisk=304cae15-3fe5-4f1c-8478-c65da5092a2c

+-------------------+------------------------------------------------------------------+
| Property          | Value                                                            |
+-------------------+------------------------------------------------------------------+
| chassis_uuid      |                                                                  |
| driver            | agent_ipmitool                                                   |
| driver_info       | {u'deploy_kernel': u'db3d131f-2fb0-4189-bb8d-424ee0886e4c',      |
|                   | u'ipmi_address': u'192.168.9.69', u'ipmi_username': u'gozer',    |
|                   | u'ipmi_password': u'******', u'deploy_ramdisk':                  |
|                   | u'304cae15-3fe5-4f1c-8478-c65da5092a2c'}                         |
| extra             | {}                                                               |
| name              | test-node-1                                                      |
| network_interface | neutron                                                          |
| properties        | {u'cpu_arch': u'x86_64', u'root_device': {u'name': u'/dev/sda'}, |
|                   | u'cpus': 2, u'capabilities': u'boot_mode:bios,boot_option:local',|
|                   | u'memory_mb': 4096, u'local_gb': 80}                             |
| resource_class    | None                                                             |
| uuid              | cb4dda0d-f3b0-48b9-ac90-ba77b8c66162                             |
+-------------------+------------------------------------------------------------------+
```

ipmi_address, ipmi_username and ipmi_password are iLo access parameters for baremetal Ironic node. Adjust memory_mb, cpus, local_gb to your node size requirements. They also need to be reflected in flavor setting (see below). Use capabilities boot_mode:bios for baremetal nodes operating in Legacy BIOS mode. For UEFI baremetal nodes, use boot_mode:uefi lookup deploy_kernel and deploy_ramdisk in glance image list output above.

⚠ **Important:** Since we are using Ironic API version 1.22, node is created initial state **enroll**. It needs to be explicitly moved to **available** state. This behavior changed in API version 1.11

4. Create port

```
$ ironic --ironic-api-version 1.22 port-create --address f0:92:1c:05:6c:40 --node cb4dda0d-f3b0-48b9-ac90-ba77b8c66162 -l switch_id=e8:f7:24:bf:07:2e -l switch_info=hp59srv1-
a-11b -l port_id="Ten-GigabitEthernet 1/0/34" --pxe-enabled true
+---------------------+-------------------------------------------------------------------+
| Property            | Value                                                             |
+---------------------+-------------------------------------------------------------------+
| address             | f0:92:1c:05:6c:40                                                 |
```

```
| extra                | {}                                                    |
| local_link_connection | {u'switch_info': u'hp59srv1-a-11b', u'port_id': u'Ten-GigabitEthernet |
|                      | 1/0/34', u'switch_id': u'e8:f7:24:bf:07:2e'}          |
| node_uuid            | cb4dda0d-f3b0-48b9-ac90-ba77b8c66162                  |
| pxe_enabled          | True                                                  |
| uuid                 | a49491f3-5595-413b-b4a7-bb6f9abec212                  |
+---------------------+-------------------------------------------------------+
```

- for --address, use MAC of 1st NIC of ironic baremetal node, which will be used for PXE boot
- for --node, use ironic node uuid (see above)
- for -l switch_id, use switch management interface MAC address. It can be retrieved by pinging switch management IP and looking up MAC address in 'arp -l -n' command output.
- for -l switch_info, use switch_id from data/ironic/ironic_config.yml file. If you have several switch config definitions, use the right switch your baremetal node is connected to.
- for -l port_id, use port ID on the switch

5. Move ironic node to manage and then available state

```
$ ironic node-set-provision-state test-node-1 manage
$ ironic node-set-provision-state test-node-1 provide
```

6. Once node is successfully moved to available state, it's resources should be included into Nova hypervisor statistics

```
$ nova hypervisor-stats
+---------------------+-------+
| Property            | Value |
+---------------------+-------+
| count               | 1     |
| current_workload    | 0     |
| disk_available_least | 80   |
| free_disk_gb        | 80    |
| free_ram_mb         | 4096  |
| local_gb            | 80    |
| local_gb_used       | 0     |
| memory_mb           | 4096  |
| memory_mb_used      | 0     |
| running_vms         | 0     |
| vcpus               | 2     |
| vcpus_used          | 0     |
+---------------------+-------+
```

7. Prepare a keypair, which will be used for logging into the node

```
$ nova keypair-add ironic_kp > ironic_kp.pem
```

8. Obtain user image and upload it to glance. Please refer to OpenStack documentation on user image creation: *https://docs.openstack.org/project-install-guide/baremetal/draft/configure-glance-images.html*.

📋 **Note:** Deployed images are already populated by HPE Helion OpenStack installer.

```
$ glance image-create --name='Ubuntu Trusty 14.04' --disk-format=qcow2 --container-format=bare --file ~/ubuntu-trusty.qcow2
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | d586d8d2107f328665760fee4c81caf0     |
| container_format | bare                                 |
| created_at       | 2017-06-13T22:38:45Z                 |
| disk_format      | qcow2                                |
| id               | 9fdd54a3-ccf5-459c-a084-e50071d0aa39 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | Ubuntu Trusty 16.04                  |
| owner            | 57b792cdcdd74d16a08fc7a396ee05b6     |
| protected        | False                                |
```

```
| size            | 371508736                          |
| status          | active                             |
| tags            | []                                 |
| updated_at      | 2017-06-13T22:38:55Z               |
| virtual_size    | None                               |
| visibility      | private                            |
+-----------------+------------------------------------+

$ glance image-list
+--------------------------------------+------------------------+
| ID                                   | Name                   |
+--------------------------------------+------------------------+
| 73205fb7-64f1-4243-8d0f-a0dd2e493c9d | cirros-0.3.4-x86_64    |
| 83eecf9c-d675-4bf9-a5d5-9cf1fe9ee9c2 | ir-deploy-iso-HOS5.0   |
| db3d131f-2fb0-4189-bb8d-424ee0886e4c | ir-deploy-kernel-HOS5.0 |
| 304cae15-3fe5-4f1c-8478-c65da5092a2c | ir-deploy-ramdisk-HOS5.0 |
| 9fdd54a3-ccf5-459c-a084-e50071d0aa39 | Ubuntu Trusty 14.04    |
+--------------------------------------+------------------------+
```

9. Create a baremetal flavor and set flavor keys specifying requested node size, architecture and boot mode. A flavor can be re-used for several nodes having the same size, architecture and boot mode

```
$ nova flavor-create m1.ironic auto 4096 80 2
+--------------------------------------+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| ID                                   | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+--------------------------------------+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| ab69881b-0d5a-497d-93ae-6e28694c87bf | m1.ironic | 4096      | 80   | 0         |      | 2     | 1.0         | True      |
+--------------------------------------+-----------+-----------+------+-----------+------+-------+-------------+-----------+

$ nova flavor-key ab69881b-0d5a-497d-93ae-6e28694c87bf set cpu_arch=x86_64
$ nova flavor-key ab69881b-0d5a-497d-93ae-6e28694c87bf set capabilities:boot_option="local"
$ nova flavor-key ab69881b-0d5a-497d-93ae-6e28694c87bf set capabilities:boot_mode="bios"
```

Parameters must match parameters of ironic node above. Use `capabilities:boot_mode="bios"` for Legacy BIOS nodes. For UEFI nodes, use `capabilities:boot_mode="uefi"`

10. Boot the node

```
$ nova boot --flavor m1.ironic --image 9fdd54a3-ccf5-459c-a084-e50071d0aa39 --key-name ironic_kp --nic net-id=256d55a6-9430-4f49-8a4c-cc5192f5321e test-node-1
+--------------------------------------+----------------------------------------------------------+
| Property                             | Value                                                    |
+--------------------------------------+----------------------------------------------------------+
| OS-DCF:diskConfig                    | MANUAL                                                   |
| OS-EXT-AZ:availability_zone          |                                                          |
| OS-EXT-SRV-ATTR:host                 | -                                                        |
| OS-EXT-SRV-ATTR:hypervisor_hostname  | -                                                        |
| OS-EXT-SRV-ATTR:instance_name        |                                                          |
| OS-EXT-STS:power_state               | 0                                                        |
| OS-EXT-STS:task_state                | scheduling                                               |
| OS-EXT-STS:vm_state                  | building                                                 |
| OS-SRV-USG:launched_at               | -                                                        |
| OS-SRV-USG:terminated_at             | -                                                        |
| accessIPv4                           |                                                          |
| accessIPv6                           |                                                          |
| adminPass                            | XXXXXXXXXXXX                                             |
| config_drive                         |                                                          |
| created                              | 2017-06-14T21:25:18Z                                     |
| flavor                               | m1.ironic (ab69881b-0d5a-497d-93ae-6e28694c87bf)         |
| hostId                               |                                                          |
| id                                   | f1a8c63e-da7b-4d9a-8648-b1baa6929682                     |
| image                                | Ubuntu Trusty 14.04 (9fdd54a3-ccf5-459c-a084-e50071d0aa39) |
| key_name                             | ironic_kp                                                |
| metadata                             | {}                                                       |
| name                                 | test-node-1                                              |
| os-extended-volumes:volumes_attached | []                                                       |
```

```
| progress                         | 0                                                  |
| security_groups                  | default                                            |
| status                           | BUILD                                              |
| tenant_id                        | 57b792cdcdd74d16a08fc7a396ee05b6                   |
| updated                          | 2017-06-14T21:25:17Z                               |
| user_id                          | cc76d7469658401fbd4cf772278483d9                   |
+----------------------------------+----------------------------------------------------+
```

- for --image, use ID of user image created at previous step
- for --nic net-id, use id of tenant network created at the beginning

> **Note:** During the node provisioning, the following is happening in the background:
>
> Neutron connects to switch management interfaces and assigns provisioning VLAN to baremetal node port on the switch. Ironic powers up the node using iLo interface. Node is booting IPA image via PXE. IPA image is writing provided user image onto specified root device (/dev/sda) and powers node down. Neutron connects to switch management interfaces and assigns tenant VLAN to baremetal node port on the switch. A VLAN ID is selected from provided range. Ironic powers up the node using iLo interface. Node is booting user image from disk.

11. Once provisioned, node will join the private tenant network. Access to private network from other networks is defined by switch configuration.

HPE Helion OpenStack® 5.0: View Ironic System Details

View details about the server using `nova show <nova-node-id>`

```
nova show  a90122ce-bba8-496f-92a0-8a7cb143007e

+----------------------------------+----------------------------------------------------------+
| Property                         | Value                                                    |
+----------------------------------+----------------------------------------------------------+
| OS-EXT-AZ:availability_zone      | nova                                                     |
| OS-EXT-SRV-ATTR:host             | helion-cp1-ir-compute0001-mgmt                           |
| OS-EXT-SRV-ATTR:hypervisor_hostname | ea7246fd-e1d6-4637-9699-0b7c59c22e67                  |
| OS-EXT-SRV-ATTR:instance_name    | instance-0000000a                                        |
| OS-EXT-STS:power_state           | 1                                                        |
| OS-EXT-STS:task_state            | -                                                        |
| OS-EXT-STS:vm_state              | active                                                   |
| OS-SRV-USG:launched_at           | 2016-03-11T12:26:25.000000                               |
| OS-SRV-USG:terminated_at         | -                                                        |
| accessIPv4                       |                                                          |
| accessIPv6                       |                                                          |
| config_drive                     |                                                          |
| created                          | 2016-03-11T12:17:54Z                                     |
| flat-net network                 | 192.3.15.14                                              |
| flavor                           | bmtest (645de08d-2bc6-43f1-8a5f-2315a75b1348)            |
| hostId                           | ecafa4f40eb5f72f7298de0ef51eb7769e3bad47cbc01aa0a076114f |
| id                               | a90122ce-bba8-496f-92a0-8a7cb143007e                     |
| image                            | ubuntu (17e4915a-ada0-4b95-bacf-ba67133f39a7)            |
| key_name                         | ironic_kp                                                |
| metadata                         | {}                                                       |
| name                             | ubuntu                                                   |
| os-extended-volumes:volumes_attached | []                                                   |
| progress                         | 0                                                        |
| security_groups                  | default                                                  |
| status                           | ACTIVE                                                   |
| tenant_id                        | d53bcaf15afb4cb5aea3adaedbaa60dd                         |
| updated                          | 2016-03-11T12:26:26Z                                     |
| user_id                          | e580c645bfec4faeadef7dbd24aaf990                         |
+----------------------------------+----------------------------------------------------------+
```

View detailed information about a node using `ironic node-show <ironic-node-id>`

```
ironic node-show  ea7246fd-e1d6-4637-9699-0b7c59c22e67

+----------------------+-------------------------------------------------------------+
| Property             | Value                                                       |
+----------------------+-------------------------------------------------------------+
| target_power_state   | None                                                        |
| extra                | {}                                                          |
| last_error           | None                                                        |
| updated_at           | 2016-03-11T12:26:25+00:00                                   |
| maintenance_reason   | None                                                        |
| provision_state      | active                                                      |
| clean_step           | {}                                                          |
| uuid                 | ea7246fd-e1d6-4637-9699-0b7c59c22e67                       |
| console_enabled      | False                                                       |
| target_provision_state | None                                                      |
| provision_updated_at | 2016-03-11T12:26:25+00:00                                   |
| maintenance          | False                                                       |
| inspection_started_at | None                                                       |
| inspection_finished_at | None                                                      |
| power_state          | power on                                                    |
| driver               | agent_ilo                                                   |
| reservation          | None                                                        |
| properties           | {u'memory_mb': 64000, u'cpu_arch': u'x86_64', u'local_gb': 99, u'cpus': |
|                      | 2, u'capabilities': u'boot_mode:bios,boot_option:local'}    |
| instance_uuid        | a90122ce-bba8-496f-92a0-8a7cb143007e                       |
| name                 | None                                                        |
| driver_info          | {u'ilo_address': u'10.1.196.117', u'ilo_password': u'******', |
|                      | u'ilo_deploy_iso': u'b9499494-7db3-4448-b67f-233b86489c1f', |
|                      | u'ilo_username': u'Administrator'}                          |
| created_at           | 2016-03-11T10:17:10+00:00                                   |
| driver_internal_info | {u'agent_url': u'http://192.3.15.14:9999', u'is_whole_disk_image': True, |
|                      | u'agent_last_heartbeat': 1457699159}                       |
| chassis_uuid         |                                                             |
| instance_info        | {u'root_gb': u'99', u'display_name': u'ubuntu', u'image_source': u |
|                      | '17e4915a-ada0-4b95-bacf-ba67133f39a7', u'capabilities': u'{"boot_mode": |
|                      | "bios", "boot_option": "local"}', u'memory_mb': u'64000', u'vcpus': |
|                      | u'2', u'image_url': u'http://192.168.12.2:8080/v1/AUTH_ba121db7732f4ac3a |
|                      | 50cc4999a10d58d/glance/17e4915a-ada0-4b95-bacf-ba67133f39a7?temp_url_sig |
|                      | =ada691726337805981ac002c0fbfc905eb9783ea&temp_url_expires=1457699878', |
|                      | u'image_container_format': u'bare', u'local_gb': u'99',     |
|                      | u'image_disk_format': u'qcow2', u'image_checksum':          |
|                      | u'2d7bb1e78b26f32c50bd9da99102150b', u'swap_mb': u'0'}      |
+----------------------+-------------------------------------------------------------+
```

View detailed information about a port using `ironic port-show <ironic-port-id>`

```
ironic port-show a17a4ef8-a711-40e2-aa27-2189c43f0b67

+------------+----------------------------------------------------------+
| Property   | Value                                                    |
+------------+----------------------------------------------------------+
| node_uuid  | ea7246fd-e1d6-4637-9699-0b7c59c22e67                    |
| uuid       | a17a4ef8-a711-40e2-aa27-2189c43f0b67                    |
| extra      | {u'vif_port_id': u'82a5ab28-76a8-4c9d-bfb4-624aeb9721ea'} |
| created_at | 2016-03-11T10:40:53+00:00                                |
| updated_at | 2016-03-11T12:17:56+00:00                                |
| address    | 5c:b9:01:88:f0:a4                                        |
```

```
+-----------+-------------------------------------------------+
```

View detailed information about a hypervisor using `nova hypervisor-list` **and** `nova hypervisor-show`

```
nova hypervisor-list

+-----+-------------------------------------+-------+---------+
| ID  | Hypervisor hostname                 | State | Status  |
+-----+-------------------------------------+-------+---------+
| 541 | ea7246fd-e1d6-4637-9699-0b7c59c22e67 | up    | enabled |
+-----+-------------------------------------+-------+---------+
```

```
nova hypervisor-show ea7246fd-e1d6-4637-9699-0b7c59c22e67

+-----------------------+--------------------------------------+
| Property              | Value                                |
+-----------------------+--------------------------------------+
| cpu_info              |                                      |
| current_workload      | 0                                    |
| disk_available_least  | 0                                    |
| free_disk_gb          | 0                                    |
| free_ram_mb           | 0                                    |
| host_ip               | 192.168.12.6                         |
| hypervisor_hostname   | ea7246fd-e1d6-4637-9699-0b7c59c22e67 |
| hypervisor_type       | ironic                               |
| hypervisor_version    | 1                                    |
| id                    | 541                                  |
| local_gb              | 99                                   |
| local_gb_used         | 99                                   |
| memory_mb             | 64000                                |
| memory_mb_used        | 64000                                |
| running_vms           | 1                                    |
| service_disabled_reason | None                               |
| service_host          | helion-cp1-ir-compute0001-mgmt       |
| service_id            | 25                                   |
| state                 | up                                   |
| status                | enabled                              |
| vcpus                 | 2                                    |
| vcpus_used            | 2                                    |
+-----------------------+--------------------------------------+
```

View a list of all running services using `nova service-list`

```
nova service-list

+----+------------------+------------------------------+----------+---------+-------+----------------------------+-----------------+
| Id | Binary           | Host                         | Zone     | Status  | State | Updated_at                 | Disabled Reason |
+----+------------------+------------------------------+----------+---------+-------+----------------------------+-----------------+
| 1  | nova-conductor   | helion-cp1-c1-m1-mgmt        | internal | enabled | up    | 2016-03-11T12:44:51.000000 | -               |
| 7  | nova-conductor   | helion-cp1-c1-m2-mgmt        | internal | enabled | up    | 2016-03-11T12:44:53.000000 | -               |
| 10 | nova-conductor   | helion-cp1-c1-m3-mgmt        | internal | enabled | up    | 2016-03-11T12:44:52.000000 | -               |
| 13 | nova-scheduler   | helion-cp1-c1-m1-mgmt        | internal | enabled | up    | 2016-03-11T12:44:53.000000 | -               |
| 16 | nova-scheduler   | helion-cp1-c1-m3-mgmt        | internal | enabled | up    | 2016-03-11T12:44:57.000000 | -               |
| 19 | nova-scheduler   | helion-cp1-c1-m2-mgmt        | internal | enabled | up    | 2016-03-11T12:44:56.000000 | -               |
| 22 | nova-consoleauth | helion-cp1-c1-m1-mgmt        | internal | enabled | up    | 2016-03-11T12:44:50.000000 | -               |
| 25 | nova-compute     | helion-cp1-ir-compute0001-mgmt | nova   | enabled | up    | 2016-03-11T12:44:50.000000 | -               |
```

```
+----+----------------+------------------------+--------+--------+------+--------------------------+----------------+
```

HPE Helion OpenStack® 5.0: Troubleshooting Ironic Installation

Sometimes the `nova boot` command does not succeed and when you do a `nova list`, you will see output like the following:

```
nova list

+--------------------------------------+--------+--------+-----------+-------------+--------------------+
| ID                                   | Name   | Status | Task State | Power State | Networks           |
+--------------------------------------+--------+--------+-----------+-------------+--------------------+
| ee08f82e-8920-4360-be51-a3f995624e5f | ubuntu | ERROR  | -         | NOSTATE     |                    |
+--------------------------------------+--------+--------+-----------+-------------+--------------------+
```

You should execute the `nova show <nova-node-id>` and `ironic node-show <ironic-node-id>` commands to get more information about the error.

No valid host was found. There are not enough hosts available.

This error is typically seen when performing the `nova boot` where there is a mismatch between the properties set on the node and the flavor used. For example, the output from a `nova show` command may look like this:

```
nova show  ee08f82e-8920-4360-be51-a3f995624e5f

+------------------------------------+----------------------------------------------------------------------------------+
| Property                           | Value                                                                            |
+------------------------------------+----------------------------------------------------------------------------------+
| OS-EXT-AZ:availability_zone        |                                                                                  |
| OS-EXT-SRV-ATTR:host               | -                                                                                |
| OS-EXT-SRV-ATTR:hypervisor_hostname | -                                                                               |
| OS-EXT-SRV-ATTR:instance_name      | instance-00000001                                                                |
| OS-EXT-STS:power_state             | 0                                                                                |
| OS-EXT-STS:task_state              | -                                                                                |
| OS-EXT-STS:vm_state                | error                                                                            |
| OS-SRV-USG:launched_at             | -                                                                                |
| OS-SRV-USG:terminated_at           | -                                                                                |
| accessIPv4                         |                                                                                  |
| accessIPv6                         |                                                                                  |
| config_drive                       |                                                                                  |
| created                            | 2016-03-11T11:00:28Z                                                             |
| fault                              | {"message": "No valid host was found. There are not enough hosts available.", "code": 500, "details": "  File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/nova/conductor/manager.py\", line 739, in build_instances |
|                                    |          request_spec, filter_properties)                                        |
```

```
|                                       |     File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/nova/scheduler/utils.py\", line 343, in wrapped
|                                       |                                                                                                                                                |
|                                       |       return func(*args, **kwargs)                                                                                                              |
|                                       |                                                                                                                                                |
|                                       |     File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/nova/scheduler/client/__init__.py\", line 52, in
  select_destinations                  |                                                                                                                                                |
|                                       |       context, request_spec, filter_properties)                                                                                                |
|                                       |                                                                                                                                                |
|                                       |     File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/nova/scheduler/client/__init__.py\", line 37, in __run_method
|                                       |                                                                                                                                                |
|                                       |       return getattr(self.instance, __name)(*args, **kwargs)                                                                                    |
|                                       |                                                                                                                                                |
|                                       |     File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/nova/scheduler/client/query.py\", line 34, in
  select_destinations                  |                                                                                                                                                |
|                                       |       context, request_spec, filter_properties)                                                                                                |
|                                       |                                                                                                                                                |
|                                       |     File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/nova/scheduler/rpcapi.py\", line 120, in select_destinations
|                                       |                                                                                                                                                |
|                                       |       request_spec=request_spec, filter_properties=filter_properties)                                                                           |
|                                       |                                                                                                                                                |
|                                       |     File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/oslo_messaging/rpc/client.py\", line 158, in call
|                                       |                                                                                                                                                |
|                                       |       retry=self.retry)                                                                                                                         |
|                                       |                                                                                                                                                |
|                                       |     File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/oslo_messaging/transport.py\", line 90, in _send
|                                       |                                                                                                                                                |
|                                       |       timeout=timeout, retry=retry)                                                                                                             |
|                                       |                                                                                                                                                |
|                                       |     File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/oslo_messaging/_drivers/amqpdriver.py\", line 462, in send
|                                       |                                                                                                                                                |
|                                       |       retry=retry)                                                                                                                             |
|                                       |                                                                                                                                                |
|                                       |     File \"/opt/stack/venv/nova-20160308T002421Z/lib/python2.7/site-packages/oslo_messaging/_drivers/amqpdriver.py\", line 453, in _send
|                                       |                                                                                                                                                |
|                                       |       raise result                                                                                                                             |
|                                       |                                                                                                                                                |
|                                       | ", "created": "2016-03-11T11:00:29Z"}                                                                                                           |
|                                       |                                                                                                                                                |
| flavor                                | bmtest (645de08d-2bc6-43f1-8a5f-2315a75b1348)                                                                                                  |
|                                       |                                                                                                                                                |
| hostId                                |                                                                                                                                                |
|                                       |                                                                                                                                                |
| id                                    | ee08f82e-8920-4360-be51-a3f995624e5f                                                                                                           |
|                                       |                                                                                                                                                |
| image                                 | ubuntu (17e4915a-ada0-4b95-bacf-ba67133f39a7)                                                                                                  |
|                                       |                                                                                                                                                |
| key_name                              | ironic_kp                                                                                                                                      |
|                                       |                                                                                                                                                |
| metadata                              | {}                                                                                                                                             |
|                                       |                                                                                                                                                |
| name                                  | ubuntu                                                                                                                                         |
|                                       |                                                                                                                                                |
| os-extended-volumes:volumes_attached  | []                                                                                                                                             |
|                                       |                                                                                                                                                |
| status                                | ERROR                                                                                                                                          |
|                                       |                                                                                                                                                |
| tenant_id                             | d53bcaf15afb4cb5aea3adaedbaa60dd                                                                                                               |
|                                       |                                                                                                                                                |
| updated                               | 2016-03-11T11:00:28Z                                                                                                                           |
|                                       |                                                                                                                                                |
| user_id                               | e580c645bfec4faeadef7dbd24aaf990                                                                                                               |
|                                       |                                                                                                                                                |
```

```
+---------------------------------
+------------------------------------------------------------------------------------------------
+
```

You can find more information about the error by inspecting the log file at `/var/log/nova/nova-scheduler.log` or alternatively by viewing the error location in the source files listed in the stack-trace (in bold above).

To find the mis-match, compare the properties of the ironic node:

```
+-----------------------+-------------------------------------------------------------------+
| Property              | Value                                                             |
+-----------------------+-------------------------------------------------------------------+
| target_power_state    | None                                                              |
| extra                 | {}                                                                |
| last_error            | None                                                              |
| updated_at            | None                                                              |
| maintenance_reason    | None                                                              |
| provision_state       | available                                                         |
| clean_step            | {}                                                                |
| uuid                  | ea7246fd-e1d6-4637-9699-0b7c59c22e67                              |
| console_enabled       | False                                                             |
| target_provision_state| None                                                              |
| provision_updated_at  | None                                                              |
| maintenance           | False                                                             |
| inspection_started_at | None                                                              |
| inspection_finished_at| None                                                              |
| power_state           | None                                                              |
| driver                | agent_ilo                                                         |
| reservation           | None                                                              |
| properties            | {u'memory_mb': 64000, u'local_gb': 99, u'cpus': 2, u'capabilities':|
|                       | u'boot_mode:bios,boot_option:local'}                              |
| instance_uuid         | None                                                              |
| name                  | None                                                              |
| driver_info           | {u'ilo_address': u'10.1.196.117', u'ilo_password': u'******',     |
|                       | u'ilo_deploy_iso': u'b9499494-7db3-4448-b67f-233b86489c1f',        |
|                       | u'ilo_username': u'Administrator'}                                |
| created_at            | 2016-03-11T10:17:10+00:00                                        |
| driver_internal_info  | {}                                                                |
| chassis_uuid          |                                                                   |
| instance_info         | {}                                                                |
+-----------------------+-------------------------------------------------------------------+
```

with the flavor characteristics:

```
nova flavor-show

+--------------------------+---------------------------------------------------------------------------------+
| Property                 | Value                                                                           |
+--------------------------+---------------------------------------------------------------------------------+
| OS-FLV-DISABLED:disabled | False                                                                           |
| OS-FLV-EXT-DATA:ephemeral | 0                                                                              |
| disk                     | 99                                                                              |
| extra_specs              | {"capabilities:boot_option": "local", "cpu_arch": "x86_64", "capabilities:boot_mode": "bios"} |
| id                       | 645de08d-2bc6-43f1-8a5f-2315a75b1348                                           |
| name                     | bmtest                                                                          |
| os-flavor-access:is_public | True                                                                          |
| ram                      | 64000                                                                           |
| rxtx_factor              | 1.0                                                                             |
| swap                     |                                                                                 |
| vcpus                    | 2                                                                               |
+--------------------------+---------------------------------------------------------------------------------+
```

In this instance, the problem is caused by the absence of the **"cpu_arch": "x86_64"** property on the ironic node. This can be resolved by updating the ironic node, adding the missing property:

```
ironic node-update ea7246fd-e1d6-4637-9699-0b7c59c22e67 add properties/cpu_arch=x86_64
```

and then re-running the `nova boot` command.

Deployment to a node fails and in "ironic node-list" command, the power_state column for the node is shown as "None"

Possible cause: The IPMI commands to the node take longer to change the power state of the server.

Resolution: Check if the node power state can be changed using the following command

```
ironic node-set-power-state $NODEUUID on
```

If the above command succeeds and the power_state column is updated correctly, then the following steps are required to increase the power sync interval time.

On the first controller, reconfigure Ironic to increase the power sync interval time. In the example below, it is set to 120 seconds. This value may have to be tuned based on the setup.

1. Go to the `~/helion/my_cloud/config/ironic/` directory and edit `ironic-conductor.conf.j2` to set the `sync_power_state_interval` value:

```
[conductor]
sync_power_state_interval = 120
```

2. Save the file and then run the following playbooks:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
ansible-playbook -i hosts/localhost ready-deployment.yml
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts ironic-reconfigure.yml
```

Error Downloading Image

If you encounter the error below during the deployment:

```
"u'message': u'Error downloading image: Download of image id 77700b53-9e15-406c-b2d5-13e7d9b96551 failed: Image download failed for all URLs.',
u'code': 500,
u'type': u'ImageDownloadError',
u'details': u'Download of image id 77700b53-9e15-406c-b2d5-13e7d9b96551 failed: Image download failed for all URLs.'"
```

you should visit the Single Sign-On Settings in the Security page of iLO and change the Single Sign-On Trust Mode setting from the default of "Trust None (SSO disabled)" to "Trust by Certificate".

Using `node-inspection` can cause temporary claim of IP addresses

Possible cause: Running `node-inspection` on a node discovers all the NIC ports including the NICs that don't have any connectivity. This causes a temporary consumption of the network IPs and increased usage of the allocated quota. As a result, other nodes are deprived of IP addresses and deployments can fail.

Resolution: You can add node properties manually added instead of using the inspection tool.

Note: Upgrade `ipmitool` to a version >= 1.8.15 or it may not return detailed information about the NIC interface for `node-inspection`.

Node permanently stuck in deploying state

Possible causes:

- Ironic conductor service associated with the node could go down.
- There might be a properties mismatch. MAC address registered for the node could be incorrect.

Resolution: To recover from this condition, set the provision state of the node to `Error` and maintenance to `True`. Delete the node and re-register again.

The NICs in the baremetal node should come first in boot order

Possible causes: By default, the boot order of baremetal node is set as NIC1, HDD and NIC2. If NIC1 fails, the nodes starts booting from HDD and the provisioning fails.

Resolution: Set boot order so that all the NICs appear before the hard disk of the baremetal as NIC1, NIC2…, HDD.

Increase in the number of nodes can cause power commands to fail

Possible causes: Ironic periodically performs a power state sync with all the baremetal nodes. When the number of nodes increase, ironic does not get sufficient time to perform power operations.
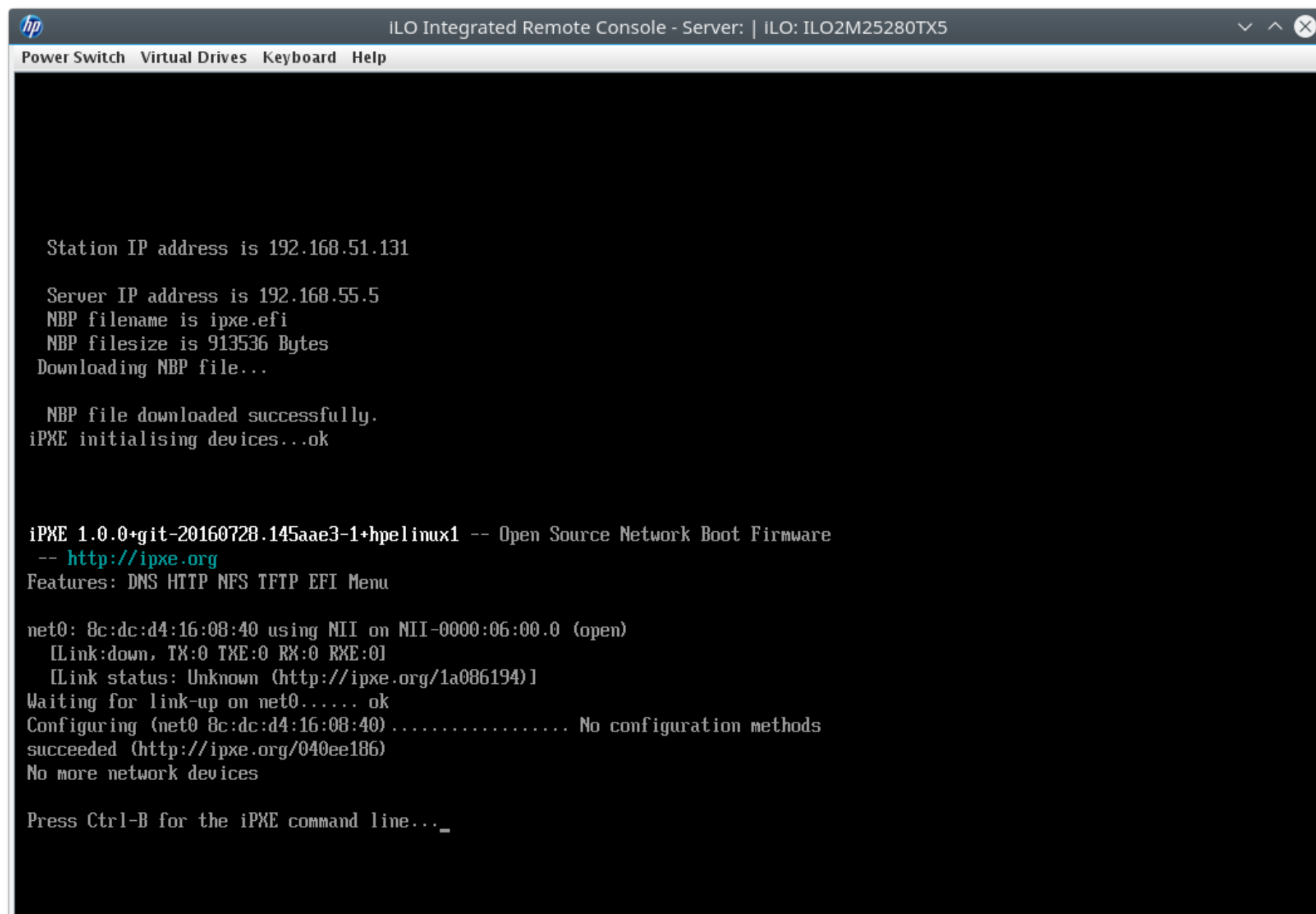
Resolution: The following procedure gives a way to increase `sync_power_state_interval`:

- Edit the file `~/helion/my_cloud/config/ironic/ironic-conductor.conf.j2` and navigate to the section for `[conductor]`
- Increase the `sync_power_state_interval`. For example, for 100 nodes, set `sync_power_state_interval = 90` and save the file.
- Execute the following set of commands to reconfigure Ironic:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
ansible-playbook -i hosts/localhost ready-deployment.yml
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts ironic-reconfigure.yml
```

DHCP succeeds with PXE but times out with iPXE

If you see DHCP error "No configuration methods succeeded" in iPXE, right after successful DHCP performed by embedded NIC firmware (please refer to screenshot below), there may be an issue with Spanning Tree Protocol on the switch.

iLO Integrated Remote Console - Server: | iLO: ILO2M25280TX5

Power Switch   Virtual Drives   Keyboard   Help

```
  Station IP address is 192.168.51.131

  Server IP address is 192.168.55.5
  NBP filename is ipxe.efi
  NBP filesize is 913536 Bytes
 Downloading NBP file...

  NBP file downloaded successfully.
iPXE initialising devices...ok



iPXE 1.0.0+git-20160728.145aae3-1+hpelinux1 -- Open Source Network Boot Firmware
 -- http://ipxe.org
Features: DNS HTTP NFS TFTP EFI Menu

net0: 8c:dc:d4:16:08:40 using NII on NII-0000:06:00.0 (open)
  [Link:down, TX:0 TXE:0 RX:0 RXE:0]
  [Link status: Unknown (http://ipxe.org/1a086194)]
Waiting for link-up on net0...... ok
Configuring (net0 8c:dc:d4:16:08:40)................ No configuration methods
succeeded (http://ipxe.org/040ee186)
No more network devices

Press Ctrl-B for the iPXE command line..._
```

To avoid this error, Rapid Spanning Tree Protocol needs to be enabled on the switch. If this is not an option due to conservative loop detection strategies, use the steps outlined below to install the iPXE binary with increased DHCP timeouts.

1. Clone iPXE source code

```
$ git clone git://git.ipxe.org/ipxe.git
$ cd ipxe/src
```

2. Modify lines 22-25 in file `config/dhcp.h`, which declare reduced DHCP timeouts (1-10 secs). Comment out lines with reduced timeouts and uncomment normal PXE timeouts (4-32)

```
//#define DHCP_DISC_START_TIMEOUT_SEC    1
//#define DHCP_DISC_END_TIMEOUT_SEC      10
#define DHCP_DISC_START_TIMEOUT_SEC    4       /* as per PXE spec */
#define DHCP_DISC_END_TIMEOUT_SEC      32      /* as per PXE spec */
```

3. Make `undionly.kpxe` (BIOS) and `ipxe.efi` (UEFI) images

```
$ make bin/undionly.kpxe
$ make bin-x86_64-efi/ipxe.efi
```

4. Copy iPXE images to lifecycle manager

```
$ scp bin/undionly.kpxe bin-x86_64-efi/ipxe.efi stack@10.0.0.4:
stack@10.0.0.4's password:
undionly.kpxe                                          100%   66KB  65.6KB/s   00:00
ipxe.efi                                               100%  918KB 918.2KB/s   00:00
```

5. From deployer, distribute image files onto all 3 controllers

```
stack@helion-cp1-c1-m1-mgmt:~$ cd ~/scratch/ansible/next/hos/ansible/

stack@helion-cp1-c1-m1-mgmt:~/scratch/ansible/next/hos/ansible$ ansible -i hosts/verb_hosts IRN-CND -m copy -b -a 'src=/home/stack/ipxe.efi dest=/tftpboot'
...
stack@helion-cp1-c1-m1-mgmt:~/scratch/ansible/next/hos/ansible$ ansible -i hosts/verb_hosts IRN-CND -m copy -b -a 'src=/home/stack/undionly.kpxe dest=/tftpboot'
...
```

There is no need to restart services. With next PXE boot attempt, iPXE binary with the increased timeout will be downloaded to the target node via TFTP.

HPE Helion OpenStack® 5.0: Node Cleaning

Cleaning is the process by which data is removed after a previous tenant has used the node. Cleaning requires use of ironic's agent_ drivers. It is extremely important to note that if the pxe_ drivers are utilized, no node cleaning operations will occur, and a previous tenant's data could be found on the node. The same risk of a previous tenant's data possibly can occur if cleaning is explicitly disabled as part of the installation.

By default, cleaning attempts to utilize ATA secure erase to wipe the contents of the disk. If secure erase is unavailable, the cleaning functionality built into the Ironic Python Agent falls back to an operation referred to as "shred" where random data is written over the contents of the disk, and then followed up by writing "0"s across the disk. This can be a time-consuming process.

An additional feature of cleaning is the ability to update firmware or potentially assert new hardware configuration, however, this is an advanced feature that must be built into the Ironic Python Agent image. Due to the complex nature of such operations, and the fact that no one size fits all, this requires a custom Ironic Python Agent image to be constructed with an appropriate hardware manager. For more information on hardware managers, see *http://docs.openstack.org/developer/ironic-python-agent/#hardware-managers*

Ironic's upstream documentation for cleaning may be found here: *http://docs.openstack.org/developer/ironic/deploy/cleaning.html*

Setup

Cleaning is enabled by default in ironic when installed via the Lifecycle Manager. You can verify this by examining the ironic-conductor.conf file. Look for:

```
[conductor]
clean_nodes=true
```

In use

When enabled, cleaning will be run automatically when nodes go from active to available state or from manageable to available. To monitor what step of cleaning the node is in, run `ironic node-show`:

```
stack@helion-cp1-c1-m1-mgmt:~$ ironic node-show 4e6d4273-2535-4830-a826-7f67e71783ed
+-----------------------+------------------------------------------------------------------+
| Property              | Value                                                            |
+-----------------------+------------------------------------------------------------------+
| target_power_state    | None                                                             |
| extra                 | {}                                                               |
| last_error            | None                                                             |
| updated_at            | 2016-04-15T09:33:16+00:00                                        |
| maintenance_reason    | None                                                             |
| provision_state       | cleaning                                                         |
| clean_step            | {}                                                               |
| uuid                  | 4e6d4273-2535-4830-a826-7f67e71783ed                             |
| console_enabled       | False                                                            |
| target_provision_state | available                                                       |
| provision_updated_at  | 2016-04-15T09:33:16+00:00                                        |
| maintenance           | False                                                            |
| inspection_started_at | None                                                             |
| inspection_finished_at | None                                                            |
| power_state           | power off                                                        |
| driver                | agent_ilo                                                        |
| reservation           | helion-cp1-c1-m1-mgmt                                            |
| properties            | {u'memory_mb': 4096, u'cpu_arch': u'amd64', u'local_gb': 80, u'cpus': 2, |
|                       | u'capabilities': u'boot_mode:uefi,boot_option:local'}            |
| instance_uuid         | None                                                             |
| name                  | None                                                             |
| driver_info           | {u'ilo_deploy_iso': u'249bf095-e741-441d-bc28-0f44a9b8cd80',     |
|                       | u'ipmi_username': u'Administrator', u'deploy_kernel':            |
|                       | u'3a78c0a9-3d8d-4764-9300-3e9c00e167a1', u'ilo_address':         |
|                       | u'10.1.196.113', u'ipmi_address': u'10.1.196.113', u'deploy_ramdisk': |
|                       | u'd02c811c-e521-4926-9f26-0c88bbd2ee6d', u'ipmi_password': u'******', |
|                       | u'ilo_password': u'******', u'ilo_username': u'Administrator'}   |
| created_at            | 2016-04-14T08:30:08+00:00                                        |
| driver_internal_info  | {u'clean_steps': None, u'hardware_manager_version':              |
|                       | {u'generic_hardware_manager': u'1.0'}, u'is_whole_disk_image': True, |
|                       | u'agent_erase_devices_iterations': 1, u'agent_url':              |
|                       | u'http://192.168.246.245:9999', u'agent_last_heartbeat': 1460633166} |
| chassis_uuid          |                                                                  |
| instance_info         | {}                                                               |
+-----------------------+------------------------------------------------------------------+
```

The status will be in the `driver_internal_info` field. You will also be able to see the `clean_steps` list there.

Troubleshooting

If an error occurs during the cleaning process, the node will enter the clean failed state so that it is not deployed. The node remains powered on for debugging purposes. The node can be moved to the manageable state to attempt a fix using the following command:

```
ironic node-set-provision-state <node id> manage
```

Once you have identified and fixed the issue, you can return the node to available state by executing the following commands:

```
ironic node-set-maintenance <node id> false
ironic node-set-provision-state <node id> provide
```

This will retry the cleaning steps and set the node to available state upon their successful completion.

Disabling Node Cleaning

To disable node cleaning, you need to edit `helion/my_cloud/definition/data/ironic/ironic_config.yml` and set `enable_node_cleaning` to false.

Commit your changes:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "disable node cleaning"
```

