NetApp Verified Architecture

NetApp HCI for Private Cloud with Red Hat

NVA Deploy

Amit Borulkar and Alan Cowles, NetApp
September 2019 | NVA-1133-DEPLOY

Abstract

NetApp® HCI for Private Cloud with Red Hat is a prevalidated, best-practice data center architecture for deploying an OpenStack-based private cloud environment in a reliable and risk-free manner. This reference architecture also showcases running Red Hat OpenShift Container Platform on OpenStack for microservices-based workloads.

In partnership with

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Solution Overview

NetApp® HCI for Private Cloud with Red Hat is a prevalidated, best-practice data center architecture for deploying OpenStack private cloud infrastructure at enterprise scale. This document describes the enterprise requirements for deploying production-grade Red Hat OpenStack Platform on NetApp HCI. It also describes the various design choices and technical requirements to achieve a flexible, reliable, and predictable infrastructure that scales independently with your application demands. The architecture described in this document is designed and engineered by subject matter experts from NetApp and Red Hat to provide the advantages of open source innovation with enterprise robustness.

NetApp HCI provides the widely recognized benefits of hyperconverged solutions, including lower TCO, ease of purchase and deployment, and effective tools for growth and management of virtualized workloads. NetApp HCI also allows IT to scale storage and compute separately.

NetApp HCI with the Red Hat OpenStack Platform provides a cloud-like experience while simplifying day-to-day operations and management. Together, these technology solutions provide for ease of procurement, deployment, and ongoing management and growth of your business-critical hardware, virtual resources, and enterprise applications.

1.1 NetApp HCI Design Principles

By providing an agile, turnkey infrastructure platform, NetApp HCI enables you to run enterprise-class virtualized and containerized workloads in an accelerated manner. At its core, NetApp HCI is designed to provide predictable performance, linear scalability, and a simple deployment and management experience.

Predictable

One of the biggest challenges in a multitenant environment is delivering consistent predictable performance for all your workloads. Running multiple enterprise-grade workloads can result in resource contention, in which one workload might interfere with the performance of another. NetApp HCI alleviates this concern with quality of service (QoS) limits that are available natively with NetApp Element software. NetApp Element software allows the granular control of every application and volume, eliminates noisy neighbors, and satisfies performance SLAs. NetApp HCI multitenancy capabilities can help eliminate more than 90% of traditional performance-related problems[1].

Flexible

Previous generations of HCI required fixed resource ratios, limiting deployments to four- to eight-node configurations. NetApp HCI, however, scales compute and storage resources independently. Independent scaling prevents costly and inefficient overprovisioning, eliminates the 10% to 30% HCI tax from controller VM overhead, and simplifies capacity and performance planning. With NetApp HCI, licensing costs are reduced. NetApp HCI is available in mix-and-match small, medium, and large storage and compute configurations. The available architectural design choices enable you to confidently scale on your terms, making HCI viable for core data-center applications and platforms.

NetApp HCI is architected in building blocks at either the chassis or the node level. Each chassis can hold four nodes made up of storage nodes, compute nodes, or both. A minimum configuration is two chassis with eight nodes consisting of four storage nodes and four compute nodes.

[1] Source: https://www.netapp.com/us/resources/esg-lab-report-quantifying-the-economic-value-of-asolidfire-deployment.

The Red Hat OpenStack Platform supports adding additional nodes in a nondisruptive manner by using the Red Hat OpenStack Platform Director. It is easy to resize OpenShift VMs and create new worker nodes as your scale needs change. You can also upgrade operating systems nondisruptively with rollback by using NetApp Snapshot™ copy technology and maintain HA with native HA and antiaffinity rules.

Simple

A driving imperative within the IT community is to automate all routine tasks, eliminating the risk of user error while freeing up resources to focus on more interesting, higher-value projects. NetApp HCI can help your IT department become more agile and responsive by simplifying deployment and ongoing management. The primary deployment mechanism for this reference architecture is the Red Hat OpenStack Platform Director, which leverages both Puppet manifests and Ansible playbooks to automate the deployment of the private cloud environment.

1.2 Solution Capabilities

NetApp HCI for Private Cloud with Red Hat enables a fully integrated and production-grade OpenStack deployment that offers the following capabilities:

- Multitenancy and quotas for resources
- Isolation of networks with overlapping IP address space
- Block devices for workloads with performance guarantees
- Object storage
- Networking constructs such as virtual routers
- Provider networks to communicate with devices in the data center
- Load Balancing as a Service (LBaaS)
- Booting instances with persistent storage
- Automated and fully supported OpenShift deployment for running containerized workloads
- Security compliance

1.3 Target Audience

The target audience for the solution includes the following groups:

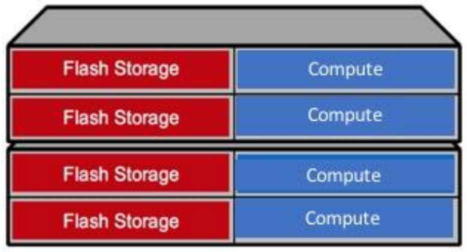
- Enterprise IT cloud administrators
- Service providers
- NetApp and Red Hat partners
- DevOps practitioners

2 Solution Technology

2.1 NetApp HCI

NetApp HCI is an enterprise-scale hybrid cloud infrastructure (HCI) solution that delivers compute and storage resources in an agile, scalable, easy-to-manage two-rack unit (2RU or 1RU), four-node building block. This solution is based on the minimum configuration shown in Figure 1. The minimum deployment as depicted in this guide consists of 4 NetApp HCI storage nodes, and 4 NetApp HCI compute nodes, with the compute nodes assigned respective roles as 3 OpenStack Controllers, and 1 OpenStack compute node. The minimum deployment as depicted can be easily scaled to fit the customer enterprise workload demands by adding additional NetApp HCI storage or compute nodes to expand available storage or add additional hypervisors for fault tolerance.

Figure 1) NetApp HCI minimum configuration for private cloud with Red Hat.



NetApp HCI for private cloud with Red Hat consists of the following components:

- NetApp H-Series all-flash storage nodes running NetApp Element software
- NetApp H-Series compute nodes running Red Hat OpenStack Platform 13, which provides enterprise-grade, hardened OpenStack cloud

For more information about compute and storage nodes in NetApp HCI, see the NetApp HCI datasheet.

2.2 Red Hat OpenStack Platform

The OpenStack Platform delivers an integrated foundation to create, deploy, and scale a secure and reliable private OpenStack cloud. The OpenStack Platform infrastructure-as-a-service (IaaS) cloud is implemented by a collection of control services that manage compute, storage, and networking resources. The environment is managed using a web-based interface that allows administrators and users to control, provision, and automate OpenStack resources. Additionally, the OpenStack infrastructure is facilitated through an extensive command line interface and API enabling full automation capabilities for administrators and end-users.

This reference architecture is based on OpenStack Platform version13 which is a long-term support release of OpenStack from Red Hat.

Figure 2 provides a high-level overview of core OpenStack services and their relationship to each other.

Figure 2) OpenStack components.

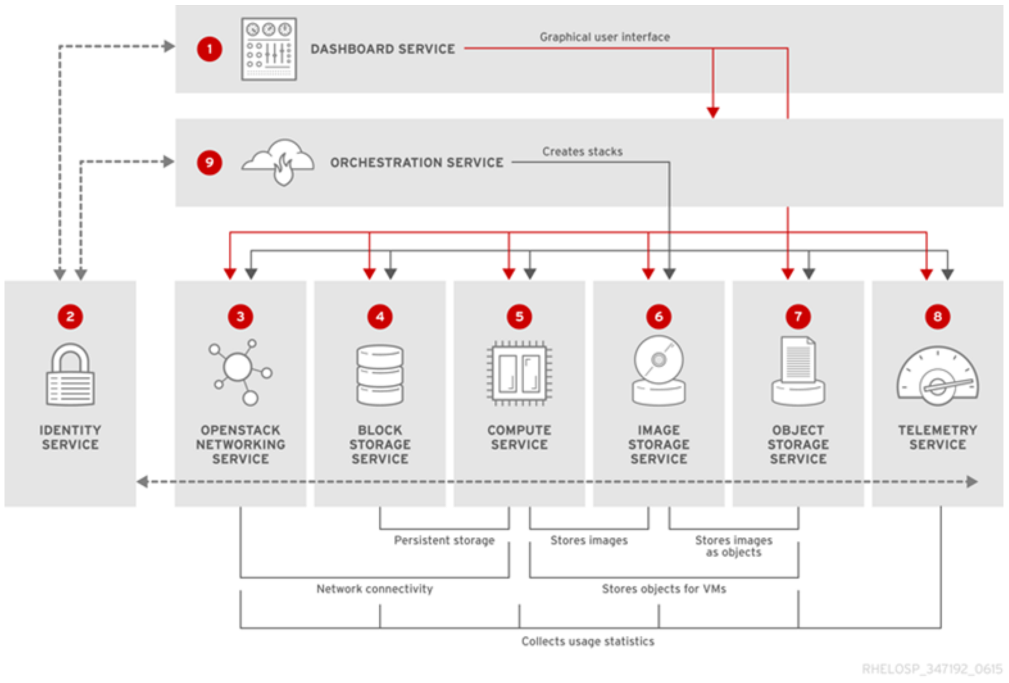


Table 1 describes each OpenStack component.

Table 1) OpenStack components.

Service	Name	Description
Dashboard	Horizon	Web browser-based dashboard that you use to manage OpenStack services
Identity	Keystone	Centralized service for authentication and authorization of OpenStack services and for managing users, projects, and roles
OpenStack networking	Neutron	Provides connectivity between the interfaces of OpenStack services
Block storage	Cinder	Manages persistent block storage volumes for virtual machines (VMs)
Compute	Nova	Manages and provisions VMs running on hypervisor nodes
Image	Glance	Registry service used to store resources such as VM images and volume snapshots
Object storage	Swift	Allows users to store and retrieve files and arbitrary data
Telemetry	Ceilometer	Provides measurements of cloud resources
Orchestration	Heat	Template-based orchestration engine that supports automatic creation of resource stacks

Containerized Services

All OpenStack Platform services are deployed as containers, which isolates services from one another and enables easy upgrades. The OpenStack Platform uses a set of containers built and managed with Kolla. The deployment of services is performed by pulling container images from the Red Hat registry. These service containers are managed using Docker container runtime and are deployed, configured, and maintained with Red Hat OpenStack Director.

2.3 NetApp Element Software

NetApp Element software provides modular, scalable performance, with each storage node delivering guaranteed capacity and throughput to the environment.

iSCSI Login Redirection and Self-Healing Capabilities

NetApp Element software leverages the iSCSI storage protocol, a standard way to encapsulate SCSI commands on a traditional TCP/IP network. When SCSI standards change or when the performance of Ethernet networks improves, the iSCSI storage protocol benefits without the need for any changes.

Although all storage nodes have a management IP and a storage IP, NetApp Element software advertises a single storage virtual IP address (SVIP address) for all storage traffic in the cluster. As a part of the iSCSI login process, storage can respond that the target volume has been moved to a different address and therefore it cannot proceed with the negotiation process. The host then reissues the login request to the new address in a process that requires no host-side reconfiguration. This process is known as iSCSI login redirection.

iSCSI login redirection is a key part of the NetApp Element software cluster. When a host login request is received, the node decides which member of the cluster should handle the traffic based on the IOPS and the capacity requirements for the volume. Volumes are distributed across the NetApp Element software cluster and are redistributed if a single node is handling too much traffic for its volumes or if a new node is added. Multiple copies of a given volume are allocated across the array. In this manner, if a node failure is followed by volume redistribution, there is no effect on host connectivity beyond a logout and login with redirection to the new location. With iSCSI login redirection, a NetApp Element software cluster is a self-healing, scale-out architecture that is capable of nondisruptive upgrades and operations.

NetApp Element Software Cluster QoS

A NetApp Element software cluster allows QoS to be dynamically configured on a per-volume basis. You can use per-volume QoS settings to control storage performance based on SLAs that you define. The following three configurable parameters define the QoS:

- **Minimum IOPS.** The minimum number of sustained IOPS that the NetApp Element software cluster provides to a volume. The minimum IOPS configured for a volume is the guaranteed level of performance for a volume. Per-volume performance does not drop below this level.
- **Maximum IOPS.** The maximum number of sustained IOPS that the NetApp Element software cluster provides to a particular volume.
- **Burst IOPS**. The maximum number of IOPS allowed in a short burst scenario. The burst duration setting is configurable, with a default of 1 minute. If a volume has been running below the maximum IOPS level, burst credits are accumulated. When performance levels become very high and are pushed, short bursts of IOPS beyond the maximum IOPS are allowed on the volume.

Multitenancy

Secure multitenancy is achieved with the following features:

- **Secure authentication.** The Challenge-Handshake Authentication Protocol (CHAP) is used for secure volume access. The Lightweight Directory Access Protocol (LDAP) is used for secure access to the cluster for management and reporting.
- **Volume access groups (VAGs).** Optionally, VAGs can be used in lieu of authentication, mapping any number of iSCSI initiator-specific iSCSI Qualified Names (IQNs) to one or more volumes. To access a volume in a VAG, the initiator’s IQN must be in the allowed IQN list for the group of volumes.
- **Tenant virtual LANs (VLANs).** At the network level, end-to-end network security between iSCSI initiators and the NetApp Element software cluster is facilitated by using VLANs. For any VLAN that is created to isolate a workload or a tenant, NetApp Element Software creates a separate iSCSI target SVIP address that is accessible only through the specific VLAN.
- **VRF-enabled VLANs.** To further support security and scalability in the data center, NetApp Element software allows you to enable any tenant VLAN for VRF-like functionality. This feature adds these two key capabilities:
  - **L3 routing to a tenant SVIP address.** This feature allows you to situate iSCSI initiators on a separate network or VLAN from that of the NetApp Element software cluster.
  - **Overlapping or duplicate IP subnets.** This feature enables you to add a template to tenant environments, allowing each respective tenant VLAN to be assigned IP addresses from the same IP subnet. This capability can be useful for in-service provider environments where scale and preservation of IPspace are important.

Enterprise Storage Efficiencies

The NetApp Element software cluster increases overall storage efficiency and performance. The following features are performed inline, are always on, and require no manual configuration by the user:

- **Deduplication.** The system only stores unique 4K blocks. Any duplicate 4K blocks are automatically associated to an already stored version of the data. Data is on block drives and is mirrored by using the NetApp Element software Helix data protection. This system significantly reduces capacity consumption and write operations within the system.

- **Compression.** Compression is performed inline before data is written to NVRAM. Data is compressed, stored in 4K blocks, and remains compressed in the system. This compression significantly reduces capacity consumption, write operations, and bandwidth consumption across the cluster.
- **Thin provisioning.** This capability provides the right amount of storage at the time that you need it, eliminating capacity consumption that caused by overprovisioned volumes or underutilized volumes.
- **Helix.** The metadata for an individual volume is stored on a metadata drive and is replicated to a secondary metadata drive for redundancy.

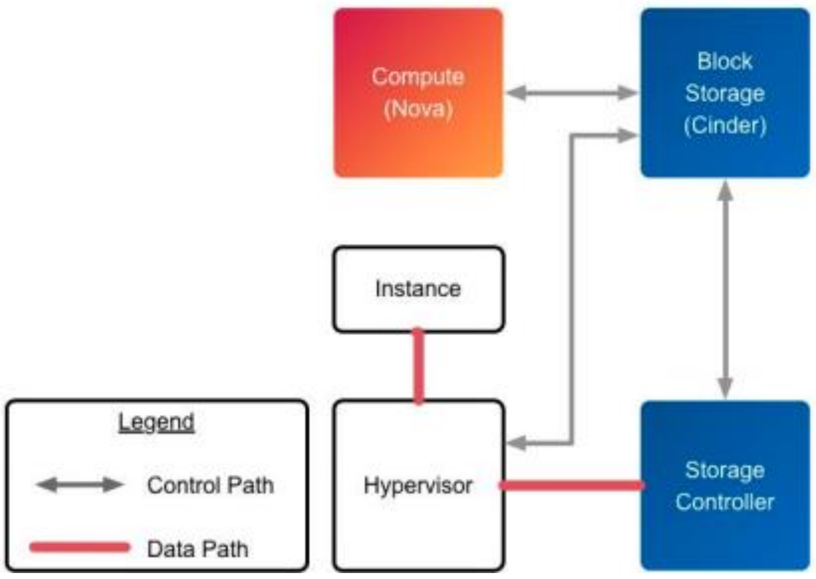
  Note: Element was designed for automation. All the storage features are available through APIs. These APIs are the only method that the UI uses to control the system.

For more information, see the [Element Software product page.]()

OpenStack Cinder

The OpenStack Block Storage service provides management of persistent block-storage resources. In addition to acting as secondarily attached persistent storage, you can write images into a Cinder volume for Nova to utilize as a bootable, persistent root volume for an instance.

Figure 3) Cinder data path and management path.



As a management service, Cinder controls the provisioning and lifecycle management of block storage volumes. It does not reside in the I/O (data) path between the hypervisor and the storage controller.

Glance

The OpenStack image service (Glance) provides discovery, registration, and delivery services for VM, disk, and server images. Glance provides a RESTful API that allows the querying of VM image metadata as well as the retrieval of the actual image. A stored image can be used as a template to start up new servers quickly and consistently, rather than provisioning multiple servers, installing a server operating system, and individually configuring additional services. Such an image can also be used to store and catalog an unlimited number of backups.

To make sure that VM images are consistent and highly available for the Glance service, Swift storage is used as a backend for Glance.

Swift

Swift is a highly available, distributed, and consistent object store powered by the Swift replication mechanism. Object storage does not present a traditional file system. Instead, object storage is a distributed storage system for static, unstructured data objects such as VM images, photo storage, e-mail storage, backups, and archives. Swift is also configured as a default back end for storing VM images for Glance.

Customers can use NetApp HCI storage to provide block storage (LUNs) for the Swift service and then scale horizontally by adding additional NetApp HCI storage nodes as the object store grows. NetApp SolidFire hosts the Swift data using the iSCSI protocol. The three OpenStack controller nodes are also used as Swift nodes and handle account, container, and object services for Swift. In addition, these three nodes also serve as proxy servers for the Swift service.

Swift uses zoning to isolate the cluster into separate partitions and isolate the cluster from failures. Swift data is replicated across the cluster in zones that are as unique as possible. Typically, zones are established using the physical attributes of the cluster, including geographical locations, separate networks, equipment racks, storage subsystems, or even single drives. Zoning allows the cluster to function and tolerate equipment failures without data loss or loss of connectivity to the remaining cluster.

By default, Swift replicates data three times across the cluster. Swift replicates data across zones in a unique way that promotes HA and high durability. Swift chooses a server in an unused zone before it chooses an unused server in a zone that already has a replica of the data.

Octavia: OpenStack LBaaS

Octavia is a supported load balancer solution that NetApp recommends using in conjunction with the OpenShift Container Platform to load balance the external incoming traffic. Octavia also provides a single view of the OpenShift Container Platform master services for the applications. Octavia is automatically deployed as a part of OpenStack deployment.

Heat Orchestration Templates for NetApp HCI

Heat Orchestration Templates (HOT) are an integral part of the deployment and configuration of the OpenStack Platform. After the Overcloud nodes are PXE booted, these templates configure the NICs on the Overcloud nodes and install and configure the relevant packages required for OpenStack deployment. As a part of this verified architecture, HOT was developed to provide a turn-key private-cloud boot-strapping experience.

Note: Manila provides a canonical storage-provisioning control plane in OpenStack for shared or distributed file systems. Manila shares are backed by file systems. OpenStack users manage shares using Manila services. Customers can use existing ONTAP-based storage systems as back ends for Manila. More details on configuring Manila for ONTAP can be founds here.

2.4 Hardware Requirements

Table 2 lists the hardware components that are required to implement the solution. The hardware components that are used in any particular implementation of the solution might vary based on customer requirements.

Table 2) Hardware requirements.

Layer	Product Family	Number of Nodes	Details
Compute	NetApp 410C	4	OpenStack Platform Overcloud nodes
Network	Mellanox SN2010	2	Mellanox switches
Storage	NetApp 410S	4	6x 960GB encrypting/non-encrypting

2.5 Software Requirements

Table 3 lists the software components required to implement the solution. The software components that are used in any particular implementation of the solution might vary based on customer requirements.

Table 3) Software requirements.

Layer	Software	Version (or Other Information)
Storage	NetApp Element software	11.1
	Trident	19.01
Network	Onyx	3.6.8008
Cloud engine (IaaS)	OpenStack Platform	13
Underlying operating system	Red Hat Enterprise Linux	7.6
Container PaaS	Red Hat OpenShift Container Platform	3.11

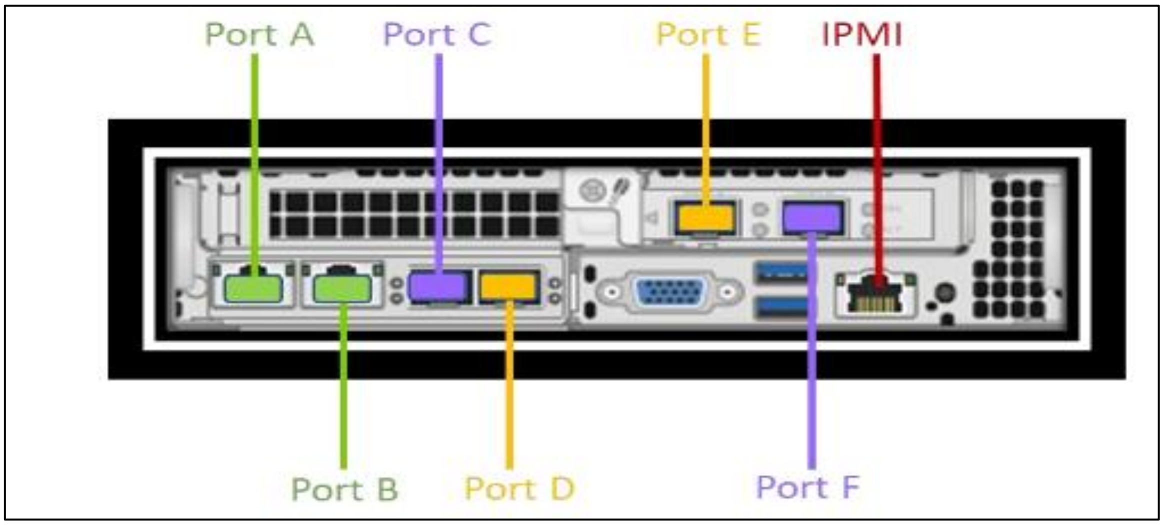
3 Networking Requirements

This section details the networking requirements for the deployment of the Red Hat Private Cloud as a validated solution. It provides physical diagrams of the network ports on both the NetApp HCI compute nodes and the Mellanox SN2010 switches deployed in the solution. The section also details each virtual network segment being used, and the intended purpose of each in the solution, along with a sample switch configuration for the Mellanox SN2010 switches.

3.1 Port Identification

NetApp HCI consists of NetApp H-Series nodes dedicated to either compute or storage. Both node configurations are available with two 1GbE ports (identified as ports A and B) and two 10/25 GbE ports identified as C and D on-board. The compute nodes have additional 10/25GbE ports identified as E and F available in the first mezzanine slot. Each node also has an additional out-of-band management port that allows for Intelligent Platform Management Interface (IPMI) functionality. Figure 4 identifies each of these ports on the rear of an H410C node.

Figure 4) NetApp HCI network ports.



3.2 Network Design

The NetApp HCI for Private Cloud with Red Hat solution uses two Mellanox SN2010 switches to provide primary data connectivity at 10/25Gbps. It also uses two additional management switches that provide connectivity at 1Gbps for in-band management for the storage nodes and out-of-band management for IPMI functionality. The Red Hat OpenStack Platform Director uses IPMI to power on the compute nodes, and PXE to boot and deploy the system images. OpenStack Platform maps the different services onto separate network traffic types that are isolated by using VLANs.

Network isolation among different tenants can be achieved by using the VXLAN encapsulation protocol with the ML2 OVS plug-in. A tenant network (an underlay network for VXLAN) is provisioned during deployment. The overlay networks corresponding to different tenants are provisioned dynamically by the Neutron service; they do not require extra configuration on the switches if the required VLANs are present. The isolation of tenant networks is enforced by Linux kernel network namespaces leveraged by Red Hat Enterprise Linux.

Table 4) Networks required for OpenStack.

Network type	VLAN ID	Description	Nodes
IPMI	15	Network used for power management of nodes. This network uses the IPMI port on the NetApp HCI compute nodes.	All nodes
Storage-Infra	201	The storage-infra network allows for controller nodes to map volumes directly from NetApp Element in support of infrastructure services.	Controller nodes
Storage-Cinder	202	This corresponds to the storage traffic to the Element Nodes within the OpenStack Platform environment.	Overcloud nodes
Internal API	301	The Internal API network is used for communication between the OpenStack services using API communication, RPC messages, and database communication.	Overcloud nodes

Network type	VLAN ID	Description	Nodes
Tenant	302	Neutron provides each tenant with their own networks via tunneling through VXLAN. Network traffic is isolated within each tenant network. Each tenant network has an IP subnet associated with it, and network namespaces mean that multiple tenant networks can use the same address range without causing conflicts.	Overcloud nodes
Storage-Mgmt	303	OpenStack Object Storage (Swift) uses this network to synchronize data objects between participating replica nodes. The proxy service acts as the intermediary interface between user requests and the underlying storage layer. The proxy receives incoming requests and locates the necessary replica to retrieve the requested data.	Controller nodes
PXE	3484	The director uses this network to deploy new nodes over PXE boot and orchestrate the installation of OpenStack Platform on the Overcloud bare metal servers. The Mellanox NIC (10/25GbE) ports on the compute nodes are enabled for PXE boot.	All nodes
External	3485	Hosts the OpenStack dashboard (Horizon) for graphical system management and the public APIs for OpenStack services.	Controller nodes
IB Management	3486	Provides access for system administration functions such as SSH access, DNS traffic, and Network Time Protocol (NTP) traffic. This network also acts as a gateway for non-controller nodes	Controller nodes

Node introspection and provisioning of nodes is performed over the PXE network. The Undercloud node runs a DHCP server. No other DHCP server can run on the PXE network VLAN.

An OpenStack administrator creates a provider network with the same VLAN segmentation ID as the datacenter network. A provider network enables connectivity between the entities connected to the provider network and the datacenter components connected on the same VLAN.

All the nodes are connected through a 1G IPMI interface. This is also the out-of-band management network for the servers. 10/25GbE interfaces on the NetApp HCI compute nodes are enabled for PXE/iPXE boot by default. The first 10/25GbE interface is used to iPXE boot the Overcloud node from the Director.

The LUNs for Swift storage are provisioned from NetApp Element software and are mounted during the deployment process. Therefore, controller nodes also have access to the storage network.

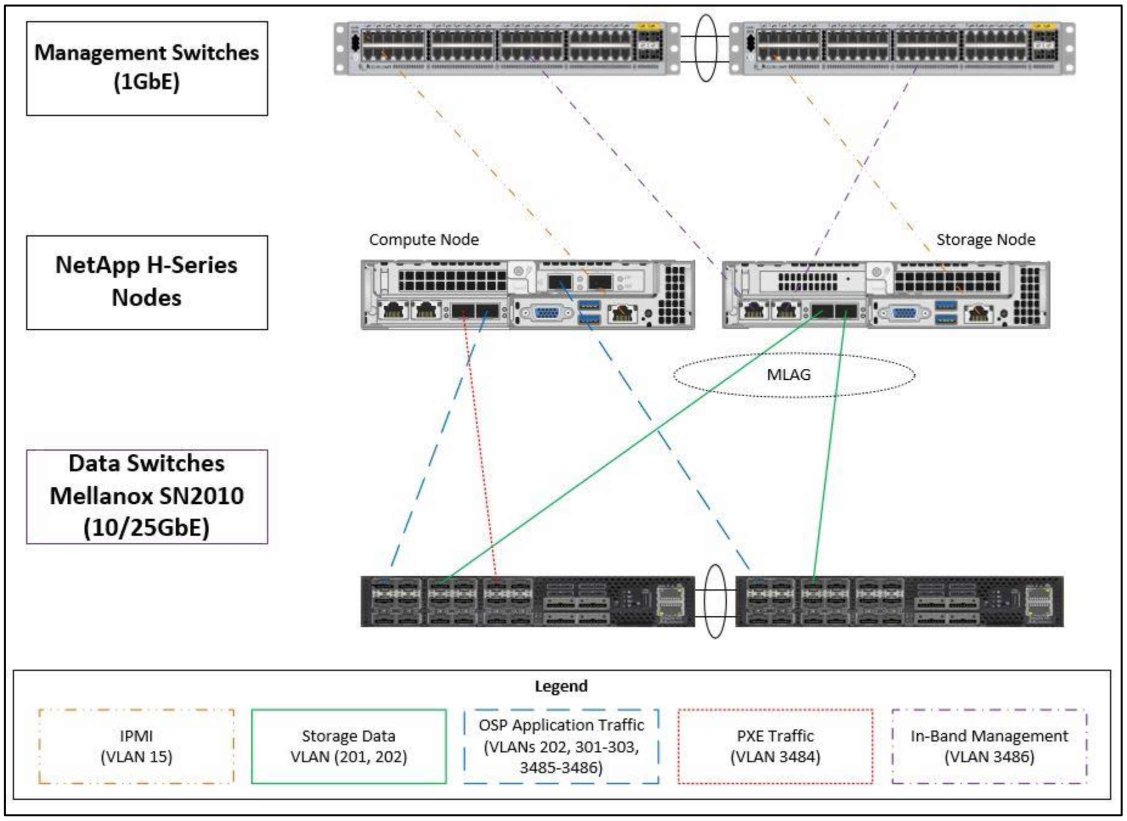
3.3 Switch Configuration

Each of the identified network ports has a defined purpose in the NetApp HCI for Private Cloud with Red Hat solution. Both the NetApp HCI compute and storage nodes allow for out-of-band management through an on-board IPMI port that is connected to the 1Gbps network management switch at the top of the environment.

