Dell OpenDayLight
Administration Guide

Notes, cautions, and warnings

NOTE: A NOTE indicates important information that helps you make better use of your computer.

CAUTION: A CAUTION indicates either potential damage to hardware or loss of data and tells you how to avoid the problem.

WARNING: A WARNING indicates a potential for property damage, personal injury, or death.

2015 - 10

Rev. 1.0.0.0

Contents

1

About this Guide

This guide describes Dell OpenDayLight (Dell ODL) and provides installation and configuration information.

Audience

This guide is intended for system administrators who are responsible for configuring and maintaining networks and assumes knowledge in Layer 2 and Layer 3 networking technologies.

Conventions

This guide uses the following conventions to describe command syntax.

<table>
<tr><td><b><code>Keyword</code></b></td><td>Keywords are in Courier (a monospaced font) and must be entered in the CLI as listed.</td></tr>
<tr><td><i>parameter</i></td><td>Parameters are in italics and require a number or word to be entered in the CLI.</td></tr>
<tr><td>{X}</td><td>Keywords and parameters within braces must be entered in the CLI.</td></tr>
<tr><td>[X]</td><td>Keywords and parameters within brackets are optional.</td></tr>
<tr><td>x|y</td><td>Keywords and parameters separated by a bar require you to choose one option.</td></tr>
<tr><td>x||y</td><td>Keywords and parameters separated by a double bar allows you to choose any or all of the options.</td></tr>
</table>

2

Dell OpenDayLight Overview

The new data center is an ecosystem based on openness and interoperability that allows you to choose the best hardware and software to meeting their needs.

OpenDayLight (ODL) is a collaborative open-source project that aims to accelerate adoption of sofware-defined networking (SDN) and network functions virtualization (NFV) for a more transparent approach that fosters new innovation and reduces risk.

What is Dell ODL?

The Dell ODL controller lies inside of the data center in a cluster or other high availability (HA) mechanism, which is accessible to the infrastructure (regardless of scale) in a similar manner and equidistant to all end-points over an out-of-band network (OOB).

Dell ODL has endpoints as physical switches, virtual switches, and wireless access points (APs). ODL software is able to program hardware based on vertical sample configurations. For more information about sample configurations, see Testing Multi-Tenancy.

The ODL controller is located on the layer above the endpoints, and it sends the instructions based on direction from applications. The application layer depends on the user profile and place-in-network. An example of place-in-network includes Lync for campus applications and OpenStack for DC applications.

Where To Start

This guide covers not only installation, but also includes sample network configurations to help you understand how to implement your data center using ODL. Once you have successfully installed and configured ODL, you are then ready to create and configure your network for multi-tenancy.

To get started, see Sample Network Configuration first to fully understand how the management network, data network, and external network is configured.

Requirements

This information outlines the hardware requirements, and hardware and software configuration requirements.

Hardware Requirements

The following lists the minimal hardware requirements:

- A minimal OpenStack deployment with controller, network, and compute nodes

- A separate host to deploy the Dell ODL controller
- Network connectivity with switches

Hardware Configuration Requirements

The following lists the suggested hardware configuration:

- Dell server/Blade with 2 to 4 CPU cores
- 128 GB RAM suggested
- 500 GB HDD Dell switches according to the network requirements
- OpenStack control and compute node requirements (see Architecture).

Software Configuration Requirements

The following lists the required software configuration:

- OpenStack Kilo version, supported on an appropriate OS with kernel-based virtual machine (KVM)
- Dell ODL controller in ODL node with Ubuntu 14.04 LTS
- Openvswitch 2.3.2

OpenSource Horizon Requirements

The following lists the system requirements for installing OpenSource Horizon:

- Python 2.7
- Django 1.7 or above

Minimum required set of running OpenStack services are:

- nova: OpenStack Compute
- keystone: OpenStack Identity
- glance: OpenStack Image service
- neutron: OpenStack Networking

All other services are optional. Horizon supports the following services in the Juno release. If you configure the keystone endpoint for a service, Horizon detects and enables it and enables its support automatically (see Installing Horizon).

OpenStack Network Node

The standard OpenStack network node typically runs in a separate node and handles two important functions:

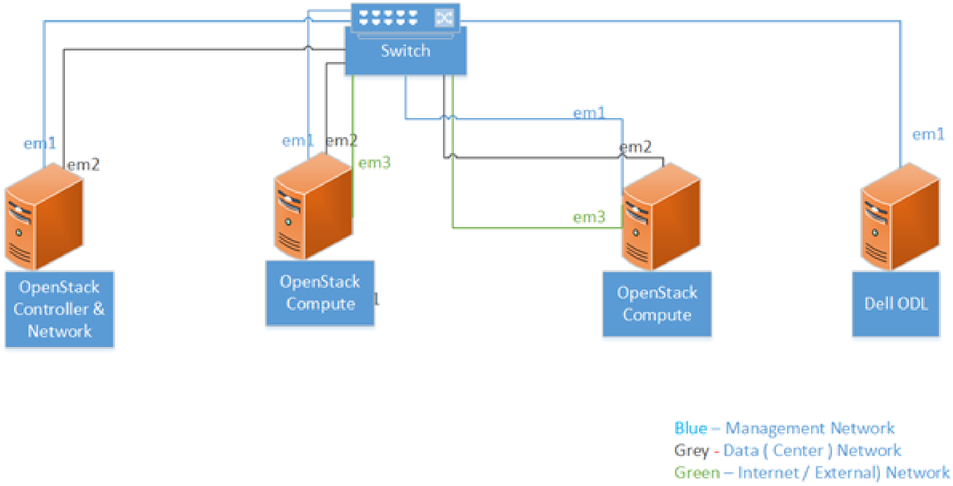
- Communication to the external network/Internet
- Dynamic host configuration protocol (DHCP) service for tenants

In the Dell ODL architecture, there is no centralized router node. Intra-tenant routing is achieved using a distributed virtual router (DVR). Access to the external network/Internet is achieved with the provision of an additional interface to each of the compute nodes.

3

Sample Network Configuration

This information provides a sample network configuration for the underlay, which can be used as a reference and modified according to your needs.


Switch
em1
em2
em1
em2
em3
em1
em2
em3
em1
OpenStack Controller & Network
OpenStack Compute
OpenStack Compute
Dell ODL
Blue – Management Network
Grey - Data (Center) Network
Green – Internet / External) Network

See the following links for complete sample configuration information:

- Management Network Configuration
- Data Network Configuration
- External Network Configuration

Sample Topology

The following table provides sample topology information.

Server	Node Type	Interface em1 (Mgmt Network)	Interface em2 (Data Network)	Interface em3 (External Network/ Internet)	Remarks
R630/R710/ R720	Controller + Network Node	10.16.148.31	20.1.1.2	N/A	
R630/R710/ R720	Compute Node	10.16.148.33	20.1.1.3	0.0.0.0	Configure floating IPs for em3
R630/R710/ R720	Compute Node	10.16.148.35	20.1.1.4	0.0.0.0	Configure floating IPs for em3

Server	Node Type	Interface em1 (Mgmt Network)	Interface em2 (Data Network)	Interface em3 (External Network/ Internet)	Remarks
R630/R710/ R720	Dell ODL Node	10.16.148.232	N/A	N/A	Configure in External ODL mode

Management Network Configuration

The following provides information on the OpenStack controller, OpenStack compute, Dell ODL controller, and switch configuration for the management network.

- OpenStack Controller

```
openflow@os-controller:~$ ifconfig em1
em1 Link encap:Ethernet HWaddr c8:1f:66:da:3e:93
inet addr:10.16.148.31 Bcast:10.16.255.255 Mask:255.255.0.0
inet6 addr: fe80::ca1f:66ff:feda:3e93/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:54349669 errors:0 dropped:79 overruns:0 frame:0
TX packets:58231618 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:12810417610 (12.8 GB) TX bytes:30485924276 (30.4 GB)
Interrupt:35
```

- OpenStack Compute

```
openflow@os-compute1:~/devstack$ ifconfig em1
em1 Link encap:Ethernet HWaddr f0:1f:af:ce:70:7a
inet addr:10.16.148.33 Bcast:10.16.255.255 Mask:255.255.0.0
inet6 addr: fe80::f21f:afff:fece:707a/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:23297904 errors:0 dropped:97 overruns:0 frame:0
TX packets:50383900 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:5957295980 (5.9 GB) TX bytes:20211685928 (20.2 GB)
Interrupt:35

openflow@os-compute3:~/devstack$ ifconfig em1
em1 Link encap:Ethernet HWaddr 74:86:7a:f2:e7:63
inet addr:10.16.148.32 Bcast:10.16.255.255 Mask:255.255.0.0
inet6 addr: fe80::7686:7aff:fef2:e788/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:23623697 errors:0 dropped:79 overruns:0 frame:0
TX packets:49663100 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:5740824268 (5.7 GB) TX bytes:19701170069 (19.7 GB)
Interrupt:35
```

- Dell ODL Controller

```
dell-odl-controller@dell-odl:~$ ifconfig eth0
eth0 Link encap:Ethernet HWaddr 00:0c:29:0f:fa:52
inet addr:10.16.148.232 Bcast:10.16.255.255 Mask:255.255.0.0
inet6 addr: fe80::20c:29ff:fe0f:fa52/64 Scope:Link UP
BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:589521382 errors:0 dropped:79 overruns:0 frame:0
TX packets:287753454 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:305252147233 (305.2 GB) TX bytes:72887415242 (72.8 GB)
```

- Switch Configuration

  ```
  s4810-controller# show running-config interface managementethernet 0/0
  !
  interface ManagementEthernet 0/0
  ip address 10.16.148.97/16
  no shutdown
  s4810-controller#
  ```

Data Network Configuration

The following provides information on the OpenStack controller, OpenStack compute, and switch configuration for the data network.

- OpenStack Controller

  ```
  openflow@os-controller$ ifconfig em2
  em2 Link encap:Ethernet HWaddr c8:1f:66:da:3e:94
  inet addr:20.1.1.2 Bcast:20.1.1.255 Mask:255.255.255.0
  inet6 addr: fe80::ca1f:66ff:feda:3e94/64 Scope:Link
  UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
  RX packets:2583875 errors:0 dropped:13 overruns:0 frame:0
  TX packets:2560468 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:372971418 (372.9 MB) TX bytes:371173086 (371.1 MB)
  Interrupt:38
  ```
- OpenStack Compute

  ```
  openflow@os-compute1:~/devstack$ ifconfig em2
  em2 Link encap:Ethernet HWaddr f0:1f:af:ce:70:7b
  inet addr:20.1.1.3 Bcast:20.1.1.255 Mask:255.255.255.0
  inet6 addr: fe80::f21f:afff:fece:707b/64 Scope:Link UP
  BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
  RX packets:2578290 errors:0 dropped:13 overruns:0 frame:0
  TX packets:2554236 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:374640748 (374.6 MB) TX bytes:373064289 (373.0 MB)
  Interrupt:38

  openflow@os-compute2:~/devstack$ ifconfig em2
  em2 Link encap:Ethernet HWaddr 74:86:7a:f2:e7:89
  inet addr:20.1.1.4 Bcast:20.1.1.255 Mask:255.255.255.0
  inet6 addr: fe80::7686:7aff:fef2:e789/64 Scope:Link UP
  BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
  RX packets:2576644 errors:0 dropped:13 overruns:0 frame:0
  TX packets:2564565 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:374403894 (374.4 MB) TX bytes:371055986 (371.0 MB)
  Interrupt:38

  openflow@os-compute3:~/devstack$ ifconfig em2
  em2 Link encap:Ethernet HWaddr 74:86:7a:f2:e7:62
  inet addr:20.1.1.5 Bcast:20.1.1.255 Mask:255.255.255.0
  inet6 addr: fe80::7686:7aff:fef2:e788/64 Scope:Link UP
  BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
  RX packets:23623697 errors:0 dropped:79 overruns:0 frame:0
  TX packets:49663100 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000  RX bytes:5740824268 (5.7 GB) TX bytes:
  19701170069 (19.7 GB)
  Interrupt:35
  ```
- Switch configuration

  ```
  s4810-controller# show running-config interface tengigabitethernet 0/0
  !
  ```

```
interface TenGigabitEthernet 0/0
description Connected to os-compute1 em2 interface
no ip address
switchport
no shutdown
s4810-controller#show running-config interface tengigabitethernet 0/2
!
interface TenGigabitEthernet 0/2
description Connected to os-controller1 em2
interface
no ip address
switchport
no shutdown
s4810-controller#show running-config interface tengigabitethernet 0/3
!
interface TenGigabitEthernet 0/3
description Connected to os-compute2 em2 interface
no ip address
switchport
no shutdown
s4810-controller#show running-config interface tengigabitethernet 0/4
!
interface TenGigabitEthernet 0/4
description Connected to os-compute3 em2 interface
no ip address
switchport
no shutdown
s4810-controller#show running-config interface vlan 2
!
interface Vlan 2
description Data Network
ip address 20.1.1.1/24
untagged TenGigabitEthernet 0/0-4
no shutdown
s4810-controller#
```

External Network Configuration

The following provides information on the OpenStack controller and compute, and switch configuration for the external network configuration.

- OpenStack Controller and Compute

  em3 interface should be physically connected with specific physical switch, and **em3** port should be in **UP** state
- Switch Configuration

  ```
  s4810-controller#show running-config interface vlan 3
  !
  interface Vlan 3
  description external network
  ip address 1.1.1.1/24
  untagged TenGigabitEthernet 0/0,2-4
  arp timeout 1
  no shutdown
  s4810-controller#show running-config interface tengigabitethernet 0/0
  !
  interface TenGigabitEthernet 0/0
  description Connected to os-compute1 em3
  no ip address
  switchport
  no shutdown
  ```

```
s4810-controller#show running-config interface tengigabitethernet 0/2
!
interface TenGigabitEthernet 0/2
description Connected to os-controller em3
no ip address
switchport
no shutdown
s4810-controller# show running-config interface tengigabitethernet 0/3
!
interface TenGigabitEthernet 0/3
description Connected to os-compute2 em3
no ip address
switchport
no shutdown
s4810-controller#show running-config interface tengigabitethernet 0/4
!
interface TenGigabitEthernet 0/4
description Connected to os-compute3 em3
no ip address
switchport
no shutdown
s4810-controller#
```

4

Installation Overview

NOTE: It is essential that Dell ODL is installed and running before you start OpenStack services — either through devstack or restarting services.

To install and configure the Dell ODL controller:

1. Dell ODL Controller Installation
2. OpenStack Installation
3. OpenStack Configuration
4. OpenStack Updates
5. Upgrading an Existing OpenStack Installation
6. Cleaning up an Existing OpenStack Configuration
7. Configuring the ODL ML2 Plugin
8. Verifying Installation

Dell ODL Controller Installation

To install the Dell ODL controller:

1. Download **Dell-ODL-1.0.0.0.tar.gz** from the Dell ODL release page, untar the release file, and change to the Dell-ODL-1.0.0.0 directory.

```
odluser@administrator-PowerEdge-M915:~/odl$
odluser@administrator-PowerEdge-M915:~/odl$ wget http://<location>/Dell-
ODL-1.0.0.0.tar.gz
odluser@administrator-PowerEdge-M915:~/odl$ ls Dell-ODL-1.0.0.0.tar.gz
odluser@administrator-PowerEdge-M915:~/odl$ tar -xvf Dell-ODL-1.0.0.0.tar.gz
.
.
odluser@administrator-PowerEdge-M915:~/odl$ cd Dell-ODL-1.0.0.0/
```

2. Enter `./bin/karaf` to start installation of the Dell ODL controller.

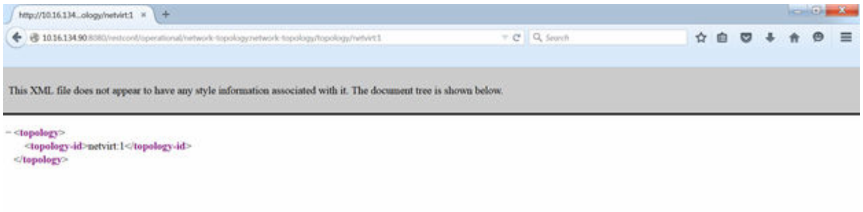
```
odluser@administrator-PowerEdge-M915:~/odl/Dell-ODL-1.0.0.0$ ./bin/karaf
karaf: JAVA_HOME not set; results may vary


    ________
________                ._  .__       .__     __
    \_____  \ ______   ____   ____ \______ \ _____  ___.__.|  | |__| ____
|  |___/  |_
     /   |   \\____ \_/ __ \ /    \ |    |  \\__  \<   |  ||  | |  |/ ___
\|  |  \   __\
    /    |    \  |_> >  ___/|   |  \|    `   \/ __ \\___  ||  |_|  / /_/
>   Y  \  |
    \_______  /   __/ \___  >___|  /_______  (____  / ____||____/__\___  /|
___|  /__|
            \/|__|        \/     \/        \/     \/\/            /
```

```
______/       \/


Hit '<tab> for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown
OpenDaylight.
```

3. Enter `tail —f karaf.log | grep "TriggerUpdates"` to verify initialization of OVSDB.

   ```
   2015-10-19 16:48:00,333 | INFO  | config-pusher    |
   SouthboundHandler                | 259 -
   org.opendaylight.ovsdb.openstack.net-virt - 1.1.1.Lithium-SR1 |
   triggerUpdates
   ```

   a. (Optional) You can use the following REST API to verify initialization of OVSDB: `http://<controller_ip>:8080/restconf/operational/network-topology:network-topology/topology/netvirt:1`

   
   http://10.16.134...ology/netvirt:1
   10.16.134.90:8080/restconf/operational/network-topology:network-topology/topology/netvirt:1
   Search

   This XML file does not appear to have any style information associated with it. The document tree is shown below.

