IMPLEMENTATION GUIDE

Security Solution Implementation Installation Guide

Intel Corporation
Datacenter Network
Solutions Group

Authors

Eduardo Castro
Solution Software Engineer,
Intel Corporation

Tarek Radi
Lead Technical Program Manager,
Intel Corporation

1.0 Introduction

Cerner Corporation, Intel Corporation, and Midokura have teamed up to deliver a complete Red Hat Enterprise OpenStack Platform*-based virtual security solution that provides visibility and control against malicious activities within the software-defined data center of Cerner Corporation, securing east-west traffic.

Cerner Corporation is providing its services with private clouds. One of the results of the strict security policy in Cerner is the use of physical devices with intrusion prevention systems (IPS) and intrusion detection systems (IDS), located at the edge of the data centers. Currently, any traffic at Cerner's private cloud must be redirected to remote, physical IPS/IDS devices for packet inspection.

Even if two interconnected virtual machines (VMs) are located on the same physical host, the traffic from the source VM is redirected via the Midokura Enterprise MidoNet* virtual switch to the router and the gateway at the edge of the cloud, to reach IPS/IDS devices. Then it is sent back through the gateway to the cloud, and through the router and vSwitch to the target VM. Such behavior is known as a trombone effect and has a significant, negative impact on the overall latency.

The proposed solution eliminates the trombone effect and minimizes latency because traffic is inspected locally within the cloud through the use of virtual security functions. In this solution, we do not replace the physical IPS/IDS appliances. These are still used to inspect the north-south and east-west traffic coming from outside the cloud. To support inspection of the east-west traffic within the same cloud, we installed virtual IPS (vIPS) functions at each compute node of the cloud.

This Solution Implementation installation document presents the installation and configuration of the software components of this virtualized security solution. The setup is based on the Red Hat Enterprise Linux* 7.1 operating system and the Red Hat OpenStack Platform 7 for the orchestration of the cloud infrastructure, enabling better scalability and automated deployment of virtual functions. The level of security and performance of the cloud solution is at least as good as that of physical IPS appliances.

The primary audiences for this document are architects and engineers who are interested in implementing security protection mechanisms in an SDN/NFV environment; however, the presented solution is by no means a large-scale, general-purpose solution that can be applied to any NFV use case. Readers should use this document as a demonstration of how effective protection against internal-to-internal attacks is possible in SDN/NFV. They can follow the steps documented herein to build their own proof-of-concept topology and scale it.

It is important to note that the details contained herein are just an example of one way of applying security functions for a customer. Intel does not aim to promote or recommend any specific hardware, software, or supplier mentioned in this document. In addition, Intel does not aim to tie customers to any specific software and hardware stack.

For an overview of this security solution, including the hardware and software components used, please refer to the Security Solution Implementation Summary.

Table of Contents

2.0 Solution Configuration

This section provides the data definitions specific to the setup and configuration that appear as examples in this installation guide. This section lists all relevant configuration items, such as network Internet Protocol (IP addresses), IP ranges, ports, and interfaces, in the form of a data dictionary and is intended to help you plan in advance the configuration of your setup.

Disclaimer: Please note that not all the data specific to the setup were listed in this section. Some type of data, including various kinds of IDs, labels, and so on might have been generated automatically or are not editable; however, such data will be present in the installation steps of this guide. For these kinds of commands and outputs, you will experience different values, specific to the setup.

For the setup presented in this installation guide, each node is connected to three different networks:

- **External network** (herein also called a Public network) is used as an external gateway. This network is needed for the uplink in Midokura Enterprise MidoNet.
- **Management Network** is the network where OpenStack is installed; however, in this document, for the sake of simplification and easier connectivity to the OpenStack Dashboard*, we use the same external network as the management network. This means that in this setup two network interfaces are connected to the same external network per server.
- **Data network** where the virtual extensible local area network (VxLAN) tunnels are allocated.

Table 1. Basic host and connectivity information.

	CONTROLLER/COMPUTE1	COMPUTE2	COMPUTE3	ANALYTIC SERVER
Host	host0	host1	host2	
Host name	MultinodeController.ch.intel.com	CernerMidonet	MultinodeCompute1.ch.intel.com	AnalyticServer
Gateway	10.250.101.1			
DNS1	10.248.2.1			
DNS2	10.2.71.6			
DNS3	10.19.1.4			

Table 2. Physical network interface addresses.