In addition, the NetApp HCI compute nodes use the 10/25GbE port C for PXE-boot and provisioning and 10/25GbE ports D and E to provide application traffic (Table 1) to the cloud environment. These network links are configured as an OVS bond by the Red Hat Enterprise Linux operating system as a part of the OpenStack Platform deployment. Therefore, no special configuration is required on the switch end.

As for the storage nodes in the solution, both 1GbE ports A and B are used to connect to the NetApp Element management network. Ports C and D (10/25GbE) are connected to separate Mellanox SN2010 switches and joined in a Multi-Chassis Link Aggregation (MLAG) port channel for data connectivity. The connectivity of each of these ports as deployed provides fault tolerance by separating the links between on-board and mezzanine ports on the NetApp HCI compute nodes. It also provides separate upstream connectivity for both storage and compute nodes by ensuring that network cable sets are connected to each separate switch in the environment. To makes sure that each node in the deployment can reach and provide access to the services as needed, the following VLANs are defined and configured onto the supporting infrastructure switches. The configuration depicted in Figure 5 provides connectivity between each of the identified network services in the environment.

Figure 5) Simplified network and cabling view.


Management Switches (1GbE)
NetApp H-Series Nodes
Data Switches Mellanox SN2010 (10/25GbE)
Compute Node
Storage Node
MLAG
Legend
IPMI (VLAN 15)
Storage Data VLAN (201, 202)
OSP Application Traffic (VLANs 202, 301-303, 3485-3486)
PXE Traffic (VLAN 3484)
In-Band Management (VLAN 3486)

The 1GbE management switches depicted in Figure 5 support both IPMI traffic when booting the OpenStack Platform Director deployed nodes and in-band management connectivity when connected to the NetApp HCI storage nodes. The network ports that each of these are connected to must be configured as access ports tagged to their appropriate VLANs, 15 and 3486 respectively.

The configuration of the Mellanox SN2010 switches to support the infrastructure requires the configuration of multiple ports. See the following syntax examples to configure the switch for the environment.

Create MLAG Cluster

To create an MLAG cluster, complete the following steps:

1. Run the following commands on each Mellanox SN2010 switch for general configuration:
   a. Enter configuration mode.

```
Switch-01 enable
```

```
Switch-01 configure terminal
```

b. Enable the Link Aggregation Control Protocol (LACP; required for the Inter Peer Link.

```
Switch-01 (config) # lacp
```

c. Enable the Link Layer Discovery Protocol (LLDP).

```
Switch-01 (config) # lldp
```

d. Enable IP routing.

```
 Switch-01 (config) # ip routing
```

e. Enable the MLAG protocol.

```
Switch-01 (config) # protocol mlag
```

f. Enable QoS globally.

```
Switch-01 (config) # dcb priority-flow-control enable force
```

2. For MLAG to function, the switches must be made peers to each other through an IPL. This should consist of two or more physical links for redundancy. The MTU for the IPL is set for jumbo frames (9126), and all VLANs are enabled by default. Run the following commands on each switch in the domain:

a. Create port channel 10 for IPL.

```
Switch-01 (config) # interface port-channel 10
Switch-01 (config interface port-channel 10) # description IPL
Switch-01 (config interface port-channel 10) # exit
```

b. Add interfaces eth 1/20 and 1/22 to the port channel.

```
Switch-01 (config) # interface ethernet 1/20 channel-group 10 mode active
Switch-01 (config) # interface ethernet 1/20 description ISL-SWB_01
Switch-01 (config) # interface ethernet 1/22 channel-group 10 mode active
Switch-01 (config) # interface ethernet 1/22 description ISL-SWB_02
```

c. Create a VLAN outside of the standard range dedicated to IPL traffic.

```
Switch-01 (config) # vlan 4000
Switch-01 (config vlan 4000) # name IPL VLAN
Switch-01 (config vlan 4000) # exit
```

d. Declare the port channel to be the IPL.

```
Switch-01 (config) # interface port-channel 10 ipl 1
Switch-01 (config) # interface port-channel 10 dcb priority-flow-control mode on force
```

e. Set an IP for each IPL member (non-routable; it is not advertised outside of the switch).

```
Switch-01 (config) # interface vlan 4000
Switch-01 (config vlan 4000) # ip address 10.0.0.1 255.255.255.0
Switch-01 (config vlan 4000) # ipl 1 peer-address 10.0.0.2
Switch-01 (config vlan 4000) # exit
```

3. Create a unique MLAG domain name for the two switches and assign a MLAG VIP (virtual IP). This IP is used for keep-alive heartbeat messages between the two switches. Run these commands on each switch in the domain:

a. Create the MLAG domain and set the IP address and subnet.

```
Switch-01 (config) # mlag-vip MLAG-VIP-DOM ip a.b.c.d /24 force
```

b. Create a virtual MAC address for the system MLAG.

```

Switch-01 (config) # mlag system-mac AA:BB:CC:DD:EE:FF
```

c. Configure the MLAG domain so that it is active globally.

```
Switch-01 (config) # no mlag shutdown
```

Note: The IP used for the MLAG VIP must be in the same subnet as the switch management network (mgmt0).

Note: The MAC address used can be any unicast MAC address and must be set to the same value on both switches in the MLAG domain.

Configure Ports to Connect to Storage and Compute Hosts

To configure ports to connect to storage and compute hosts, complete the following steps:

1. Create each of the VLANs needed to support the services for OpenStack Platform. Run these commands on each switch in the domain:
   a. Create the VLANs identified in Table 4.

```
Switch-01 (config) # vlan 201-202
Switch-01 (config vlan 201-202) exit
Switch-01 (config) # vlan 301-303
Switch-01 (config vlan 301-303) exit
Switch-01 (config) # vlan 602
Switch-01 (config vlan 602) exit
Switch-01 (config) # vlan 3484-3486
Switch-01 (config vlan 3484-3486) exit
```

   b. Create names for each VLAN for easier tracking purposes.

```
Switch-01 (config) # vlan 201 name "Storage-Infra"
Switch-01 (config) # vlan 202 name "Storage-Cinder"
Switch-01 (config) # vlan 301 name "Internal-API"
Switch-01 (config) # vlan 302 name "Tenant"
Switch-01 (config) # vlan 303 name "Storage-Mgmt"
Switch-01 (config) # vlan 303 name "OpenShift-stg"
Switch-01 (config) # vlan 3484 name "PXE"
Switch-01 (config) # vlan 3484 name "External"
Switch-01 (config) # vlan 3484 name "Internal-Mgmt"
```

2. Create Hybrid VLAN ports on ports Eth1/1-4 so that you can tag the appropriate VLANs for your NetApp HCI compute nodes.
   a. Select the ports you want to work with.

```
Switch-01 (config) # interface ethernet 1/1-1/4
```

   b. Set the MTU for each port.

```
Switch-01 (config interface ethernet 1/1-1/4) # mtu 9216 force
```

   c. Modify spanning tree settings for each of the ports.

```
Switch-01 (config interface ethernet 1/1-1/4) # spanning-tree bpdufilter enable
Switch-01 (config interface ethernet 1/1-1/4) # spanning-tree port type edge
Switch-01 (config interface ethernet 1/1-1/4) # spanning-tree bpduguard enable
```

   d. Set the switchport mode to hybrid.

```
Switch-01 (config interface ethernet 1/1-1/4 ) # switchport mode hybrid
Switch-01 (config interface ethernet 1/1-1/4 ) # exit
```

   e. Create descriptions for each port that is being modified.

```
Switch-01 (config) # interface ethernet 1/1 description HCI-CMP-01 eth1:PortD
Switch-01 (config) # interface ethernet 1/2 description HCI-CMP-02 eth1:PortD
Switch-01 (config) # interface ethernet 1/3 description HCI-CMP-03 eth1:PortD
Switch-01 (config) # interface ethernet 1/4 description HCI-CMP-04 eth1:PortD
```

   f. Tag the appropriate VLANs for the OpenStack Platform environment.

```
Switch-01 (config) # interface ethernet 1/1 switchport hybrid allowed-vlan add 201-202
Switch-01 (config) # interface ethernet 1/1 switchport hybrid allowed-vlan add 301-303
```

```
Switch-01 (config) # interface ethernet 1/1 switchport hybrid allowed-vlan add 3485-3486
Switch-01 (config) # interface ethernet 1/2 switchport hybrid allowed-vlan add 201-202
Switch-01 (config) # interface ethernet 1/2 switchport hybrid allowed-vlan add 301-303
Switch-01 (config) # interface ethernet 1/2 switchport hybrid allowed-vlan add 3485-3486
Switch-01 (config) # interface ethernet 1/3 switchport hybrid allowed-vlan add 201-202
Switch-01 (config) # interface ethernet 1/3 switchport hybrid allowed-vlan add 301-303
Switch-01 (config) # interface ethernet 1/3 switchport hybrid allowed-vlan add 3485-3486
Switch-01 (config) # interface ethernet 1/4 switchport hybrid allowed-vlan add 201-202
Switch-01 (config) # interface ethernet 1/4 switchport hybrid allowed-vlan add 301-303
Switch-01 (config) # interface ethernet 1/4 switchport hybrid allowed-vlan add 3485-3486
```

3. Create MLAG interfaces and hybrid VLAN ports on ports Eth1/5-8 so that you can distribute connectivity between the switches and tag the appropriate VLANs for the NetApp HCI storage nodes.
   a. Select the ports you want to work with.

```
Switch-01 (config) # interface ethernet 1/5-1/8
```

   b. Set the MTU for each port.

```
Switch-01 (config interface ethernet 1/5-1/8) # mtu 9216 force
```

   c. Modify spanning tree settings for each of the ports.

```
Switch-01 (config interface ethernet 1/5-1/8) # spanning-tree bpdufilter enable
Switch-01 (config interface ethernet 1/5-1/8) # spanning-tree port type edge
Switch-01 (config interface ethernet 1/5-1/8) # spanning-tree bpduguard enable
```

   d. Set the switchport mode to hybrid.

```
Switch-01 (config interface ethernet 1/5-1/8 ) # switchport mode hybrid
Switch-01 (config interface ethernet 1/8-1/8 ) # exit
```

   e. Create descriptions for each port that is being modified.

```
Switch-01 (config) # interface ethernet 1/5 description HCI-STG-01 eth0:PortC
Switch-01 (config) # interface ethernet 1/6 description HCI-STG-02 eth0:PortC
Switch-01 (config) # interface ethernet 1/7 description HCI-STG-03 eth0:PortC
Switch-01 (config) # interface ethernet 1/8 description HCI-STG-04 eth0:PortC
```

   f. Create and configure the MLAG port channels.

```
Switch-01 (config) # interface mlag-port-channel 115-118
Switch-01 (config interface mlag-port-channel 115-118) # exit
Switch-01 (config) # interface mlag-port-channel 115-118 no shutdown
Switch-01 (config) # interface mlag-port-channel 115-118 mtu 9216 force
Switch-01 (config) # interface mlag-port-channel 115-118 lacp-individual enable force
Switch-01 (config) # interface ethernet 1/5-1/8 lacp port-priority 10
Switch-01 (config) # interface ethernet 1/5-1/8 lacp rate fast
Switch-01 (config) # interface ethernet 1/5 mlag-channel-group 115 mode active
Switch-01 (config) # interface ethernet 1/6 mlag-channel-group 116 mode active
Switch-01 (config) # interface ethernet 1/7 mlag-channel-group 117 mode active
Switch-01 (config) # interface ethernet 1/8 mlag-channel-group 118 mode active
```

   g. Tag the appropriate VLANs for the storage environment.

```
Switch-01 (config) # interface mlag-port-channel 115-118 switchport mode hybrid
Switch-01 (config) # interface mlag-port-channel 115 switchport hybrid allowed-vlan add 201-202
Switch-01 (config) # interface mlag-port-channel 116 switchport hybrid allowed-vlan add 201-202
Switch-01 (config) # interface mlag-port-channel 117 switchport hybrid allowed-vlan add 201-202
Switch-01 (config) # interface mlag-port-channel 118 switchport hybrid allowed-vlan add 201-202
```

4. Create access ports on ports Eth1/13-14 to support PXE services for the NetApp HCI compute nodes.
   a. Select the ports you want to work with.

```
Switch-01 (config) # interface ethernet 1/13-1/14
```

   b. Set the MTU for each port.

```
Switch-01 (config interface ethernet 1/13-1/14) # mtu 9216 force
```

c. Modify spanning tree settings for each of the ports.

```
Switch-01 (config interface ethernet 1/13-1/14) # spanning-tree bpdufilter enable
Switch-01 (config interface ethernet 1/13-1/14) # spanning-tree port type edge
Switch-01 (config interface ethernet 1/13-1/14) # spanning-tree bpduguard enable
Switch-01 (config interface ethernet 1/13-1/14) # exit
```

d. Set the switchport mode to access and set the port VLAN ID.

```
Switch-01 (config) # interface ethernet 1/13
Switch-01 (config interface ethernet 1/13) # switchport mode access
Switch-01 (config interface ethernet 1/13) # switchport access vlan 3484
Switch-01 (config interface ethernet 1/13) # exit
Switch-01 (config) # interface ethernet 1/14
Switch-01 (config interface ethernet 1/13) # switchport mode access
Switch-01 (config interface ethernet 1/13) # switchport access vlan 3484
Switch-01 (config interface ethernet 1/13) # exit
```

e. Create descriptions for each port that is being modified.

```
Switch-01 (config) # interface ethernet 1/13 description HCI-CMP-01 eth0:PortC
Switch-01 (config) # interface ethernet 1/14 description HCI-CMP-03 eth0:PortC
```

Create the Uplink Ports for the Switches

To create the uplink ports for the switches, complete the following steps:

1. Create an MLAG interface to provide uplink to both Mellanox SN2010 switches from your core network.

```
Switch-01 (config) # interface mlag port-channel 101
Switch-01 (config interface mlag port-channel) # description Uplink CORE-SWITCH port PORT
Switch-01 (config interface mlag port-channel) # exit
```

2. Configure the MLAG members.

```
Switch-01 (config) # interface ethernet 1/18 description Uplink to CORE-SWITCH port PORT
Switch-01 (config) # interface ethernet 1/18 speed 10000 force
Switch-01 (config) # interface mlag-port-channel 101 mtu 9216 force
Switch-01 (config) # interface ethernet 1/18 mlag-channel-group 101 mode active
```

3. Set the switchport mode to hybrid and allow all VLANs from the core uplink switches.

```
Switch-01 (config) # interface mlag-port-channel switchport mode hybrid
Switch-01 (config) # interface mlag-port-channel switchport hybrid allowed-vlan all
```

4. Verify that the MLAG interface is up.

```
Switch-01 (config) # interface mlag-port-channel 101 no shutdown
Switch-01 (config) # exit
```

3.4 Cabling the Validated Solution

The NetApp HCI for Private Cloud with Red Hat solution is designed to be deployed in a 4 x 4 form factor, with four nodes dedicated to the compute environment and four nodes dedicated to storage. Each of these node types can be scaled separately to meet the needs of the end user.

Figure 6) Cabling for four-node compute.

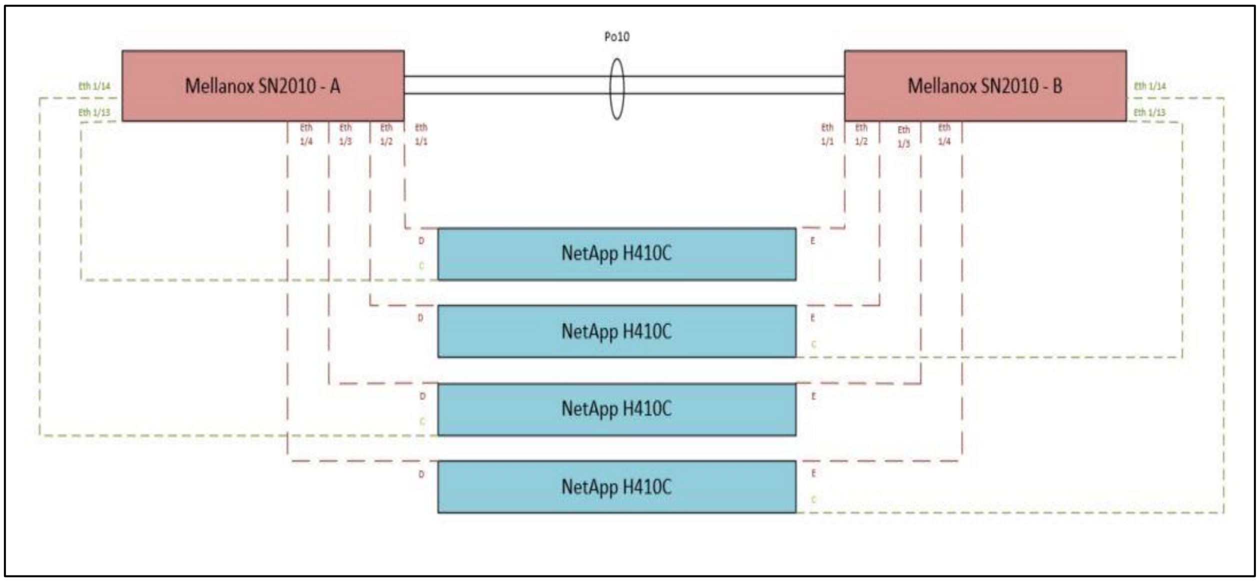
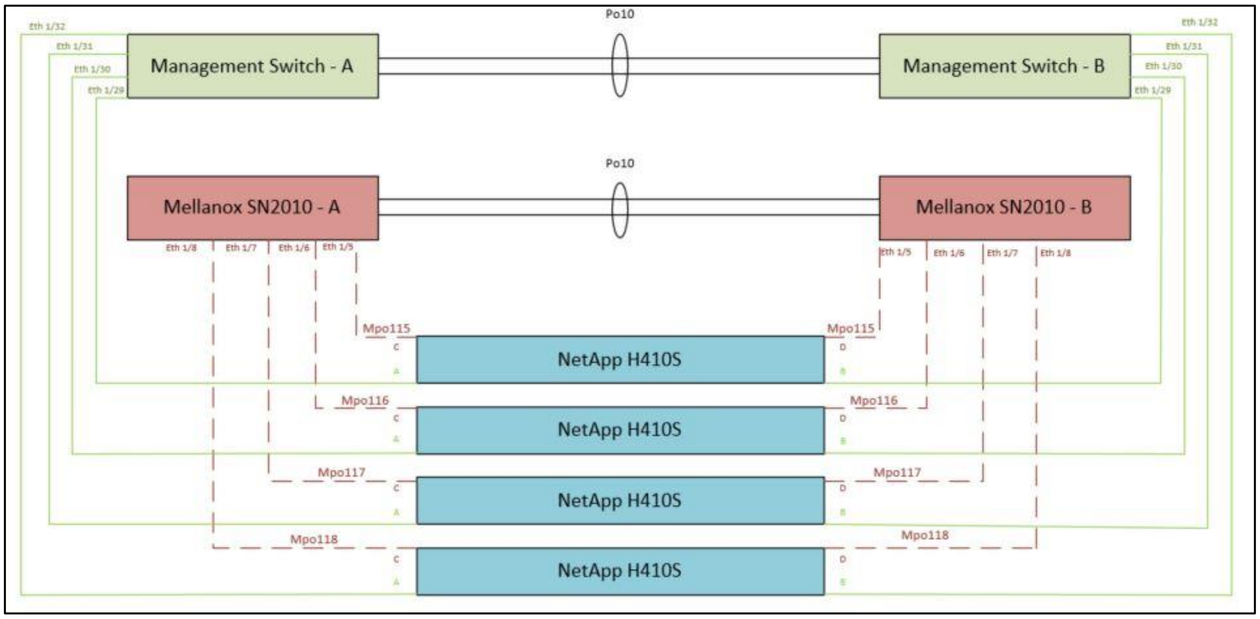


Figure 7) Cabling for four-node storage.



Each of the four NetApp HCI compute nodes has a link from 10/25GbE port D to a port on a Mellanox SN2010-A, and a link from 10/25GbE port E to Mellanox SN2010-B at ports Eth 1/1-4 respectively. These are the network connections that are managed as bonds at the OS level by Red Hat Enterprise Linux during the OpenStack Platform deployment. Therefore, they require no special configuration on the switch ports, other than to ensure that the designated VLANs for application data are available as required.

Each node also has 10/25GbE port C connected to one of the Mellanox switches at ports Eth 1/13-14 respectively to provide PXE and provisioning services. Because only one port is connected from each node to the infrastructure switches, we decided to balance the connections between the switches. This configuration provides both switches with several ports available for future expansion. A cabling diagram of this portion of the solution is available in Figure 8.

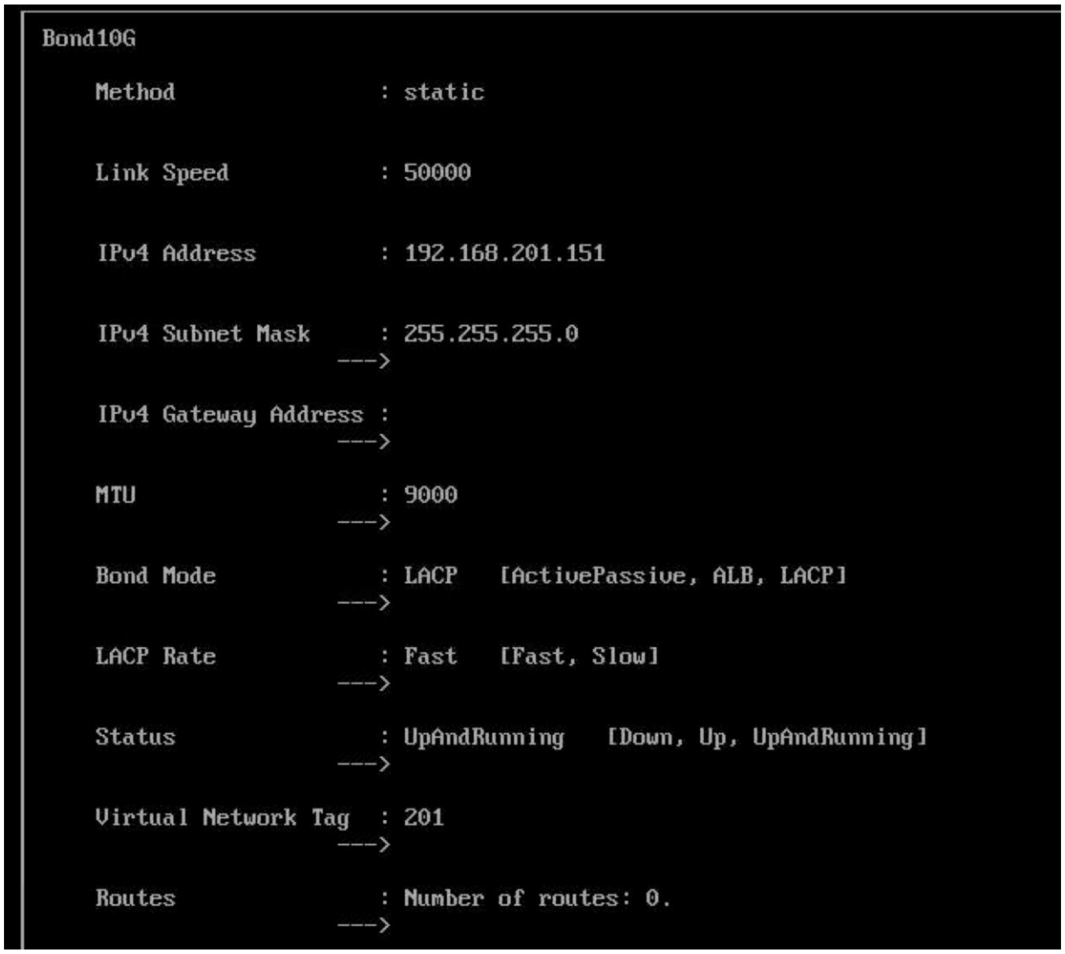
Each of the four NetApp HCI storage nodes has two links from 1GbE ports A and B to the 1Gbps management switches in the environment through ports Eth 1/29-32 respectively. These connections provide access to the NetApp Element management network.

There are also two links from 10/25GbE ports C and D to each Mellanox switch on ports Eth 1/5-8 respectively. These connections provide Element data network connectivity to the Cinder service in the cloud environment. Each of these ports must be configured as part of an MLAG distributed port channel between the two switches. This is denoted by the Mpo115-118 designations as demonstrated in the cable diagram of this portion of the solution in Figure 8.

4 NetApp Element Software Setup

4.1 Basic NetApp Storage Configuration

The NetApp Element cluster setup is performed in a manner similar to a standalone NetApp SolidFire storage setup. To setup a storage cluster, follow this sample configuration for the 10G interfaces with inputs appropriate for your system:

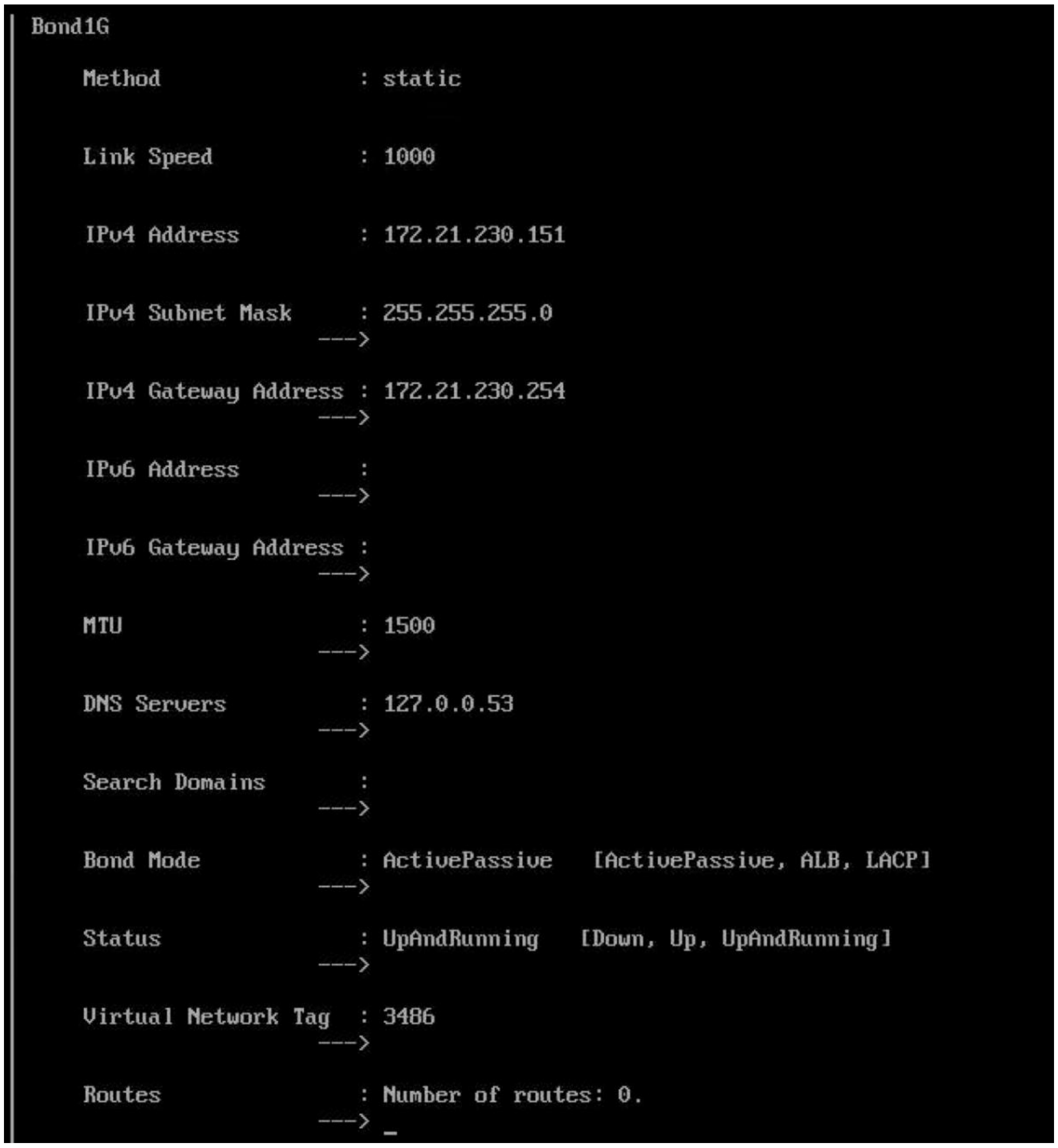


```
Bond10G

    Method                : static

    Link Speed            : 50000

    IPv4 Address          : 192.168.201.151

    IPv4 Subnet Mask      : 255.255.255.0
                      --->

    IPv4 Gateway Address :
                      --->

    MTU                   : 9000
                      --->

    Bond Mode             : LACP   [ActivePassive, ALB, LACP]
                      --->

    LACP Rate             : Fast   [Fast, Slow]
                      --->

    Status                : UpAndRunning   [Down, Up, UpAndRunning]
                      --->

    Virtual Network Tag   : 201
                      --->

    Routes                : Number of routes: 0.
                      --->
```

Note: The virtual network tag corresponds to the `Storage-Infra` VLAN ID.

Note: The 10/25G interfaces are configured in the LACP mode.