   ```
   - <topology>
       <topology-id>netvirt:1</topology-id>
     </topology>
   ```

   The following shows the expected response:

   ```
   <topology xmlns="urn:TBD:params:xml:ns:yang:network-topology">
   <topology-id>netvirt:1</topology-id>
   </topology>
   ```

OpenStack Devstack Installation

To install the OpenStack devstack:

NOTE: Skip these steps if the OpenStack devstack is installed and configured (see OpenStack Updates).

1. Locate the devstack installation file at *https://github.com/openstack-dev/devstack*, then download the file.
2. Open the hosts file for editing located in */etc/hosts*, then save and close the file.

   ```
   //configure hostname to IP mapping for all the nodes

   <IP_address><control_node_hostname>

   <IP_address><compute_node_1_hostname>

   ...

   <IP_address><compute_node_n_hostname>
   ```
3. Open the `local.conf` file for the controller/network for editing, make any necessary changes, then save and close the file.
4. Open the `local.conf.controller` file for editing, make any necessary changes, then save and close the file.
5. Open the `local.conf.compute` file for editing, make any necessary changes, then save and close the file.

6. Go to the devstack folder, then execute the `stack.sh` script on all the nodes. Once the script has executed, OpenStack should be up and running.

For more information on configuration files, see local.conf for the Compute Node, local.conf for the Controller Node, and plugin.sh.

OpenStack Configuration

The following topics provide configuration information for the compute and controller nodes:

- Compute Node Configuration
- Controller Node Configuration

Compute Node Configuration

To configure the compute node for the OpenStack configuration:

NOTE: The following must be done on the OpenStack side.

1. Locate and open the **local.conf** file and verify the settings match the following example. Change the settings if they do not match the example.

```
enable_plugin networking-odl https://github.com/stackforge/networking-odl
stable/kilo
HOST_IP=<CURRENT_MACHINE_IP>
HOST_NAME=<HOSTNAME_MACHINE>
SERVICE_HOST=<OPENSTACK_CONTROLLER_IP>
SERVICE_HOST_NAME=<OPENSTACK_CONTROLLER_HOSTNAME>
ODL_MODE=compute
ODL_MGR_IP=<ODL_CONTROLLER_IP>
ODL_LOCAL_IP=<IP_OF_DATA_NETWORK>
Q_PLUGIN=ml2
Q_ML2_TENANT_NETWORK_TYPE=vxlan
ENABLE_TENANT_TUNNELS=True
# Use the following to automatically add eth1 to br-ex
PUBLIC_INTERFACE=<EXTERNALGATEWAYINTERFACE>
ODL_L3=True
ODL_PROVIDER_MAPPINGS=br-ex:<EXTERNALGATEWAYINTERFACE>
```

2. Save and close the configuration file if you made changes.

Controller Node Configuration

To configure the OpenStack Controller Node:

1. Locate and open the **local.conf** file, and verify the settings match the following example. Change the settings if they do not match the example.

```
disable_service swift
disable_service center
disable_service n-net
enable_service q-svc
enable_service q-dhcp
enable_service q-meta
enable_service odl-neutron odl-compute
enable_service mysql rabbit
HOST_IP=<CURRENT_MACHINE_IP>
HOST_NAME=<HOSTNAME_OF_MACHINE>
```

```
NEUTRON_CREATE_INITIAL_NETWORKS=False
Q_PLUGIN=ml2
Q_ML2_TENANT_NETWORK_TYPE=vxlan
Q_ML2_PLUGIN_MECHANISM_DRIVERS=opendaylight,logger
ENABLE_TENANT_TUNNELS=True
ODL_MODE=externalodl
enable_plugin networking-odl https://github.com/stackforge/networking-odl
stable/kilo
ODL_NETVIRT_DEBUG_LOGS=True
ODL_MGR_IP=<ODL_CONTROLLER_IP>
ODL_PORT=8080
ODL_BOOT_WAIT=123
ODL_LOCAL_IP=<IP_OF_DATA_NETWORK>
ODL_L3=True
```

2. Save and close the configuration if you made changes.

OpenStack Updates

NOTE: Follow the steps after installation and before `stack.sh`.

The easiest way to update the OpenStack configuration is to download a new file, and simply overwrite the existing file.

1. Open */opt/stack/networking-odl/devstack/***plugin.sh**.
2. Save the files as **plugin.sh** in the same path to overwrite the file.

NOTE: Each time the `reclone=yes` is set in the *local.conf* file, the configuration file may be overwritten. You may need to download the configuration file again.

If you are upgrading an existing OpenStack installation, see Upgrading an Existing OpenStack Installation.

If you are not upgrading an existing OpenStack installation, you are now ready to configure the ML2 plugin (see Configuring the ODL ML2 Plugin).

Upgrading an Existing OpenStack Installation

To use Dell ODL in an existing OpenStack installation with VMs and networks based on a ML2 driver, you must first cleanup your configuration (see Cleaning Up an Existing Configuration, and configure the ML2 plugin (see Configuring the ODL ML2 Plugin.

Cleaning Up an Existing Configuration

To cleanup an existing configuration:

1. If any VMs exist on the compute nodes, delete the nodes through the Horizon or command line.
2. Delete all networks/routers through the OpenStack Horizon command line.
3. Stop the neutron service.

4. Follow the prompts for each compute node to cleanup leftover configuration, then restart the openvswitch.

```
nvo@compute-2:~$ sudo service openvswitch-switch stop
openvswitch-switch stop/waiting
nvo@compute-2:~$
nvo@compute-2:~$
nvo@compute-2:~$ cd /etc/openvswitch/
nvo@compute-2:/etc/openvswitch$
nvo@compute-2:/etc/openvswitch$ sudo rm -f system-id.conf conf.db

nvo@compute-2:~$ sudo service openvswitch-switch start
openvswitch-switch start/running
nvo@compute-2:~$
nvo@compute-2:~$ sudo ovs-vsctl show
3dbd3354-5d37-46c2-a5cb-afbbbbeecb9b
    ovs_version: "2.3.2"
nvo@compute-2:~$
```

5. Start the neuron service.

```
^Codser@administrator-PowerEdge-M915:/opt/stack/networking-odl$ python /usr/
locol/bin/neutron-server --config-file /etc/neutron/neutron.conf --config-
file /etc/neutron/plugins/ml2/ml2_conf.ini & echo $! >/opt/stack/status/
stack/q-svc.pid; fg || echo "q-svc failed to start" | tee "/opt/stack/
status/stack/q-svc.failure"
```

Configuring the ODL ML2 Plugin

This topic explains how to configure the ODL ML2 plugin.

In a devstack-based environment, you must edit two files to setup the ODL ML2 plugin — *neutron.conf* and *ml2_conf.ini*.

NOTE: The Dell ODL is based on the ODL SDN controller.

1. Stop the neutron service before changing the configuration.
2. Open the *neutron.conf* file for editing located in */etc/neutron/neutron.conf*.
3. Set the following variables, then save and close the file.

```
service_plugins=networking_odl.l3.l3_odl.OpenDayLightL3RouterPlugin
```

```
core_plugin=neutron.plugins.ml2.plugin.Ml2Plugin
```

4. Open the *ml2_conf.ini* file for editing located in */etc/neutron/plugins/ml2/ml2_conf.ini*.
5. Set the following variables, then save and close the file:

```
tenant_network_types=vxlan
type_drivers=local,flat,vlan,gre,vxlan
mechanism_drivers=opendaylight,logger
[ovs]Local_ip=20.1.1.2 (IP of the Eth interface corresponding to the data
network)
[ml2_odl]password=admin
[ml2_odl]username=admin
[ml2_odl]url=http://<ODL_CONTROLLER_IP>:8080/controller/nb/v2/neutron
```

6. Re-enable the neutron service after both configuration files have been edited and saved.

   For information about setting up the neutron network service in control and compute node, see Useful Links.

Verifying Installation

To verify installation:

1. After `stack.sh` and the following commands in the compute nodes, indicate that the compute nodes have been detected correctly by the OpenStack through the Dell ODL controller.

   ```
   nvo@compute-2:~$
   nvo@compute-2:~$ sudo ovs-vsctl show
   3dbd3354-5d37-46c2-a5cb-afbbbbeecb9b
       Manager "tcp:10.16.148.232:6640"
           is_connected: true
       Bridge br-int
           Controller "tcp:10.16.148.232:6653"
               is_connected: true
           fail_mode: secure
           Port br-int
               Interface br-int
                   type: internal
       Bridge br-ex
           Controller "tcp:10.16.148.232:6653"
               is_connected: true
           fail_mode: secure
           Port "em3"
               Interface "em3"
           Port br-ex
               Interface br-ex
                   type: internal
       ovs_version: "2.3.2"
   nvo@compute-2:~$
   ```

2. Verify the flow programming in each of the compute nodes, as shown in the following example.

   ```
   nvo@compute-2:~$
   nvo@compute-2:~$ sudo ovs-ofctl -O Openflow13 dump-flows br-int
   OFPST_FLOW reply (OF1.3) (xid=0x2):
    cookie=0x0, duration=87.397s, table=0, n_packets=0, n_bytes=0, priority=0
   actions=goto_table:20
    cookie=0x0, duration=91.368s, table=0, n_packets=0, n_bytes=0,
   dl_type=0x88cc actions=CONTROLLER:65535
    cookie=0x0, duration=87.389s, table=20, n_packets=0, n_bytes=0, priority=0
   actions=goto_table:30
    cookie=0x0, duration=87.382s, table=30, n_packets=0, n_bytes=0, priority=0
   actions=goto_table:40
    cookie=0x0, duration=87.375s, table=40, n_packets=0, n_bytes=0, priority=0
   actions=goto_table:50
    cookie=0x0, duration=87.369s, table=50, n_packets=0, n_bytes=0, priority=0
   actions=goto_table:60
    cookie=0x0, duration=87.361s, table=60, n_packets=0, n_bytes=0, priority=0
   actions=goto_table:70
    cookie=0x0, duration=87.353s, table=70, n_packets=0, n_bytes=0, priority=0
   actions=goto_table:80
    cookie=0x0, duration=87.349s, table=80, n_packets=0, n_bytes=0, priority=0
   actions=goto_table:90
    cookie=0x0, duration=87.339s, table=90, n_packets=0, n_bytes=0, priority=0
   actions=goto_table:100
    cookie=0x0, duration=87.332s, table=100, n_packets=0, n_bytes=0,
   priority=0 actions=goto_table:110
    cookie=0x0, duration=87.322s, table=110, n_packets=0, n_bytes=0,
   priority=0 actions=drop
   nvo@compute-2:~$
   ```

```
nvo@compute-2:~$
nvo@compute-2:~$ sudo ovs-ofctl -O Openflow13 dump-flows br-ex
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=117.539s, table=0, n_packets=1, n_bytes=87,
priority=0 actions=NORMAL
 cookie=0x0, duration=117.538s, table=0, n_packets=4, n_bytes=240,
dl_type=0x88cc actions=CONTROLLER:65535
nvo@compute-2:~$
```

Congratulations! You have completed installing and configuring Dell ODL. You are now ready to create a data center network (see Using Horizon for Multi-Tenancy).

5

Using Horizon for Multi-Tenancy

Dell ODL is a software-defined networking (SDN) controller which manages the network entity for OpenStack.

To create and configure your network using the Dell ODL controller and OpenStack:

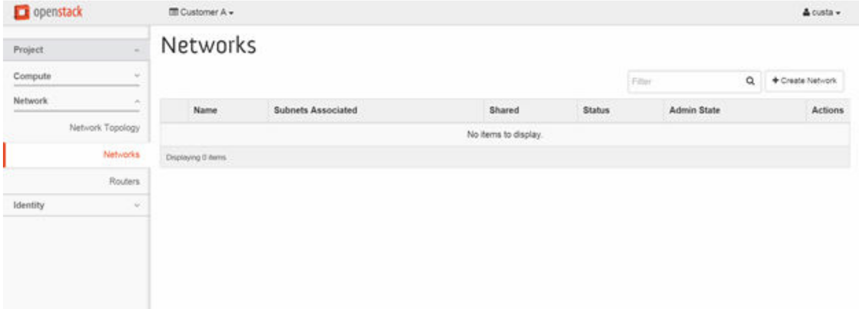
1. Creating a Network
2. Instantiating a VM in a Network
3. Creating a Router
4. Adding an Interface to a Router
5. Creating an External Network
6. Assigning a Floating IP to a VM

For sample customer configurations, see Testing Multi-Tenancy.

Creating a Network

To create a network:

1. Log into Horizon with **admin** credentials.
2. Open Horizon, select **Network** from the left, then select **Networks**.

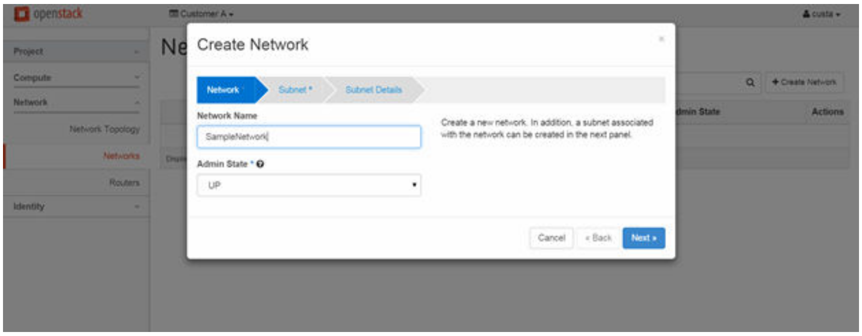


Networks

Name	Subnets Associated	Shared	Status	Admin State	Actions
No items to display.					

Displaying 0 items

3. Click **Create Network**.

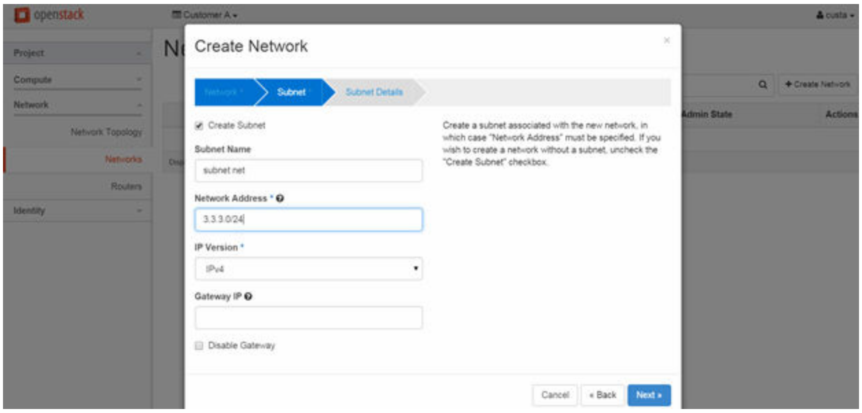


Network Name: SampleNetwork

Admin State *: UP

Create a new network. In addition, a subnet associated with the network can be created in the next panel.

4. You are now ready to create the subnet. Select the **Subnet** tab, enter the subnet name and address, then click **Next**.



☑ Create Subnet

Subnet Name: subnet net

Network Address *: 3.3.3.0/24

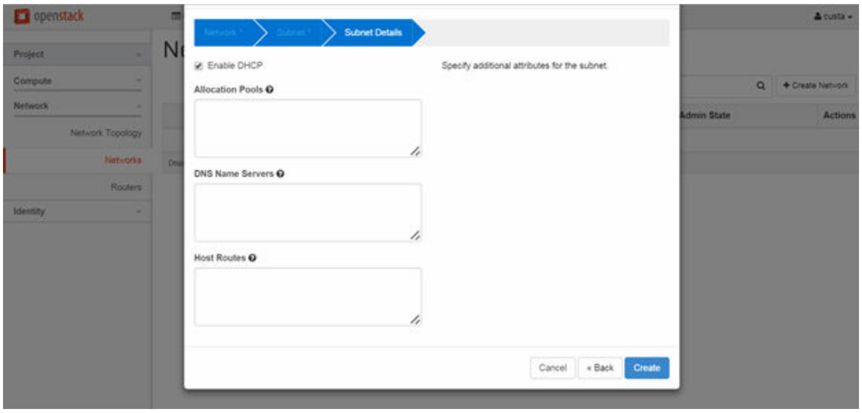
IP Version *: IPv4

Gateway IP:

☐ Disable Gateway

Create a subnet associated with the new network, in which case "Network Address" must be specified. If you wish to create a network without a subnet, uncheck the "Create Subnet" checkbox.

5. Verify that **Enable DHCP** is checked, and click **Create**.



☑ Enable DHCP

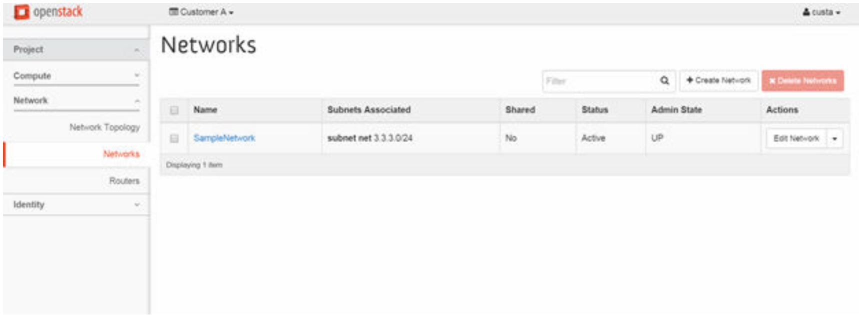
Allocation Pools:

DNS Name Servers:

Host Routes:

Specify additional attributes for the subnet.

After successful network creation, the new network displays on the Network page, similar to the following example.