	EXTERNAL NETWORK	MANAGEMENT NETWORK	DATA NETWORK
Subnet ID	10.250.101.0	10.250.101.0	172.16.10.0
Subnet mask	255.255.255.0	255.255.255.0	255.255.255.0

Table 3. Physical network interface devices.

NETWORK	CONTROLLER/COMPUTE1	COMPUTE2	COMPUTE3	MIDONET ANALYTIC SERVER
External	enp4s0f3	enp4s0f0	ens20f0	ens20f0
Management	enp4s0f0	enp4s0f3	ens20f1	ens20f1
Data	ens787f0	ens787f0	enp133s0f0	enp133s0f0

Table 4. Physical network interface addresses.

NETWORK	CONTROLLER/COMPUTE1	COMPUTE2	COMPUTE3	MIDONET ANALYTIC SERVER
External	10.250.101.11	10.250.101.13	10.250.101.15	10.250.101.16
Management	10.250.101.10	10.250.101.12	10.250.101.14	10.250.101.17
Data	172.16.10.10	172.16.10.12	172.16.10.14	172.16.10.16

Table 5. Floating IP address pool ranges for External Network.

Start with IP address	10.250.101.133
End with IP address	10.250.101.149

Figure 1 explains the physical network connections and OpenStack deployment.