In addition, the following screenshot provides a sample configuration for 1G management interfaces:



```
Bond1G

    Method               : static

    Link Speed           : 1000

    IPv4 Address         : 172.21.230.151

    IPv4 Subnet Mask     : 255.255.255.0
                      --->
    IPv4 Gateway Address : 172.21.230.254
                      --->
    IPv6 Address         :
                      --->
    IPv6 Gateway Address :
                      --->
    MTU                  : 1500
                      --->
    DNS Servers          : 127.0.0.53
                      --->
    Search Domains       :
                      --->
    Bond Mode            : ActivePassive   [ActivePassive, ALB, LACP]
                      --->
    Status               : UpAndRunning   [Down, Up, UpAndRunning]
                      --->
    Virtual Network Tag  : 3486
                      --->
    Routes               : Number of routes: 0.
                      ---> _
```

Note: The virtual network tag corresponds to the `IB Management` network.

4.2 Configuration Options for Element in Support of OpenStack Platform Deployment

In our NetApp HCI for Private Cloud with Red Hat solution we will be making use of our NetApp Element storage system to provide backing storage for the OpenStack Swift service for object storage in addition to its use to provide block storage for the Cinder service. While OpenStack has integrated Element support for deploying Cinder volumes with virtual instances once the Overcloud is deployed, the volumes we will use with Swift will need to be made available prior to the deployment, as they are formatted and configured for use by the OpenStack Platform deployment templates. To support this portion of the deployment we will need to manually configure an Account, 3 Volumes of appropriate size, and the associated Initiators and Access Groups to allow the OpenStack Platform Controller nodes to map and mount the volumes for use. Each of these actions can be performed via the web user interface for Element or through API calls to the Element system. For this deployment guide we will step through the steps via the GUI.

Using your preferred web browser, login to your NetApp Element cluster at the mVIP address and click on the Management tab at the top of the screen. From there you will see 5 sub-tabs that you can use to perform each of the following actions:

1. Create an account:
   a. Click on the Accounts sub-tab.
   b. Click the green Create Account button.
   c. Create an account with the username OSP-Controller.
   d. Click the green Create Account button.
2. Create volumes:
   a. Click on the Volumes sub-tab.
   b. Click the green Create Volume button.
   c. Name the volume OSP-Controller-00, size the volume appropriately, 200GB is recommended to allow for cloud images to be uploaded to Swift-backed images stores for Glance.

   Note: A larger volume size might be needed to support larger images or telemetry operations if enabled.

   a. Select the OSP-Controller account you created in the previous step from the drop down.
   b. Click the green button to Create Volume.
   c. Repeat these steps for OSP-Controller-01, and OSP-Controller-02.
3. Create initiators:
   a. Click on the Initiators sub-tab.
   b. Click the green Create Initiator button.
   c. Select the radio button for Bulk Create Initiators.
   d. Create initiators for each controller as a comma separated list.

```
iqn.1994-05.com.redhat:OSP-Controller-00, iqn.1994-05.com.redhat:OSP-Controller-01, iqn.1994-
05.com.redhat:OSP-Controller-02
```

4. Click the Add Initiators button at the bottom of the window.
5. Create Access Groups
6. Click on the Access Groups sub-tab.
7. Click on the green Create Access Group button.
8. Name the access group OSP-Controller-00 and click the respective drop-down menus to choose both the initiator and volume for that host.
9. Click the green Create Access Group button.

   Repeat these steps for OSP-Controller-01, and OSP-Controller-02.

   Note: Each of these volumes should be mapped by the OpenStack Controller nodes to `/dev/sdc` which the deployment templates expect to find, format and mount persistently for future use by the Swift service.

5 Red Hat OpenStack Platform Deployment

Deploying the solution involves the following tasks:

- Install Red Hat Enterprise Linux 7.6 on the Undercloud node
- Install and configure the Undercloud node
- Prepare the Overcloud for deployment

- Deploy the Overcloud

5.1 Install Red Hat Enterprise Linux 7.6 on the Undercloud Node

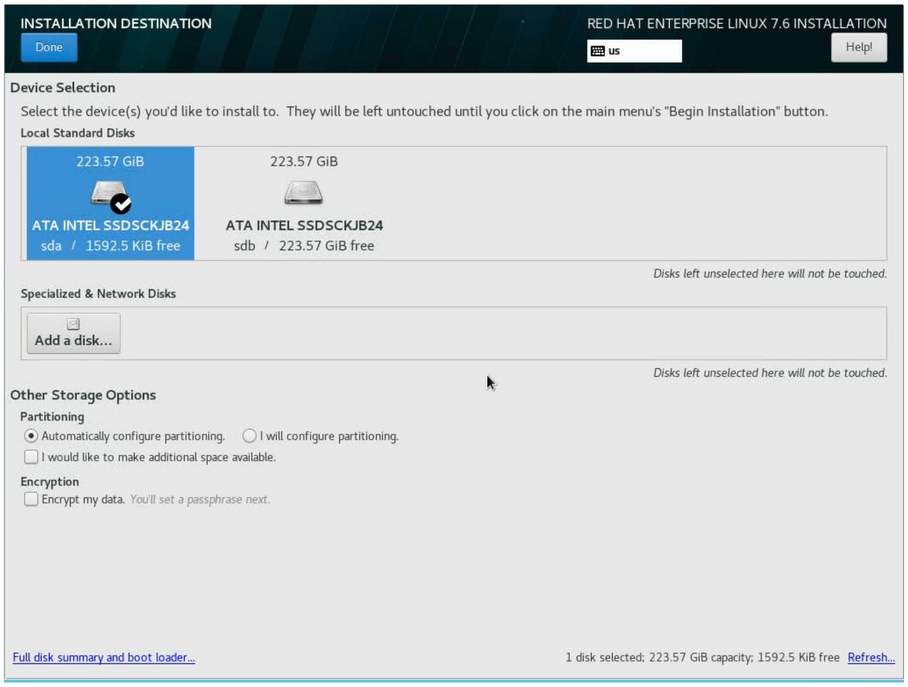
Red Hat supports deploying the Undercloud node on a VM or a bare-metal server. The resource requirements for each scenario are the same. The Undercloud node's resource requirements are specified here. Running the Undercloud node as a VM allows you to use the exact quantity of resources needed for the Undercloud as opposed to dedicating a bare-metal server, however this VM will still need to be run on a hypervisor external to the deployed cloud. For this deployment guide, the Undercloud was installed to a bare-metal node external to the solution.

Note: The Undercloud node is responsible for deployment and life-cycle management of Overcloud. Therefore, it is important that the Undercloud node is still accessible after Overcloud deployment.

Note: It's critical that the Undercloud node and the Overcloud nodes are on the same layer-2 segment with no other DHCP server running in the environment.

To install the operating system on the Undercloud node, complete the following steps:

1. Download Red Hat Enterprise Linux 7.6 from the Red Hat Customer Portal.
2. Use the out-of-band management IP of the NetApp HCI compute node and connect to the console session or use the VM console in VMware if Director is a VM.
3. Mount the Red Hat Enterprise Linux 7.6 ISO to the console and start the installation process for Undercloud.
4. Identify the root disks for the operating system.



Device Selection

Select the device(s) you'd like to install to. They will be left untouched until you click on the main menu's "Begin Installation" button.

Local Standard Disks

223.57 GiB	223.57 GiB
ATA INTEL SSDSCKJB24	ATA INTEL SSDSCKJB24
sda / 1592.5 KiB free	sdb / 223.57 GiB free

Disks left unselected here will not be touched.

Specialized & Network Disks

Disks left unselected here will not be touched.

Other Storage Options

Partitioning

[✓ Automatically configure partitioning.] I will configure partitioning.

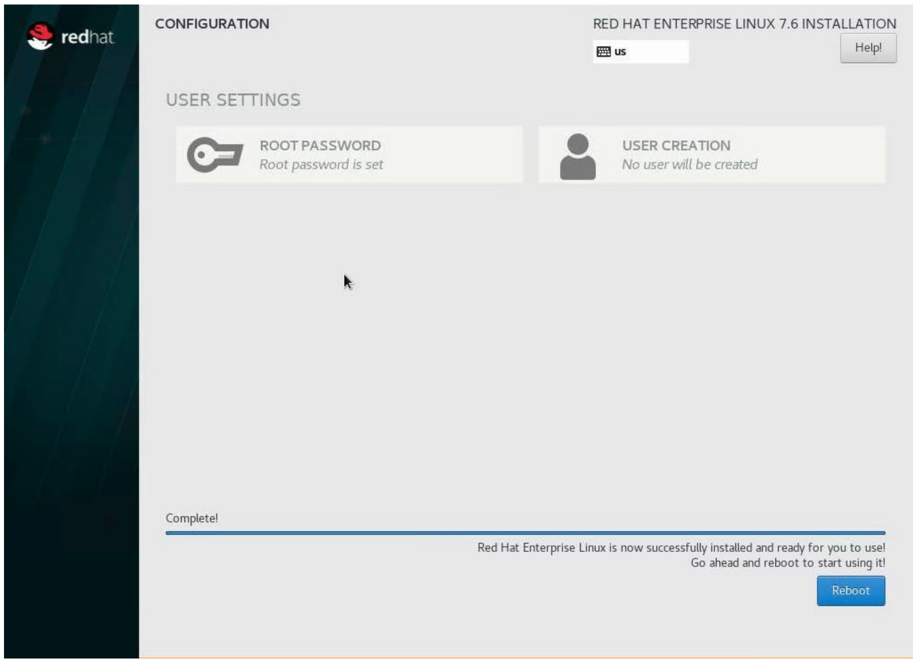
I would like to make additional space available.

Encryption

Encrypt my data. *You'll set a passphrase next.*

1 disk selected; 223.57 GiB capacity; 1592.5 KiB free

5. Provide a root password and complete Red Hat Enterprise Linux installation.
6. After successful installation, reboot the Undercloud node after deployment.


redhat
CONFIGURATION
RED HAT ENTERPRISE LINUX 7.6 INSTALLATION
us
Help!
USER SETTINGS
ROOT PASSWORD
Root password is set
USER CREATION
No user will be created
Complete!
Red Hat Enterprise Linux is now successfully installed and ready for you to use!
Go ahead and reboot to start using it!
Reboot

7. Configure the network interfaces on the Undercloud node.
8. After the system boots for the first time, verify that you can reach external entities. If you can't, fix the server's network configuration in the `/etc/sysconfig/network-scripts` directory before proceeding further.

```
[root@osp-director ~]# ping google.com -c 4
PING google.com (172.217.168.206) 56(84) bytes of data.
64 bytes from 172.217.168.206 (172.217.168.206): icmp_seq=1 ttl=49 time=88.3 ms
64 bytes from 172.217.168.206 (172.217.168.206): icmp_seq=2 ttl=49 time=88.4 ms
64 bytes from 172.217.168.206 (172.217.168.206): icmp_seq=3 ttl=49 time=88.3 ms
64 bytes from 172.217.168.206 (172.217.168.206): icmp_seq=4 ttl=49 time=88.4 ms

--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 88.331/88.392/88.464/0.368 ms
[root@osp-director ~]#
```

Note: Sample network interface files are provided in the appendix.

Configure Red Hat OpenStack Platform Director

To configure Red Hat OpenStack Platform Director, complete the following steps:

1. Create a stack user.

```
[root@osp-director ~]# useradd stack
```

2. Create a password for the stack user.

```
[root@osp-director ~]# passwd stack
Changing password for user stack.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@osp-director ~]#
```

3. Add the stack user to the sudoers group.

```
[root@osp-director ~]# echo "stack ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/stack
stack ALL=(root) NOPASSWD:ALL
```

```
[root@osp-director ~]# chmod 0440 /etc/sudoers.d/stack
[root@osp-director ~]#
```

4. Switch to the new stack user.

```
[root@osp-director ~]# su - stack
[stack@osp-director ~]$
```

5. Create directories for the images and templates.

```
[stack@osp-director ~]$ mkdir ~/templates
[stack@osp-director ~]$ mkdir ~/images
```

6. Set the FQDN for the Undercloud node.

```
[stack@osp-director ~]$ hostnamectl set-hostname --transient  osp-director.hci.rtp.com
[stack@osp-director ~]$
```

Verify the new FQDN.

```
[stack@osp-director ~]$ hostname -f
osp-director.hci.rtp.com
```

7. Director also requires the hostname and base hostname in `/etc/hosts` for public Undercloud API.

```
172.21.230.75 osp-director.hci.rtp.com osp-director
```

8. Register the Director node with Red Hat Subscription Manager.

```
[stack@osp-director ~]$ sudo subscription-manager register
Registering to: subscription.rhsm.redhat.com:443/subscription
Username: aborulka@netapp.com
Password:
The system has been registered with ID:
The registered system name is: osp-director.hci.rtp.com
```

8. Subscribe to relevant repositories.

```
[stack@osp-director ~]$ sudo subscription-manager list --available --all --matches="Red Hat
OpenStack"
+-------------------------------------------+
    Available Subscriptions
+-------------------------------------------+
Subscription Name:   Red Hat OpenStack Platform, Standard Support (4 Sockets, NFR, Partner Only)
Provides:            dotNET on RHEL Beta (for RHEL Server)
                     Red Hat CodeReady Linux Builder for x86_64
                     Red Hat Enterprise Linux FDIO Early Access (RHEL 7 Server)
                     Red Hat Ansible Engine
                     Red Hat Ceph Storage
                     Red Hat OpenStack Certification Test Suite
                     Red Hat Software Collections (for RHEL Server for IBM Power LE)
                     Red Hat Enterprise Linux Atomic Host Beta
                     Red Hat Enterprise Linux Fast Datapath
                     Red Hat OpenStack Beta
                     Red Hat CloudForms
                     Red Hat Software Collections Beta (for RHEL Server for IBM Power LE)
                     Red Hat Enterprise Linux Load Balancer (for RHEL Server)
                     Red Hat Beta
                     Red Hat Enterprise Linux Fast Datapath (for RHEL Server for IBM Power LE)
                     Red Hat Enterprise Linux High Availability for Power, little endian
                     Red Hat Enterprise Linux High Availability for x86_64
                     Red Hat Single Sign-On
                     dotNET on RHEL (for RHEL Server)
                     Red Hat Certification (for RHEL Server)
                     Red Hat Ceph Storage Calamari
                     Red Hat Developer Tools (for RHEL Server for IBM Power LE)
                     Red Hat OpenStack Beta Certification Test Suite
                     Red Hat CloudForms Beta
                     Red Hat Developer Tools Beta (for RHEL Server for IBM Power LE)
                     Red Hat Enterprise Linux High Availability (for IBM Power LE) - Extended
SKU:                 SER0505
```

```
Contract:             11867055
Pool ID:              valid-pool-id
Provides Management: No
Available:            10
Suggested:            1
Service Level:        Standard
Service Type:         L1-L3
Subscription Type:    Standard
Starts:               02/20/2019
Ends:                 02/19/2020
System Type:          Physical

[stack@osp-director ~]$
```

9. Attach the Red Hat OpenStack Platform 13 entitlement.

```
[stack@osp-director ~]$  sudo subscription-manager attach --pool=valid-pool-id
```

10. Disable default repositories and enable the required Red Hat Enterprise Linux repositories.

```
[stack@osp-director ~]$ sudo subscription-manager repos --disable=*
[stack@osp-director ~]$ sudo subscription-manager repos --enable=rhel-7-server-rpms --
enable=rhel-7-server-extras-rpms --enable=rhel-7-server-rh-common-rpms --enable=rhel-ha-for-rhel-
7-server-rpms --enable=rhel-7-server-openstack-13-rpms
```

11. Perform an update on the system to provide the latest repositories.

```
[stack@osp-director ~]$ sudo yum update -y
```

12. Reboot the Undercloud node.

```
[stack@osp-director ~]$ sudo reboot -y
```

13. Install and configure the NTP.

```
[stack@osp-director ~]$ sudo yum install ntp -y
```

14. Edit `/etc/ntp.conf` and modify the server parameter to match your local NTP server. Start the NTP server.

```
[stack@osp-director ~]$ sudo systemctl start ntpd
[stack@osp-director ~]$ sudo systemctl enable ntpd
```

5.2 Install and Configure the Undercloud Node

To install and configure the Undercloud node, complete the following steps:

1. Install the command line tools for Director installation and configuration.

```
[stack@osp-director ~]$ sudo yum install -y  python-tripleoclient
```

2. The Director installation process requires certain settings to determine your network configurations. The settings are stored in `undercloud.conf`. Copy the basic template to get started and determine which of the required settings need to be modified for your installation.

```
[stack@osp-director ~]$ cp /usr/share/instack-undercloud/undercloud.conf.sample \
> ~/undercloud.conf
[stack@osp-director ~]$
```

3. Make sure that the settings listed in Table 5 are configured in the `undercloud.conf`.

Table 5) Undercloud node settings.

Parameter	Value
`local_interface`	enp25s0f0
`masquerade`	True

Note: The `local_interface` corresponds to the provisioning network. Make sure that this network has the PXE VLAN as the native VLAN.

Note: Ensure that no other DHCP server is running on the PXE provisioning network.

Note: We use the default network settings for the inspection subnet and the deployment subnet for the Undercloud network.

4. Install Undercloud.

```
[stack@osp-director ~]$ openstack undercloud install
```

The installation runs for a few minutes and completes with the following message.

```
#############################################################################
Undercloud install complete.

The file containing this installation's passwords is at
/home/stack/undercloud-passwords.conf.

There is also a stackrc file at /home/stack/stackrc.

These files are needed to interact with the OpenStack services, and should be
secured.

#############################################################################

[stack@osp-director ~]$
```

5. Add the stack user to the Docker group to give access to the stack user for container management commands.

```
stack@osp-director ~]$ exec su -l stack
```

6. Initialize the `stack` user to use the command line tools.

```
[stack@osp-director ~]$ source ~/stackrc
```

Obtain Images for the Overcloud

To obtain the images for Overcloud, complete the following steps:

1. Install the `rhosp-director-images` and `rhosp-director-images-ipa` packages.

```
(undercloud) [stack@osp-director ~]$ sudo yum install rhosp-director-images rhosp-director-
images-ipa
```

2. Extract the images archives to the Images directory in `/home/stack/images.`

```
(undercloud) [stack@osp-director ~]$ cd ~/images
(undercloud) [stack@osp-director images]$ for i in /usr/share/rhosp-director-images/overcloud-
full-latest-13.0.tar /usr/share/rhosp-director-images/ironic-python-agent-latest-13.0.tar; do tar
-xvf $i; done
```

3. Import these images into Director.

```
(undercloud) [stack@director images]$ openstack overcloud image upload --image-path
/home/stack/images/
```

4. Verify that the images have uploaded successfully.

```
(undercloud) [stack@osp-director images]$ openstack image list
+--------------------------------------+------------------------+--------+
| ID                                   | Name                   | Status |
+--------------------------------------+------------------------+--------+
| d72c40d6-58e9-431d-9a88-bc97f425f5ed | bm-deploy-kernel       | active |
| a86ae335-b099-44e5-8691-6131088c9bf4 | bm-deploy-ramdisk      | active |
| d2c19107-22f3-4296-8c6c-00121fe71527 | overcloud-full         | active |
| c9bff49d-984e-4591-ba9f-136de761b514 | overcloud-full-initrd  | active |
| 83994e25-4b44-4575-94d6-30405e6ec347 | overcloud-full-vmlinuz | active |
+--------------------------------------+------------------------+--------+
(undercloud) [stack@osp-director images]$
```

5. The introspection images can be found at the following location:

```
(undercloud) [stack@osp-director images]$ ls -l /httpboot/
total 417380
-rwxr-xr-x. 1 root             root               6644016 May 28 12:06 agent.kernel
-rw-r--r--. 1 root             root             420737911 May 28 12:06 agent.ramdisk
-rw-r--r--. 1 ironic           ironic                 758 May 28 09:00 boot.ipxe
-rw-r--r--. 1 ironic-inspector ironic-inspector       470 May 28 08:54 inspector.ipxe
(undercloud) [stack@osp-director images]$
```

6. Set a nameserver for the control plane.

```
(undercloud) [stack@osp-director images]$ openstack subnet set --dns-nameserver 8.8.8.8
ctlplane-subnet
```

7. Verify that the nameserver is set.

```
(undercloud) [stack@osp-director images]$ openstack subnet show ctlplane-subnet
+-------------------+-----------------------------------------------------------+
| Field             | Value                                                     |
+-------------------+-----------------------------------------------------------+|
| dns_nameservers   | 8.8.8.8                                                   |
|
+-------------------+-----------------------------------------------------------+
(undercloud) [stack@osp-director images]$
```

Prepare Container Images for the Overcloud

The Undercloud uses the `docker-distribution` service as a registry. This allows Director to synchronize the images from `registry.access.redhat.com` and push them to the `docker-distribution` service. When creating the Overcloud, the Overcloud pulls the container images from the Undercloud `docker-distribution` registry. This method allows you to store a registry internally, which can speed up the deployment and reduce network congestion. However, Undercloud only acts as a basic registry and provides limited life-cycle management for container images.

1. In addition to the default images for the Overcloud service, you should also prepare images for Ironic and Octavia for provisioning bare-metal hosts and creating a load-balancing service respectively.

```
(undercloud) [stack@osp-director services-docker]$  openstack overcloud container image prepare \
>  -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
> -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/ironic.yaml \
> --namespace=registry.access.redhat.com/rhosp13 \
>   --push-destination=192.168.24.1:8787 \
>  --prefix=openstack- \
>   --tag-from-label {version}-{release} \
>   --output-env-file=/home/stack/templates/overcloud_images.yaml \
>   --output-images-file /home/stack/local_registry_images.yaml
```

2. Pull the container images from `registry.access.redhat.com.`

```
(undercloud) [stack@osp-director ~]$ openstack overcloud container image upload \
>   --config-file  /home/stack/local_registry_images.yaml \
>   --verbose
```

3. Verify that the images have been uploaded to the local registry.

```
(undercloud) [stack@osp-director ~]$ curl http://192.168.24.1:8787/v2/_catalog | jq
.repositories[]
```

5.3 Prepare the Overcloud for Deployment

The node definition template (`instackenv.json`) uses the JSON file format and contains the hardware and the IPMI details of Overcloud. The following file registers the Overcloud nodes:

```
{
    "nodes":[
        {
            "name":"controller-01",
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"ADMIN",
            "pm_password":"ADMIN",
            "pm_addr":"172.16.14.133"
        },
        {
            "name":"controller-02",
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"ADMIN",
            "pm_password":"ADMIN",
            "pm_addr":"172.16.14.134"
        },
        {
            "name":"controller-03",
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"ADMIN",
            "pm_password":"ADMIN",
            "pm_addr":"172.16.14.31"
        },
        {
            "name":"compute-01",
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"ADMIN",
            "pm_password":"ADMIN",
            "pm_addr":"172.16.14.131"
        }
    ]
}
```

4. Verify that the `pm_addr` corresponds to the IPMI address and that `pm_user` and `pm_password` correspond to the IPMI credentials and are accessible from the Undercloud node.
5. Import the Overcloud nodes.

```
(undercloud) [stack@osp-director ~]$ openstack overcloud node import ~/instack.json
```

6. View the list of newly registered nodes.

```
(undercloud) $ openstack baremetal node list
```

7. Introspect all the nodes.

```
(undercloud) $ openstack overcloud node introspect --all-manageable –provide
```

8. After a successful introspection, list all the bare-metal nodes.

```
(undercloud) [stack@osp-director ~]$ openstack baremetal node list
+--------------------------------------+---------------+---------------+-------------+-----------
---------+-------------+
| UUID                                 | Name          | Instance UUID | Power State |
Provisioning State | Maintenance |
+--------------------------------------+---------------+---------------+-------------+-----------
---------+-------------+
| f6e9dd9c-9d8e-424e-8057-70b624cd57ec | controller-01 | None          | power off   | available
| False       |
| 6a29c67a-84a1-440b-9890-6b504f133544 | controller-02 | None          | power off   | available
| False       |
| aff2e2b1-6885-4ea4-a264-e301664c144a | controller-03 | None          | power off   | available
| False       |
| a35ead30-8151-422e-940c-e007b71ed201 | compute-01    | None          | power off   | available
| False       |
+--------------------------------------+---------------+---------------+-------------+-----------
---------+-------------+
(undercloud) [stack@osp-director ~]$
```

9. Verify that the characteristics of the hardware nodes such as memory and CPU are correctly identified.

```
+------------------------+-----------------------------------------------------------------------
-------------------------------------------------------------------------------------------------
----------------------------------------------------+
| Field                  | Value
|
+------------------------+-----------------------------------------------------------------------
-------------------------------------------------------------------------------------------------
----------------------------------------------------+
| boot_interface         | pxe
|
| chassis_uuid           | None
|
| clean_step             | {}
|
| console_enabled        | False
|
| console_interface      | ipmitool-socat
|
| created_at             | 2019-05-28T21:17:45+00:00
|
| deploy_interface       | iscsi
|
| driver                 | ipmi
|
| driver_info            | {u'ipmi_password': u'******', u'ipmi_address': u'172.16.14.133',
u'deploy_ramdisk': u'a86ae335-b099-44e5-8691-6131088c9bf4', u'deploy_kernel': u'd72c40d6-58e9-
431d-9a88-bc97f425f5ed', u'ipmi_username': u'ADMIN'}          |
| driver_internal_info   | {}
|
| extra                  | {u'hardware_swift_object': u'extra_hardware-f6e9dd9c-9d8e-424e-8057-
70b624cd57ec'}
|
| inspect_interface      | inspector
|
| inspection_finished_at | None
|
| inspection_started_at  | None
|
| instance_info          | {}
|
| instance_uuid          | None
|
| last_error             | None
|
| maintenance            | False
|
| maintenance_reason     | None
|
| management_interface   | ipmitool
|
| name                   | controller-01
|
| network_interface      | flat
|
| power_interface        | ipmitool
|
| power_state            | power off
|
| properties             | {u'memory_mb': u'393216', u'cpu_arch': u'x86_64', u'local_gb': u'222',
u'cpus': u'32', u'capabilities':
u'cpu_vt:true,cpu_hugepages:true,boot_option:local,cpu_txt:true,cpu_aes:true,cpu_hugepages_1g:tru
e,boot_mode:bios'} |
| provision_state        | available
|
| provision_updated_at   | 2019-05-28T21:30:49+00:00
|
| raid_config            | {}
|
| raid_interface         | no-raid
|
| reservation            | None
|
| resource_class         | baremetal
|
| storage_interface      | noop
|
| target_power_state     | None
|
| target_provision_state | None
|
| target_raid_config     | {}
|
| updated_at             | 2019-05-28T21:31:00+00:00
|
| uuid                   | f6e9dd9c-9d8e-424e-8057-70b624cd57ec
|
| vendor_interface       | ipmitool
|
+------------------------+-------------------------------------------------------------------
-------------------------------------------------------------------------------------------------
-----------------------------------------------------+
(undercloud) [stack@osp-director ~]$
```

10. After registering and inspecting the hardware of each node, tag the nodes into the control and compute flavors.

```
(undercloud) [stack@osp-director ~]$ openstack baremetal node set --property
capabilities='profile:compute,boot_option:local' NODE-UUID
(undercloud) [stack@osp-director ~]$ openstack baremetal node set --property
capabilities='profile:control,boot_option:local' NODE-UUID
```

11. Verify that each node is appropriately tagged before starting your Overcloud deployment.

```
(undercloud) [stack@osp-director ~]$ openstack overcloud profiles list
+--------------------------------------+---------------+-----------------+-----------------+----+
| Node UUID                            | Node Name     | Provision State | Current Profile |
Possible Profiles |
+--------------------------------------+---------------+-----------------+-----------------+----+
| f6e9dd9c-9d8e-424e-8057-70b624cd57ec | controller-01 | active          | control         |
|
| 6a29c67a-84a1-440b-9890-6b504f133544 | controller-02 | active          | control         |
|
| aff2e2b1-6885-4ea4-a264-e301664c144a | controller-03 | active          | control         |
|
| a35ead30-8151-422e-940c-e007b71ed201 | compute-01    | active          | compute         |
|
+--------------------------------------+---------------+-----------------+-----------------+----+
```

5.4 Deploy the Overcloud

Overcloud is deployed through an automated process by referencing environment variables contained within the HOT. These files, formatted in YAML, contain the information needed to perform an initial deployment of the environment and modify or scale it in the future.

1. Download the HOT for Overcloud deployment. Extract the templates to `/home/stack/templates`.

Table 6) Types of templates.

Template Name	Template Purpose
`node-info.yaml`	Specifies the count and flavor of nodes
`roles_data.yaml`	Specifies the networks and services for each node
`network_data.yaml`	Specifies the DHCP pools and VLAN information for each network
`network-isolation.yaml`	Specifies the base HOT for network creation
`network-environment.yaml`	Specifies the network interfaces for Overcloud nodes
`octavia.yaml`	Default template that enables LBaaS
`octavia_timeouts.yaml`	Defines timeout values for LBaaS
`netapp-extra/netapp-allcontrollers-pre.yaml`	Predeployment configuration on all the controller nodes
`netapp-extra/netapp-swift.yaml`	Predeployment configuration on all the controller nodes to modify the IQN and mount LUNs for Swift
`netapp-extra/netapp-hci.yaml`	Sets up the Cinder backend for Overcloud
`netapp-extra/netapp-persist-mounts.yaml`	Supplies persistent mounting options for the Element volumes supporting the Swift Service.
`netapp-extra/post_config.yaml`	Runs a post config script to for all of the nodes.

Note: `node-info.yaml` and `roles_data.yaml` are autogenerated using TripleO.