Networks

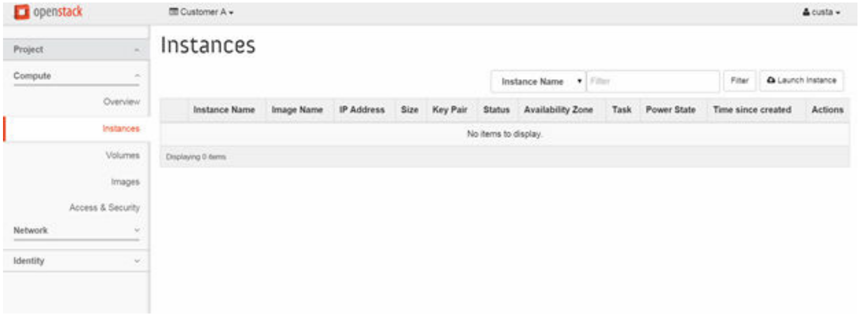
	Name	Subnets Associated	Shared	Status	Admin State	Actions
□	SampleNetwork	**subnet net** 3.3.3.0/24	No	Active	UP	Edit Network

Displaying 1 item

Instantiating a VM in a Network

To instantiate a virtual machine (VM) in the new network:

1. Select **Instances** from the left to create a VM.



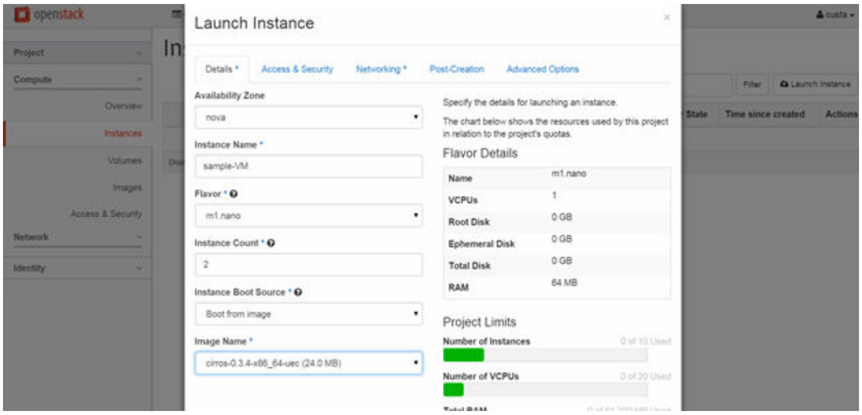
Instances

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
No items to display.										

Displaying 0 items

2. Click **Launch Instance**, and then complete the details.

   If you select the **Boot from Image** option from the Instance Boot Source, the default image list in the **Image Name** area displays. Select any image from the drop-down of the image name.



Launch Instance

Availability Zone: nova

Instance Name *: sample-VM

Flavor *: m1.nano

Instance Count *: 2

Instance Boot Source *: Boot from image

Image Name *: cirros-0.3.4-x86_64-uec (24.0 MB)

Specify the details for launching an instance.

The chart below shows the resources used by this project in relation to the project's quotas.

Flavor Details

Name	m1.nano
VCPUs	1
Root Disk	0 GB
Ephemeral Disk	0 GB
Total Disk	0 GB
RAM	64 MB

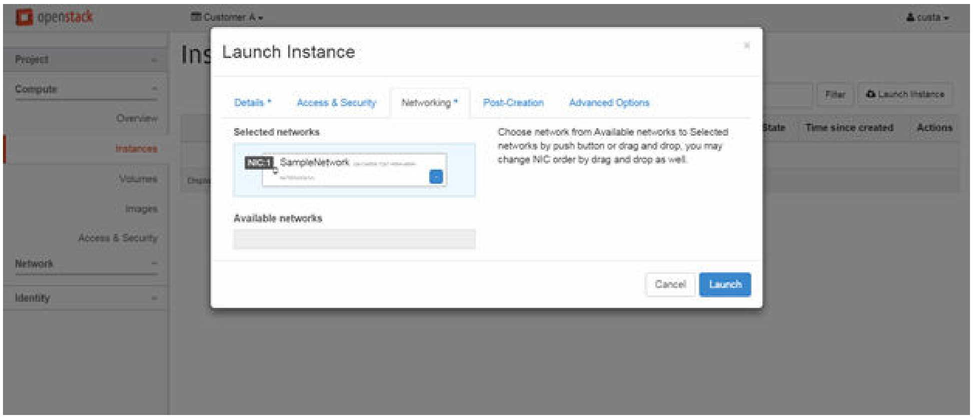
Project Limits

Number of Instances — 0 of 10 Used

Number of VCPUs — 0 of 20 Used

Total RAM

3. From the **Network** tab, select the created network (such as **SampleNetwork**), and then click **Launch**.



Launch Instance

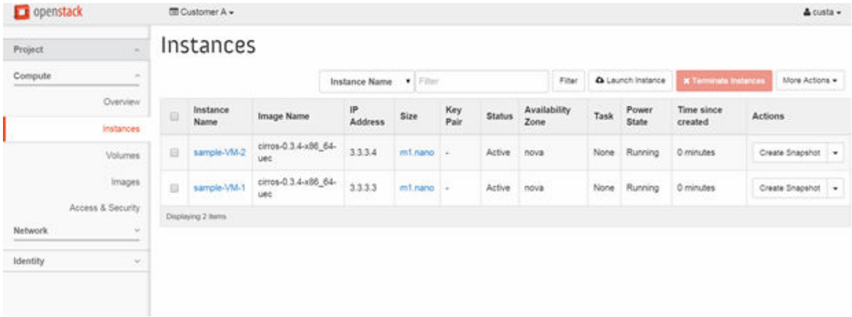
Selected networks

SampleNetwork

Available networks

Choose network from Available networks to Selected networks by push button or drag and drop, you may change NIC order by drag and drop as well.

The following example shows the created VMs.



Instances

	Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
□	sample-VM-2	cirros-0.3.4-x86_64-uec	3.3.3.4	m1.nano	-	Active	nova	None	Running	0 minutes	Create Snapshot
□	sample-VM-1	cirros-0.3.4-x86_64-uec	3.3.3.3	m1.nano	-	Active	nova	None	Running	0 minutes	Create Snapshot

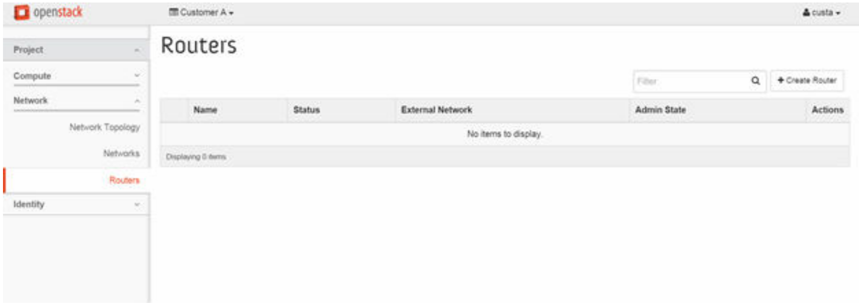
Displaying 2 items

4. Select **Network Topology** to view the VMs and networks.

Creating a Router

To create a router:

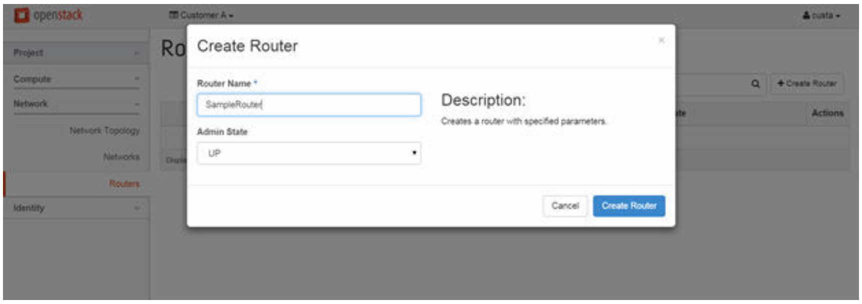
1. Select **Routers** from the left to create a router.



Name	Status	External Network	Admin State	Actions
No items to display.				

Displaying 0 items

2. Enter the router name and admin state, then click **Create Router**.



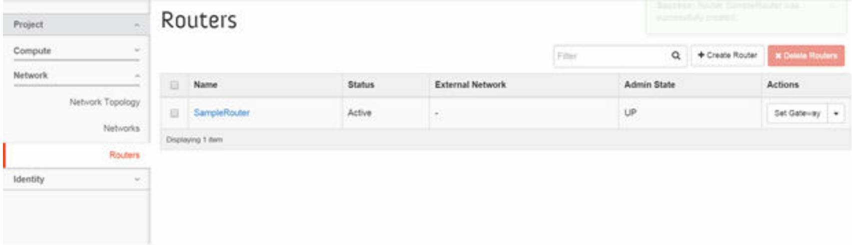
Router Name * SampleRouter

Admin State UP

Description:

Creates a router with specified parameters.

The following example shows the new router.



Success: Router SampleRouter was successfully created.

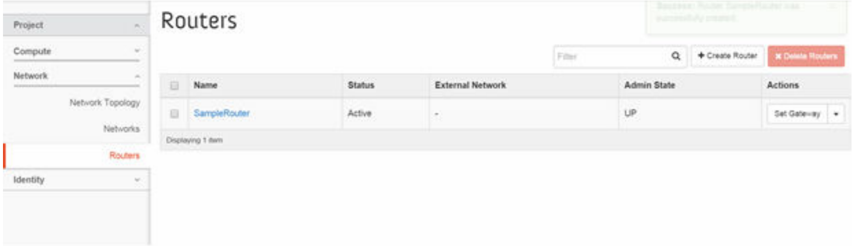
□	Name	Status	External Network	Admin State	Actions
□	SampleRouter	Active	-	UP	Set Gateway ▾

Displaying 1 item

Adding an Interface to a Router

To add an interface to a new router:

1. Select **Router** from the left, then select the **SampleRouter** link.

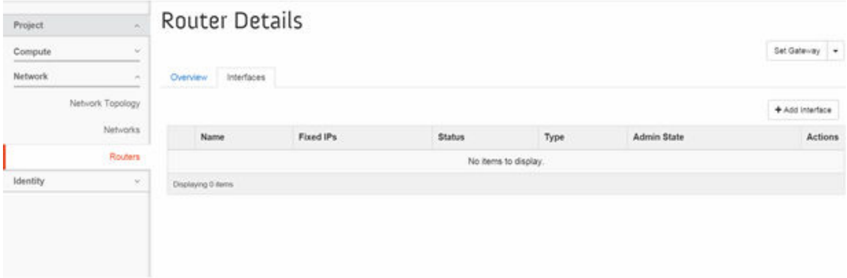


Success: Router SampleRouter was successfully created.

□	Name	Status	External Network	Admin State	Actions
□	SampleRouter	Active	-	UP	Set Gateway ▾

Displaying 1 item

2. Select the **Interfaces** tab, then click **Add Interface**.



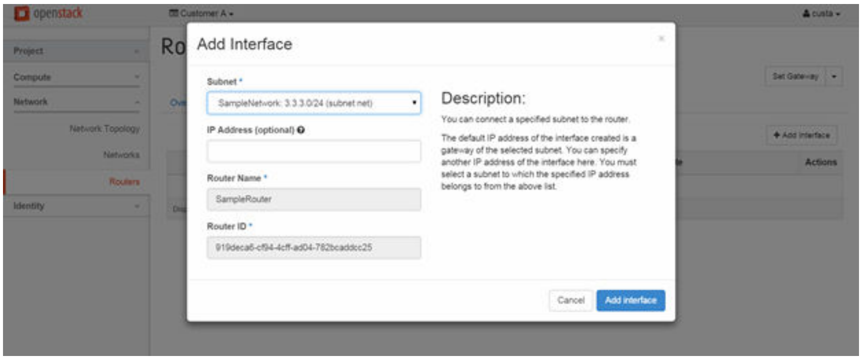
Router Details

Name	Fixed IPs	Status	Type	Admin State	Actions
No items to display.					

Displaying 0 items

3. Select the subnet of the router that you created.

   If the network is not shown in the drop-down, go back and create the network (see Creating a Network).



Add Interface

Subnet *: SampleNetwork: 3.3.3.0/24 (subnet net)

IP Address (optional)

Router Name *: SampleRouter

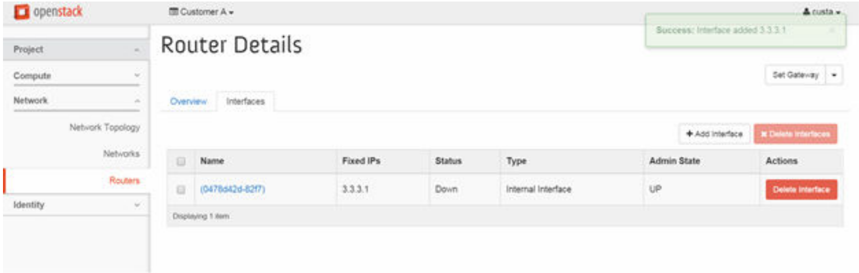
Router ID *: 919deca6-cf94-4cff-ad04-782bcaddcc25

Description:

You can connect a specified subnet to the router.

The default IP address of the interface created is a gateway of the selected subnet. You can specify another IP address of the interface here. You must select a subnet to which the specified IP address belongs to from the above list.

4. Click **Add Interface** to add additional interfaces to your network, if desired.



Success: Interface added 3.3.3.1

Router Details

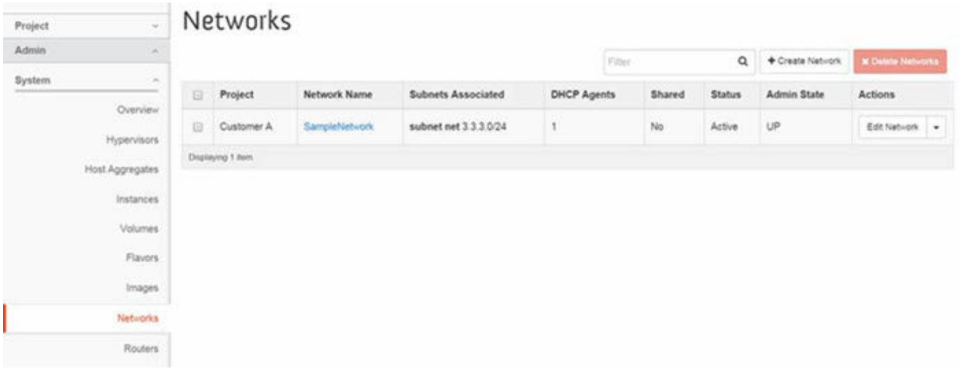
☐	Name	Fixed IPs	Status	Type	Admin State	Actions
☐	(0478d42d-82f7)	3.3.3.1	Down	Internal Interface	UP	Delete Interface

Displaying 1 item

Creating an External Network

To create an external network:

1. Select **Networks** under System, then click **Create Network**.

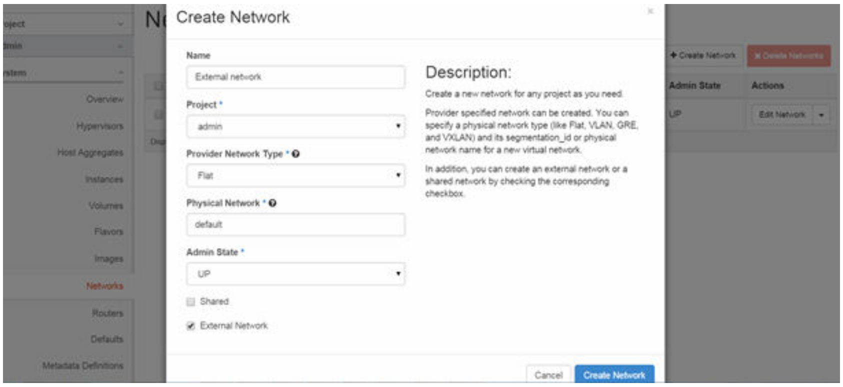


Networks

□	Project	Network Name	Subnets Associated	DHCP Agents	Shared	Status	Admin State	Actions
□	Customer A	SampleNetwork	**subnet net** 3.3.3.0/24	1	No	Active	UP	Edit Network ▾

Displaying 1 item

2. Select the network options as shown in the following example, then click **Create Network**.



Create Network

Name: External network

Project *: admin

Provider Network Type *: Flat

Physical Network *: default

Admin State *: UP

□ Shared

☑ External Network

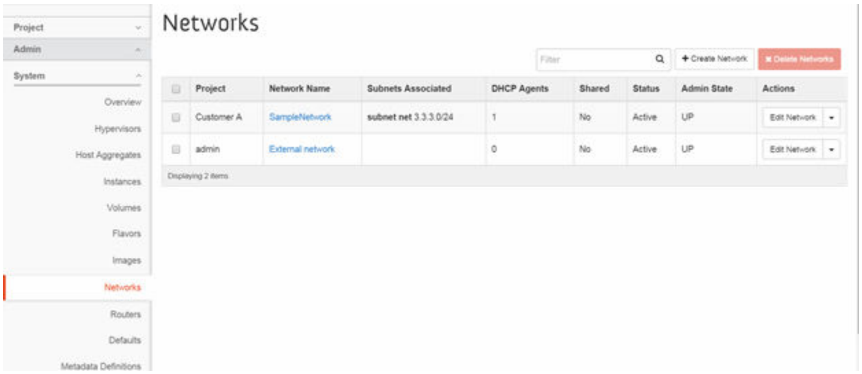
Description:

Create a new network for any project as you need.

Provider specified network can be created. You can specify a physical network type (like Flat, VLAN, GRE, and VXLAN) and its segmentation_id or physical network name for a new virtual network.

In addition, you can create an external network or a shared network by checking the corresponding checkbox.

The following example shows the new external network.