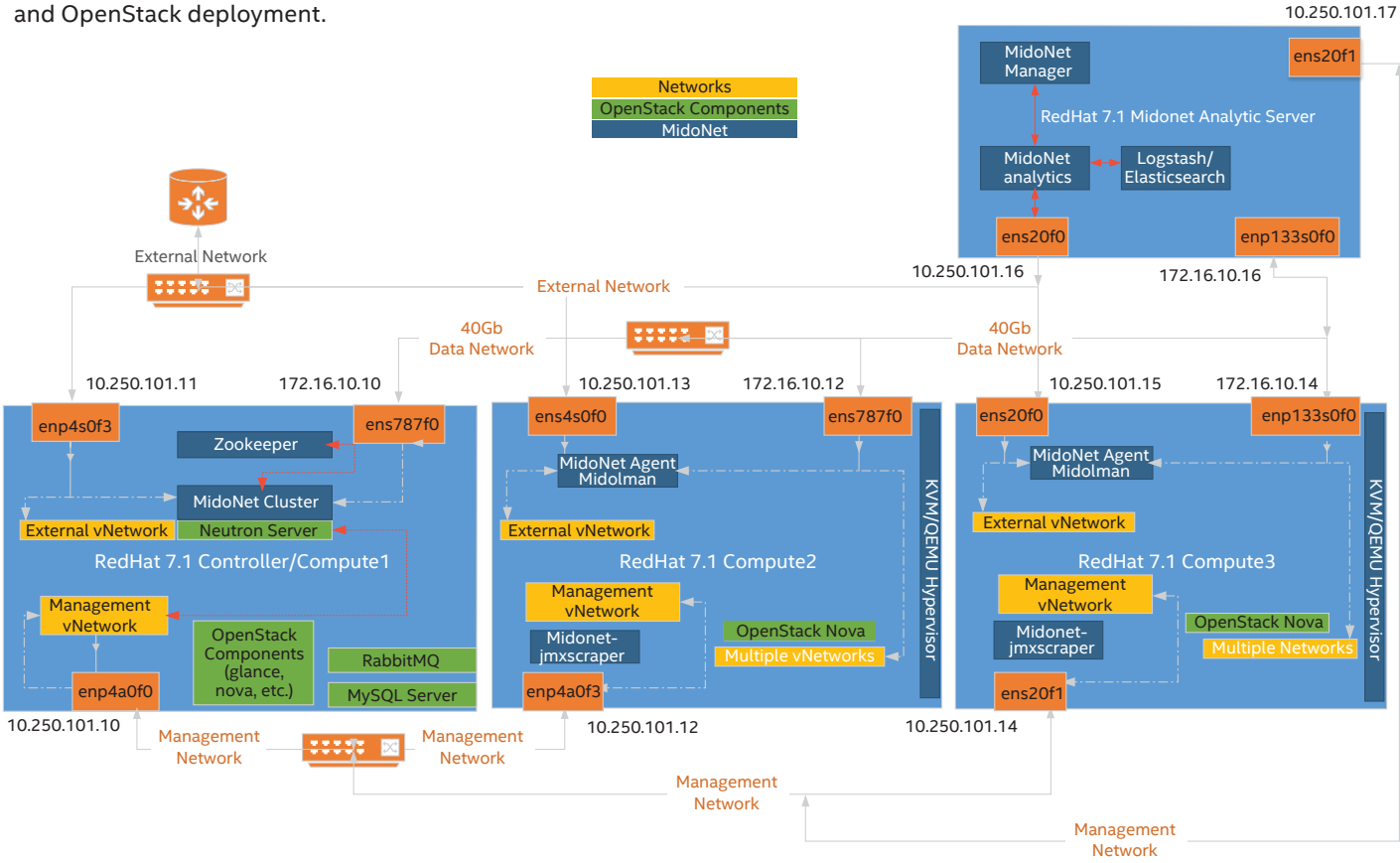


Figure 1. Network connectivity for the target setup.

3.0 Installation Guide

This chapter contains instructions for installation and configuration of the software stack.

3.1 Prerequisites to Installation of Red Hat Enterprise Linux 7.1

Note that the instructions for installing Red Hat Enterprise Linux 7.1 are not within the scope of this document; however, this section contains some information that a user needs to follow during OS installation or configuration.

To help ensure a successful installation process, make sure that the following steps occur:

- Create a RAID 0 virtual disk from all the physical disks.
- Create custom partitioning as presented in Table 6:

Table 6. Partitioning schema.

PARTITION	SIZE
Biosboot	2 MB
/boot	5 GB
/swap	Double the size of physical memory
/ (root partition)	Remaining space

Perform the following steps to install the operating system and configure the network interfaces on all the machines in the setup.

1. Install Red Hat Enterprise Linux 7.1 and set the following:
   - Hostname: /etc/hostname
   - Hosts: /etc/host
2. Disable and stop the firewall.

```
# systemctl disable firewalld
# systemctl stop firewalld
```

3. Configure all of the network interfaces using files in the `/etc/sysconfig/network-scripts/ directory`. In this setup, all of the nodes have three network interfaces. The following example is presented for the `ens20f0` interface of the controller. In this case, the configuration file name is `ifcfg-ens20f0`.

Note: The example below shows the configuration specific to the setup presented in this installation guide. IP addresses, device names, and universally unique identifier (UUID) should be respectively changed for a different setup.

```
TYPE=Ethernet
BOOTPROTO=static
IPADDR=10.250.101.10
NETMASK=255.255.255.0
GATEWAY=10.250.101.1
DNS1=10.248.2.1
DNS2=10.2.71.6
DNS3=10.19.1.4
DEFROUTE=yes
PEERDNS=yes
PEERROUTES=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
IPV6_FAILURE_FATAL=no
NAME=ens20f0
UUID=633b582e-7696-4e43-ad1d-ac53bf074250
DEVICE=ens20f0
ONBOOT=yes
NM_CONTROLLED=no
```

4. Disable and stop the `NetworkManager` service, enable/start the `network` service, and bring up all the configured `network` interfaces with the following commands. Note that in the listing below only the interface `ens20f0` is brought up.

```
# yum install net-tools
# systemctl disable NetworkManager
# systemctl stop NetworkManager
# systemctl start network
# chkconfig network on
# ifup ens20f0
```

3.2 Installation of Red Hat OpenStack Platform* 7

For this implementation, PackStack was used to create a multi-host test environment to showcase the security implementation. However, a production deployment would require high availability of the control plane as well as other considerations such as Secure Socket Layer (SSL) endpoints. Red Hat OpenStack Platform Director* is Red Hat's solution to meet the needs of provisioning production environments. You can consult Red Hat production documentation for more information on Director. In addition, a reference architecture for a highly available deployment can be found at:

https://access.redhat.com/articles/1610453

Installation of Red Hat Open Stack Platform 7 must be performed on all of the machines in the setup.

1. Register and subscribe to the Red Hat and Open Stack Platform 7.

```
# subscription-manager register
# subscription-manager subscribe --auto
# subscription-manager list --consumed
```

2. Clear the initially set-up repositories and enable the appropriate ones.

```
# subscription-manager repos --disable=*
# subscription-manager repos
--enable=rhel-7-server-rpms
# subscription-manager repos
--enable=rhel-7-server-rh-common-rpms
# subscription-manager repos
--enable=rhel-7-server-openstack-7.0-rpms
```

3. Install the PackStack installer.

```
# yum install openstack-packstack
```

4. Create PackStack answer file and save it on the disk. In this example, the `/home/myanswerfile.txt` file is used. The answer file used in this setup is presented in Appendix A: The PackStack Answer File.