2. Run the `deploy.sh` script provided with the other templates.

```
#!/bin/bash
openstack  overcloud deploy --templates \
--timeout 80 \
--stack overcloud \
-e /home/stack/templates/node-info.yaml \
-e /home/stack/templates/octavia.yaml \
-e /home/stack/templates/octavia_timeouts.yaml \
-r /home/stack/templates/roles_data.yaml \
-n /home/stack/templates/network_data.yaml \
-e /home/stack/templates/network-isolation.yaml \
-e /home/stack/templates/network-environment.yaml \
-e /home/stack/templates/netapp-extra/netapp-hci.yaml \
-e /home/stack/templates/netapp-extra/post-config.yaml
```

After a successful deployment of the Overcloud, an `overcloudrc` file is created that contains the admin credentials and the connection information to access the Overcloud.

```
 Stack overcloud CREATE_COMPLETE

Started Mistral Workflow tripleo.deployment.v1.get_horizon_url. Execution ID: 77a09b5c-8660-4d0e-
8235-346013f13fb7
Overcloud Endpoint: http://172.21.229.22:5000/
Overcloud Horizon Dashboard URL: http://172.21.229.22:80/dashboard
Overcloud rc file: /home/stack/overcloudrc
Overcloud Deployed
(undercloud) [stack@osp-director ~]$
```

This is the `overcloudrc` file from this deployment example.

Note: The highlighted portions show the admin URL and password.

```
(undercloud) [stack@osp-director ~]$ cat overcloudrc
# Clear any old environment that may conflict.
for key in $( set | awk '{FS="="}  /^OS_/ {print $1}' ); do unset $key ; done
export OS_NO_CACHE=True
export COMPUTE_API_VERSION=1.1
export OS_USERNAME=admin
export no_proxy=,172.21.229.22,192.168.24.9
export OS_USER_DOMAIN_NAME=Default
export OS_VOLUME_API_VERSION=3
export OS_CLOUDNAME=overcloud
export OS_AUTH_URL=http://172.21.229.22:5000//v3
export NOVA_VERSION=1.1
export OS_IMAGE_API_VERSION=2
export OS_PASSWORD=S0m3r@nd0m
export OS_PROJECT_DOMAIN_NAME=Default
export OS_IDENTITY_API_VERSION=3
export OS_PROJECT_NAME=admin
export OS_AUTH_TYPE=password
export PYTHONWARNINGS="ignore:Certificate has no, ignore:A true SSLContext object is not
available"

# Add OS_CLOUDNAME to PS1
if [ -z "${CLOUDPROMPT_ENABLED:-}" ]; then
    export PS1=${PS1:-""}
    export PS1=\${OS_CLOUDNAME:+"(\$OS_CLOUDNAME)"}\ $PS1
    export CLOUDPROMPT_ENABLED=1
fi
```

5.5 Additional Customization of the Overcloud

In addition to the validated deployment detailed in this document, there are several other OpenStack projects and features that can be configured for use prior to deployment to fully customize the Overcloud for each end user’s needs. Additional HOT template samples provided with OpenStack Platform can be referenced during the deployment to enable these services. While configuration of these additional services are beyond the scope of this deployment guide, some examples include the configuration of Barbican for encrypted key stores, and the enabling of SSL/TLS for the entire Overcloud, including the management plane.

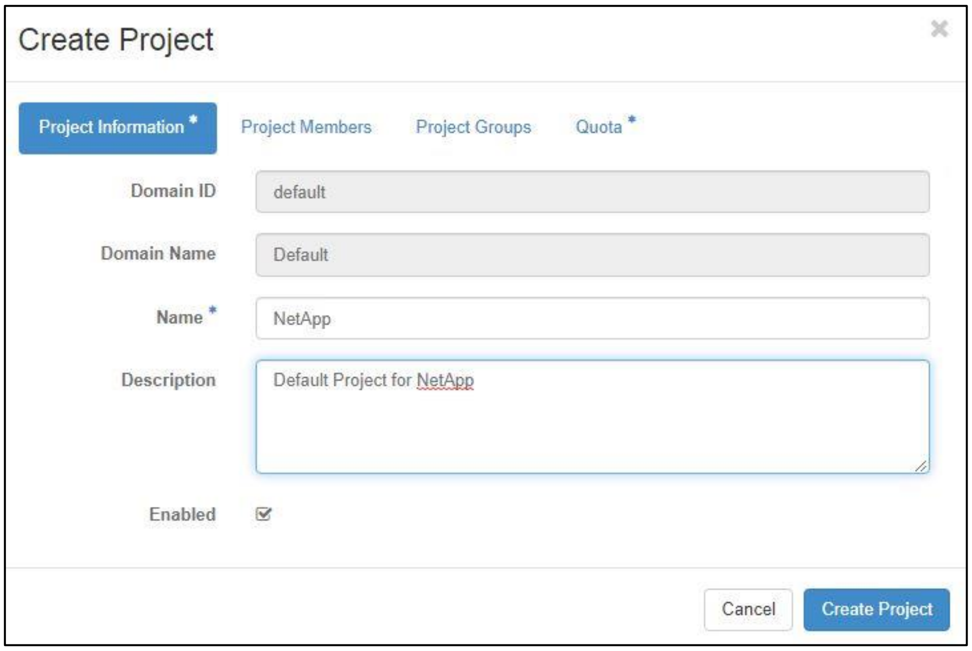
6 Solution Verification

This section demonstrates a suggested workflow for verifying operations of the cloud environment after the initial deployment has completed. It provides detailed instructions for suggested use cases including creating a project and a user account, creating flavors for virtual instances, uploading images to Glance to boot from, and lastly, configuring the networking and security groups for the instances to allow access to the outside world.

6.1 Create a Project and a User

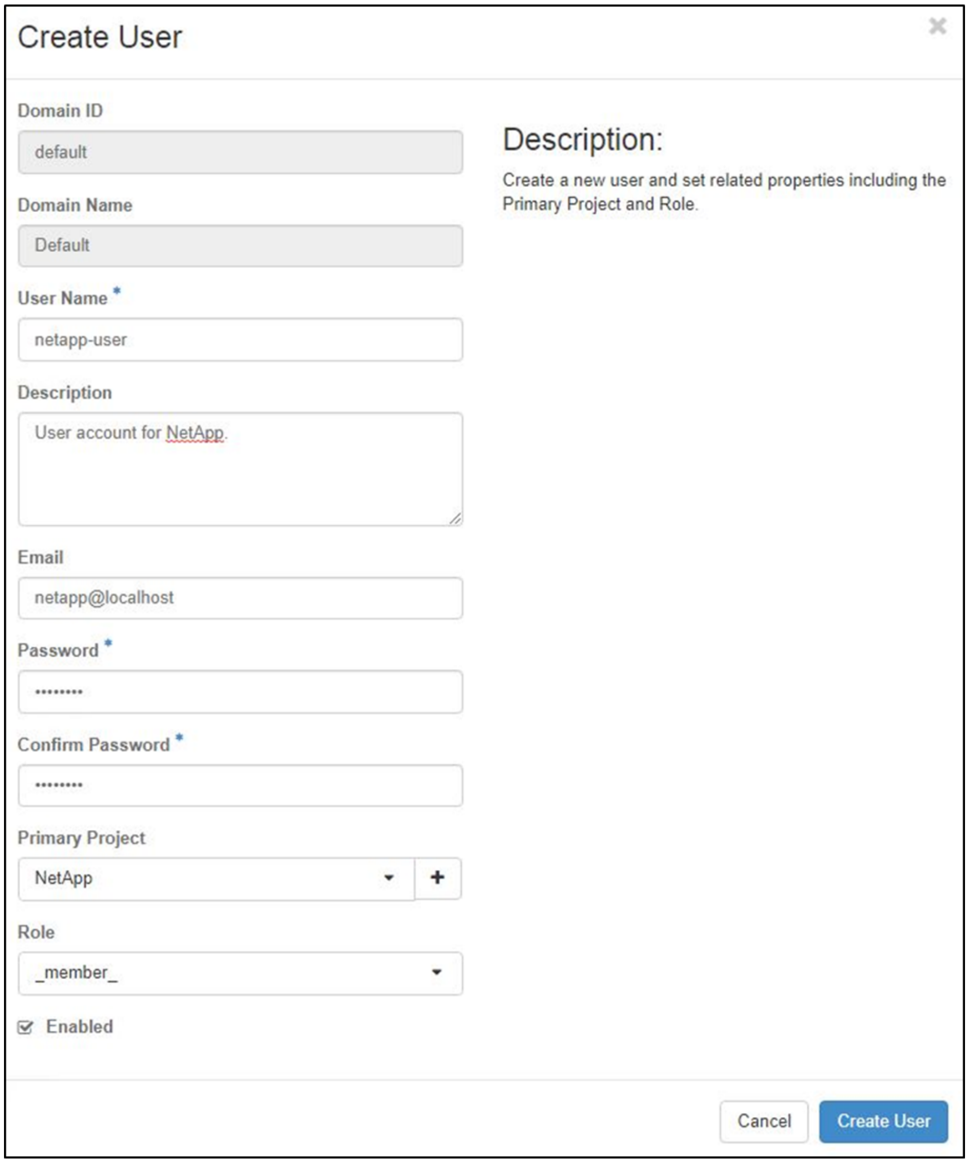
To create a project and a user, complete the following steps:

1. Log in into the Overcloud, using the URL and the access credentials specified in the `overcloudrc` file (highlighted above).
2. After you have logged into Overcloud, the administrator is presented with the Projects screen under the Identity tab. Projects are the OpenStack method of separating users and workloads into multiple tenants. Ideally, you would create a new project for each specific use case and the users that need access to it.
3. To create a project, click Create Project in the upper-right corner. This brings up the Create Project window in which you can add the name and description of the project, as well as enable or disable access to it. The other tabs here allow you perform the following tasks:
   – Manage project members
   – Create trusts with other projects as a project group
   – Modify the quotas for physical and virtual resources allowed to the instances created in this project


Create Project

Domain ID: default
Domain Name: Default
Name *: NetApp
Description: Default Project for NetApp
Enabled: ✓

4. After the project is created, visit the Users menu under the Identity tab to create a new user account. When you first click, notice each of the service accounts that already exist to run each of the independent services that make up the OpenStack cloud.
5. Click Create User to create a new user account.


Create User

Domain ID: default

Domain Name: Default

User Name *: netapp-user

Description: User account for NetApp.

Email: netapp@localhost

Password *: ••••••••

Confirm Password *: ••••••••

Primary Project: NetApp

Role: _member_

☑ Enabled

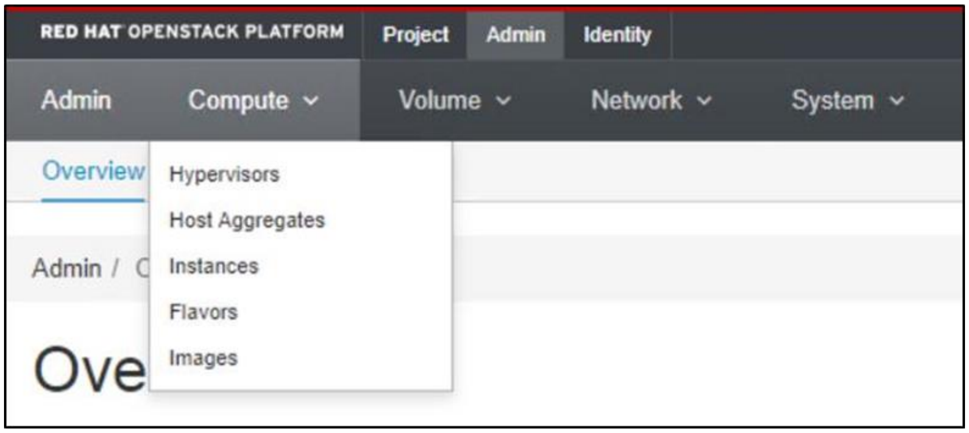
Description:

Create a new user and set related properties including the Primary Project and Role.

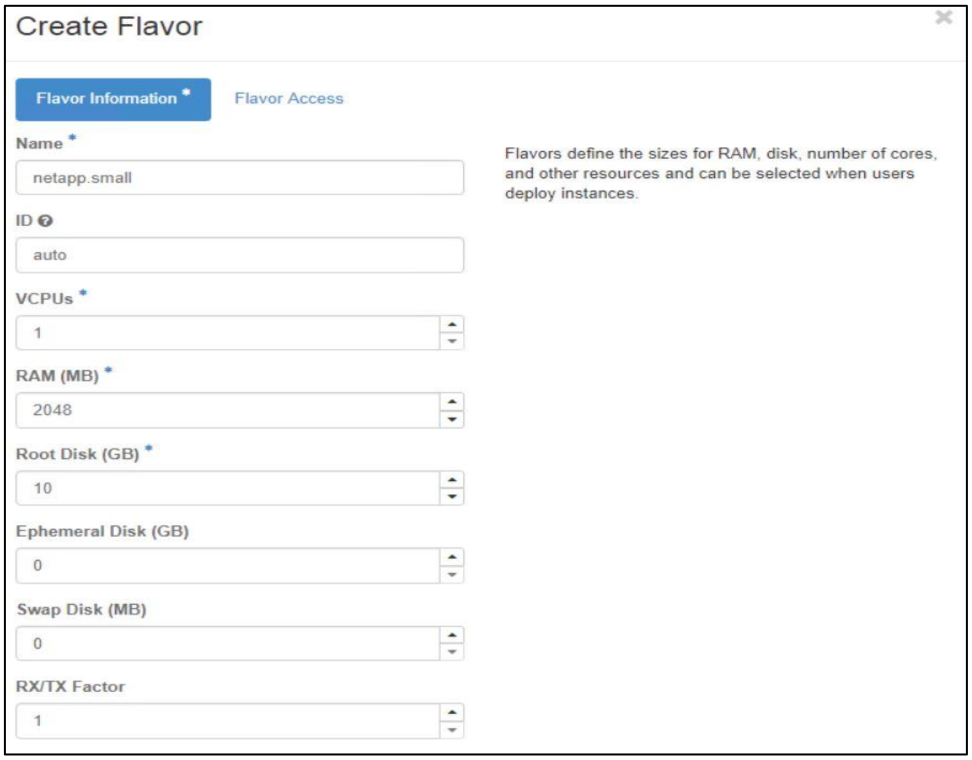
6.2 Create Flavors

Flavors are created in OpenStack to define the virtual resources available to each instance in a rapid, predefined template. Most cloud administrators define their flavors to be like those of the public cloud provider they use in hybrid deployments. Amazon AWS specifies a Small flavor with one vCPU and two GB of RAM. NetApp recommends creating a similar flavor and naming scheme for this OpenStack cloud.

1. To create a flavor, click the Compute menu under the Admin tab and select the Flavors drop-down menu.



2. Click Create Flavor to launch a wizard to name your flavor and select its basic resources, including vCPU; memory; and sizes for root, ephemeral, and swap disk.



Create Flavor

Flavor Information * | Flavor Access

Name * netapp.small

ID auto

VCPUs * 1

RAM (MB) * 2048

Root Disk (GB) * 10

Ephemeral Disk (GB) 0

Swap Disk (MB) 0

RX/TX Factor 1

Flavors define the sizes for RAM, disk, number of cores, and other resources and can be selected when users deploy instances.

6.3 Create Images

Click Create Image on the Images page to bring up a new window, which allows you to name the image and write a brief description. You can also choose the file and file-type to upload, and, if required, specify any special requirements for the machine to launch. You can decide if an image is public for all users in all projects or private to just their testing. You can also decide whether it's protected from deletion. The Metadata tab allows for extra customization of the image as well.

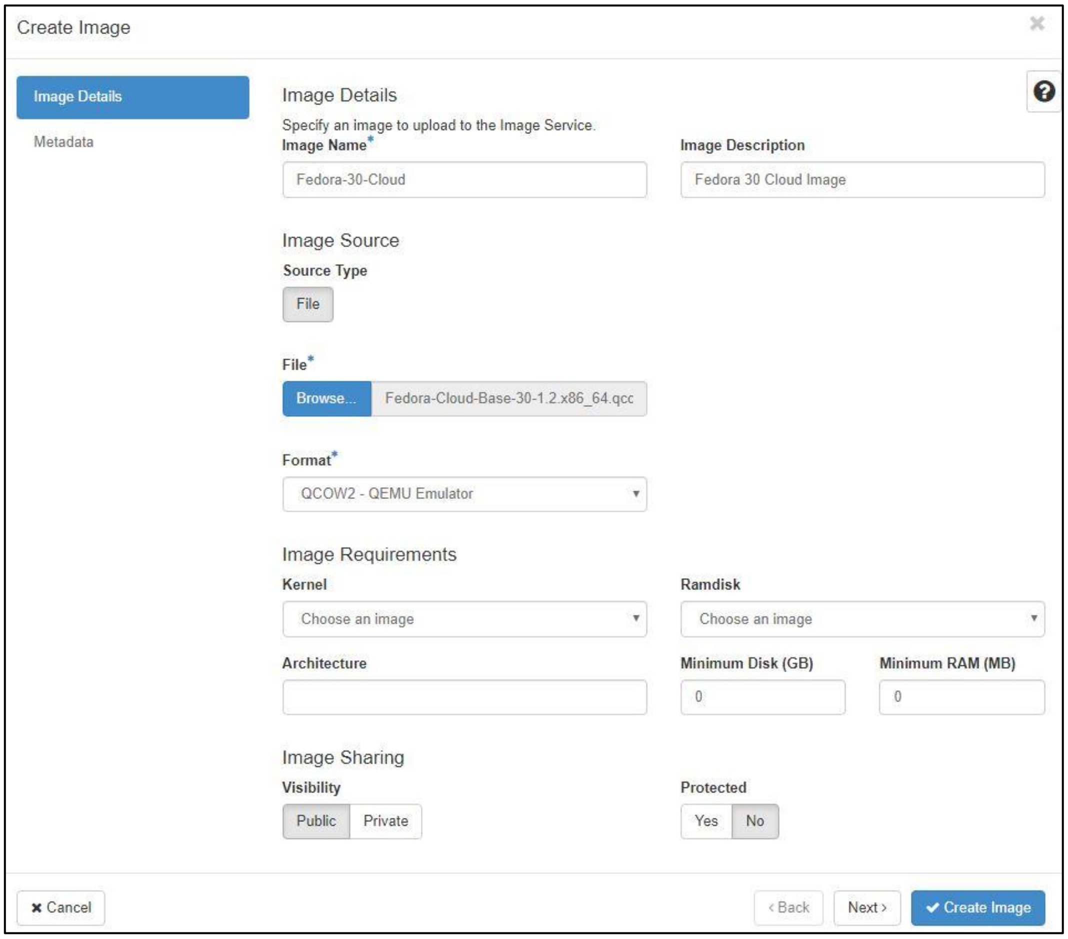

Create Image

Image Details
Metadata

Image Details

Specify an image to upload to the Image Service.

Image Name*: Fedora-30-Cloud

Image Description: Fedora 30 Cloud Image

Image Source

Source Type: File

File*: Browse... Fedora-Cloud-Base-30-1.2.x86_64.qcc

Format*: QCOW2 - QEMU Emulator

Image Requirements

Kernel: Choose an image

Ramdisk: Choose an image

Architecture:

Minimum Disk (GB): 0

Minimum RAM (MB): 0

Image Sharing

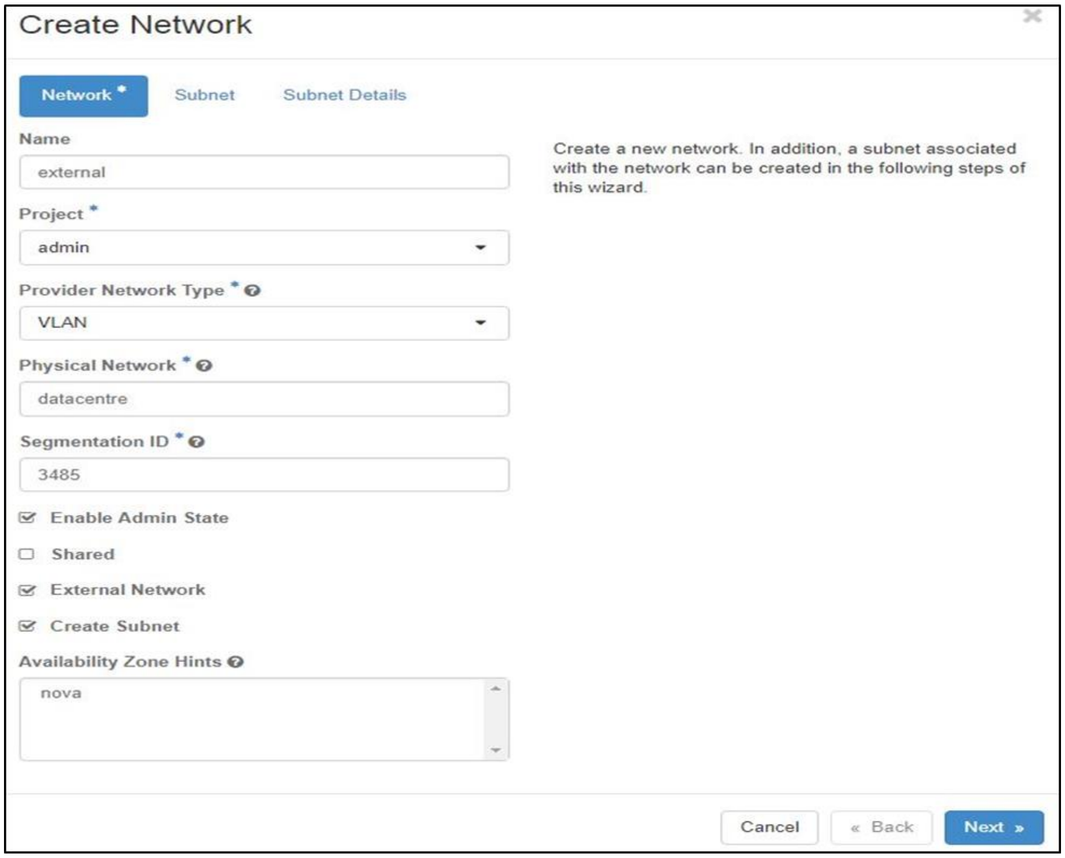
Visibility: [✓ Public] Private

Protected: Yes [✓ No]

6.4 Create an External Network and Floating IP Pool

External networks enable ingress and egress access to the Overcloud instances. OpenStack routers also use the external network to enable egress access from tenant networks. To create an external network and floating IP pool, complete the following steps:

1. Click Create Network to start the wizard and create the external network and its associated subnet. In this example, this network goes on our public VLAN 3485 that we used during deployment. You can select the network to be externally facing and enable it for use.


Create Network

Network * | Subnet | Subnet Details

Name: external

Project *: admin

Provider Network Type *: VLAN

Physical Network *: datacentre

Segmentation ID *: 3485

☑ Enable Admin State

☐ Shared

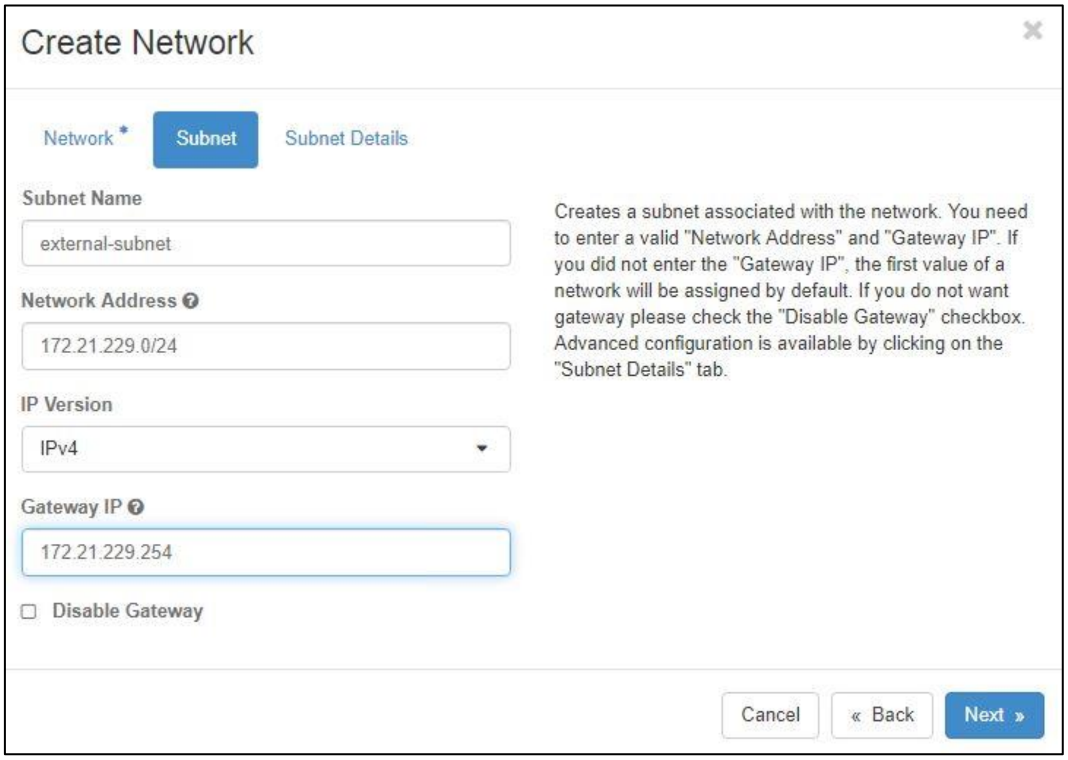
☑ External Network

☑ Create Subnet

Availability Zone Hints: nova

Create a new network. In addition, a subnet associated with the network can be created in the following steps of this wizard.

2. Clicking Next takes you to the menu to create a subnet to go with the newly created network. Here you can define your subnet name, the Classless Inter-Domain Routing (CIDR) for the network, and the gateway address it uses to communicate with the world.


Create Network

Subnet Name: external-subnet

Network Address: 172.21.229.0/24

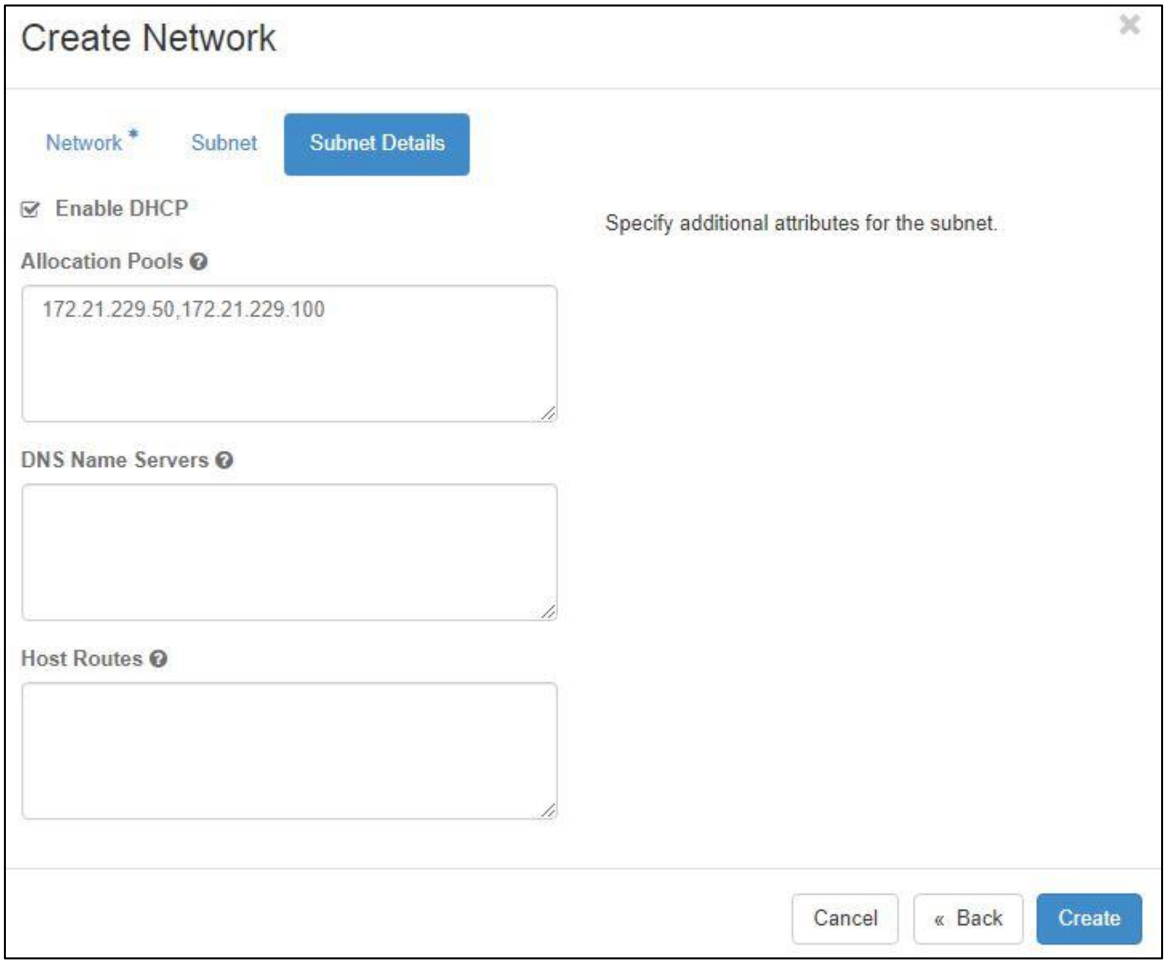
IP Version: IPv4

Gateway IP: 172.21.229.254

☐ Disable Gateway

Creates a subnet associated with the network. You need to enter a valid "Network Address" and "Gateway IP". If you did not enter the "Gateway IP", the first value of a network will be assigned by default. If you do not want gateway please check the "Disable Gateway" checkbox. Advanced configuration is available by clicking on the "Subnet Details" tab.

3. Click Next to configure the DHCP server for this network and the IP ranges available for use publicly as floating IP addresses for each virtual instance. You might be using the same public network for this as you did when deploying your cloud. If so, then you should verify that the DHCP range does not overlap with the service range provided in your `network-environment.yaml` file.



☑ Enable DHCP

Specify additional attributes for the subnet.