Deploy these changes by re-running the configuration processor and reconfigure the ironic installation:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
ansible-playbook -i hosts/localhost ready-deployment.yml
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts ironic-reconfigure.yml
```

HPE Helion OpenStack® 5.0: Installation for HPE Helion Entry-scale Cloud with Swift Only

This page describes the installation step requirements for the HPE Helion Entry-scale Cloud with Swift Only model.

Important Notes

- If you are looking for information about when to use the GUI installer and when to use the CLI, see the *Installation Overview*.
- Review the *recommended minimum hardware requirements* that we have listed.
- Review the *Release Notes* to make yourself aware of any known issues and limitations.
- The installation process can occur in different phases. For example, you can install the control plane only and then add Compute nodes afterwards if you would like.
- If you run into issues during installation, we have put together a list of *Installation Troubleshooting Steps* you can reference.
- Make sure all disks on the system(s) are wiped before you begin the install. (For Swift, refer to *Swift Requirements for Device Group Drives*)
- There is no requirement to have a dedicated network for OS-install and system deployment, this can be shared with the management network. More information can be found on the *Example Configuration* page.
- You may see the terms deployer and lifecycle manager used interchangeably. These are referring to the same nodes in your environment.
- When running the Ansible playbook in this installation guide, if a runbook fails you will see in the error response to use the `--limit` switch when retrying a playbook. This should be avoided. You can simply re-run any playbook without this switch.
- DVR is not supported with ESX compute.
- When you attach a Cinder volume to the VM running on the ESXi host, the volume will not get detected automatically. Make sure to set the image metadata **vmware_adaptertype=lsiLogicsas** for image before launching the instance. This will help to discover the volume change appropriately.

Before You Start

We have put together a *Pre-Installation Checklist* that should help with the recommended pre-installation tasks.

Set up the Lifecycle Manager

Installing the Lifecycle Manager

The lifecycle manager will contain the installation scripts and configuration files to deploy your cloud. You can set up the lifecycle manager on a dedicated node or you do so on your first controller node. The default choice is to use the first controller node as the lifecycle manager.

1. Sign in and download the product and signature files at the link below:

   a. *Software Entitlement Portal*
2. You can verify the download was complete via the signature verification process outlined *here*.
3. Boot your lifecycle manager from the ISO contained in the download.
4. Enter "install" to start installation.

> 📝 **Note:** "install" is all lower case

5. Select the language. Note that only the English language selection is currently supported.
6. Select the location.
7. Select the keyboard layout.
8. Select the primary network interface, if prompted:

   a. Assign IP address, subnet mask, and default gateway
9. Create new account:

   a. Enter a username.
   b. Enter a password.
   c. Enter time zone.

Once the initial installation is finished, complete the lifecycle manager setup with these steps:

1. Ensure your lifecycle manager has a valid DNS nameserver specified in `/etc/resolv.conf`.
2. Set the environment variable LC_ALL:

```
export LC_ALL=C
```

> 📝 **Note:** This can be added to `~stack/.bashrc` or `/etc/bash.bashrc`.

At the end of this section you should have a node set up with Linux for HPE Helion on it.

Configure and Run the Lifecycle Manager

> ⚠️ **Important:** It is critical that you don't run any of the commands below as the `root` user or use `sudo`, unless it is stated explicitly in the steps. Run then as the user you just created (or `stack` if you left the default of "stack").

1. Log into your lifecycle manager as the user you created and mount the install media at `/media/cdrom`. It may be necessary to use `wget` or another file transfer method to transfer the install media to the lifecycle manager before completing this step. Here is the command to mount the media:

   Example for HPE Helion OpenStack® 5.0

   ```
   sudo mount HelionOpenStack-5.0.iso /media/cdrom
   ```
2. Unpack the tarball that is in the `/media/cdrom/hos/` directory:

   Example for HPE Helion OpenStack 5.0

   ```
   tar xvf /media/cdrom/hos/hos-5.0.0-<timestamp>.tar
   ```
3. Run the hos-init.bash script which is included in the build:

   Example for HPE Helion OpenStack® 5.0

   ```
   ~/hos-5.0.0/hos-init.bash
   ```

   You will be prompted to enter an optional SSH passphrase when running `hos-init.bash`. This passphrase is used to protect the key used by Ansible when connecting to its client nodes. If you do not want to use a passphrase then just press return at the prompt.

   For automated installation (e.g. CI) it is possible to disable SSH passphrase prompting by setting the `HOS_INIT_AUTO` environment variable before running `hos-init.bash`, like this:

   ```
   export HOS_INIT_AUTO=y
   ```

If you have protected the SSH key with a passphrase then execute the following commands to avoid having to enter the passphrase on every attempt by Ansible to connect to its client nodes:

```
eval $(ssh-agent)
ssh-add ~/.ssh/id_rsa
```

At the end of this section you should have a local directory structure, as described below:

```
~/helion/                    Top level directory
~/helion/examples/           Directory contains the config input files of the example clouds
~/helion/my_cloud/definition/  Directory contains the config input files
~/helion/my_cloud/config/    Directory contains .j2 files which are symlinks to the ~/helion/hos/ansible directory
~/helion/hos/                Directory contains files used by the installer
```

⚠️ **Warning:** It's important that you do not add any extra files in the ~/helion directory on your lifecycle manager. This includes temporary save files. Doing so will cause errors during the installation.

⚠️ **Important:** You should be aware that the files in the ~/helion/my_cloud/config/ directory are symlinks to the ~/helion/hos/ansible/ directory. For example, ~/helion/my_cloud/config/keepalived/defaults.yml is a symbolic link to ~/helion/hos/ansible/roles/keepalived/defaults/main.yml

```
ls -al ~/helion/my_cloud/config/keepalived/defaults.yml
lrwxrwxrwx 1 stack stack 55 May 24 20:38 /home/stack/helion/my_cloud/config/keepalived/defaults.yml -> ../../../hos/ansible/roles/keepalived/defaults/main.yml
```

If you are using a tool like sed to make edits to files in this directory, you might break the symbolic link and create a new copy of the file. To maintain the link, you will need to force sed to follow the link:

```
sed -i --follow-symlinks 's$keepalived_vrrp_offset: 0$keepalived_vrrp_offset: 2$' ~/helion/my_cloud/config/keepalived/defaults.yml
```

Alternatively, you could directly edit the target of the link ~/helion/hos/ansible/roles/keepalived/defaults/main.yml.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Configure Your Environment

This part of the install is going to depend on the specific cloud configuration you are going to use.

1. Setup your configuration files, as follows:

   a. See the sample sets of configuration files in the ~/helion/examples/ directory. Each set will have an accompanying README.md file that explains the contents of each of the configuration files.

   b. Copy the example configuration files into the required setup directory and edit them to contain the details of your environment:

   ```
   cp -r ~/helion/examples/entry-scale-swift/* ~/helion/my_cloud/definition/
   ```

   c. Begin inputting your environment information into the configuration files in the ~/helion/my_cloud/definition directory.

   Full details of how to do this can be found here: *HPE Helion OpenStack 5.0 Input Model*.

   In many cases, the example models provide most of the data you need to create a valid input model. However, there are two important aspects you must plan and configure before starting a deploy as follows:

   • Check the disk model used by your nodes. Specifically, check that all disk drives are correctly named and used as described in *Swift Requirements for Device Group Drives*.
   • Select an appropriate partition power for your rings. Detailed information about this is provided at *Swift Ring Specifications*.

Optionally, you can use the hosencrypt.py script to encrypt your iLo passwords. This script uses OpenSSL.

   a. Change to the Ansible directory:

   ```
   cd ~/helion/hos/ansible
   ```

   b. Put the encryption key into the following environment variable:

   ```
   export HOS_USER_PASSWORD_ENCRYPT_KEY=<encryption key>
   ```

   c. Run the python script below and follow the instructions. Enter a password that you want to encrypt.

   ```
   hosencrypt.py
   ```

   d. Take the string generated and place it in the "ilo_password" field in your ~/helion/my_cloud/definition/data/servers.yml file, remembering to enclose it in quotes.

e. Repeat the above for each server.

📝 **Note:** Before you run any playbooks, remember that you need to export the encryption key in the following environment variable: export HOS_USER_PASSWORD_ENCRYPT_KEY=<encryption key>

Commit your configuration to the *local git repo*, as follows:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

⚠️ **Important:** This step needs to be repeated any time you make changes to your configuration files before you move onto the following steps. See *Using Git for Configuration Management* for more information.

Run the Configuration Processor

Once you have your configuration files setup you need to run the configuration processor to complete your configuration.

When you run the configuration processor you will be prompted for two passwords. Enter the first password to make the configuration processor encrypt its sensitive data, which is comprised of the random inter-service passwords that it generates and the ansible group_vars and host_vars that it produces for subsequent deploy runs. You will need this password for subsequent ansible deploy and configuration processor runs. If you wish to change an encryption password that you have already used when running the configuration processor then enter the new password at the second prompt, otherwise just press carriage return to bypass this.

Run the configuration processor with this command:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

For automated installation (e.g. CI) you can specify the required passwords on the ansible command line. For example, the command below will disable encryption by the configuration processor

```
ansible-playbook -i hosts/localhost config-processor-run.yml -e encrypt="" -e rekey=""
```

If you receive an error during this step then there is probably an issue with one or more of your configuration files. We recommend that you verify that all of the information in each of your configuration files is correct for your environment and then commit those changes to git using the instructions in the previous section before re-running the configuration processor again.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Provision Your Baremetal Nodes

To provision the baremetal nodes in your cloud deployment you can either use the automated operating system installation process provided by HPE Helion OpenStack or you can use the 3rd party installation tooling of your choice. We will outline both methods below:

Using 3rd Party Baremetal Installers

If you do not wish to use the automated operating system installation tooling included with 5.0 then the requirements that have to be met using the installation tooling of your choice are:

- The operating system must be installed via the HPE Linux for HPE Helion OpenStack ISO provided on the *Software Entitlement Portal*.
- Each node must have SSH keys in place that allows the same user from the lifecycle manager node who will be doing the deployment to SSH to each node without a password.
- Passwordless sudo needs to be enabled for the user.
- There should be a LVM logical volume as /root on each node.
- If the LVM volume group name for the volume group holding the "root" LVM logical volume is hlm-vg then it will align with the disk input models in the examples.
- Ensure that openssh-server, python, python-apt, and rsync are installed.

If you chose this method for installing your baremetal hardware, skip forward to the *Run the Configuration Processor* step.

If you would like to use the automated operating system installation tools provided by 5.0 then complete all of the steps below.

Using the Automated Operating System Installation Provided by HPE Helion OpenStack

Part One: Deploy Cobbler

This phase of the install process takes the baremetal information that was provided in servers.yml and installs the Cobbler provisioning tool and loads this information into Cobbler. This sets each node to netboot-enabled: true in Cobbler. Each node will be automatically marked as netboot-enabled: false when it completes its operating system install successfully. Even if the node tries to PXE boot subsequently, Cobbler will not serve it. This is deliberate so that you can't reimage a live node by accident.

The `cobbler-deploy.yml` playbook prompts for a password - this is the password that will be encrypted and stored in Cobbler, which is associated with the user running the command on the lifecycle manager, that you will use to log in to the nodes via their consoles after install. The username is the same as the user set up in the initial dialogue when installing the lifecycle manager from the iso, and is the same user that is running the cobbler-deploy play.

1. Run the following playbook which confirms that there is iLo connectivity for each of your nodes so that they are accessible to be re-imaged in a later step:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost bm-power-status.yml
```

2. Run the following playbook to deploy Cobbler:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost cobbler-deploy.yml
```

Part Two: Image the Nodes

This phase of the install process goes through a number of distinct steps:

1. Powers down the nodes to be installed
2. Sets the nodes hardware boot order so that the first option is a network boot.
3. Powers on the nodes. (The nodes will then boot from the network and be installed using infrastructure set up in the previous phase)
4. Waits for the nodes to power themselves down (this indicates a success install). This can take some time.
5. Sets the boot order to hard disk and powers on the nodes.
6. Waits for the nodes to be ssh-able and verifies that they have the signature expected.

The reimage command is:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost bm-reimage.yml [-e nodelist=node1,node2,node3]
```

If a nodelist is not specified then the set of nodes in cobbler with `netboot-enabled: True` is selected. The playbook pauses at the start to give you a chance to review the set of nodes that it is targeting and to confirm that it's correct.

You can use the command below which will list all of your nodes with the `netboot-enabled: True` flag set:

```
sudo cobbler system find --netboot-enabled=1
```

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Deploy the Cloud

1. Use the playbook below to create a deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

2. [OPTIONAL] - Run the `wipe_disks.yml` playbook to ensure all of your partitions on your nodes are completely wiped before continuing with the installation. If you are using fresh machines this step may not be necessary.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts wipe_disks.yml
```

If you have used an encryption password when running the configuration processor use the command below and enter the encryption password when prompted:

```
ansible-playbook -i hosts/verb_hosts wipe_disks.yml --ask-vault-pass
```

3. Run the `site.yml` playbook below:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts site.yml
```

If you have used an encryption password when running the configuration processor use the command below and enter the encryption password when prompted:

```
ansible-playbook -i hosts/verb_hosts site.yml --ask-vault-pass
```

Note: The step above runs `osconfig` to configure the cloud and `hlm-deploy` to deploy the cloud. Therefore, this step may run for a while, perhaps 45 minutes or more, depending on the number of nodes in your environment.

4. Verify that the network is working correctly. Ping each IP in the `/etc/hosts` file from one of the controller nodes.

For any troubleshooting information regarding these steps, see *Troubleshooting Your Installation*

Post-Installation Verification and Administration

We recommend verifying the installation using the *Cloud Verification* page.

There are also a list of other common post-installation administrative tasks listed in the *Common Post-Installation Tasks* list.

HPE Helion OpenStack® 5.0: Installing SLES Compute

HPE Helion OpenStack® 5.0: SLES Compute Node Installation Overview

Introduction

HPE Helion OpenStack 5.0 supports SLES compute nodes, specifically SLES 12 SP2. HPE does not ship a SLES ISO with HPE Helion OpenStack so you will need to download a copy of the SLES iso (SLE-12-SP2-Server-DVD-x86_64-GM-DVD1.iso) and the SLES SDK ISO (SLE-12-SP2-SDK-DVD-x86_64-GM-DVD1.iso) from SUSE. You can use the following link to download the ISO. To do so, you will need to sign in or create a SUSE customer service account before downloading: *https://www.suse.com/products/server/ download/*

There are two approaches for deploying SLES compute nodes in HPE Helion OpenStack:

- Using the lifecycle manager to automatically deploy SLES Compute nodes.
- Provisioning SLES nodes yourself, either manually or using a 3rd party tool, and then provide the relevant information to the lifecycle manager.

These two approaches can be used whether you are installing a cloud for the first time or adding a compute node to an existing cloud. Regardless of your approach, you should be certain to register your SLES compute nodes in order to get product updates as they come available. See *Register SUSE Linux* for more information.

HPE Helion OpenStack® 5.0: SLES Support

SLES Support

SUSE Linux Enterprise Server (SLES) Host OS KVM and/or supported SLES guests have been tested and qualified by HPE to run on HPE Helion OpenStack. Refer to the published compatibility matrix for your version of HPE Helion OpenStack at the following URL *http://docs.hpcloud.com/#home.html*. HPE is one of SUSE's largest OEMs with follow the sun global support coverage.

- **One Number to Call**

  HPE customers who have purchased both HPE Helion OpenStack and SLES subscriptions with support from HPE will have one number to call for troubleshooting, fault isolation and support from HPE technical support specialists in HPE Helion OpenStack and SUSE technologies. If the problem is isolated to SLES software itself the issue will be replicated on a SUSE certified platform and escalated to SUSE for resolution.

- **A Dual Support Model**

  HPE will troubleshoot and fault isolate an issue at the Helion software level. If HPE Helion OpenStack software is excluded as the cause of the problem, then customers who did not purchase SLES support from HPE will be directed to the vendor from whom they purchased SLES for continued support.

HPE Helion OpenStack® 5.0: Using the Lifecycle Manager to Deploy SLES Compute Nodes

The method used for deploying SLES compute nodes using Cobbler on the lifecycle manager uses legacy BIOS.

Note: UEFI and Secure boot are currently not supported on SLES compute.

Deploying legacy BIOS SLES compute nodes

The installation process for SLES nodes is almost identical to that of HPE Linux nodes as described in the topic for *Installation for Helion Entry-scale KVM Cloud"*. The key differences are:

- The standard SLES ISO (SLE-12-SP2-Server-DVD-x86_64-GM-DVD1.iso) must be accessible via `/home/stack/sles12.iso`. Rename the ISO or create a symbolic link.

```
mv SLE-12-SP2-Server-DVD-x86_64-GM-DVD1.iso /home/stack/sles12.iso
```

- The contents of the SLES SDK ISO (SLE-12-SP2-SDK-DVD-x86_64-GM-DVD1.iso) must be mounted or copied to `/opt/hlm_packager/hlm/sles12/zypper/SDK/`. If you choose to mount the ISO, we recommend creating an `/etc/fstab` entry to ensure the ISO is mounted after a reboot.
- You must identify the node(s) on which you want to install SLES, by adding the key/value pair `distro-id: sles12sp2-x86_64` to server details in `servers.yml`. You will also need to update `net_interfaces.yml`, `server_roles.yml`, `disk_compute.yml` and `control_plane.yml`. For more information on configuration of the Input Model for SLES, see *SLES Compute Model*.
- HPE Helion OpenStack playbooks currently do not take care of the SDK, therefore it needs to be added manually. The following command needs to be run on every SLES compute node:

```
deployer_ip=192.168.10.254
zypper addrepo --no-gpgcheck --refresh http://$deployer_ip:79/hlm/sles12/zypper/SDK SLES-SDK
```

There is no need to add OS repo as in case of *Provisioning SLES Yourself*, because SLES already comes with OS repo populated after a Cobbler managed install.

HPE Helion OpenStack® 5.0: Provisioning SLES Yourself

Introduction

This article outlines the steps needed to manually provision a SLES node so that it can be added to a new or existing HPE Helion OpenStack 5.0 cloud.

Configure Lifecycle Manager to Enable SLES

- Take note of the lifecycle manager's IP address. It will be used below during *Add zypper repository*.
- Mount or copy the contents of SLE-12-SP2-Server-DVD-x86_64-GM-DVD1.iso to `/opt/hlm_packager/hlm/sles12/zypper/OS/`
- Mount or copy the contents of SLE-12-SP2-SDK-DVD-x86_64-GM-DVD1.iso to `/opt/hlm_packager/hlm/sles12/zypper/SDK/`

Note: If you choose to mount an ISO, we recommend creating an /etc/fstab entry to ensure the ISO is mounted after a reboot.

Install SLES 12 SP2

Install SLES 12 SP2 using the standard iso (SLE-12-SP2-Server-DVD-x86_64-GM-DVD1.iso)

Assign a static IP

1. Use the `ip addr` command to find out what network devices are on your system:

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether f0:92:1c:05:89:70 brd ff:ff:ff:ff:ff:ff
    inet 10.13.111.178/26 brd 10.13.111.191 scope global eno1
       valid_lft forever preferred_lft forever
    inet6 fe80::f292:1cff:fe05:8970/64 scope link
       valid_lft forever preferred_lft forever
3: eno2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether f0:92:1c:05:89:74 brd ff:ff:ff:ff:ff:ff
```

2. Identify the one that matches the MAC address of your server and edit the corresponding config file in `/etc/sysconfig/network-scripts`.

```
vi /etc/sysconfig/network-scripts/ifcfg-eno1
```

3. Edit the `IPADDR` and `NETMASK` values to match your environment. Note that the `IPADDR` is used in the corresponding stanza in `servers.yml`. You may also need to set `BOOTPROTO` to `none`.

```
TYPE=Ethernet
BOOTPROTO=none
```

```
DEFROUTE=yes
PEERDNS=yes
PEERROUTES=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
IPV6_FAILURE_FATAL=no
NAME=eno1
UUID=36060f7a-12da-469b-a1da-ee730a3b1d7c
DEVICE=eno1
ONBOOT=yes
NETMASK=255.255.255.192
IPADDR=10.13.111.14
```

4. [OPTIONAL] Reboot your SLES node and ensure that it can be accessed from the lifecycle manager.

Add `stack` user and home directory

```
useradd -m stack
passwd stack
```

Allow user `stack` to `sudo` without password

Setting up sudo on SLES is covered in the SUSE documentation, *Configuring sudo*.

The recommendation is to create user specific sudo config files under /etc/sudoers.d, therefore creating an /etc/sudoers.d/stack config file with the following content will allow sudo commands without the requirement of a password.

```
stack ALL=(ALL) NOPASSWD:ALL
```

Add zypper repository

Using the ISO-based repositories created above, add the zypper repositories.

Follow these steps. Update the value of deployer_ip as necessary.

```
deployer_ip=192.168.10.254
zypper addrepo --no-gpgcheck --refresh http://$deployer_ip:79/hlm/sles12/zypper/OS SLES-OS
zypper addrepo --no-gpgcheck --refresh http://$deployer_ip:79/hlm/sles12/zypper/SDK SLES-SDK
```

To verify that the repositories have been added, run:

```
zypper repos --detail
```

You can find more information on Using Zypper in the SLES 12 Admin guide: *https://www.suse.com/documentation/sles-12/book_sle_admin/data/sec_zypper.html*

⚠ **Warning:** If you intend on attaching encrypted volumes to any of your SLES Compute nodes, you'll need to install the crypt libraries through cryptsetup on each node. Run the following command to install the necessary crypt libraries:

```
sudo zypper in cryptsetup
```

Add Required Packages

As documented in the *Using 3rd Party Baremetal Installers* section of *Installation for Helion Entry-scale KVM Cloud*, you will need to add some extra packages that are required. Ensure that `openssh-server`, `python`, and `rsync` are installed.

Set up passwordless SSH access

Once you have started your installation using the lifecycle manager, or if you are adding a SLES node to an existing cloud, you need to copy the lifecycle manager public key to the SLES node. One way of doing this is to copy the `/home/stack/.ssh/` `authorized_keys` from another node in the cloud to the same location on the SLES node. If you are installing a new cloud, this file will be available on the nodes after running the `bm-reimage.yml` playbook.

⚠ **Important:** Ensure that there is global read access to the file `/home/stack/.ssh/authorized_keys`.

Now test passwordless ssh from the deployer and check your ability to remotely execute sudo commands:

```
ssh  stack@<<ip of sles node>> "sudo tail -5 /var/log/messages"
```

HPE Helion OpenStack® 5.0: Using SLES as a Ceph Client

⚠ **Warning:** SUSE Linux Enterprise Server 12 SP2 comes with Ceph packages from the Ceph Jewel release. HPE Helion OpenStack 5.0 Ceph server is based on the Ceph Hammer release, therefore it is not advised that client related Ceph management commands from a SLES compute node be performed. Doing so could trigger an update of the Ceph config which would not be compatible with other non-SLES Ceph clients in the same HPE Helion OpenStack 5.0 system.

HPE Helion OpenStack® 5.0: Installing RHEL Compute

HPE Helion OpenStack® 5.0: RHEL Compute Node Installation Overview

Introduction

HPE Helion OpenStack 5.0 supports RHEL compute nodes, specifically RHEL 7.2. HPE does not ship a RedHat iso with HPE Helion OpenStack so you will need to provide a copy of the standard RHEL 7.2 iso (RHEL-7.2-20151030.0-Server-x86_64-dvd1.iso) that can be downloaded from the RedHat website.

There are two approaches for deploying RHEL compute nodes in HPE Helion OpenStack:

- Using the lifecycle manager to automatically configure and deploy RHEL Compute nodes.
- Provisioning RHEL nodes yourself, either manually or using a 3rd party tool, and then provide the relevant information to the lifecycle manager .

These two approaches can be used whether you are installing a cloud for the first time or adding a compute node to an existing cloud.

HPE Helion OpenStack® 5.0: RHEL Support

RHEL Support

RedHat Enterprise Linux (RHEL) Host OS KVM and/or supported RHEL guests have been tested and qualified by HPE to run on HPE Helion OpenStack. Refer to the published compatibility matrix for your version of HPE Helion OpenStack at the following URL *http://docs.hpcloud.com/#home.html*. HPE is one of RedHat's largest OEMs with follow the sun global support coverage.

- **One Number to Call**

  HPE customers who have purchased both HPE Helion OpenStack and RHEL subscriptions with support from HPE will have one number to call for troubleshooting, fault isolation and support from HPE technical support specialists in HPE Helion OpenStack and RedHat technologies. If the problem is isolated to RHEL software itself the issue will be replicated on a RedHat certified platform and escalated to RedHat for resolution.

- **A Dual Support Model**

  HPE will troubleshoot and fault isolate an issue at the Helion software level. If HPE Helion OpenStack software is excluded as the cause of the problem, then customers who did not purchase RHEL support from HPE will be directed to the vendor from whom they purchased RHEL for continued support.

HPE Helion OpenStack® 5.0: Using the Lifecycle Manager to Deploy RHEL Compute Nodes

The method used for deploying RHEL compute nodes using Cobbler on the lifecycle manager will depend on whether your RHEL nodes are using legacy BIOS or UEFI. We provide steps for both.

Deploying legacy BIOS RHEL compute nodes

The installation process for RHEL nodes is almost identical to that of HPE Linux nodes as described in the topic for *Installation for Helion Entry-scale KVM Cloud"*. The key differences are:

- The standard RHEL 7.2 iso must be placed in the `stack` home directory on the deployer node before you begin the installation procedure. The file must be named `rhel7.iso`.
- You must identify the node(s) on which you want to install RHEL, by adding the key/value pair `distro-id: rhel72-x86_64` to server details in `servers.yml`. You will also need to update `net_interfaces.yml`, `server_roles.yml`, `disk_compute.yml` and `control_plane.yml`. For more information on configuration of the Input Model for RHEL, see *RHEL Compute Model*.

Deploying UEFI RHEL compute nodes

HPE Helion OpenStack 3.0 required a manual procedure if you wanted to use the lifecycle manager to install RHEL 7.2 on UEFI nodes as described in the article *here*. This procedure has been automated in the HPE Helion OpenStack 5.0 lifecycle manager in `prepare-rhel-loader.yml` and `prepare-rhel-grub2.yml` playbooks..

Before you attempt to use the lifecycle manager to install RHEL on UEFI nodes, you should install any Linux for HPE Helion nodes or any RHEL on legacy BIOS nodes first.

Execute the following steps to re-image one or more nodes after you have run the `ready-deployment.yml` playbook:

1. Prior to beginning these steps, the following must be true:

   • All of your nodes using Linux for HPE Helion (controller, VSA, etc nodes) must already be installed, either manually for via Cobbler.
   • Your input model should be configured for your RHEL nodes, per the instructions at *RHEL Compute Model*.
   • You should have run the configuration processor and the `ready-deployment.yml` playbook.

2. Run the following playbook, which will extract the `grubx64.efi` file which will be needed when imaging your RHEL UEFI nodes.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook prepare-rhel-loader.yml
```

3. Run the following playbook, ensuring that you specify only your UEFI RHEL nodes using the nodelist. This playbook will reconfigure Cobbler for the nodes listed.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook prepare-rhel-grub2.yml -e nodelist=node1[,node2,node3]
```

4. Re-image the node(s), ensuring that you only specify your UEFI RHEL nodes using the nodelist.

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost bm-reimage.yml -e nodelist=node1[,node2,node3]
```

5. Make backups of the `grub.cfg-*` and `grubx64.efi` files in `/srv/tftp/grub/` as they will be overwritten when running the cobbler-deploy playbook on the next step. You will need these files if you need to reimage the nodes in the future.

6. Run the `cobbler-deploy` playbook, which will reset Cobbler back to the default values:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost cobbler-deploy.yml
```

UEFI Secure Boot

Secure Boot is a method used to restrict binaries execution for booting the system. With this option enabled, system BIOS will only allow boot loaders with trusted cryptographic signatures to be executed, thus enable preventing malware from hiding embedded code in the boot chain. Each boot loader launched during the boot process is digitally signed and that signature is validated against a set of trusted certificates embedded in the UEFI BIOS. Secure Boot is completely implemented in the BIOS and does not require special hardware.

Thus Secure Boot is:

1. Intended to prevent boot-sector malware or kernel code injection.
2. Hardware-based code signing.
3. Extension of the UEFI BIOS architecture.
4. Optional with the ability to enable or disable it through the BIOS.

In Boot Options of RBSU, `Boot Mode` needs to be set to `UEFI Mode` and `UEFI Optimized Boot` should be `Enabled`.

Secure Boot is enabled at

```
System Configuration > BIOS/Platform Configuration (RBSU) > Server Security > Secure Boot Configuration > Secure Boot Enforcement
```

HPE Helion OpenStack® 5.0: Provisioning RHEL Yourself

Introduction

This article outlines the steps needed to manually provision a RHEL node so that it can be added to a new or existing HPE Helion OpenStack 5.0 cloud.

Install RHEL 7.2

Install RHEL 7.2 using the standard iso (RHEL-7.2-20151030.0-Server-x86_64-dvd1.iso)

Assign a static IP

1. Use the `ip addr` command to find out what network devices are on your system:

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether f0:92:1c:05:89:70 brd ff:ff:ff:ff:ff:ff
    inet 10.13.111.178/26 brd 10.13.111.191 scope global eno1
       valid_lft forever preferred_lft forever
    inet6 fe80::f292:1cff:fe05:8970/64 scope link
       valid_lft forever preferred_lft forever
3: eno2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether f0:92:1c:05:89:74 brd ff:ff:ff:ff:ff:ff
```

2. Identify the one that matches the MAC address of your server and edit the corresponding config file in `/etc/sysconfig/network-scripts`.

```
vi /etc/sysconfig/network-scripts/ifcfg-eno1
```

3. Edit the `IPADDR` and `NETMASK` values to match your environment. Note that the `IPADDR` is used in the corresponding stanza in `servers.yml`. You may also need to set `BOOTPROTO` to `none`.

```
TYPE=Ethernet
BOOTPROTO=none
DEFROUTE=yes
PEERDNS=yes
PEERROUTES=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
IPV6_FAILURE_FATAL=no
NAME=eno1
UUID=36060f7a-12da-469b-a1da-ee730a3b1d7c
DEVICE=eno1
ONBOOT=yes
NETMASK=255.255.255.192
IPADDR=10.13.111.14
```

4. [OPTIONAL] Reboot your RHEL node and ensure that it can be accessed from the lifecycle manager.

Add `stack` user and home directory

```
useradd -m stack
passwd stack
```

Allow user `stack` to `sudo` without password

There are a number of different ways to achieve this - here is one possibility using the pre-existing `wheel` group.

1. Add the `stack` user to the `wheel` group.

```
usermod -aG wheel stack
```

2. Run the `visudo` command
3. Uncomment the line specifying `NOPASSWD: ALL` for the `wheel` group.

```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)        ALL

## Same thing without a password
 %wheel ALL=(ALL)        NOPASSWD: ALL
```

4. To facilitate using ssh from the deployer and running a command via sudo, comment out the lines for `requiretty` and `!visiblepw`

```
#
# Disable "ssh hostname sudo <cmd>", because it will show the password in clear.
#       You have to run "ssh -t hostname sudo <cmd>".
#
#Defaults    requiretty

#
# Refuse to run if unable to disable echo on the tty. This setting should also be
# changed in order to be able to use sudo without a tty. See requiretty above.
#
#Defaults    !visiblepw
```

Set up yum repository

You need to set up a yum repository, either external or local, containing a HPE Helion OpenStack supported RHEL distro. You must have a full repository including ResilientStorage and HighAvailability addons. One possible method for setting up a local repository is outlined in this section.

1. Mount the RHEL iso and expand it

```
mkdir /tmp/localrhel
mount -o loop rhel7.iso /mnt
cd /mnt
tar cvf - . | (cd /tmp/localrhel ; tar xvf -)
cd /
umount /mnt
```

2. Create a repo file `/etc/yum.repos.d/localrhel.repo`

```
[localrhel]
name=localrhel
baseurl=file:///tmp/localrhel
enabled=1
gpgcheck=0

[localrhel-1]
name=localrhel-1
baseurl=file:///tmp/localrhel/addons/ResilientStorage
enabled=1
gpgcheck=0

[localrhel-2]
name=localrhel-2
baseurl=file:///tmp/localrhel/addons/HighAvailability
enabled=1
gpgcheck=0
```

3. Run `yum clean all`.

Add Required Packages

As documented in the *Using 3rd Party Baremetal Installers* section of *Installation for Helion Entry-scale KVM Cloud*, you will need to add some extra packages that are required. Ensure that `openssh-server`, `python`, and `rsync` are installed.

Set up passwordless SSH access

Once you have started your installation using the lifecycle manager, or if you are adding a RHEL node to an existing cloud, you need to copy the deployer public key to the RHEL node. One way of doing this is to copy the `/home/stack/.ssh/authorized_keys` from another node in the cloud to the same location on the RHEL node. If you are installing a new cloud, this file will be available on the nodes after running the `bm-reimage.yml` playbook.

⚠️ **Important:** Ensure that there is global read access to the file `/home/stack/.ssh/authorized_keys`.

Now test passwordless ssh from the deployer and check your ability to remotely execute sudo commands:

```
ssh  stack@<<ip of rhel node>> "sudo tail -5 /var/log/messages"
```

HPE Helion OpenStack® 5.0: Using RHEL as a Ceph Client

HPE Helion OpenStack 5.0 supports the use of a RHEL compute node (specifically RHEL 7.2) as a Ceph client node. However, the rpm packages (to be installed on the RHEL compute nodes) to enable it to act as a Ceph client are not shipped with HPE Helion OpenStack 5.0. So, you will need to provide the necessary rpm packages on the HPE Helion OpenStack lifecycle manager node. This document describes the steps to achieve this.

📄 **Note:**

1. Upgrade: If you are upgrading your HPE Helion OpenStack cloud to 5.0, you should execute the steps in the following *section* **before** executing the `hlm-upgrade.yml` playbook (so that the existing RHEL compute nodes upgrade to the appropriate version of Ceph client).

2. New Install: If you are deploying a new HPE Helion OpenStack 5.0 cloud, you should execute the steps in the following *section* **before** running the `site.yml` playbook.

3. Existing 5.0 Cloud: If you alrady have a HPE Helion OpenStack 5.0 cloud, you should execute the steps in the following *section* and the execute these commands to upgrade the Ceph client package on your RHEL compute nodes.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts osconfig-run.yml
ansible-playbook -i hosts/verb_hosts ceph-client-prepare.yml
```

Setting up a yum repo on Lifecycle Manager Node for Hosting the Ceph Client Packages for RHEL 7.2

On the lifecycle manager, execute the following steps:

1. Create the directory where the RHEL packages will be downloaded:

```
mkdir -p ~/third-party/ceph/pkgs/rhel/
cd ~/third-party/ceph/pkgs/rhel/
```

2. Download the following packages into the `~/third-party/ceph/pkgs/rhel/` directory:

```
ceph-common-0.94.7-0.el7.x86_64.rpm
libbabeltrace-1.2.4-3.el7.x86_64.rpm
libcephfs1-0.94.7-0.el7.x86_64.rpm
librados2-0.94.7-0.el7.x86_64.rpm
librbd1-0.94.7-0.el7.x86_64.rpm
lttng-ust-2.4.1-1.el7.x86_64.rpm
python-cephfs-0.94.7-0.el7.x86_64.rpm
python-rados-0.94.7-0.el7.x86_64.rpm
python-rbd-0.94.7-0.el7.x86_64.rpm
userspace-rcu-0.7.16-1.el7.x86_64.rpm
```

3. Execute the following commands to deploy and populate the yum repo on the lifecycle manager with above packages:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost third-party-deploy.yml
ansible-playbook -i hosts/localhost third-party-import.yml
```

HPE Helion OpenStack® 5.0: Integrations

Once you have completed your cloud installation, these are some of the common integrations you may want to perform.

HPE Helion OpenStack® 5.0: Ceph Overview

HPE Helion OpenStack® 5.0 supports the Hammer version of Ceph cluster. Ceph cluster is a distributed object storage solution that can scale horizontally up to multiple petabytes and is based on the Reliable Autonomic Distributed Object Store (RADOS) object-based storage system.

Overview

HPE Helion OpenStack® 5.0 supports the Hammer version of Ceph cluster. Ceph cluster is a distributed object storage solution that can scale horizontally up to multiple petabytes and is based on the Reliable Autonomic Distributed Object Store (RADOS) object-based storage system. Ceph cluster stores all data as objects. It also provides components that support other storage protocols. For example: RBD (RADOS Block Devices) supports block storage protocol, RADOS Gateway support S3/Swift API protocol for object storage access and so on. For example, RBD (RADOS Block Devices) supports the block storage protocol, RADOS Gateway supports the S3/Swift API protocol for object storage access, and so on. The following are Ceph's key features :

1. Uses of any commodity hardware.
2. Scales horizontally upto a petabytes (there is no theoretical limit).
3. Self healing, self managing.
4. No single point of failure.
5. Supports various client protocol for block device access, object storage access using the S3/Swift API and more.

Ceph deployment is flexible. Although it supports a variety of storage protocols based on deployed components, you can add extra components (except mandatory ones) *only* if you need to support specific storage protocols. For example, if you are not going to support the S3/Swift protocol, then you might choose not to deploy the RADOS Gateway. For easy readability, the various aspects of cluster deployment, such as hardware configuration, service configuration parameters, and deployment architecture, are categorized into separate sections. Each major section is further categorized into following segments:

• Core Ceph
• RADOS Gateway

Each sub-section provides recommendations vs. supported configurations. We recommend the production configuration, although all supported configurations are tested and validated. Use alternative supported configurations only if you have a strong need for them after evaluating their pros and cons.

Core Ceph

Core Ceph has two primary components, Object Storage Daemon (OSD) and Monitor, which are mandatory for cluster function. The following table briefly describes these components.

Components	Description
OSD	A Ceph OSD Daemon (OSD) stores data, handles data replication, recovery, backfilling, and rebalancing, and provides monitoring information to Ceph Monitors by checking other Ceph daemons for a activity.
Monitor	Ceph Monitor maintains maps of the cluster state including the monitor map, the OSD map, the placement group (PG) map, and the CRUSH map. It also maintains a history (called an "epoch") of each state change in the Ceph Monitors, Ceph OSD Daemons, and PGs.

RADOS Gateway

The RADOS Gateway service is an object storage interface that enables user to perform HTTP-based CRUD operations on an object. It supports both the OpenStack Swift and Amazon S3 REST APIs. It has the following two types of users, unlike the rest of OpenStack services, which rely completely on Keystone for user management (see more details at *Use RADOS Gateway to access objects using S3/Swift API*).

1. Keystone
2. RADOS Gateway user (managed by Ceph itself and does not require Keystone)

RADOS Gateway is an optional component. Deploy it only if you need to access objects storage functionality using Swift or S3 API. Features include the following:

1. RADOS Gateway is configured to run in a simple (non-federated or single region) mode.

2. The HAProxy on the HPE Helion OpenStack® controller node acts as a load balancer for RADOS Gateway servers (in least connection mode, the load balancer selects the server with the least number of connections) for RADOS Gateway servers.

3. Provides OpenStack Keystone integration (users having a configured set of roles can access Swift APIs served by `radosgw`).

4. The default HPE Helion OpenStack® configuration installs RADOS Gateway on standalone nodes.

5. Access to Amazon S3 APIs is limited to RADOS Gateway users.

6. The RADOS Gateway external and internal endpoints are SSL/TLS- enabled, including the public end points represented by HAProxy.

Deployment Architecture

Consider the following issues when deploying Ceph:

• Ceph networking
• Placement of service components (such as OSD, monitor, RADOS Gateway) across nodes. For example, RADOS Gateway and monitor can be deployed on standalone nodes or together.

Ceph Networking

Ceph clients transmits traffic directly to OSD daemons for storage operations instead of the client routing requests to a specific gateways. OSD daemons perform data replication and participate in recovery activities. In general, a storage pool is configured with a replica count of 3, causing daemons to transact three sets of client data over the cluster network. As a result, 4 MB of write traffic results in a total of 12 MB of data movement in the Ceph clusters (4 MB * 3 replicas = 12MB). Also, Ceph clusters routinely share data among themselves. It is important to segregate Ceph data traffic into three segments, as follows:

• Management traffic, which includes monitoring and logging.
• Client traffic (often termed as data traffic) includes client requests sent to OSD daemons.
• Cluster traffic (often termed as replication traffic), which includes replication and recovery data traffic among OSD daemons.

For a high-performance clusters, a proper network configuration is very important. Use multiple networks for different data traffics. For a cluster of a reasonable size (a few terabytes), we recommend having a cluster with at least two networks, such as a single network for management and client data traffic (front-side) and a cluster (back-side) network. For large Ceph clusters, we recommend segregating all three traffic configurations. Segregation enables helps make for a secure connection because a cluster network is not required to be connected to the Internet directly. It allows OSD daemons to keep communicating without intervention so that placement groups can be brought to active and clean states relatively easily, whenever required. Apart from network sepration(using VLANs), be sure to consider NICs used for Ceph servers too. We strongly recommend using bonded NICs to prevent single points of failure. Note that the network (and hence VLAN) separation is different from NIC separation even though they are linked to each other. In a multi-network model, emphasize VLAN separation, which ideally should be complemented by NIC separation, using the following priority order:

1. Separation of VLAN
2. Separation of NIC for VLANs.

The number of NICs you have will depend on your number of bonded interfaces, as follows:

1. Three bonded interfaces with six NICs: first for management, second for OSD client, and third for OSD internal (preferred setup).
2. Two bonded interfaces with four NICs: management VLANs hooked to the first bonded interface, and OSD networks hooked to the second bonded interface (most-used setup).
3. Only one bonded interface: two NICs for all VLANs (for a few NICs only).

HPE Helion OpenStack® Ceph software offers significant flexibility when defining and deploying OpenStack-based clouds, letting you implement a wide variety of different Ceph configurations. This allows you to design, model, and deploy cloud based on your requirements. You have the following VLAN choices based on traffic types:

1. Single: A single VLAN is used for all traffic, meant primarily for a small cluster.
2. Double deployment: One VLAN is used for cloud management and client traffic and another VLAN is used for Ceph internal traffic.
3. Triple deployment: Separate VLANs are used for management, client, and internal traffic.

As mentioned previously, you can link all VLANs to the same bonded interface, to a separate bonded interface, or a combination of interfaces. For a large cluster, you can use a separate bonded interface which in turn necessiates having at least six NICs for OSD nodes and four NICs for monitor nodes.

Placement of service component

Although you can place service components in multiple ways, this section focuses on recommended deployments only, covering Core Ceph (OSD and monitors) and the RADOS Gateway. For details on alternative supported deployment architecture, please refer to *Alternative supported choices.*

• Core Ceph (i.e. OSD and monitors)
• Rados Gateway

• **Core Ceph**

We recommend the following deployment composition is recommended to avoid single points of failure for the Ceph cluster deployment.

1. Three monitor nodes, to retain the odd number criteria of the monitor quorum.
2. At least three OSD nodes, to ensure that the object is replicated on three separate physical nodes, if the replica count of the pool is set to three (the recommended storage pool configuration is to set the replica count to three).

As mentioned in *Ceph networking*, we recommend using separate VLANS to separate Ceph traffic. The cloud management network is used for logging and/or monitoring, the OSD client network is used for Ceph client traffic, and the OSD internal network is used for internal Ceph traffic, such as replication.

- **RADOS Gateway**

We recommend deploying at least two instances of RADOS Gateway on a standalone node front- ended by HAProxy.

The following diagram illustrates the physical architecture of the RADOS Gateway and using the `entry-scale-kvm-ceph` configuration.
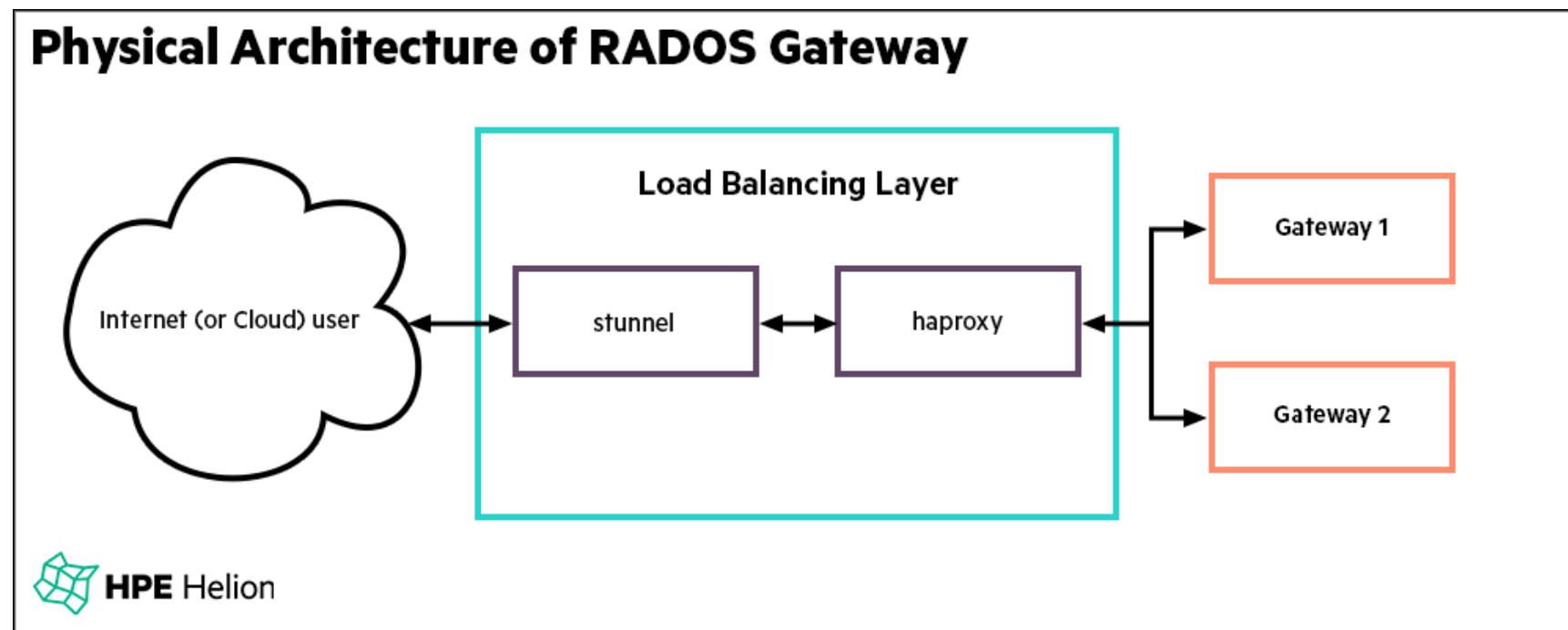
The default example model (`entry-scale-kvm-ceph`) is the recommended mechanism to deploy the Ceph cluster with the RADOS Gateway.