5. Run PackStack on the created answer file.

```
# packstack —-answer-file=/home/
myanswerfile.txt
```

6. Increase the number of open files in MySQL*.

```
$ mkdir -p /etc/systemd/system/mariadb.
service.d/
$ cat /etc/systemd/system/mariadb.
service.d/limits.conf
[Service]
LimitNOFILE=10000

$ systemctl daemon-reload
$ systemctl restart mariadb
```

7. Verify the open files limit in MySQL.

```
$ mysql
SHOW VARIABLES LIKE 'open%';
```

3.3 Installing Midokura Enterprise MidoNet

Note: The following steps were taken from Midokura's installation guide, which is available at http://docs.midokura.com/docs/latest-en/quick-start-guide/rhel-7_kilo-osp/content/index.html, with respective changes tailored to the setup presented in this guide.

Uninstall the Open vSwitch agent in OpenStack Networking* and install MidoNet, following these steps to integrate ingredients for MidoNet with PackStack. Steps 1 and 2 can be done before OpenStack is installed, so the `yum` upgrade will include the Midokura and DataStax repositories.

1. Verify that the `yum` repositories have the `midokura.repo`.

```
# cat /etc/yum.repos.d/midokura.repo
[mem]
name=MEM
baseurl=http://username:password@repo.
```

```
midokura.com/mem-5/stable/el7/
enabled=1
gpgcheck=1
gpgkey=https://repo.midokura.com/midorepo.
key

[mem-openstack-integration]
name=MEM OpenStack Integration
baseurl=http://repo.midokura.com/
openstack-kilo/stable/el7/
enabled=1
gpgcheck=1
gpgkey=https://repo.midokura.com/midorepo.
key

[mem-misc]
name=MEM 3rd Party Tools and Libraries
baseurl=http://repo.midokura.com/misc/
stable/el7/
enabled=1
gpgcheck=1
gpgkey=https://repo.midokura.com/midorepo.
key
```

2. Enable DataStax repository.

```
# cat /etc/yum.repos.d/datastax.repo
# DataStax (Apache Cassandra)
[datastax]
name = DataStax Repo for Apache
Cassandra
baseurl = http://rpm.datastax.com/
community
enabled = 1
gpgcheck = 1
gpgkey = https://rpm.datastax.com/rpm/
repo_key
```

Change the OpenStack Compute* configuration in all the compute nodes with the following steps.

3. Configure `libvirt`. Edit the `/etc/libvirt/qemu.conf` file to contain the following.

```
user = "root"
group = "root"

cgroup_device_acl = [
  "/dev/null", "/dev/full", "/dev/zero",
  "/dev/random", "/dev/urandom",
  "/dev/ptmx", "/dev/kvm", "/dev/kqemu",
  "/dev/rtc","/dev/hpet", "/dev/vfio/vfio",
  "/dev/net/tun"
]
```

4. Add all the permissions to the folder `/dev/net/tun` to root user and OpenStack Compute service.

5. Restart the `libvirt` service.

```
# systemctl restart libvirtd.service
```

6. Install nova-rootwrap network filters.

```
# yum install openstack-nova-network
# systemctl disable openstack-nova-
network.service
```

7. Restart the OpenStack Compute service.

```
# systemctl restart openstack-nova-
compute.service
```

3.3.1 OpenStack Networking* Integration

Note: The following steps should be executed on the controller/neutron node.

1. Verify that all these components are installed.

```
yum install openstack-neutron openstack-
utils openstack-selinux python-neutron-
plugin-midonet
```

2. Set the OpenStack Networking plug-in. Edit the `/etc/neutron/neutron.conf` file, and configure the following keys in the `[DEFAULT]` section.

```
[DEFAULT]
...
core_plugin = midonet.neutron.plugin_
v2.MidonetPluginV2
...
dhcp_agent_notification = False
...
allow_overlapping_ips = True
```

3. Comment all other plug-ins different from core_plugin.

4. Create the directory for the MidoNet plug-in.

```
mkdir /etc/neutron/plugins/midonet
```

5. Create the `/etc/neutron/plugins/midonet/midonet.ini` file, and edit it to contain the following.

```
[DATABASE]
sql_connection = mysql://neutron:NEUTRON_
DBPASS@controller/neutron

[MIDONET]
# Midonet API URL
midonet_uri = http://controller:8181/
midonet-api
# Midonet administrative user in Keystone
username = midonet
password = MIDONET_PASS
# Midonet administrative user's tenant
project_id = services
```