Allocation Pools: 172.21.229.50,172.21.229.100

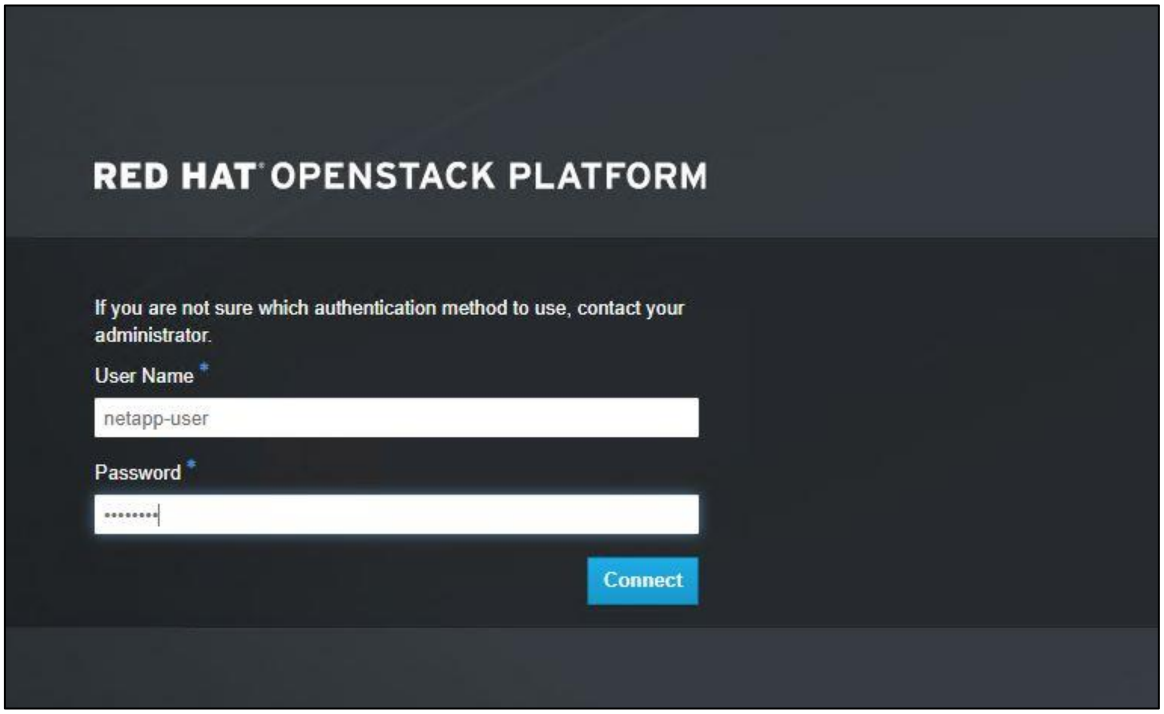
DNS Name Servers:

Host Routes:

6.5 Create a Router and Internal Network

To create a router and internal network, complete the following steps:

1. Log in to the Red Hat OpenStack Platform as the tenant user.


RED HAT® OPENSTACK PLATFORM

If you are not sure which authentication method to use, contact your administrator.

User Name *

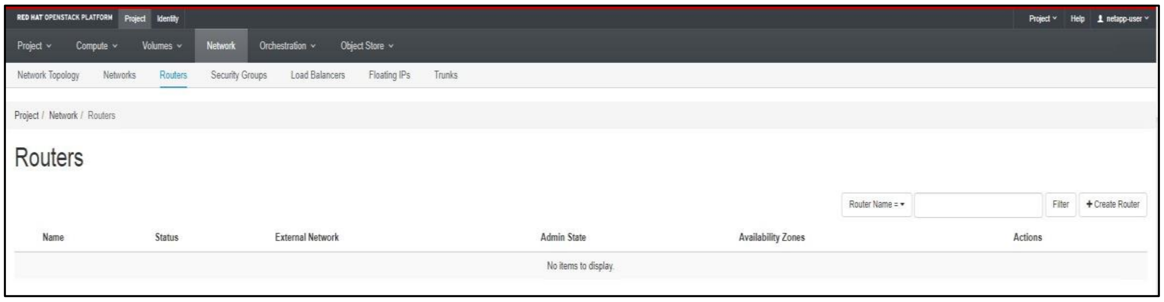
netapp-user

Password *

••••••••

Connect

2. Click the Network tab and then the Routers tab. Click Create Router.


RED HAT OPENSTACK PLATFORM | Project | Identity | Project | Help | netapp-user

Project | Compute | Volumes | Network | Orchestration | Object Store

Network Topology | Networks | Routers | Security Groups | Load Balancers | Floating IPs | Trunks

Project / Network / Routers

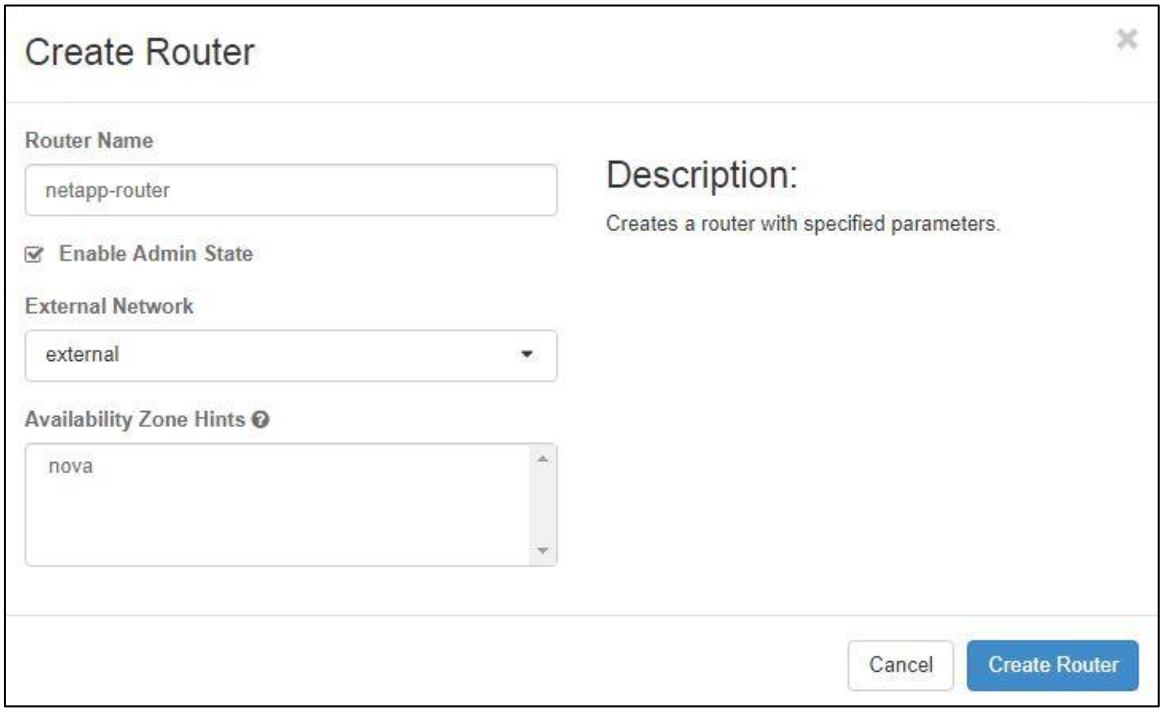
Routers

Router Name = | Filter | + Create Router

Name	Status	External Network	Admin State	Availability Zones	Actions

No items to display.

3. This starts a wizard that allows you to name your router, enable it, and select which external network it should connect to.


Create Router

Router Name

netapp-router

☑ Enable Admin State

External Network

external

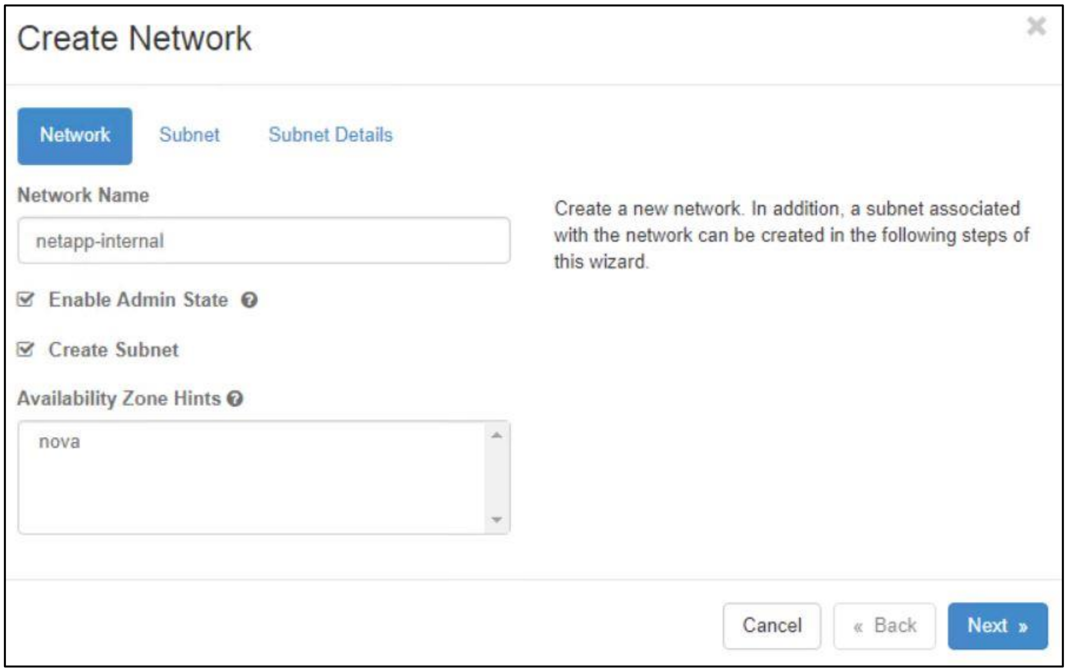
Availability Zone Hints

nova

Description:

Creates a router with specified parameters.

4. Next you can create a network for your project that allows it to connect to an external network. This is done by creating an internal network and connecting to your router. You can then select available IP addresses from your external network to be a part of a floating IP pool that can be assigned to each instance at boot. Since you are logged in as your project user, click on Networks menu. You should see your external network already available that was created by the administrator to be available globally. Click Create Network to create a new tenant network.
5. In the Create Network window, enter the network's name and enable it for use within your project. Click Next.


Create Network

Network Name

netapp-internal

☑ Enable Admin State

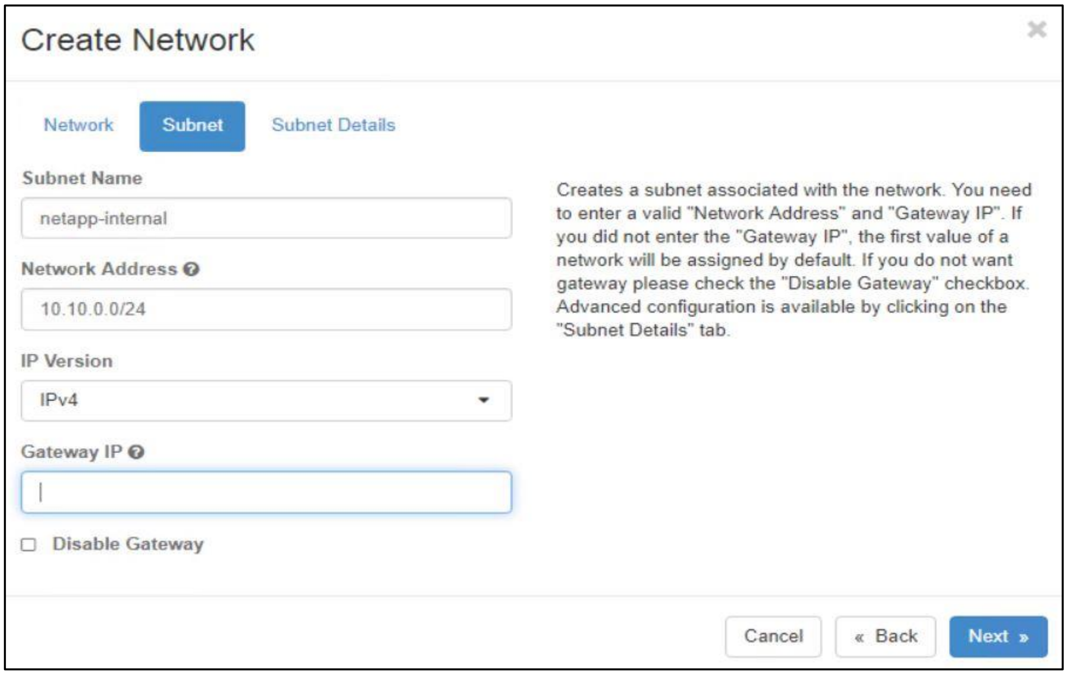
☑ Create Subnet

Availability Zone Hints

nova

Create a new network. In addition, a subnet associated with the network can be created in the following steps of this wizard.

6. In the Subnet window, name the subnet and select the CIDR for the network. In most cases this is a private network that only routes within the project. Click next.



Create Network

Network | Subnet | Subnet Details

Subnet Name: netapp-internal

Network Address: 10.10.0.0/24

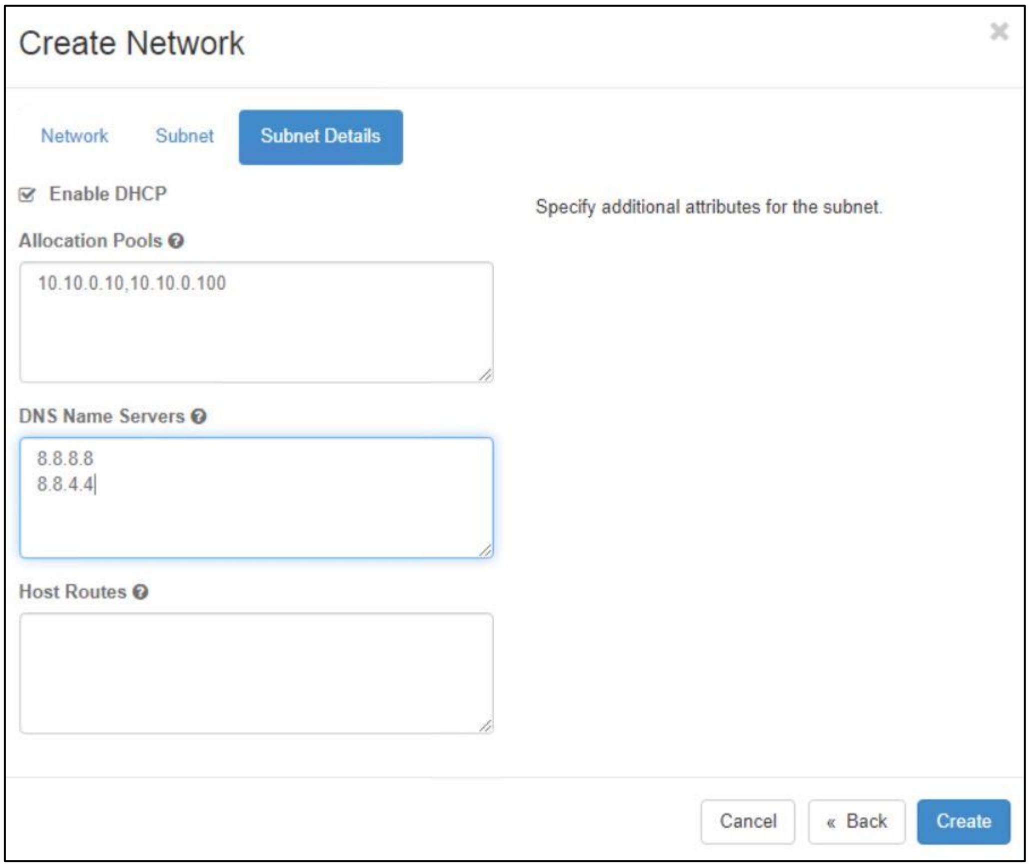
IP Version: IPv4

Gateway IP:

☐ Disable Gateway

Creates a subnet associated with the network. You need to enter a valid "Network Address" and "Gateway IP". If you did not enter the "Gateway IP", the first value of a network will be assigned by default. If you do not want gateway please check the "Disable Gateway" checkbox. Advanced configuration is available by clicking on the "Subnet Details" tab.

7. In the Subnet Details window, define the range of DHCP addresses available to the booted instances, and enter the preferred DNS servers.



Create Network

Network | Subnet | Subnet Details

☑ Enable DHCP

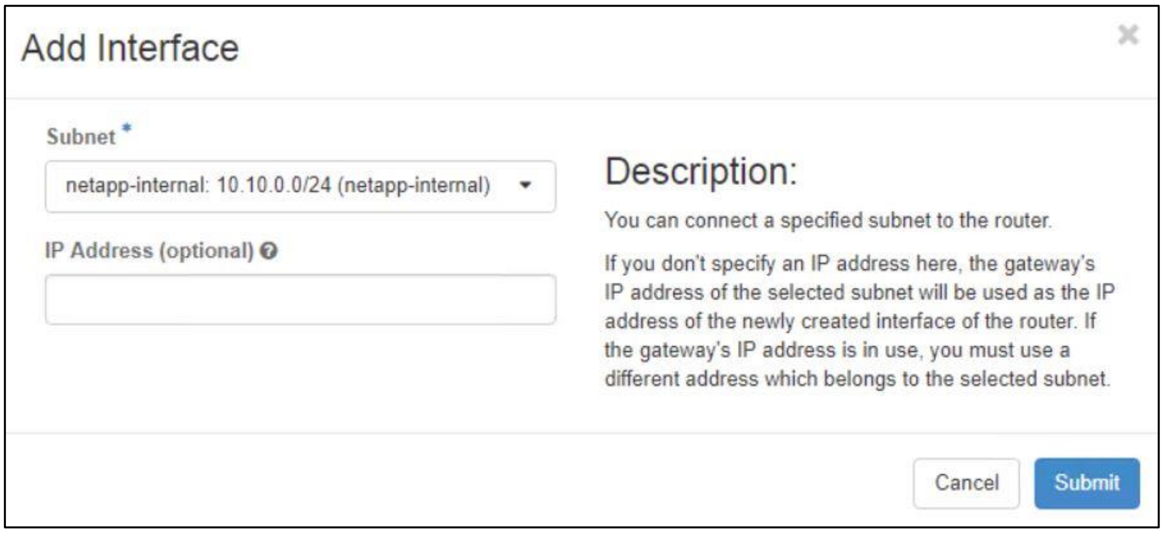
Allocation Pools: 10.10.0.10,10.10.0.100

DNS Name Servers:
8.8.8.8
8.8.4.4

Host Routes:

Specify additional attributes for the subnet.

8. After you have created the internal network, you must connect it to the router, which enables you to make connections to the external network. Click the Routers menu and select the netapp-router that you created in the previous step. Then click Add Interface. You can select the internal network you created from a drop-down menu to create the connection to the router.



Add Interface

Subnet *

netapp-internal: 10.10.0.0/24 (netapp-internal)

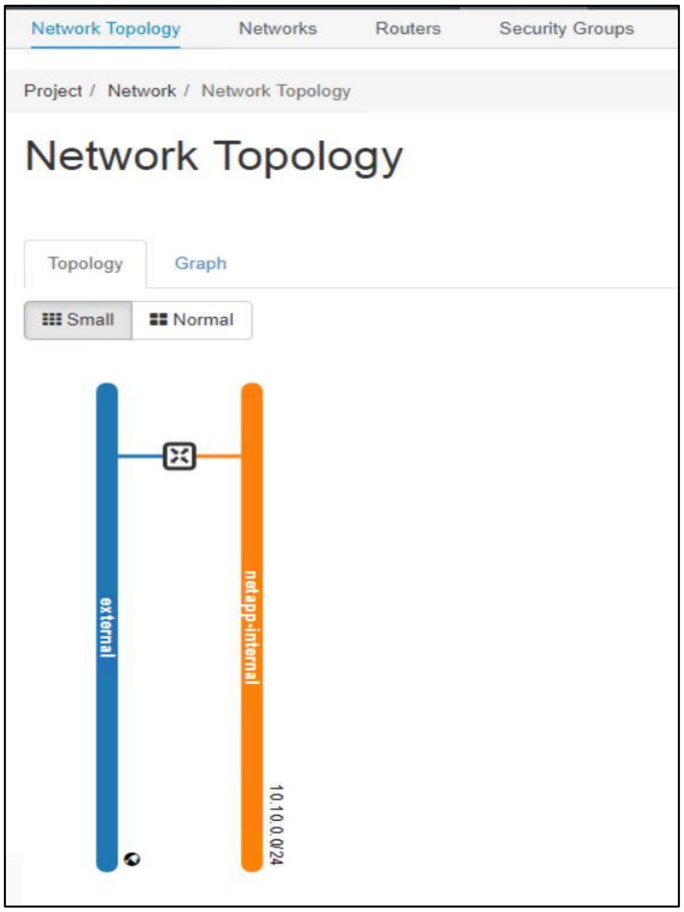
IP Address (optional)

Description:

You can connect a specified subnet to the router.

If you don't specify an IP address here, the gateway's IP address of the selected subnet will be used as the IP address of the newly created interface of the router. If the gateway's IP address is in use, you must use a different address which belongs to the selected subnet.

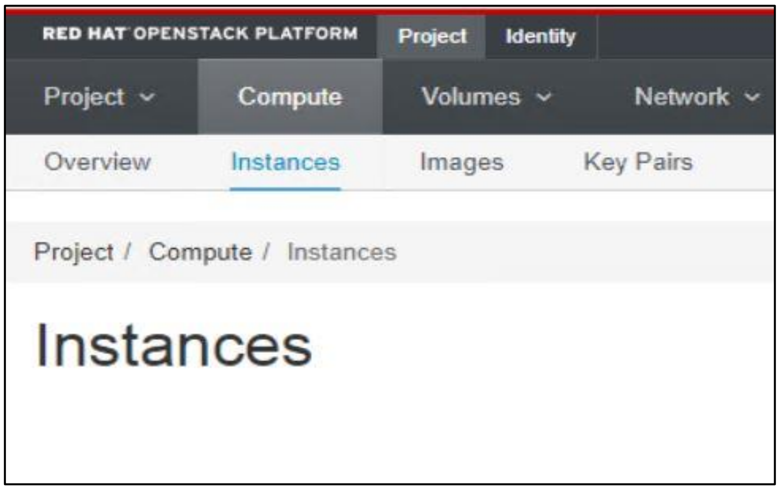
9. With this, you now have a connection between your internal and external networks. You can verify this by clicking on the Network Topology menu to verify that the two networks are connected by the router instance.



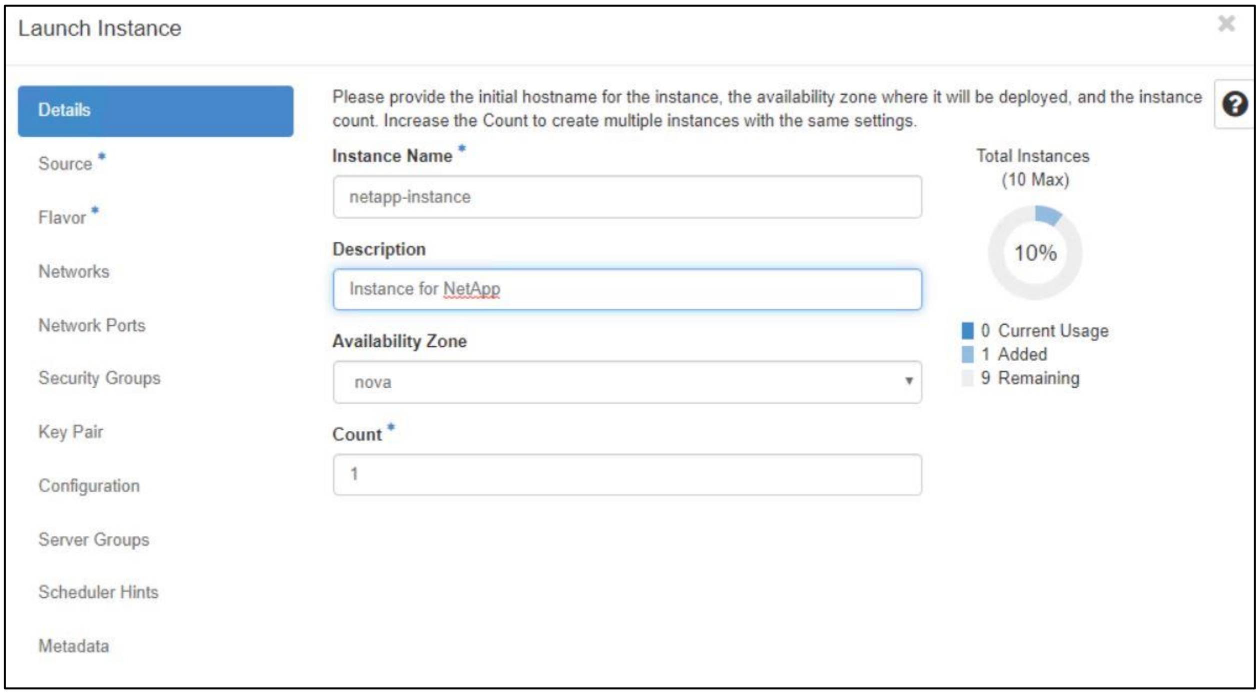
6.6 Boot an Instance from a Volume

To boot an instance from a volume, complete the following steps:

1. Now that your images are loaded, your network is configured, and your flavor is defined, you can launch your first virtual instance. Click the Compute menu and then the Instances heading.



2. Click Launch Instance to the right to start the wizard and deploy your first instance. The instance deployment wizard has multiple steps. First, name the instance and then tell the system how many instances to deploy. Required steps have an asterisk (*) next to them.



Please provide the initial hostname for the instance, the availability zone where it will be deployed, and the instance count. Increase the Count to create multiple instances with the same settings.

Instance Name *: netapp-instance

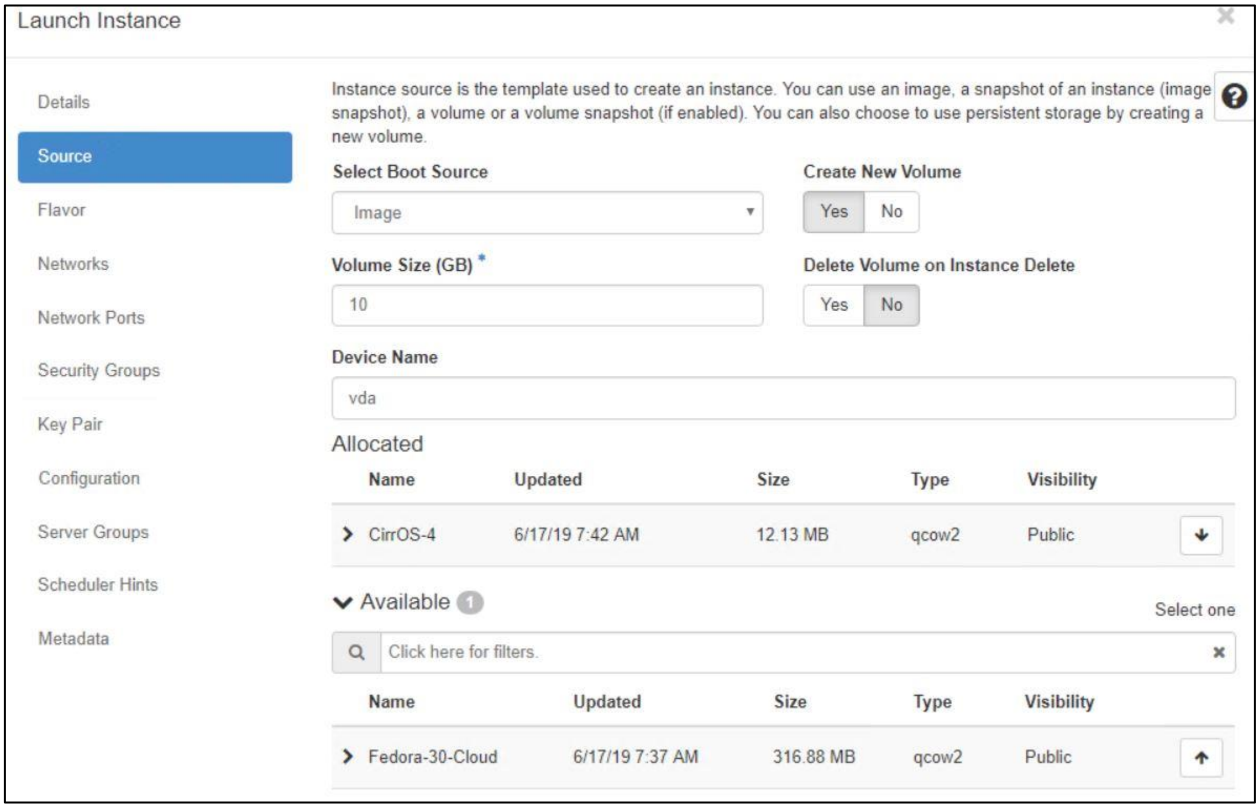
Description: Instance for NetApp

Availability Zone: nova

Count *: 1

0 Current Usage
1 Added
9 Remaining

3. Choose the source of our guest, which is usually from one of the images you uploaded in the previous step. Then set the size of the virtual disk needed to support the instance.


Launch Instance

Instance source is the template used to create an instance. You can use an image, a snapshot of an instance (image snapshot), a volume or a volume snapshot (if enabled). You can also choose to use persistent storage by creating a new volume.