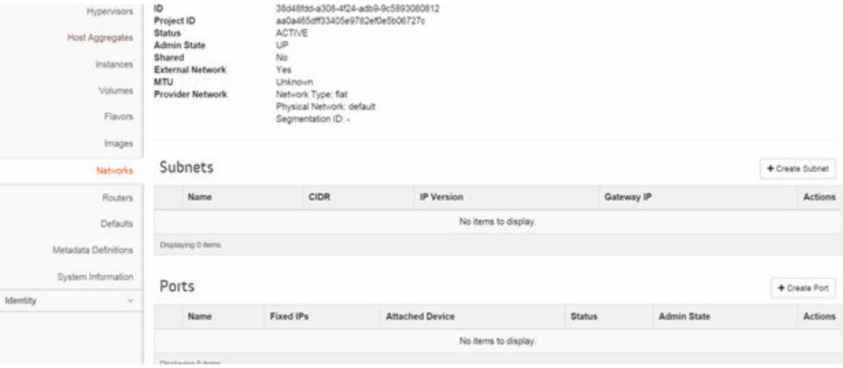


Networks

□	Project	Network Name	Subnets Associated	DHCP Agents	Shared	Status	Admin State	Actions
□	Customer A	SampleNetwork	**subnet net** 3.3.3.0/24	1	No	Active	UP	Edit Network ▾
□	admin	External network		0	No	Active	UP	Edit Network ▾

Displaying 2 items

3. Select the new external network, then click **Create Subnet**.



ID	38d48fdd-a308-4f24-adb9-9c5893080812
Project ID	aa0a465dff33405e9782ef0e5b06727c
Status	ACTIVE
Admin State	UP
Shared	No
External Network	Yes
MTU	Unknown
Provider Network	Network Type: flat Physical Network: default Segmentation ID: -

Subnets

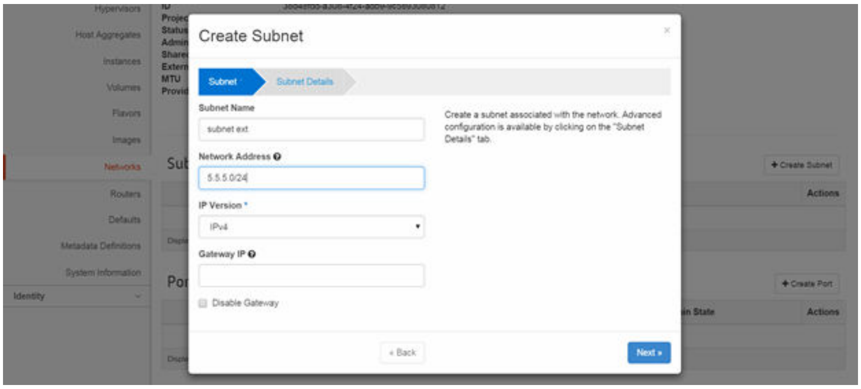
Name	CIDR	IP Version	Gateway IP	Actions
No items to display.				

Displaying 0 items

Ports

Name	Fixed IPs	Attached Device	Status	Admin State	Actions
No items to display.					

4. Enter the network address, then click **Next**.



Create Subnet

Subnet Name: subnet ext

Network Address: 5.5.5.0/24

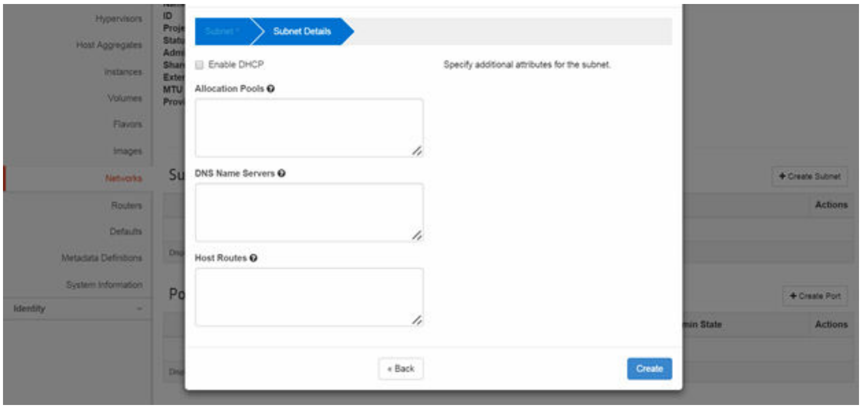
IP Version *: IPv4

Gateway IP:

□ Disable Gateway

Create a subnet associated with the network. Advanced configuration is available by clicking on the "Subnet Details" tab.

5. Disable the DHCP checkbox, then click **Create** to create the subnet for the external network.



□ Enable DHCP

Allocation Pools

DNS Name Servers

Host Routes

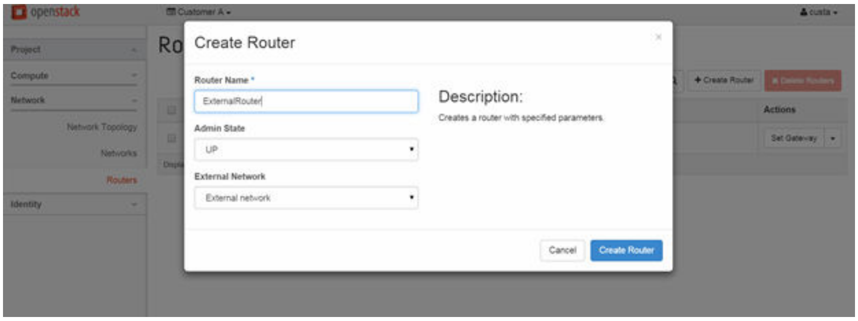
Specify additional attributes for the subnet.

Assigning a Floating IP to a VM

To assign a floating IP to a VM, instead of using DHCP:

NOTE: An external network interface on the router is required to assign a floating IP.

1. Select **Routers** from the Network section, and then select the external network that you previously created in the External Network section.
2. Enter a router name, then click **Create Router** to create the router.



Create Router

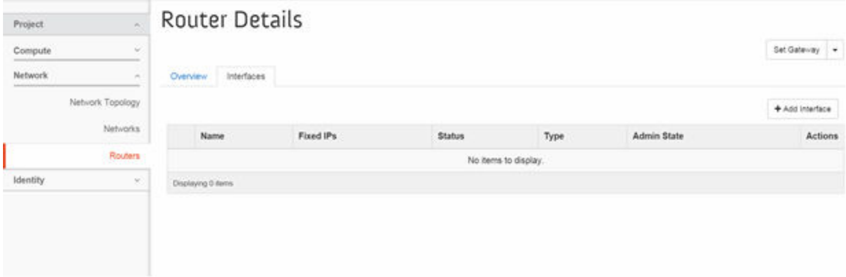
Router Name *
ExternalRouter

Admin State
UP

External Network
External network

Description:
Creates a router with specified parameters.

3. Select the router you just created, select the **Interfaces** tab, then click **Add Interface**.

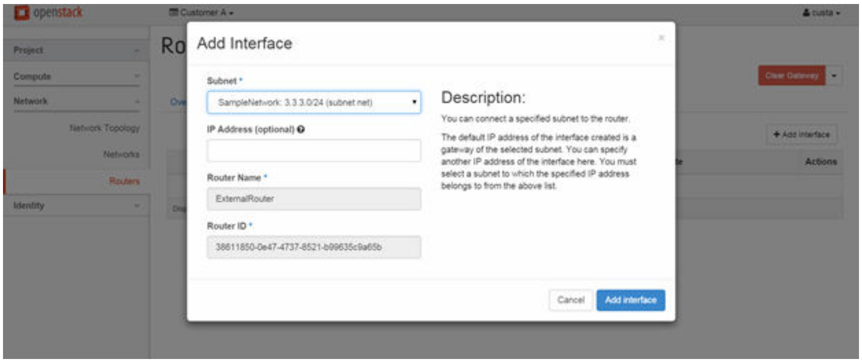


Router Details

Name	Fixed IPs	Status	Type	Admin State	Actions
No items to display.					

Displaying 0 items

4. Select the subnet of the internal network, then click **Add Interface**.



Add Interface

Subnet *
SampleNetwork: 3.3.3.0/24 (subnet net)

IP Address (optional)

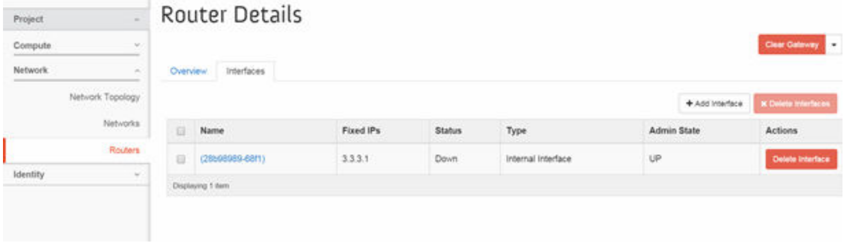
Router Name *
ExternalRouter

Router ID *
38611850-0e47-4737-8521-b99635c9a65b

Description:
You can connect a specified subnet to the router.

The default IP address of the interface created is a gateway of the selected subnet. You can specify another IP address of the interface here. You must select a subnet to which the specified IP address belongs to from the above list.

The interface is now attached to the router, as shown in the following example.

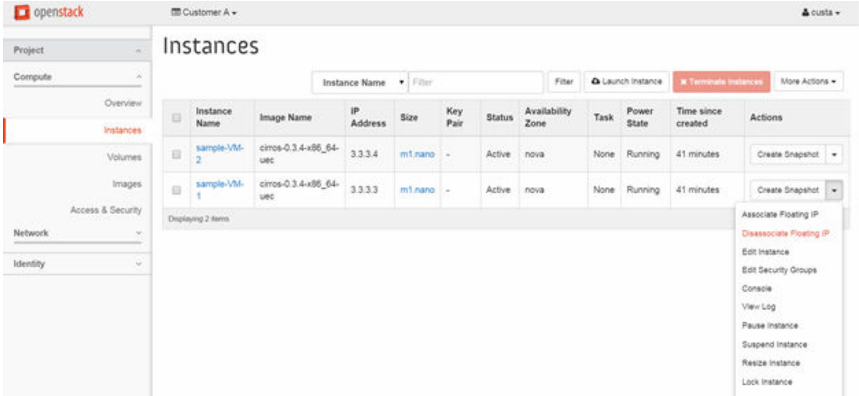


Router Details

□	Name	Fixed IPs	Status	Type	Admin State	Actions
□	(28b98989-68f1)	3.3.3.1	Down	Internal Interface	UP	Delete Interface

Displaying 1 item

5. Select **Instances** from the Compute section, then select **Associate Floating IP** for any VM.

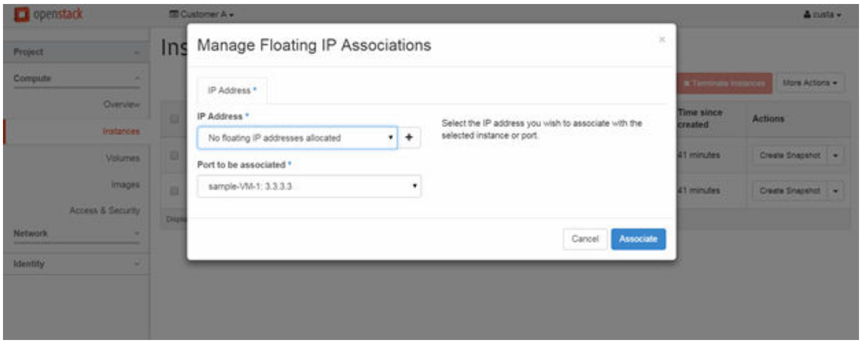


Instances

□	Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
□	sample-VM-2	cirros-0.3.4-x86_64-uec	3.3.3.4	m1.nano	-	Active	nova	None	Running	41 minutes	Create Snapshot
□	sample-VM-1	cirros-0.3.4-x86_64-uec	3.3.3.3	m1.nano	-	Active	nova	None	Running	41 minutes	Create Snapshot

Displaying 2 items

6. Select the + symbol to allocate a floating IP from an existing external network, then click **Associate**.



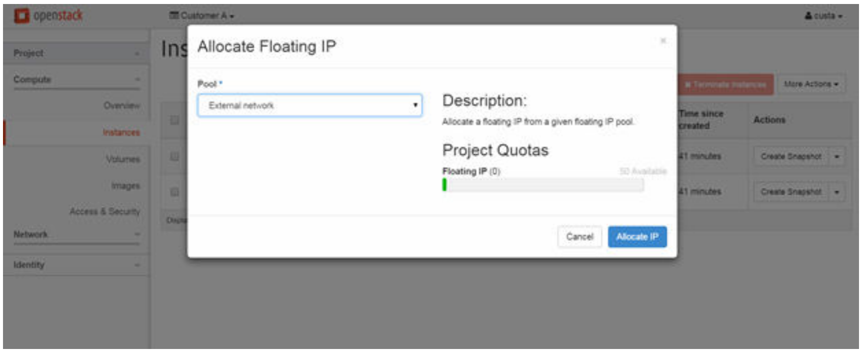
Manage Floating IP Associations

IP Address *: No floating IP addresses allocated

Port to be associated *: sample-VM-1: 3.3.3.3

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

7. Select **External network** if not already selected, then click **Allocate IP**.



Allocate Floating IP

Pool *: External network

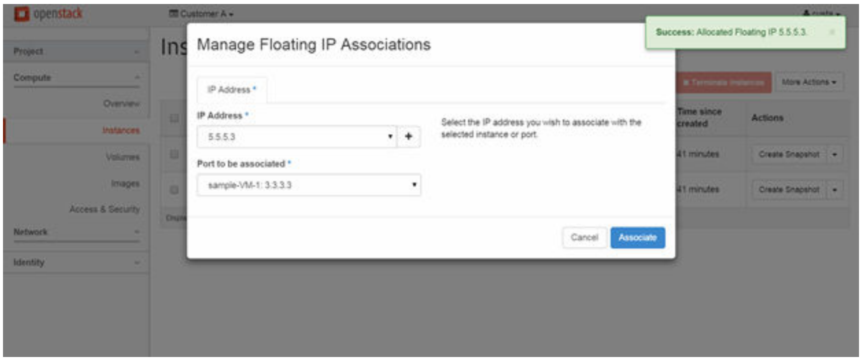
Description:

Allocate a floating IP from a given floating IP pool.

Project Quotas

Floating IP (0) 50 Available

8. Click **Associate** to associate the floating IP with the external network.



Success: Allocated Floating IP 5.5.5.3.

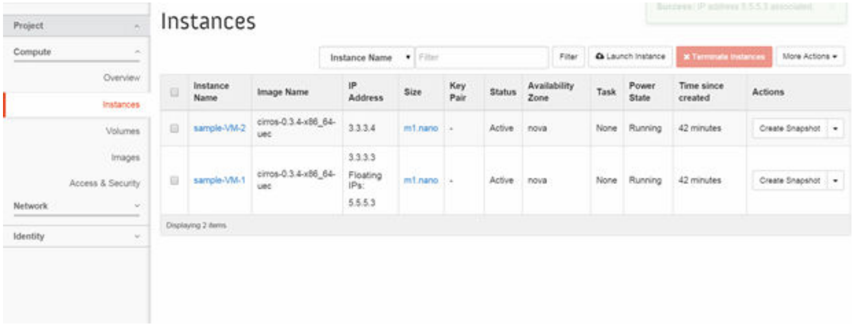
Manage Floating IP Associations

IP Address *: 5.5.5.3

Port to be associated *: sample-VM-1: 3.3.3.3

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

9. Select the IP address section to verify the floating IP is assigned to the VM.

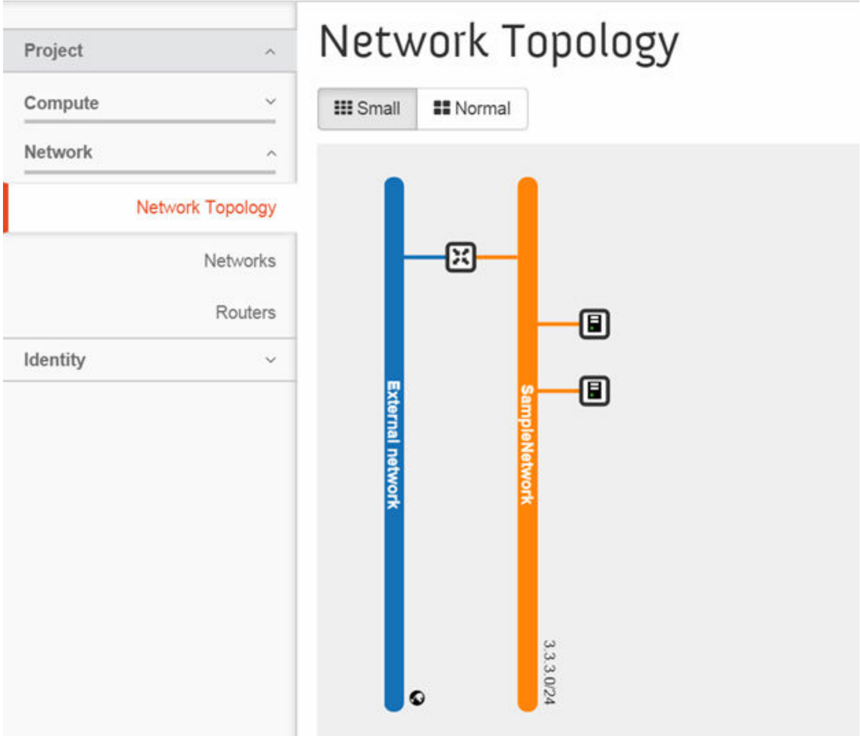


Instances

□	Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
□	sample-VM-2	cirros-0.3.4-x86_64-uec	3.3.3.4	m1.nano	-	Active	nova	None	Running	42 minutes	Create Snapshot ▾
□	sample-VM-1	cirros-0.3.4-x86_64-uec	3.3.3.3 Floating IPs: 5.5.5.3	m1.nano	-	Active	nova	None	Running	42 minutes	Create Snapshot ▾

Displaying 2 items

The following shows the network topology of a VM connected to an external network and router.