Ceph Deployment Architecture

The following diagram illustrates the Ceph deployment architecture.

Ceph Deployment Architecture

EXTERNAL-VM

EXTERNAL-API

OSD Client

OSD Internal

GUEST

Provider VLANS

OCTAVIA-MGMT

MANAGEMENT

EXTERNAL ROUTING

INTERNAL ROUTING

bond0   bond0   bond0   bond0

Cloud Controller

c-api	n-api
c-scheduler	n-scheduler
c-volume	stunnel
c-backup	monasca-agent
ceph-common	monitor
deployer	

CEPH-OSD

| monasca-agent | ceph-common |
| ceph-osd | |

RADOS Gateway

| monasca-agent | ceph-common |
| apache | rgw |

Compute

| n-compute | ceph-common |
| librbd | python-rbd |

Cloud Controller

c-api	n-api
c-scheduler	n-scheduler
c-volume	stunnel
c-backup	monasca-agent
ceph-common	monitor
deployer	

CEPH-OSD

| monasca-agent | ceph-common |
| ceph-osd | |

RADOS Gateway

| monasca-agent | ceph-common |
| apache | rgw |

Cloud Controller

| c-api | n-api |

CEPH-OSD

| monasca-agent | ceph-common |

The preceding diagram illustrates the Ceph deployment scenario of `entry-scale-kvm-ceph input model`. Monitors are deployed on three controller nodes. Three standalone nodes are used for OSD and two standalone nodes are used for the RADOS Gateway, front-ended by HAProxy running on controller nodes. Management, client, and internal data traffic is separated using independent VLANs.

Alternative supported architecture

HPE Helion OpenStack® 5.0 also supports an alternate deployment architecture, for which the following points are relevant.

- Core networking
- Placement of service components
  - Core Ceph
    - Monitor on standalone node
  - RadosGateway
    - Co-hosted on cluster nodes hosting monitor service components
    - Co-hosted on controller nodes

Core networking

Ceph clients transmit traffic directly to OSD daemons for storage operations instead of the client routing requests to a specific gateway. For more information, refer to *Ceph networking*.

Placement of service components

- **Core Ceph**

  The architecture choices which is supported in HPE Helion OpenStack®5.0 is as follows.

  - Monitor on standalone node

    You can deploy the Ceph monitor service on a one or more dedicated clusters or resource nodes. Ensure that you modify your environment after installing the lifecycle manager. For more details, refer to *Install a monitor service on a dedicated resource node*.

- **RADOS Gateway**

  RADOS Gateway service can be co-hosted with other HPE Helion OpenStack® services as follows.

  - Alternate RADOS Gateway deployment architecture choice
    - Co-hosted on cluster nodes hosting monitor service components
    - Co-hosted on controller nodes

    The default `entry-scale-kvm-ceph` input model deploys the `radosgw` services on two dedicated cluster nodes. However, you can also install the RADOS Gateway on a cluster node hosting the Ceph monitor service or on controller nodes. Refer to *Installing RADOS Gateway on (dedicated) cluster node(s) that host Ceph Monitor service* or *Installing RADOS Gateway on controller nodes* for more details.

    Note: Because the RADOS Gateway service shares server resources with multiple services, these alternate configurations will result in sub-optimal performance, as compared to the default configuration.

Hardware recommendations

For hardware recommendations, refer to *HPE Helion OpenStack 5.0: Recommended Hardware Minimums for an Entry-scale KVM with Ceph Model*.

HPE Helion OpenStack® 5.0: Ceph Deployment and Configurations
Installation and configuration steps for your Ceph backend.

HPE Helion OpenStack® 5.0 Ceph deployment leverages the cloud lifecycle operations supported by Helion lifecycle management. It provides the simplified lifecycle management of critical cluster operations such as service check, upgrade, and reconfiguring service components. This section assumes that you understand cloud input models and highlights only important aspects of cloud input models pertaining to Ceph. We focus on deployment aspects of the `entry-scale-kvm-ceph` input model, which is the most widely used configuration. You can see *Alternative Supported Choices* for the deployment of Ceph with various supported options., To ensure the proper deployment and verification of Ceph, it is important to read the topics and perform the steps in order. This section provides insight on how to alter the `entry-scale-kvm-ceph` input model to deploy Ceph with various supported options. We recommend that you deploye your supported choice only after evaluating all pros and cons.

1. *Predeployment*

   a. Define an OSD Disk Model for an OSD Disk
   b. Customize Your Service Configuration
2. *Deploying Ceph*
3. *Verifying Ceph Cluster Status*

Predeployment

Before you start deploying the HPE Helion OpenStack® cloud with Ceph, you must understand the following aspects of Ceph clusters,

• **Define an OSD Disk Model for an OSD Disk**

This section focus on expressing the storage requirements of an OSD (object-storage daemon) node. OSD nodes have system, data, and journal disks.

System disks are used for OSD components, logging, and other tasks. The configuration of data and journal disks in important for Ceph deployment. A sample disk model file for the `entry-scale-kvm-ceph` cloud is as follows.

```
---
  product:
    version: 2

  disk-models:
  - name: OSD-DISKS
    # Disk model to be used for Ceph OSD nodes
    # /dev/sda_root is used as a volume group for /, /var/log and /var/crash
    # sda_root is a templated value to align with whatever partition is really used
    # This value is checked in os config and replaced by the partition actually used
    # on sda e.g. sda1 or sda5

    volume-groups:
      - name: hlm-vg
        physical-volumes:
          - /dev/sda_root

        logical-volumes:
        # The policy is not to consume 100% of the space of each volume group.
        # 5% should be left free for snapshots and to allow for some flexibility.
          - name: root
            size: 30%
            fstype: ext4
            mount: /
          - name: log
            size: 45%
            mount: /var/log
            fstype: ext4
            mkfs-opts: -O large_file
          - name: crash
            size: 20%
            mount: /var/crash
            fstype: ext4
            mkfs-opts: -O large_file
        consumer:
            name: os

    # Disks to be used by Ceph
    # Additional disks can be added if available
    device-groups:
      - name: ceph-osd-data-and-journal
        devices:
          - name: /dev/sdc
        consumer:
            name: ceph
            attrs:
              usage: data
              journal_disk: /dev/sdd
      - name: ceph-osd-data-and-shared-journal-set-1
        devices:
          - name: /dev/sde
        consumer:
            name: ceph
            attrs:
```

```
              usage: data
              journal_disk: /dev/sdg
     - name: ceph-osd-data-and-shared-journal-set-2
       devices:
         - name: /dev/sdf
       consumer:
          name: ceph
          attrs:
              usage: data
              journal_disk: /dev/sdg
```

The disk model has the following parameters:

Value	Description
device-groups	The name of the device group. There can be several device groups, which allows different sets of disks to be used for different purposes.
name	An arbitrary name for the device group. The name must be unique.
devices	A list of devices allocated to the device group. A `name` field containing `/dev/sdb`, `/dev/sdc`, `/dev/sde`, and `/dev/sdf` indicates that the device group is used by Ceph.
consumer	The service that uses the device group. A `name` field containing **ceph** indicates that the device group is used by Ceph.
attrs	Attributes associated with the consumer.
usage	Devices for a particular service can have several uses. In the preceding sample, the `usage` field contains **data**, which indicates that the device is used for data storage.
journal_disk [OPTIONAL]	The disk to be used for storing journal data. When running multiple Ceph OSDs on a single node, a journal disk can be shared between OSDs of the node.
	If you do not specify this value, Ceph stores the journal on the OSD's data disk (in a separate partition).

The preceding sample file represents the following:

- The first disk is used for OS and system purposes.
- There are three OSD data disks (sdc, sde, and sdf) and two journal disks (sdd and sdg). This configuration shows that we can share journal disks for multiple OSDs. It is recommended to use an OSD journal disk for four OSD data disks. See *Usage of Journal Disk* for more details.
- The drive type is not mentioned for the journal or data disks. You can consume any drive type but we **recommend** using an SSD (solid-state drive) for the journal disk.

Although the preceding model illustrates mixed use of the journal disk, we strongly advise that you keep journal data separate from OSD data, which means that your disk model **should not** have journal disks shared on the same data disks. For more information, see *Usage of Journal Disk*.

Usage of Journal Disk

HPE Helion OpenStack® 5.0 recommends storing the Ceph OSD journal on an SSD and the OSD object data on a separate hard disk drive. SSD drives are costly, so it saves money to use multiple partitions in a single SSD drive for multiple OSD journals. We recommend not more than four or five OSD journals on each SSD disk as a reasonable balance between cost and optimal performance. If you have too many OSD journals on a single SSD, and the journal disk crashes, you might lose your data on those disks. Also, too many journals in a single SSD can negatively affect performance.

Using an OSD journal as a partition on the data disk itself is supported. However, you might see a significant decline in Ceph performance because each client request to store an object is first written to the journal disk before sending an acknowledgment to the client.

The Ceph OSD journal size defaults to 5120 MB (5 GB) in HPE Helion OpenStack® 5.0. This value can be changed, but it does not apply to any existing journal partitions. It will affect new OSDs created after the journal size is changed (whether the journal is on the same disk or a separate disk than the data disk). To change the journal size, edit the `osd_journal_size` parameter in the `~/helion/my_cloud/config/ceph/settings.yml` file.

To summarize:

1. Use SSD for the journal disk.
2. The ratio of OSD data disks to the journal disk is recommended to 4:1.
3. The default journal partition size is 5 GB, which you can change. Actual journal size depends upon your disk drive `rpm` and expected throughput. The formula is: OSD journal size = {2 * (expected throughput * filestore max sync interval)}
4. The journal size for previously configured OSD disks does not change even if you change the `osd_journal_size` parameter in the `~/helion/my_cloud/config/ceph/settings.yml` file. If you want to resize the journal partition of prevously configured OSD disks, you should flush journal data, remove the OSD from the cluster, and then add it again.

- **Customize Your Service Configuration**

  You must customize the paramters in the following files:

- Customize the parameters in the `~/helion/my_cloud/config/ceph/settings.yml` file.

  HPE Helion OpenStack® makes it easy to configure service parameters. All common parameters are available in the `~/helion/my_cloud/config/ceph/settings.yml` file. You can deploy your cluster without altering any of the parameters but we advise that you review and understand the parameters before deploying your cluster. The following link will assist you in the understanding of CEPH placement-groups: *http://docs.ceph.com/docs/master/rados/operations/placement-groups/* The following table provides details about parameters you can change and descriptions of those parameters.

  Core Service Parameters

Parameter	Description	Default Value	Recommendation
ceph_cluster	The name of the Ceph clusters. The default value is Ceph.	Ceph	Customize to suit your requirements.
ceph_release	The name of the Ceph release.	hammer	Do not change the default value.
osd_pool_default_size	The number of replicas for objects in the pool.	3	Do not lower the default value. The value can be increased to the maximum number of OSD nodes in the environment (increasing it beyond this limit will cause the cluster to never reach an active+clean state).
osd_pool_default_pg_num	The default number of placement groups for a pool. This value changes based on the number of OSDs available.	128	The value can be changed based on the number of OSD servers/nodes in the deployment. Refer to the Ceph PG calculator at *http://ceph.com/pgcalc/* to customize it.
fstype	Storage filesystem type for OSDs.	xfs	Only the xfs file system is certified.
zap_data_disk	Zap partition table and contents of the disk.	True	Not recommended to change the default value.
persist_mountpoint	Place to persist OSD data disk mount point.	fstab	Not recommended to change the default value (as it ensures that the OSD data disks are mounted automatically on a system reboot).
osd_settle_time	The time in seconds to wait for after starting/restarting Ceph OSD services.	10 seconds	Increase this value only if the number of OSD servers is more than three or the servers have a slow network.
osd_journal_size	The size of the journal in megabytes.	5120	You can increase this value to achieve optimal use of the journal disk (if it is shared between multiple OSDs).
data_disk_poll_attempts	The maximum number of attempts before attempting to activate an OSD for a new disk (default value 5).	5	Increase this value only if the OSD data disk drives are under-performing or slower than expected. Because this parameter and `data_disk_poll_interval` (following) have a combined effect, we recommend that you consider both while tweaking either of them.
data_disk_poll_interval	The time interval in seconds to wait between `data_disk_poll_attempts`.	12	You can customize this value to suit your requirements. However, because this parameter and `data_disk_poll_attempts` (preceding) have a combined effect, we recommend that you consider both while tweaking either of them,
osd_max_open_files	Maximum number of file descriptors for OSD.	32768	Do not change the default value.
mon_default_dir	Directory to store monitor data.	`/var/lib/ceph/mon/<ceph_cluster>`	Do not change the default value.
mon_max_open_files	Maximum number of file descriptors for monitor.	16384	Do not change the default value.
root_bucket	Ceph CRUSH map root bucket name.	default	Not recommended to change the default value. Changing this value affects CRUSH map and data placement. If you change the default value ensure to create a new rule set with a new root bucket and map the Ceph storage pools to use a new rule set.

Parameter	Description	Default Value	Recommendation
			It is strongly recommended not to change this value post day-zero deployment. Changing the value after deployment results in a newly added OSD nodes to go to a newer `root_bucket`. It affects placement of the placement groups (data) of the storage pools. If you are upgrading cluster(s) from HPE Helion OpenStack® 3.0 to HPE Helion OpenStack® 5.0, you are strongly advice not to change the default value of `root_bucket`.

RADOS Gateway (RGW) Parameters

Parameter	Description	Default Value	Recommendation
radosgw_user	The name of the Ceph client user for `radosgw`.	gateway	Customize to suit your requirements.
radosgw_admin_email	The email address of the server administrator.	`admin@hpe.com`	Update the email address of the server administrator.
rgw_keystone_service_type	**DEPRECATED** To configure RADOS Gateway before deployment refer to *Configuring the service type before deployment*.		
rgw_keystone_accepted_roles	Only users having either of the roles listed here will be able to access the Swift APIs of `radosgw`.	`admin`, *_member_*	Do not change the default value.

Note: The default service password for the RADOS Gateway service can be modified by following the steps documented at *Changing RADOS Gateway Credentials*.

Configuring the RADOS Gateway service type in the Keystone catalog

Configuring the service type before deployment

1. You can configure the RADOS Gateway service type in Keystone catalog by replacing the `ceph-object-store` with desired value in `~/helion/hos/services/ceph/rgw.yml` file on the life cycle manager node.

```
advertises-to-services:
    - service-name: KEY-API
      entries:
      - service-name: ceph-rgw
        service-type: ceph-object-store
        service-description: "Ceph Object Storage Service"
        url-suffix: "/swift/v1"
```

2. After you have modified the **service-type**, commit the change to the local git repository.

```
cd ~/helion
git checkout site
git add ~/helion/hos/services/ceph/rgw.yml
git commit -m "Updating the RADOS Gateway service type"
```

3. Rerun the configuration processor.

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

4. Rerun the deployment area preparation playbooks.

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

5. Run reconfiguration playbook in deployment area.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts keystone-reconfigure.yml
```

Configuring the RADOS Gateway service type after deployment

To update the RADOS Gateway service type in a deployed or a running cloud, you must delete the `ceph-rgw` service from the Keystone catalog and perform the same steps as mentioned in the preceeding section (*Configuring the service type before deployment*).

1. To delete the `ceph-rgw` service, you must know the service-id. Execute the following command from a controller node.

```
source ~/keystone.osrc
openstack service list |grep ceph-rgw | awk '{print $2}'
openstack service delete <service-id>
```

Ceph client parameters

Value	Description	Default Value	Recommendation
pg_active_delay_time	The delay time for Ceph PGs to come into active state.	10	You can increase this value if the number of OSD servers/nodes in the deployment is more than three. Because this parameter and **pg_active_retries** (following) have a combined effect, we recommend that you consider both while tweaking either of them.
pg_active_retries	The number of retries for Ceph placement groups to come into active state with a duration of `pg_active_delay_time` seconds between entries.	5	You can customize this value to suit your requirements. However, because this parameter and **pg_active_delay_time** (preceding) have a combined effect, we recommend that you consider both while tweaking either of them.

- Customize parameters at `~/helion/hos/ansible/roles/_CEP-CMN/defaults/main.yml`.

The following table provides parameter descriptions. You can edit each parameter in the `main.yml` file.

Value	Description
fsid	A unique identifier, File System ID, for the Ceph cluster that you should generate prior to deploying a cluster (use the `uuidgen` command to generate a new FSID). When set, this value cannot be changed.

Deploying Ceph

To deploy a new HPE Helion OpenStack® Ceph cloud using the default `entry-scale-kvm-ceph` model, follow these steps.

Edit Your Ceph Environment Input Files

Perform the following steps:

1. Log in to the lifecycle manager.
2. Copy the example configuration files into the required setup directory and edit them to contain the details of your environment:

```
cp -r ~/helion/examples/entry-scale-kvm-ceph/* ~/helion/my_cloud/definition/
```

Enter your environment information into the configuration files in the `~/helion/my_cloud/definition` directory.

You can find details of how to do this at *Input Model*.

3. Edit the `~/helion/my_cloud/definition/data/servers.yml` file and enter details. If you are using alternative RADOS Gateway deployments, see *Alternative Supported Choice* before editing the `servers.yml` file.

```
# Ceph OSD Nodes
   - id: osd1
     ip-addr: 192.168.10.9
     role: OSD-ROLE
     server-group: RACK1
     nic-mapping: MY-2PORT-SERVER
     mac-addr: "8b:f6:9e:ca:3b:78"
```

```
        ilo-ip: 192.168.9.9
        ilo-password: password
        ilo-user: admin

     - id: osd2
        ip-addr: 192.168.10.10
        role: OSD-ROLE
        server-group: RACK2
        nic-mapping: MY-2PORT-SERVER
        mac-addr: "8b:f6:9e:ca:3b:79"
        ilo-ip: 192.168.9.10
        ilo-password: password
        ilo-user: admin

     - id: osd3
        ip-addr: 192.168.10.11
        role: OSD-ROLE
        server-group: RACK3
        nic-mapping: MY-2PORT-SERVER
        mac-addr: "8b:f6:9e:ca:3b:7a"
        ilo-ip: 192.168.9.11
        ilo-password: password
        ilo-user: admin

 # Ceph RGW Nodes
     - id: rgw1
        ip-addr: 192.168.10.12
        role: RGW-ROLE
        server-group: RACK1
        nic-mapping: MY-2PORT-SERVER
        mac-addr: "8b:f6:9e:ca:3b:62"
        ilo-ip: 192.168.9.12
        ilo-password: password
        ilo-user: admin

     - id: rgw2
        ip-addr: 192.168.10.13
        role: RGW-ROLE
        server-group: RACK2
        nic-mapping: MY-2PORT-SERVER
        mac-addr: "8b:f6:9e:ca:3b:63"
        ilo-ip: 192.168.9.13
        ilo-password: password
        ilo-user: admin
```

The preceding sample file contains three OSD nodes and two RADOS Gateway nodes.

4. Edit the `~/helion/my_cloud/definition/data/disks_osd.yml` file to align the disk model to fit the server specification in your environment. For details on disk models, refer to *disk model*.

Ceph service configuration parameters can be modified as described in the preceding **Predeployment** section.

5. Commit your configuration to the *local git repo*:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

6. After you set up your configuration files, continue with the installation procedure from *Provision Your Baremetal Nodes* .

   Note: For any troubleshooting information regarding the OSD node failure, see *Ceph Storage Troubleshooting*.

Verifying Ceph Cluster Status

If you have deployed RADOS Gateway with core Ceph, then you need to ensure that all service components including RADOS Gateway are functioning as expected.

Verify Core Ceph

Perform the following steps to check the status of the Ceph cluster:

1. Log in to the monitor node.
2. Execute the following command and make sure that the result is HEALTH_OK or HEALTH_WARN:

```
$ ceph health
```

Optionally, you can also set up the lifecycle manager as a Ceph client node (refer to *Set Up the Lifecycle Manager as a Ceph Client*) and execute the preceding command from the lifecycle manager.

Verify RADOS Gateway

To make sure that a Keystone user can access RADOS Gateway using Swift, perform the following steps:

1. Log in to a controller node.
2. Source the `service.osrc` file:

```
source ~/service.osrc
```

3. Execute the following command to generate a list of the containers associated with the user:

```
swift --os-service-type ceph-object-store list
```

If the containers are listed, this indicates that RADOS Gateway is accessible.

HPE Helion OpenStack® 5.0: Alternative Supported Choices

This section provides insight on how to alter the `entry-scale-kvm-ceph` input model to deploy Ceph with various supported options. We recommend that you deploy your supported choice only after evaluating all pros and cons.

This section provides insight on how to alter the `entry-scale-kvm-ceph` input model to deploy Ceph with various supported options. We recommend that you deploy your supported choice only after evaluating all pros and cons. For technical details, please consult with the technical support team. The choices available can impact the performance and scaling of clusters. Choices are illustrated for reference purposes and you can combine one or more of them as needed. The content is categorized as follows:

1. Core Ceph
   - *Installing the Monitor Service on Standalone Nodes*
   - *Using a Single VLAN for All Ceph Traffic (Management, Client, and Internal OSD)*
   - *Using Two VLANs: For Management and Client Traffic and for Internal OSD` Traffic*
2. RADOS Gateway
   - *Installing RADOS Gateway on Dedicated Cluster Nodes that Host the Ceph Monitor Service*
   - *Installing RADOS Gateway on Controller Nodes*
   - *Installing More than Two RADOS Gateway Servers*

Core Ceph

Installing the Monitor Service on Standalone Nodes

The following section provides the procedure for installing the monitor service on standalone nodes instead of installing on controller nodes, as mentioned in `entry-scale-kvm-ceph`.

Attention: If you want to install the monitor service as a dedicated resource node, you must decide before deploying Ceph. HPE Helion OpenStack 5.0 does not support deployment transition. After Ceph is deployed, you cannot migrate the monitor service from controller nodes to dedicated resource nodes.

Prerequisite

Perform the following steps to install the Ceph monitor on a dedicated node. Note that the Ceph requires at least three monitoring servers to form a cluster in case of a node failure.

1. Log in to the lifecycle manager.

2. Copy the `entry-scale-kvm-ceph` input model to the `~/helion/my_cloud/definition` directory before you begin the editing process:

```
cp -r ~/helion/examples/entry-scale-kvm-ceph/* ~/helion/my_cloud/definition/
```

3. Edit the `control_plane.yml` to create a new cluster, such as with `ceph-mon`, as shown here.

```
clusters:
  - name: cluster1
    cluster-prefix: c1
    server-role: CONTROLLER-ROLE
    member-count: 3
    allocation-policy: strict
    service-components:
      - lifecycle-manager
      - ntp-server
      ...

  - name: ceph-mon
    cluster-prefix: ceph-mon
    server-role: CEP-MON-ROLE
    min-count: 3
    allocation-policy: strict
    service-components:
      - ntp-client
      - ceph-monitor

  - name: rgw
    cluster-prefix: rgw
    server-role: RGW-ROLE
    ...
```

4. Edit the `~/helion/my_cloud/definition/data/servers.yml` file to define the Ceph monitor node (monitor services). The following example shows three nodes for monitor services. We recommend using an odd number of monitor nodes.

```
# Ceph Monitor Nodes
- id: ceph-mon1
  ip-addr: 10.13.111.141
  server-group: RACK1
  role: CEP-MON-ROLE
  nic-mapping: MY-4PORT-SERVER
  mac-addr: "f0:92:1c:05:69:10"
  ilo-ip: 10.12.8.217
  ilo-password: password
  ilo-user: admin

- id: ceph-mon2
  ip-addr: 10.13.111.142
  server-group: RACK2
  role: CEP-MON-ROLE
  nic-mapping: MY-4PORT-SERVER
  mac-addr: "83:92:1c:55:69:b0"
  ilo-ip: 10.12.8.218
  ilo-password: password
  ilo-user: admin

- id: ceph-mon3
  ip-addr: 10.13.111.143
  server-group: RACK3
  role: CEP-MON-ROLE
  nic-mapping: MY-4PORT-SERVER
  mac-addr: "d9:92:1c:25:69:e0"
  ilo-ip: 10.12.8.219
  ilo-password: password
  ilo-user: admin
```

```
# Ceph RGW Nodes
- id: rgw1
  ...
```

5. Edit the `~/helion/my_cloud/definition/data/net_interfaces.yml` file to define a new network interface set for your Ceph monitors, as shown here.

```
## This defines the interface used for management
## traffic such as logging, monitoring, etc.
- name: CEP-MON-INTERFACES
  network-interfaces:
    - name: BOND0
      device:
          name: bond0
      bond-data:
          options:
              mode: active-backup
              miimon: 200
              primary: hed1
          provider: linux
          devices:
            - name: hed1
            - name: hed2
      network-groups:
        - MANAGEMENT

- name: RGW-INTERFACES
  network-interfaces:
  ...
```

6. Edit `~/helion/my_cloud/definition/data/disks_ceph_monitor.yml` to define the disk model for monitor nodes.

```
disk-models:
- name: CEP-MON-DISKS
  # Disk model to be used for Ceph monitor nodes
  # /dev/sda_root is used as a volume group for /, /var/log and /var/crash
  # sda_root is a templated value to align with whatever partition is really used
  # This value is checked in os config and replaced by the partition actually used
  # on sda e.g. sda1 or sda5

  volume-groups:
    - name: hlm-vg
      physical-volumes:
        - /dev/sda_root

      logical-volumes:
      # The policy is not to consume 100% of the space of each volume group.
      # 5% should be left free for snapshots and to allow for some flexibility.
        - name: root
          size: 30%
          fstype: ext4
          mount: /
        - name: log
          size: 45%
          mount: /var/log
          fstype: ext4
          mkfs-opts: -O large_file
        - name: crash
          size: 20%
          mount: /var/crash
          fstype: ext4
          mkfs-opts: -O large_file
      consumer:
```

```
          name: os
```

7. Edit the `~/helion/my_cloud/definition/data/server_roles.yml` file to define a new server role for your Ceph monitors:

```
- name: CEP-MON-ROLE
  interface-model: CEP-MON-INTERFACES
  disk-model: CEP-MON-DISKS
```

8. Commit your configuration:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "<commit message>"
```

9. Run the following playbook to add your nodes into Cobbler:

```
cd ~/helion/hos/ansible/
ansible-playbook -i hosts/localhost cobbler-deploy.yml
```

10. To reimage all the nodes using PXE, run the following playbook:

```
cd ~/helion/hos/ansible/
ansible-playbook -i hosts/localhost bm-reimage.yml
```

11. Run the configuration processor:

```
cd ~/helion/hos/ansible/
ansible-playbook -i hosts/localhost config-processor-run.yml
```

12. Update your deployment directory with this playbook:

```
cd ~/helion/hos/ansible/
ansible-playbook -i hosts/localhost ready-deployment.yml
```

13. Deploy these changes:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts site.yml
```

Using a Single VLAN for All Ceph Traffic (Management, Client, and Internal OSD)

You can use a single VLAN to transmit all Ceph traffic. This configuration is recommended for a small cluster deployment.

Perform the following steps to to configure `Entry-scale-kvm-ceph-single-network`.

1. Log in to the lifecycle manager.

2. Copy the `entry-scale-kvm-ceph` input model to the `~/helion/my_cloud/definition` directory before you begin the editing process:

```
cp -r ~/helion/examples/entry-scale-kvm-ceph/* ~/helion/my_cloud/definition/
```

3. Validate that NIC interfaces are correctly specified in `nic_mapping.yml` for servers that are used in the cloud.

4. Ensure that you have at least two NICs for Ceph nodes to create a bonded interface for it.

5. Validate that your servers are mapped to a correct NIC interface specification in `servers.yml`. The following is an example of a server node used for OSD deployment:

```
# Ceph OSD Nodes
    - id: osd1
      ip-addr: 192.168.10.9
      role: OSD-ROLE
      server-group: RACK1
      nic-mapping: MY-2PORT-SERVER
      mac-addr: "8b:f6:9e:ca:3b:78"
```

```
        ilo-ip: 192.168.9.9
        ilo-password: password
        ilo-user: admin
```

6. Delete the OSD-INTERNAL and OSD-CLIENT network groups from `network_groups.yml`. This is necessary because only the management network is used for Ceph traffic, thus OSD-INTERNAL and OSD-CLIENT network groups are not required.

7. Define `net_interfaces.yml` to use only management network groups, as shown here.

```
  - name: CONTROLLER-INTERFACES
    network-interfaces:
      - name: BOND0
        device:
            name: bond0
        bond-data:
            options:
                mode: active-backup
                miimon: 200
                primary: hed3
            provider: linux
            devices:
              - name: hed3
              - name: hed4
        network-groups:
          - EXTERNAL-API
          - EXTERNAL-VM
          - GUEST
          - MANAGEMENT

  - name: COMPUTE-INTERFACES
    network-interfaces:
      - name: BOND0
        device:
            name: bond0
        bond-data:
            options:
                mode: active-backup
                miimon: 200
                primary: hed3
            provider: linux
            devices:
              - name: hed3
              - name: hed4
        network-groups:
          - EXTERNAL-VM
          - GUEST
          - MANAGEMENT

  - name: OSD-INTERFACES
    network-interfaces:
      - name: BOND0
        device:
            name: bond0
        bond-data:
            options:
                mode: active-backup
                miimon: 200
                primary: hed3
            provider: linux
            devices:
              - name: hed3
              - name: hed4
        network-groups:
          - MANAGEMENT
```

```
        - name: RGW-INTERFACES
          network-interfaces:
            - name: BOND0
              device:
                  name: bond0
              bond-data:
                  options:
                      mode: active-backup
                      miimon: 200
                      primary: hed3
                  provider: linux
                  devices:
                      - name: hed3
                      - name: hed4
              network-groups:
                - MANAGEMENT
```

8. Delete VLAN information for OSD-INTERNAL-NET and OSD-CLIENT-NET from `networks.yml`. Only Management VLANs are used.

9. After you set up your configuration files, perform steps **8 to 13** in *Deploying the Monitor Service on Standalone Nodes*.

Using Two VLANs: For Management and Client Traffic and for Internal OSD Traffic

You can use dual VLANs to transmit Ceph traffic. In this configuration one VLAN transmits management and client traffic and the other VLAN transmits internal OSD traffic. A separate bonded interface for two VLANs is used with four NICs. This configuration provides two aspects:

• Use of two networks, such as VLANs.

• Use of separate bonded interfaces for each VLAN (different from what is provided in `entry-scale-kvm-ceph`).

The use of separate NICs segregates traffic at the interface level and requires your server to have at least four NICs. But using a separate bonded interface for each VLAN is not mandatory, and thus you can use a single bonded interface (or server with only two NICs) for Ceph deployment.

Perform the following steps to to configure `Entry-scale-kvm-ceph-dual-network`.

1. Log in to the lifecycle manager.

2. Copy the `entry-scale-kvm-ceph` input model to the `~/helion/my_cloud/definition` directory before you begin the editing process:

```
cp -r ~/helion/examples/entry-scale-kvm-ceph/* ~/helion/my_cloud/definition/
```

3. Validate that NIC interfaces are correctly specified in `nic_mapping.yml` for servers that are used in the cloud. For Ceph OSD nodes, four port servers are required. You can use **HP-DL360-4PORT** as it is defined in `nic_mapping.yml` of `entry-scale-kvm-ceph` or define a new NIC mapping (as shown here) for new sets of servers having four port servers.

```
  - name: HP-4PORT-SERVER
      physical-ports:
        - logical-name: hed1
          type: simple-port
          bus-address: "0000:07:00.0"

        - logical-name: hed2
          type: simple-port
          bus-address: "0000:08:00.0"

        - logical-name: hed3
          type: simple-port
          bus-address: "0000:09:00.0"

        - logical-name: hed4
          type: simple-port
          bus-address: "0000:0a:00.0"
```

4. Modify OSD nodes to use four port servers, as shown here. Change the NIC mapping attribute from **HP-DL360-4PORT** to use any other name defined in `nic_mapping.yml`.

```
  # Ceph OSD Nodes
    - id: osd1
```

```
            ip-addr: 192.168.10.9
            role: OSD-ROLE
            server-group: RACK1
            nic-mapping: HP-DL360-4PORT
            mac-addr: "8b:f6:9e:ca:3b:78"
            ilo-ip: 192.168.9.9
            ilo-password: password
            ilo-user: admin
```

5. Delete OSD-CLIENT from `network_groups.yml`. Note that no dedicated network group exists for client traffic. Only the management network group is used for client traffic.

6. Edit `net_interfaces.yml` with a bonded NIC as shown here.

```
 - name: CONTROLLER-INTERFACES
     network-interfaces:
      - name: BOND0
        device:
            name: bond0
        bond-data:
            options:
                mode: active-backup
                miimon: 200
                primary: hed1
            provider: linux
            devices:
             - name: hed1
             - name: hed2
        network-groups:
          - EXTERNAL-API
          - EXTERNAL-VM
          - GUEST
          - MANAGEMENT

   - name: COMPUTE-INTERFACES
     network-interfaces:
      - name: BOND0
        device:
            name: bond0
        bond-data:
            options:
                mode: active-backup
                miimon: 200
                primary: hed1
            provider: linux
            devices:
             - name: hed1
             - name: hed2
        network-groups:
          - EXTERNAL-VM
          - GUEST
          - MANAGEMENT

   - name: OSD-INTERFACES
     network-interfaces:
      - name: BOND0
        device:
            name: bond0
        bond-data:
            options:
                mode: active-backup
                miimon: 200
                primary: hed1
            provider: linux
            devices:
```

```
                    - name: hed1
                    - name: hed2
              network-groups:
                - MANAGEMENT
            - name: BOND1
              device:
                  name: bond1
              bond-data:
                  options:
                      mode: active-backup
                      miimon: 200
                      primary: hed3
                  provider: linux
                  devices:
                    - name: hed3
                    - name: hed4
              network-groups:
                - OSD-INTERNAL

    - name: RGW-INTERFACES
      network-interfaces:
        - name: BOND0
          device:
              name: bond0
          bond-data:
              options:
                  mode: active-backup
                  miimon: 200
                  primary: hed1
              provider: linux
              devices:
                - name: hed1
                - name: hed2
          network-groups:
            - MANAGEMENT
```

7. Delete OSD-CLIENT from `server_groups.yml`.

8. Delete VLAN information for OSD-CLIENT-NET from `networks.yml`. Only management VLANs are used for client traffic.

9. After you set up your configuration files, perform steps **8 to 13** in *Installing Monitor on Standalone Node*.

RADOS Gateway

Installing RADOS Gateway on Dedicated Cluster Nodes that Host the Ceph Monitor Service

You can configure RADOS Gateway to install on one or more dedicated cluster nodes hosting the Ceph monitor service as follows:

1. Remove the sections for servers in the `~/helion/my_cloud/definition/data/servers.yml` file that have the `role: RGW-ROLE` attribute.

2. Edit the `~/helion/my_cloud/definition/data/control_plane.yml` file and add the following lines to the `service-components` section for the cluster nodes that have the `server-role: MON-ROLE` attribute.

```
 - ceph-radosgw
 - apache2
```

3. Edit the `~/helion/my_cloud/definition/data/net_interfaces.yml` file to remove the RGW-INTERFACES section. This section defines RADOS Gateway network interfaces, which are not required in this configuration:

```
 - name: RGW-INTERFACES
   network-interfaces:
     - name: BOND0
       device:
           name: bond0
       bond-data:
           options:
               mode: active-backup
               miimon: 200
```

```
        primary: hed3
        provider: linux
        devices:
           - name: hed3
           - name: hed4
      network-groups:
        - MANAGEMENT
        - OSD-CLIENT
```

4. After you set up your configuration files, perform steps **8 to 13** in *Deploying the Monitor on Standalone Nodes*.

Installing RADOS Gateway on Controller Nodes

You can configure RADOS Gateway to install on controller nodes. To do this, perform the following steps:

1. Remove the sections for servers in the `~/helion/my_cloud/definition/data/servers.yml` file that have the `role: RGW-ROLE` attribute.

2. Edit the `~/helion/my_cloud/definition/data/net_interfaces.yml` file to remove the RGW-INTERFACES section. This section defines RADOS Gateway network interfaces, which are not required in this configuration:

```
  - name: RGW-INTERFACES
    network-interfaces:
      - name: BOND0
        device:
           name: bond0
        bond-data:
           options:
              mode: active-backup
              miimon: 200
              primary: hed3
           provider: linux
           devices:
              - name: hed3
              - name: hed4
      network-groups:
        - MANAGEMENT
        - OSD-CLIENT
```

3. Edit the `~/helion/my_cloud/definition/data/control_plane.yml` file and add the following line to `service-components` for the cluster with the `server-role: CONTROLLER-ROLE` attribute.

```
  - ceph-radosgw
```

4. After you set up your configuration files, perform steps **8 to 13** in *Deploying the Monitor Service on Standalone Nodes*.

Installing More than Two RADOS Gateway Servers

To deploy more than two RADOS Gateway servers, you need to add a section to the `~/helion/my_cloud/definition/data/servers.yml` file for each additional RADOS Gateway node.

> **Note:** Installing additional RADOS Gateway servers is possible only if RADOS Gateway is installed on dedicated cluster nodes or on dedicated cluster nodes that host the Ceph monitor service. Additional RADOS Gateway servers cannot be added if RADOS Gateway is installed on a controller node.

HPE Helion OpenStack® 5.0: Usage of Ceph Storage
Installation and configuration steps for your Ceph backend.

Ceph is a versatile storage technology that facilitates the consumption of storage by using multiple protocols. Ceph is closely integrated with HPE Helion OpenStack® services to support various storage use cases such as the use of storage pools for cinder-volume and glance data store. HPE Helion OpenStack® further simplifies administrator tasks by providing automated playbooks for various storage use cases.

- Creating storage pools
- Deploying Ceph client components on client nodes
- Configuring OpenStack services with storage pools

The following section guides you through the following common scenarios that you might come across while consuming Ceph for various storage use cases.

1. *Set Up Your Deployer as a Ceph Client Node*
2. *Using Core Ceph for OpenStack Services*

   a. *Prerequisites*

  b. *Use Ceph Storage Pools as Glance Data Stores*
  c. *Use Ceph Storage Pools as Cinder Volume Backends*
  d. *Use Ceph Storage Pools as Cinder Backup Devices*
3. *Use RADOS Gateway to Access Objects Using S3/Swift API*

Although you can also use Ceph storage pools as ephemeral storage backends for Nova, this practice is not formally supported.

Set Up Your Deployer as a Ceph Client Node

By default, your deployer node is not configured as a Ceph client node. As a result, you cannot start your Ceph operation here. It is usually more helpful to set up your deployer node as an admin client node to perform various Ceph operations. This will help you manage the Ceph cluster without logging in to the monitor node. You can turn your deployer node into a client node by executing the following playbook.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts ceph-setup-deployer-as-client.yml
```

After you complete the preceding playbook, your deployer node will be configured with an admin key ring and thus act as an admin node.

Using Core Ceph OpenStack Services

The use of Cinder for glance, cinder-volume, and cinder-backup requires pool creation and setting up ceph-clients on respective OpenStack service nodes. To simplify the workflow, the playbook `ceph-client-prepare.yml` is provided. This playbook reads pool configurations followng the specification provided in `ceph_user_model.yml` and sets up an OpenStack client node. Perform the following steps:

1. Define the user model.
2. Run `hosts/verb_hosts ceph-client-prepare.yml`.

The default user model provided with HPE Helion OpenStack® is mentioned below. The default configuration contains pool configuration for cinder-volume, glance, and cinder-backup. You can edit the file if you do not intend to use Ceph for all services or want to change pool attributes based on your cluster configuration. For example, if you have large number of disks then you can increase the Placement Group (PG) number for a given pool.

```
---
product:
    version: 2

ceph_user_models:
    - user:
        name: cinder
        type: openstack
        secret_id: 457eb676-33da-42ec-9a8c-9293d545c337
      pools:
        - name: volumes
          attrs:
            creation_policy: eager
            type: replicated
            replica_size: 3
            permission: rwx
            pg: 100
          usage:
            consumer: cinder-volume
        - name: vms
          attrs:
            creation_policy: eager
            type: replicated
            replica_size: 3
            permission: rwx
            pg: 100
          usage:
            consumer: nova
        - name: images
          attrs:
            creation_policy: lazy
            permission: rx
```

```
  - user:
      name: glance
      type: openstack
    pools:
      - name: images
        attrs:
          creation_policy: eager
          type: replicated
          replica_size: 3
          permission: rwx
          pg: 128
        usage:
          consumer: glance-datastore
  - user:
      name: cinder-backup
      type: openstack
    pools:
      - name: backups
        attrs:
          creation_policy: eager
          type: replicated
          replica_size: 3
          permission: rwx
          pg: 128
        usage:
          consumer: cinder-backup
```

The following table provides the descriptions of the preceding parameters.

Paramter	Value	Description	Recommendation
user.name	A user-defined string.	Defines the name of the user who can access a set of pools. A user is created with same name in the Ceph system.	Retain the default names for some of the well-known OpenStack users as described in the default user model. For example, cinder user for volume access who needs to access volumes, vms, and the images pool, as defined in the default model.
user.type	OpenStack | user.	Indicates whether the user is specific for OpenStack services. This parameter does not have semantic implications.	Use **openstack** for pools used by cinder-volume, cinder-backup, glance, and Nova services. For other services, you can choose **user**.
user.secret_id	UUID instance.	A unique UUID.	Generate a value for your cluster before setting up the Cinder client. The libvirt process needs this value to access the cluster while attaching a block device from Cinder.
pools.name	A user-defined string.	A user-defined name.	The name has to be unique in the Ceph namespace.
pools.attrs.creation_policy	eager | lazy	If creation_policy is "eager," the playbooks will create the user. If the creation_policy is set to "lazy," the pool will be created externally (not by Ceph ansible playbooks) out of band.	Retain default values for pools meant to be consumed by OpenStack services.
pools.attrs.type	Replicated.	Defines the type of pool. Ceph supports erasure-coded and replicated pools.	Retain the value as replicated if you do want to use ceph-client-prepare.yml for pool creation. This does not mean that you cannot create an erasure-code pool with your cluster deployed using HPE Helion OpenStack® . It just means that erasure-code pools are not officially supported.
pools.attrs.replica_size	1..n	Replica count.	Use 3 as the replica count because it is used in most common scenarios providing the right level of protection and is the minimum setting that HPE Helion OpenStack supports.

Paramter	Value	Description	Recommendation
pools.attrs.permission	rwx	Read, write, and execute the permission a user will have on given pool.	Retain the values for OpenStack user as mentioned in the default user model. Altering these values might affect your client setup configuration.
pools.attrs.pg	Placement group number.	Number of placement groups.	The default value is 128. You can change the value of this parameter if your disk size is larger.
pools.usage.consumer	Predefined choices.	Defines pool consumers, and indicates which services are expected to consume a given pool with which access. It has semantic meaning for the following predefined choices: 1. nova 2. glance-datastore 3. cinder-volume 4. cinder-back-up	Do not pick the value from glance-datastore, nova, cinder-volume, or cinder-backup for user-defined pools. For user-defined pools, you can skip this parameter.