6. Create a symbolic link to direct OpenStack Networking to the Midonet configuration.

```
# ln -s /etc/neutron/plugins/midonet/
midonet.ini /etc/neutron/plugin.ini
```

7. Create the OpenStack Networking database.

```
$ mysql -u root -p
```

```
CREATE DATABASE neutron character set
utf8;
GRANT ALL PRIVILEGES ON neutron.* TO
'neutron'@'localhost' IDENTIFIED BY
'NEUTRON_DBPASS';
GRANT ALL PRIVILEGES ON neutron.* TO
'neutron'@'%' IDENTIFIED BY 'NEUTRON_
DBPASS';
FLUSH PRIVILEGES;
quit
```

8. Run the `neutron-db-manage` command.

```
# neutron-db-manage \
 --config-file /usr/share/neutron/neutron-
dist.conf \
 --config-file /etc/neutron/neutron.conf \
 --config-file /etc/neutron/plugin.ini \
 upgrade head
```

9. Run the `midonet-db-manage` command.

```
# midonet-db-manage upgrade head
```

10. If the command `midonet-db-manage` has issues with extra fields (such as alembic), perform the next steps.
    - Install the most recent version of `python-neutron-plugin-midonet.<version>.noarch.rpm`
    - Remove any Open vSwitch installation (on all compute and controller nodes).

```
yum erase openstack-neutron-openvswitch
openvswitch python-openvswitch
```

3.3.2 Apache ZooKeeper* Installation

Note: ZooKeper should be installed in at least one NSDB physical machine.

1. Install the Apache ZooKeeper packages.

```
# yum install java-1.7.0-openjdk-headless
# yum install zookeeper zkdump nmap-ncat
```

Common Configuration

2. Configure Apache ZooKeeper. Edit the `/etc/zookeeper/zoo.cfg` file to contain the following:

```
server.1=nsdb1:2888:3888
server.2=nsdb2:2888:3888
server.3=nsdb3:2888:3888
```

3. Create the data directory.

```
# mkdir /var/lib/zookeeper/data
# chown zookeeper:zookeeper /var/lib/
zookeeper/data
```

Note: For production deployments, it is recommended to configure the storage of snapshots in a different disk than the commit log. This can be set by changing the parameter **`dataDir`** in `zoo.cfg` to a different disk.

Node-Specific Configuration

1. NSDB Node 1: Create the `/var/lib/zookeeper/data/myid` file, and edit it to contain the host's ID.

```
# echo 1 > /var/lib/zookeeper/data/myid
```

   NSDB Node 2: Create the `/var/lib/zookeeper/data/myid` file, and edit it to contain the host's ID.

```
# echo 2 > /var/lib/zookeeper/data/myid
```

   NSDB Node 3: Create the `/var/lib/zookeeper/data/myid` file, and edit it to contain the host's ID.

```
# echo 3 > /var/lib/zookeeper/data/myid
```

2. Create the Java* symbolic link.

```
# mkdir -p /usr/java/default/bin/
# ln -s /usr/lib/jvm/jre-1.7.0-openjdk/
bin/java /usr/java/default/bin/java
```

3. Enable and start Apache ZooKeeper.

```
# systemctl enable zookeeper.service
# systemctl start zookeeper.service
```

4. Verify Apache ZooKeeper operation (use 127.0.0.1 only if you are testing it from the controller; otherwise, use the public IP address of the controller).

   After installation of all nodes has been completed, verify that Apache ZooKeeper is operating properly.

   A basic check can be done by executing the `ruok` (Are you ok?) command on all nodes. This will reply with `imok` (I am ok.) if the server is running in a non-error state.

```
$ echo ruok | nc 127.0.0.1 2181
imok
```

   More detailed information can be requested with the stat command, which lists statistics about performance and connected clients.

```
$ echo stat | nc 127.0.0.1 2181
Zookeeper version: 3.4.5--1, built on
06/10/2013 17:26 GMT
Clients:
 /127.0.0.1:34768[0]
(queued=0,recved=1,sent=0)
 /192.0.2.1:49703[1](queued=0,recved=1053,se
nt=1053)
Latency min/avg/max: 0/4/255
Received: 1055
Sent: 1054
Connections: 2
Outstanding: 0
Zxid: 0x260000013d
Mode: follower
Node count: 3647
```

3.3.3 Apache Cassandra* Installation

Perform Apache Cassandra installation on the controller node.