Testing Multi-Tenancy

Use the following links to view sample customer configurations:

- Customer A Configuration
- Customer B Configuration
- Customer C Configuration
- Overlap IP Between Tenants

Customer A Configuration

The following shows how Customer A is using Dell ODL:

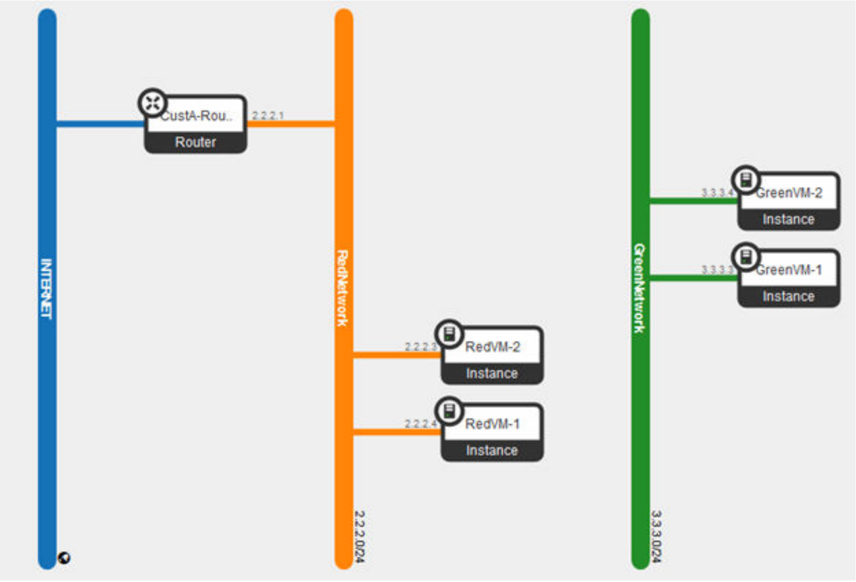
Networks:

- Red Network: 2.2.2.0/24
- Green Network: 3.3.3.0/24
- External Network: 1.1.1.0/24

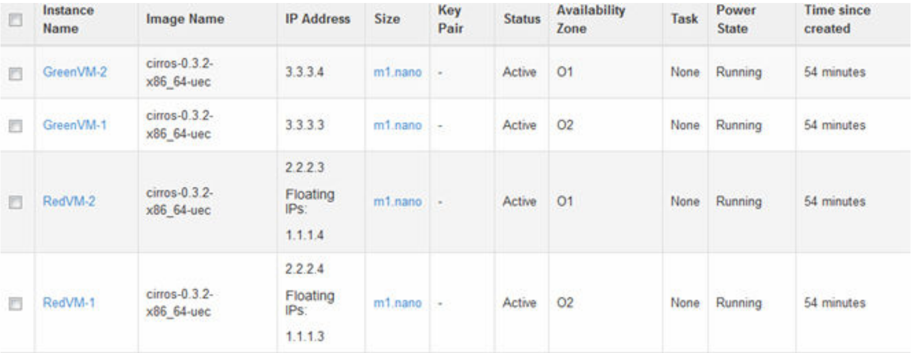
Customer A is using the following for connectivity:

- Same network East-West connectivity in the Green network
- North-South/external connectivity between the Red and Internet network

Following is an example topology for Customer A.

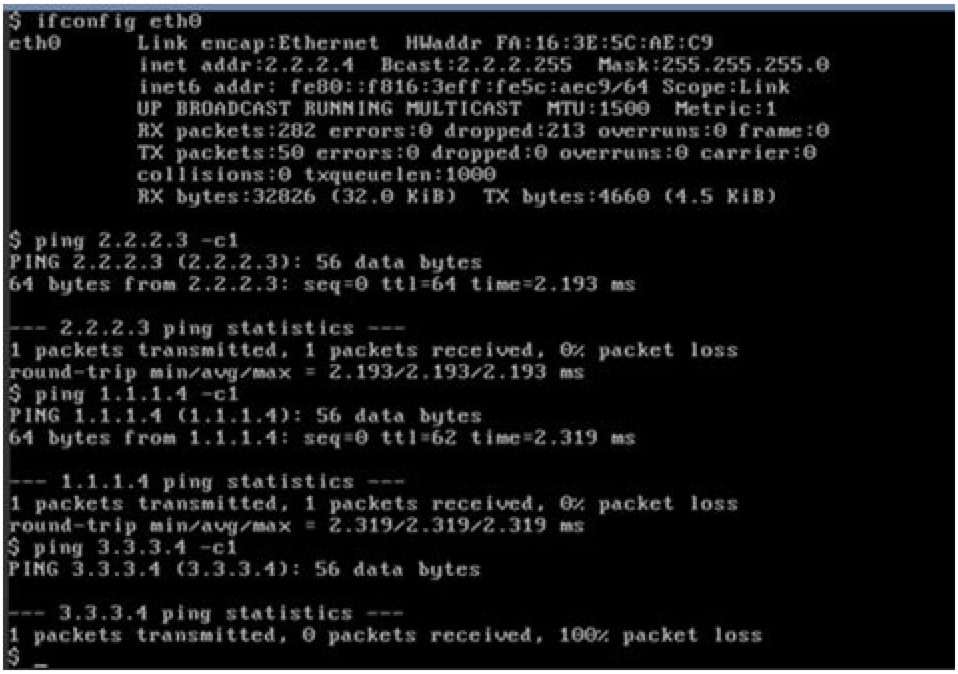


The following example shows VM and floating IP details for Customer A.



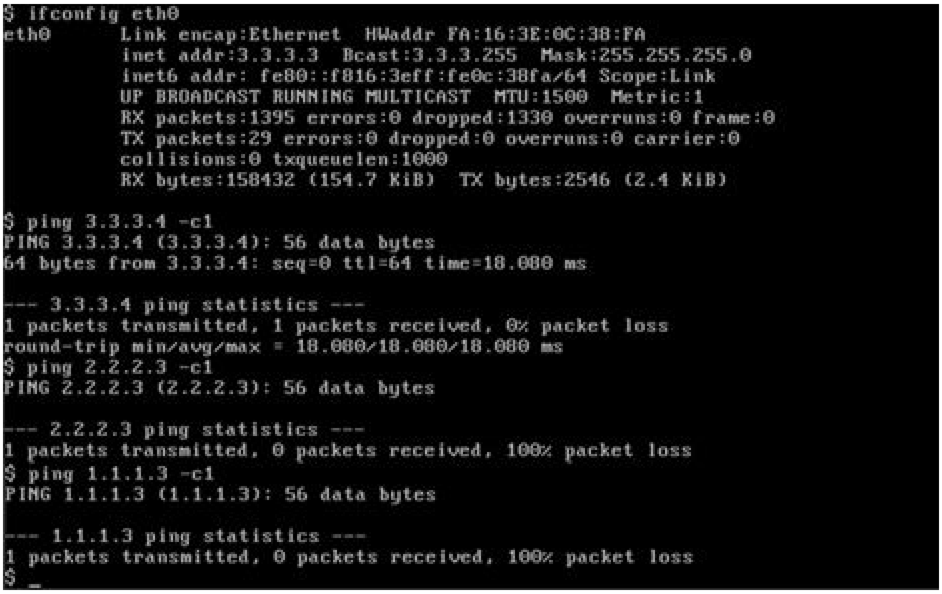
☐	Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created
☐	GreenVM-2	cirros-0.3.2-x86_64-uec	3.3.3.4	m1.nano	-	Active	O1	None	Running	54 minutes
☐	GreenVM-1	cirros-0.3.2-x86_64-uec	3.3.3.3	m1.nano	-	Active	O2	None	Running	54 minutes
☐	RedVM-2	cirros-0.3.2-x86_64-uec	2.2.2.3 Floating IPs: 1.1.1.4	m1.nano	-	Active	O1	None	Running	54 minutes
☐	RedVM-1	cirros-0.3.2-x86_64-uec	2.2.2.4 Floating IPs: 1.1.1.3	m1.nano	-	Active	O2	None	Running	54 minutes

The following example shows the Red network, VM Red VM-1’s reachability to other networks.



```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:5C:AE:C9
          inet addr:2.2.2.4  Bcast:2.2.2.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe5c:aec9/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:282 errors:0 dropped:213 overruns:0 frame:0
          TX packets:50 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:32826 (32.0 KiB)  TX bytes:4660 (4.5 KiB)

$ ping 2.2.2.3 -c1
PING 2.2.2.3 (2.2.2.3): 56 data bytes
64 bytes from 2.2.2.3: seq=0 ttl=64 time=2.193 ms

--- 2.2.2.3 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.193/2.193/2.193 ms
$ ping 1.1.1.4 -c1
PING 1.1.1.4 (1.1.1.4): 56 data bytes
64 bytes from 1.1.1.4: seq=0 ttl=62 time=2.319 ms

--- 1.1.1.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.319/2.319/2.319 ms
$ ping 3.3.3.4 -c1
PING 3.3.3.4 (3.3.3.4): 56 data bytes

--- 3.3.3.4 ping statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
$ _
```

The following example shows the Green network VM Green VM-1’s reachability to other networks.



```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:0C:38:FA
          inet addr:3.3.3.3  Bcast:3.3.3.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe0c:38fa/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1395 errors:0 dropped:1330 overruns:0 frame:0
          TX packets:29 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:158432 (154.7 KiB)  TX bytes:2546 (2.4 KiB)

$ ping 3.3.3.4 -c1
PING 3.3.3.4 (3.3.3.4): 56 data bytes
64 bytes from 3.3.3.4: seq=0 ttl=64 time=18.080 ms

--- 3.3.3.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 18.080/18.080/18.080 ms
$ ping 2.2.2.3 -c1
PING 2.2.2.3 (2.2.2.3): 56 data bytes

--- 2.2.2.3 ping statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
$ ping 1.1.1.3 -c1
PING 1.1.1.3 (1.1.1.3): 56 data bytes

--- 1.1.1.3 ping statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
$ _
```

Customer B Configuration

The following shows how Customer B is using Dell ODL:

Networks:

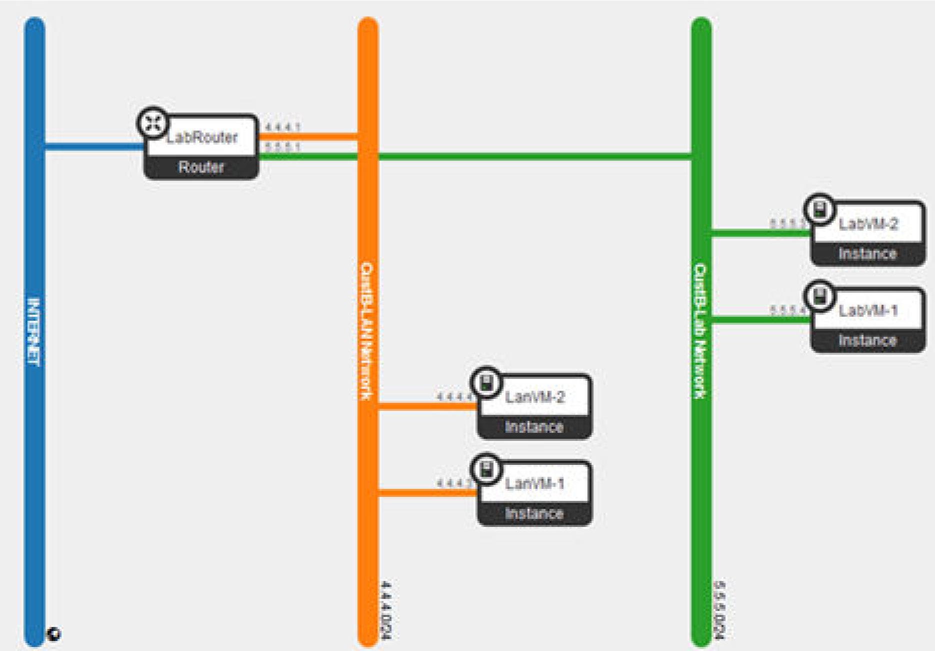
- LAN Networks: 4.4.4.0/24
- Lab Network: 5.5.5.0/24
- External Network: 1.1.1.0/24

The Customer B configuration shows:

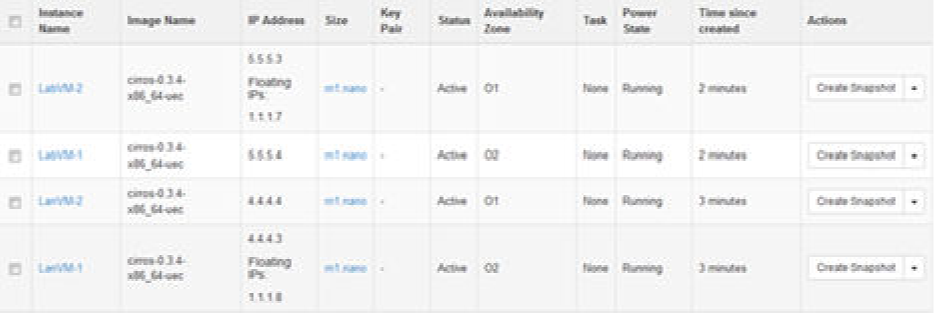
- Different network East-West connectivity between the Lab network and the LAN network.
- North-South/external connectivity for one VM in the Lab network and one VM in the LAN network.

  North-South communication between tenants Customer A and Customer B through the Internet network 1.1.1.0/24
- Addition of new compute node

The following shows an example topology for Customer B:

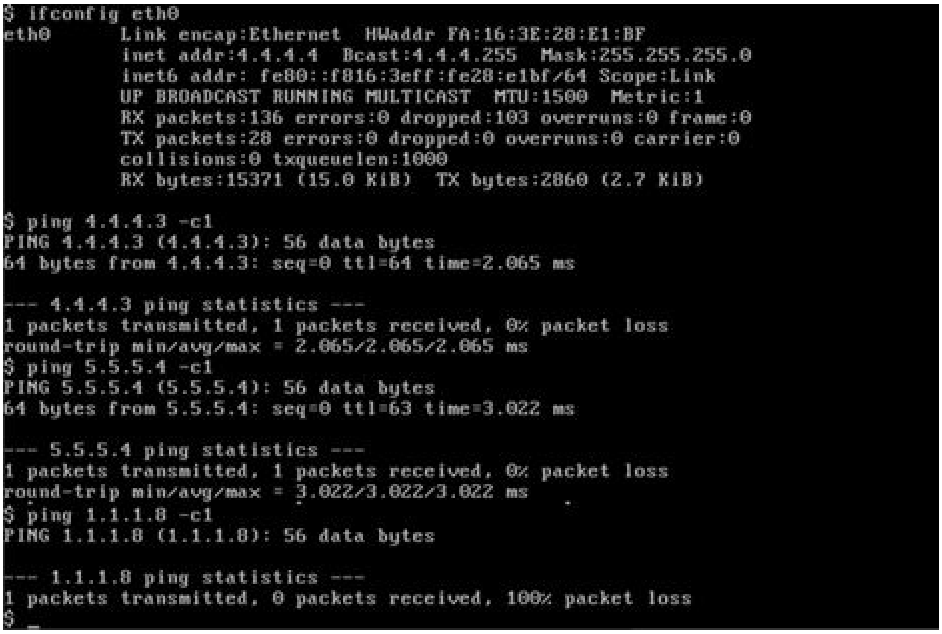


The following shows VM provision and floating IP details for Customer B:



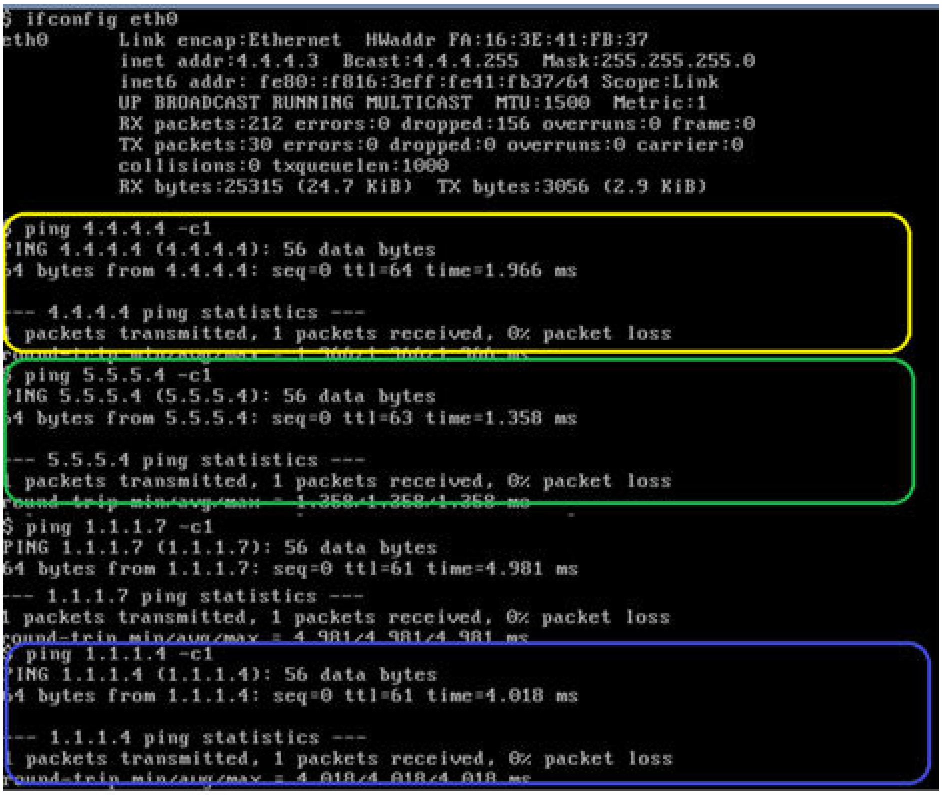
□	Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
□	LabVM-2	cirros-0.3.4-x86_64-uec	5.5.5.3 Floating IPs: 1.1.1.7	m1.nano	-	Active	O1	None	Running	2 minutes	Create Snapshot ▾
□	LabVM-1	cirros-0.3.4-x86_64-uec	5.5.5.4	m1.nano	-	Active	O2	None	Running	2 minutes	Create Snapshot ▾
□	LanVM-2	cirros-0.3.4-x86_64-uec	4.4.4.4	m1.nano	-	Active	O1	None	Running	3 minutes	Create Snapshot ▾
□	LanVM-1	cirros-0.3.4-x86_64-uec	4.4.4.3 Floating IPs: 1.1.1.8	m1.nano	-	Active	O2	None	Running	3 minutes	Create Snapshot ▾