Prerequisites

To use Ceph as a volume backend, the nodes running these services should have the Ceph client installed on them. Use the `ceph-client-prepare.yml` playbook to deploy the Ceph client on these nodes.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts ceph-client-prepare.yml
```

This playbook also creates Ceph users and Ceph pools on the resource nodes.

> **Note:** The following steps install packages and configure the existing client nodes (such as the Cinder, Glance, and Nova Compute nodes) required to use the Ceph cluster. For any new client nodes added later on that need to be configured to use the Ceph cluster, just execute the preceding playbook with the addition of the `--limit <new-client-node>` switch.

Use Ceph Storage Pools as Glance Data Stores

To enable Ceph as a Glance data store, perform the following steps:

1. Log in to the lifecycle manager.
2. Make the following changes in the the `~/helion/my_cloud/config/glance/glance-api.conf.j2` file:

   a. Copy the following content and paste it into the `glance-api.conf.j2` file under **[glance_store]**:

   ```
   [glance_store]
   default_store = rbd
   stores = rbd
   rbd_store_pool = images
   rbd_store_user = glance
   rbd_store_ceph_conf = /etc/ceph/ceph.conf
   rbd_store_chunk_size = 8
   ```

   b. In the same file, comment out the following references to Swift:

   ```
   stores = {{ glance_stores }}
   default_store = {{ glance_default_store }}
   ```

3. **IMPORTANT:** If you have preexisting images in your Glance repo and want to use Ceph exclusively as a backend, use the Glance CLI or other tool to back up your images before configuring Ceph as your Glance backend:

   a. Snapshot or delete all Nova instances using those images.
   b. Download the images locally that you want to save.
   c. Delete all the images from Glance.

After you finish the Ceph configuration you will need to add those images again.

4. Commit your configuration to the *local git repo*, as follows:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

5. Run the configuration processor:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

6. Update your deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

7. Run the Glance reconfigure playbook to configure Ceph as a Glance backend:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts glance-reconfigure.yml
```

After you execute the preceding command successfully, the glance is configured to use Ceph as a data store. You can use glance operations to upload images, which will be stored in the Ceph cluster.

Use Ceph Storage Pools as Cinder Volume Backends

To enable Ceph as a Cinder volume backend, follow these steps:

1. Log in to the lifecycle manager.

2. Make the following changes to the `~/helion/my_cloud/config/cinder/cinder.conf.j2` file:

    a. Add your Ceph backend to the `enabled_backends` section:

```
# Configure the enabled backends
enabled_backends=ceph1
```

    ⚠   **Important:** If you are using multiple backend types, you can use a comma-delimited list here. For example, if you are going to use both VSA and Ceph backends, specify `enabled_backends=vsa-1,ceph1`.

    b. [optional] If you want your volumes to use a default volume type, enter the name of the volume type in the `[DEFAULT]` section with the following syntax. **Make note of this value because you will need it when you create your volume type later.**

    ⚠   **Important:** If you do not specify a default type, then your volumes will default to a nonredundant RAID configuration. We recommend that you create a volume type that meets your environments needs and specify it here.

```
[DEFAULT]
# Set the default volume type
default_volume_type = <your new volume type>
```

    c. Uncomment the `ceph` section and supply the values to match your cluster information. If you have more than one cluster, you will need to add another similar section with respective values. The following example adds only one cluster.

```
[ceph1]
rbd_secret_uuid = <secret-uuid>
rbd_user = <ceph-cinder-user>
rbd_pool = <ceph-cinder-volume-pool>
rbd_ceph_conf = <ceph-config-file>
rbd_cluster_name = <cluster_name>
volume_driver = cinder.volume.drivers.rbd.RBDDriver
volume_backend_name = <ceph-backend-name>
```

    This example has the following values:

Value	Description
rbd_secret_uuid	The secret_id value from the `~/helion/my_cloud/config/ceph/user_model.yml` file, as highlighted here: ```- user: name: cinder type: openstack secret_id: <secret ID will be here> pools: - name: volumes``` ⚠️ **Important:** You should generate and use your own secret ID. You can use any UUID generation tool.
rbd_user	The username value from the `~/helion/my_cloud/config/ceph/user_model.yml` file, as highlighted here: ```- user: name: cinder type: openstack secret_id: <secret ID will be here> pools: - name: volumes```
rbd_pool	The pool name value from the `~/helion/my_cloud/config/ceph/user_model.yml` file, as highlighted here: ```- user: name: cinder type: openstack secret_id: 457eb676-33da-42ec-9a8c-9293d545c337 pools: - name: volumes```
rbd_ceph_conf	The Ceph configuration file location, usually `/etc/ceph/ceph.conf`.
rbd_cluster_name	Name of the Ceph cluster.
volume_driver	The Cinder volume driver. Leave this as the default value specified for Ceph.
volume_backend_name	The name given to the Ceph backend.

3. To enable attaching Ceph volumes to Nova provisioned instances, make the following changes to the `~/helion/my_cloud/config/nova/kvm-hypervisor.conf.j2` file:

   a. Uncomment the Ceph backend lines and edit them as follows:

```
[libvirt]
rbd_user = <ceph-user>
rbd_secret_uuid = <secret-uuid>
```

The values are as follows:

Value	Description
rbd_user	The username value from the `~/helion/my_cloud/config/ceph/user_model.yml` file, as highlighted here: ```- user: name: cinder type: openstack secret_id: 457eb676-33da-42ec-9a8c-9293d545c337```

Value	Description

rbd_secret_uuid	The secret_id value from the `~/helion/my_cloud/config/ceph/user_model.yml` file, as highlighted here:
	
	``
	- user:
	        name: cinder
	        type: openstack
	        secret_id: 457eb676-33da-42ec-9a8c-9293d545c337
	``

Note: To attach a volume provisioned out of a newly added Ceph backend to an existing OpenStack instance, the instance must be rebooted after the new backend has been added.

Attention: Do not use `backend_host` variable in `cinder.conf` file. If `backend_host` is set, it will override the [DEFAULT]/host value which HPE Helion OpenStack 5.0 is dependent on.

4. Commit your configuration to the *local git repo*, as follows:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

5. Run the configuration processor:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

6. Update your deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

7. Run the Cinder reconfigure playbook:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts cinder-reconfigure.yml
```

8. If Nova has been configured to attach Ceph backend volumes, run the Nova reconfigure playbook:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts nova-reconfigure.yml
```

After you execute the preceding command successfully, the cinder-volume is configured to use Ceph as a data store. You can use volume lifecycle operations as described at *HPE Helion OpenStack 5.0: Verifying the Installation*

Use Ceph Storage Pools as Cinder Backup Devices

To enable Cinder backup devices for Ceph, make the following changes to the `~/helion/my_cloud/config/cinder/cinder.conf.j2` file:

1. Uncomment the `ceph backup` section and supply the values:

```
[DEFAULT]
backup_driver = cinder.backup.drivers.ceph
backup_ceph_conf = <ceph-config-file>
backup_ceph_user = <ceph-backup-user>
backup_ceph_pool = <ceph-backup-pool>
```

The values are as follows:

Value	Description
backup_driver	The Cinder volume driver. Leave this as the default value specified for Ceph.
backup_ceph_conf	The Ceph configuration file location, usually: `/etc/ceph/ceph.conf`.
backup_ceph_user	The username value from the `~/helion/my_cloud/config/ceph/user_model.yml` file, as highlighted here: ``` - user: name: cinder-backup type: openstack pools: - name: backups ```
backup_ceph_pool	The pool name value from the `~/helion/my_cloud/config/ceph/user_model.yml` file, as highlighted here: ``` pools: - name: backups ```

2. Commit your configuration to the *local git repo*, as follows:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

3. Run the configuration processor:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

4. Update your deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

5. Run the Cinder reconfigure playbook:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts cinder-reconfigure.yml
```

After you execute the preceding command successfully, the cinder-backup service is configured to use Ceph as a data store. You can use volume lifecycle operations as described in *Associate the Volume Type to the Backend*.

Use RADOS Gateway to Access Objects Using S3/Swift API

RADOS Gateway provides object storage functionality through S3 and Swift API. It has a built-in authentication mechanism and it can use Keystone as an authentication backend, unlike the OpenStack Services, where Keystone is the only authentication backend. Users primarily see the following combinations of object accessibility:

1. Keystone users using:

   a. Swift API
   b. S3 API

2. RADOS Gateway users using:

   a. Swift API
   b. S3 API

Here's a pictorial view of preceding perspective:
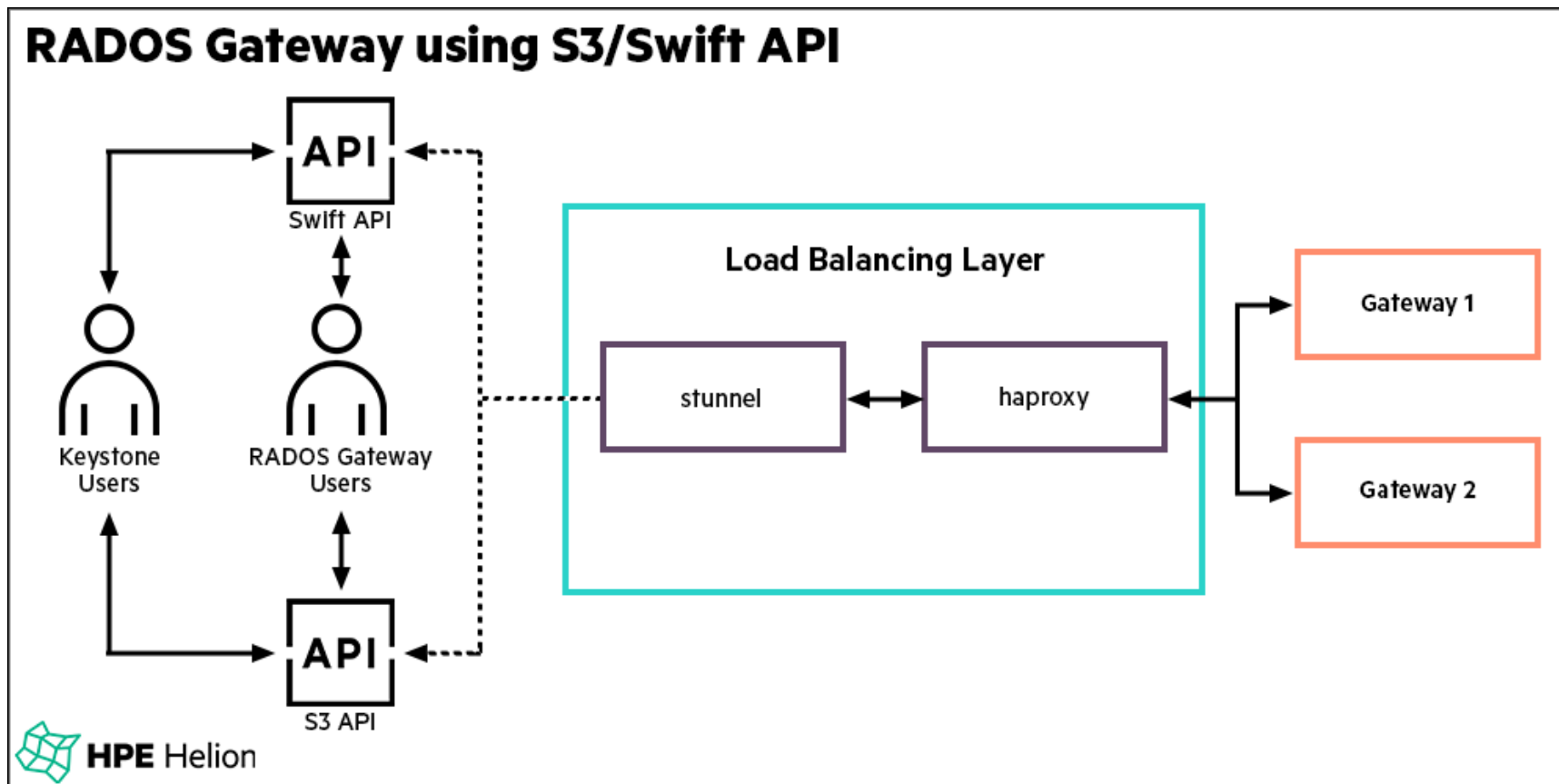
The first view uses Keystone as the authentication backend while the second one uses RADOS Gateway (internal authentication backend). Default deployment of RADOS Gateway in HPE Helion OpenStack®5.0 enables 1.a, 2.a, and 2.b only.

Enabling choice 1.b for Keystone users accessing S3 API requires Keystone to be the default authentication backend. In this case all user authentication goes first to Keystone, with failover to RADOS Gateway even for non-Keystone users (such as in case 2.b), but this situation causes a serious performance drop for RADOS Gateway users.

Lab tests performed on a 10 TB Ceph cluster have shown that enabling S3 API access for Keystone users degrades the performance of S3 API access for RADOS Gateway users roughly by 50%. We ran the rest-bench tool for 10 seconds with a concurrency of 16 and an object size of 4096 (4K) bytes on Ceph deployed using the `entry-scale-kvm-ceph` input model. Our lab finding was that the system performed an average of 1350 (object size 4K) write operations when Keystone was enabled as the default authentication backend, in contrast to 2500 (object size 4K) operations in the default configuration, which was a degradation of roughly 49%. Note that these numbers are for illustration purposes to provide an insight on the degree of degradation. However, storing all credentials in Keystone provides the advantage of reducing the maintenance burden. It is not required to create or manage credentials in a database for S3 authentication. You can use standard administrative tools such as Horizon instead. Aspect 1.b does not apply if you do not intend to enable S3 API access for Keystone users.

Steps to Access S3 API for RADOS Gateway Users

Perform these steps:

1. Create a user by executing the following command:

```
sudo radosgw-admin user create --uid="{username}" --display-name="{Display Name}" --email="{email}"
```

Example:

```
sudo radosgw-admin user create --uid=rgwuser --display-name="RadosGatewayUser" --email=rgwuser@test.com
```

2. Perform the object operation. You can use S3 clients. Example: `python-boto`.

Steps to Access Swift API for RADOS Gateway Users

Perform these steps:

1. Create a user by executing the following command:

```
sudo radosgw-admin user create --uid="{username}" --display-name="{Display Name}" --email="{email}"
```

Example:

```
sudo radosgw-admin user create --uid=rgwuser --display-name="RadosGatewayUser" --email=rgwuser@test.com
```

2. Create a subuser by executing the following command:

```
sudo radosgw-admin subuser create --uid={username} --subuser={username}:{subusername} --access=full --key-type=swift --gen-secret
```

Example:

```
sudo radosgw-admin subuser create --uid="rgwuser" --subuser="rgwuser:rgwswiftuser" --access=full --key-type=swift --gen-secret
```

3. Perform the object operation using Swift CLI or any compatible client.

Steps to Access Swift API for Keystone Users

When RADOS Gateway is deployed per the default `entry-scale-kvm-ceph` model, it exists alongside OpenStack Swift (this is because the Keystone service type for RADOS Gateway defaults to `ceph-object-store`). In this (coexisting) mode, OpenStack Horizon communicates only with OpenStack Swift for all the object store operations. However, the OpenStack Swift command line interface can be tuned to communicate with both OpenStack Swift and RADOS Gateway as follows:

*   # to interact with Swift (default):

```
export OS_SERVICE_TYPE=object-store
```

*   # to interact with RADOS Gateway:

```
 export OS_SERVICE_TYPE=ceph-object-store
```

You can use Swift CLI to perform object operations on Ceph.

Example:

Log in to to the monitor node to list the objects in Ceph using the credentials available in the `~/service.osrc` file.

```
source ~/service.osrc
```

Execute the following command to list objects in the Ceph object store:

```
export OS_SERVICE_TYPE=ceph-object-store
swift list
```

Execute the following command to list objects in the Swift object store:

```
export OS_SERVICE_TYPE=object-store
```

```
swift list
```

Steps to Access S3 API for Keystone Users

By default, a Keystone user can access the RADOS Gateway functionality using the Swift API. To configure a Keystone user for S3 API to access the RADOS Gateway, perform the following steps:

1. Login to lifecycle manager.
2. Edit the `~/helion/my_cloud/config/ceph/settings.yml` file and set the `rgw_s3_auth_use_keystone` value to **true**.

```
rgw_s3_auth_use_keystone: true
```

3. Commit your configuration to local repo.

```
cd ~/helion/hos/ansible
git add -A
git commit -m <"commit message">
```

4. Run ready-deployment playbook.

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

5. Run the ceph-reconfigure playbook.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts ceph-reconfigure.yml
```

Create the EC2 credentials

a. Login to a controller node and source the admin user credentials.

```
source ~/service.osrc
```

b. Execute the following commands to generate the EC2 credentials for the OpenStack user.

```
openstack ec2 credentials create --project <project-name> --user <user-name>
```

For example, to generate the EC2 credentials for user demo for the project demo.

```
openstack ec2 credentials create --project demo --user demo
```

⚠ **Important:** Please contact the Professional Services team for details on how to perform the preceding steps.

HPE Helion OpenStack® 5.0: Managing Ceph Clusters After Deployment
After you successfully deploy your Ceph cluster and its is functioning as expected, you can perform various maintenance operations for the Ceph cluster when required

Managing Ceph Clusters After Deployment

After you successfully deploy your Ceph cluster and its is functioning as expected, you can perform various maintenance operations for the Ceph cluster when required. For example, you can change the configuration of Ceph service components, or scale up storage nodes. The following list provides details about various maintenance options that you might encounter for cluster management. Text links take you to the instructions for each step.

1. Modify service configuration.
2. Scale out cluster.

   a. *Add a Ceph monitor node*.
   b. *Add a data disk to an existing storage node*.
   c. See *Add new OSD node*.

3. Scale down cluster.

      a. *Remove a Ceph monitor node*.

      b. *Remove a RADOS Gateway node*.

      c. *Remove a data disk from an OSD node*.

4. Repair scenarios.

      a. *Replace a failed OSD node*.

      b. *Replace a failed montior node*.

      c. *Replace a failed OS disk in an OSD node*.

      d. *Replace a failed data disk in an OSD node*.

      e. *Replace a failed journal disk in an OSD node*.

5. *Recover Ceph nodes after reboot.*.

Note that the preceding maintenance procedure is based on the `entry-scale-kvm-ceph` input model. If you have altered your deployment, such as by deploying the monitor service on standalone nodes, you might have to adjust the procedure accordingly.

HPE Helion OpenStack® 5.0: Configuring for VSA Block Storage Backend

Installation and configuration steps for your VSA backend.This page describes how to configure your VSA backend for the Helion Entry-scale with KVM cloud model. This procedure can be performed during or after installation.

Prerequisites

- Your HPE Helion OpenStack cloud must be fully deployed using the a KVM cloud model with VSA added. The Entry-Scale KVM with VSA cloud model is an example of such a model. For more details on the installation refer to *HPE Helion OpenStack 5.0: Installing Mid-scale and Entry-scale KVM*.
- It's important that all of your systems have the correct date/time because the HPE StoreVirtual VSA license has a start date. If the start date is later than the system time then the installation will fail.

Notes

- The license for the HPE StoreVirtual VSA license is bundled with HPE Helion OpenStack and comes with a free trial which allows a maximum limit of 50 TB per node. Hence the total amount of the configured storage on an individual VSA node should not exceed 50 TB. To extend the 50 TB per node limit, you can add nodes. A VSA cluster can support up to 16 nodes, which means configured storage on a VSA cluster can be as much as 800 TB.

   Notice: The HPE StoreVirtual VSA default license is for five (5) years with all supported functionality detailed above. During installation, the CMC utility may display a message that some features are not licensed. This message displays in error and can be ignored.

   You can deploy VSA with Adaptive Optimization (AO) or without. The deployment process for each of these options is similar, you just need to make a change in the disk input model. For more detailed information, refer to **VSA with or without Adaptive Optimization (AO)** in *HPE Helion OpenStack 5.0: Modifying the Entry-scale KVM with VSA Model for Your Environment*.

- A single VSA node can have a maximum of seven raw disks (excluding the operating system disks) attached to it, which is defined in the disk input model for your VSA nodes. It is expected that no more than seven disks are specified (including Adaptive Optimization disks) per VSA node. For example, if you want to deploy VSA with two disks for Adaptive Optimization then your disk input model should not specify more than five raw disks for data and two raw disks for Adaptive Optimization. Exceeding the disk limit causes VSA deployment failure.
- You can create more than one VSA cluster of same or different type by specifying the configuration in cloud model. For more details, refer to *HPE Helion OpenStack 5.0: Creating Multiple VSA Clusters*.
- Usernames and passwords designated during VSA configuration or repair must be alphabetic only.

Concerning using multiple backends: If you are using multiple backend options, ensure that you specify each of the backends you are using when configuring your `cinder.conf.j2` file using a comma delimited list. An example would be `enabled_backends=vsa-1,3par_iSCSI,ceph1` and is included in the steps below. You will also want to create multiple volume types so you can specify which backend you want to utilize when creating volumes. These instructions are included below as well. In addition to our documentation, you can also read the OpenStack documentation at *Configure multiple storage backends* as well.

VSA driver has updated name: In the OpenStack Mitaka release, the VSA driver used for HPE Helion integration had its name updated from `hplefthand_` to `hpelefthand_`. To prevent issues when upgrading from previous HPE Helion OpenStack releases, we left the names as-is in the release and provided a mapping so that the integration would continue to work. This will produce a warning in the `cinder-volume.log` file advising you of the deprecated name. The warning will look similar to this:

```
Option "hplefthand_username" from group "<your section>" is deprecated. Use option "hpelefthand_username" from group "<your section>"
should be ignored.
```

These are just warnings and can be ignored.

Configure HPE StoreVirtual VSA

The process for configuring HPE StoreVirtual VSA for HPE Helion OpenStack involves two major steps:

- Creating a cluster
- Configuring VSA as the block storage backend

You can perform these tasks in an automated process using the provided Ansible playbooks or manually using the HPE StoreVirtual Centralized Management Console (CMC) GUI.

Note: Using Ansible playbooks to create clusters is strongly recommended. Cluster creation using HPE StoreVirtual Centralized Management Console (CMC) GUI is deprecated in HPE Helion OpenStack 5.0. Clusters created using CMC manually cannot be reconfigured using the Ansible playbooks. If you used CMC to create the clusters, you must continue to use CMC to administer those clusters.

Using Ansible

Perform the following steps to create the cluster using the provided Ansible playbooks.

1. Log in to the lifecycle manager.
2. Run the following playbook, which will both create your management group and your cluster:

```
cd ~/scratch/ansible/next/hos/ansible/
ansible-playbook -i hosts/verb_hosts vsalm-configure-cluster.yml
```

3. When prompted, enter an administrative user name and password you will use to administer the CMC utility.

Administrative user naming requirements:

- 3 to 30 characters
- Must begin with a letter
- Allowed characters: 0-9, a-z, A-Z, hyphen (-), underline (_)
- Disallowed characters: Multibyte character set

Administrative user password requirements:

- 5 to 40 characters
- Must begin with a letter
- Many ASCII and extended ASCII characters, Multibyte character set
- Disallowed characters: dot (.), colon (:), forward slash (/), comma (,), backward slash (\), semi-colon (;), single-quote ('), space ( ), equals (=)

Important: You will need to remember these values to access the cluster using the CMC utility or to re-run this playbook later if further management of your VSA system is needed. Do not change the credentials for one or more clusters using CMC as it will make the cluster automation playbook ignore those clusters for reconfiguration.

4. In the output of the Ansible playbook you will see different steps where you can locate the values for your VSA cluster name, the virtual IP address, and the IP addresses of each of the VSA hosts in the cluster. You will use these values later when configuring your backend.

The following is an example, with the important information bolded.

This step displays the VSA cluster name and the virtual IP address of the cluster:

```
TASK: [vsalm | _create_cluster | Display the status of cluster creation performed in above step] ***
ok: [helion-cp1-vsa0001-mgmt] => {
    "msg": "Created cluster - Name=cluster-vsa, IP=10.13.111.142"
}
```

5. You can view the added node to the cluster in CMC GUI as follows:

    a. Launch the CMC utility. See *Launching the CMC utility GUI* for more details.
    b. Find the new VSA system.
    c. View the added node in the cluster.

Using the CMC Utility

In HPE Helion OpenStack, cluster creation using HPE StoreVirtual Centralized Management Console (CMC) GUI is deprecated. The clusters created by CMC manually cannot be configured using Ansible playbooks.

Perform the following steps to create the cluster using CMC.

Launching the CMC utility GUI

The HPE StoreVirtual Centralized Management Console (CMC) GUI is an interface where you can configure VSA.

The CMC utility requires a GUI to access it. You can use either of the following methods to launch the CMC GUI.

- *RDP/VNC connect*
- *Any X Display Tool*

RDP/VNC connect method

Setup VNC connect on the Controller Node

The following steps will allow you to setup a VNC connect to your controller node so you can view the GUI.

1. Log in to your first controller node.
2. Run the following command to install the package that is required to launch CMC:

```
sudo apt-get install -y xrdp
```

3. Start `vnc4server` using the instructions below. You will be prompted for a password (min 6 characters). Enter a password and proceed. A sample output is shown below:

```
stack@helion-cp1-c1-m1-mgmt:~$ vnc4server

You will require a password to access your desktops.

Password:
Verify:
xauth:  file /home/stack/.Xauthority does not exist

New 'helion-cp1-c1-m1-mgmt:3 (stack)' desktop is helion-cp1-c1-m1-mgmt:3

Creating default startup script /home/stack/.vnc/xstartup
Starting applications specified in /home/stack/.vnc/xstartup
Log file is /home/stack/.vnc/helion-cp1-c1-m1-mgmt:3.log
```

📄 **Note:** If you directly use xrdp to connect to the first controller node without using the VNC server then a remote session is created whenever you login. To avoid this, a dedicated VNC server instance is launched and connected to that instance by xrdp. This helps to maintain the session.

4. Run `netstat -anp | grep vnc` to determine the public port that VNC is using. In the example below, the port is 5903:

```
stack@helion-cp1-c1-m1-mgmt:~$ netstat -anp | grep vnc
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
tcp        0      0 0.0.0.0:6003            0.0.0.0:*              LISTEN      1413/Xvnc4
tcp6       0      0 :::5903                 :::*                  LISTEN      1413/Xvnc4
```

📄 **Note:** If you reboot the controller node then you must repeat the steps **3** and **4**.

5. Connect to your controller node through any remote desktop or VNC client. We will show the xrdp method first and the VNC method is below it:
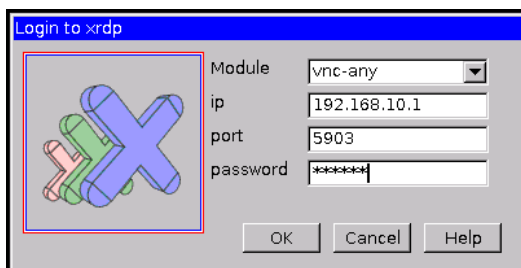
   a. Connecting through remote desktop client

      a. Login to your remote desktop. You will be prompted with xrdp login screen.
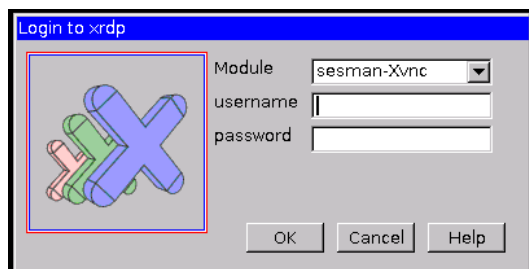
   b. Click the **Module** drop-down list and select `vnc-any`.
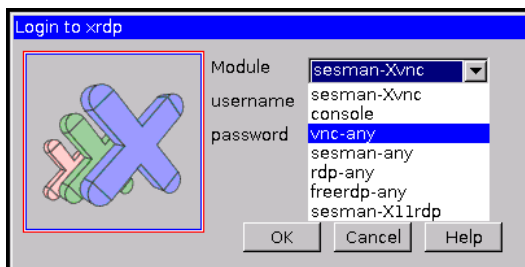
c. Enter the IP address, port and password in the respective fields.

d. Click **Ok**.

b. Connecting through a VNC client, such as *VNC Viewer*:

a. Enter the IP address and port and click **Connect**. You will be prompted for your password once the connection is established.
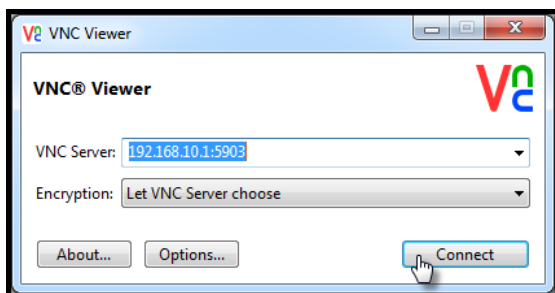
A terminal emulator will be displayed where you can enter the CMC launch command. Note that the CMC launch with this method will have the following limitations: by default, CMC-xterm disables all the title bars and borders. This is an expected behavior.

Install (any) X Display Tool

Note: You can use SSH to an X server but the performance may be poor.

You must configure an X display tool to launch CMC. User can select **any** X display tool. In this section we are using **Xming** tool as an example to launch CMC. The following example provides the steps to install Xming and launch CMC in a Windows environment. Another alternative (not shown in the documentation) is *MobaXterm*.

1. Download and install **Xming** on a Windows machine that can access the lifecycle manager. You can download Xming from *Sourceforge.net*.
2. Select **Enable X11 forwarding** checkbox on the PuTTy session for lifecycle manager. You can do this in PuTTY by:

   a. Navigate to the `Connection -> SSH -> X11` option in PuTTy
   b. Click the `Enable X11 forwarding box to ensure it has a checkmark in it`
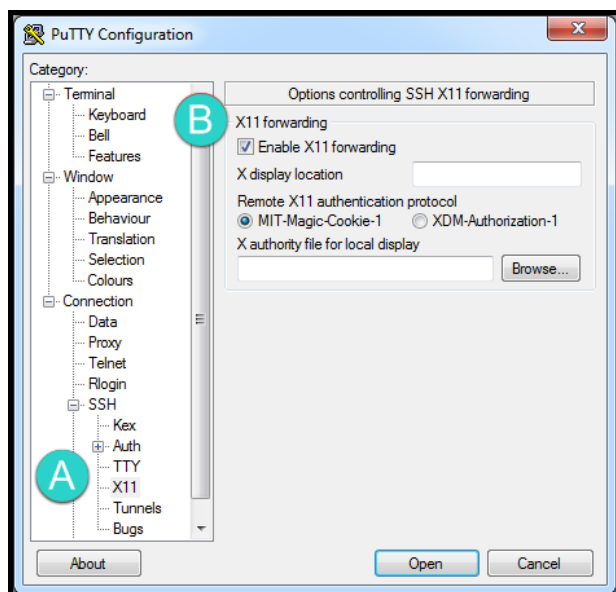
3. SSH to first control plane node.

```
ssh -X
```

and enter the CMC command (as mentioned below) to launch CMC.

1. Run the following command from your first controller node which will open the CMC GUI on your local machine:

```
/opt/HP/StoreVirtual/UI/jre/bin/java -jar /opt/HP/StoreVirtual/UI/UI.jar
```

By default, the CMC GUI is configured to discover the VSA nodes in the subnet in which it is installed. This discovery functionality of VSA nodes using the CMC controller node is not supported in HPE Helion OpenStack. Instead, you must manually add each VSA node, as shown below.

2. In the CMC GUI, click the **Find** menu and then select the **Find Systems** options.
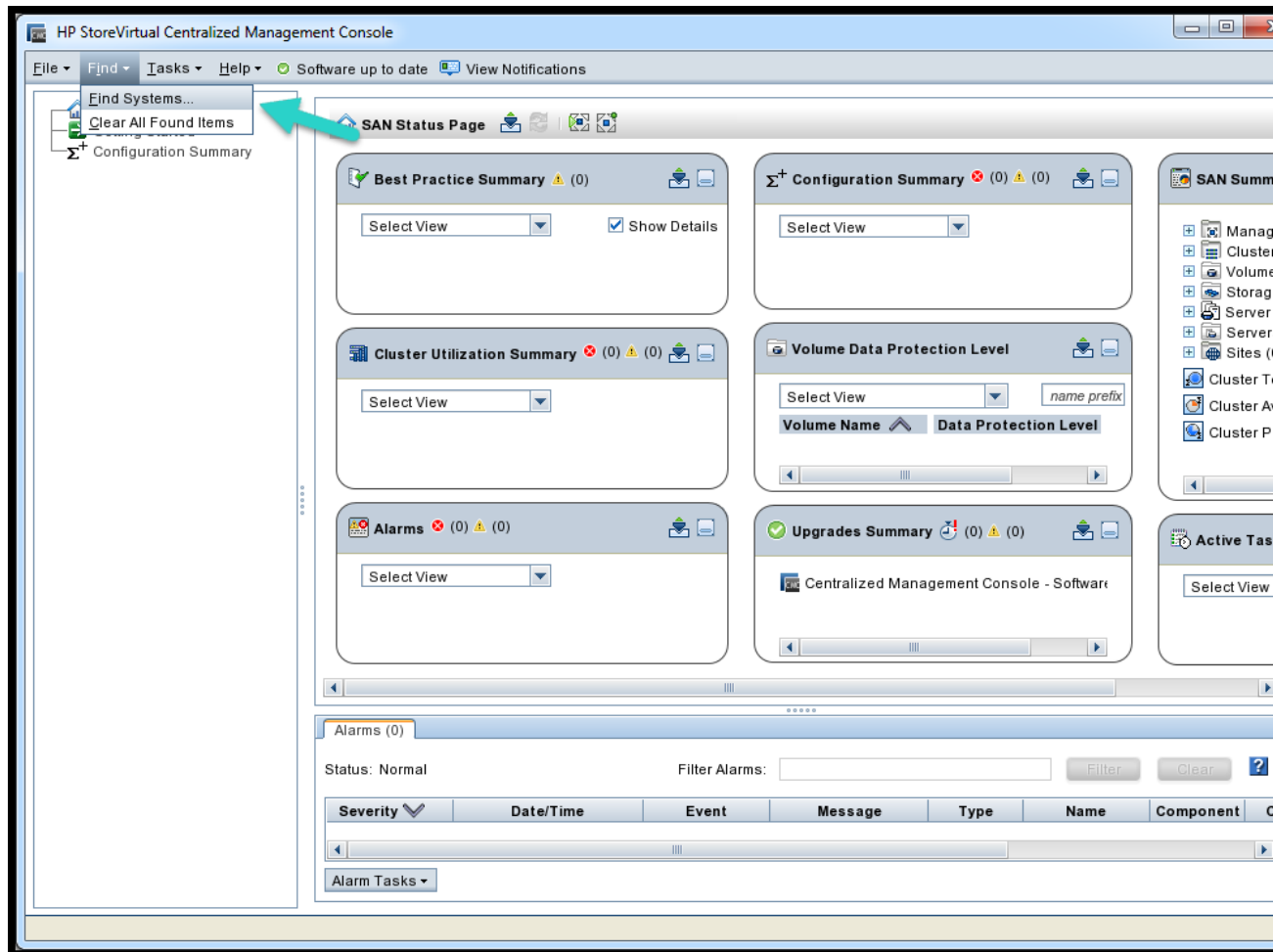
3. Click the **Add** button which will open the **Enter IP Address** dialogue box where you can enter the IP address of your VSA nodes which you noted earlier from your `~/scratch/ansible/next/my_cloud/stage/info/net_info.yml` file.
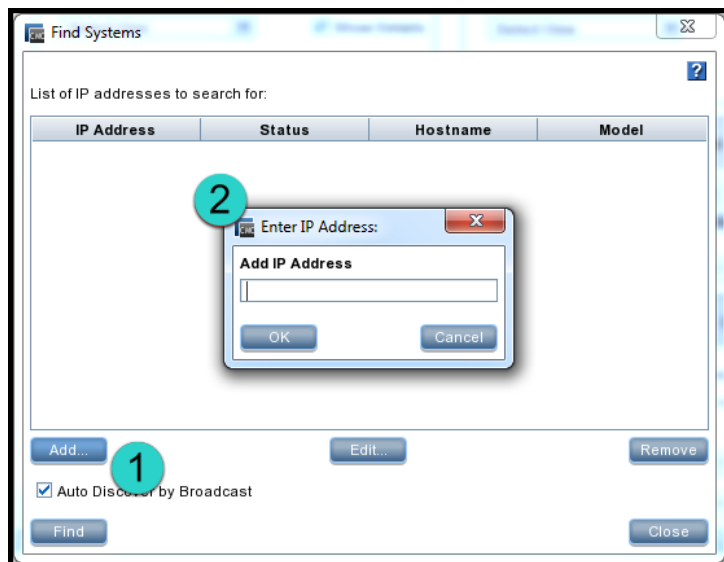
4. Once you have all of your VSA nodes entered, click the **Close** button.
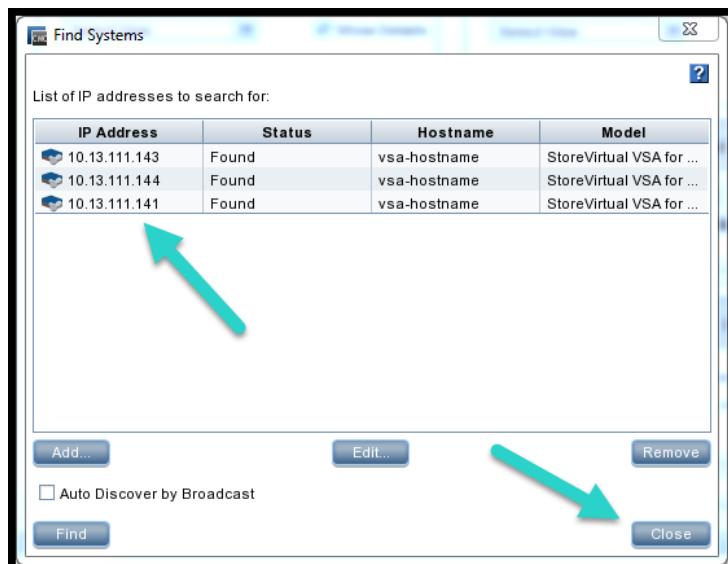
5. Next, click the **Tasks** menu and then navigate to the **Management Group** submenu and select the **New Management Group** option.
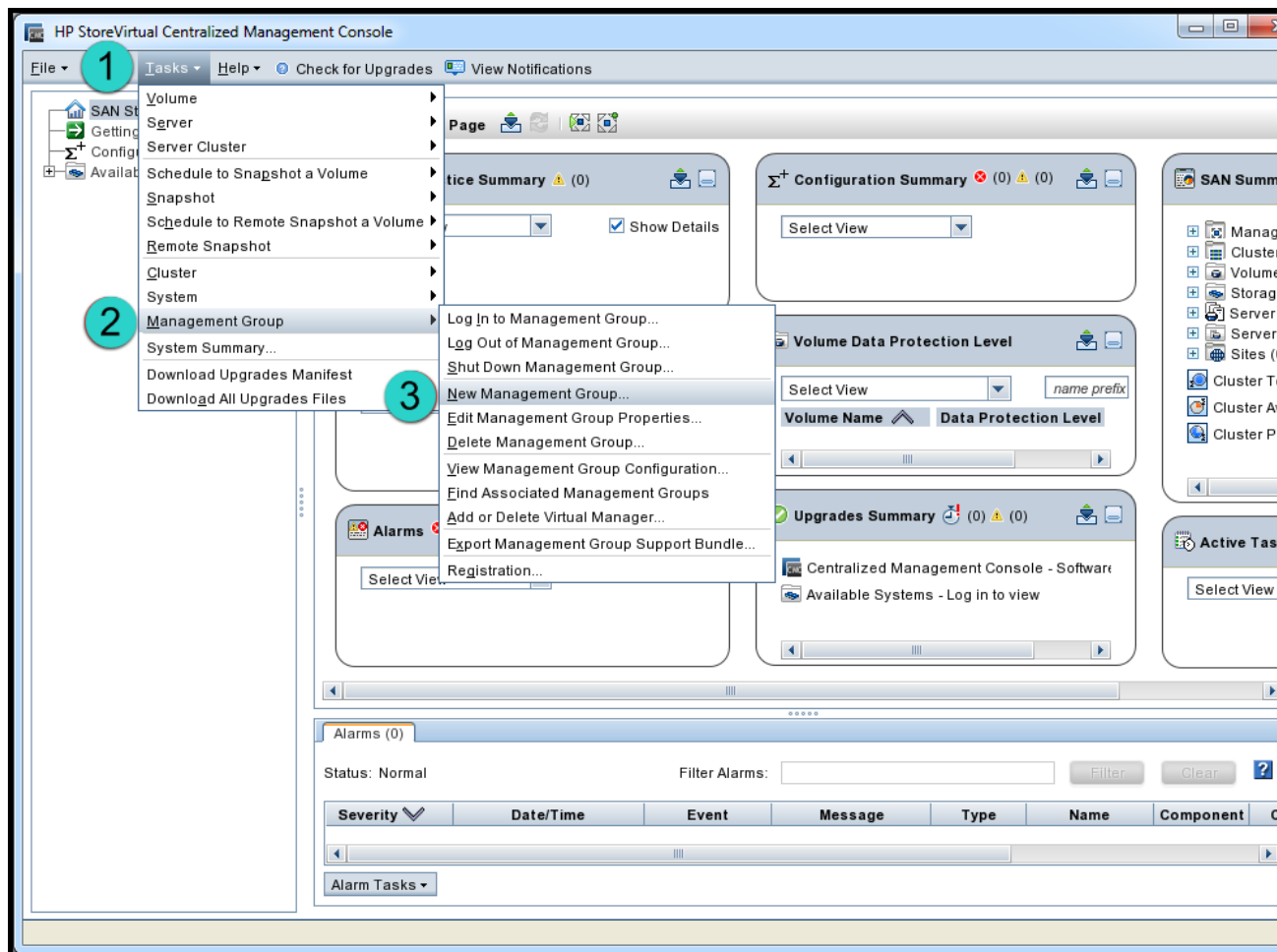
6. In the Management Group wizard, click **Next** and then select **New Management Group** and then **Next** again to continue.
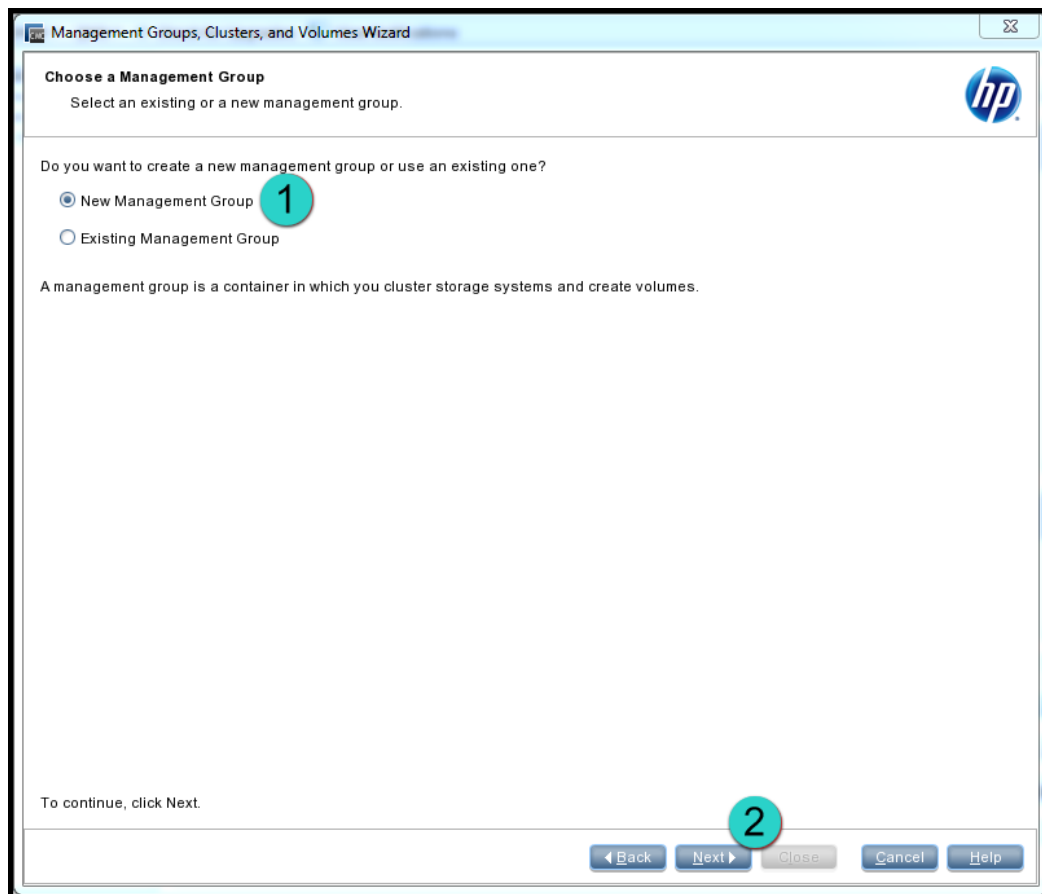
7. Enter a name in the **New Management Group Name** field and then click **Next**.

8. On the **Add Administrative User** you will enter a username and password you will use to administer the CMC utility.

⚠ **Important:** You will need to remember these values as you will input them into your `cinder.conf.j2` file later.