Select Boot Source: Image

Create New Volume: [✓ Yes] No

Volume Size (GB) *: 10

Delete Volume on Instance Delete: Yes [✓ No]

Device Name: vda

Allocated

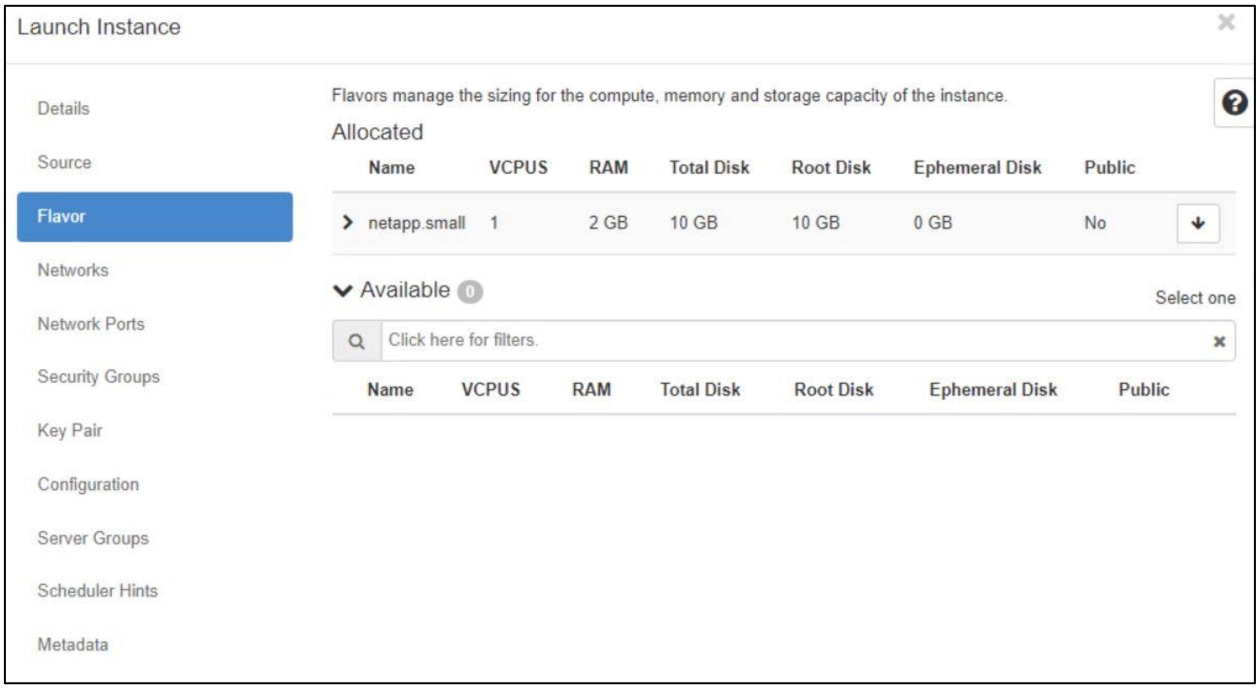
Name	Updated	Size	Type	Visibility
CirrOS-4	6/17/19 7:42 AM	12.13 MB	qcow2	Public

Available 1 — Select one

Click here for filters.

Name	Updated	Size	Type	Visibility
Fedora-30-Cloud	6/17/19 7:37 AM	316.88 MB	qcow2	Public

4. Select the flavor of instance that you would like to deploy. As previously discussed, the flavor defines the VM's resources.


Launch Instance

Flavors manage the sizing for the compute, memory and storage capacity of the instance.

Allocated

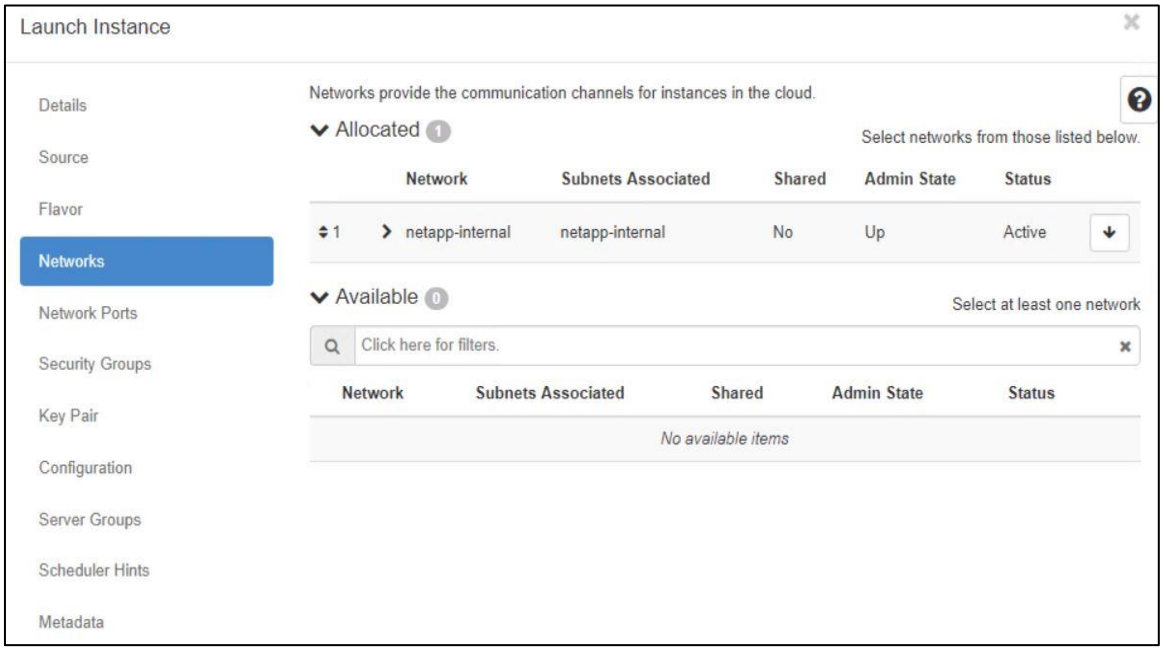
Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
netapp.small	1	2 GB	10 GB	10 GB	0 GB	No

Available 0 — Select one

Click here for filters.

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public

5. Choose the network that the instance connects to. In this example, the `netapp-internal` network is the only one available for the project, so it is selected by default.



Networks provide the communication channels for instances in the cloud.

Allocated 1 — Select networks from those listed below.

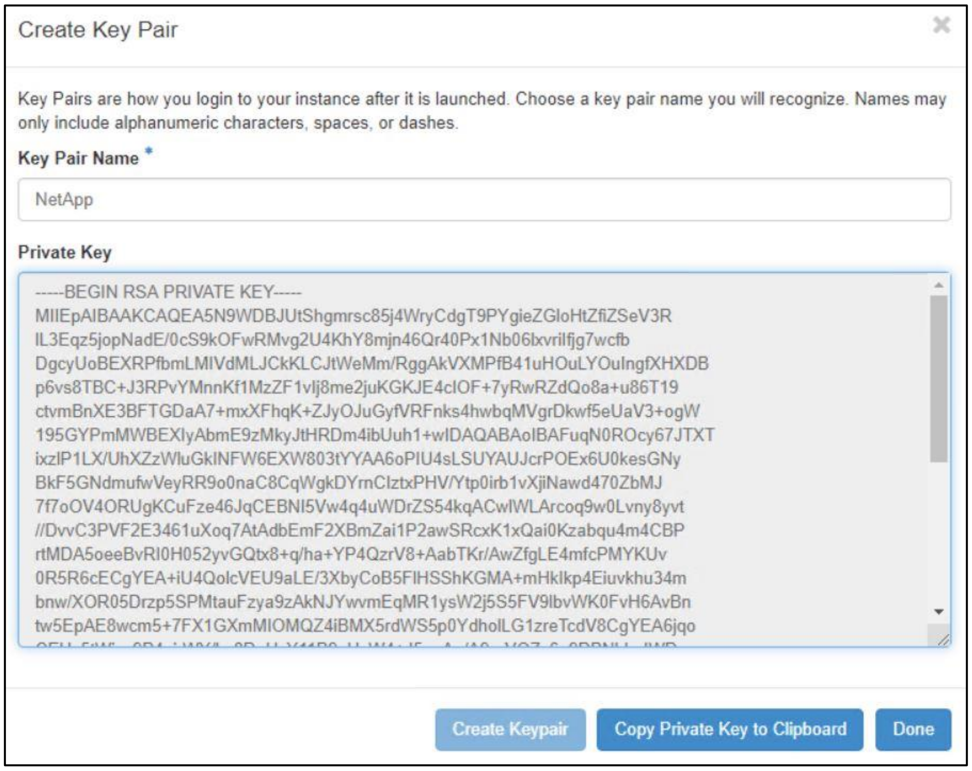
	Network	Subnets Associated	Shared	Admin State	Status
1	netapp-internal	netapp-internal	No	Up	Active

Available 0 — Select at least one network

Click here for filters.

Network	Subnets Associated	Shared	Admin State	Status
No available items				

6. Verify that you can access the deployed instance. If the instances does not permit local authentication, then you must create or upload a private key so that you can connect to your instance after it is deployed. Click the Key Pair menu to do this.



Key Pairs are how you login to your instance after it is launched. Choose a key pair name you will recognize. Names may only include alphanumeric characters, spaces, or dashes.

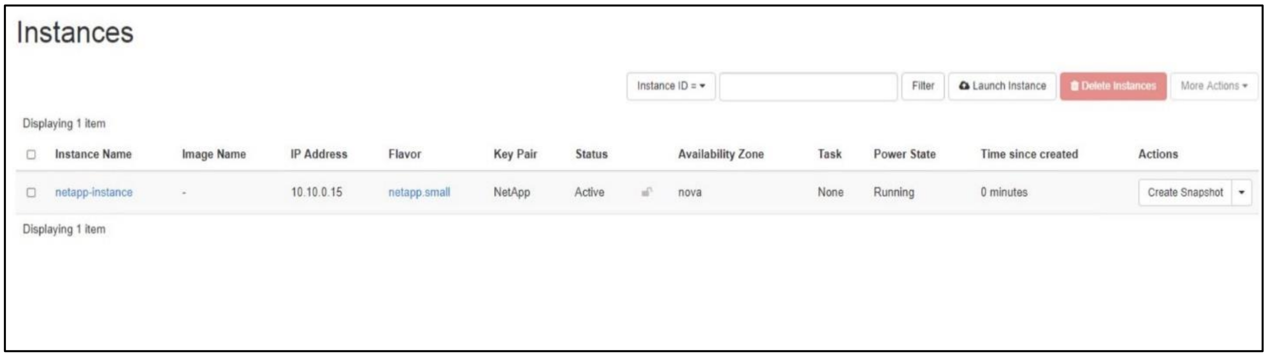
Key Pair Name *

NetApp

Private Key

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA5N9WDBJUtShgmrsc85j4WryCdgT9PYgieZGloHtZfiZSeV3R
IL3Eqz5jopNadE/0cS9kOFwRMvg2U4KhY8mjn46Qr40Px1Nb06lxvrilfjg7wcfb
DgcyUoBEXRPfbmLMIVdMLJCkKLCJtWeMm/RggAkVXMPfB41uHOuLYOuIngfXHXDB
p6vs8TBC+J3RPvYMnnKf1MzZF1vlj8me2juKGKJE4cIOF+7yRwRZdQo8a+u86T19
ctvmBnXE3BFTGDaA7+mxXFhqK+ZJyOJuGyfVRFnks4hwbqMVgrDkwf5eUaV3+ogW
195GYPmMWBEXlyAbmE9zMkyJtHRDm4ibUuh1+wIDAQABAoIBAFuqN0ROcy67JTXT
ixzlP1LX/UhXZzWluGkINFW6EXW803tYYAA6oPIU4sLSUYAUJcrPOEx6U0kesGNy
BkF5GNdmufwVeyRR9o0naC8CqWgkDYrnClztxPHV/Ytp0irb1vXjiNawd470ZbMJ
7f7oOV4ORUgKCuFze46JqCEBNI5Vw4q4uWDrZS54kqACwlWLArcoq9w0Lvny8yvt
//DvvC3PVF2E3461uXoq7AtAdbEmF2XBmZai1P2awSRcxK1xQai0Kzabqu4m4CBP
rtMDA5oeeBvRI0H052yvGQtx8+q/ha+YP4QzrV8+AabTKr/AwZfgLE4mfcPMYKUv
0R5R6cECgYEA+iU4QolcVEU9aLE/3XbyCoB5FIHSShKGMA+mHkIkp4Eiuvkhu34m
bnw/XOR05Drzp5SPMtauFzya9zAkNJYwvmEqMR1ysW2j5S5FV9lbvWK0FvH6AvBn
tw5EpAE8wcm5+7FX1GXmMIOMQZ4iBMX5rdWS5p0YdholLG1zreTcdV8CgYEA6jqo
```

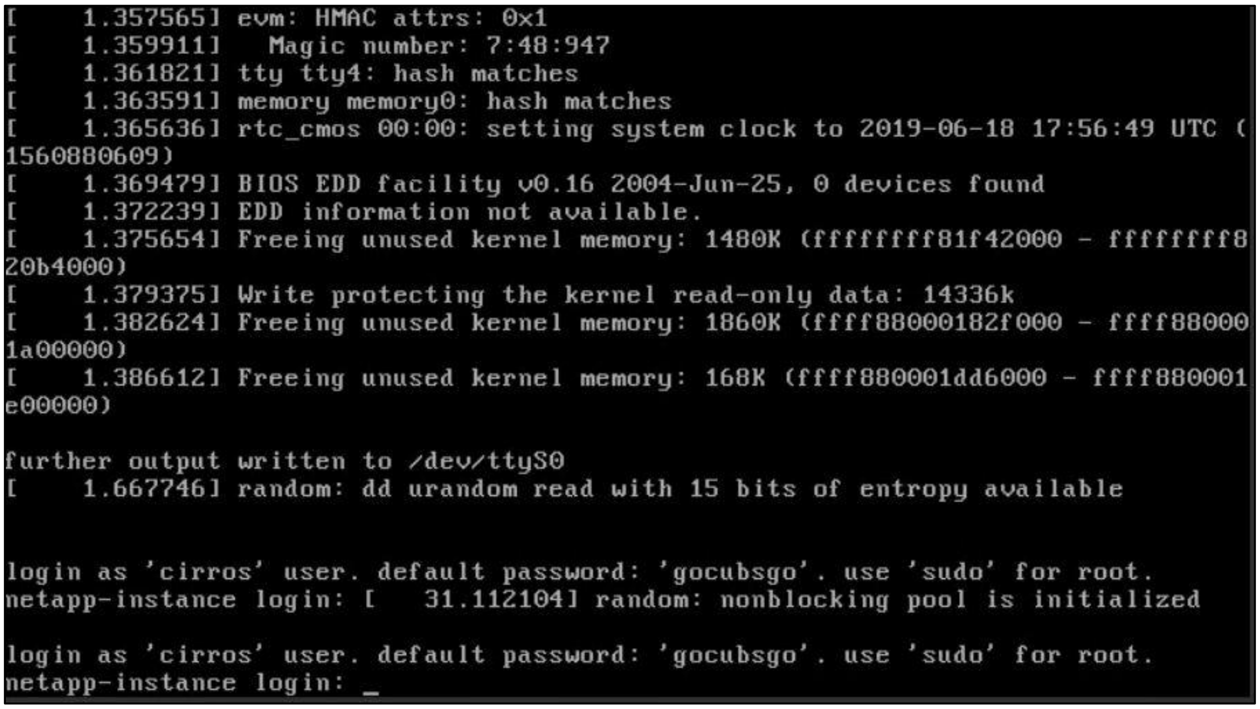
7. After creating the Key Pair, you are ready to launch your instance. Click Launch Instance. The newly launched instance is assigned an IP address from the internal pool.


Instances

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
netapp-instance	-	10.10.0.15	netapp.small	NetApp	Active	nova	None	Running	0 minutes	Create Snapshot

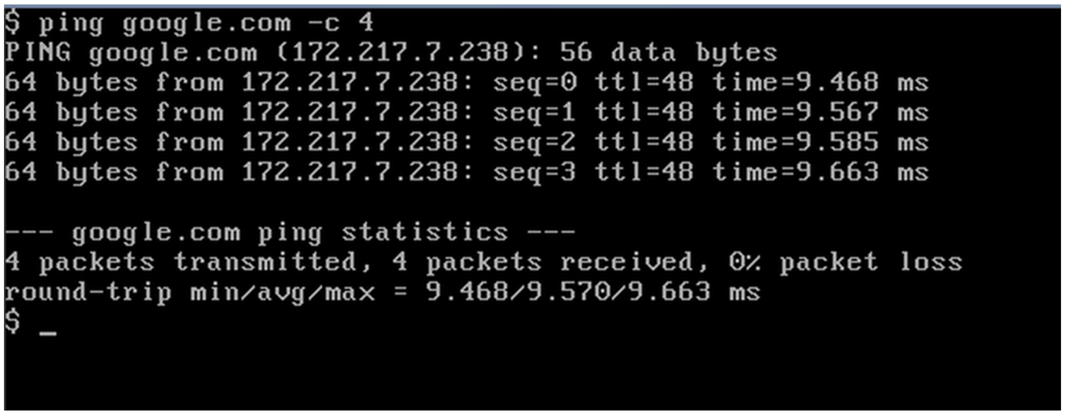
Displaying 1 item

8. The drop-down menu on the right presents various options for interacting with the instance, and you can launch a console session to confirm that the machine is running as expected.


```
[    1.357565] evm: HMAC attrs: 0x1
[    1.359911]   Magic number: 7:48:947
[    1.361821] tty tty4: hash matches
[    1.363591] memory memory0: hash matches
[    1.365636] rtc_cmos 00:00: setting system clock to 2019-06-18 17:56:49 UTC (
1560880609)
[    1.369479] BIOS EDD facility v0.16 2004-Jun-25, 0 devices found
[    1.372239] EDD information not available.
[    1.375654] Freeing unused kernel memory: 1480K (ffffffff81f42000 - ffffffff8
20b4000)
[    1.379375] Write protecting the kernel read-only data: 14336k
[    1.382624] Freeing unused kernel memory: 1860K (ffff88000182f000 - ffff88000
1a00000)
[    1.386612] Freeing unused kernel memory: 168K (ffff880001dd6000 - ffff880001
e00000)

further output written to /dev/ttyS0
[    1.667746] random: dd urandom read with 15 bits of entropy available


login as 'cirros' user. default password: 'gocubsgo'. use 'sudo' for root.
netapp-instance login: [   31.112104] random: nonblocking pool is initialized

login as 'cirros' user. default password: 'gocubsgo'. use 'sudo' for root.
netapp-instance login: _
```

6.7 Verify Egress Connectivity from the Host

To verify egress connectivity from the host, attempt to ping netapp.com from your running instance. The following screenshot shows the expected ping response:



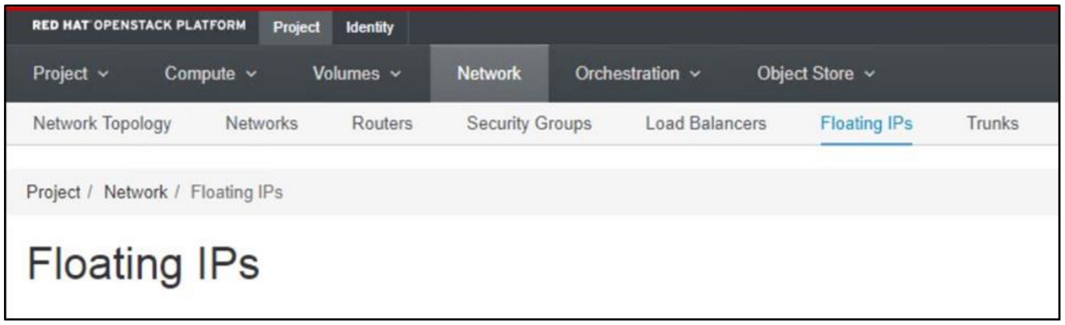
```
$ ping google.com -c 4
PING google.com (172.217.7.238): 56 data bytes
64 bytes from 172.217.7.238: seq=0 ttl=48 time=9.468 ms
64 bytes from 172.217.7.238: seq=1 ttl=48 time=9.567 ms
64 bytes from 172.217.7.238: seq=2 ttl=48 time=9.585 ms
64 bytes from 172.217.7.238: seq=3 ttl=48 time=9.663 ms

--- google.com ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 9.468/9.570/9.663 ms
$ _
```

6.8 Associate a Floating IP and Connect to the Instance

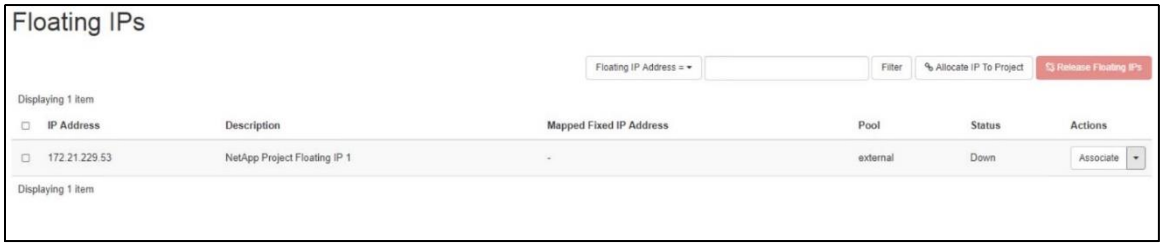
Floating IP addresses are ephemeral in nature and can be moved from one virtual instance to another without issue. They are derived from the pool of addresses that were created with the external network in the previous steps.

To associate a floating IP from the pool to the project, complete the following steps:

1. Click the Network tab, and then click the Floating IPs menu.



2. Select Allocate IP To Project to start the wizard.

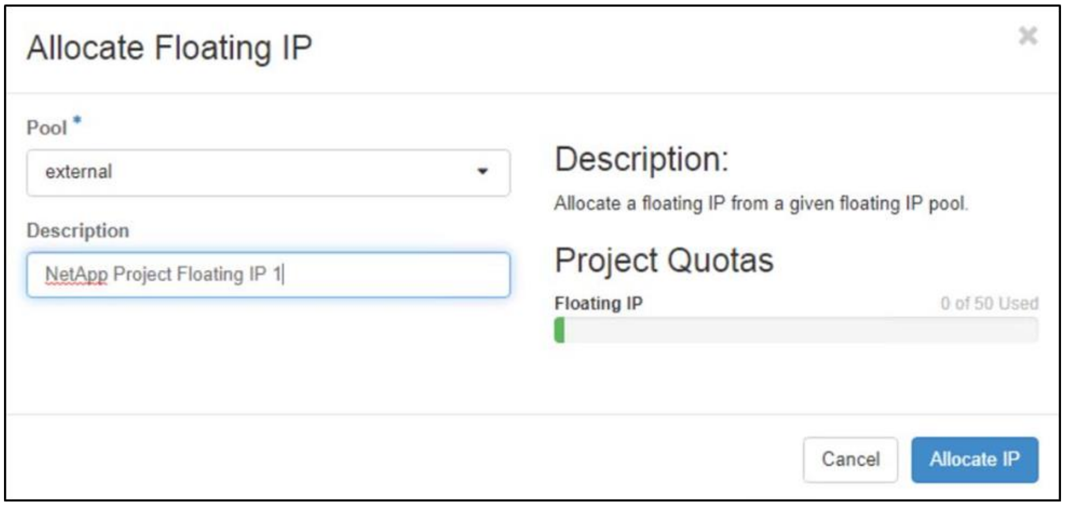


Displaying 1 item

	IP Address	Description	Mapped Fixed IP Address	Pool	Status	Actions
☐	172.21.229.53	NetApp Project Floating IP 1	-	external	Down	Associate

Displaying 1 item

3. Choose the pool and write a description for the IP address. Click Allocate IP to generate an IP address from the pool.


Allocate Floating IP

Pool * external

Description NetApp Project Floating IP 1

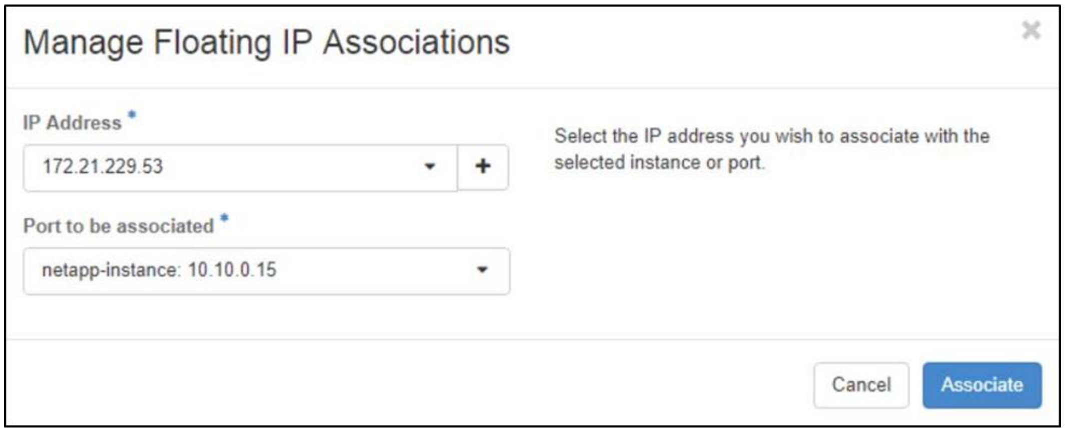
Description:

Allocate a floating IP from a given floating IP pool.

Project Quotas

Floating IP 0 of 50 Used

4. After deriving an IP from your pool, you must associate it with the virtual instance. Click Associate. A new wizard starts that shows the new floating IP address and allows you to select the running instance from the drop-down Port to be Associated menu.

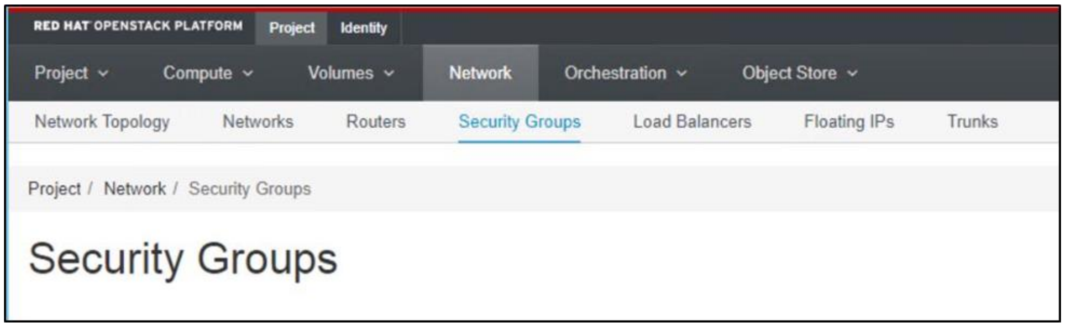

Manage Floating IP Associations

IP Address * 172.21.229.53

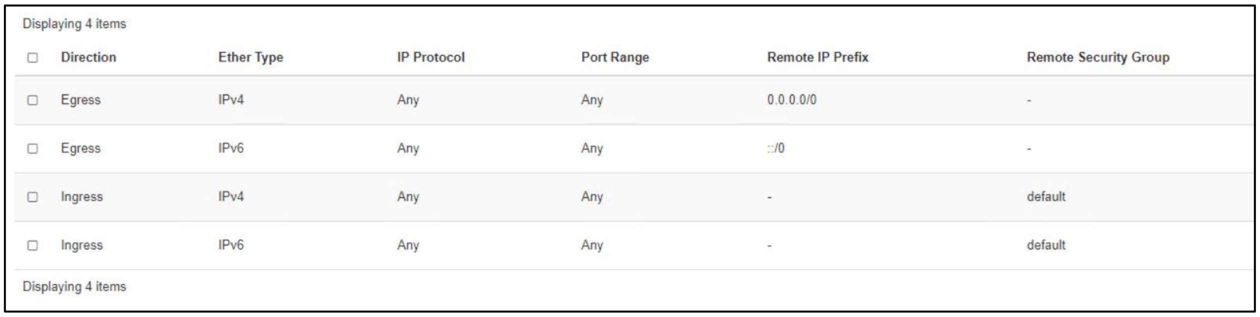
Port to be associated * netapp-instance: 10.10.0.15

Select the IP address you wish to associate with the selected instance or port.

5. The instance is now associated with a public-facing IP from the floating IP pool, but attempts to connect or verify connectivity via ping will be refused. To remedy this, click the Security Groups menu under the Network tab.



6. The default security policy provided by a fresh OpenStack deployment allows for all traffic to egress the network, but it blocks all ingress traffic. Because the new instance is using this policy, you cannot connect. Instead, you must modify this policy to allow both ICMP ping and SSH. Click Manage Rules to see the currently defined rules.

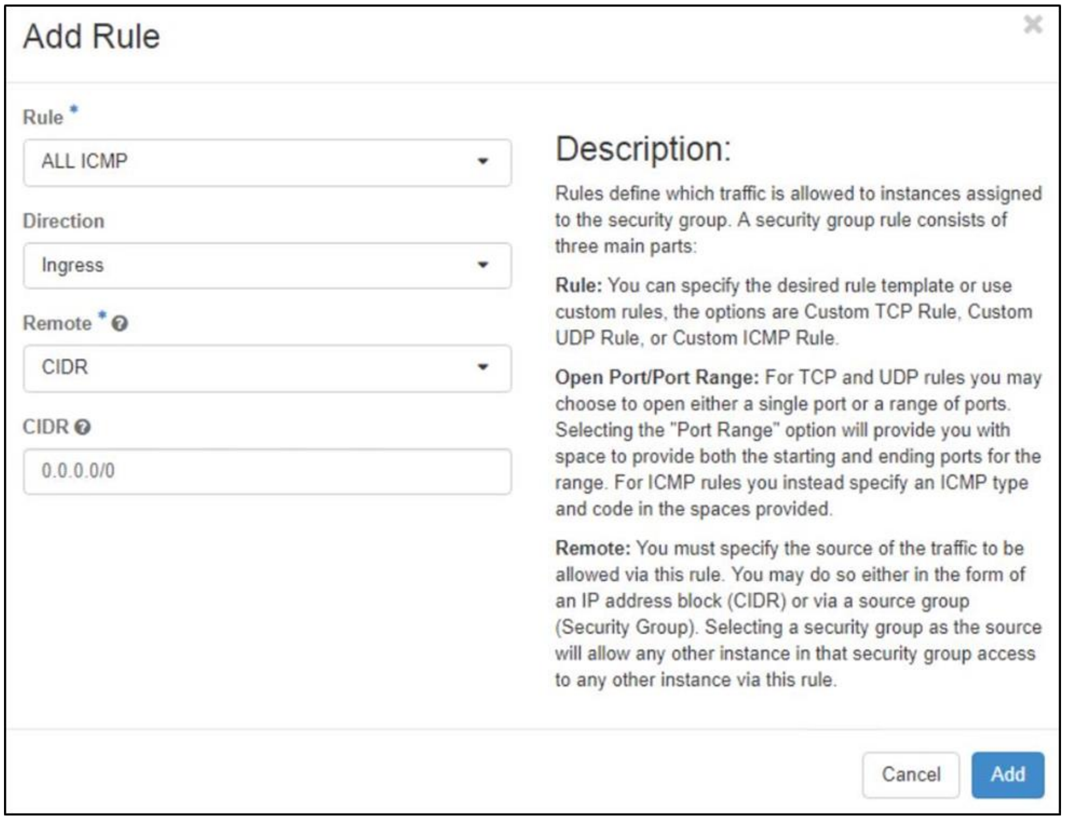


Displaying 4 items

☐	Direction	Ether Type	IP Protocol	Port Range	Remote IP Prefix	Remote Security Group
☐	Egress	IPv4	Any	Any	0.0.0.0/0	-
☐	Egress	IPv6	Any	Any	::/0	-
☐	Ingress	IPv4	Any	Any	-	default
☐	Ingress	IPv6	Any	Any	-	default

Displaying 4 items

7. Click Add Rule and select ALL ICMP from the Rule drop-down menu. This should be an ingress rule, and, if you need to define it, you can limit the scope of the CIDR network. In this example, we leave it open for testing validation. Click Add.



Add Rule

Rule *: ALL ICMP

Direction: Ingress

Remote *: CIDR

CIDR: 0.0.0.0/0

Description:

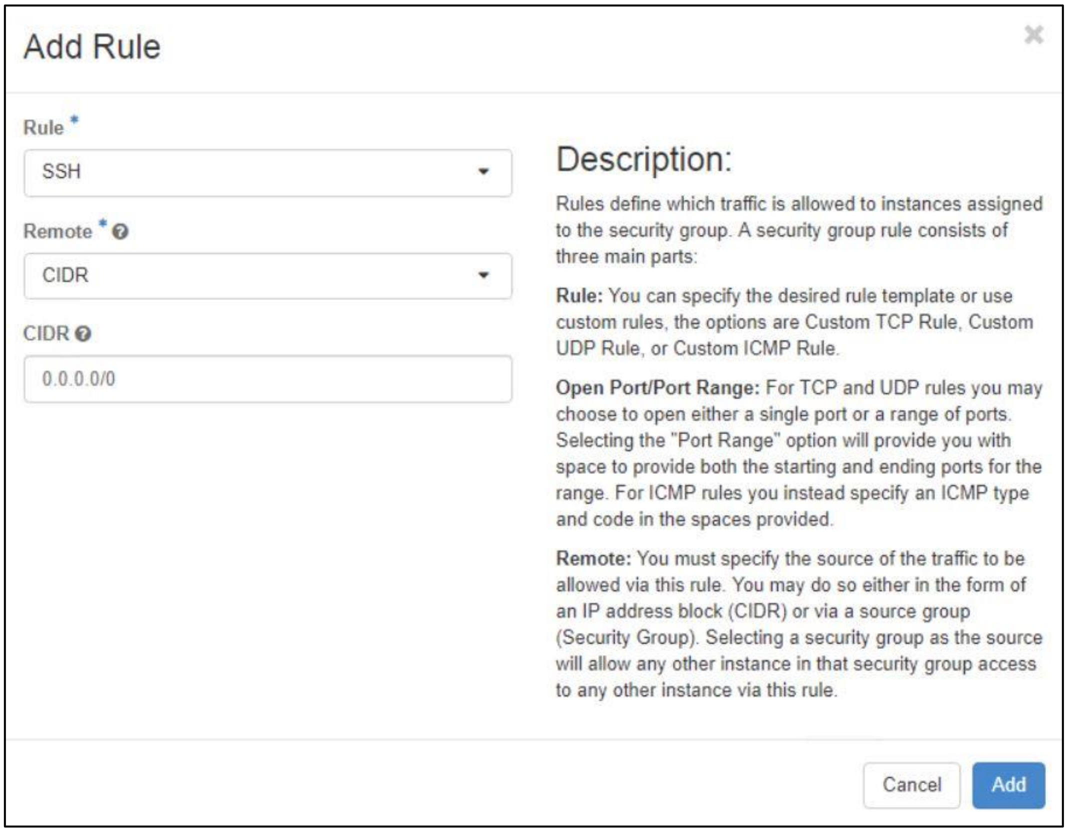
Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

Rule: You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

Open Port/Port Range: For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

Remote: You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

8. Click Add Rule again and repeat this process for SSH.



Rule *

SSH

Remote * ?

CIDR

CIDR ?

0.0.0.0/0

Description:

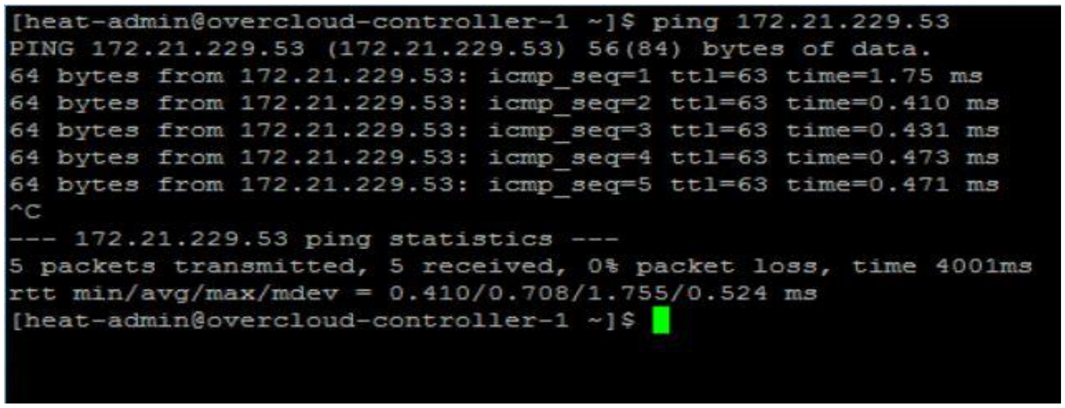
Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

Rule: You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

Open Port/Port Range: For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

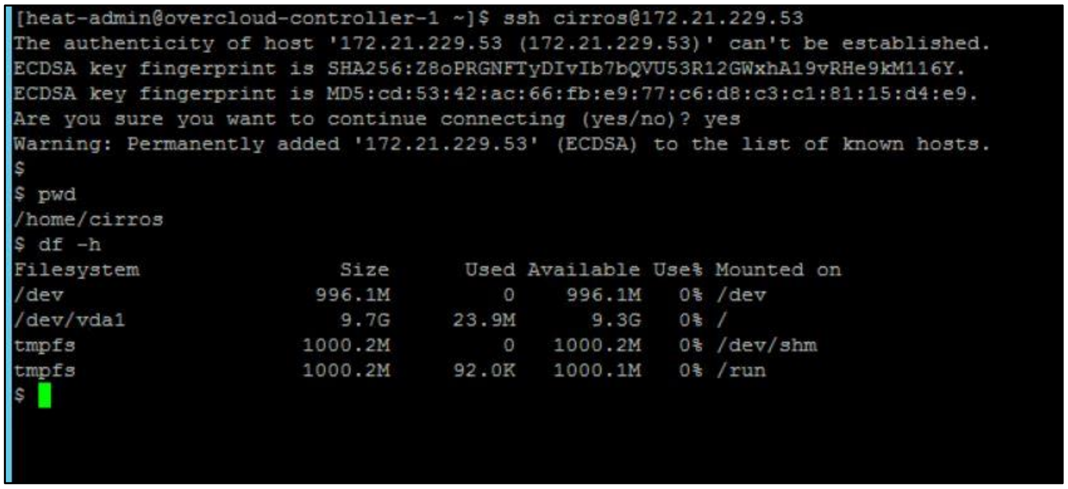
Remote: You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

9. Now attempt to both ping and SSH to your running instance at its floating IP address. The following image shows the expected ping response.