The following shows the LAN network VM LanM-2’s reachability to other networks:



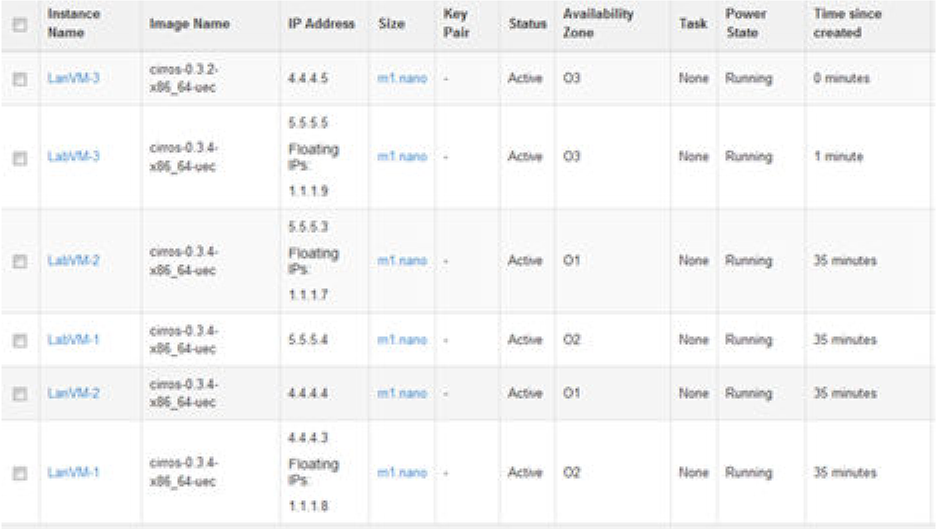
```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:28:E1:BF
          inet addr:4.4.4.4  Bcast:4.4.4.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe28:e1bf/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:136 errors:0 dropped:103 overruns:0 frame:0
          TX packets:28 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:15371 (15.0 KiB)  TX bytes:2860 (2.7 KiB)

$ ping 4.4.4.3 -c1
PING 4.4.4.3 (4.4.4.3): 56 data bytes
64 bytes from 4.4.4.3: seq=0 ttl=64 time=2.065 ms

--- 4.4.4.3 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.065/2.065/2.065 ms
$ ping 5.5.5.4 -c1
PING 5.5.5.4 (5.5.5.4): 56 data bytes
64 bytes from 5.5.5.4: seq=0 ttl=63 time=3.022 ms

--- 5.5.5.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 3.022/3.022/3.022 ms
$ ping 1.1.1.8 -c1
PING 1.1.1.8 (1.1.1.8): 56 data bytes

--- 1.1.1.8 ping statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
$ _
```

The following shows the LAN network VM LanVM-1's reachability to other networks:



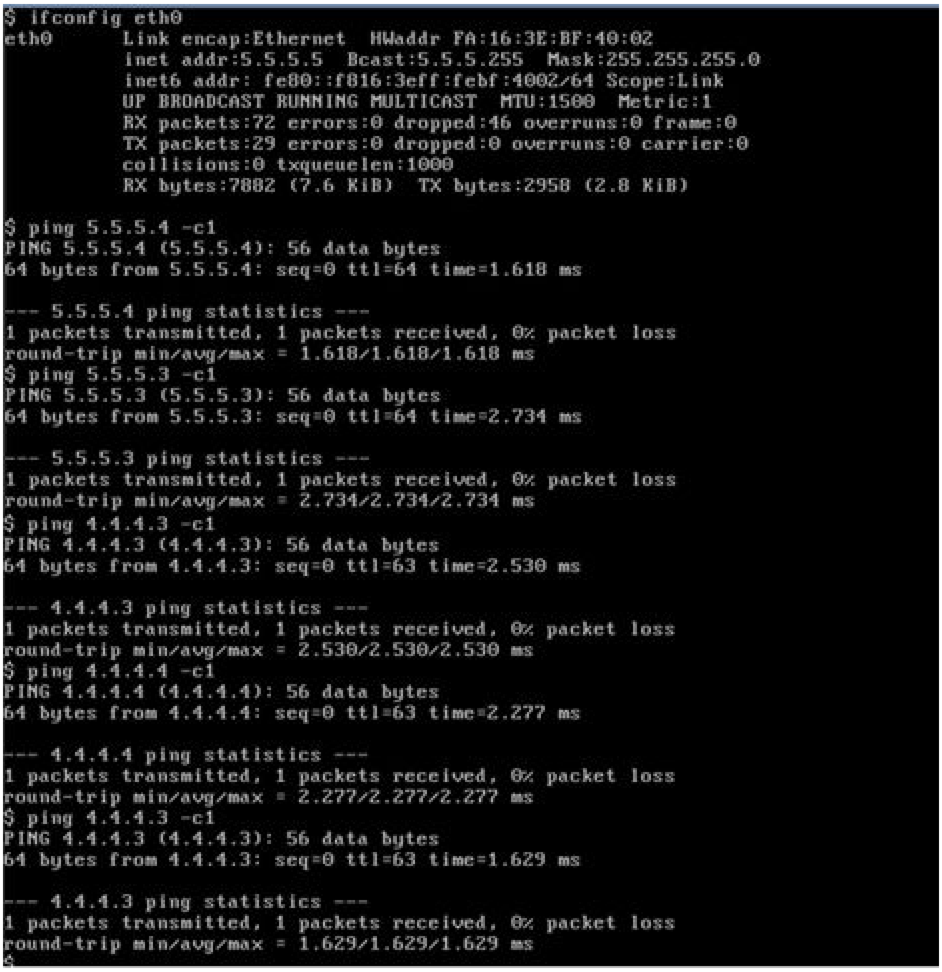
```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:41:FB:37
          inet addr:4.4.4.3  Bcast:4.4.4.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe41:fb37/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:212 errors:0 dropped:156 overruns:0 frame:0
          TX packets:30 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:25315 (24.7 KiB)  TX bytes:3056 (2.9 KiB)

$ ping 4.4.4.4 -c1
PING 4.4.4.4 (4.4.4.4): 56 data bytes
64 bytes from 4.4.4.4: seq=0 ttl=64 time=1.966 ms

--- 4.4.4.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 1.966/1.966/1.966 ms
$ ping 5.5.5.4 -c1
PING 5.5.5.4 (5.5.5.4): 56 data bytes
64 bytes from 5.5.5.4: seq=0 ttl=63 time=1.358 ms

--- 5.5.5.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 1.358/1.358/1.358 ms
$ ping 1.1.1.7 -c1
PING 1.1.1.7 (1.1.1.7): 56 data bytes
64 bytes from 1.1.1.7: seq=0 ttl=61 time=4.981 ms
--- 1.1.1.7 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 4.981/4.981/4.981 ms
$ ping 1.1.1.4 -c1
PING 1.1.1.4 (1.1.1.4): 56 data bytes
64 bytes from 1.1.1.4: seq=0 ttl=61 time=4.018 ms

--- 1.1.1.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 4.018/4.018/4.018 ms
```

The following shows an addition of new compute for Customer B:



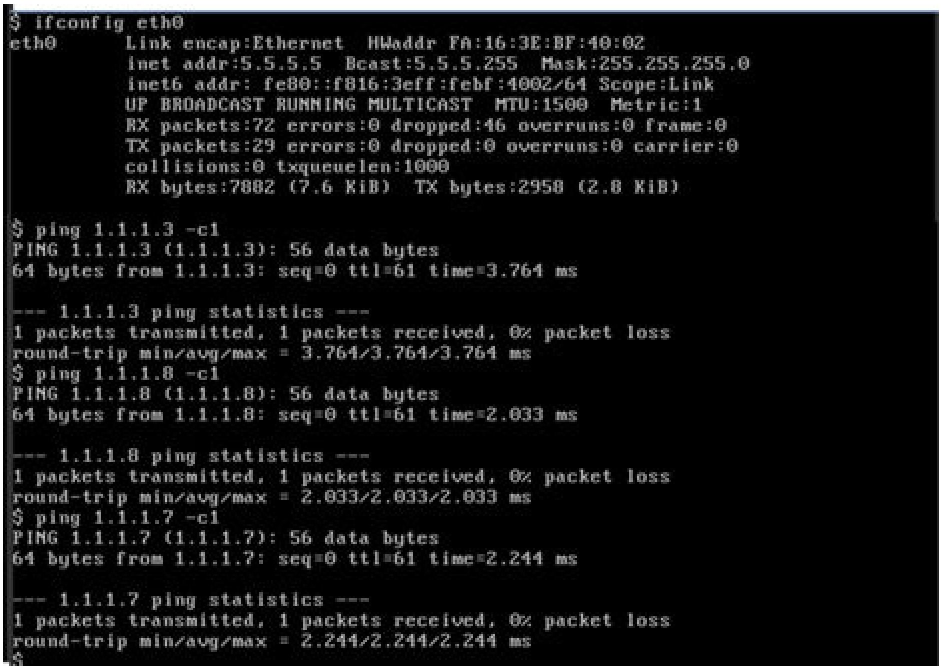
□	Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created
□	LanVM-3	cirros-0.3.2-x86_64-uec	4.4.4.5	m1.nano	-	Active	O3	None	Running	0 minutes
□	LabVM-3	cirros-0.3.4-x86_64-uec	5.5.5.5 Floating IPs: 1.1.1.9	m1.nano	-	Active	O3	None	Running	1 minute
□	LabVM-2	cirros-0.3.4-x86_64-uec	5.5.5.3 Floating IPs: 1.1.1.7	m1.nano	-	Active	O1	None	Running	35 minutes
□	LabVM-1	cirros-0.3.4-x86_64-uec	5.5.5.4	m1.nano	-	Active	O2	None	Running	35 minutes
□	LanVM-2	cirros-0.3.4-x86_64-uec	4.4.4.4	m1.nano	-	Active	O1	None	Running	35 minutes
□	LanVM-1	cirros-0.3.4-x86_64-uec	4.4.4.3 Floating IPs: 1.1.1.8	m1.nano	-	Active	O2	None	Running	35 minutes

The following shows the Lab network VM LabVM-3's reachability to other networks:



```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:BF:40:02
          inet addr:5.5.5.5  Bcast:5.5.5.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:febf:4002/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:72 errors:0 dropped:46 overruns:0 frame:0
          TX packets:29 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7882 (7.6 KiB)  TX bytes:2958 (2.8 KiB)

$ ping 5.5.5.4 -c1
PING 5.5.5.4 (5.5.5.4): 56 data bytes
64 bytes from 5.5.5.4: seq=0 ttl=64 time=1.618 ms

--- 5.5.5.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 1.618/1.618/1.618 ms
$ ping 5.5.5.3 -c1
PING 5.5.5.3 (5.5.5.3): 56 data bytes
64 bytes from 5.5.5.3: seq=0 ttl=64 time=2.734 ms

--- 5.5.5.3 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.734/2.734/2.734 ms
$ ping 4.4.4.3 -c1
PING 4.4.4.3 (4.4.4.3): 56 data bytes
64 bytes from 4.4.4.3: seq=0 ttl=63 time=2.530 ms

--- 4.4.4.3 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.530/2.530/2.530 ms
$ ping 4.4.4.4 -c1
PING 4.4.4.4 (4.4.4.4): 56 data bytes
64 bytes from 4.4.4.4: seq=0 ttl=63 time=2.277 ms

--- 4.4.4.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.277/2.277/2.277 ms
$ ping 4.4.4.3 -c1
PING 4.4.4.3 (4.4.4.3): 56 data bytes
64 bytes from 4.4.4.3: seq=0 ttl=63 time=1.629 ms

--- 4.4.4.3 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 1.629/1.629/1.629 ms
$
```

The following reachability information from LabVM-3 to the Internet including Customer A:



```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:BF:40:02
          inet addr:5.5.5.5  Bcast:5.5.5.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:febf:4002/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:72 errors:0 dropped:46 overruns:0 frame:0
          TX packets:29 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7882 (7.6 KiB)  TX bytes:2958 (2.8 KiB)

$ ping 1.1.1.3 -c1
PING 1.1.1.3 (1.1.1.3): 56 data bytes
64 bytes from 1.1.1.3: seq=0 ttl=61 time=3.764 ms

--- 1.1.1.3 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 3.764/3.764/3.764 ms
$ ping 1.1.1.8 -c1
PING 1.1.1.8 (1.1.1.8): 56 data bytes
64 bytes from 1.1.1.8: seq=0 ttl=61 time=2.033 ms

--- 1.1.1.8 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.033/2.033/2.033 ms
$ ping 1.1.1.7 -c1
PING 1.1.1.7 (1.1.1.7): 56 data bytes
64 bytes from 1.1.1.7: seq=0 ttl=61 time=2.244 ms

--- 1.1.1.7 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.244/2.244/2.244 ms
$
```

Customer C Configuration

The following shows how Customer C is using Dell ODL:

Networks:

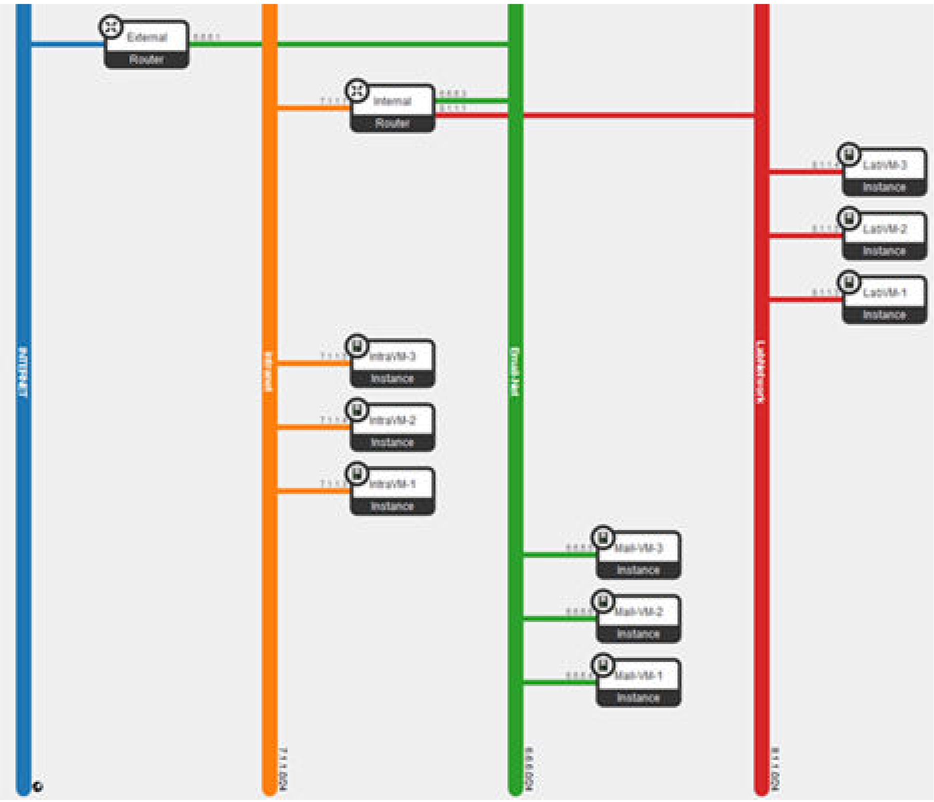
- Mail Networks: 6.6.6.0/24
- Intra Network: 7.1.1.0/24Lab Network: 8.1.1.0/24
- Internet: 1.1.1.0/24

The Customer C configuration shows:

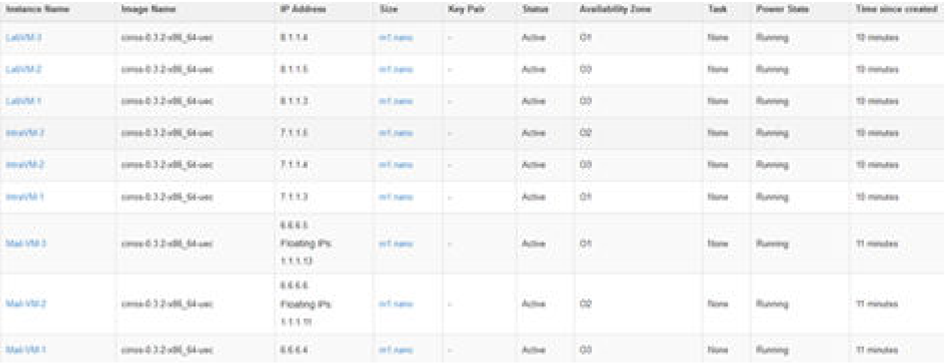
- Different network East-West connectivity between the Intra network and the Lab network.
- North-South/external network for VMs connected to Mail networks.