9. Click **Next** to display the **Management Group Time** page.
10. Add your NTP server information and click **Next**
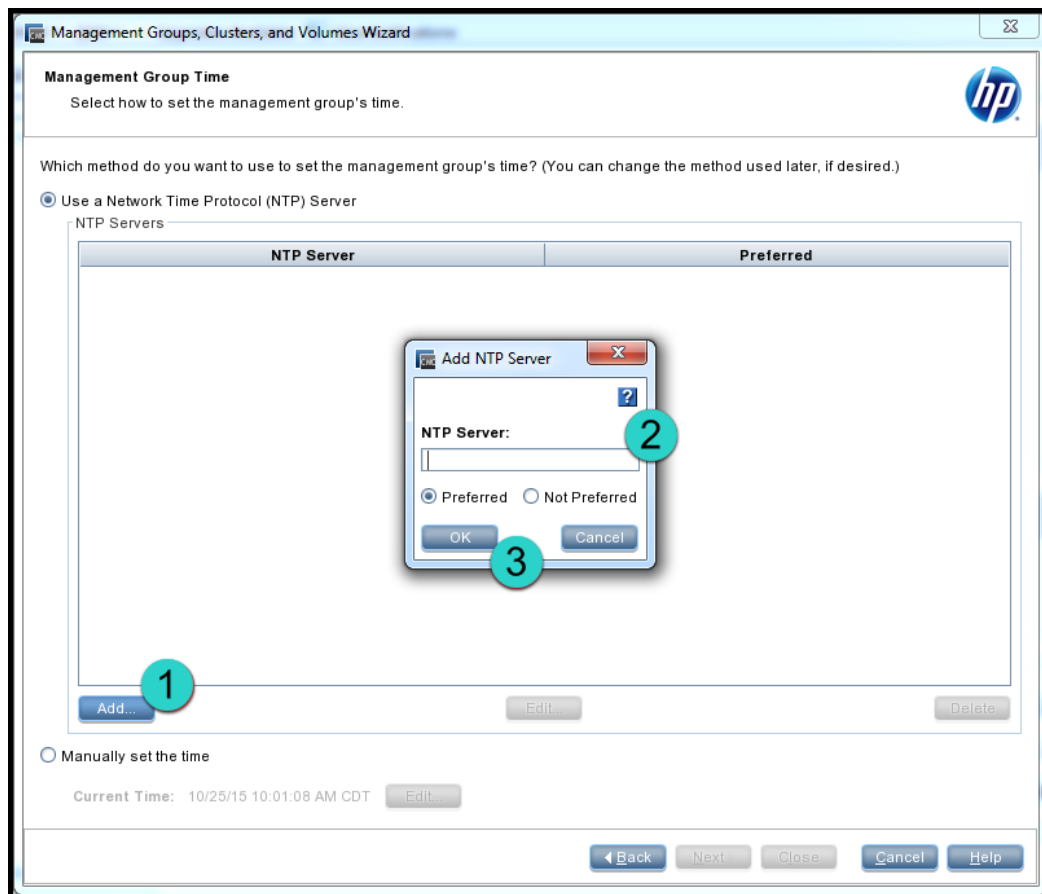
11. Skip the DNS and SMTP sections. To do so, click **Next** and a popup will display where you can choose the **Accept Incomplete** option. Repeat this to skip SMTP section as well.

12. On the **Create a Cluster** options, select **Standard Cluster** from the displayed options and click **Next**.
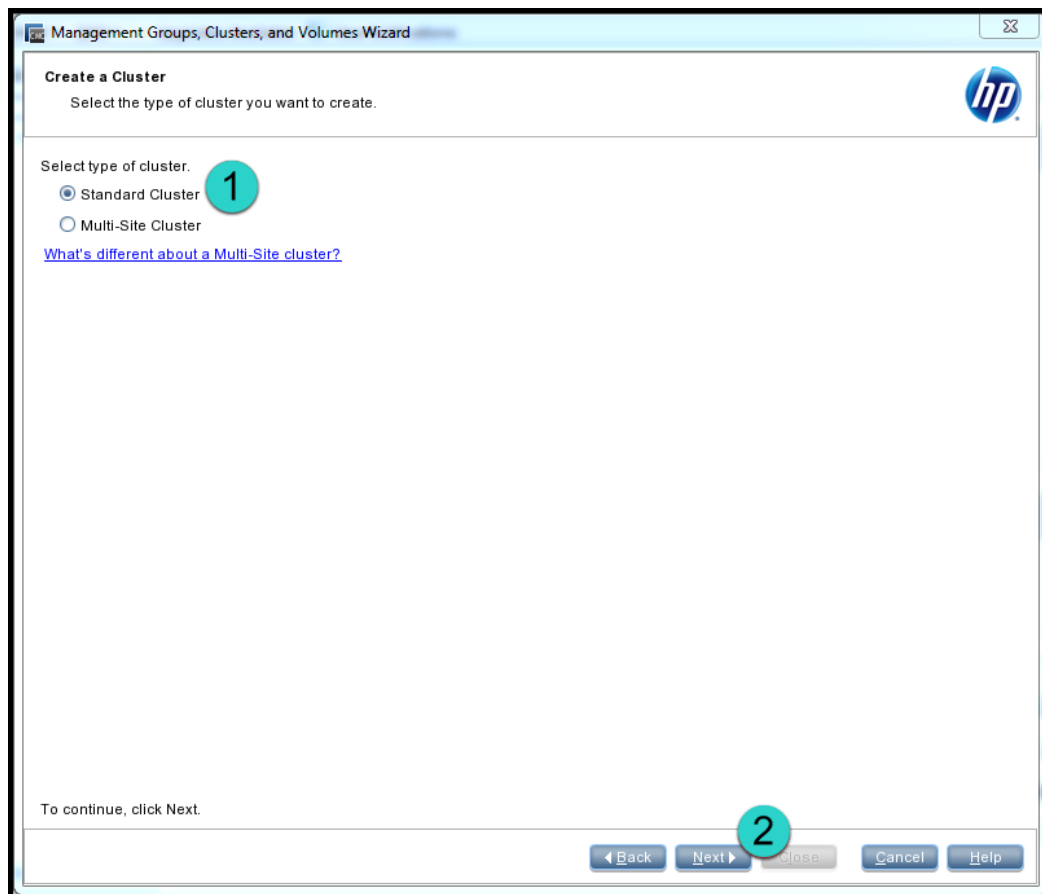
13. In the **Cluster Name** field, enter a name for the cluster and click **Next**.

14. On the **Assign Virtual IPs and Subnet Masks** page, click **Add** and enter the virtual IP address and subnet mask of the cluster in the respective boxes and click **OK**.

Note: The virtual IP address will be found as the `cluster_ip` value in your `~/scratch/ansible/next/my_cloud/stage/info/net_info.yml` file and your subnet mask will be the subnet address from the network your VSA nodes are attached to, usually your `MANAGEMENT` network.
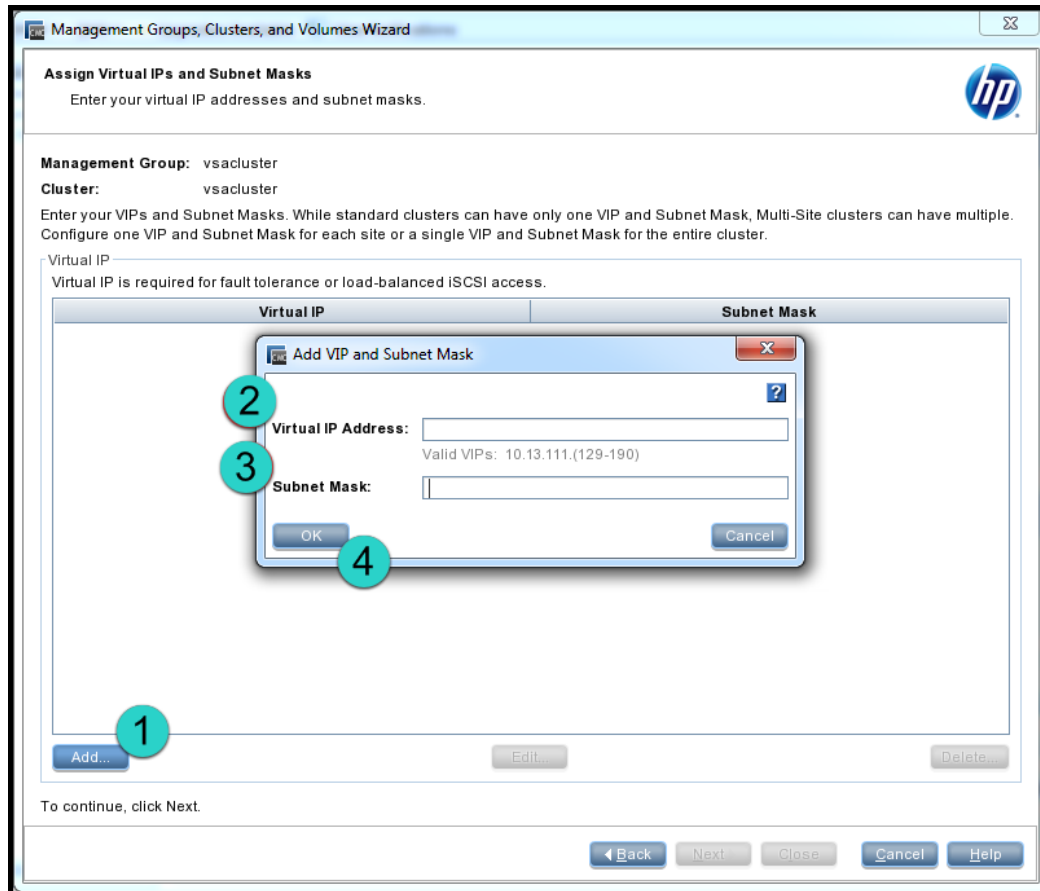
15. The CMC utility will verify the virtual IP address information and then you can click the **Next** button.

16. Select the checkbox for **Skip Volume Creation** and click the **Finish** button which will display your VSA management cluster.

Cluster creation using CMC takes approximately 10 minutes or longer.

Attention: You may get a pop-up notice telling you that your hostnames are not unique. This can be ignored by clicking the OK button.

17. If this process is successful you will see a summary page at the end which outlines what you have completed.

Important: You can create more than one VSA cluster of same or different type by specifying the configuration in cloud model. For more details, refer to Modifying Cloud Model to Create Multiple Clusters

Configure VSA as the Backend

You will use the information you input to the CMC utility to configure your Cinder backend to use your VSA environment.

To update your Cinder configuration to add VSA storage you must modify the `~/helion/my_cloud/config/cinder/cinder.conf.j2` file on your lifecycle manager as follows:

1. Log in to the lifecycle manager.
2. Make the following changes to the `~/helion/my_cloud/config/cinder/cinder.conf.j2` file:

   a. Add your VSA backend to the `enabled_backends` section:

   ```
   # Configure the enabled backends
   enabled_backends=vsa-1
   ```

   ⚠ **Important:** If you are using multiple backend types, you can use a comma delimited list here. For example, if you are going to use both VSA and Ceph backends, you would specify something like this: `enabled_backends=vsa-1,ceph1`.

   b. [OPTIONAL] If you want your volumes to use a default volume type, then enter the name of the volume type in the `[DEFAULT]` section with the syntax below. **You will want to remember this value when you create your volume type in the next section.**

   ⚠ **Important:** If you do not specify a default type then your volumes will default to a non-redundant RAID configuration. It is recommended that you create a volume type and specify it here that meets your environments needs.

   ```
   [DEFAULT]
   # Set the default volume type
   default_volume_type = <your new volume type>
   ```

   c. Uncomment the `StoreVirtual (VSA) cluster` section and fill the values as per your cluster information. If you have more than one cluster, you will need to add another similar section with its respective values. In the following example only one cluster is added.

   ```
   [vsa-1]
   hplefthand_password: <vsa-cluster-password>
   hplefthand_clustername: <vsa-cluster-name>
   hplefthand_api_url: https://<vsa-cluster-vip>:8081/lhos
   hplefthand_username: <vsa-cluster-username>
   hplefthand_iscsi_chap_enabled: false
   volume_backend_name: <vsa-backend-name>
   volume_driver: cinder.volume.drivers.san.hp.hp_lefthand_iscsi.HPLeftHandISCSIDriver
   hplefthand_debug: false
   ```

   where:

Value	Description
hplefthand_password	Password entered during cluster creation in the CMC utility. If you have chosen to encrypt this password, enter the value in this format: ```hplefthand_password: {{ '<encrypted vsa-cluster-password>' | hos_user_password_decrypt }}``` See *Encryption of Passwords and Sensitive Data* for more details. ⚠ **Warning:** The `hos_user_password_decrypt` filter is applied to strings in `cinder.conf.j2`, even if the line is commented out. So if you run a playbook on a `cinder.conf.j2` file that has a commented out encrypted password, and if the key in `HOS_USER_PASSWORD_ENCRYPT_KEY` is not the one used to encrypt the password, then the decryption will fail and the ansible play will fail. Therefore you should not comment out any lines containing the `hos_user_password_decrypt` filter, delete them instead.
hplefthand_clustername	Name of the VSA cluster provided while creating a cluster in the CMC utility.
hplefthand_api_url	Virtual IP address of your VSA cluster, found in your `~/scratch/ansible/next/my_cloud/stage/info/net_info.yml` file.
hplefthand_username	Username given during cluster creation in the CMC utility.
hplefthand_iscsi_chap_enabled	If you set this option as **true** then the hosts will not be able to access the storage without the generated secrets. And if you set this option as **false** then no CHAP authentication is required for the ISCSI connection.

Value	Description
volume_backend_name	Name given to the VSA backend. You will specify this value later in the Associate the Volume Type to a Backend steps.
volume_driver	Cinder volume driver. Leave this as the default value for VSA.
hplefthand_debug	If you set this option as true then the Cinder driver for the VSA will generate logging in debug mode; these logging entries can be found in **cinder-volume.log**.

[OPTIONAL] HPE Helion OpenStack 5.0 supports VSA deployment for KVM hypervisor only but it can be used as pre-deployed (or out of the band deployed) Lefthand storage boxes or VSA appliances (running on ESX/hyper-v/KVM hypervisor). It also supports Cinder configuration of physical Lefthand storage device and VSA appliances. Depending upon your setup, you will have to edit the below section if your storage array is running LeftHand OS lower than version 11:

```
[<unique-section-name>]
volume_driver=cinder.volume.drivers.san.hp.hp_lefthand_iscsi.HPLeftHandISCSIDriver
volume_backend_name=lefthand-cliq
san_ip=<san-ip>
san_login=<san_username>
If adding a password here, then the password can be encrypted using the
mechanism specified in the documentation. If the password has been encrypted
add the value and the hos_user_password_decrypt filter like so:
san_password= {{ '<encrypted san_password>' | hos_user_password_decrypt }}
Note that the encrypted value has to be enclosed in quotes
If you choose not to encrypt the password then the unencrypted password
must be set as follows:
san_password=<san_password>
san_ssh_port=16022
san_clustername=<vsa-cluster-name>
volume_backend_name=<vsa-backend-name>
```

Attention:

Similar to your hplefthand_password in the previous example, encryption for your san_password is supported. If you chose to use encryption you would use the syntax below to express that:

```
san_password= {{ '<encrypted san_password>' | hos_user_password_decrypt }}
```

See *Encryption of Passwords and Sensitive Data* for more details.

Attention: Do not use backend_host variable in cinder.conf file. If backend_host is set, it will override the [DEFAULT]/host value which HPE Helion OpenStack 5.0 is dependent on.

3. Commit your configuration to a *local repository*:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "configured VSA backend"
```

Note: Before you run any playbooks, remember that you need to export the encryption key in the following environment variable: export HOS_USER_PASSWORD_ENCRYPT_KEY=<encryption key> See *HPE Helion OpenStack 5.0: Installing Mid-scale and Entry-scale KVM* for reference.

4. Run the configuration processor:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

5. Run the following command to create a deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

6. Run the Cinder Reconfigure Playbook:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts cinder-reconfigure.yml
```

Post-Installation Tasks

After you have configured VSA as your Block Storage backend, here are some tasks you will want to complete:

- *Create a Volume Type for your Volumes*
- *Verify Your Block Storage Configuration*

HPE Helion OpenStack® 5.0: Configuring for 3PAR Block Storage Backend

Installation and configuration steps for your 3PAR backend.This page describes how to configure your 3PAR backend for the HPE Helion Entry-scale with KVM cloud model. It consists of the following steps:

Expand All Sections
Collapse All Sections

Prerequisites

- You must have the license for the following software before you start your 3PAR backend configuration for the Helion Entry-scale with KVM cloud model:

  - Thin Provisioning
  - Virtual Copy
  - System Reporter
  - Dynamic Optimization
  - Priority Optimization
- Your HPE Helion Entry-scale KVM Cloud should be up and running. Installation steps can be found *here*.
- Your 3PAR Storage Array should be available in the cloud management network or routed to the cloud management network and the 3PAR FC and iSCSI ports configured.
- The 3PAR management IP and iSCSI port IPs must have connectivity from the controller and compute nodes.
- Please refer to the system requirements for 3PAR in the OpenStack configuration guide, which can be found here: *3PAR System Requirements*.

Notes

Encrypted 3Par Volume: Attaching an encrypted 3Par volume is possible after installation by setting `volume_use_multipath = true` under the libvirt stanza in the `nova/kvm-hypervisor.conf.j2` file and reconfigure nova.

Concerning using multiple backends: If you are using multiple backend options, ensure that you specify each of the backends you are using when configuring your `cinder.conf.j2` file using a comma delimited list. An example would be `enabled_backends=vsa-1,3par_iSCSI,ceph1` and is included in the steps below. You will also want to create multiple volume types so you can specify which backend you want to utilize when creating volumes. These instructions are included below as well. In addition to our documentation, you can also read the OpenStack documentation at *Configure multiple storage backends* as well.

Concerning iSCSI and Fiber Channel: You should not configure cinder backends so that multipath volumes are exported over both iSCSI and Fiber Channel from a 3PAR backend to the same Nova compute server.

3PAR driver has updated name: In the OpenStack Mitaka release, the 3PAR driver used for HPE Helion integration had its name updated from `HP3PARFCDriver` and `HP3PARISCSIDriver` to `HPE3PARFCDriver` and `HPE3PARISCSIDriver`. To prevent issues when upgrading from previous HPE Helion OpenStack releases, we left the names as-is in the release and provided a mapping so that the integration would continue to work. This will produce a warning in the `cinder-volume.log` file advising you of the deprecated name. The warning will look similar to this:

```
Option "hp3par_api_url" from group "<your section>" is deprecated. Use option "hpe3par_api_url" from group "<your section>"'
```

These are just warnings and can be ignored.

Multipath Support

If you want to enable multipath for Cinder volumes carved from 3PAR FC/iSCSI storage system, please go through the `~/helion/hos/ansible/roles/multipath/README.md` file on the lifecycle manager. The README.md file contains detailed procedures for configuring multipath for 3PAR FC/iSCSI Cinder volumes.

We have also included an additional set of steps needed if you are using 3PAR FC/iSCSI multipath which is included below.

If you are using 3PAR FC/iSCSI multipath, an additional configuration is required:

> **Note:** If you are planning on attaching an encrypted 3Par volume after istallation, ensure that you `volume_use_multipath = true` under the libvirt section in the `nova/kvm-hypervisor.conf.j2` file before configuring Cinder.

1. Log in to the lifecycle manager.
2. Edit the `~/helion/my_cloud/config/nova/kvm-hypervisor.conf.j2` file add this line under the `[libvirt]` section:

   Example:

   ```
   [libvirt]
   ...
   iscsi_use_multipath=true
   ```

3. Edit the `~/helion/my_cloud/config/cinder/cinder.conf.j2` file add this line under the `[DEFAULT]` section:

   Example:

   ```
   [DEFAULT]
   ...
   use_multipath_for_image_xfer=true
   ```

4. Commit your configuration to the *local git repo*, as follows:

   ```
   cd ~/helion/hos/ansible
   git add -A
   git commit -m "My config or other commit message"
   ```

5. Run the configuration processor:

   ```
   cd ~/helion/hos/ansible
   ansible-playbook -i hosts/localhost config-processor-run.yml
   ```

6. Use the playbook below to create a deployment directory:

   ```
   cd ~/helion/hos/ansible
   ansible-playbook -i hosts/localhost ready-deployment.yml
   ```

7. Run the Nova reconfigure playbook:

   ```
   cd ~/scratch/ansible/next/hos/ansible
   ansible-playbook -i hosts/verb_hosts nova-reconfigure.yml
   ```

Configure 3PAR FC as a Cinder Backend

You must modify the `cinder.conf.j2` to configure the FC details.

Perform the following steps to configure 3PAR FC as Cinder backend:

1. Log in to lifecycle manager.
2. Make the following changes to the `~/helion/my_cloud/config/cinder/cinder.conf.j2` file:

   a. Add your 3PAR backend to the `enabled_backends` section:

   ```
   # Configure the enabled backends
   enabled_backends=3par_FC
   ```

   > **Important:** If you are using multiple backend types, you can use a comma delimited list here. For example, if you are going to use both VSA and 3par backends, you would specify something like this: `enabled_backends=vsa-1,3par_FC`.

   b. [OPTIONAL] If you want your volumes to use a default volume type, then enter the name of the volume type in the `[DEFAULT]` section with the syntax below. **You will want to remember this value when you create your volume type in the next section.**

⚠️ **Important:** If you do not specify a default type then your volumes will default to a non-redundant RAID configuration. It is recommended that you create a volume type and specify it here that meets your environments needs.

```
[DEFAULT]
# Set the default volume type
default_volume_type = <your new volume type>
```

c. Uncomment the `StoreServ (3par) iscsi cluster` section and fill the values per your cluster information. Here is an example:

```
[3par_FC]
san_ip: <3par-san-ipaddr>
san_login: <3par-san-username>
san_password: <3par-san-password>
hp3par_username: <3par-username>
hp3par_password: <hp3par_password>
hp3par_api_url: https://<3par-san-ipaddr>:8080/api/v1
hp3par_cpg: <3par-cpg-name-1>[,<3par-cpg-name-2>, ...]
volume_backend_name: <3par-backend-name>
volume_driver: cinder.volume.drivers.san.hp.hp_3par_fc.HP3PARFCDriver
```

🛑 **Attention:** Do not use `backend_host` variable in `cinder.conf` file. If `backend_host` is set, it will override the [DEFAULT]/host value which HPE Helion OpenStack 5.0 is dependent on.

3. Commit your configuration to the *local git repo*, as follows:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

4. Run the configuration processor:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

5. Update your deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

6. Run the following playbook to complete the configuration:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts cinder-reconfigure.yml
```

Configure 3PAR iSCSI as Cinder backend

You must modify the `cinder.conf.j2` to configure the iSCSI details.

Perform the following steps to configure 3PAR iSCSI as Cinder backend:

1. Log in to lifecycle manager.
2. Make the following changes to the `~/helion/my_cloud/config/cinder/cinder.conf.j2` file:

a. Add your 3PAR backend to the `enabled_backends` section:

```
# Configure the enabled backends
enabled_backends=3par_iSCSI
```

b. Uncomment the `StoreServ (3par) iscsi cluster` section and fill the values per your cluster information. Here is an example:

```
[3par_iSCSI]
san_ip: <3par-san-ipaddr>
san_login: <3par-san-username>
san_password: <3par-san-password>
hp3par_username: <3par-username>
hp3par_password: <hp3par_password>
hp3par_api_url: https://<3par-san-ipaddr>:8080/api/v1
hp3par_cpg: <3par-cpg-name-1>[,<3par-cpg-name-2>, ...]
volume_backend_name: <3par-backend-name>
volume_driver: cinder.volume.drivers.san.hp.hp_3par_iscsi.HP3PARISCSIDriver
hp3par_iscsi_ips: <3par-ip-address-1>[,<3par-ip-address-2>,<3par-ip-address-3>, ...]
hp3par_iscsi_chap_enabled=true
```

Attention: Do not use `backend_host` variable in `cinder.conf` file. If `backend_host` is set, it will override the [DEFAULT]/host value which HPE Helion OpenStack 5.0 is dependent on.

3. Commit your configuration your local git repository:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "<commit message>"
```

4. Run the configuration processor:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

When you run the configuration processor you will be prompted for two passwords. Enter the first password to make the configuration processor encrypt its sensitive data, which is comprised of the random inter-service passwords that it generates and the Ansible group_vars and host_vars that it produces for subsequent deploy runs. You will need this key for subsequent Ansible deploy runs and subsequent configuration processor runs. If you wish to change an encryption password that you have already used when running the configuration processor then enter the new password at the second prompt, otherwise just press carriage return.

For CI purposes you can specify the required passwords on the ansible command line. For example, the command below will disable encryption by the configuration processor

```
ansible-playbook -i hosts/localhost config-processor-run.yml -e encrypt="" -e rekey=""
```

If you receive an error during either of these steps then there is an issue with one or more of your configuration files. We recommend that you verify that all of the information in each of your configuration files is correct for your environment and then commit those changes to git using the instructions above.

5. Run the following command to create a deployment directory.

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

6. Run the following command to complete the configuration:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts cinder-reconfigure.yml
```

Post-Installation Tasks

After you have configured 3PAR as your Block Storage backend, here are some tasks you will want to complete:

- *Create a Volume Type for your Volumes*
- *Verify Your Block Storage Configuration*

HPE Helion OpenStack® 5.0: Ironic OneView Integration

Summary

HPE Helion OpenStack 5.0 supports integration of Ironic (Baremetal) service with HPE OneView using *agent_pxe_oneview* driver. Please refer to *OpenStack Documentation* for more information.

Prerequesites

1. Installed HPE Helion OpenStack 5.0 with entry-scale-ironic-flat-network or entry-scale-ironic-multi-tenancy model.
2. OneView 3.0 instance is running and connected to management network.
3. OneView configuration is set into `definition/data/ironic/ironic_config.yml` (and `ironic-reconfigure.yml` playbook ran if needed). This should enable *agent_pxe_oneview* driver in ironic conductor.
4. Managed node(s) should support PXE booting in legacy BIOS mode.
5. Managed node(s) should have PXE boot NIC listed first. That is, embedded 1Gb NIC must be disabled (otherwise it always goes first).

Integrating with OneView

1. On the lifecycle manager, open the file ~/helion/my_cloud/definition/data/ironic/ironic_config.yml

```
~$ cd ~/helion
~/helion$ vi my_cloud/definition/data/ironic/ironic_config.yml
```

2. Modify the settings listed below:

    a. `enable_oneview`: should be set to "true" for Oneview integration
    b. `oneview_manager_url`: HTTPS endpoint of Oneview management interface, for example: **https://10.0.0.10/**
    c. `oneview_username`: OneView username, for example: **Administrator**
    d. `oneview_encrypted_password`: OneView password in encrypted or clear text form. Encrypted form is distinguished by presence of '@hos@' at the beginning of the string. Encrypted form can be created by running hosencrypt.py program. This program is shipped as part of HPE Helion OpenStack and can be found in ~/helion/hos/ansible directory on lifecycle manager.
    e. `oneview_allow_insecure_connections`: should be set to "true" if OneView is using self-generated certificate.

3. Once you have saved your changes and exited the editor, add files, commit changes to local git repository, and run `config-processor-run.yml` and `ready-deployment.yml` playbooks, as described at *Using Git for Configuration Management*

```
~/helion$ git add my_cloud/definition/data/ironic/ironic_config.yml
~/helion$ cd hos/ansible
~/helion/hos/ansible$ ansible-playbook -i hosts/localhost config-processor-run.yml
...
~/helion/hos/ansible$ ansible-playbook -i hosts/localhost ready-deployment.yml
```

4. Run ironic-reconfigure.yml playbook.

```
$ cd ~/scratch/ansible/next/hos/ansible/

# This is needed if password was encrypted in ironic_config.yml file
~/scratch/ansible/next/hos/ansible$ export HOS_USER_PASSWORD_ENCRYPT_KEY=your_password_encrypt_key

~/scratch/ansible/next/hos/ansible$ ansible-playbook -i hosts/verb_hosts ironic-reconfigure.yml
...
```

Registering Node in OneView

In the OneView web interface:

1. Navigate to Menu -> **Server Hardware**. Add new **Server Hardware** item, using managed node iLO IP and credentials. If this is the first node of this type being added, corresponding **Server Hardware Type** will be created automatically.
2. Navigate to Menu -> **Server Profile Template**. Add **Server Profile Template**. Use **Server Hardware Type** corresponding to node being registered. In **BIOS Settings** section, set **Manage Boot Mode** and **Manage Boot Order** options must be turned on:

Edit server-profile-template-1 | Connections ∨

Connections

Local Storage

Integrated storage controller mode   managed manually ✏

SAN Storage

not supported for this server hardware type

Boot Settings

☑ Manage boot mode

Boot mode | Legacy BIOS ∨ |

☑ Manage boot order

↕ | 1 | CD

↕ | 2 | USB

↕ | 3 | Hard disk

3. Verify that node is powered off. Power the node off if needed.

> ⚠️ **Warning:** Oneview does not support managing boot order for HPE DL servers in UEFI mode. Therefore, HPE DL servers can be only managed in Legacy BIOS mode .

Provisioning Ironic Node

1. Login to the lifecycle manager and source respective credentials file (e.g. service.osrc for admin account).

2. Review glance images with `glance image-list`

```
$ glance image-list
+--------------------------------------+----------------------------+
| ID                                   | Name                       |
+--------------------------------------+----------------------------+
| c61da588-622c-4285-878f-7b86d87772da | cirros-0.3.4-x86_64        |
| 6616ef6a-ee7a-478c-b1bf-4c93088a123f | ir-deploy-iso-HOS5.0       |
| 633d379d-e076-47e6-b56d-582b5b977683 | ir-deploy-kernel-HOS5.0    |
| d5828785-edf2-49fa-8de2-3ddb7f3270d5 | ir-deploy-ramdisk-HOS5.0   |
| d6b5d971-42fd-492b-b33e-6e5f2d88e942 | Ubuntu Trusty 14.04 BIOS   |
+--------------------------------------+----------------------------+
```

Ironic deploy images ("ir-deploy-*") were created automatically by HOS installer. We will be using **agent_pxe_oneview** Ironic driver which needs **ir-deploy-kernel** and **ir-deploy-ramdisk** images. Managed node will be deployed using **Ubuntu Trusty 14.04 BIOS** image.

3. Create node using `agent_pxe_oneview` driver.

```
$ ironic --ironic-api-version 1.22 node-create -d agent_pxe_oneview --name test-node-1 --network-interface neutron -p memory_mb=131072 -p cpu_arch=x86_64 -p local_gb=80 -p
 cpus=2 \
    -p 'capabilities=boot_mode:bios,boot_option:local,server_hardware_type_uri:/rest/server-hardware-types/E5366BF8-7CBF-48DF-A752-8670CF780BB2,server_profile_template_uri:/
rest/server-profile-templates/00614918-77f8-4146-a8b8-9fc276cd6ab2' \
    -i 'server_hardware_uri=/rest/server-hardware/32353537-3835-584D-5135-313930373046' \
    -i dynamic_allocation=True \
    -i deploy_kernel=633d379d-e076-47e6-b56d-582b5b977683 \
    -i deploy_ramdisk=d5828785-edf2-49fa-8de2-3ddb7f3270d5


+-------------------+--------------------------------------------------------------------------+
| Property          | Value                                                                    |
+-------------------+--------------------------------------------------------------------------+
| chassis_uuid      |                                                                          |
| driver            | agent_pxe_oneview                                                        |
| driver_info       | {u'server_hardware_uri': u'/rest/server-                                 |
|                   | hardware/32353537-3835-584D-5135-313930373046', u'dynamic_allocation':   |
|                   | u'True', u'deploy_ramdisk': u'd5828785-edf2-49fa-8de2-3ddb7f3270d5',      |
|                   | u'deploy_kernel': u'633d379d-e076-47e6-b56d-582b5b977683'}                |
| extra             | {}                                                                        |
| name              | test-node-1                                                              |
| network_interface | neutron                                                                  |
| properties        | {u'memory_mb': 131072, u'cpu_arch': u'x86_64', u'local_gb': 80, u'cpus': |
|                   | 2, u'capabilities':                                                      |
|                   | u'boot_mode:bios,boot_option:local,server_hardware_type_uri:/rest        |
|                   | /server-hardware-types/E5366BF8-7CBF-                                    |
|                   | 48DF-A752-8670CF780BB2,server_profile_template_uri:/rest/server-profile- |
|                   | templates/00614918-77f8-4146-a8b8-9fc276cd6ab2'}                         |
| resource_class    | None                                                                     |
| uuid              | c202309c-97e2-4c90-8ae3-d4c95afdaf06                                     |
+-------------------+--------------------------------------------------------------------------+
```

> 📝 **Note:**
> - For deployments created via Ironic/OneView integration, `memory_mb` property must reflect physical amount of RAM installed in managed node. That is, for a server with 128 Gb of RAM it works out to 132*1024=13072.
> - Boot mode in capabilities property must reflect boot mode used by the server, i.e. 'bios' for Legacy BIOS and 'uefi' for UEFI.

- Values for `server_hardware_type_uri`, `server_profile_template_uri` and `server_hardware_uri` can be grabbed from browser URL field while navigating to respective objects in OneView UI. URI corresponds to the part of URL which starts form the token '/rest'. That is, URL https://oneview.mycorp.net/#/profile-templates/show/overview/r/rest/server-profile-templates/12345678-90ab-cdef-0123-012345678901 corresponds to URI /rest/server-profile-templates/12345678-90ab-cdef-0123-012345678901.

- Grab IDs of `deploy_kernel` and `deploy_ramdisk` from **glance image-list** output above.

4. Create port.

```
$ ironic --ironic-api-version 1.22 port-create \
    --address aa:bb:cc:dd:ee:ff \
    --node c202309c-97e2-4c90-8ae3-d4c95afdaf06 \
    -l switch_id=ff:ee:dd:cc:bb:aa \
    -l switch_info=MY_SWITCH \
    -l port_id="Ten-GigabitEthernet 1/0/1" \
    --pxe-enabled true
+-----------------------+-----------------------------------------------------------+
| Property              | Value                                                     |
+-----------------------+-----------------------------------------------------------+
| address               | 8c:dc:d4:b5:7d:1c                                         |
| extra                 | {}                                                        |
| local_link_connection | {u'switch_info': u'C20DATA', u'port_id': u'Ten-GigabitEthernet |
|                       | 1/0/1',    u'switch_id': u'ff:ee:dd:cc:bb:aa'}           |
| node_uuid             | c202309c-97e2-4c90-8ae3-d4c95afdaf06                     |
| pxe_enabled           | True                                                      |
| uuid                  | 75b150ef-8220-4e97-ac62-d15548dc8ebe                     |
+-----------------------+-----------------------------------------------------------+
```

⚠ **Warning:** Ironic Multi-Tenancy networking model is used in this example. Therefore, ironic port-create command contains information about the physical switch. OneView integration can also be performed using the Ironic Flat Networking model. Please refer to *Ironic Examples* for details.

5. Move node to manageable provisioning state. The connectivity between Ironic and OneView will be verified, Server Hardware Template settings validated, and Server Hardware power status retrieved from OneView and set into the Ironic node.

```
$ ironic node-set-provision-state test-node-1 manage
```

6. Verify that node power status is populated.

```
$ ironic node-show test-node-1
+----------------------+-----------------------------------------------------------------+
| Property             | Value                                                           |
+----------------------+-----------------------------------------------------------------+
| chassis_uuid         |                                                                 |
| clean_step           | {}                                                              |
| console_enabled      | False                                                           |
| created_at           | 2017-06-30T21:00:26+00:00                                       |
| driver               | agent_pxe_oneview                                               |
| driver_info          | {u'server_hardware_uri': u'/rest/server-                        |
|                      | hardware/32353537-3835-584D-5135-313930373046', u'dynamic_allocation': |
|                      | u'True', u'deploy_ramdisk': u'd5828785-edf2-49fa-8de2-3ddb7f3270d5', |
|                      | u'deploy_kernel': u'633d379d-e076-47e6-b56d-582b5b977683'}      |
| driver_internal_info | {}                                                              |
| extra                | {}                                                              |
| inspection_finished_at | None                                                          |
| inspection_started_at | None                                                           |
| instance_info        | {}                                                              |
| instance_uuid        | None                                                            |
| last_error           | None                                                            |
| maintenance          | False                                                           |
| maintenance_reason   | None                                                            |
| name                 | test-node-1                                                     |
| network_interface    |                                                                 |
| power_state          | power off                                                       |
| properties           | {u'memory_mb': 131072, u'cpu_arch': u'x86_64', u'local_gb': 80, u'cpus': |
|                      | 2, u'capabilities':                                             |
```

```
|                       | u'boot_mode:bios,boot_option:local,server_hardware_type_uri:/rest       |
|                       | /server-hardware-types/E5366BF8-7CBF-                                  |
|                       | 48DF-A752-8670CF780BB2,server_profile_template_uri:/rest/server-profile- |
|                       | templates/00614918-77f8-4146-a8b8-9fc276cd6ab2'}                        |
| provision_state       | manageable                                                              |
| provision_updated_at  | 2017-06-30T21:04:43+00:00                                              |
| raid_config           |                                                                        |
| reservation           | None                                                                   |
| resource_class        |                                                                        |
| target_power_state    | None                                                                   |
| target_provision_state| None                                                                   |
| target_raid_config    |                                                                        |
| updated_at            | 2017-06-30T21:04:43+00:00                                              |
| uuid                  | c202309c-97e2-4c90-8ae3-d4c95afdaf06                                   |
+-----------------------+------------------------------------------------------------------------+
```

7. Move node to available provisioning state. The Ironic node will be reported to Nova as available.

```
$ ironic node-set-provision-state test-node-1 provide
```

8. Verify that node resources were added to Nova hypervisor stats.

```
$ nova hypervisor-stats
+---------------------+--------+
| Property            | Value  |
+---------------------+--------+
| count               | 1      |
| current_workload    | 0      |
| disk_available_least| 80     |
| free_disk_gb        | 80     |
| free_ram_mb         | 131072 |
| local_gb            | 80     |
| local_gb_used       | 0      |
| memory_mb           | 131072 |
| memory_mb_used      | 0      |
| running_vms         | 0      |
| vcpus               | 2      |
| vcpus_used          | 0      |
+---------------------+--------+
```

9. Create Nova flavor.

```
$ nova flavor-create m1.ironic auto 131072 80 2
+--------------------------------------+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| ID                                   | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+--------------------------------------+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| 33c81884-b8aa-46b7-ab3f-076f3b72f8d8 | m1.ironic | 131072    | 80   | 0         |      | 2     | 1.0         | True      |
+--------------------------------------+-----------+-----------+------+-----------+------+-------+-------------+-----------+
$ nova flavor-key m1.ironic set capabilities:boot_mode="bios"
$ nova flavor-key m1.ironic set capabilities:boot_option="local"
$ nova flavor-key m1.ironic set cpu_arch=x86_64
```

📝 **Note:** All parameters (specifically, amount of RAM and boot mode) must correspond to ironic node parameters.

10. Create Nova keypair if needed.

```
$ nova keypair-add ironic_kp --pub-key ~/.ssh/id_rsa.pub
```

11. Boot Nova instance.

```
$ nova boot --flavor m1.ironic --image d6b5d971-42fd-492b-b33e-6e5f2d88e942 --key-name ironic_kp --nic net-id=5f36f0d9-2df3-4ee3-810c-baffd458dcf3 test-node-1
+--------------------------------------+---------------------------------------------------------------+
```

```
| Property                            | Value                                                      |
+-------------------------------------+------------------------------------------------------------+
| OS-DCF:diskConfig                   | MANUAL                                                     |
| OS-EXT-AZ:availability_zone         |                                                           |
| OS-EXT-SRV-ATTR:host                | -                                                         |
| OS-EXT-SRV-ATTR:hypervisor_hostname | -                                                         |
| OS-EXT-SRV-ATTR:instance_name       |                                                           |
| OS-EXT-STS:power_state              | 0                                                         |
| OS-EXT-STS:task_state               | scheduling                                                |
| OS-EXT-STS:vm_state                 | building                                                  |
| OS-SRV-USG:launched_at              | -                                                         |
| OS-SRV-USG:terminated_at            | -                                                         |
| accessIPv4                          |                                                           |
| accessIPv6                          |                                                           |
| adminPass                           | pE3m7wRACvYy                                              |
| config_drive                        |                                                           |
| created                             | 2017-06-30T21:08:42Z                                      |
| flavor                              | m1.ironic (33c81884-b8aa-46b7-ab3f-076f3b72f8d8)          |
| hostId                              |                                                           |
| id                                  | b47c9f2a-e88e-411a-abcd-6172aea45397                      |
| image                               | Ubuntu Trusty 14.04 BIOS (d6b5d971-42fd-492b-b33e-6e5f2d88e942) |
| key_name                            | ironic_kp                                                 |
| metadata                            | {}                                                        |
| name                                | test-node-1                                               |
| os-extended-volumes:volumes_attached | []                                                       |
| progress                            | 0                                                         |
| security_groups                     | default                                                   |
| status                              | BUILD                                                     |
| tenant_id                           | c8573f7026d24093b40c769ca238fddc                          |
| updated                             | 2017-06-30T21:08:42Z                                      |
| user_id                             | 2eae99221545466d8f175eeb566cc1b4                          |
+-------------------------------------+------------------------------------------------------------+
```

During nova instance boot, the following operations will be performed by Ironic via OneView REST API.

- In OneView, new Server Profile is generated for specified Server Hardware, using specified Server Profile Template. Boot order in Server Profile is set to list PXE as the first boot source.
- Managed node is powered on and boots IPA image from PXE.
- IPA image writes user image onto disk and reports success back to Ironic.
- Ironic modifies Server Profile in OneView to list 'Disk' as default boot option.
- Ironic reboots the node (via OneView RESt API call).

HPE Helion OpenStack® 5.0: Troubleshooting the Installation

We have gathered some of the common issues that occur during installation and organized them by when they occur during the installation.

We have gathered some of the common issues that occur during installation and organized them by when they occur during the installation. These sections will coincide with the steps labeled in the installation instructions.

- *Issues during Lifecycle-manager Setup*
- *Issues while Provisioning your Baremetal Nodes*
- *Issues while Updating Configuration Files*
- *Issues while Deploying the Cloud*
- *Issues during Block Storage Backend Configuration*

Issues during Lifecycle-manager Setup

Issue: Running the hos-init.bash script when configuring your lifecycle manager does not complete

Part of what the `~/hos-3.0.0/hos-init.bash` script does is install git and so if your DNS nameserver(s) is not specified in your `/etc/resolv.conf` file, is not valid, or is not functioning properly on your lifecycle manager then it won't be able to complete.

To resolve this issue, double check your nameserver in your `/etc/resolv.conf` file and then re-run the script.

Issues while Provisioning your Baremetal Nodes

Issue: The bm-power-status.yml playbook returns an error.

If the output of the bm-power-status.yml playbook gives you an error like the one below for one of your servers then the most likely cause is an issue with the information located in the ~/helion/my_cloud/definition/servers.yml file. It could be that the information is incorrect or the iLO username you are using does not have administrator privileges. Verify all of the information for that server and recommit the changes to git and re-run the playbook if the information was incorrect. If it's a rights issue, ensure the iLO user has administrative privileges.

Error:

```
failed: [vsa1] => {"cmd": "ipmitool -I lanplus -E -N 5 -R 12 -U iLOAdmin -H 10.12.11.185 power status", "failed": true, "rc": 1}
stderr: Error: Unable to establish IPMI v2 / RMCP+ session
```

This error below, which may come with running the `bm-power-status.yml` playbook, may be resolved by placing the `ilo-password` values in your `servers.yml` file in double quotes.

Error:

```
failed: [controller2] => {"failed": true}
msg: ipmi: 'int' object has no attribute 'startswith'
```

Cobbler Deployment Fails due to CIDR Error

The error will look similar to this:

```
TASK: [cobbler | set variables | Find baremetal nic] ***************************
failed: [localhost] => {"failed": true}
msg: matchcidr: no nic matching 10.242.115.0/24
lo 127.0.0.1/8
eth0 10.242.115.18/26
```

The most likely cause of this is that your `netmask` entry in the `~/helion/my_cloud/definition/data/servers.yml` file is incorrect.

To resolve this issue ensure that you have the correct `subnet` and `netmask` entries in the `~/helion/my_cloud/definition/data/servers.yml` for your Management network.

You can also verify these values exist in the `/etc/network/interfaces.d/ethX` file on your lifecycle manager. These should match the entries in the `servers.yml` file.

Once you make your corrections to your configuration files, commit the changes with these steps:

1. Ensuring that you stay within the `~/helion` directory, commit the changes you just made:

```
cd ~/helion
git commit -a -m "commit message"
```

2. Redeploy Cobbler:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost cobbler-deploy.yml
```

Issue: Configuration changes needed after Cobbler deploy

If you've made a mistake or wish to change your `~/helion/my_cloud/definition/data/servers.yml` configuration file after you've already run the `cobbler-deploy.yml` playbook, follow these steps to ensure Cobbler gets updated with the new server information:

1. Ensure your `servers.yml` file is updated

2. Ensuring that you stay within the `~/helion` directory, commit the changes you just made:

```
cd ~/helion
git commit -a -m "commit message"
```

3. Determine which nodes you have entered into Cobbler by using:

```
sudo cobbler system list
```

4. Remove the nodes that had the old information from Cobbler using:

```
sudo cobbler system remove --name <nodename>
```

5. Re-run the `cobbler-deploy.yml` playbook to update the new node definitions to Cobbler:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost cobbler-deploy.yml
```

> **Note:** If you've also already run the `bm-reimage.yml` playbook then read the *How to re-image an existing node* section below for how to ensure your nodes get re-imaged when re-running this playbook.

Issue: bm-reimage.yml playbook doesn't find any nodes to image

You may receive this error when running the `bm-reimage.yml` playbook:

```
TASK: [cobbler | get-nodelist | Check we have targets] ************************
failed: [localhost] => {"failed": true}
msg: There is no default set of nodes for this command, use -e nodelist

FATAL: all hosts have already failed -- aborting
```

This behavior occurs when you don't specify the `-e nodelist` switch to your command and all of your nodes are marked as `netboot-enabled: false` in Cobbler. By default, without the `-e nodelist` switch, the `bm-reimage.yml` playbook will only reimage the nodes marked as `netboot-enabled: True`, which you can verify which nodes you have that are marked as such with this command:

```
sudo cobbler system find --netboot-enabled=1
```

If this is on a fresh install and none of your nodes have been imaged with the HPE Helion OpenStack ISO, then you should be able to remove all of your nodes from Cobbler and re-run the `cobbler-deploy.yml` playbook. You can do so with these commands:

1. Get a list of your nodes in Cobbler:

```
sudo cobbler system list
```

2. Remove each of them with this command:

```
sudo cobbler system remove --name SYSTEM_NAME
```

3. Confirm they are all removed in Cobbler:

```
sudo cobbler system list
```

4. Re-run `cobbler-deploy.yml`:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost cobbler-deploy.yml
```

5. Confirm they are all now set to `netboot-enabled: True`:

```
sudo cobbler system find --netboot-enabled=1
```

Issue: The --limit switch does not do anything

If the `cobbler-deploy.yml` playbook fails, it makes a mention that to retry you should use the `--limit` switch, as seen below:

```
to retry, use: --limit @/home/headmin/cobbler-deploy.retry
```

This is a standard Ansible message but it is not needed in this context. Please use our instructions described above to remove your systems, if necessary, from Cobbler before re-running the playbook as the `--limit` is not needed. Note that using the `--limit` switch does not cause any harm and won't prevent the playbook from completing, it's just not a necessary step.