1. Add the Apache Cassandra repository to the /etc/yum.repos.d/datastax.repo.

```
# DataStax (Apache Cassandra)
[datastax]
name = DataStax Repo for Apache Cassandra
baseurl = http://rpm.datastax.com/community
enabled = 1
gpgcheck = 1
gpgkey = https://rpm.datastax.com/rpm/repo_key
```

2. Install the Apache Cassandra packages.

```
# yum install java-1.7.0-openjdk-headless
# yum install dsc20
```

Common Configuration

3. Edit the /etc/cassandra/conf/cassandra.yaml file to contain the following items.

```
# The name of the cluster.
cluster_name: 'midonet'

...
# Addresses of hosts that are deemed contact points.
seed_provider:
  - class_name: org.apache.cassandra.locator.SimpleSeedProvider
    parameters:
      - seeds: "nsdb1,nsdb2,nsdb3"
```

Node-Specific Configuration

4. NSDB Node 1: Edit the /etc/cassandra/conf/cassandra.yaml file to contain the following:

```
# Address to bind to and tell other Cassandra nodes to connect to.
listen_address: nsdb1

...
# The address to bind the Thrift RPC service.
rpc_address: nsdb1
```

NSDB Node 2: Edit the /etc/cassandra/conf/cassandra.yaml file to contain the following:

```
# Address to bind to and tell other Cassandra nodes to connect to.
listen_address: nsdb2
...
# The address to bind the Thrift RPC service.
rpc_address: nsdb2
```

NSDB Node 3: Edit the /etc/cassandra/conf/cassandra.yaml file to contain the following:

```
# Address to bind to and tell other Cassandra nodes to connect to.
listen_address: nsdb3
...
# The address to bind the Thrift RPC service.
rpc_address: nsdb3
```

5. Edit the service's init script.

   On installation, the /var/run/cassandra directory is created, but because it is located on a temporary file system, it will be lost after system reboot. As a result, it is not possible to stop or restart the cassandra service anymore.

   To avoid this, edit the /etc/init.d/cassandra file to create the directory on service start.

```
[...]
case "$1" in
  start)
    # Cassandra startup
    echo -n "Starting Cassandra: "
    mkdir -p /var/run/cassandra
    chown cassandra:cassandra /var/run/cassandra
    su $CASSANDRA_OWNR -c "$CASSANDRA_PROG -p $pid_file" > $log_file 2>&1
    retval=$?
[...]
```

6. Enable and start Apache Cassandra.

```
# systemctl enable cassandra.service
# systemctl start cassandra.service
```

7. Verify that Apache Cassandra is operating properly (use 127.0.0.1 only if you are testing it from the controller; otherwise use the public IP address of the controller).

Note: If Apache Cassandra fails to start and prints a "buffer overflow" error message in its log file, try associating 127.0.0.1 with the hostname in etc/hosts (so that hostname -i will show 127.0.0.1). This may solve the Apache Cassandra start problem.

A basic check can be done by executing the nodetool status command. This will reply with UN (Up / Normal) in the first column if the servers are running in a non-error state.

```
$ nodetool -host 127.0.0.1 status
[...]
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address    Load       Tokens  Owns   Host ID                               Rack
UN  192.0.2.1  123.45 KB  256     33.3%  11111111-2222-3333-4444-555555555555  rack1
UN  192.0.2.2  234.56 KB  256     33.3%  22222222-3333-4444-5555-666666666666  rack1
UN  192.0.2.3  345.67 KB  256     33.4%  33333333-4444-5555-6666-777777777777  rack1
```

3.3.4 Midokura Enterprise MidoNet Cluster Installation

Perform the following steps in controller node.

1. Install the `midonet-cluster` package.

```
# yum install midonet-cluster
```

2. Set up mn-conf. Edit the `/etc/midonet/midonet.conf` to point mn-conf to the Apache ZooKeeper cluster (The `nsdb1`, `nsdb2`, and `nsdb3` should be the actual IP addresses of the servers.):

```
[zookeeper]
zookeeper_hosts = nsdb1:2181,nsdb2:2181,nsdb3:2181
```

3. Configure access to the NSDB.

   This step needs to happen only once; it will set up access to the NSDB for the MidoNet cluster and agent nodes.