  North-South connectivity between tenants Customer A, Customer B, and Customer C — connected through the Internet network 1.1.1.0/24

The following shows an example topology for Customer C:

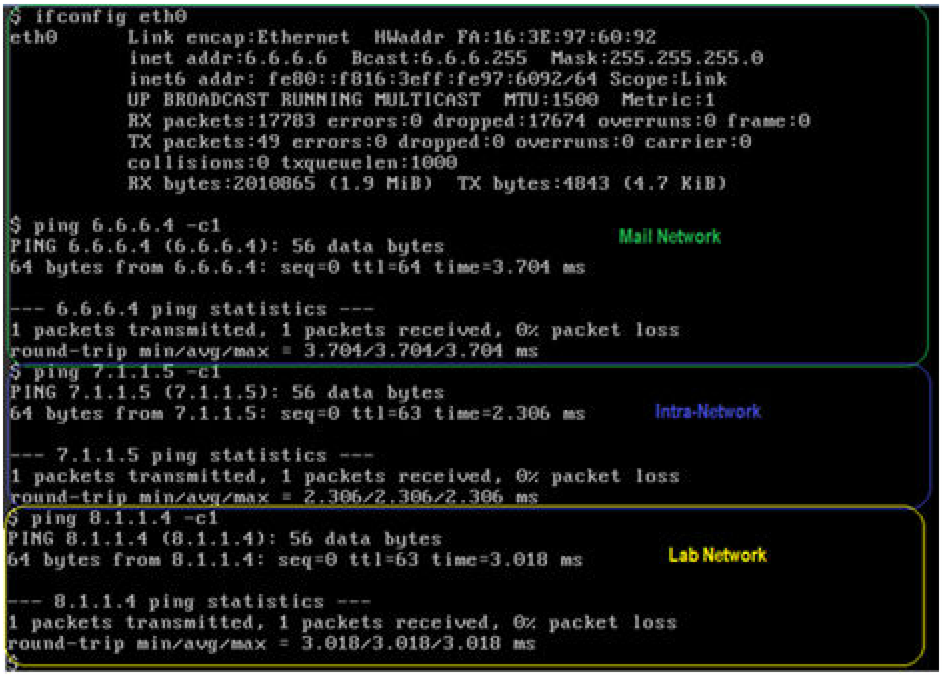


The following shows VM identification for Customer C:



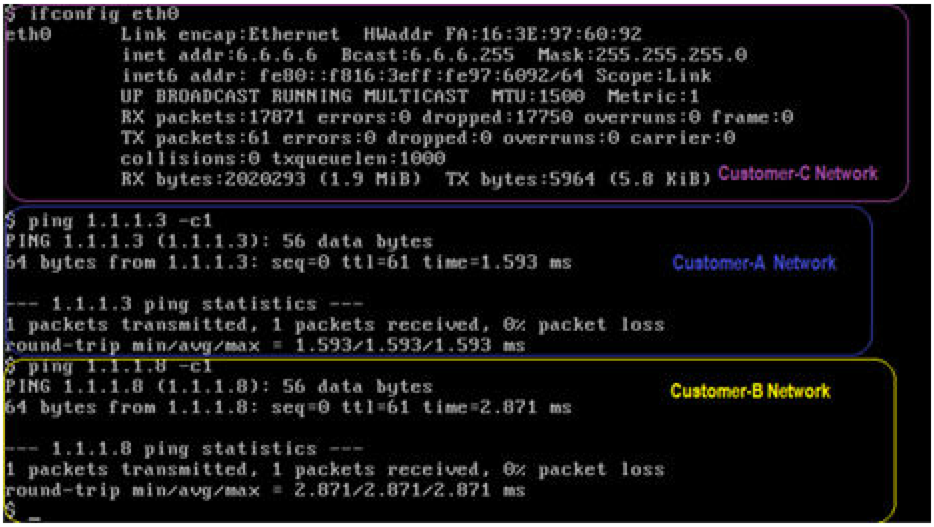
Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created
LabVM-3	cirros-0.3.2-x86_64-uec	8.1.1.4	m1.nano	-	Active	O1	None	Running	10 minutes
LabVM-2	cirros-0.3.2-x86_64-uec	8.1.1.5	m1.nano	-	Active	O3	None	Running	10 minutes
LabVM-1	cirros-0.3.2-x86_64-uec	8.1.1.3	m1.nano	-	Active	O3	None	Running	10 minutes
IntraVM-3	cirros-0.3.2-x86_64-uec	7.1.1.5	m1.nano	-	Active	O2	None	Running	10 minutes
IntraVM-2	cirros-0.3.2-x86_64-uec	7.1.1.4	m1.nano	-	Active	O3	None	Running	10 minutes
IntraVM-1	cirros-0.3.2-x86_64-uec	7.1.1.3	m1.nano	-	Active	O1	None	Running	10 minutes
Mail-VM-3	cirros-0.3.2-x86_64-uec	6.6.6.5 Floating IPs: 1.1.1.13	m1.nano	-	Active	O1	None	Running	11 minutes
Mail-VM-2	cirros-0.3.2-x86_64-uec	6.6.6.6 Floating IPs: 1.1.1.11	m1.nano	-	Active	O2	None	Running	11 minutes
Mail-VM-1	cirros-0.3.2-x86_64-uec	6.6.6.4	m1.nano	-	Active	O3	None	Running	11 minutes

The following shows reachability of Mail, Intra, and Lab networks for Customer C’s MailVM-1 VM:



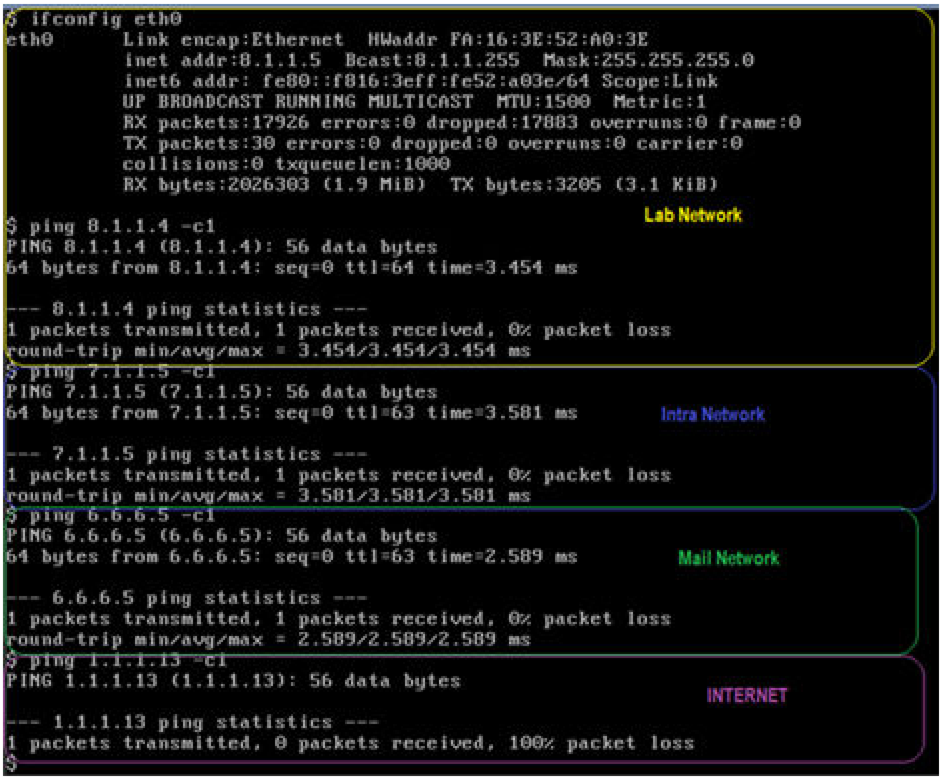
```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:97:60:92
          inet addr:6.6.6.6  Bcast:6.6.6.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe97:6092/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:17783 errors:0 dropped:17674 overruns:0 frame:0
          TX packets:49 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2010865 (1.9 MiB)  TX bytes:4843 (4.7 KiB)

$ ping 6.6.6.4 -c1
PING 6.6.6.4 (6.6.6.4): 56 data bytes
64 bytes from 6.6.6.4: seq=0 ttl=64 time=3.704 ms

--- 6.6.6.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 3.704/3.704/3.704 ms
$ ping 7.1.1.5 -c1
PING 7.1.1.5 (7.1.1.5): 56 data bytes
64 bytes from 7.1.1.5: seq=0 ttl=63 time=2.306 ms

--- 7.1.1.5 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.306/2.306/2.306 ms
$ ping 8.1.1.4 -c1
PING 8.1.1.4 (8.1.1.4): 56 data bytes
64 bytes from 8.1.1.4: seq=0 ttl=63 time=3.018 ms

--- 8.1.1.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 3.018/3.018/3.018 ms
$
```

The following shows reachability information to Customer A, and Customer B network from Customer C network:



```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:97:60:92
          inet addr:6.6.6.6  Bcast:6.6.6.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe97:6092/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:17871 errors:0 dropped:17750 overruns:0 frame:0
          TX packets:61 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2020293 (1.9 MiB)  TX bytes:5964 (5.8 KiB)
$ ping 1.1.1.3 -c1
PING 1.1.1.3 (1.1.1.3): 56 data bytes
64 bytes from 1.1.1.3: seq=0 ttl=61 time=1.593 ms

--- 1.1.1.3 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 1.593/1.593/1.593 ms
$ ping 1.1.1.8 -c1
PING 1.1.1.8 (1.1.1.8): 56 data bytes
64 bytes from 1.1.1.8: seq=0 ttl=61 time=2.871 ms

--- 1.1.1.8 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.871/2.871/2.871 ms
$ _
```

The following shows reachability information fromVM LabVM-2 of Customer C:



```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FA:16:3E:52:A0:3E
          inet addr:8.1.1.5  Bcast:8.1.1.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe52:a03e/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:17926 errors:0 dropped:17883 overruns:0 frame:0
          TX packets:30 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2026303 (1.9 MiB)  TX bytes:3205 (3.1 KiB)

$ ping 8.1.1.4 -c1
PING 8.1.1.4 (8.1.1.4): 56 data bytes
64 bytes from 8.1.1.4: seq=0 ttl=64 time=3.454 ms

--- 8.1.1.4 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 3.454/3.454/3.454 ms
$ ping 7.1.1.5 -c1
PING 7.1.1.5 (7.1.1.5): 56 data bytes
64 bytes from 7.1.1.5: seq=0 ttl=63 time=3.581 ms

--- 7.1.1.5 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 3.581/3.581/3.581 ms
$ ping 6.6.6.5 -c1
PING 6.6.6.5 (6.6.6.5): 56 data bytes
64 bytes from 6.6.6.5: seq=0 ttl=63 time=2.589 ms

--- 6.6.6.5 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.589/2.589/2.589 ms
$ ping 1.1.1.13 -c1
PING 1.1.1.13 (1.1.1.13): 56 data bytes

--- 1.1.1.13 ping statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
$
```

Overlap IP Between Tenants

The following explains how to create overlap IP between Customer A and Customer B:

- Create a Network Overlap-CustA with subnet 10.10.10.0/24 in Customer A.
- Create a Network Overlap-CustB with subnet 10.10.10.0/24 in Customer B.
- Create a new router and select **Internet Network** as an external router, and add an interface from Overlap-CustA network.
- Create a new router and select **Internet Network** as an external router, and add an interface from Overlap-CustB network.
- Spawn a VM in Overlap-CustA and Overlap-CustB and observe the same IP assigned to both of them (10.10.10.3).
- Associate floating IPs to the VMs in Overlap-CustA and Overlap-CustB.
- Ping between the VM in Overlap-CustA and VM in Overlap-CustB with floating IP — the ping should be successful.

A

Sample Configuration Files

The following lists the available configuration files:

- local.conf for the Compute Node
- local.conf for the Controller Node
- plugin.sh

NOTE: The *plugin.sh* file is located in */path/stack/networking-odl/devstack/*.

local.conf for the Compute Node

```
[[local|localrc]]
#ODL Compute local conf
LOGFILE=stack.sh.log
LOG_COLOR=False
SCREEN_LOGDIR=/opt/stack/data/log
#OFFLINE=True
RECLONE=yes

disable_all_services
enable_service nova n-cpu quantum n-novnc n-cauth rabbit

#OpenStack-Dell-ODL integration Start
HOST_IP=10.16.148.33
HOST_NAME=os-compute1
SERVICE_HOST=10.16.148.31
SERVICE_HOST_NAME=os-controller
#OpenStack-Dell-ODL integration End

VNCSERVER_PROXYCLIENT_ADDRESS=$HOST_IP
VNCSERVER_LISTEN=0.0.0.0

#OpenStack-Dell-ODL integration Start
Q_PLUGIN=ml2
ENABLE_TENANT_TUNNELS=True
Q_ML2_TENANT_NETWORK_TYPE=vxlan
#OpenStack-Dell-ODL integration End

#NOTE: Set the database type
DATABASE_TYPE=mysql
KEYSTONE_CATALOG_BACKEND=sql

Q_HOST=$SERVICE_HOST
MYSQL_HOST=$SERVICE_HOST
RABBIT_HOST=$SERVICE_HOST
GLANCE_HOSTPORT=$SERVICE_HOST:9292
KEYSTONE_AUTH_HOST=$SERVICE_HOST
KEYSTONE_SERVICE_HOST=$SERVICE_HOST

MYSQL_PASSWORD=mysql
RABBIT_PASSWORD=rabbit
SERVICE_TOKEN=service
```

```
SERVICE_PASSWORD=admin
ADMIN_PASSWORD=admin

#working networking odl
#enable_plugin networking-odl https://github.com/stackforge/networking-odl
#OS-Dell-ODL integration change
enable_plugin networking-odl https://github.com/stackforge/networking-odl
stable/kilo
ODL_MODE=compute
#Open#OS-Dell-ODL integration change
Daylight IP address
ODL_MGR_IP=10.16.148.232
#DataNetworks (i.e eth2 ip address)
ODL_LOCAL_IP=20.1.1.3
#L3 Enable and External Networks
PUBLIC_INTERFACE=em3
disable_ODL_PROVIDER_MAPPINGS=br-ex:em3
service q-l3
Q_L3_ENABLED=True
ODL_L3=True

[[post-config|$NOVA_CONF]]
[oslo_messaging_rabbit]
heartbeat_timeout_threshold = 0
```

local.conf for the Controller Node

```
[local|localrc]]
LOGFILE=stack.sh.log
SCREEN_LOGDIR=/opt/stack/data/log
LOG_COLOR=True

#flip OFFLINE and RECLONE to lock (RECLONE=no) or update the source.
#OFFLINE=False
RECLONE=yes
VERBOSE=True
#PIP_UPGRADE=True

disable_service swift
disable_service cinder
disable_service n-net
enable_service q-svc
enable_service q-dhcp
enable_service q-meta
enable_service horizon
enable_service neutron
enable_service tempest
enable_service odl-neutron odl-compute
enable_service mysql rabbit

#CONFIGURATION CHANGE HERE
#OpenStack-Dell-ODL integration Start
HOST_IP=10.16.148.31
HOST_NAME=os-controller
SERVICE_HOST=$HOST_IP
SERVICE_HOST_NAME=$HOST_NAME
NEUTRON_CREATE_INITIAL_NETWORKS=False
Q_PLUGIN=ml2
Q_ML2_TENANT_NETWORK_TYPE=vxlan
ENABLE_TENANT_TUNNELS=True
#OpenStack-Dell-ODL integration End
```

```
VNCSERVER_PROXYCLIENT_ADDRESS=${HOST_IP}
VNCSERVER_LISTEN=0.0.0.0

MULTI_HOST=True

MYSQL_HOST=$SERVICE_HOST
RABBIT_HOST=$SERVICE_HOST
GLANCE_HOSTPORT=$SERVICE_HOST:9292
KEYSTONE_AUTH_HOST=$SERVICE_HOST
KEYSTONE_SERVICE_HOST=$SERVICE_HOST

MYSQL_PASSWORD=mysql
RABBIT_PASSWORD=rabbit
SERVICE_TOKEN=service
SERVICE_PASSWORD=admin
ADMIN_PASSWORD=admin

#OpenDaylight Integration Configurations
#OpenStack-Dell-ODL integration Start
enable_plugin networking-odl https://github.com/stackforge/networking-odl
stable/kilo
ODL_NETVIRT_DEBUG_LOGS=True
ODL_MGR_IP=10.16.148.232
ODL_PORT=8080
ODL_BOOT_WAIT=123
#DATA NETWORK (i.e em2 IP address)
ODL_LOCAL_IP=20.1.1.2
#L3 Network or Connect to External Network
PUBLIC_INTERFACE=em3
ODL_PROVIDER_MAPPINGS=br-ex:em3
# If using ODL outside devstack-control, replace ODL_MODE
ODL_MODE=externalodl
disable_service q-l3
Q_L3_ENABLED=True
ODL_L3=True
[[post-config|$NEUTRON_CONF]]
[DEFAULT]
service_plugins = networking_odl.l3.l3_odl.OpenDaylightL3RouterPlugin
#OpenStack-Dell-ODL integration End

[[post-config|/etc/neutron/plugins/ml2/ml2_conf.ini]]
[agent]
minimize_polling=True

[[post-config|$NOVA_CONF]]
[oslo_messaging_rabbit]
heartbeat_timeout_threshold = 0
[[post-config|$CINDER_CONF]]
[oslo_messaging_rabbit]
heartbeat_timeout_threshold = 0
[[post-config|$NEUTRON_CONF]]
[oslo_messaging_rabbit]
heartbeat_timeout_threshold = 0
[[post-config|$GLANCE_API_CONF]]
[oslo_messaging_rabbit]
heartbeat_timeout_threshold = 0
```