```
[heat-admin@overcloud-controller-1 ~]$ ping 172.21.229.53
PING 172.21.229.53 (172.21.229.53) 56(84) bytes of data.
64 bytes from 172.21.229.53: icmp_seq=1 ttl=63 time=1.75 ms
64 bytes from 172.21.229.53: icmp_seq=2 ttl=63 time=0.410 ms
64 bytes from 172.21.229.53: icmp_seq=3 ttl=63 time=0.431 ms
64 bytes from 172.21.229.53: icmp_seq=4 ttl=63 time=0.473 ms
64 bytes from 172.21.229.53: icmp_seq=5 ttl=63 time=0.471 ms
^C
--- 172.21.229.53 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4001ms
rtt min/avg/max/mdev = 0.410/0.708/1.755/0.524 ms
[heat-admin@overcloud-controller-1 ~]$
```

10. An attempt to perform SSH authenticates using the private key defined in the earlier step. It also allows you to log in and run commands to verify the instance, such as confirming the presence of the 10GB Element volume.



```
[heat-admin@overcloud-controller-1 ~]$ ssh cirros@172.21.229.53
The authenticity of host '172.21.229.53 (172.21.229.53)' can't be established.
ECDSA key fingerprint is SHA256:Z8oPRGNFTyDIvIb7bQVU53R12GWxhA19vRHe9kM116Y.
ECDSA key fingerprint is MD5:cd:53:42:ac:66:fb:e9:77:c6:d8:c3:c1:81:15:d4:e9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.21.229.53' (ECDSA) to the list of known hosts.
$
$ pwd
/home/cirros
$ df -h
Filesystem                Size      Used Available Use% Mounted on
/dev                    996.1M         0    996.1M   0% /dev
/dev/vda1                 9.7G     23.9M      9.3G   0% /
tmpfs                  1000.2M         0   1000.2M   0% /dev/shm
tmpfs                  1000.2M     92.0K   1000.1M   0% /run
$
```

7 Red Hat OpenShift Container Platform on OpenStack

As an OpenStack administrator, complete the following steps to run the Red Hat OpenShift Container Platform on OpenStack:

1. Create a project (tenant) to store the OpenShift instances.

```
 openstack project create openshift-hci
```

2. Create a user to own the project.

```
openstack user create --password  password netapphci
```

3. Set the role of the user.

```
openstack role add --user netapphci --project openshift-hci _member_
```

4. Adjust the default quotas to satisfy the OpenShift requirements.

```
openstack quota set --secgroups 30 --secgroup-rules 200 --ports 200 --volumes 200 -gigabytes 1000
-ram 100 openshift-hci
```

5. Create the `m1.master` and `m1.node` flavors for the OpenShift master and node instances.

```
openstack flavor create m1.master \
    --id auto \
    --ram 16384 \
    --vcpus 4
openstack flavor create m1.node \
    --id auto \
    --ram 8192 \
    --vcpus 1
```

Note: The instances are booted from Cinder volumes, so make sure that the disk is set to zero.

Note: The resources defined for each flavor are the minimums required for deploy OCP nodes and can be modified as required by the end user.

6. Copy `overcloudrc` to `openshiftrc` and modify the highlighted sections to reflect the newly created project and user credentials.

```
cat openshiftrc
# Clear any old environment that may conflict.
for key in $( set | awk '{FS="="}  /^OS_/ {print $1}' ); do unset $key ; done
export OS_NO_CACHE=True
export COMPUTE_API_VERSION=1.1
export OS_USERNAME=netapphci
export no_proxy=,172.21.229.22,192.168.24.9,172.21.229.22,192.168.24.9
export OS_USER_DOMAIN_NAME=Default
export OS_VOLUME_API_VERSION=3
export OS_CLOUDNAME=overcloud
export OS_AUTH_URL=http://172.21.229.22:5000//v3
export NOVA_VERSION=1.1
export OS_IMAGE_API_VERSION=2
export OS_PASSWORD=password
export OS_PROJECT_DOMAIN_NAME=Default
export OS_IDENTITY_API_VERSION=3
export OS_PROJECT_NAME=openshift-hci
export OS_AUTH_TYPE=password
export PYTHONWARNINGS="ignore:Certificate has no, ignore:A true SSLContext object is not
available"

# Add OS_CLOUDNAME to PS1
if [ -z "${CLOUDPROMPT_ENABLED:-}" ]; then
    export PS1=${PS1:-""}
    export PS1=\${OS_CLOUDNAME:+"(\$OS_CLOUDNAME)"}\ $PS1
    export CLOUDPROMPT_ENABLED=1
fi
(overcloud) [stack@osp-director ~]$
```

7. The Red Hat OpenStack Platform uses `cloud-init` to place an SSH public key on each instance as it is created to allow SSH access to the instance. Red Hat OpenStack Platform expects the user to hold the private key.

```
(overcloud) [stack@osp-director ~]$ openstack keypair create netapp-key > /home/stack/netapp-
key.pem
(overcloud) [stack@osp-director ~]$ chmod 600 netapp-key.pem
```

8. To upload images for Red Hat Enterprise Linux 7.6 in Glance, follow the instructions in section 6.3, “Create Images.”

7.1 OpenStack Deployment Host Security Group

OpenStack networking allows the user to define inbound and outbound traffic filters that can be applied to each instance on a network. This process allows the user to limit network traffic to each instance based on the function of the instance services and not depend on host-based filtering. The OpenShift Ansible installer handles the proper creation of all the ports and services required for each type of host that is part of the OpenShift Container Platform cluster except for the deployment host.

```
(overcloud) [stack@osp-director ~]$ source openshiftrc
(overcloud) [stack@osp-director ~]$ openstack security group create openshift-hci
```

The deployment instance only needs to allow inbound SSH and ICMP.

```
openstack security group rule create --ingress --protocol icmp openshift-hci
openstack security group rule create --ingress --protocol tcp --dst-port 22 openshift-hci
```

Configure Registry Volume

The OpenShift image registry requires a Cinder volume to make sure that images are saved if the registry needs to migrate to another node.

```
Openstack volume create –size 100 registry
```

Create and Configure the Deployment Instance

The deployment instance serves as a utility host for the deployment and management of the OpenShift Container Platform.

Create the Deployment Host Network and Router

To create the deployment host network and router, run the following commands:

```
(overcloud) [stack@osp-director ~]$ openstack network create openshift-network

(overcloud) [stack@osp-director ~]$ openstack subnet create –network openshift-network –subnet-
range 192.160.10.0/24 –dns-nameserver 8.8.8.8 openshift-network-subnet

(overcloud) [stack@osp-director ~]$ openstack router create openshift-router

(overcloud) [stack@osp-director ~]$ openstack router set –external-gateway external openshift-
router

(overcloud) [stack@osp-director ~]$ openstack router add subnet openshift-router openshift-
network-subnet
```

Note: The external network was created in section 6.4, “Create an External Network and Floating IP Pool.”

Deploy the Deployment Instance

To deploy the deployment instance, complete the following steps:

1. Run the following command:

```
(overcloud) [stack@osp-director ~]$openstack server create --block-device
source=image,id=<glance-image-id>,dest=volume,size=50,shutdown=remove,bootindex=0 --nic net-
id=$netid1 --flavor m1.node --key-name netapp-key --security-group openshift-hci  deployment
```

2. Create a floating IP for the instance.

```
(overcloud) [stack@osp-director ~]$ openstack floating ip create external
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| created_at          | 2019-07-14T21:36:35Z                 |
| description         |                                      |
| fixed_ip_address    | None                                 |
| floating_ip_address | 172.21.229.79                        |
| floating_network_id | 7f11fd42-c71a-4730-b02e-6a5fe7027e89 |
| id                  | d9ce2bed-d7b2-474d-91d8-e1ab36d09331 |
| name                | 172.21.229.79                        |
| port_id             | None                                 |
| project_id          | 8bc35517a8f84a458b841f660abe7868     |
| qos_policy_id       | None                                 |
| revision_number     | 0                                    |
| router_id           | None                                 |
| status              | DOWN                                 |
| subnet_id           | None                                 |
| updated_at          | 2019-07-14T21:36:35Z                 |
+---------------------+--------------------------------------+
(overcloud) [stack@osp-director ~]$
```

3. Associate the floating IP address provided in the previous command output with the deployment instance.

```
(overcloud) [stack@osp-director ~]$ openstack server add floating ip deployment 172.21.229.79
```

4. Set up SSH forwarding to the deployment instance by completing the following commands:
   a. From the Director node, run the following command.

```
(overcloud) [stack@osp-director ~]$ ssh-add netapp-key.pem
```

   b. Edit the `~/.ssh/config` file with the following information.

```
Host deployment
    HostName        172.21.229.79
    User            cloud-user
    IdentityFile    /home/stack/netapp-key.pem
    ForwardAgent     yes
```

5. Verify that permissions are read write only for the owner of the `~/.ssh/config` file.

```
(overcloud) [stack@osp-director ~]$ chmod 600 ~/.ssh/config
```

c. Add the RC file to the deployment host.

```
(overcloud) [stack@osp-director ~]$ scp -i netapp-key.pem openshiftrc cloud-
user@172.21.229.79:/home/cloud-user
```

Enable OpenShift Container Platform Repositories

To enable OpenShift Container Platform repositories, complete the following steps:

1. Log in to the deployment instance.

```
(overcloud) [stack@osp-director ~]$ ssh -A cloud-user@172.21.229.79
```

2. Subscribe to the relevant repositories using your Red Hat portal credentials.

```
[cloud-user@deployment ~]$ sudo subscription-manager register
```

3. Attach the Red Hat OpenShift Container Platform pool.

```
[cloud-user@deployment ~]$ sudo subscription-manager attach --pool=<pool-id>
Successfully attached a subscription for: Red Hat OpenShift, Standard Support (10 Cores, NFR,
Partner Only)
[cloud-user@deployment ~]$
```

4. Enable the repositories for Red Hat OpenShift Container Platform 3.11.

```
[cloud-user@deployment ~]$ sudo subscription-manager repos --disable=*
[cloud-user@deployment ~]$ sudo subscription-manager repos --enable="rhel-7-server-rpms" --
enable="rhel-7-server-extras-rpms"  --enable="rhel-7-server-ose-3.11-rpms” --enable="rhel-7-
server-ansible-2.5-rpms"  --enable="rhel-7-server-openstack-13-rpms"   --enable="rhel-7-server-
openstack-13-tools-rpms"
```

5. Install the following packages on the deployment host.

```
$ sudo yum -y install openshift-ansible python-openstackclient python2-heatclient python2-
octaviaclient python2-shade python-dns git ansible
```

6. Ansible is installed on the deployment instance to perform the registration, installation of packages, and the deployment of the OpenShift Container Platform environment on the master and node instances. Before running playbooks, you must create an `ansible.cfg` file to reflect the environment you want to deploy.

```
cat ~/ansible.cfg

[defaults]
forks = 20
host_key_checking = False
remote_user = openshift
gathering = smart
fact_caching = jsonfile
fact_caching_connection = $HOME/ansible/facts
fact_caching_timeout = 600
log_path = $HOME/ansible.log
nocows = 1
callback_whitelist = profile_tasks
inventory = /usr/share/ansible/openshift-ansible/playbooks/openstack/inventory.py,/home/cloud-
user/inventory

[ssh_connection]
ssh_args = -o ControlMaster=auto -o ControlPersist=600s -o UserKnownHostsFile=/dev/null -o
StrictHostKeyChecking=false
control_path = %(directory)s/%%h-%%r
pipelining = True
timeout = 10

[persistent_connection]
connect_timeout = 30
connect_retries = 30
```

```
connect_interval = 1
```

7. Copy the sample inventory directory.

```
cp -r /usr/share/ansible/openshift-ansible/playbooks/openstack/sample-inventory/ ~/inventory
```

8. The `all.yml` file enables you to customize the OpenShift instances required for successful deployment of an OpenShift Container Platform. Table 7 specifies some of the parameters you need to modify in the `all.yml` file.

Table 7) `all.yml` file parameters.

Variable	Description	Value
openshift_openstack_clusterid	Cluster identification name	openshift
openshift_openstack_public_dns_domain	Public DNS domain name	netapp-hci.rtp.com
openshift_openstack_dns_nameservers	IP address of the DNS server	["192.160.10.20"]
openshift_openstack_keypair_name	Keypair for the instances you created above	netapp-key
openshift_openstack_external_network_name	External network name for floating IPs	external
openshift_openstack_default_image_name	RHEL 7.6 image name for OpenStack instances	rhel
openshift_openstack_num_masters	Number of OpenShift Master instances to deploy	3
openshift_openstack_num_infra	Number of OpenShift Infra node instances to deploy	3
openshift_openstack_num_nodes	Number of OpenShift node instances to deploy	2
openshift_openstack_node_subnet_name	Subnet name of the private network	openshift-network-subnet
openshift_openstack_router_name	OpenStack router deployed above	openshift-router
openshift_openstack_master_floating_ip	Use floating IPs	false
openshift_openstack_etcd_floating_ip	Use floating IPs	false
openshift_openstack_load_balancer_floating_ip	Use floating IPs	false
openshift_openstack_compute_floating_ip	Use floating IPs	false
openshift_openstack_master_flavor	Flavor for master instances	m1.master
openshift_openstack_default_flavor	Flavor for other instances	m1.node
openshift_openstack_use_lbaas_load_balancer	Enable Octavia LB	true
openshift_openstack_docker_volume_size	Docker volume size	50
rhsub_user	Red Hat subscription user	Username
rhsub_pass	Red Hat subscription password	password

Note: `all.yml` and `OSEv3.yml` have been provided as references in the appendix.

Run the OpenStack Prerequisites Playbook

To run the OpenStack Prerequisites Playbook, complete the following steps:

1. Source the RC file.

```
source openshiftrc
```

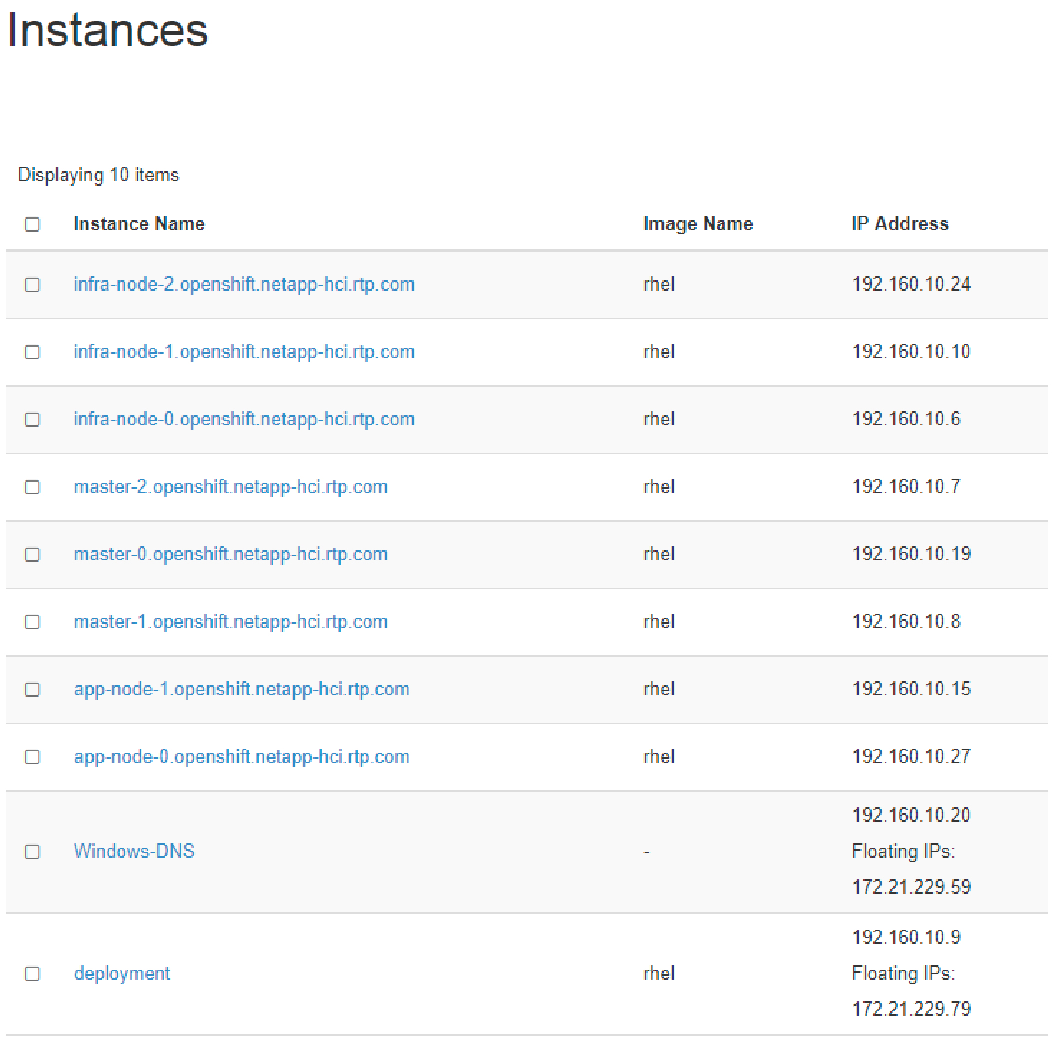
2. Make sure that all prerequisites are met using the `prerequisites.yml` playbook.

```
ansible-playbook /usr/share/ansible/openshift-ansible/playbooks/openstack/openshift-
cluster/prerequisites.yml
```

3. After the prerequisite playbook completes successfully, run the provision playbook as follows:

```
ansible-playbook /usr/share/ansible/openshift-ansible/playbooks/openstack/openshift-
cluster/provision.yml
```

This step should deploy the required OpenShift instances.

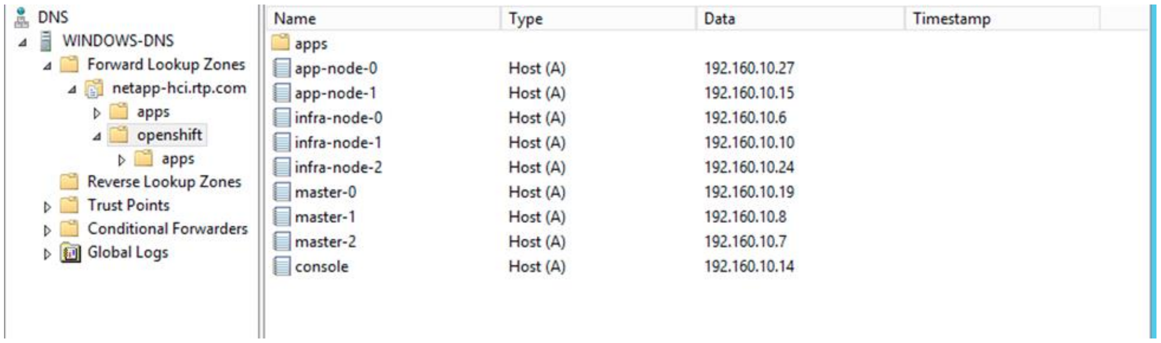


Instances

Displaying 10 items

☐	Instance Name	Image Name	IP Address
☐	infra-node-2.openshift.netapp-hci.rtp.com	rhel	192.160.10.24
☐	infra-node-1.openshift.netapp-hci.rtp.com	rhel	192.160.10.10
☐	infra-node-0.openshift.netapp-hci.rtp.com	rhel	192.160.10.6
☐	master-2.openshift.netapp-hci.rtp.com	rhel	192.160.10.7
☐	master-0.openshift.netapp-hci.rtp.com	rhel	192.160.10.19
☐	master-1.openshift.netapp-hci.rtp.com	rhel	192.160.10.8
☐	app-node-1.openshift.netapp-hci.rtp.com	rhel	192.160.10.15
☐	app-node-0.openshift.netapp-hci.rtp.com	rhel	192.160.10.27
☐	Windows-DNS	-	192.160.10.20 Floating IPs: 172.21.229.59
☐	deployment	rhel	192.160.10.9 Floating IPs: 172.21.229.79

4. Red Hat OpenShift Container Platform requires a fully functional DNS server for name resolution. Setting up a DNS server is outside the scope of this document. Make sure that you add host entries in the DNS server.



Name	Type	Data	Timestamp
apps			
app-node-0	Host (A)	192.160.10.27	
app-node-1	Host (A)	192.160.10.15	
infra-node-0	Host (A)	192.160.10.6	
infra-node-1	Host (A)	192.160.10.10	
infra-node-2	Host (A)	192.160.10.24	
master-0	Host (A)	192.160.10.19	
master-1	Host (A)	192.160.10.8	
master-2	Host (A)	192.160.10.7	
console	Host (A)	192.160.10.14	

Note: Host IPs for DNS records for `*.apps.netapp-hci.rtp.com` and `console.openshift.netapp-hci.rtp.com` can be found from Octavia LB.

Register OpenShift Container Platform Instances with Subscription Manager

To register OpenShift Container Platform instances with Subscription Manager, complete the following steps:

1. With the nodes successfully provisioned, you must make sure that all nodes are successfully registered with Subscription Manager to install all the required packages for a successful OpenShift Container Platform installation. For simplicity, we have provided a `repos.yml` file.