Issue: Dealing With Nodes that Fail to Install

The `bm-reimage.yml` playbook will take every node as far as it can through the baremetal install process. Nodes that fail will not prevent the others from continuing to completion. At the end of the run you will get a list of the nodes (if any) that failed to install.

If you run `bm-reimage.yml` a second time, by default it will target only the failed nodes the second time round (because the others are already marked as "completed"). Alternatively you can target specific nodes for reimage using `-e nodelist` as described in the section below.

The places where you are most likely to see a node fail is timeout in the "wait for shutdown" step, which means that the node did not successfully install an operating system (e.g. it could be stuck in POST) or a timeout in "wait for ssh" at the end of the baremetal install. This means that the node did not come back up after being powered on. To fix the issues you'll need to connect to the nodes' consoles and investigate.

How to re-image an existing node

Once Linux for HPE Helion has been successfully installed on a node, that node will be marked as "installed" and subsequent runs of `bm-reimage.yml` (and related playbooks) will not target it. This is deliberate so that you can't reimage the node by accident. If you do need to reimage existing nodes you will need to use the `-e nodelist` option to target them specifically. For example:

```
ansible-playbook -i hosts/localhost bm-reimage.yml -e nodelist=cpn-0044,cpn-0045
```

Note: You can target all nodes with `-e nodelist=all`

This will power cycle the specified nodes and reinstall the operating system on them, using the existing settings stored in Cobbler.

If you want to change settings for a node in the configuration files, see the *Configuration changes needed after Cobbler deploy* section above.

Issue: Wait for SSH phase in bm-reimage.yml hangs

This issue has been observed during deployment to systems configured with QLogic based BCM578XX network adapters utilizing the bnx2x driver and is currently under investigation. The symptom manifests following a cold boot during deployment at the "wait for SSH" phase in bm-reimage.yml which will result in a hang and eventually timeout, causing the baremetal install to fail. The presence of this particular issue can be further confirmed by connecting to the remote console of the server via iLO or by checking the server's dmesg output for the presence of bnx2_panic_dump messages, similar to the following:

```
bnx2x: [bnx2x_prev_unload_common:10433(eth%d)]Failed to empty BRB, hope for the best  ...
bnx2x: [bnx2x_stats_update:1268(eth0)]storm stats were not updated for 3 times
bnx2x: [bnx2x_stats_update:1269(eth0)]driver assert
bnx2x: [bnx2x_panic_dump:929(eth0)]begin crash dump ----------------  .....
bnx2x: [bnx2x_panic_dump:1163(eth0)]end crash dump ----------------
```

This workaround is completed by rebooting the server.

Subsequent runs of bm-reimage.yml sometimes fail to trigger network install

If you've run the `site.yml` playbook during an install and have a reason to run the `bm-reimage.yml` playbook afterwards, it may fail to trigger the network install over PXE due to the name of the interface changing during this process.

To resolve this issue, you can run the `cobbler-deploy.yml` playbook to refresh these settings and then proceed to run `bm-reimage.yml`.

Issue: Soft lockup at imaging

When imaging your nodes, if you see a kernel error of the type "BUG: soft lockup - CPU#10 stuck...." you should reset the node and make sure it is imaged properly next time. Note that depending on when you get the error, you may have to rerun `cobbler-deploy.yml`. If the bm-reimage playbook says it failed to image the node, then cobbler knows this has occurred and you can reset the node. If not, you can follow the instructions above for *reimaging the node*.

Blank Screen Seen When Monitoring the Imaging Step

If you are watching the `os-install` process on a node via the console output, there can be a pause between 2-3 minutes in length where nothing gets reported to the console screen. This is just after the grub menu has been displayed on a UEFI-based system.

This is normal and nothing to be concerned with. The imaging process will continue on after this pause.

Unable to SSH to a Compute Node after the bm-reimage Step Even Though SSH Keys Are In Place

If after running the `bm-reimage.yml` playbook you get a reported failure when attempting to SSH to one or more Compute nodes stating that permission was denied and you've confirmed that the SSH keys are correctly coped from the lifecycle manager to the Compute node then you can follow the steps below to resolve this issue.

The suspected root cause of this issue is that the bm-reimage playbook is finding a different MAC address when attempting to SSH to the Compute node than was specified in the `servers.yml` file. This could be because you entered an incorrect MAC address or you may have specified the same IP address to two different Compute nodes.

To resolve this issue you can follow these steps:

1. Remove the Compute node that failed from Cobbler using this command:

```
sudo cobbler system remove --name <node name>
```

> **Note:** Use `sudo cobbler system list` to get a list of your nodes in Cobbler.

2. Correct the IP and MAC address information in your `~/helion/my_cloud/definition/data/servers.yml` file.
3. Commit your changes to git:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

4. Re-run the `bm-reimage.yml` playbook and confirm the error is not received again.

Server Crashes During Imaging

If your server suffers a kernel panic just after it gets PXE booted, a possible cause could be that you have an incorrect bus address for the server specified in the `nic_mappings.yml` file.

To resolve this issue you will need to obtain the proper bus address. You can do this by installing HP Linux for OpenStack on the node and once you receive a terminal prompt you can use the command below to obtain the proper bus address:

```
sudo lspci -D | grep -i eth
```

Once you have the proper bus address, follow these steps to finish the reimage:

1. Remove the node that failed from Cobbler using this command:

```
sudo cobbler system remove --name <node name>
```

> **Note:** Use `sudo cobbler system list` to get a list of your nodes in Cobbler.

2. Correct the bus address in your `~/helion/my_cloud/definition/data/nic_mappings.yml` file.
3. Commit your changes to git:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

4. Re-run the `bm-reimage.yml` playbook and confirm the error is not received again.

Issues while Updating Configuration Files

Configuration Processor Fails Due to Wrong yml Format

If you receive the error below when running the configuration processor then you may have a formatting error:

```
TASK: [fail msg="Configuration processor run failed, see log output above for details"]
```

First you should check the ansible log in the location below for more details on which yml file in your input model has the error:

```
~/.ansible/ansible.log
```

Check the configuration file to locate and fix the error, keeping in mind the following tips below.

Check your files to ensure that they don't contain the following:

- Non-ascii characters
- Unneeded spaces

Once you have fixed the formatting error in your files, commit the changes with these steps:

1. Commit your changes to git:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "My config or other commit message"
```

2. Re-run the configuration processor playbook and confirm the error is not received again.

Configuration processor fails with provider network OCTAVIA-MGMT-NET error

If you receive the error below when running the configuration processor then you have not correctly configured your VLAN settings for Octavia.

```
"###############################################################################",
"# The configuration processor failed.  ",
"#   config-data-2.0        ERR: Provider network OCTAVIA-MGMT-NET host_routes: destination '192.168.10.0/24' is not defined as a Network in the input model. Add 'external:
 True' to this host_route if this is for an external network.",
"###############################################################################"
```

To resolve the issue, ensure that your settings in ~/helion/my_cloud/definition/data/neutron/neutron_config.yml are correct for the VLAN setup for Octavia.

Changes Made to your Configuration Files

If you have made corrections to your configuration files and need to re-run the Configuration Processor, the only thing you need to do is commit your changes to your local git:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "commit message"
```

You can then re-run the configuration processor:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

Configuration Processor Fails Because Encryption Key Does Not Meet Requirements

If you choose to set an encryption password when running the configuration processor, you may receive the following error if the chosen password does not meet the complexity requirements:

```
###############################################################################
            # The configuration processor failed.
            #   encryption-key ERR: The Encryption Key does not meet the following requirement(s):
            #      The Encryption Key must be at least 12 characters
            #      The Encryption Key must contain at least 3 of following classes of characters: Uppercase Letters, Lowercase Letters, Digits, Punctuation
###############################################################################
```

If you receive the above error, simply run the configuration processor again and select a password that meets the complexity requirements detailed in the error message:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

Issues while Deploying the Cloud

Issue: If the site.yml playbook fails, you can query the log for the reason

Ansible is good about outputting the errors into the command line output, however if you'd like to view the full log for any reason the location is:

```
~/.ansible/ansible.log
```

This log is updated real time as you run ansible playbooks.

> **Tip:** Use grep to parse through the log. Usage: `grep <text> ~/.ansible/ansible.log`

Issue: How to Wipe the Disks of your Machines

If you have re-run the `site.yml` playbook, you may need to wipe the disks of your nodes

You would generally run the playbook below after re-running the `bm-reimage.yml` playbook but before you re-run the `site.yml` playbook.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts wipe_disks.yml
```

The playbook will show you the disks to be wiped in the output and allow you to confirm that you want to complete this action or abort it if you do not want to proceed. You can optionally use the `--limit <NODE_NAME>` switch on this playbook to restrict it to specific nodes.

If you receive an error stating that `osconfig` has already run on your nodes then you will need to remove the `/etc/hos/osconfig-ran` file on each of the nodes you want to wipe with this command:

```
sudo rm /etc/hos/osconfig-ran
```

That will clear this flag and allow the disk to be wiped.

Issue: Errors during create_db Task

If you are doing a fresh installation on top of a previously used system and you have not completely wiped your disk prior to the installation, you may run into issues when the installation attempts to create new Vertica databases. The recommendation here is to wipe your disks and re-start the installation, however we have included a playbook for cleaning your Vertica databases if they need to be re-used.

To run this playbook, follow these steps:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts monasca-vertica-dbclean.yml
```

Issue: Freezer installation fails if an independent network is used for the External_API.

Currently the Freezer installation fails if an independent network is used for the External_API. If you intend to deploy the External API on an independent network, the following changes need to be made:

In `roles/freezer-agent/defaults/main.yml` add the following line:

```
backup_freezer_api_url: "{{ FRE_API | item('advertises.vips.private[0].url', default=' ') }}"
```

In `roles/freezer-agent/templates/backup.osrc.j2` add the following line:

```
export OS_FREEZER_URL={{ backup_freezer_api_url }}
```

Error Received if Root Logical Volume is Too Small

When running the `site.yml` playbook, you may receive the error below if your root logical-volume is too small:

```
2015-09-29 15:54:02,751 p=26345 u=stack |  TASK: [osconfig | disk config | Extend root LV] *******************************
2015-09-29 15:54:03,021 p=26345 u=stack |  failed: [helion-ccp-swpac-m1-mgmt] => (item=({'physical_volumes': ['/dev/sda_root'], 'consumer': {'name': 'os'},
'name': 'hlm-vg'}, {'mount': '/', 'fstype': 'ext4', 'name': 'root', 'size': '10%'})) => {"changed": true, "cmd": ["lvextend", "-l", "10%VG", "/dev/hlm-
vg/root"], "delta": "0:00:00.022983", "end": "2015-09-29 10:54:18.925855", "failed": true, "failed_when_result": true, "item": [{"consumer": {"name":
"os"}, "name": "hlm-vg", "physical_volumes": ["/dev/sda_root"]}, {"fstype": "ext4", "mount": "/", "name": "root", "size": "10%"}], "rc": 3, "start": "2015-
09-29 10:54:18.902872", "stdout_lines": [], "warnings": []}
2015-09-29 15:54:03,022 p=26345 u=stack |  stderr:   New size given (7128 extents) not larger than existing size (7629 extents)
```

The specific part of this error to parse out and resolve is:

```
stderr:   New size given (7128 extents) not larger than existing size (7629 extents)
```

The error also references the root volume:

```
"name": "root", "size": "10%"
```

The problem is that the root logical-volume, as specified in the `disks_controller.yml` file, is set to `10%` of the overall physical volume and this value is too small.

To resolve this issue you need to ensure that the percentage is set properly for the size of your logical-volume. The default values in the configuration files is based on a 500GB disk, so if your logical-volumes are smaller you may need to increase the percentage so there is enough room.

Multiple Keystone Failures Received during Site.yml

If you receive the Keystone error below during your `site.yml` run then follow these steps:

```
TASK: [OPS-MON | _keystone_conf | Create Ops Console service in Keystone] *****
failed: [helion-cp1-c1-m1-mgmt] => {"failed": true}
msg: An unexpected error prevented the server from fulfilling your request. (HTTP 500) (Request-ID: req-23a09c72-5991-4685-b09f-df242028d742), failed

FATAL: all hosts have already failed -- aborting
```

The most likely cause of this error is that the virtual IP address is having issues and the Keystone API communication through the virtual IP address is not working properly. You will want to check the Keystone log on the controller where you will likely see authorization failure errors.

Verify that your virtual IP address is active and listening on the proper port on all of your controllers using this command:

```
netstat -tplan | grep 35357
```

Ensure that your lifecycle manager did not pick the wrong (unusable) IP address from the list of IP addresses assigned to your Management network.

The lifecycle manager will take the first available IP address after the `gateway-ip` defined in your `~/helion/my_cloud/definition/data/networks.yml` file. This IP will be used as the virtual IP address for that particular network. If this IP address is used and reserved for another purpose outside of your HPE Helion OpenStack deployment then you will receive the error above.

To resolve this issue we recommend that you utilize the `start-address` and possibly the `end-address` (if needed) options in your `networks.yml` file to further define which IP addresses you want your cloud deployment to use. See *Input Model - Networks* for more details.

After you have made changes to your `networks.yml` file, follow these steps to commit the changes:

1. Ensuring that you stay within the `~/helion` directory, commit the changes you just made:

```
cd ~/helion
git commit -a -m "commit message"
```

2. Run the configuration processor:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
```

3. Update your deployment directory:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost ready-deployment.yml
```

4. Re-run the site.yml playbook:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts site.yml
```

VSA installer failed during deployment

VSA deployment might fails sporadically due to a defect in the HPE StoreVirtual VSA installation files for the KVM host. The following error messages is displayed and VSA installation will hang for more than 45-60 minutes:

```
14:10:13 TASK: [VSA-DEP | deploy | Create VSA appliance] *******************************
14:18:32 changed: [h1m003-cp1-vsa0003-mgmt]
```

```
15:32:36 fatal: [hlm003-cp1-vsa0002-mgmt] => SSH Error: Shared connection to 10.240.20.200 closed.
15:32:36 It is sometimes useful to re-run the command using -vvvv, which prints SSH debug output to help diagnose the issue.
15:32:36 fatal: [hlm003-cp1-vsa0001-mgmt] => SSH Error: Shared connection to 10.240.20.199 closed.
15:32:36 It is sometimes useful to re-run the command using -vvvv, which prints SSH debug output to help diagnose the issue.
15:32:37
```

Note: VSA deployment can take approximately usually 15-30 minutes, depending on how many disks are configured. Do not terminate the installation until 60 minutes have passed and/or you see the error.

To correct this issue:

1. Log into the failed VSA node.
2. Change to the root user:

```
sudo -i
```

3. Destroy the VSA VM:

```
virsh destroy <VSA_VM_Name>
virsh undefine <VSA_VM_Name>
```

For example:

```
virsh destroy VSA-VM-vsa2
virsh undefine VSA-VM-vsa2
```

To get the name of the VSA VM, execute the following command:

```
virsh list --all
Id    Name                             State
-------------------------------------------------
 2    VSA-VM-vsa2                      running
```

4. Destroy the VSA network:

```
virsh net-destroy vsa-network
virsh net-undefine vsa-network
```

5. Destroy the VSA storage-pool:

```
virsh pool-destroy vsa-storage-pool
virsh pool-undefine vsa-storage-pool
```

6. Remove the `vsa-installer` directory:

```
rm -rf /home/vsa-installer
```

7. Remove the VSA image:

```
rm -rf /mnt/state/vsa-kvm-storage
```

8. Reboot the VSA VM:

```
reboot
```

9. Log into your lifecycle manager.
10. Run the deployment playbooks from the resulting scratch directory:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts site.yml
```

HPE Helion OpenStack® 5.0: Troubleshooting the Block Storage Backend Configuration
We have gathered some of the common issues that occur during the Block Storage configuration steps and organized them by product.

We have gathered some of the common issues that occur during the Block Storage configuration steps and organized them by product.

Troubleshooting the VSA Configuration

Issue: You have forgotten the identifying information about your VSA cluster

If you have forgotten the cluster name or IP address after creating the cluster in the CMC utility then you can perform the following steps to retrieve them:

1. Log in to CMC console using the `/opt/HP/StoreVirtual/UI/jre/bin/java -jar /opt/HP/StoreVirtual/UI/UI.jar` command from the lifecycle manager/deployer node.
2. Log in to the Management group.
3. On the left side of the CMC console screen, click **mgmt**. It expands and you will be able to see the cluster name.
4. Click the cluster name. The page on the right hand side will populate with the cluster information.
5. Click `iSCSI` on the right hand side of the page. The virtual IP address will be displayed in the tabular form. You can view your cluster ID here.

Issue: Error During VSA Deployment Exhibits When Running site.yml or hlm-deploy.yml

When you are running either `site.yml` or `hlm-deploy.yml` and you get either of the errors below during the VSA deployment task, there is an error in your locale settings.

Errors:

```
TASK: [VSA-DEP | deploy | Create VSA appliance] ********************************
failed: [helion-cp1-vsa0001-mgmt] => {"changed": true, "cmd": ["./vsa_automation", "-f", "/etc/vsa/vsa_disks.json", "-ao", "False"], "delta": "0:00:00.368050", "end": "2015-10-27
 11:02:18.378824", "rc": 1, "start": "2015-10-27 11:02:18.010774", "warnings": []}
stderr: libvirt: Network Driver error : Network not found: no network with matching name 'vsa-network'
Traceback (most recent call last):
    File "./vsa_automation", line 83, in <module>
      deployer.install_vsa()
    File "/opt/stack/venv/storevirtual-installer-20151022T080854Z/lib/python2.7/site-packages/storevirtual_installer/deployer.py", line 88, in install_vsa
      self._roll_back_installation()
    File "/opt/stack/venv/storevirtual-installer-20151022T080854Z/lib/python2.7/site-packages/storevirtual_installer/deployer.py", line 479, in _roll_back_installation
      self._vsa_network_destroy()
    File "/opt/stack/venv/storevirtual-installer-20151022T080854Z/lib/python2.7/site-packages/storevirtual_installer/deployer.py", line 493, in _vsa_network_destroy
      raise vsa_excs.NetworkDestroyFailed(msg)
storevirtual_installer.vsa_excs.NetworkDestroyFailed: Failed to destroy and undefine the network

FATAL: all hosts have already failed -- aborting
```

OR

```
Unable to get XML definition of storage pool ----------------------------------------. Err: exit status 1: setlocale: No such file or directory
error: failed to get pool '----------------------------------------'
error: Storage pool not found: no storage pool with matching name '----------------------------------------'
```

Fix:

Make sure that the locale variables on the lifecycle manager are valid and then run the playbook again. You can set this variable to set your locale:

```
export LC_ALL=C
```

HPE Helion OpenStack® 5.0: Troubleshooting the ESX
We have gathered some of the common issues that occur during installation and organized them by when they occur during the installation.

This section contains troubleshooting tasks for your HPE Helion OpenStack®5.0 for ESX.

Issue: If hlm-ux-services.service is not running, the EON resource-activate and resource-deactivate commands fails

If you perform any maintenance work or reboot the lifecycle manager/deployer node, make sure to restart the HLM UX service for standalone deployer node and shared HLM/controller node based on your environment.

For standalone deployer node, execute `hlm-start.yml` playbook to restart the HLM UX services on the deployer node after a reboot.

For shared deployer/controller node, execute `hlm-start.yml` playbook on all the controllers to restart HLM UX services.

For example:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts hlm-start.yml --limit <host name of the HLM node or HLM Node/Shared Controller>
```

Issue: ESX onboard Compute

1. If there is a failure in provisioning phase

   a. The spawned applicances will be deleted.
   b. Check for `/var/log/eon/eon-conductor.log` for specific errors to troubleshoot the error.

2. If there is a failures in activating phase

   a. The spawned appliances will be left intact, and you can try executing activate command again.

      b. Check the `~/.ansible/ansible.log` or `/var/log/hlm-ux-service/service.log` in the `hlm-deployer`.

      In case of SSH failure, check the background connectivity and try to ping again.

3. Failures leading to inconsistent Database (DB) state, execute the following command:

```
eon resource-deactivate <id> --forced True
```

Issue: Migrate eon-conductor

Currently eon-conductor service is running in the first node of the control plane.

Procedure to migrate eon-conductor

Perform the following steps to migrate eon-conductor service to any other node in the control plane.

1. SSH to lifecycle manager.
2. Change the directory.

```
cd /home/user/scratch/ansible/next/hos/ansible
```

3. Search where the eon conductor is running currently. In the following example, eon-conductor is running in `helion-cp1-c1-m1-mgmt`.

```
(helion-cp1-c1-m1-mgmt)
stack@helion-cp1-c1-m1-mgmt:~$ ps aux | grep eon
eon       11236  5.5  0.0 142104 52088 ?        Ss   18:53   0:01 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon       11281 23.2  0.1 401964 238080 ?       Ss   18:53   0:06 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-conductor/venv/bin/eon-conductor --
config-file=/opt/stack/service/eon-conductor-20160503T082438Z/etc/eon/eon-conductor.conf
eon       11590  1.5  0.0 144420 49084 ?        S    18:53   0:00 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon       11591  1.4  0.0 144416 49056 ?        S    18:53   0:00 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon       11592  1.1  0.0 144420 48828 ?        S    18:53   0:00 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
stack     12086  0.0  0.0  12736  2064 pts/9    S+   18:54   0:00 grep eon
```

```
(helion-cp1-c1-m2-mgmt)
stack@helion-cp1-c1-m2-mgmt:~$ ps aux | grep eon
```

```
eon         7050  1.9  0.0 142100 52052 ?         Ss   18:53   0:03 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon         7677  1.1  0.0 144552 49156 ?         S    18:53   0:02 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon         7678  1.2  0.0 144544 49256 ?         S    18:53   0:02 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon         7679  1.2  0.0 144548 49136 ?         S    18:53   0:02 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
stack      13523  0.0  0.0  12732  2076 pts/0     S+   18:56   0:00 grep eon
```

4. Stop the eon service.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts eon-stop.yml
```

Example:

```
stack@helion-cp1-c1-m1-mgmt:~/scratch/ansible/next/hos/ansible$ ansible-playbook -i hosts/verb_hosts eon-stop.yml

PLAY [EON-API] *****************************************************

TASK: [eon-common | stop | stop eon-api service] ******************************
changed: [helion-cp1-c1-m3-mgmt]
changed: [helion-cp1-c1-m1-mgmt]
changed: [helion-cp1-c1-m2-mgmt]

PLAY [EON-CND--first-member] **************************************

TASK: [eon-common | stop | stop eon-conductor service] ************************
changed: [helion-cp1-c1-m1-mgmt]

PLAY [EON-ONEVIEW--first-member] **********************************
skipping: no hosts matched

PLAY RECAP ********************************************************
eon-common | stop | stop eon-conductor service ------------------------- 8.39s
eon-common | stop | stop eon-api service ------------------------------- 0.42s
-----------------------------------------------------------------
Total: --------------------------------------------------------- 8.99s
helion-cp1-c1-m1-mgmt      : ok=2   changed=2   unreachable=0   failed=0
helion-cp1-c1-m2-mgmt      : ok=1   changed=1   unreachable=0   failed=0
helion-cp1-c1-m3-mgmt      : ok=1   changed=1   unreachable=0   failed=0
```

5. Edit the /home/user/scratch/ansible/next/hos/ansible/hosts/verb_hosts. In the following example, we change the node name from helion-cp1-c1-m1-mgmt to helion-cp1-c1-m2-mgmt in the [EON-CND--first-member:children] group to move the eon-conductor service to node helion-cp1-c1-m2-mgmt.

```
[EON-CND--first-member:children]   (ORIGINAL)
helion-cp1-c1-m1-mgmt  <- original node that eon-conductor is setup on.

[EON-CND--first-member:children]   (MODIFIED)
helion-cp1-c1-m2-mgmt <- new node that eon-conductor is setup on.
```

6. Execute the following playbook in sequence.

```
ansible-playbook -i hosts/verb_hosts eon-deploy.yml
ansible-playbook -i hosts/verb_hosts eon-start.yml
```

Example:

```
TASK: [eon-conductor | start | start eon-conductor service] *******************
changed: [helion-cp1-c1-m2-mgmt]
```

```
PLAY RECAP *********************************************************************
eon-api | start | start eon-api service -------------------------------- 0.26s
eon-conductor | start | start eon-conductor service -------------------- 0.23s
eon-api | start | activate the latest installed version ---------------- 0.07s
eon-api | start | restart eon-api service ------------------------------ 0.07s
eon-conductor | start | restart eon-conductor service ------------------ 0.02s
eon-conductor | start | activate the latest installed version ---------- 0.02s
-------------------------------------------------------------------------------
Total: --------------------------------------------------------------- 1.09s
helion-cp1-c1-m1-mgmt     : ok=1    changed=0    unreachable=0    failed=0
helion-cp1-c1-m2-mgmt     : ok=2    changed=0    unreachable=0    failed=0 <- Service is moved and started
helion-cp1-c1-m3-mgmt     : ok=1    changed=0    unreachable=0    failed=0
```

7. Ensure that the conductor is running on the node that you have migrated. In the following example observe that the eon-conductor have migrated to node `helion-cp1-c1-m2-mgmt`.

```
(helion-cp1-c1-m1-mgmt)
stack@helion-cp1-c1-m1-mgmt:~$ ps aux | grep eon
stack     2624  0.0  0.0  12732  2132 pts/9    S+   18:46   0:00 grep eon
eon      27786  1.3  0.0 142112 52064 ?        Ss   16:21   1:55 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon      27816  1.2  0.0 149712 55060 ?        S    16:21   1:50 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon      27817  1.2  0.0 150380 55724 ?        S    16:21   1:49 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon      27818  1.2  0.0 148516 53844 ?        S    16:21   1:49 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf

(helion-cp1-c1-m2-mgmt)
stack@helion-cp1-c1-m2-mgmt:~$ ps aux | grep eon
eon       1567  1.3  0.0 142108 52076 ?        Ss   16:21   1:56 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon       1606  1.2  0.0 144816 49888 ?        S    16:21   1:51 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon       1607  1.2  0.0 145068 49888 ?        S    16:21   1:50 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon       1608  1.2  0.0 145064 49888 ?        S    16:21   1:49 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-api/venv/bin/eon-api --config-file=/
opt/stack/service/eon-api-20160503T082438Z/etc/eon/eon-api.conf
eon      22720  0.0  0.1 404872 241220 ?       Ss   16:15   0:08 /opt/stack/venv/eon-20160503T082438Z/bin/python2 /opt/stack/service/eon-conductor/venv/bin/eon-conductor --
config-file=/opt/stack/service/eon-conductor-20160503T082438Z/etc/eon/eon-conductor.conf
stack    31965  0.0  0.0  12732  2308 pts/0    S+   18:50   0:00 grep eon
```

Issue: ESX Cluster shows UNKNOWN in OpsConsole

In the OpsConsole Alarms dashboard, if all the alarms for ESX cluster are showing UNKNOWN then restart the `monasca-agent` running in ESX compute proxy.

1. SSH to the respective compute proxy. You can find the hostname of the proxy from the dimensions list shown against the respective alarm.

2. Restart the `monasca-agent` service.

```
sudo systemctl restart monasca-agent
```

Issue: Unable to view the VM console in Horizon UI

By default the gdbserver firewall is disabled in ESXi host which results in a Handshake error when accessing the VM instance console in the Horizon UI.
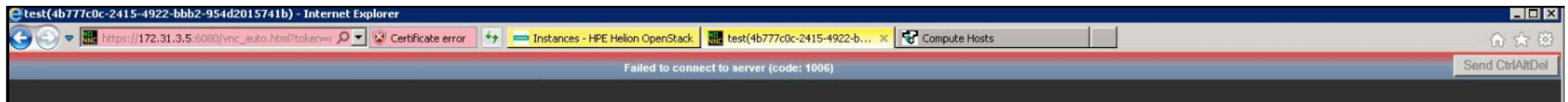
Procedure to enable gdbserver

1. Login to vSphere Client.
2. Select the ESXi Host and click **Configuration** tab in the menu bar. You must perform the following actions on all the ESXi hosts in the compute clusters.

3. On the left hand side select **Security Profile** from the list of **Software**. Click **Properties** on the right hand side.

10.244.209.38 VMware ESXi, 6.0.0, 3380124

Getting Started | Summary | Virtual Machines | Performance | Configuration | Tasks & Events | Alarms | Permissions | Maps

Hardware

Processors
Memory
Storage
Networking
Storage Adapters
Network Adapters
Advanced Settings
Power Management

Software

Licensed Features
Time Configuration
DNS and Routing
Authentication Services
Power Management
Virtual Machine Startup/Shutdown
Virtual Machine Swapfile Location
Security Profile ①
Host Cache Configuration
System Resource Reservation
Agent VM Settings
Advanced Settings

Security Profile

Services   Refresh   Properties...

SNMP Server
PC/SC Smart Card Daemon
Load-Based Teaming Daemon
vSphere High Availability Agent
ESXi Shell
X.Org Server
VMware vCenter Agent
NTP Daemon
Active Directory Service
VProbe Daemon
SSH
Syslog Server
Direct Console UI
CIM Server

Firewall ②   Refresh   Properties...

Incoming Connections
vMotion   8000 (TCP)   All
vSphere Web Client   902,443 (TCP)   All
SNMP Server   161 (UDP)   All
Fault Tolerance   8100,8200,8300 (TCP,UDP)   All
vsanvp   8080 (TCP)   All
Virtual SAN Clustering Service 12345,23451,12321 (UDP)   All
CIM SLP   427 (UDP,TCP)   All
vSphere High Availability Agent 8182 (TCP,UDP)   All
NFC   902 (TCP)   All
SSH Server   22 (TCP)   All
DHCP Client   68 (UDP)   All
DVSSync   8301,8302 (UDP)   All
gdbserver   1000-9999,50000-50999 (TCP)   All
vSphere Web Access   80 (TCP)   All
DNS Client   53 (UDP)   All
CIM Server   5988 (TCP)   All
CIM Secure Server   5989 (TCP)   All
Virtual SAN Transport   2233 (TCP)   All
Outgoing Connections
vMotion   8000 (TCP)   All
HPProvider   63000-63005 (TCP)   All
Software iSCSI Client   3260 (TCP)   All
Fault Tolerance   80,8100,8200,8300 (TCP,UDP)   All
vsanvp   8080 (TCP)   All
Virtual SAN Clustering Service 12345,23451,12321 (UDP)   All
CIM SLP   427 (UDP,TCP)   All
vSphere High Availability Agent 8182 (TCP,UDP)   All
NFC   902 (TCP)   All
DHCP Client   68 (UDP)   All
DVSSync   8302,8301 (UDP)   All
VMware vCenter Agent   902 (UDP)   All
rabbitmqproxy   5671 (TCP)   All
HBR   31031,44046 (TCP)   All
vCenter Update Manager   80,9000-9100 (TCP)   All
DNS Client   53 (UDP,TCP)   All
Virtual SAN Transport   2233 (TCP)   All

Firewall Properties box displays.
4. Select **gdbserver** checkbox and click **OK**.

HPE Helion OpenStack® 5.0: Post-Installation Overview

Once you have completed your cloud deployment, these are some of the common post-installation tasks you may need to perform. Take a look at the descriptions below to determine which of these you need to do.

HPE Helion OpenStack® 5.0: Cloud Verification

Once you have completed your cloud deployment, these are some of the common post-installation tasks you may need to perform to verify your cloud installation.

HPE Helion OpenStack® 5.0: API Verification
We provide a tool and some steps to help you verify that your cloud deployed correctly.

HPE Helion OpenStack 5.0 provides a tool, Tempest, that you can use to verify that your cloud deployment completed successfully:

- *Prerequisites*
- *Tempest Integration Tests*

  - *Running the Tests*
  - *Viewing Test Results*
  - *Customizing the Test Run*
- *Verifying Your Block Storage Backend*
- *Validate Your Object Storage (Swift) Setup*

Prerequisites

The verification tests rely on you having an external network setup and a cloud image in your image (Glance) repository. Run the following playbook to configure your cloud:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts hlm-cloud-configure.yml
```

Note: In HPE Helion OpenStack 5.0, the EXT_NET_CIDR setting for the external network is now specified in the input model - see *neutron-external-networks*.

Tempest Integration Tests

Tempest is a set of integration tests for OpenStack API validation, scenarios, and other specific tests to be run against a live OpenStack cluster. In HPE Helion OpenStack 5.0, Tempest has been modelled as a service and this gives you the ability to locate Tempest anywhere in the cloud. It is recommended that you install Tempest on your lifecycle manager node - that is where it resides by default in a new installation.

A version of the upstream *Tempest* integration tests is pre-deployed on the HPE Helion OpenStack lifecycle manager node. For details on what Tempest is testing, you can check the contents of this file on your lifecycle manager:

```
/opt/stack/tempest/run_filters/ci.txt
```

You can use these embedded tests to verify if the deployed cloud is functional.

For more information on running Tempest tests, see *Tempest - The OpenStack Integration Test Suite*.

Important: Running these tests requires access to the deployed cloud's identity admin credentials

Tempest creates and deletes test accounts and test resources for test purposes.

In certain cases Tempest might fail to clean-up some of test resources after a test is complete, for example in case of failed tests.

Running the Tests

To run the default set of Tempest tests:

1. Log in to the lifecycle manager.
2. Ensure you can access your cloud:

```
cd ~/scratch/ansible/next/hos/ansible
```

```
ansible-playbook -i hosts/verb_hosts cloud-client-setup.yml
source /etc/environment
```

3. Run the tests:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts tempest-run.yml
```

Optionally, you can *customize Tempest to run specific tests*.

Viewing Test Results

Tempest is deployed under `/opt/stack/tempest`. Test results are written in a log file in the following directory:

```
/opt/stack/tempest/logs
```

A detailed log file is written to:

```
/opt/stack/tempest/tempest.log
```

The results are also stored in the `testrepository` database.

To access the results after the run:

1. Log in to the lifecycle manager.
2. Change to the `tempest` directory and list the test results:

```
cd /opt/stack/tempest
./venv/bin/testr last
```

> ⚠ **Important:** If you encounter an error saying "local variable 'run_subunit_content' referenced before assignment", you may need to log in as the `tempest` user to run this command. This is due to a known issue as reported *here*.

See *Test Repository Users Manual* for more details on how to manage the test result repository.

Customizing the Test Run

There are several ways available to customize which tests will be executed.

- *Run Tests for Specific Services and Exclude Specific Features*
- *Run Tests Matching a Series of White and Blacklists*

Run Tests for Specific Services and Exclude Specific Features

Tempest allows you to test specific services and features using the `tempest.conf` configuration file.

A working configuration file with inline documentation is deployed under `/opt/stack/tempest/etc/`.

To use this, follow these steps:

1. Log in to the lifecycle manager.
2. Edit the `/opt/stack/tempest/configs/tempest_region1.conf` file.
3. To test specific service, edit the `[service_available]` section and clear the comment character # and set a line to `true` to test that service or `false` to not test that service.

```
cinder = true
neutron = false
```

4. To test specific features, edit any of the `*_feature_enabled` sections to enable or disable tests on specific features of a service.

```
[volume-feature-enabled]
[compute-feature-enabled]
```

```
[identity-feature-enabled]
[image-feature-enabled]
[network-feature-enabled]
[object-storage-feature-enabled]
```

```
#Is the v2 identity API enabled (boolean value)
api_v2 = true
#Is the v3 identity API enabled (boolean value)
api_v3 = false
```

5. Then run tests normally

Run Tests Matching a Series of White and Blacklists

You can run tests against specific scenarios by editing or creating a run filter file.

Run filter files are deployed under /opt/stack/tempest/run_filters.

Use run filters to whitelist or blacklist specific tests or groups of tests:

- lines starting with # or empty are ignored
- lines starting with + are whitelisted
- lines starting with - are blacklisted
- lines not matching any of the above conditions are blacklisted

If whitelist is empty, all available tests are fed to blacklist. If blacklist is empty, all tests from whitelist are returned.

Whitelist is applied first. The blacklist is executed against the set of tests returned by the whitelist.

To run whitelist and blacklist tests:

1. Log in to the lifecycle manager.

2. Make sure you can access the cloud:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts cloud-client-setup.yml
source /etc/environment
```

3. Run the tests:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts tempest-run.yml  -e run_filter <run_filter_name>
```

Note that the run_filter_name is the name of the run_filter file except for the extension. For instance, to run using the filter from the file /opt/stack/tempest/run_filters/ci.txt, use the following:

```
ansible-playbook -i hosts/verb_hosts tempest-run.yml -e run_filter=ci
```

Documentation on the format of white and black-lists is available at:

```
/opt/stack/tempest/tests2skip.py
```

Example:

The following entries run API tests, exclude tests that are less relevant for deployment validation, such as negative, admin, cli and thirdparty (EC2) tests:

```
+tempest\.api\.*
*[Aa]dmin.*
*[Nn]egative.*
- tempest\.cli.*
- tempest\.thirdparty\.*
```

HPE Helion OpenStack® 5.0: UI Verification

Once you have completed your cloud deployment, these are some of the common post-installation tasks you may need to perform to verify your cloud installation.

 HPE Helion OpenStack® 5.0: Verifying Your Block Storage Backend
We provide steps to verify that your Block Storage backend was setup properly.

The sections below will show you the steps to verify that your Block Storage backend was setup properly.

Create a Volume

Perform the following steps to create a volume using Horizon dashboard.

1. Log into the Horizon dashboard. See *HPE Helion OpenStack 5.0: Cloud Admin Actions with the Dashboard* for details.
2. Under the **Project** menu in the navigation pane, click on **Volumes** under the **Compute** subheading.
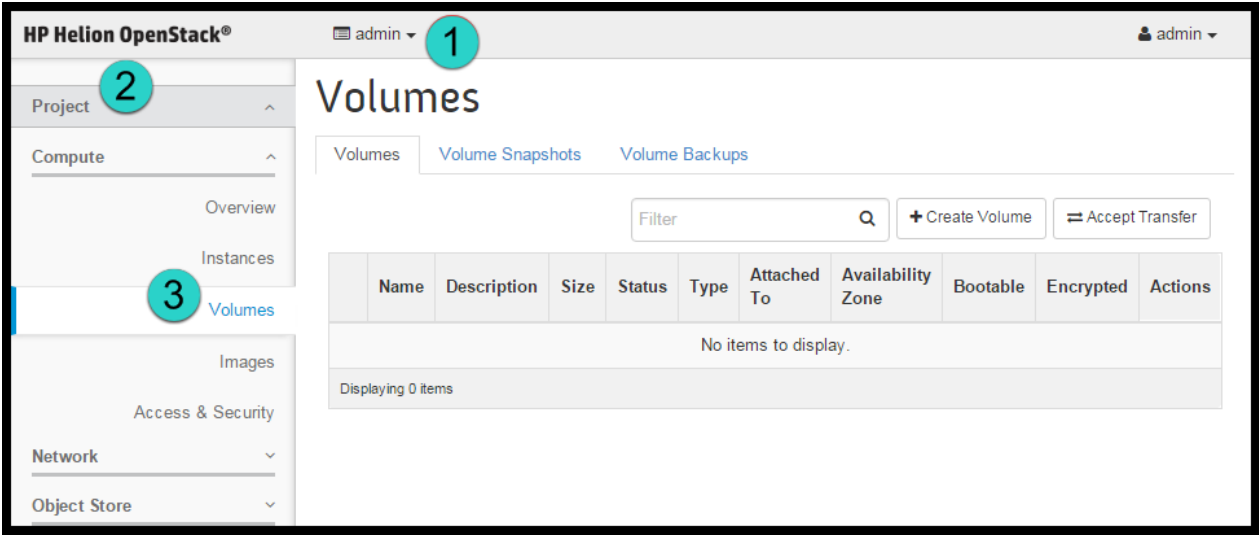
3. On the **Volumes** tabs, click the **Create Volume** button to create a volume.
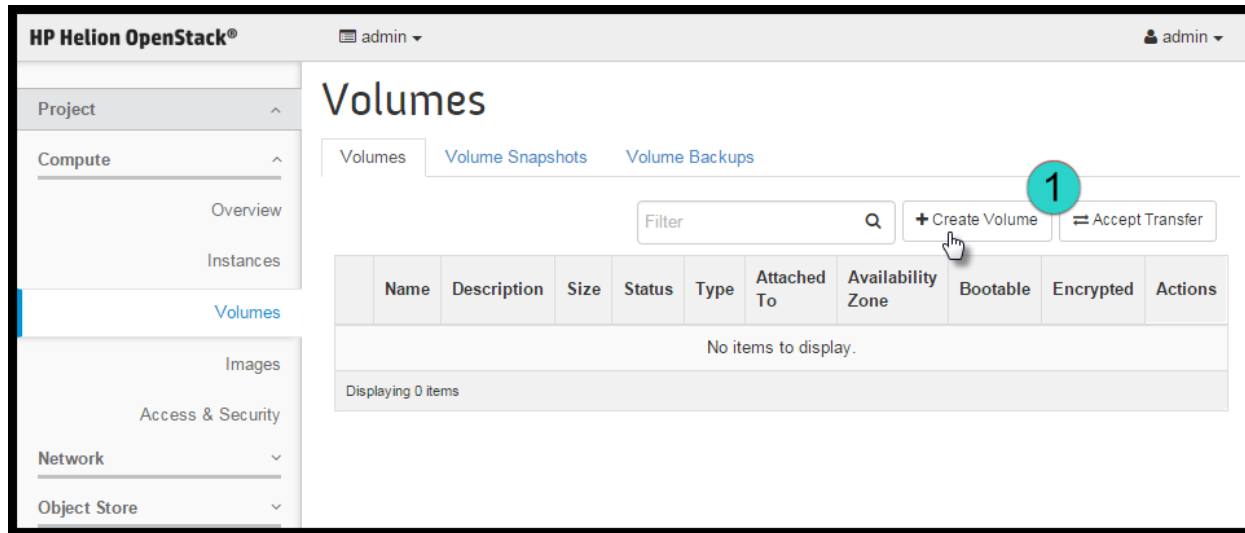
4. In the **Create Volume** options, enter the required details into the fields and then click the **Create Volume** button:

   a. Volume Name - This is the name you specify for your volume.

   b. Description (optional) - This is an optional description for the volume.

   c. Type - Select the volume type you have created for your volumes from the drop down.

   d. Size (GB) - Enter the size, in GB, you would like the volume to be.

   e. Availability Zone - You can either leave this at the default option of **Any Availability Zone** or select a specific zone from the drop down.