plugin.sh

```
#!/bin/bash
#
# devstack/plugin.sh
# Functions to control the configuration and operation of the opendaylight
service

# Dependencies:
#
# ``functions`` file
# ``DEST`` must be defined
# ``STACK_USER`` must be defined
# ``DATA_DIR`` must be defined

# ``stack.sh`` calls the entry points in this order:
#
# - is_opendaylight_enabled
# - is_opendaylight-compute_enabled
# - install_opendaylight
# - install_opendaylight-compute
# - configure_opendaylight
# - init_opendaylight
# - start_opendaylight
# - stop_opendaylight-compute
# - stop_opendaylight
# - cleanup_opendaylight

# Save trace setting
XTRACE=$(set +o | grep xtrace)
set +o xtrace

# OpenDaylight directories
NETWORKING_ODL_DIR=$DEST/networking-odl
ODL_DIR=$DEST/opendaylight

# Make sure $ODL_DIR exists
mkdir -p $ODL_DIR

# Import common functions
source $TOP_DIR/functions

# For OVS_BRIDGE and PUBLIC_BRIDGE
source $TOP_DIR/lib/neutron_plugins/ovs_base

# Source global ODL settings
source $NETWORKING_ODL_DIR/devstack/settings.odl

# Test with a finite retry loop.
# NOTE: ONLY NEEDED in stable/kilo, already in
# devstack master (commit: 442e4e96)
#
function odl_test_with_retry {
    local testcmd=$1
    local failmsg=$2
    local until=${3:-10}
    local sleep=${4:-0.5}

    if ! timeout $until sh -c "while ! $testcmd; do sleep $sleep; done"; then
        die $LINENO "$failmsg"
    fi
}

# Source specific ODL release settings
function odl_update_maven_metadata_xml {
    local MAVENMETAFILE=$1
    local NEXUSPATH=$2
    local BUNDLEVERSION=$3
    if [[ "$OFFLINE" == "True" ]]; then
        return
    fi

    # Remove stale MAVENMETAFILE for cases where you switch releases
    rm -f $MAVENMETAFILE

    # Acquire the timestamp information from maven-metadata.xml
    wget -O $MAVENMETAFILE ${NEXUSPATH}/${BUNDLEVERSION}/maven-metadata.xml
}

source $NETWORKING_ODL_DIR/devstack/odl-releases/$ODL_RELEASE

# Entry Points
# ------------

# Test if OpenDaylight is enabled
# is_# Test if OpenDaylight is enabled
opendaylight_enabled
function is_opendaylight_enabled {
    [[ ,${ENABLED_SERVICES} =~ ,"odl-" ]] && return 0
    return 1
}

# cleanup_opendaylight() - Remove residual data files, anything left over from
previous
# runs that a clean run would need to clean up
function cleanup_opendaylight {
    :
}

# configure_opendaylight() - Set config files, create data dirs, etc
function configure_opendaylight {
    echo "Configuring OpenDaylight"

    sudo ovs-vsctl --no-wait -- --may-exist add-br $OVS_BR
    sudo ovs-vsctl --no-wait br-set-external-id $OVS_BR bridge-id $OVS_BR

    # The logging config file in ODL
    local ODL_LOGGING_CONFIG=${ODL_DIR}/${ODL_NAME}/etc/
org.ops4j.pax.logging.cfg

    # Add netvirt feature in Karaf, if it's not already there
    local ODLFEATUREMATCH=$(cat $ODL_DIR/$ODL_NAME/etc/
org.apache.karaf.features.cfg | grep featuresBoot= | grep
$ODL_NETVIRT_KARAF_FEATURE)
    if [ "$ODLFEATUREMATCH" == "" ]; then
        sed -i "/^featuresBoot=/ s/$/,$ODL_NETVIRT_KARAF_FEATURE/" $ODL_DIR/
$ODL_NAME/etc/org.apache.karaf.features.cfg
    fi

    if [[ "$ODL_RELEASE" =~ "helium" ]]; then
        # Move Tomcat to $ODL_PORT
        local _ODLPORT=$(cat $ODL_DIR/$ODL_NAME/configuration/tomcat-server.xml
| grep $ODL_PORT)
        if [ "$_ODLPORT" == "" ]; then
            sed -i "/\<Connector port/ s/808./$ODL_PORT/" $ODL_DIR/$ODL_NAME/
configuration/tomcat-server.xml
        fi
    else
        # Move Jetty to $ODL_PORT
        local _ODLPORT=$(cat $ODL_DIR/$ODL_NAME/etc/jetty.xml | grep $ODL_PORT)
        if [ "$_ODLPORT" == "" ]; then
            sed -i "/\<Property name\=\"jetty\.port/ s/808./$ODL_PORT/"
```

```
$ODL_DIR/$ODL_NAME/etc/jetty.xml
        fi
    fi

    # Configure L3 if the user wants it
    if [ "${ODL_L3}" == "True" ]; then
        # Configure L3 FWD if it's not there
        local L3FWD=$(cat $ODL_DIR/$ODL_NAME/etc/custom.properties | grep
^ovsdb.l3.fwd.enabled)
        if [ "$L3FWD" == "" ]; then
            echo "ovsdb.l3.fwd.enabled=yes" >> $ODL_DIR/$ODL_NAME/etc/
custom.properties
        fi
    fi

    # Remove existing logfiles
    rm -f "/opt/stack/logs/$ODL_KARAF_LOG_BASE*"
    # Log karaf output to a file
    _LF=/opt/stack/logs/$ODL_KARAF_LOG_NAME
    LF=$(echo $_LF | sed 's/\//\\\//g')
    # Soft link for easy consumption
    ln -sf $_LF "/opt/stack/logs/screen-karaf.txt"

    # Change the karaf logfile
    sed -i "/^log4j\.appender\.out\.file/ s/.*/log4j\.appender\.out\.file\=
$LF/" \
    $ODL_DIR/$ODL_NAME/etc/org.ops4j.pax.logging.cfg

    # Configure DEBUG logs for network virtualization in odl, if the user wants
it
    if [ "${ODL_NETVIRT_DEBUG_LOGS}" == "True" ]; then
        local OVSDB_DEBUG_LOGS=$(cat $ODL_LOGGING_CONFIG | grep
^log4j.logger.org.opendaylight.ovsdb)
        if [ "${OVSDB_DEBUG_LOGS}" == "" ]; then
            echo 'log4j.logger.org.opendaylight.ovsdb = TRACE, out' >>
$ODL_LOGGING_CONFIG
            echo 'log4j.logger.org.opendaylight.ovsdb.lib = INFO, out' >>
$ODL_LOGGING_CONFIG
            echo
'log4j.logger.org.opendaylight.ovsdb.openstack.netvirt.impl.NeutronL3Adapter =
DEBUG, out' >> $ODL_LOGGING_CONFIG
            echo
'log4j.logger.org.opendaylight.ovsdb.openstack.netvirt.impl.TenantNetworkManager
Impl = DEBUG, out' >> $ODL_LOGGING_CONFIG
            echo
'log4j.logger.org.opendaylight.ovsdb.plugin.md.OvsdbInventoryManager = INFO,
out' >> $ODL_LOGGING_CONFIG
      fi
        if [[ "$ODL_RELEASE" =~ "helium" ]]; then
            local ODL_NEUTRON_DEBUG_LOGS=$(cat $ODL_LOGGING_CONFIG | grep
^log4j.logger.org.opendaylight.controller.networkconfig.neutron)
            if [ "${ODL_NEUTRON_DEBUG_LOGS}" == "" ]; then
                echo
'log4j.logger.org.opendaylight.controller.networkconfig.neutron = TRACE, out'
>> $ODL_LOGGING_CONFIG
            fi
        else
            local ODL_NEUTRON_DEBUG_LOGS=$(cat $ODL_LOGGING_CONFIG | grep
^log4j.logger.org.opendaylight.neutron)
            if [ "${ODL_NEUTRON_DEBUG_LOGS}" == "" ]; then
                echo 'log4j.logger.org.opendaylight.neutron = TRACE, out' >>
$ODL_LOGGING_CONFIG
            fi
        fi
        # Bump up how man logfiles we save after rotation if debug is turned on
        sed -i "/^log4j.appender.out.maxBackupIndex=/ s/
10/$ODL_LOGFILES_TO_SAVE/" $ODL_LOGGING_CONFIG
    fi
}

function configure_ml2_odl {
    echo "Configuring ML2 for OpenDaylight"
    populate_ml2_config /$Q_PLUGIN_CONF_FILE ml2_odl url=$ODL_ENDPOINT
    populate_ml2_config /$Q_PLUGIN_CONF_FILE ml2_odl username=$ODL_USERNAME
    populate_ml2_config /$Q_PLUGIN_CONF_FILE ml2_odl password=$ODL_PASSWORD
}

# init_opendaylight() - Initialize databases, etc.
function init_opendaylight {
    # clean up from previous (possibly aborted) runs
    # create required data files
    :
}

# install_opendaylight() - Collect source and prepare
function install_opendaylight {
    echo "Installing OpenDaylight and dependent packages"

    if is_ubuntu; then
        install_package maven openjdk-7-jre openjdk-7-jdk
    else
        yum_install maven java-1.7.0-openjdk
    fi

    install_opendaylight_neutron_thin_ml2_driver

    # Download OpenDaylight
    cd $ODL_DIR


    if [[ "$OFFLINE" != "True" ]]; then
          wget -N $ODL_URL/$ODL_PKG
    fi
    unzip -u -o $ODL_PKG
}

function install_opendaylight_neutron_thin_ml2_driver {
    cd $NETWORKING_ODL_DIR
    echo "Installing the Networking-ODL driver for OpenDaylight"
    sudo python setup.py install
}

# install_opendaylight-compute - Make sure OVS is installed
function install_opendaylight-compute {
    # packages are the same as for Neutron OVS agent
    _neutron_ovs_base_install_agent_packages
}

# start_opendaylight() - Start running processes, including screen
function start_opendaylight {
    echo "Starting OpenDaylight"
    if is_ubuntu; then
        JHOME=/usr/lib/jvm/java-1.7.0-openjdk-amd64
    else
        JHOME=/usr/lib/jvm/java-1.7.0-openjdk
    fi

    # Wipe out the data directory ... grumble grumble grumble
    rm -rf $ODL_DIR/$ODL_NAME/data

    # The following variables are needed by the running karaf process.
    # See the "bin/setenv" file in the OpenDaylight distribution for
    # their individual meaning.
    export JAVA_HOME=$JHOME
    export JAVA_MIN_MEM=$ODL_JAVA_MIN_MEM
    export JAVA_MAX_MEM=$ODL_JAVA_MAX_MEM
    export JAVA_MAX_PERM_MEM=$ODL_JAVA_MAX_PERM_MEM
    run_process odl-server "$ODL_DIR/$ODL_NAME/bin/start"

    if [ -n "$ODL_BOOT_WAIT_URL" ]; then
        echo "Waiting for Opendaylight to start via $ODL_BOOT_WAIT_URL ..."
        # Probe ODL restconf for netvirt until it is operational
        local sleep_interval=3
        local testcmd="curl -o /dev/null --fail --silent --head -u $
{ODL_USERNAME}:${ODL_PASSWORD} http://${ODL_MGR_IP}:${ODL_PORT}/$
{ODL_BOOT_WAIT_URL}"
        odl_test_with_retry "$testcmd" "Opendaylight did not start after
$ODL_BOOT_WAIT" $ODL_BOOT_WAIT $sleep_interval
    else
        echo "Waiting for Opendaylight to start ..."
        # Sleep a bit to let OpenDaylight finish starting up
        sleep $ODL_BOOT_WAIT
    fi
}

# stop_opendaylight() - Stop running processes (non-screen)
function stop_opendaylight {
    # Stop the karaf container
    $ODL_DIR/$ODL_NAME/bin/stop
    stop_process odl-server
}

# stop_opendaylight-compute() - Remove OVS bridges
function stop_opendaylight-compute {
#OpenStack-Dell-ODL integration Start
    for port in $(sudo ovs-vsctl show | grep Port | awk '{print $2}'  | cut -d
'"' -f 2 | grep patch); do
        sudo ovs-vsctl del-port ${port}
    done
    #OpenStack-Dell-ODL integration End

    # remove all OVS ports that look like Neutron created ports
    for port in $(sudo ovs-vsctl list port | grep -o -e tap[0-9a-f\-]* -e q[rg]-
[0-9a-f\-]*); do
        sudo ovs-vsctl del-port ${port}
    done

#OpenStack-Dell-ODL integration Start
    for port in $(sudo ovs-vsctl list port | grep name | grep vxlan | awk
'{print $3}'  | cut -d '"' -f 2); do
             sudo ovs-vsctl del-port ${port}
    done
#OpenStack-Dell-ODL integration End
}

# main loop
if is_service_enabled odl-server; then
    if [[ "$1" == "source" ]]; then
        # no-op
        :
    elif [[ "$1" == "stack" && "$2" == "install" ]]; then
        setup_opendaylight_package
        install_opendaylight
        configure_opendaylight
        init_opendaylight
    elif [[ "$1" == "stack" && "$2" == "post-config" ]]; then
        configure_ml2_odl
        # This has to start before Neutron
        start_opendaylight
    elif [[ "$1" == "stack" && "$2" == "post-extra" ]]; then
        # no-op
        :
    fi

    if [[ "$1" == "unstack" ]]; then
        stop_opendaylight
        cleanup_opendaylight
    fi

    if [[ "$1" == "clean" ]]; then
        # no-op
        :
    fi
fi

if is_service_enabled odl-neutron; then
    if [[ "$1" == "source" ]]; then
        # no-op
        :
    elif [[ "$1" == "stack" && "$2" == "install" ]]; then
        install_opendaylight_neutron_thin_ml2_driver
    elif [[ "$1" == "stack" && "$2" == "post-config" ]]; then
        configure_ml2_odl
    elif [[ "$1" == "stack" && "$2" == "post-extra" ]]; then
        # no-op
        :
    fi

    if [[ "$1" == "unstack" ]]; then
        # no-op
        :
    fi

    if [[ "$1" == "clean" ]]; then
        # no-op
        :
    fi
fi

if is_service_enabled odl-compute; then
    if [[ "$1" == "source" ]]; then
        # no-op
        :
    elif [[ "$1" == "stack" && "$2" == "install" ]]; then
        install_opendaylight-compute
    elif [[ "$1" == "stack" && "$2" == "post-config" ]]; then
        if is_service_enabled nova; then
            create_nova_conf_neutron
        fi
    elif [[ "$1" == "stack" && "$2" == "extra" ]]; then
        echo_summary "Initializing OpenDaylight"
        ODL_LOCAL_IP=${ODL_LOCAL_IP:-$HOST_IP}
        ODL_MGR_PORT=${ODL_MGR_PORT:-6640}
        read ovstbl <<< $(sudo ovs-vsctl get Open_vSwitch . _uuid)
        sudo ovs-vsctl set-manager tcp:$ODL_MGR_IP:$ODL_MGR_PORT
        if [[ -n "$ODL_PROVIDER_MAPPINGS" ]] && [[ "$ENABLE_TENANT_VLANS" ==
"True" ]]; then
            sudo ovs-vsctl set Open_vSwitch $ovstbl \
                other_config:provider_mappings=$ODL_PROVIDER_MAPPINGS
        fi
        sudo ovs-vsctl set Open_vSwitch $ovstbl other_config:local_ip=
$ODL_LOCAL_IP

        # Configure public bridge to be used by ODL_L3
        if [ "${ODL_L3}" == "Do_not_enable" ]; then
#OpenStack-Dell-ODL integration Start
        #    sudo ovs-vsctl --no-wait -- --may-exist add-br $PUBLIC_BRIDGE
        #    sudo ovs-vsctl --no-wait br-set-external-id $PUBLIC_BRIDGE bridge-
id $PUBLIC_BRIDGE
#OpenStack-Dell-ODL integration End
            # Add public interface to public bridge, if provided
            if [ -n "$PUBLIC_INTERFACE" ]; then
#OpenStack-Dell-ODL integration Start
            # sudo ovs-vsctl add-port $PUBLIC_BRIDGE $PUBLIC_INTERFACE
#OpenStack-Dell-ODL integration  End
                sudo ip link set $PUBLIC_INTERFACE up
            fi
        fi
    elif [[ "$1" == "stack" && "$2" == "post-extra" ]]; then
        # no-op
        :
    fi

    if [[ "$1" == "unstack" ]]; then
         stop_opendaylight-compute
        sudo ovs-vsctl del-manager
        BRIDGES=$(sudo ovs-vsctl list-br)
        for bridge in $BRIDGES ; do
            sudo ovs-vsctl del-controller $bridge
#OpenStack-Dell-ODL integration Start
     sudo ovs-vsctl del-br $bridge
#OpenStack-Dell-ODL integration End
        done
    fi

    if [[ "$1" == "clean" ]]; then
        # no-op
        :
    fi
fi
# Restore xtrace
XTRACE

# Tell emacs to use shell-script-mode
## Local variables:
## mode: shell-script
## End:
```

B

Troubleshooting

The Dell ODL Controller 1.0.0.0 release is built based on OpenDayLight LIthium SR1 release. The controller facilitiates migration to software-defined networking (SDN) for network virtualization overlay (NVO) uses cases, using the VxLAN tunneling mechanism.

The following lists possible problems and resolutions.

Problem	Resolution
If a floating IP assigned to a VM is reassigned to another VM on a different compute node, North-South traffic using that floating IP does not work.	Before reassigning the floating IP to a VM on a different compute node, clear the corrresponding `arp` entry on the external gateway, or wait until the `arp` entry times out on the external gateway.
When a compute node with external network reachability is unstacked, ODL throws an error log `MAC address for gateway <gateway-ip> cannot be resolved` every 10 seconds.	Restack the same compute node, then reassign the floating IP to one VM on this compute node.
If a compute node is unstacked and restacked, East-West traffic between different subnets fails on that particular compute node.	Instantiate a VM instance on each subnet that you want to communicate across. East-West traffic should start working.

C

Useful Links

This topic contains useful links to help you install the Dell ODL.

Topic	Link
Ubuntu 1404 LTS	http://www.ubuntu.com/download/server
OpenStack Installation Guide for Ubuntu	http://docs.openstack.org/kilo/install-guide/install/apt/content/
OpenStack hardware — minimal requirements	http://docs.openstack.org/kilo/install-guide/install/apt/content/ch_overview.html
Neutron Configuration for OpenStack Reference	http://docs.openstack.org/kilo/config-reference/content/section_neutron.conf.html
Dell S4810 Configuration Guide	ftp://ftp.dell.com/Manuals/all-products/esuprt_ser_stor_net/esuprt_networking/esuprt_net_fxd_prt_swtchs/force10-s4810_Owner%27s%20Manual9_en-us.pdf
Neutron Service in Controller Node	http://docs.openstack.org/kilo/install-guide/install/apt/content/neutron-controller-node.html
Neutron Service in Compute Node	http://docs.openstack.org/kilo/install-guide/install/apt/content/neutron-compute-node.html