```
---
- name: Enable the proper repositories for OpenShift installation
  hosts: OSEv3
  become: yes
  tasks:
  - name: Register with activationkey and consume subscriptions matching Red Hat Cloud Suite or
Red Hat OpenShift Container Platform
    redhat_subscription:
      state: present
      username: aborulka@netapp.com
      password: your-rhsm-passwod
      pool_ids: pool-id-for-OpenShift-Container-Platform

  - name: Disable all current repositories
    rhsm_repository:
      name: '*'
      state: disabled

  - name: Enable Repositories
    rhsm_repository:
      name: "{{ item }}"
      state: enabled
    with_items:
      - rhel-7-server-rpms
      - rhel-7-server-extras-rpms
      - rhel-7-server-ansible-2.5-rpms
      - rhel-7-server-ose-3.11-rpms
(overcloud) [cloud-user@deployment ~]$
```

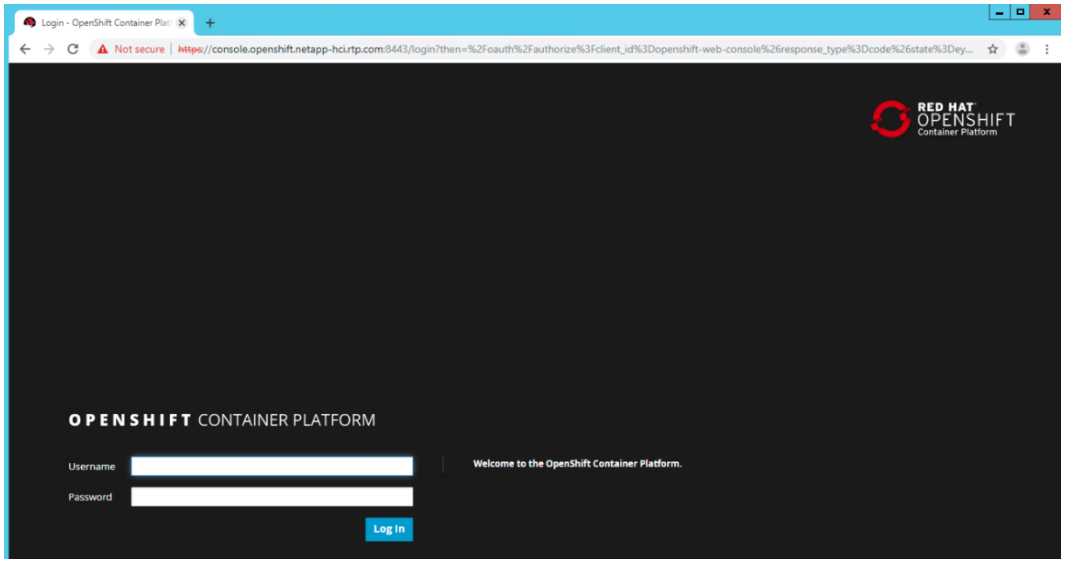
2. Review the `OSEv3.yml` file to make sure that all the options have been properly set.

```
ansible-playbook /usr/share/ansible/openshift-ansible/playbooks/prerequisites.yml
```

3. Run the Installer to install Red Hat OpenShift Container Platform.

```
ansible-playbook /usr/share/ansible/openshift-ansible/playbooks/openstack/openshift-
cluster/install.yml
```

4. After the installation finishes successfully, you should be able to navigate to the console URL.



Login - OpenShift Container Plat

Not secure | https://console.openshift.netapp-hci.rtp.com:8443/login?then=%2Foauth%2Fauthorize%3Fclient_id%3Dopenshift-web-console%26response_type%3Dcode%26state%3Dey...

RED HAT OPENSHIFT Container Platform

OPENSHIFT CONTAINER PLATFORM

Username

Password

Log In

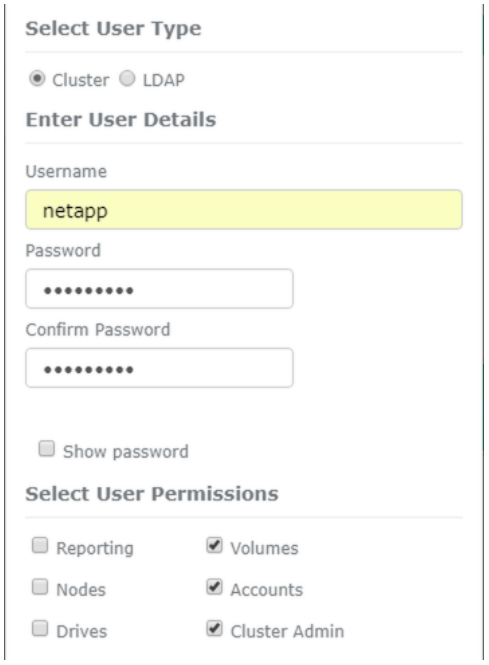
Welcome to the OpenShift Container Platform.

7.2 Deploy and Configure Trident

This section describes the Trident deployment and configuration for NetApp HCI.

Note: For comprehensive and detailed documentation for deploying and configuring Trident, see the Trident documentation page.

1. The Trident user requires privileges to `volumes`, `accounts`, and `clusterAdmins` in Element OS. NetApp recommends creating a separate user with these privileges instead of using the default admin user.



Select User Type

[✓ Cluster] LDAP

Enter User Details

Username: netapp

Password: •••••••••

Confirm Password: •••••••••

☐ Show password

Select User Permissions

☐ Reporting	✓ Volumes
☐ Nodes	✓ Accounts
☐ Drives	✓ Cluster Admin

2. Verify that the `netapp` user has sufficient privileges.

```
[root@bastion ~]# oc auth can-i '*' '*' --all-namespaces
```

```
Yes
```

3. Verify that you can download an image from the registry and access the management virtual IP address (MVIP) of your NetApp Element cluster.

```
[root@bastion ~]# oc run -i --tty ping --image=busybox --restart=Never --rm --   ping
172.21.230.150
If you don't see a command prompt, try pressing enter.
64 bytes from 172.21.230.150: seq=1 ttl=62 time=0.776 ms
64 bytes from 172.21.230.150: seq=2 ttl=62 time=1.157 ms
64 bytes from 172.21.230.150: seq=3 ttl=62 time=0.800 ms
64 bytes from 172.21.230.150: seq=4 ttl=62 time=1.050 ms
64 bytes from 172.21.230.150: seq=5 ttl=62 time=0.871 ms
```

Note: Creation of a provider network with the Element MVIP and storage virtual IP address (SVIP) VLANs might be necessary. Attach the Element MVIP network to the OpenStack router, and add an extra interface for the SVIP VLAN on the OpenShift Infra and node instances.

4. Download Trident installer bundle from the Downloads section and extract it.

```
wget https://github.com/NetApp/trident/releases/download/v19.04.1/trident-installer-
19.04.1.tar.gz
tar -xvf trident-installer-19.04.1.tar.gz
cd trident-installer
```

5. Create a `setup/backend.json` file with the following configuration:

```
{
    "version": 1,
    "storageDriverName": "solidfire-san",
    "Endpoint": "https://netapp:your-password@172.21.230.150/json-rpc/8.0",
    "SVIP": "192.168.62.150:3260",
    "TenantName": "trident",
    "Types": [{"Type": "Bronze", "Qos": {"minIOPS": 1000, "maxIOPS": 2000, "burstIOPS": 4000}},
              {"Type": "Silver", "Qos": {"minIOPS": 4000, "maxIOPS": 6000, "burstIOPS": 8000}},
              {"Type": "Gold", "Qos": {"minIOPS": 6000, "maxIOPS": 8000, "burstIOPS": 10000}}]
}
```

`Endpoint` corresponds to the Trident user created in Step 1 and the MVIP of the NetApp HCI Element cluster. `SVIP` corresponds to the `openshift_storage_network`.

`Types` corresponds to different QoS bands. New persistent volumes can be created with specific QoS settings by specifying the exact storage pool.

6. Create a new project called trident in OpenShift and configure the OpenShift cluster to run Trident on OpenShift Infra nodes.

```
oc adm new-project trident --node-selector=node-role.kubernetes.io/infra=true
```

7. Install Trident.

```
[root@bastion trident-installer]# ./tridentctl install  -n trident
INFO Created installer service account.            serviceaccount=trident-installer
INFO Waiting for object to be created.             objectName=clusterRole
INFO Created installer cluster role.               clusterrole=trident-installer
INFO Waiting for object to be created.             objectName=clusterRoleBinding
INFO Created installer cluster role binding.       clusterrolebinding=trident-installer
INFO Added security context constraint user.       scc=privileged user=trident-installer
INFO Created installer configmap.                  configmap=trident-installer
INFO Waiting for object to be created.             objectName=installerPod
INFO Created installer pod.                        pod=trident-installer
INFO Waiting for Trident installer pod to start.
INFO Trident installer pod started.                namespace=trident pod=trident-installer
INFO Starting storage driver.                      backend=/setup/backend.json
ERRO Error detected in API response.               ID=276 code=500 message=xUnknownAccount
name=xUnknownAccount
INFO Volume Access Group use not detected. Defaulting to using CHAP.  platform= version=1.11.0
INFO Storage driver initialized.                   driver=solidfire-san
INFO Created new storage backend.                  backend="&{0xc42038e3c0
solidfire_192.168.62.150 true online map[Silver:0xc4203b2680 Gold:0xc4203b26c0
Bronze:0xc4203b2600] map[]}"
```

```
INFO Storage driver loaded.                         driver=solidfire-san
INFO Starting Trident installation.                 namespace=trident
INFO Created service account.
INFO Created cluster role.
INFO Created cluster role binding.
INFO Added security context constraint user.        scc=anyuid user=trident
INFO Created PVC.
INFO Created iSCSI CHAP secret.                     secret=trident-chap-solidfire-192-168-62-150-
trident
INFO Created PV.                                    pv=trident
INFO Waiting for PVC to be bound.                   pvc=trident
INFO Controller serial numbers.                     serialNumbers="deleted"
INFO Created Trident deployment.
INFO Waiting for Trident pod to start.
INFO Trident pod started.                           namespace=trident pod=trident-8549f65cf5-6wdjb
INFO Waiting for Trident REST interface.
INFO Trident REST interface is up.                  version=19.04.1
INFO Added backend to Trident.                      backendFile=setup/backend.json
backendName=solidfire_192.168.62.150
INFO Trident installation succeeded.
INFO Waiting for Trident installer pod to finish.
INFO Trident installer pod finished.                namespace=trident pod=trident-installer
INFO Deleted installer pod.                         pod=trident-installer
INFO Deleted installer configmap.                   configmap=trident-installer
INFO In-cluster installation completed.
INFO Deleted installer cluster role binding.
INFO Deleted installer cluster role.
INFO Deleted installer service account.
INFO Removed security context constraint user.      scc=privileged user=trident-installer
```

8. Verify that Trident is running with the correct version.

```
[root@bastion trident-installer]# oc get pod -n trident
NAME                       READY     STATUS    RESTARTS   AGE
trident-8549f65cf5-6wdjb   2/2       Running   0          2m
[root@bastion trident-installer]# ./tridentctl -n trident version
+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 19.04.1        | 19.04.1        |
+----------------+----------------+
[root@bastion trident-installer]#
```

9. Create a storage backend. This can be the same `backend.json` we used in step 5.

```
./tridentctl -n trident create backend -f setup/backend.json
+--------------------------+----------------+--------+---------+
|           NAME           | STORAGE DRIVER | STATE  | VOLUMES |
+--------------------------+----------------+--------+---------+
| solidfire_192.168.62.150 | solidfire-san  | online |       0 |
+--------------------------+----------------+--------+---------+
```

The storage backend specifies the Element cluster in NetApp HCI. We also specify sample bronze, silver, and gold types with corresponding QoS specs.

10. Create a StorageClass that specifies Trident as the provisioner and the storage backend as `solidfire-san`.

```
[root@bastion trident-installer]# vi sample-input/storage-class-solidfire-bronze.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-bronze
provisioner: netapp.io/trident
parameters:
  backendType: "solidfire-san"
  IOPS: "1500"
  fsType: "ext4"
```

An OpenShift administrator can define multiple storage classes corresponding to different QoS requirements based upon their applications. A relational database management system like Oracle might have a different QoS requirements when compared to a key-value store like Redis.

Trident selects a storage backend that can satisfy all the criteria specified in the parameters section in the storage class definition. The IOPS parameter helps to identify the QoS settings to be applied to the newly created volume. Trident selects the type based on the following condition:

- minimum IOPS <= IOPS <= maximum IOPS

Therefore, NetApp recommends creating a type in your storage backend that covers the anticipated IOPS range of your application.

```
 [root@bastion trident-installer]# oc create -f sample-input/storage-class-solidfire-bronze.yaml
storageclass.storage.k8s.io "solidfire-bronze" created
[root@bastion trident-installer]#
```

7.3 Deploy OpenShift Registry

By default, the OpenShift registry uses `emptydir` as the storage backend for storing the registry images. NetApp strongly recommends patching the OpenShift registry service to use a persistent volume as the storage backend for registry. To do so, complete the following steps:

1. Create a persistent volume claim (PVC) request.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: openshift-registry
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 500Gi
  storageClassName: solidfire-bronze
```

2. Verify that the PVC request is bound.

```
[root@bastion trident-installer]# oc get pvc
NAME                 STATUS    VOLUME                            CAPACITY   ACCESS MODES
STORAGECLASS   AGE
openshift-registry   Bound     default-openshift-registry-82e21   500Gi      RWO
solidfire-bronze        5s
[root@bastion trident-installer]#
```

3. Patch the docker-registry deployment config.

```
[root@bastion trident-installer]# oc volume deploymentconfigs/docker-registry --add --
name=registry-storage -t pvc \
> --claim-name=openshift-registry --overwrite
deploymentconfig "docker-registry" updated
[root@bastion trident-installer]#
```

8 Conclusion

The NetApp HCI for Private Cloud with Red Hat solution enables enterprises to adopt a cloud-like consumption model in a reliable and risk-free manner, by building a production-grade OpenStack cloud based on a validated hardware design. This solution also includes Red Hat OpenShift Container Platform hosted within the private cloud to run microservices workloads on the premises to accelerate application development and deployment.

Appendix A: Example OpenShift YAML Configurations

This appendix provides example YAML files referenced in the deployment of OpenShift Container Platform in the section titled, “Enable OpenShift Container Platform Repositories.”

OSEv3.yml

```
---
## Openshift product versions and repos to install from
#openshift_deployment_type: origin
#openshift_repos_enable_testing: true
openshift_deployment_type: openshift-enterprise
openshift_release: v3.11
oreg_url: registry.access.redhat.com/openshift3/ose-${component}:${version}
openshift_examples_modify_imagestreams: true
#oreg_auth_user: rhusername
#oreg_auth_password: rhpasswd
rhsub_pool: '8a85f99b6b498682016beec8a4bf0d2f'

# Default domain to access the applications running on OpenShift.
# This uses the `openshift_openstack_clusterid`
# and `openshift_openstack_public_dns_domain` values from all.yml.
# It will be set to `apps.openshift.example.com` by default.
# Feel free to change this to a value you prefer. It should be under the
# domain the OpenShift cluster is configured, though.
openshift_master_default_subdomain: "apps.{{ (openshift_openstack_clusterid|trim == '') |
ternary(openshift_openstack_public_dns_domain, openshift_openstack_clusterid + '.' +
openshift_openstack_public_dns_domain) }}"

# Domain to access the OpenShift UI and API.
# This uses the `openshift_openstack_clusterid`
# and `openshift_openstack_public_dns_domain` values from all.yml.
# It will be set to `console.openshift.example.com` by default.
# Feel free to change this to a value you prefer. It should be under the
# domain the OpenShift cluster is configured, though.
openshift_master_cluster_public_hostname: "console.{{ (openshift_openstack_clusterid|trim == '')
| ternary(openshift_openstack_public_dns_domain, openshift_openstack_clusterid + '.' +
openshift_openstack_public_dns_domain) }}"


openshift_hosted_router_wait: True
openshift_hosted_registry_wait: True


## Kuryr label configuration
#kuryr_openstack_pool_driver: multi
#
#openshift_node_groups:
#   - name: node-config-master
#     labels:
#       - 'node-role.kubernetes.io/master=true'
#       - 'pod_vif=nested-vlan'
#     edits: []
#   - name: node-config-infra
#     labels:
#       - 'node-role.kubernetes.io/infra=true'
#       - 'pod_vif=nested-vlan'
#     edits: []
#   - name: node-config-compute
#     labels:
#       - 'node-role.kubernetes.io/compute=true'
#       - 'pod_vif=nested-vlan'
#     edits: []


## Openstack credentials
#openshift_cloudprovider_kind: openstack
#openshift_cloudprovider_openstack_auth_url: "{{ lookup('env','OS_AUTH_URL') }}"
#openshift_cloudprovider_openstack_username: "{{ lookup('env','OS_USERNAME') }}"
#openshift_cloudprovider_openstack_password: "{{ lookup('env','OS_PASSWORD') }}"
```

```
#openshift_cloudprovider_openstack_tenant_name: "{{ lookup('env','OS_TENANT_NAME') }}"
#openshift_cloudprovider_openstack_region: "{{ lookup('env', 'OS_REGION_NAME') }}"
#openshift_cloudprovider_openstack_blockstorage_version: v2

# Optionally specify a local openstack.conf
#openshift_cloudprovider_openstack_conf_file: /path/to/openstack.conf

## Use Cinder volume for Openshift registry:
#openshift_hosted_registry_storage_kind: openstack
#openshift_hosted_registry_storage_access_modes: ['ReadWriteOnce']
#openshift_hosted_registry_storage_openstack_filesystem: xfs
#openshift_hosted_registry_storage_volume_size: 100Gi

## If you want a Cinder volume created automaticaly, uncomment this:
#openshift_hosted_registry_storage_volume_name: registry

## If you're using a Cinder volume you've set up yourself, uncomment this:
#openshift_hosted_registry_storage_openstack_volumeID: 094fa166-8b27-4b8b-ae98-50fbcdb4c039


# NOTE(shadower): the hostname check seems to always fail because the
# host's floating IP address doesn't match the address received from
# inside the host.
openshift_hostname_check: false

# For POCs or demo environments that are using smaller instances than
# the official recommended values for RAM and DISK, uncomment the line below.
openshift_disable_check: disk_availability,memory_availability

# NOTE(shadower): Always switch to root on the OSEv3 nodes.
# openshift-ansible requires an explicit `become`.
ansible_become: true

# # Flannel networking
#osm_cluster_network_cidr: 10.128.0.0/14
#openshift_use_openshift_sdn: false
#openshift_use_flannel: true
#flannel_interface: eth1

#Setting SDN (defaults to ovs-networkpolicy) not part of OSEv3.yml
#For more info, on which to choose, visit:
#https://docs.openshift.com/container-platform/3.11/architecture/networking/sdn.html#overview
networkPluginName: redhat/ovs-networkpolicy
#networkPluginName: redhat/ovs-multitenant
openshift_master_identity_providers: [{'name': 'allow_all', 'login': 'true', 'challenge': 'true',
'kind': 'DenyAllPasswordIdentityProvider'}]
```

all.yml

```
---
openshift_openstack_clusterid: "openshift"
openshift_openstack_public_dns_domain: "netapp-hci.rtp.com"
openshift_openstack_dns_nameservers: ["192.160.10.20"]

# # Used Hostnames
# # - set custom hostnames for roles by uncommenting corresponding lines
#openshift_openstack_master_hostname: "master"
#openshift_openstack_infra_hostname: "infra-node"
#openshift_openstack_cns_hostname: "cns"
#openshift_openstack_node_hostname: "app-node"
#openshift_openstack_lb_hostname: "lb"
#openshift_openstack_etcd_hostname: "etcd"

openshift_openstack_keypair_name: "netapp-key"
openshift_openstack_external_network_name: "external"
#openshift_openstack_private_network_name:  "openshift-ansible-{{ openshift_openstack_stack_name
}}-net"
# # A dedicated Neutron network name for containers data network
# # Configures the data network to be separated from openshift_openstack_private_network_name
# # NOTE: this is only supported with Flannel SDN yet
```

```
#openstack_private_data_network_name: "openshift-ansible-{{ openshift_openstack_stack_name }}-
data-net"

## Kuryr networking
# TODO: Allow the user to specify pre-existing subnets for pod and services
#openshift_openstack_kuryr_service_subnet_cidr: "172.30.0.0/16"

## You should set the following if you want to use Kuryr/Neutron as your SDN
#openshift_use_kuryr: True
#openshift_use_openshift_sdn: False

# NOTE: you must uncomment these for Kuryr to work properly as well:
# openshift_master_open_ports:
# - service: dns tcp
#   port: 53/tcp
# - service: dns udp
#   port: 53/udp
# openshift_node_open_ports:
# - service: dns tcp
#   port: 53/tcp
# - service: dns udp
#   port: 53/udp

#use_trunk_ports: True
#os_sdn_network_plugin_name: cni
#openshift_node_proxy_mode: userspace

# # Kuryr needs to know the network or the subnet you will be taking Floating
# IPs for the loadbalancer services from.
# kuryr_openstack_public_net_id: uuid_of_my_fip_net
# If you have different subnet at the Floating IP network you can also specify
# it but need to ensure default policy allows your tennat to retrieve that
# information
# kuryr_openstack_public_subnet_id: uuid_of_my_fip_subnet

# # Kuryr can use a different subnet per namespace by enabling:
# openshift_kuryr_subnet_driver: namespace

# # Kuryr can use different security group driver to provide different
# isolation policies. The two options available are namespace isolation and
# network policies. and both of them require the subnet per namespace driver
# to be enabled. To enable one or the other use the next, by setting it to
# 'namespace' or 'policy'
# openshift_kuryr_sg_driver: policy

# If you VM images will name the ethernet device different than 'eth0',
# override this
#kuryr_cni_link_interface: eth0

## Ports pooling is enabled by default (with nested driver).
## To disable, uncomment the next:
#kuryr_openstack_pool_driver: noop
#
## You can also alter the port pooling defaults here
#kuryr_openstack_pool_max: 0
#kuryr_openstack_pool_min: 1
#kuryr_openstack_pool_batch: 5
#kuryr_openstack_pool_update_frequency: 20
#
## You can pre-populate the pools with subports by uncommenting the next line
## and specifying the amount of subports you want to pre-create per
## origin-node, e.g.: openshift_kuryr_precreate_subports: 5
#openshift_kuryr_precreate_subports: false
#
## You can also change the default device owner for the precreated subports
#openshift_kuryr_device_owner: compute:kuryr
#
# You can also disable the kuryr controller and cni healthcheck probes by
# uncommenting the next
# enable_kuryr_controller_probes: False
# enable_kuryr_cni_probes: False
```

```
## If you want to use a provider network, set its name here.
## NOTE: the `openshift_openstack_external_network_name` and
## `openshift_openstack_private_network_name` options will be ignored when using a
## provider network.
#openshift_openstack_provider_network_name: "provider"

# # Used Images
# # - set specific images for roles by uncommenting corresponding lines
# # - note: do not remove openshift_openstack_default_image_name definition
#openshift_openstack_master_image_name: "centos7"
#openshift_openstack_infra_image_name: "centos7"
#openshift_openstack_cns_image_name: "centos7"
#openshift_openstack_node_image_name: "centos7"
#openshift_openstack_lb_image_name: "centos7"
#openshift_openstack_etcd_image_name: "centos7"
openshift_openstack_default_image_name: "rhel"

openshift_openstack_num_masters: 3
openshift_openstack_num_infra: 3
openshift_openstack_num_cns: 0
openshift_openstack_num_nodes: 2
openshift_openstack_num_etcd: 0

# # Public IP Allocation
# # - manage which node roles are allocated public IP addresses
# # - by default, all roles are given Public IP addresses
openshift_openstack_node_subnet_name: "openshift-network-subnet"
openshift_openstack_router_name: "openshift-router"
openshift_openstack_master_floating_ip: false
openshift_openstack_infra_floating_ip: false
openshift_openstack_etcd_floating_ip: false
openshift_openstack_load_balancer_floating_ip: false
openshift_openstack_compute_floating_ip: false

# # Used Flavors
# # - set specific flavors for roles by uncommenting corresponding lines
# # - note: do note remove openshift_openstack_default_flavor definition
openshift_openstack_master_flavor: "m1.master"
#openshift_openstack_infra_flavor: "m1.medium"
#openshift_openstack_cns_flavor: "m1.medium"
#openshift_openstack_node_flavor: "m1.medium"
#openshift_openstack_lb_flavor: "m1.medium"
#openshift_openstack_etcd_flavor: "m1.medium"
openshift_openstack_default_flavor: "m1.node"

# # Numerical index of nodes to remove
# openshift_openstack_nodes_to_remove: []


## Select a load balancer solution you desire. Only one of these can be
## `true` at a time. If they're both `false`, no load balancer will be deployed.
openshift_openstack_use_lbaas_load_balancer: true
#openshift_openstack_use_vm_load_balancer: false


# # Docker volume size
# # - set specific volume size for roles by uncommenting corresponding lines
# # - note: do not remove docker_default_volume_size definition
#openshift_openstack_master_volume_size: "15"
#openshift_openstack_infra_volume_size: "15"
#openshift_openstack_cns_volume_size: "15"
#openshift_openstack_node_volume_size: "15"
#openshift_openstack_etcd_volume_size: "2"
#openshift_openstack_lb_volume_size: "5"
openshift_openstack_docker_volume_size: "40"

## Specify server group policies for master and infra nodes. Nova must be configured to
## enable these policies. 'anti-affinity' will ensure that each VM is launched on a
## different physical host.
```

```
openshift_openstack_master_server_group_policies: []
openshift_openstack_infra_server_group_policies: []


# The Classless Inter-Domain Routing (CIDR) for the OpenStack VM subnet.
#openshift_openstack_subnet_cidr: "192.168.99.0/24"
# The starting IP address for the OpenStack subnet allocation pool.
#openshift_openstack_pool_start: "192.168.99.3"
# The ending IP address for the OpenStack subnet allocation pool.
#openshift_openstack_pool_end: "192.168.99.254"

## Red Hat subscription using user/password
rhsub_user: rhusername
rhsub_pass: rhpassword
# Or if using activation key:
#rhsub_ak: '<ak>'
#rhsub_orgid: '<orgid>'
#rhsub_pool: '<pool name>'


# # Roll-your-own DNS
#openshift_openstack_external_nsupdate_keys:
#  public:
#    key_secret:
'SKqKNdpfk7llKxZ57bbxUnUDobaaJp9t8CjXLJPl+fRI5mPcSBuxTAyvJPa6Y9R7vUg9DwCy/6WTpgLNqnV4Hg=='
#    key_algorithm: 'hmac-md5'
#    server: '192.168.1.1'
#  private:
#    key_secret:
'kVE2bVTgZjrdJipxPhID8BEZmbHD8cExlVPR+zbFpW6la8kL5wpXiwOh8q5AAosXQI5t95UXwq3Inx8QT58duw=='
#    key_algorithm: 'hmac-md5'
#    server: '192.168.1.2'


# NOTE(shadower): Do not change this value. The Ansible user is currently
# hardcoded to `openshift`.
ansible_user: openshift

# Resolve heat stack outputs. Disabling outputs helps stack scalability. (default: True)
#openshift_openstack_resolve_heat_outputs: True

# # Use a single security group for a cluster (default: false)
#openshift_openstack_flat_secgrp: false

# If you want to use the VM storage instead of Cinder volumes, set this to `true`.
# NOTE: this is for testing only! Your data will be gone once the VM disappears!
# openshift_openstack_ephemeral_volumes: false

## cloud config
openshift_openstack_disable_root: true
openshift_openstack_user: openshift
```

Appendix B: Example ifcfg File for OpenStack Platform Director

This appendix provides a sample network interface file for use with the OpenStack Platform Director, as referenced in the section titled, “Installation of Red Hat Enterprise Linux Operating System for the Red Hat OpenStack Platform Director (Undercloud) Node.”

```
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
IPV4_FAILURE_FATAL=no
NAME=enp59s0f0
UUID=87bee55e-c2cc-4095-8dbd-a61bcf6eb219
DEVICE=enp59s0f0
ONBOOT=yes
```

```
DEVICE=enp59s0f0.3486
BOOTPROTO=none
DEFROUTE=yes
NAME=enp59s0f0
ONBOOT=yes
VLAN=yes
IPADDR=172.21.230.75
PREFIX=24
GATEWAY=172.21.230.254
DNS1=172.21.230.254
```

Where to Find Additional Information

To learn more about the information that is described in this document, see the following documents and/or websites:

- NetApp HCI Datasheet | DS-3881
  https://www.netapp.com/us/media/ds-3881.pdf
- NetApp HCI with Mellanox Switches | TR-4735
  https://www.netapp.com/us/media/tr-4735.pdf
- NetApp HCI Documentation Center
  http://docs.netapp.com/hci/index.jsp
- NetApp HCI Documentation Resources
  https://www.netapp.com/us/documentation/hci.aspx
- NetApp Product Documentation
  https://docs.netapp.com
- NetApp HCI product page
  https://www.netapp.com/us/products/converged-systems/hyper-converged-infrastructure.aspx
- NetApp Trident Documentation page
  https://netapp-trident.readthedocs.io/en/stable-v19.04/#
- Red Hat OpenStack Platform 13
  https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/13/
- Red Hat OpenShift Container Platform 3.11
  https://access.redhat.com/documentation/en-us/openshift_container_platform/3.11/
- Deploying OpenShift on OpenStack
  https://docs.openshift.com/container-platform/3.11/install_config/configuring_openstack.html
- OpenStack Kolla
  https://github.com/openstack/kolla
- NIST Definition of Cloud Computing
  https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf
- OpenStack: A Business Perspective
  https://www.openstack.org/assets/pdf-downloads/business-perspectives.pdf

Version History

Version	Date	Document Version History
Version 2.0	September 2019	Updated document after internal and Red Hat team review.
Version 1.0	July 2019	Initial release.

NVA-1133-DEPLOY-0919