The dashboard will then show the volume you have just created.

Attach Volume to an Instance

Perform the following steps to attach a volume to an instance:

1. Log into the Horizon dashboard. See *HPE Helion OpenStack 5.0: Cloud Admin Actions with the Dashboard* for details.
2. Under the **Project** menu in the navigation pane, click the **Instances** under the **Compute** subheading.
3. In the **Action** column, choose the **Edit Attachments** in the menu drop down box next to the instance you want to attach the volume to.
4. In the **Attach To Instance** drop-down, select the volume that you want to attach.
5. Edit the **Device Name** if necessary.
6. Click **Attach Volume** to complete the action.
7. Verify that the volume you attached is displayed in the **Attached To** columns on the **Volumes** screen.

Detach Volume from Instance

Perform the following steps to detach the volume from instance:

1. Log into the Horizon dashboard. See *HPE Helion OpenStack 5.0: Cloud Admin Actions with the Dashboard* for details.
2. Under the **Project** menu in the navigation pane, click the **Instances** under the **Compute** subheading.
3. Click the check box next to the name of the volume you want to detach.
4. In the **Action** column, choose the **Edit Attachments** in the menu drop down box next to the instance you want to attach the volume to.
5. Click **Detach Attachment**. A confirmation dialog box appears.
6. Click **Detach Attachment** to confirm the detachment of the volume from the associated instance.

Delete Volume

Perform the following steps to delete a volume using Horizon dashboard:

1. Log into the Horizon dashboard. See *HPE Helion OpenStack 5.0: Cloud Admin Actions with the Dashboard* for details.
2. Under the **Project** menu in the navigation pane, click on **Volumes** under the **Compute** subheading.
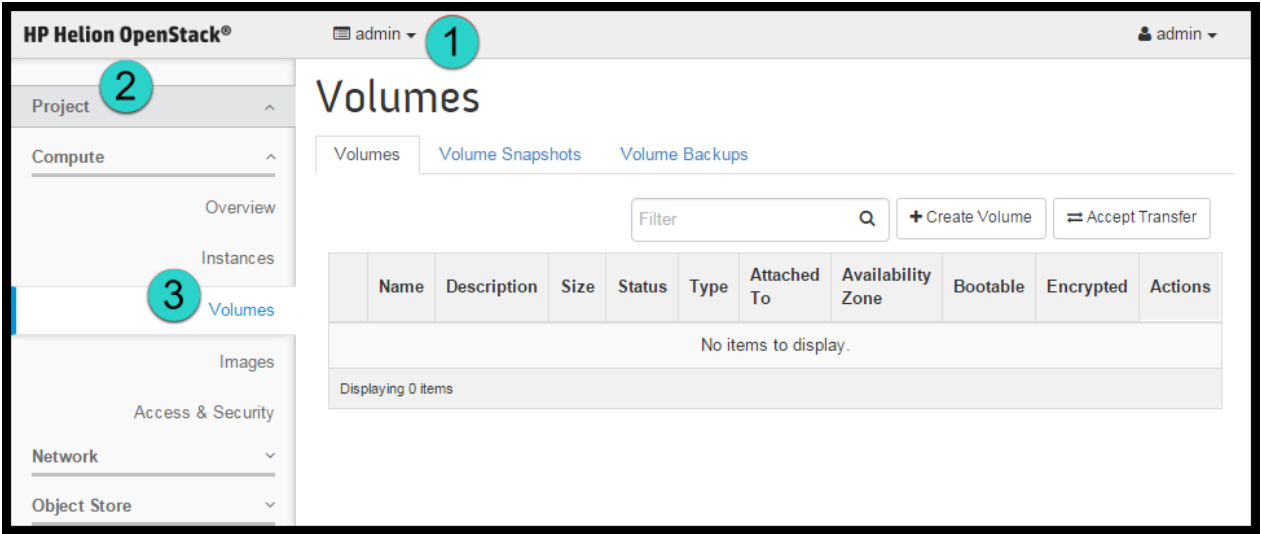
3. In the **Actions** column, click **Delete Volume** next to the volume you would like to delete.
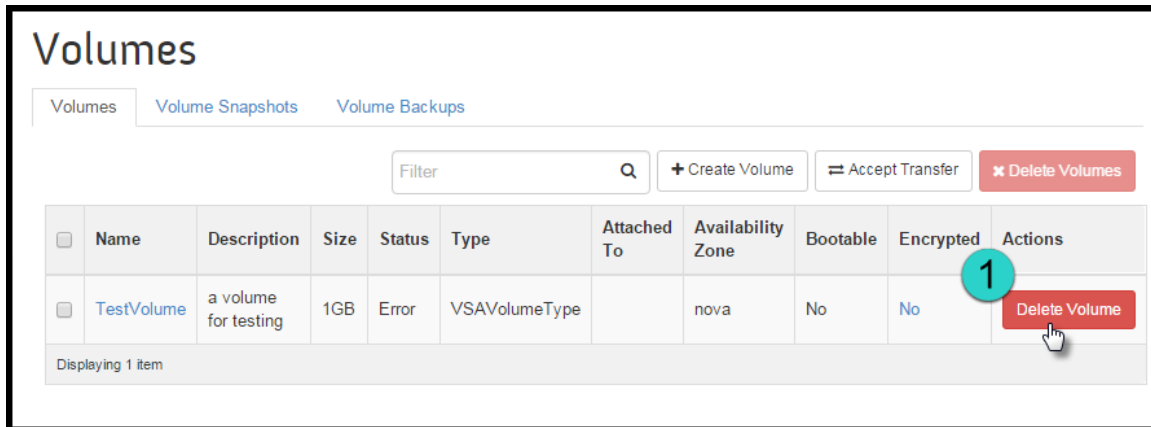
4. To confirm and delete the volume, click **Delete Volume** again.
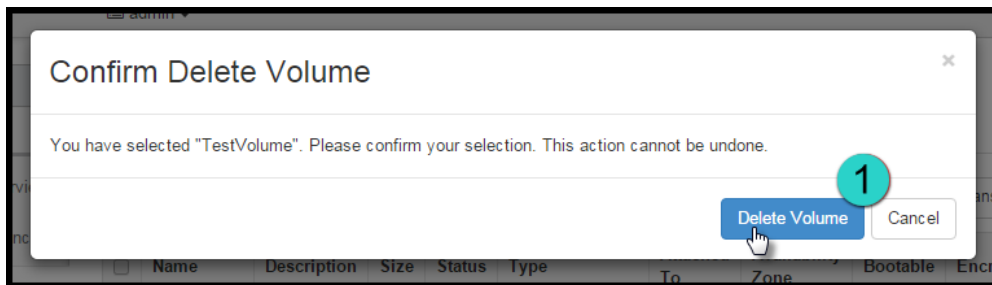
5. Verify that the volume was removed from the **Volumes** screen.

Verifying Your Object Storage (Swift)

Validate That All Servers Have Been Added to the Swift Rings

1. Run the swift-compare-model-rings.yml playbook as follows:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts swift-compare-model-rings.yml
```

2. Search for output similar to the following. Specifically, look at the number of drives that are proposed to be added.

```
TASK: [swiftlm-ring-supervisor | validate-input-model | Print report] *********
ok: [helion-cp1-c1-m1-mgmt] => {
    "var": {
        "report.stdout_lines": [
            "Rings:",
            "  ACCOUNT:",
            "    ring exists",
            "    no device changes",
```

```
                "    ring will be rebalanced",
                "  CONTAINER:",
                "    ring exists",
                "    no device changes",
                "    ring will be rebalanced",
                "  OBJECT-0:",
                "    ring exists",
                "    no device changes",
                "    ring will be rebalanced"
            ]
        }
    }
}
```

3. If the text contains "no device changes" then the deploy was successful and no further action is needed.

4. If there are more drives need to be added, it indicates that the deploy failed on some nodes and that you restarted the deploy to include those nodes. However, the nodes are not in the Swift rings because enough time has not elapsed to allow the rings to be rebuilt. You have two options to continue:

   a. Repeat the deploy. There are two steps involved as follows:

      a. Delete the ring builder files as described in *Restarting the Object Storage Deployment* .

      b. Repeat the installation process starting by running the `site.yml` playbook as described in the *Deploy the Cloud* section of the installation instructions.

   b. Rebalance the rings several times until all drives are incorporated in the rings. This process may take several hours to complete (because you need to wait one hour between each rebalance). The steps are as follows:

      a. Change the min-part-hours to 1 hour. See *Changing min-part-hours in Swift*.

      b. Use the "First phase of ring rebalance" and "Final rebalance phase" as described in *Applying Input Model Changes to Existing Rings*. The "Weight change phase of ring rebalance" does not apply because you have not set the weight-step attribute at this stage.

      c. Set the min-part-hours to the recommended 16 hours as described in *How to Change min-part-hours.*

If you receive errors during the validation, read the *Interpreting the Swift Validate Input Model Output* documentation.

Verify the Object Storage (Swift) Operations

Full details on how to verify the operations can be found in *HPE Helion OpenStack 5.0: Running the Swift Dispersion Report*.

HPE Helion OpenStack® 5.0: Uploading an Image for Use

To create a Compute instance, you need to obtain an image that you can use. The HPE Helion OpenStack lifecycle manager provides an Ansible playbook that will download a CirrOS Linux image, and then upload it as a public image to your image repository for use across your projects.

Running the Playbook

Use the following command to run this playbook:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts glance-cloud-configure.yml -e proxy=<PROXY>
```

The table below shows the optional switch that you can use as part of this playbook to specify environment-specific information:

Switch	Description
`-e proxy="<proxy_address:port>"`	Optional. If your environment requires a proxy for the internet, use this switch to specify the proxy information.

How to Curate Your Own Images

OpenStack has created a guide to show you how to obtain, create, and modify images that will be compatible with your cloud:

OpenStack Virtual Machine Image Guide

Using the GlanceClient CLI to Create Images

You can use the GlanceClient on a machine accessible to your cloud or it's also installed automatically on your lifecycle manager.

The GlanceClient allows you to create, update, list, and delete images as well as manage your image member lists, which allows you to share access to images across multiple tenants. As with most of the OpenStack CLI tools, you can use the `glance help` command to get a full list of commands as well as their syntax.

If you would like to use the `--copy-from` option when creating an image, you will need to have your Administrator enable the http store in your environment using the instructions outlined at *Allowing the Glance copy-from option in your environment*.

HPE Helion OpenStack® 5.0: Creating an External Network
You must have an external network set up to allow your Compute instances to reach the internet. We show you how to create the external network here.

You must have an external network set up to allow your Compute instances to reach the internet. There are multiple methods you can use to create this external network and we provide two of them here. The HPE Helion OpenStack installer provides an Ansible playbook that will create this network for use across your projects. We also show you how to create this network via the command line tool from your lifecycle manager.

- *Using the Ansible Playbook*
- *Using the NeutronClient CLI*

Notes

If you have multiple regions in your cloud environment, it is important that when you set up your external networks in each region that you allocate unique IP ranges for each. If you have overlapping CIDR's then you risk having the same floating IP being allocated to two different virtual machines which will cause a conflict.

Using the Ansible Playbook

This playbook will query the Networking service for an existing external network, and then create a new one if you do not already have one. The resulting external network will have the name `ext-net` with a subnet matching the CIDR you specify in the command below.

If you need to specify more granularity, for example specifying an allocation pool for the subnet then you should utilize the *NeutronClient CLI* instructions at the bottom of the page.

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts neutron-cloud-configure.yml -e EXT_NET_CIDR=<CIDR>
```

The table below shows the optional switch that you can use as part of this playbook to specify environment-specific information:

Switch	Description
`-e EXT_NET_CIDR=<CIDR>`	Optional. You can use this switch to specify the external network CIDR. If you choose not to use this switch, or use a wrong value, the VMs will not be accessible over the network. This CIDR will be from the `EXTERNAL VM` network. **Note:** If this option is not defined the default value is "172.31.0.0/16"

Using the NeutronClient CLI

For more granularity you can utilize the Neutron command line tool to create your external network.

1. Log in to the lifecycle manager.
2. Source the Admin creds:

```
source ~/service.osrc
```

3. Create the external network and then the subnet using these commands below.

   Creating the network:

```
neutron net-create --router:external <external-network-name>
```

   Creating the subnet:

```
neutron subnet-create <external-network-name> <CIDR> --gateway <gateway> --allocation-pool start=<IP_start>,end=<IP_end> [--disable-dhcp]
```

Where:

Value	Description
external-network-name	This is the name given to your external network. This is a unique value that you will choose. The value `ext-net` is usually used.
CIDR	You can use this switch to specify the external network CIDR. If you choose not to use this switch, or use a wrong value, the VMs will not be accessible over the network. This CIDR will be from the EXTERNAL VM network.
--gateway	Optional switch to specify the gateway IP for your subnet. If this is not included then it will choose the first available IP.
--allocation-pool start end	Optional switch to specify a start and end IP address to use as the allocation pool for this subnet.
--disable-dhcp	Optional switch if you want to disable DHCP on this subnet. If this is not specified then DHCP will be enabled.

Next Steps

Once the external network is created, users can create a Private Network to complete their networking setup. For instructions, see *HPE Helion OpenStack 5.0: Creating a Private Network*.

HPE Helion OpenStack® 5.0: Installing OpenStack Clients

The way OpenStack clients are installed changed in HPE Helion OpenStack 4.0. If you have a standalone deployer, the OpenStack CLI and other clients will not be installed automatically on that node. If you require access to these clients, you will need to follow the procedure below to add the appropriate software.

⚠️ **Important:** In the `entry-scale-esx-kvm-vsa` and `entry-scale-kvm-esx-vsa-mml` example configurations, `eon-client` is installed only on `esx-compute` and `esx-ovsvapp` resources. With the changes to the OpenStack client installation, `eon-client` won't get installed on either dedicated HLM node or Controller nodes. You need to install the `eon-client` on either the dedicated deployer node or controller nodes so that you can use it to add the vCenter and activate the ESX compute clusters.

1. [OPTIONAL] Connect to your standalone deployer and try to use the OpenStack CLI:

```
source ~/keystone.osrc
openstack project list

-bash: openstack: command not found
```

2. Edit the configuration file containing details of you Control Plane, typically `~/helion/my_cloud/definition/data/control_plane`.

3. Locate the stanza for the cluster where you want to install the client(s). For a standalone deployer, this will look like the following extract:

```
    clusters:
      - name: cluster0
        cluster-prefix: c0
        server-role: LIFECYCLE-MANAGER-ROLE
        member-count: 1
        allocation-policy: strict
        service-components:
          - ntp-server
          - lifecycle-manager
```

4. Choose the client(s) you wish to install from the following list of available clients:

```
  - openstack-client
  - ceilometer-client
  - cinder-client
  - designate-client
```

```
    - glance-client
    - heat-client
    - ironic-client
    - keystone-client
    - neutron-client
    - nova-client
    - swift-client
    - monasca-client
    - barbican-client
```

5. Add the client(s) to the list of `service-components` - in this example, we add the `openstack-client` to the standalone deployer:

```
    clusters:
      - name: cluster0
        cluster-prefix: c0
        server-role: LIFECYCLE-MANAGER-ROLE
        member-count: 1
        allocation-policy: strict
        service-components:
          - ntp-server
          - lifecycle-manager
          - openstack-client
          - ceilometer-client
          - cinder-client
          - designate-client
          - glance-client
          - heat-client
          - ironic-client
          - keystone-client
          - neutron-client
          - nova-client
          - swift-client
          - monasca-client
          - barbican-client
```

6. Commit the configuration changes:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "Add explicit client service deployment"
```

7. Run the configuration processor, followed by the `ready-deployment` playbook:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml -e encrypt="" -e rekey=""
ansible-playbook -i hosts/localhost ready-deployment.yml
```

8. Add the software for the clients using the following command:

```
cd /home/stack/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts clients-upgrade.yml
```

9. Check that the software has been installed correctly. In this instance, connect to your standalone deployer and try to use the OpenStack CLI:

```
source ~/keystone.osrc
```

```
openstack project list
```

You should now see a list of projects returned:

```
stack@helion-cp1-c0-m1-mgmt:~$ openstack project list

+---------------------------------+-----------------+
| ID                              | Name            |
+---------------------------------+-----------------+
| 076b6e879f324183bbd28b46a7ee7826 | kronos          |
| 0b81c3a9e59c47cab0e208ea1bb7f827 | backup          |
| 143891c2a6094e2988358afc99043643 | octavia         |
| 1d3972a674434f3c95a1d5ed19e0008f | glance-swift    |
| 2e372dc57cac4915bf06bbee059fc547 | glance-check    |
| 383abda56aa2482b95fb9da0b9dd91f4 | monitor         |
| 606dd3b1fa6146668d468713413fb9a6 | swift-monitor   |
| 87db9d1b30044ea199f0293f63d84652 | admin           |
| 9fbb7494956a483ca731748126f50919 | demo            |
| a59d0c682474434a9ddc240ddfe71871 | services        |
| a69398f0f66a41b2872bcf45d55311a7 | swift-dispersion|
| f5ec48d0328d400992c1c5fb44ec238f | cinderinternal  |
+---------------------------------+-----------------+
```

HPE Helion OpenStack® 5.0: Configuring Transport Layer Security (TLS)

TLS is enabled by default during the installation of HPE Helion OpenStack 5.0 and additional configuration options are available to secure your environment.TLS is enabled by default during the installation of HPE Helion OpenStack 5.0 and additional configuration options are available to secure your environment, as described below.

HPE Helion OpenStack® 5.0: TLS Configuration

In HPE Helion OpenStack 5.0, you can provide your own certificate authority and certificates for internal and public virtual IP addresses (VIPs), and you should do so for any production cloud. The certificates automatically generated by HPE Helion OpenStack are useful for testing and setup, but you should always install your own for production use. Certificate installation is discussed below.

Please read the following if you're using the default cert-name: my-public-cert in your model.

The bundled test cert for public endpoints, located at ~/helion/my_cloud/config/tls/certs/my-public-cert, is now expired but was left in the product in case you changed the content with your valid cert. Please verify if the certificate is expired and generate your own by following the guidelines further down on this page or by using a generic instruction from the web.

You can verify the expiry by running this command:

```
openssl x509 -in ~/helion/my_cloud/config/tls/certs/my-public-cert -noout -enddate
notAfter=Oct  8 09:01:58 2016 GMT
```

Before you begin, the following list of terms will be helpful when generating and installing certificates.

Helion generated public CA	A Helion OpenStack generated public CA (helion_frontend_cacert.crt) is availble for you to use in /usr/local/share/ca-certificates.
Fully qualified domain name (FQDN) of the public VIP	The registered domain name. A FQDN is not mandatory. It is perfectly valid to have no FQDN and use IP addresses instead. Note that you can use FQDNs on public endpoints, and you may change them whenever the need arises.
Certificate authority (CA) certificate	Your certificates must be signed by a CA, such as your internal IT department or a public certificate authority. For this example we will use a self-signed certificate.
Server certificate	It is easy to confuse server certificates and CA certificates. Server certificates reside on the server and CA certificates reside on the client. A server certificate affirms that the server that sent it serves a set of IP addresses, domain names, and set of services. A CA certificate is used by the client to authenticate this claim.
SAN (subject-alt-name)	The set of IP addresses and domain names in a server certificate request: A template for a server certificate.
Certificate signing request (CSR)	A blob of data generated from a certificate request and sent to a CA, which would then sign it, produce a server certificate, and send it back.

External VIP	External virtual IP address
Internal VIP	Internal virtual IP address

The major difference between an external VIP certificate and an internal VIP certificate is that the internal VIP has approximately 40 domain names in the SAN. This is because each service has a different domain name in HPE Helion OpenStack 5.0. So it is unlikely that you can create an internal server certificate before running the configuration processor. But after a configuration processor run, a certificate request would be created for each of your cert-names.

Configuring TLS in the input model

For this example certificate configuration, let's assume there's no FQDN for the external VIP and that you're going to use the default IP address provided by HPE Helion OpenStack 5.0. Let's also assume that for the internal VIP you will use the defaults as well. If you were to call your certificate authority "example-CA," the CA certificate would then be called "example-CA.crt" and the key would be called "example-CA.key." In the following examples, the external VIP certificate will be named "example-public-cert" and the internal VIP certificate will be named "example-internal-cert."

Note: Cautions:

Any time you make a cert change when using your own CA:

- You should use a distinct name from those already existing in config/tls/cacerts. This also means that you should not *reuse* your CA names (and use unique and distinguishable names such as MyCompanyXYZ_PrivateRootCA.crt). A new name is what indicates that a file is new or changed, so reusing a name means that the file is not considered changed even its contents have changed.
- You should not remove any existing CA files from config/tls/cacerts
- If you want to remove an existing CA you must

  1. First remove the file.
  2. Then run:

```
ansible -i hosts/verb_hosts FND-STN -a 'sudo keytool -delete -alias debian:<filename to remove> \
-keystore /usr/lib/jvm/java-7-openjdk-amd64/jre/lib/security/cacerts -storepass changeit'
```

Attention: Be sure to install your own certificate for all production clouds after installing and testing your cloud. If you ever want to test or troubleshoot later, you will be able to revert to the sample certificate to get back to a stable state for testing.

Note: Unless this is a new deployment, do not update both the certificate and the CA together. Add the CA first and then run a site deploy. Then update the certificate and run tls-reconfigure, FND-CLU-stop, FND-CLU-start and then hlm-reconfigure. If a playbook has failed, rerun it with -vv to get detailed error information. The configure, HAproxy restart, and reconfigure steps are included below. If this is a new deployment and you are adding your own certs/CA before running site.yml this caveat does not apply.

You can add your own certificate by following the instructions below. All changes must go into the following file:

```
~/helion/my_cloud/definition/data/network_groups.yml
```

Below are the entries for TLS for the internal and admin load balancers:

```
- provider: ip-cluster
      name: lb
      tls-components:
      - default
      components:
      # These services do not currently support TLS so they are not listed
      # under tls-components
      - nova-metadata
      roles:
      - internal
      - admin
      cert-file: helion-internal-cert
      # The helion-internal-cert is a reserved name and
      # this certificate will be autogenerated. You
      # can bring in your own certificate with a different name

      # cert-file: customer-provided-internal-cert
      # replace this with name of file in "config/tls/certs/"
```

The configuration processor will also create a request template for each named certificate under `info/cert_reqs/` This will be of the form:

```
info/cert_reqs/customer-provided-internal-cert
```

These request templates contain the subject `Alt-names` that the certificates need. You can add to this template before generating your certificate signing request .

You would then send the CSR to your CA to be signed, and once you receive the certificate, place it in `config/tls/certs`

When you bring in your own certificate, you may want to bring in the trust chains (or CA certificate) for this certificate. This is usually not required if the CA is a public signer that is typically bundled with the operating system. However, we suggest you include it anyway by copying the file into the directory `config/cacerts/`.

User-provided certificates and trust chains

HPE Helion OpenStack generates its own internal certificates but is designed to allow you to bring in your own certificates for the VIPs. Here is the general process.

1. You must have a server certificate and a CA certificate to go with it (unless the signer is a public CA and it's already bundled with most distributions).
2. You must decide the names of the server certificates and configure the `network_groups.yml` file in the input model such that each load balancer provider has at least one cert-name associated with it.
3. Run the configuration processor. Note that you may or may not have the certificate file at this point. The configuration processor would create certificate request file artefacts under `info/cert_reqs/` for each of the cert-name(s) in the `network_groups.yml` file. While there's no special reason to use the request file created for an external endpoint VIP certificate, it is important to use the request files created for internal certificates since the canonical names for the internal VIP can be many and service specific and each of these need to be in the Subject Alt Names attribute of the certificate.
4. Create a certificate signing request for this request file and send it to your internal CA or a public CA to get it certified and issued with a certificate. You will now have a server certificate and possibly a trust chain or CA certificate.
5. Next, upload it to the lifecycle manager. Server certificates should be added to `config/tls/certs` and CA certificates should be added to `config/tls/cacerts`. The file extension should be `.crt` for the CA certificate to be processed by HPE Helion OpenStack. Detailed steps are next.

Edit the input model to include your certificate files

Edit the load balancer configuration in `helion/my_cloud/definition/data/network_groups.yml`:

```
load-balancers:
 - provider: ip-cluster
   name: lb
   tls-components:
   - default
   components:
   - vertica
   - nova-metadata
   roles:
   - internal
   - admin
   cert-file: example-internal-cert #<<<-------- Certificate name for the internal VIP

 - provider: ip-cluster
   name: extlb
   external-name: myhelion.test #<<<------ Use just IP for the external VIP in this example
   tls-components:
   - default
   roles:
   - public
   cert-file: example-public-cert #<<<-------- Certificate name for the external VIP
```

Commit your changes to the local git repository and run the configuration processor:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "changed VIP certificates"
ansible-playbook -i hosts/localhost config-processor-run.yml
```

Verify that certificate requests have been generated by the configuration processor for every certificate file configured in the `networks_groups.yml` file. In this example, there are two files, as shown from the list command:

```
ls ~/helion/my_cloud/info/cert_reqs
example-internal-cert
```

```
example-public-cert
```

Generate a self-signed CA

Note: In a production setting you will not perform this step. You will use your company's CA or a valid public CA.

This section demonstrates to how you can create your own self-signed CA and then use this CA to sign server certificates. This CA can be your organization's IT internal CA that is self-signed and whose CA certificates are deployed on your organization's machines. This way the server certificate becomes legitimate.

Note: Please use a unique CN for your example Certificate Authority and do not install multiple CA certificates with the same CN into your cloud.

Copy the commands below to the command line and execute. This will cause the two files, `example-CA.key` and `example-CA.crt` to be created:

```
export EXAMPLE_CA_KEY_FILE='example-CA.key'
export EXAMPLE_CA_CERT_FILE='example-CA.crt'
openssl req -x509 -batch -newkey rsa:2048 -nodes -out "${EXAMPLE_CA_CERT_FILE}" \
-keyout "${EXAMPLE_CA_KEY_FILE}" \
-subj "/C=UK/O=hp/CN=YourOwnUniqueCertAuthorityName" \
-days 365
```

You can tweak the subj and days settings above to meet your needs, or to test. For instance, if you want to test what happens when a CA expires, you can set 'days' to a very low value. Grab the configuration processor-generated request file from `info/cert_reqs/`:

```
cat ~/helion/my_cloud/info/cert_reqs/example-internal-cert
```

Now, copy this file to your working directory and appended a `.req` extension to it.

```
cp ~/helion/my_cloud/info/cert_reqs/example-internal-cert example-internal-cert.req
```

Expand the following headline to see a request file:

Certificate request file

```
[req]
distinguished_name = req_distinguished_name
req_extensions = v3_req
prompt = no

[ req_distinguished_name ]
CN = "helion-vip"

[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
subjectAltName = @alt_names

[ alt_names ]
DNS.1 = "deployerincloud-ccp-c0-m1-mgmt"
DNS.2 = "deployerincloud-ccp-vip-CEI-API-mgmt"
DNS.3 = "deployerincloud-ccp-vip-CND-API-mgmt"
DNS.4 = "deployerincloud-ccp-vip-DES-API-mgmt"
DNS.5 = "deployerincloud-ccp-vip-FND-MDB-mgmt"
DNS.6 = "deployerincloud-ccp-vip-FND-RMQ-mgmt"
DNS.7 = "deployerincloud-ccp-vip-FND-VDB-mgmt"
DNS.8 = "deployerincloud-ccp-vip-FRE-API-mgmt"
DNS.9 = "deployerincloud-ccp-vip-GLA-API-mgmt"
DNS.10 = "deployerincloud-ccp-vip-GLA-REG-mgmt"
DNS.11 = "deployerincloud-ccp-vip-HEA-ACF-mgmt"
DNS.12 = "deployerincloud-ccp-vip-HEA-ACW-mgmt"
```

```
DNS.13 = "deployerincloud-ccp-vip-HEA-API-mgmt"
DNS.14 = "deployerincloud-ccp-vip-HUX-SVC-mgmt"
DNS.15 = "deployerincloud-ccp-vip-HZN-WEB-mgmt"
DNS.16 = "deployerincloud-ccp-vip-KEY-API-mgmt"
DNS.17 = "deployerincloud-ccp-vip-KEYMGR-API-mgmt"
DNS.18 = "deployerincloud-ccp-vip-LOG-API-mgmt"
DNS.19 = "deployerincloud-ccp-vip-LOG-SVR-mgmt"
DNS.20 = "deployerincloud-ccp-vip-MON-API-mgmt"
DNS.21 = "deployerincloud-ccp-vip-NEU-SVR-mgmt"
DNS.22 = "deployerincloud-ccp-vip-NOV-API-mgmt"
DNS.23 = "deployerincloud-ccp-vip-NOV-MTD-mgmt"
DNS.24 = "deployerincloud-ccp-vip-OCT-API-mgmt"
DNS.25 = "deployerincloud-ccp-vip-OPS-WEB-mgmt"
DNS.26 = "deployerincloud-ccp-vip-SHP-API-mgmt"
DNS.27 = "deployerincloud-ccp-vip-SWF-PRX-mgmt"
DNS.28 = "deployerincloud-ccp-vip-admin-CEI-API-mgmt"
DNS.29 = "deployerincloud-ccp-vip-admin-CND-API-mgmt"
DNS.30 = "deployerincloud-ccp-vip-admin-DES-API-mgmt"
DNS.31 = "deployerincloud-ccp-vip-admin-FND-MDB-mgmt"
DNS.32 = "deployerincloud-ccp-vip-admin-FRE-API-mgmt"
DNS.33 = "deployerincloud-ccp-vip-admin-GLA-API-mgmt"
DNS.34 = "deployerincloud-ccp-vip-admin-HEA-ACF-mgmt"
DNS.35 = "deployerincloud-ccp-vip-admin-HEA-ACW-mgmt"
DNS.36 = "deployerincloud-ccp-vip-admin-HEA-API-mgmt"
DNS.37 = "deployerincloud-ccp-vip-admin-HUX-SVC-mgmt"
DNS.38 = "deployerincloud-ccp-vip-admin-HZN-WEB-mgmt"
DNS.39 = "deployerincloud-ccp-vip-admin-KEY-API-mgmt"
DNS.40 = "deployerincloud-ccp-vip-admin-KEYMGR-API-mgmt"
DNS.41 = "deployerincloud-ccp-vip-admin-MON-API-mgmt"
DNS.42 = "deployerincloud-ccp-vip-admin-NEU-SVR-mgmt"
DNS.43 = "deployerincloud-ccp-vip-admin-NOV-API-mgmt"
DNS.44 = "deployerincloud-ccp-vip-admin-OPS-WEB-mgmt"
DNS.45 = "deployerincloud-ccp-vip-admin-SHP-API-mgmt"
DNS.46 = "deployerincloud-ccp-vip-admin-SWF-PRX-mgmt"
DNS.47 = "192.168.245.5"
IP.1 = "192.168.245.5"

=============end of certificate request file.
```

Note: In the case of a public VIP certificate, please add all the FQDNs you want it to support Currently Helion does not add the hostname for the external-name specified in `network_groups.yml` to the certificate request file . However, you can add it to the certificate request file manually. Here we assume that `myhelion.test` is your external-name. In that case you would add this line (to the full *sample certificate request file* that is shown above):

```
DNS.48 = "myhelion.test"
```

Note: Any attempt to use IP addresses rather than FQDNs in certificates must use subject alternate name entries that list both the IP address (needed for Google) and DNS with an IP (needed for a Python bug workaround). Failure to create the certificates in this manner will cause future installations of Go-based tools (such as Cloud Foundry, Stackato and other PaaS components) to fail.

Generate a certificate signing request

Now that you have a CA and a certificate request file, it's time to generate a CSR.

```
export EXAMPLE_SERVER_KEY_FILE='example-internal-cert.key'
export EXAMPLE_SERVER_CSR_FILE='example-internal-cert.csr'
export EXAMPLE_SERVER_REQ_FILE=example-internal-cert.req
openssl req -newkey rsa:2048 -nodes -keyout "$EXAMPLE_SERVER_KEY_FILE" -out "$EXAMPLE_SERVER_CSR_FILE" -extensions v3_req -config "$EXAMPLE_SERVER_REQ_FILE"
```

Note that in production you would usually send the generated `example-internal-cert.csr` file to your IT department. But in this example you are your own CA, so sign and generate a server certificate.

Generate a server certificate

📝 **Note:** In a production setting you will not perform this step. You will send the CSR created in the previous section to your company CA or a to a valid public CA and have them sign and send you back the certificate.

This section demonstrates how you would use your own self-signed CA that your created earlier to sign and generate a server certificate. A server certificate is essentially a signed public key, the signer being a CA and trusted by a client. When you install this the signing CA's certificate (called CA certificate or trust chain) on the client machine, you are telling the client to trust this CA, and thereby implicitly trusting any server certificates that are signed by this CA, thus creating a trust anchor.

CA configuration file

When the CA signs the certificate, it uses a configuration file that tells it to verify the CSR. Note that in a production scenario the CA takes care of this for you.

Create a file called `openssl.cnf` and add the following contents to it.

```
# Copyright 2010 United States Government as represented by the
# Administrator of the National Aeronautics and Space Administration.
# All Rights Reserved.
#...

# OpenSSL configuration file.
#

# Establish working directory.

dir = .

[ ca ]
default_ca = CA_default

[ CA_default ]
serial = $dir/serial
database = $dir/index.txt
new_certs_dir = $dir/
certificate = $dir/cacert.pem
private_key = $dir/cakey.pem
unique_subject = no
default_crl_days = 365
default_days = 365
default_md = md5
preserve = no
email_in_dn = no
nameopt = default_ca
certopt = default_ca
policy = policy_match
copy_extensions = copy

# NOTE(dprince): stateOrProvinceName must be 'supplied' or 'optional' to
# work around a stateOrProvince printable string UTF8 mismatch on
# RHEL 6 and Fedora 14 (using openssl-1.0.0-4.el6.x86_64 or
# openssl-1.0.0d-1.fc14.x86_64)
[ policy_match ]
countryName = optional
stateOrProvinceName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional

[ req ]
default_bits = 1024 # Size of keys
default_keyfile = key.pem # name of generated keys
default_md = md5 # message digest algorithm
string_mask = nombstr # permitted characters
distinguished_name = req_distinguished_name
```

```
req_extensions = v3_req
x509_extensions = v3_ca

[ req_distinguished_name ]
# Variable name Prompt string
#-------------------- ----------------------------------
0.organizationName = Organization Name (company)
organizationalUnitName = Organizational Unit Name (department, division)
emailAddress = Email Address
emailAddress_max = 40
localityName = Locality Name (city, district)
stateOrProvinceName = State or Province Name (full name)
countryName = Country Name (2 letter code)
countryName_min = 2
countryName_max = 2
commonName = Common Name (hostname, IP, or your name)
commonName_max = 64

# Default values for the above, for consistency and less typing.
# Variable name Value
#-------------------------------- -------------------------------
0.organizationName_default = Hewlett-Packard-Enterprise
localityName_default = Bristol
stateOrProvinceName_default = Bristol
countryName_default = UK

[ v3_ca ]
basicConstraints = CA:TRUE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer:always
subjectAltName = @alt_names

[ v3_req ]
basicConstraints = CA:FALSE
subjectKeyIdentifier = hash

[ alt_names ]

######### end of openssl.cnf #########
```

Sign and create a server certificate

Now you can sign the server certificate with your CA. Copy the comands below to the command line and execute. This will cause the one file, example-internal-cert.crt, to be created:

```
export EXAMPLE_SERVER_CERT_FILE='example-internal-cert.crt'
export EXAMPLE_SERVER_CSR_FILE='example-internal-cert.csr'
export EXAMPLE_CA_KEY_FILE='example-CA.key'
export EXAMPLE_CA_CERT_FILE='example-CA.crt'

touch index.txt
openssl rand -hex -out serial 6

openssl ca -batch -notext -md sha256 -in "$EXAMPLE_SERVER_CSR_FILE" \
-cert "$EXAMPLE_CA_CERT_FILE" \
-keyfile "$EXAMPLE_CA_KEY_FILE" \
-out "$EXAMPLE_SERVER_CERT_FILE" \
-config openssl.cnf -extensions v3_req
```

Finally, concatenate both the server key and certificate in preparation for uploading to the lifecycle manager.

```
cat example-internal-cert.key example-internal-cert.crt > example-internal-cert
```

Note that you have only created the internal-cert in this example. Repeat the above sequence for example-public-cert. Make sure you use the appropriate certificate request generated by the configuration processor.

Upload to the lifecycle manager

The following two files created from the example run above will need to be uploaded to the lifecycle manager and copied into config/tls:

• example-internal-cert
• example-CA.crt

Once on the lifecycle manager, execute the following two copy commands to copy to their respective directories. Note if you had created an external cert, you can copy that in a similar manner, specifying its name using the copy command as well.

```
cp example-internal-cert ~/helion/my_cloud/config/tls/certs/
cp example-CA.crt ~/helion/my_cloud/config/tls/cacerts/
```

Continue with the deployment

Next, log into the lifecycle manager node, and save and commit the changes to the local git repository:

```
cd ~/helion/hos/ansible
git add -A
git commit -m "updated certificate and CA"
```

Next, rerun the config-processor-run playbook, and run ready-deployment.yml:

```
cd ~/helion/hos/ansible
ansible-playbook -i hosts/localhost config-processor-run.yml
ansible-playbook -i hosts/localhost ready-deployment.yml
```

If you receive any prompts, enter the required information.

> **Note:** For automated installation (e.g. CI) you can specify the required passwords on the Ansible command line. For example, the command below will disable encryption by the configuration processor:
>
> ```
> ansible-playbook -i hosts/localhost config-processor-run.yml -e encrypt="" -e rekey=""
> ```

Run this series of runbooks to complete the deployment:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts tls-reconfigure.yml
ansible-playbook -i hosts/verb_hosts FND-CLU-stop.yml
ansible-playbook -i hosts/verb_hosts FND-CLU-start.yml
ansible-playbook -i hosts/verb_hosts monasca-stop.yml
ansible-playbook -i hosts/verb_hosts monasca-start.yml
ansible-playbook -i hosts/verb_hosts hlm-reconfigure.yml
```

Configuring the cipher suite

By default, the cipher suite is set to: HIGH:!aNULL:!eNULL:!DES:!3DES. This setting is recommended in the *OpenStack documentation site*. You may override this. To do so, open config/haproxy/defaults.yml and edit it. The parameters can be found under the haproxy_globals list.

```
- "ssl-default-bind-ciphers HIGH:!aNULL:!eNULL:!DES:!3DES"
- "ssl-default-server-ciphers HIGH:!aNULL:!eNULL:!DES:!3DES"
```

Make the changes as needed. It's best to keep the two options identical.

Testing
You can easily determine if an endpoint is behind TLS. To do so, run the following command, which probes a Keystone identity service endpoint that's behind TLS:

```
echo | openssl s_client -connect 192.168.245.5:5000 | openssl x509 -fingerprint -noout
        depth=0 CN = helion-vip
```

```
                     verify error:num=20:unable to get local issuer certificate
                     verify return:1
                     depth=0 CN = helion-vip
                     verify error:num=27:certificate not trusted
                     verify return:1
                     depth=0 CN = helion-vip
                     verify error:num=21:unable to verify the first certificate
                     verify return:1
                     DONE
                     SHA1 Fingerprint=C6:46:1E:59:C6:11:BF:72:5E:DD:FC:FF:B0:66:A7:A2:CC:32:1C:B8
```

The next command probes a MySQL endpoint that is not behind TLS:

```
echo | openssl s_client -connect 192.168.245.5:3306 | openssl x509 -fingerprint -noout
140448358213264:error:140770FC:SSL routines:SSL23_GET_SERVER_HELLO:unknown protocol:s23_clnt.c:795:
unable to load certificate
140454148159120:error:0906D06C:PEM routines:PEM_read_bio:no start line:pem_lib.c:703:Expecting: TRUSTED CERTIFICATE
```

Verifying that the trust chain is correctly deployed
You can determine if the trust chain is correctly deployed by running the following commands:

```
echo | openssl s_client -connect 192.168.245.9:5000 2>/dev/null | grep code
Verify return code: 21 (unable to verify the first certificate)
echo | openssl s_client -connect 192.168.245.9:5000 -CAfile /usr/local/share/ca-certificates/helion_frontend_cacert.crt 2>/dev/null | grep code
Verify return code: 0 (ok)
```

Here, the first command produces error 21, which is then fixed by providing the CA certificate file. This verifies that the CA certificate matches the server certificate.

Turning TLS on or off
You should leave TLS enabled in production. However, if you need to disable it for any reason, you must change "tls-components" to "components" in `network_groups.yml` (as shown earlier) and comment out the cert-file. Additionally, if you have a `network_groups.yml` file from a previous installation, you won't have TLS enabled unless you change "components" to "tls-components" in that file. By default, Horizon is configured with TLS in the input model. Note that you should not disable TLS in the input model for Horizon as that is a public endpoint and is required. Additionally, you should keep all services behind TLS, but using the input model file `network_groups.yml` you may turn TLS off for a service for troubleshooting or debugging. TLS should always be enabled for production environments.

If you are using an example input model on a clean install, all supported TLS services will be enabled before deployment of your cloud. If you want to change this setting later, for example, when upgrading, you can change the input model and reconfigure the system. The process is as follows:

Edit the input model `network_groups.yml` file appropriately, as described above. Then, commit the changes to the git repository:

```
cd ~/helion/hos/ansible/
git add -A
git commit -m "TLS change"
```

Change directories again and run the configuration processor and ready deployment playbooks:

```
ansible-playbook -i hosts/localhost config-processor-run.yml
ansible-playbook -i hosts/localhost ready-deployment.yml
```

Change directories again and run the reconfigure playbook:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts hlm-reconfigure.yml
```

HPE Helion OpenStack® 5.0: Configuring Availability Zones
Availability Zones are not configured by default during installation. Use `nova-cloud-configure.yml` to configure them post-install.

The lifecycle manager only creates a default availability zone during installation. If your system has multiple failure/availability zones defined in your input model, these zones will not get created automatically.

Once the installation has finished, you can run the `nova-cloud-configure.yml` playbook to configure availability zones and assign compute nodes to those zones based on the configuration specified in the model.

You can run the playbook `nova-cloud-configure.yml` any time you make changes to the configuration of availability zones in your input model. Alternatively, you can use Horizon or the command line to perform the configuration.

For more details, see *HPE Helion OpenStack 5.0: Managing Compute Hosts using Aggregates and Scheduler Filters*.

HPE Helion OpenStack® 5.0: Configuring Load Balancer as a Service

The HPE Helion OpenStack Neutron LBaaS service supports several load balancing providers. By default, both Octavia and the namespace haproxy driver are configured to be used. We describe this in more detail here.

⚠️ **Warning:** If you are planning to upgrade from HPE Helion OpenStack 3.0 or HPE Helion OpenStack 4.0, please contact F5 and HPE PointNext to determine which F5 drivers have been certified for use with Helion OpenStack. Loading drivers not certified by HPE may result in catastrophic failure of your cloud deployment. The last tested versions are 8.0.8 for HPE Helion OpenStack 3.x and 9.0.3 for HPE Helion OpenStack 4.x . More information is expected in 4th quarter 2017, including the correct drivers for HPE Helion OpenStack 5.x.

The HPE Helion OpenStack Neutron LBaaS service supports several load balancing providers. By default, both Octavia and the namespace haproxy driver are configured to be used. A user can specify which provider to use with the `--provider` flag upon load balancer creation.

Example:

```
neutron lbaas-loadbalancer-create --name <name> --provider [octavia|haproxy] <subnet>
```

If you don't specify the `--provider` option it will default to Octavia. The Octavia driver provides more functionality than the haproxy namespace driver which is deprecated. The haproxy namespace driver will be retired in a future version of HPE Helion OpenStack.

There are additional drivers for 3rd party hardware load balancers. Please refer to the vendor directly. The `neutron service-provider-list` command displays not only the currently installed load balancer drivers but also other installed services such as VPN. You can see a list of available services as follows:

```
$ neutron service-provider-list
+---------------+----------+---------+
| service_type  | name     | default |
+---------------+----------+---------+
| LOADBALANCERV2 | octavia  | True    |
| VPN           | openswan | True    |
| LOADBALANCERV2 | haproxy  | False   |
| LOADBALANCERV2 | octavia  | True    |
| VPN           | openswan | True    |
| LOADBALANCERV2 | haproxy  | False   |
+---------------+----------+---------+
```

📝 **Note:** In the example above, the providers are listed twice. This is a limitation in HPE Helion OpenStack 3.0. Also note that the Octavia load balancer provider is listed as the default.

Prerequisites

You will need to create an external network and create an image to test LBaaS functionality. If you have already created an external network and registered an image, this step can be skipped.

Creating an external network

HPE Helion OpenStack 5.0: Creating an External Network

Creating and uploading a Glance image

HPE Helion OpenStack 5.0: Creating and Uploading a Glance Image

Octavia Load Balancing Provider

The Octavia Load balancing provider bundled with HPE Helion OpenStack 5.0 is an operator grade load balancer for OpenStack. It is based on the Mitaka version of Octavia. It differs from the namespace driver by starting a new nova virtual machine to house the haproxy load balancer software, called an *amphora*, that provides the load balancer function. A virtual machine for each load balancer requested provides a better separation of load balancers between tenants and makes it easier to grow load balancing capacity alongside compute node growth. Additionally, if the virtual machine fails for any reason Octavia will replace it with a replacement VM from a pool of spare VM's, assuming that the feature is configured.

📝 **Note:** The Health Monitor will not create or replace failed amphorae. If the pool of spare VM's is exhausted there will be no additional virtual machines to handle load balancing requests.

Octavia uses two-way SSL encryption to communicate with the amphora. There are demo Certificate Authority (CA) certificates included with HPE Helion OpenStack 5.0 in `/home/stack/scratch/ansible/next/hos/ansible/roles/octavia-common/files` on the lifecycle manager. For additional security in production deployments, all certificate authorities should be replaced with ones you generated yourself by running the following commands:

```
openssl genrsa -passout pass:foobar -des3 -out cakey.pem 2048
openssl req -x509 -passin pass:foobar -new -nodes -key cakey.pem -out ca_01.pem
openssl genrsa -passout pass:foobar -des3 -out servercakey.pem 2048
openssl req -x509 -passin pass:foobar -new -nodes -key cakey.pem -out serverca_01.pem
```

For more details refer to the *openssl man page*.

> **Note:** If you change the certificate authority and have amphora running with an old CA you won't be able to control the amphora. The amphora's will need to be failed over so they can utilize the new certificate. If you change the CA password for the server certificate you need to change that in the Octavia configuration files as well. See the *Octavia Administration* guide for more information.

Setup of prerequisites

Octavia Network and Management Network Ports

The Octavia management network and Management network must have access to each other. If you have a configured firewall between the Octavia management network and Management network, you must open up the following ports to allow network traffic between the networks.

- From Management network to Octavia network
  - TCP 9443 (amphora API)
- From Octavia network to Management network
  - TCP 9876 (Octavia API)
  - UDP 5555 (Octavia Health Manager)

Installing the Amphora Image

Octavia is utilizing Nova VMs for its load balancing function and HPE provides images used to boot those VM's called `octavia-amphora-haproxy`.

> **Warning:** Without these images the Octavia load balancer will not work.

You can download the **Octavia Amphora HA Proxy Guest Image** from HPE Software Entitlement Portal.

- Sign in and download the product and signature files from the following link: *Software Entitlement Portal*
- Once the image is downloaded it needs to be placed on the lifecycle manager node and the image registered.

Register the image.

1. Switch to the ansible directory and register the image by giving the full path and name (e.g. `/tmp/octavia-amphora-haproxy-guest-image.tgz`) as argument to service_package:

```
$ cd ~/scratch/ansible/next/hos/ansible/
$ ansible-playbook -i hosts/verb_hosts -e service_package=<image path/name> service-guest-image.yml
```

2. Source the service user (this can be done on a different computer)

```
$ source service.osrc
```

3. Verify that the image was uploaded and registered (this can be done on a computer with access to the glance CLI client)

```
$ glance image-list
+--------------------------------------+-------------------------------------+
| ID                                   | Name                                |
+--------------------------------------+-------------------------------------+
| 01ff1f0d-fc35-4e3e-bae2-e7e2ee1f65b6 | cirros-0.3.3-x86_64                 |
| e64cb914-15d2-4ad8-a63c-b7c60a6c232e | octavia-amphora-x64-haproxy_hos-3.0.0 |
+--------------------------------------+-------------------------------------+
```

> **Note:** In rare circumstances the image won't be registered and you will have to run the whole registration again. If you run the registration by accident, the system will only upload a new image if the underlying image has been changed.

4. Check the status of the image by running `glance image-show <image-id>`.

```
$ glance image-show e64cb914-15d2-4ad8-a63c-b7c60a6c232e
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | 094a553d38fb5bfdb43f4a662d84ec2e     |
| container_format | bare                                 |
| created_at       | 2016-04-14T23:05:21Z                 |
| disk_format      | qcow2                                |
| id               | e64cb914-15d2-4ad8-a63c-b7c60a6c232e |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | octavia-amphora-x64-haproxy_hos-3.0.0 |
| owner            | 0671a8d4d71c44ffb210c11cb5d11f7f     |
| protected        | False                                |
| size             | 434379264                            |
| status           | active                               |
| tags             | []                                   |
| updated_at       | 2016-04-14T23:05:36Z                 |
| virtual_size     | None                                 |
| visibility       | private                              |
+------------------+--------------------------------------+
```

> **Important:** In the example above, the status of the image is *active* which means the image was successfully registered. If a status of the images is *queued*, you must run the image registration again.

Please be aware that if you have already created load balancers they won't receive the new image. Only load balancers created after the image has been successfully installed will use the new image. If existing load balancers need to be switched to the new image please follow the instructions in the *Octavia Administration* guide.

Setup network, subnet, router, security and IP's

If you have already created a network, subnet, router, security settings and IP's you can skip the following steps and go directly to creating the load balancers.

1. Create a network.

```
neutron net-create lb_net1
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 71a1ac88-30a3-48a3-a18b-d98509fbef5c |
| mtu                       | 0                                    |
| name                      | lb_net1                              |
| provider:network_type     | vxlan                                |
| provider:physical_network |                                      |
| provider:segmentation_id  | 1061                                 |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 4b31d0508f83437e83d8f4d520cda22f     |
+---------------------------+--------------------------------------+
```

2. Create a subnet.

```
neutron subnet-create --name lb_subnet1 lb_net1 10.247.94.128/26 --gateway 10.247.94.129
+-------------------+------------------------------------------------------+
| Field             | Value                                                |
+-------------------+------------------------------------------------------+
```

```
| allocation_pools  | {"start": "10.247.94.130", "end": "10.247.94.190"} |
| cidr              | 10.247.94.128/26                                   |
| dns_nameservers   |                                                    |
| enable_dhcp       | True                                               |
| gateway_ip        | 10.247.94.129                                      |
| host_routes       |                                                    |
| id                | 6fc2572c-53b3-41d0-ab63-342d9515f514               |
| ip_version        | 4                                                  |
| ipv6_address_mode |                                                    |
| ipv6_ra_mode      |                                                    |
| name              | lb_subnet1                                         |
| network_id        | 71a1ac88-30a3-48a3-a18b-d98509fbef5c               |
| subnetpool_id     |                                                    |
| tenant_id         | 4b31d0508f83437e83d8f4d520cda22f                   |
+-------------------+----------------------------------------------------+
```

3. Create a router.

```
neutron router-create --distributed False lb_router1
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| distributed           | False                                |
| external_gateway_info |                                      |
| ha                    | False                                |
| id                    | 6aafc9a9-93f6-4d7e-94f2-3068b034b823 |
| name                  | lb_router1                           |
| routes                |                                      |
| status                | ACTIVE                               |
| tenant_id             | 4b31d0508f83437e83d8f4d520cda22f     |
+-----------------------+--------------------------------------+
```

4. Add interface to router. In this example, `neutron router-interface-add lb_router1 lb_subnet1` will add interface 426c5898-f851-4f49-b01f-7a6fe490410c to router lb_router1.

```
neutron router-interface-add lb_router1 lb_subnet1
```

5. Set gateway for router.

```
neutron router-gateway-set lb_router1 ext-net
```

6. Check networks.

```
 neutron net-list
+-----------------------------+-----------------+-----------------------------+
| id                          | name            | subnets                     |
+-----------------------------+-----------------+-----------------------------+
| d3cb12a6-a000-4e3e-         | ext-net         | f4152001-2500-4ebe-ba9d-    |
| 82c4-ee04aa169291           |                 | a8d6149a50df 10.247.96.0/23 |
| 8306282a-3627-445a-a588-c18 | OCTAVIA-MGMT-NET | f00299f8-3403-45ae-ac4b-    |
| 8b6a13163                   |                 | 58af41d57bdc                |
|                             |                 | 10.247.94.128/26            |
| 71a1ac88-30a3-48a3-a18b-    | lb_net1         | 6fc2572c-                   |
| d98509fbef5c                |                 | 53b3-41d0-ab63-342d9515f514 |
|                             |                 | 10.247.94.128/26            |
+-----------------------------+-----------------+-----------------------------+
```

7. Create security group.

```
neutron security-group-create lb_secgroup1
+-----------------------+---------------------------------------------------------------+
| Field                 | Value                                                         |
+-----------------------+---------------------------------------------------------------+
```

```
| description        |                                                                                                     |
| id                 | 75343a54-83c3-464c-8773-802598afaee9                                                                |
| name               | lb_secgroup1                                                                                        |
| security_group_rules | {"remote_group_id": null, "direction": "egress", "remote_ip_prefix": null, "protocol": null, "tenant_id": |
|                    | "4b31d0508f83437e83d8f4d520cda22f", "port_range_max": null, "security_group_id": "75343a54-83c3-464c-8773-802598afaee9", |
|                    | "port_range_min": null, "ethertype": "IPv4", "id": "20ae3723-050c-44ed-a8f7-c5da6fa97a7a"}          |
|                    | {"remote_group_id": null, "direction": "egress", "remote_ip_prefix": null, "protocol": null, "tenant_id": |
|                    | "4b31d0508f83437e83d8f4d520cda22f", "port_range_max": null, "security_group_id": "75343a54-83c3-464c-8773-802598afaee9", |
|                    | "port_range_min": null, "ethertype": "IPv6", "id": "563c5ca3-3bbd-4c96-9324-1ca9bc3aaef9"}          |
| tenant_id          | 4b31d0508f83437e83d8f4d520cda22f                                                                    |
+--------------------+-----------------------------------------------------------------------------------------------------+
```

8. Create icmp security group rule.

```
neutron security-group-rule-create lb_secgroup1 --protocol icmp
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 16d74150-a5b2-4cf6-82eb-a6c49a972d93 |
| port_range_max    |                                      |
| port_range_min    |                                      |
| protocol          | icmp                                 |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | 75343a54-83c3-464c-8773-802598afaee9 |
| tenant_id         | 4b31d0508f83437e83d8f4d520cda22f     |
+-------------------+--------------------------------------+
```

9. Create TCP port 22 rule.

```
neutron security-group-rule-create lb_secgroup1 --protocol tcp --port-range-min 22 --port-range-max 22
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 472d3c8f-c50f-4ad2-97a1-148778e73af5 |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | 75343a54-83c3-464c-8773-802598afaee9 |
| tenant_id         | 4b31d0508f83437e83d8f4d520cda22f     |
+-------------------+--------------------------------------+
```

10. Create TCP port 80 rule.

```
neutron security-group-rule-create lb_secgroup1 --protocol tcp --port-range-min 80 --port-range-max 80
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 10a76cad-8b1c-46f6-90e8-5dddd279e5f7 |
| port_range_max    | 80                                   |
| port_range_min    | 80                                   |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | 75343a54-83c3-464c-8773-802598afaee9 |
```

```
| tenant_id        | 4b31d0508f83437e83d8f4d520cda22f    |
+------------------+-------------------------------------+
```

11. If you haven't already created a keypair, create one now with `nova keypair-add`. You'll use the keypair to boot images.

```
nova keypair-add lb_kp1 > lb_kp1.pem

chmod 400 lb_kp1.pem

cat lb_kp1.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEqAIBAAKCAQEAkbW5W/XWGRGC0LAJI7lttR7EdDfiTDeFJ7A9b9Cff+OMXjhx
WL26eKIr+jp8DR64YjV2mNnQLsDyCxekFpkyjnGRId3KVAeV5sRQqXgtaCXI+Rvd
IyUtd8p1cp3DRgTd1dxO0oL6bBmwrZatNrrRn4HgKc2c7ErekeXrwLHyE0Pia/pz
C6qs0coRdfIeXxsmS3kXExP0YfsswRS/OyDl8QhRAF0ZW/zV+DQIi8+HpLZT+RW1
8sTTYZ6b0kXoH9wLER4IUBj1I1IyrYdxlAhe2VIn+tF0Ec4nDBn1py9iwEfGmn0+
N2jHCJAkrK/QhWdXO4O8zeXfL4mCZ9FybW4nzQIDAQABAoIBACe0PvgB+v8FuIGp
FjR32J8b7ShF+hIOpufzrCoFzRCKLruV4bzuphstBZK/0QG6Nz/7lX99Cq9SwCGp
pXrK7+3EoGl8CB/xmTUylVA4gRb6BNNsdkuXW9ZigrJirs0rkk8uIwRV0GsYbP5A
Kp7ZNTmjqDN75aC1ngRfhGgTlQUOdxBH+4xSb7GukekD13V8V5MF1Qft089asdWp
l/TpvhYeW9O92xEnZ3qXQYpXYQgEFQoM2PKa3VW7FGLgfw9gdS/MSqpHuHGyKmjl
uT6upUX+Lofbe7V+9kfxuV32sLL/S5YFvkBy2q8VpuEV1sXI7O7Sc411WX4cqmlb
YoFwhrkCggCBALkYE7OMTtdCAGcMotJhTiiS5l8d4U/fn1x0zus43XV5Y7wCnMuU
r5vCoK+a+TR9Ekzc/GjccAx7Wz/YYKp6G8FXW114dLcADXZjqjI1X7ifUud4sLCS
y+x3KAJa7LqyzH53I6FOts9RaB5xx4gZ2WjcJquCTbATZWj7j1yGeNgvAoIAgQDJ
h0r0Te5I1iYbCRg+ES9YRZzH/PSLuIn00bbLvpOPNEoKe2Pxs+KI8Fqp6ZIDAB3c
4EPOK5QrJvAny9Z58ZArrNZi15t84KEVAkWUATl+c4SmHc8sW/atgmUoqIzgDQXe
AlwadHLY7JCdg7EYDuUxuTKLLOdqfpf6fKkhNxtEwwKCAIAMxi+d5aIPUxvKAOI/
2L1XKYRCrkI9i/ZooBsjusH1+JG8iQWfOzy/aDhExlJKoBMiQOIerpABHIZYqqtJ
OLIvrsK8ebK8aoGDWS+G1HN9v2kuVnMDTK5MPJEDUJkj7XEVjU1lNZSCTGD+MOYP
a5FInmEA1zZbX4tRKoNjZFh0uwKCAIEAiLs7drAdOLBu4C72fL4KI1jwu5t7jATD
zRAwduIxmZq/lYcMU2RaEdEJonivsUt193NNbeeRWwnLLSUWupvT114pAt0ISNzb
TbbB4F5IVOwpls9ozc8DecubuM9K7YTIc02kkepqNZWjtMsx74HDrU3a5iSsSkvj
73Z/BeMupCMCggCAS48BsrcsDsHSHE3tO4D8pAIr1r+6WPaQn49pT3GIrdQNc7aO
d9PfXmPoe/PxUlqaXoNAvT99+nNEadp+GTId21VM0Y28pn3EkIGE1Cqoey13BEO8
f9SUiRNruDnH4F4OclsDBlmqWXImuXRfeiDHxM8X03UDZoqyHmGD3RqA53I=
-----END RSA PRIVATE KEY-----
```

12. Check and boot images.

```
nova image-list
+--------------------------------------+---------------------------------------+--------+--------+
| ID                                   | Name                                  | Status | Server |
+--------------------------------------+---------------------------------------+--------+--------+
| 04b1528b-b1e2-45d4-96d1-fbe04c6b2efd | cirros-0.3.3-x86_64                   | ACTIVE |        |
| 8aa51e01-e6f4-480f-b91e-08ea14178f2f | octavia-amphora-x64-haproxy_hos-3.0.0 | ACTIVE |        |
+--------------------------------------+---------------------------------------+--------+--------+
```

Boot first VM.

```
nova boot --flavor 1 --image 04b1528b-b1e2-45d4-96d1-fbe04c6b2efd --key-name lb_kp1 --security-groups lb_secgroup1 --nic net-id=71a1ac88-30a3-48a3-a18b-d98509fbef5c lb_vm1 --
poll
+--------------------------------------+------------------------------------------------------------+
| Property                             | Value                                                      |
+--------------------------------------+------------------------------------------------------------+
| OS-DCF:diskConfig                    | MANUAL                                                      |
| OS-EXT-AZ:availability_zone          |                                                            |
| OS-EXT-SRV-ATTR:host                 | -                                                          |
| OS-EXT-SRV-ATTR:hypervisor_hostname  | -                                                          |
| OS-EXT-SRV-ATTR:instance_name        | instance-00000031                                          |
| OS-EXT-STS:power_state               | 0                                                          |
| OS-EXT-STS:task_state                | scheduling                                                 |
| OS-EXT-STS:vm_state                  | building                                                   |
```

```
| OS-SRV-USG:launched_at           | -                                                       |
| OS-SRV-USG:terminated_at         | -                                                       |
| accessIPv4                       |                                                         |
| accessIPv6                       |                                                         |
| adminPass                        | NeVvhP5E8iCy                                            |
| config_drive                     |                                                         |
| created                          | 2016-06-15T16:53:00Z                                    |
| flavor                           | m1.tiny (1)                                             |
| hostId                           |                                                         |
| id                               | dfdfe15b-ce8d-469c-a9d8-2cea0e7ca287                    |
| image                            | cirros-0.3.3-x86_64 (04b1528b-b1e2-45d4-96d1-fbe04c6b2efd) |
| key_name                         | lb_kp1                                                  |
| metadata                         | {}                                                      |
| name                             | lb_vm1                                                  |
| os-extended-volumes:volumes_attached | []                                                  |
| progress                         | 0                                                       |
| security_groups                  | lb_secgroup1                                            |
| status                           | BUILD                                                   |
| tenant_id                        | 4b31d0508f83437e83d8f4d520cda22f                        |
| updated                          | 2016-06-15T16:53:00Z                                    |
| user_id                          | fd471475faa84680b97f18e55847ec0a                        |
+----------------------------------+---------------------------------------------------------+

           Server building... 100% complete
           Finished
```

Boot second VM.

```
nova boot --flavor 1 --image 04b1528b-b1e2-45d4-96d1-fbe04c6b2efd --key-name lb_kp1 --security-groups lb_secgroup1 --nic net-id=71a1ac88-30a3-48a3-a18b-d98509fbef5c lb_vm2 --
poll
+----------------------------------+---------------------------------------------------------+
| Property                         | Value                                                   |
+----------------------------------+---------------------------------------------------------+
| OS-DCF:diskConfig                | MANUAL                                                  |
| OS-EXT-AZ:availability_zone      |                                                         |
| OS-EXT-SRV-ATTR:host             | -                                                       |
| OS-EXT-SRV-ATTR:hypervisor_hostname | -                                                    |
| OS-EXT-SRV-ATTR:instance_name    | instance-00000034                                       |
| OS-EXT-STS:power_state           | 0                                                       |
| OS-EXT-STS:task_state            | scheduling                                              |
| OS-EXT-STS:vm_state              | building                                                |
| OS-SRV-USG:launched_at           | -                                                       |
| OS-SRV-USG:terminated_at         | -                                                       |
| accessIPv4                       |                                                         |
| accessIPv6                       |                                                         |
| adminPass                        | 3nFXjNrTrmNm                                            |
| config_drive                     |                                                         |
| created                          | 2016-06-15T16:55:10Z                                    |
| flavor                           | m1.tiny (1)                                             |
| hostId                           |                                                         |
| id                               | 3844bb10-2c61-4327-a0d4-0c043c674344                    |
| image                            | cirros-0.3.3-x86_64 (04b1528b-b1e2-45d4-96d1-fbe04c6b2efd) |
| key_name                         | lb_kp1                                                  |
| metadata                         | {}                                                      |
| name                             | lb_vm2                                                  |
| os-extended-volumes:volumes_attached | []                                                  |
| progress                         | 0                                                       |
| security_groups                  | lb_secgroup1                                            |
| status                           | BUILD                                                   |
| tenant_id                        | 4b31d0508f83437e83d8f4d520cda22f                        |
| updated                          | 2016-06-15T16:55:09Z                                    |
| user_id                          | fd471475faa84680b97f18e55847ec0a                        |
+----------------------------------+---------------------------------------------------------+
```

```
                Server building... 100% complete
                Finished
```

13. List the running VM with `nova list`

```
nova list
+--------------------------------------+--------+--------+------------+-------------+----------------------+
| ID                                   | Name   | Status | Task State | Power State | Networks             |
+--------------------------------------+--------+--------+------------+-------------+----------------------+
| dfdfe15b-ce8d-469c-a9d8-2cea0e7ca287 | lb_vm1 | ACTIVE | -          | Running     | lb_net1=10.247.94.132 |
| 3844bb10-2c61-4327-a0d4-0c043c674344 | lb_vm2 | ACTIVE | -          | Running     | lb_net1=10.247.94.133 |
+--------------------------------------+--------+--------+------------+-------------+----------------------+
```

14. Check ports.

```
neutron port-list
+--------------------------------------+------+-------------------+----------------------------------------------------------+
| id                                   | name | mac_address       | fixed_ips                                                |
+--------------------------------------+------+-------------------+----------------------------------------------------------+
...
| 7e5e0038-88cf-4f97-a366-b58cd836450e |      | fa:16:3e:66:fd:2e | {"subnet_id": "6fc2572c-53b3-41d0-ab63-342d9515f514",    |
|                                      |      |                   | "ip_address": "10.247.94.132"}                           |
| ca95cc24-4e8f-4415-9156-7b519eb36854 |      | fa:16:3e:e0:37:c4 | {"subnet_id": "6fc2572c-53b3-41d0-ab63-342d9515f514",    |
|                                      |      |                   | "ip_address": "10.247.94.133"}                           |
+--------------------------------------+------+-------------------+----------------------------------------------------------+
```

15. Create the first floating IP.

```
neutron floatingip-create ext-net --port-id 7e5e0038-88cf-4f97-a366-b58cd836450e
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    | 10.247.94.132                        |
| floating_ip_address | 10.247.96.26                         |
| floating_network_id | d3cb12a6-a000-4e3e-82c4-ee04aa169291 |
| id                  | 3ce608bf-8835-4638-871d-0efe8ebf55ef |
| port_id             | 7e5e0038-88cf-4f97-a366-b58cd836450e |
| router_id           | 6aafc9a9-93f6-4d7e-94f2-3068b034b823 |
| status              | DOWN                                 |
| tenant_id           | 4b31d0508f83437e83d8f4d520cda22f     |
+---------------------+--------------------------------------+
```

16. Create the second floating IP.

```
neutron floatingip-create ext-net --port-id ca95cc24-4e8f-4415-9156-7b519eb36854
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    | 10.247.94.133                        |
| floating_ip_address | 10.247.96.27                         |
| floating_network_id | d3cb12a6-a000-4e3e-82c4-ee04aa169291 |
| id                  | 680c0375-a179-47cb-a8c5-02b836247444 |
| port_id             | ca95cc24-4e8f-4415-9156-7b519eb36854 |
| router_id           | 6aafc9a9-93f6-4d7e-94f2-3068b034b823 |
| status              | DOWN                                 |
| tenant_id           | 4b31d0508f83437e83d8f4d520cda22f     |
+---------------------+--------------------------------------+
```

17. List the floating IP's.

```
neutron floatingip-list
+--------------------------------------+------------------+---------------------+--------------------------------------+
```

```
| id                                   | fixed_ip_address | floating_ip_address | port_id                              |
+--------------------------------------+------------------+---------------------+--------------------------------------+
| 3ce608bf-8835-4638-871d-0efe8ebf55ef | 10.247.94.132    | 10.247.96.26        | 7e5e0038-88cf-4f97-a366-b58cd836450e |
| 680c0375-a179-47cb-a8c5-02b836247444 | 10.247.94.133    | 10.247.96.27        | ca95cc24-4e8f-4415-9156-7b519eb36854 |
+--------------------------------------+------------------+---------------------+--------------------------------------+
```

18. Show first Floating IP.

```
neutron floatingip-show 3ce608bf-8835-4638-871d-0efe8ebf55ef
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    | 10.247.94.132                        |
| floating_ip_address | 10.247.96.26                         |
| floating_network_id | d3cb12a6-a000-4e3e-82c4-ee04aa169291 |
| id                  | 3ce608bf-8835-4638-871d-0efe8ebf55ef |
| port_id             | 7e5e0038-88cf-4f97-a366-b58cd836450e |
| router_id           | 6aafc9a9-93f6-4d7e-94f2-3068b034b823 |
| status              | ACTIVE                               |
| tenant_id           | 4b31d0508f83437e83d8f4d520cda22f     |
+---------------------+--------------------------------------+
```

19. Show second Floating IP.

```
neutron floatingip-show 680c0375-a179-47cb-a8c5-02b836247444
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    | 10.247.94.133                        |
| floating_ip_address | 10.247.96.27                         |
| floating_network_id | d3cb12a6-a000-4e3e-82c4-ee04aa169291 |
| id                  | 680c0375-a179-47cb-a8c5-02b836247444 |
| port_id             | ca95cc24-4e8f-4415-9156-7b519eb36854 |
| router_id           | 6aafc9a9-93f6-4d7e-94f2-3068b034b823 |
| status              | ACTIVE                               |
| tenant_id           | 4b31d0508f83437e83d8f4d520cda22f     |
+---------------------+--------------------------------------+
```

20. Ping first Floating IP.

```
ping -c 1 10.247.96.26
PING 10.247.96.26 (10.247.96.26) 56(84) bytes of data.
64 bytes from 10.247.96.26: icmp_seq=1 ttl=62 time=3.50 ms

--- 10.247.96.26 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 3.505/3.505/3.505/0.000 ms
```

21. Ping second Floating IP.

```
ping -c 1 10.247.96.27
PING 10.247.96.27 (10.247.96.27) 56(84) bytes of data.
64 bytes from 10.247.96.27: icmp_seq=1 ttl=62 time=3.47 ms

--- 10.247.96.27 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 3.473/3.473/3.473/0.000 ms
```

22. Listing the VM's will give you both the fixed and floating IP's for each virutal machine.

```
nova list
+--------------------------------------+-------+--------+------------+-------------+----------------------------------------+
| ID                                   | Name  | Status | Task State | Power State | Networks                               |
```

```
+--------------------------------------+--------+--------+------------+-------------+-------------------------------------+
| dfdfe15b-ce8d-469c-a9d8-2cea0e7ca287 | lb_vm1 | ACTIVE | -          | Running     | lb_net1=10.247.94.132, 10.247.96.26 |
| 3844bb10-2c61-4327-a0d4-0c043c674344 | lb_vm2 | ACTIVE | -          | Running     | lb_net1=10.247.94.133, 10.247.96.27 |
+--------------------------------------+--------+--------+------------+-------------+-------------------------------------+
```

23. List Floating IP's.

```
neutron floatingip-list
+--------------------------------------+-----------------+---------------------+--------------------------------------+
| id                                   | fixed_ip_address | floating_ip_address | port_id                              |
+--------------------------------------+-----------------+---------------------+--------------------------------------+
| 3ce608bf-8835-4638-871d-0efe8ebf55ef | 10.247.94.132   | 10.247.96.26        | 7e5e0038-88cf-4f97-a366-b58cd836450e |
| 680c0375-a179-47cb-a8c5-02b836247444 | 10.247.94.133   | 10.247.96.27        | ca95cc24-4e8f-4415-9156-7b519eb36854 |
+--------------------------------------+-----------------+---------------------+--------------------------------------+
```

Create Load Balancers

The following steps will setup new Octavia Load Balancers.

> **Note:** The following examples assume names and values from the previous section.

1. Create load balancer for subnet

```
neutron lbaas-loadbalancer-create --provider octavia --name lb1 6fc2572c-53b3-41d0-ab63-342d9515f514
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| description         |                                      |
| id                  | 3d9170a1-8605-43e6-9255-e14a8b4aae53 |
| listeners           |                                      |
| name                | lb1                                  |
| operating_status    | OFFLINE                              |
| provider            | octavia                              |
| provisioning_status | PENDING_CREATE                       |
| tenant_id           | 4b31d0508f83437e83d8f4d520cda22f     |
| vip_address         | 10.247.94.134                        |
| vip_port_id         | da28aed3-0eb4-4139-afcf-2d8fd3fc3c51 |
| vip_subnet_id       | 6fc2572c-53b3-41d0-ab63-342d9515f514 |
+---------------------+--------------------------------------+
```

2. List load balancers. You will need to wait until the load balancer `provisioning_status` is `ACTIVE` before proceeding to the next step.

```
neutron lbaas-loadbalancer-list
+--------------------------------------+------+---------------+---------------------+---------+
| id                                   | name | vip_address   | provisioning_status | provider |
+--------------------------------------+------+---------------+---------------------+---------+
| 3d9170a1-8605-43e6-9255-e14a8b4aae53 | lb1  | 10.247.94.134 | ACTIVE              | octavia |
+--------------------------------------+------+---------------+---------------------+---------+
```

3. Once the load balancer is created, create the listener. This may take some time.

```
neutron lbaas-listener-create --loadbalancer lb1 --protocol HTTP --protocol-port=80 --name lb1_listener
+--------------------------+--------------------------------------+
| Field                    | Value                                |
+--------------------------+--------------------------------------+
| admin_state_up           | True                                 |
| connection_limit         | -1                                   |
| default_pool_id          |                                      |
| default_tls_container_ref |                                     |
| description              |                                      |
```

```
| id                       | c723b5c8-e2df-48d5-a54c-fc240ac7b539         |
| loadbalancers            | {"id": "3d9170a1-8605-43e6-9255-e14a8b4aae53"} |
| name                     | lb1_listener                                 |
| protocol                 | HTTP                                         |
| protocol_port            | 80                                           |
| sni_container_refs       |                                              |
| tenant_id                | 4b31d0508f83437e83d8f4d520cda22f             |
+--------------------------+----------------------------------------------+
```

4. Create the load balancing pool. During the creation of the load balancing pool, the status for the load balancer goes to `PENDING_UPDATE`. Use `neutron lbaas-loadbalancer-list` to watch for the change to `ACTIVE`. Once the load balancer returns to `ACTIVE`, proceed with the next step.

```
neutron lbaas-pool-create --lb-algorithm ROUND_ROBIN --listener lb1_listener --protocol HTTP --name lb1_pool
+---------------------+----------------------------------------------+
| Field               | Value                                        |
+---------------------+----------------------------------------------+
| admin_state_up      | True                                         |
| description         |                                              |
| healthmonitor_id    |                                              |
| id                  | 0f5951ee-c2a0-4e62-ae44-e1491a8988e1         |
| lb_algorithm        | ROUND_ROBIN                                  |
| listeners           | {"id": "c723b5c8-e2df-48d5-a54c-fc240ac7b539"} |
| members             |                                              |
| name                | lb1_pool                                     |
| protocol            | HTTP                                         |
| session_persistence |                                              |
| tenant_id           | 4b31d0508f83437e83d8f4d520cda22f             |
+---------------------+----------------------------------------------+
```

5. Create first member of the load balancing pool.

```
neutron lbaas-member-create --subnet 6fc2572c-53b3-41d0-ab63-342d9515f514 --address 10.247.94.132 --protocol-port 80 lb1_pool
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| address        | 10.247.94.132                        |
| admin_state_up | True                                 |
| id             | 61da1e21-e0ae-4158-935a-c909a81470e1 |
| protocol_port  | 80                                   |
| subnet_id      | 6fc2572c-53b3-41d0-ab63-342d9515f514 |
| tenant_id      | 4b31d0508f83437e83d8f4d520cda22f     |
| weight         | 1                                    |
+----------------+--------------------------------------+
```

6. Create the second member.

```
neutron lbaas-member-create --subnet 6fc2572c-53b3-41d0-ab63-342d9515f514 --address 10.247.94.133 --protocol-port 80 lb1_pool
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| address        | 10.247.94.133                        |
| admin_state_up | True                                 |
| id             | 459c7f21-46f7-49e8-9d10-dc7da09f8d5a |
| protocol_port  | 80                                   |
| subnet_id      | 6fc2572c-53b3-41d0-ab63-342d9515f514 |
| tenant_id      | 4b31d0508f83437e83d8f4d520cda22f     |
| weight         | 1                                    |
+----------------+--------------------------------------+
```

7. You should check to make sure the load balancer is active and check the pool members.

```
neutron lbaas-loadbalancer-list
+--------------------------------------+------+---------------+---------------------+----------+
```

```
| id                                   | name | vip_address   | provisioning_status | provider |
+--------------------------------------+------+---------------+---------------------+----------+
| 3d9170a1-8605-43e6-9255-e14a8b4aae53 | lb1  | 10.247.94.134 | ACTIVE              | octavia  |
+--------------------------------------+------+---------------+---------------------+----------+

neutron lbaas-member-list lb1_pool
+--------------------------------------+---------------+---------------+--------+--------------------------------------+----------------+
| id                                   | address       | protocol_port | weight | subnet_id                            | admin_state_up |
+--------------------------------------+---------------+---------------+--------+--------------------------------------+----------------+
| 61da1e21-e0ae-4158-935a-c909a81470e1 | 10.247.94.132 |            80 |      1 | 6fc2572c-53b3-41d0-ab63-342d9515f514 | True           |
| 459c7f21-46f7-49e8-9d10-dc7da09f8d5a | 10.247.94.133 |            80 |      1 | 6fc2572c-53b3-41d0-ab63-342d9515f514 | True           |
+--------------------------------------+---------------+---------------+--------+--------------------------------------+----------------+
```

8. You can view the details of the load balancer, listener and pool.

```
neutron lbaas-loadbalancer-show 3d9170a1-8605-43e6-9255-e14a8b4aae53
+---------------------+----------------------------------------------+
| Field               | Value                                        |
+---------------------+----------------------------------------------+
| admin_state_up      | True                                         |
| description         |                                              |
| id                  | 3d9170a1-8605-43e6-9255-e14a8b4aae53         |
| listeners           | {"id": "c723b5c8-e2df-48d5-a54c-fc240ac7b539"} |
| name                | lb1                                          |
| operating_status    | ONLINE                                       |
| provider            | octavia                                      |
| provisioning_status | ACTIVE                                       |
| tenant_id           | 4b31d0508f83437e83d8f4d520cda22f             |
| vip_address         | 10.247.94.134                                |
| vip_port_id         | da28aed3-0eb4-4139-afcf-2d8fd3fc3c51         |
| vip_subnet_id       | 6fc2572c-53b3-41d0-ab63-342d9515f514         |
+---------------------+----------------------------------------------+

neutron lbaas-listener-list
+--------------------------------------+--------------------------------------+--------------+----------+---------------+----------------+
| id                                   | default_pool_id                      | name         | protocol | protocol_port | admin_state_up |
+--------------------------------------+--------------------------------------+--------------+----------+---------------+----------------+
| c723b5c8-e2df-48d5-a54c-fc240ac7b539 | 0f5951ee-c2a0-4e62-ae44-e1491a8988e1 | lb1_listener | HTTP     |            80 | True           |
+--------------------------------------+--------------------------------------+--------------+----------+---------------+----------------+

neutron lbaas-pool-list
+--------------------------------------+----------+----------+----------------+
| id                                   | name     | protocol | admin_state_up |
+--------------------------------------+----------+----------+----------------+
| 0f5951ee-c2a0-4e62-ae44-e1491a8988e1 | lb1_pool | HTTP     | True           |
+--------------------------------------+----------+----------+----------------+
```

Create Floating IP's for Load Balancer

To create the floating IP's for the load balancer, you will need to list the current ports to get the load balancer id. Once you have the id, you can then create the floating IP.

1. List the current ports.

```
neutron port-list
+--------------------------------------+------+-------------------+--------------------------------------+
| id                                   | name | mac_address       | fixed_ips                            |
+--------------------------------------+------+-------------------+--------------------------------------+
...
| 7e5e0038-88cf-4f97-a366-b58cd836450e |      | fa:16:3e:66:fd:2e | {"subnet_id": "6fc2572c-             |
|                                      |      |                   | 53b3-41d0-ab63-342d9515f514",        |
|                                      |      |                   | "ip_address": "10.247.94.132"}       |
```

```
| a3d0b0fe-e7ff-4b00-a033-44e833b55efe |                                        | fa:16:3e:91:a2:5b | {"subnet_id": "f00299f8-3403-45ae-ac4b- |
|                                      |                                        |                   | 58af41d57bdc", "ip_address":            |
|                                      |                                        |                   | "10.247.94.142"}                        |
| ca95cc24-4e8f-4415-9156-7b519eb36854 |                                        | fa:16:3e:e0:37:c4 | {"subnet_id": "6fc2572c-                 |
|                                      |                                        |                   | 53b3-41d0-ab63-342d9515f514",           |
|                                      |                                        |                   | "ip_address": "10.247.94.133"}          |
| da28aed3-0eb4-4139-afcf-2d8fd3fc3c51 | loadbalancer-                          | fa:16:3e:1d:a2:1c | {"subnet_id": "6fc2572c-                 |
|                                      | 3d9170a1-8605-43e6-9255-e14a8b4aae53   |                   | 53b3-41d0-ab63-342d9515f514",           |
|                                      |                                        |                   | "ip_address": "10.247.94.134"}          |
+--------------------------------------+----------------------------------------+-------------------+----------
```

2. Create the floating IP for the load balancer.

```
neutron floatingip-create ext-net --port-id da28aed3-0eb4-4139-afcf-2d8fd3fc3c51
          Created a new floatingip:
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    | 10.247.94.134                        |
| floating_ip_address | 10.247.96.28                         |
| floating_network_id | d3cb12a6-a000-4e3e-82c4-ee04aa169291 |
| id                  | 9a3629bd-b0a6-474c-abe9-89c6ecb2b22c |
| port_id             | da28aed3-0eb4-4139-afcf-2d8fd3fc3c51 |
| router_id           | 6aafc9a9-93f6-4d7e-94f2-3068b034b823 |
| status              | DOWN                                 |
| tenant_id           | 4b31d0508f83437e83d8f4d520cda22f     |
+---------------------+--------------------------------------+
```

Testing the Octavia Load Balancer

To test the load balancers, create the following web server script so you can run it on each virtual machine. You will use `curl <ip address>` to test if the load balance services are responding properly.

1. Start running web servers on both of the virtual machines. Create the webserver.sh script with below contents. In this example, the port is 80.

```
vi webserver.sh

#!/bin/bash

MYIP=$(/sbin/ifconfig eth0|grep 'inet addr'|awk -F: '{print $2}'| awk '{print $1}');
while true; do
    echo -e "HTTP/1.0 200 OK\r\n\r\nWelcome to $MYIP" | sudo nc -l -p 80
done
```

2. Deploy the web server and run it on the first virtual machine.

```
ssh-keygen -R 10.247.96.26
/home/stack/.ssh/known_hosts updated.
Original contents retained as /home/stack/.ssh/known_hosts.old

scp -o StrictHostKeyChecking=no -i lb_kp1.pem webserver.sh cirros@10.247.96.26:
webserver.sh                                     100% 263      0.3KB/s   00:00

ssh -o StrictHostKeyChecking=no -i lb_kp1.pem cirros@10.247.96.26 'chmod +x ./webserver.sh'
ssh -o StrictHostKeyChecking=no -i lb_kp1.pem cirros@10.247.96.26 ./webserver.sh
```

3. Test the first web server.

```
curl 10.247.96.26
 Welcome to 10.247.94.132
```

4. Deploy and start the web server on the second virtual machine like you did in the previous steps. Once the second web server is running, list the floating IP's.

```
neutron floatingip-list
+--------------------------------------+-----------------+---------------------+--------------------------------------+
| id                                   | fixed_ip_address | floating_ip_address | port_id                              |
+--------------------------------------+-----------------+---------------------+--------------------------------------+
| 3ce608bf-8835-4638-871d-0efe8ebf55ef | 10.247.94.132   | 10.247.96.26        | 7e5e0038-88cf-4f97-a366-b58cd836450e |
| 680c0375-a179-47cb-a8c5-02b836247444 | 10.247.94.133   | 10.247.96.27        | ca95cc24-4e8f-4415-9156-7b519eb36854 |
| 9a3629bd-b0a6-474c-abe9-89c6ecb2b22c | 10.247.94.134   | 10.247.96.28        | da28aed3-0eb4-4139-afcf-2d8fd3fc3c51 |
+--------------------------------------+-----------------+---------------------+--------------------------------------+
```

5. Display the floating IP for the load balancer.

```
neutron floatingip-show 9a3629bd-b0a6-474c-abe9-89c6ecb2b22c
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    | 10.247.94.134                        |
| floating_ip_address | 10.247.96.28                         |
| floating_network_id | d3cb12a6-a000-4e3e-82c4-ee04aa169291 |
| id                  | 9a3629bd-b0a6-474c-abe9-89c6ecb2b22c |
| port_id             | da28aed3-0eb4-4139-afcf-2d8fd3fc3c51 |
| router_id           | 6aafc9a9-93f6-4d7e-94f2-3068b034b823 |
| status              | ACTIVE                               |
| tenant_id           | 4b31d0508f83437e83d8f4d520cda22f     |
+---------------------+--------------------------------------+
```

6. Finally, test the load balancing.

```
curl 10.247.96.28
Welcome to 10.247.94.132

curl 10.247.96.28
Welcome to 10.247.94.133

curl 10.247.96.28
Welcome to 10.247.94.132

curl 10.247.96.28
Welcome to 10.247.94.133

curl 10.247.96.28
Welcome to 10.247.94.132

curl 10.247.96.28
Welcome to 10.247.94.133
```

HPE Helion OpenStack® 5.0: Other Common Post-Installation Tasks

This is a collection of other common post-install tasks such as how to determine your admin user credentials and how to configure your environmental variables to start using the command-line tools.

Determining Your User Credentials

On your lifecycle manager, in the `~/scratch/ansible/next/hos/ansible/group_vars/` directory you will find several files. In the one labeled as first control plane node you can locate the user credentials for both the Administrator user (`admin`) and your Demo user (`demo`) which you will use to perform many other actions on your cloud.

For example, if you are using the Entry-scale KVM with VSA model and used the default naming scheme given in the example configuration files, you can use these commands on your lifecycle manager to GREP for your user credentials:

Administrator

```
grep keystone_admin_pwd entry-scale-kvm-vsa-control-plane-1
```

Demo

```
grep keystone_demo_pwd entry-scale-kvm-vsa-control-plane-1
```

Configure your lifecycle manager to use the command-line tools

This playbook will do a series of steps to update your environment variables for your cloud so you can use command-line clients.

Run the following command, which will replace /etc/hosts on the lifecycle manager:

```
cd ~/scratch/ansible/next/hos/ansible
ansible-playbook -i hosts/verb_hosts cloud-client-setup.yml
```

As the /etc/hosts file no longer has entries for Helion lifecycle management, sudo commands may become a bit slower. To fix this issue, once this step is complete, add "hlm" after "127.0.0.1 localhost". The result will look like this:

```
...
# Localhost Information
127.0.0.1 localhost hlm
```

Protect home directory

The home directory of the user that owns the HPE Helion OpenStack 5.0 scripts should not be world readable. Change the permissions so that they are only readable by the owner:

```
chmod 0700 ~
```

Back up Your SSH Keys

As part of the cloud deployment setup process, SSH keys to access the systems are generated and stored in ~/.ssh on your lifecycle manager.

These SSH keys allow access to the subsequently deployed systems and should be included in the list of content to be archived in any backup strategy.

Retrieving Service Endpoints

1. Log in to your lifecycle manager.
2. Source the keystone admin creds:

```
unset OS_TENANT_NAME
source ~/keystone.osrc
```

3. Using the OpenStack command-line tool you can then query the Keystone service for your endpoints:

```
openstack endpoint list
```

> **Tip:** You can use `openstack -h` to access the client help file and a full list of commands.

To learn more about Keystone, see *Keystone Overview*

Other Common Post-Installation Tasks

Here are the links to other common post-installation tasks that either the Administrator or Demo users can perform:

- *HPE Helion OpenStack 5.0: Enabling the Nova Resize and Migrate Features*
- *HPE Helion OpenStack 5.0: Creating an External Network*
- *HPE Helion OpenStack 5.0: Uploading an Image for Use*
- *HPE Helion OpenStack 5.0: Creating a Private Network*
- *HPE Helion OpenStack 5.0: Running the Swift Dispersion Report*