   Run the following command to set the cloud-wide values for the Apache ZooKeeper and Apache Cassandra server addresses (The `nsdb1`, `nsdb2`, and `nsdb3` should be the actual IP addresses of the servers.):

```
$ cat << EOF | mn-conf set -t default
zookeeper {
  zookeeper_hosts = "nsdb1:2181,nsdb2:2181,nsdb3:2181"
}

cassandra {
  servers = "nsdb1,nsdb2,nsdb3"
}
EOF
```

   Run the following command to set the Apache Cassandra replication factor.

```
$ echo "cassandra.replication_factor : 3" | mn-conf set -t default
```

4. Configure OpenStack Identity* access. This step needs to happen only once; it will set up access to OpenStack Identity for the MidoNet cluster node(s).

   This step will configure the local MidoNet cluster node to be able to use OpenStack Identity.

```
$ cat << EOF | mn-conf set -t default
cluster.auth {
  provider_class = "org.midonet.cluster.auth.keystone.v2_0.KeystoneService"
  admin_role = "admin"
  keystone.tenant_name = "admin"
  keystone.admin_token = "ADMIN_TOKEN"
  keystone.host = controller
  keystone.port = 35357
}
EOF
```

5. Start the MidoNet cluster.

```
# systemctl enable midonet-cluster.service
# systemctl start midonet-cluster.service
```

3.3.5 Midokura Enterprise MidoNet CLI Installation

Perform the following steps in controller node.

1. Install the MidoNet CLI package.

```
# yum install python-midonetclient
```

2. Configure MidoNet CLI. Create the `~/.midonetrc` file and edit it to contain the following:

```
[cli]
api_url = http://MultinodeController.ch.intel.com:8181/midonet-api
username = admin
password = ADMIN_PASS
project_id = admin
```

3.3.6 MidoNet Agent* (Midolman) Installation

Note: The following steps should be executed on compute nodes.

The MidoNet Agent (Midolman) has to be installed on all nodes where traffic enters or leaves the virtual topology; in this guide these are all of three compute nodes.

1. Install the Midolman package.

```
# yum install java-1.8.0-openjdk-headless
# yum install midolman
```

2. Set up `mn-conf`. Edit the `/etc/midolman/midolman.conf` file to point `mn-conf` to the Apache ZooKeeper cluster.

```
[zookeeper]
zookeeper_hosts = nsdb1:2181,nsdb2:2181,nsdb3:2181
```

3. Configure resource usage. Run these steps on each agent host in order to configure resource usage.

Note: For production environments the large templates are strongly recommended.

Midolman resource template

Run the following command to configure the Midolman resource template.

```
$ mn-conf template-set -h local -t TEMPLATE_NAME
```

Replace TEMPLATE_NAME with one of the following templates:

- agent-compute-large
- agent-compute-medium
- agent-gateway-large
- agent-gateway-medium
- default

Java Virtual Machine (JVM) resource template

Replace the default `/etc/midolman/midolman-env.sh` file with one of the following to configure the JVM resource template:

- /etc/midolman/midolman-env.sh.compute.large
- /etc/midolman/midolman-env.sh.compute.medium
- /etc/midolman/midolman-env.sh.gateway.large
- /etc/midolman/midolman-env.sh.gateway.medium

4. Configure MidoNet metadata proxy for all agents.

   This step needs to happen only once; it will set up MidoNet metadata proxy for all MidoNet Agent nodes.

   Run the following commands to set the cloud-wide values for the Midonet Metadata Proxy.

```
$ echo "agent.openstack.metadata.
nova_metadata_url : \"http://
MultinodeController.ch.intel.com:8775\"" |
mn-conf set -t default
$ echo "agent.openstack.metadata.shared_
secret : shared_secret" | mn-conf set -t
default
$ echo "agent.openstack.metadata.enabled
: true" | mn-conf set -t default
```

   `MultinodeController.ch.intel.com`, `8775`, and `shared_secret` should be replaced with the appropriate values. They need to match with the corresponding OpenStack Compute metadata API configuration.

   `MultinodeController.ch.intel.com` and `8775` specify the address on which OpenStack Compute accepts Metadata API requests. `shared_secret` has to be the same as specified by the `metadata_proxy_shared_secret` field in the neutron section of the `nova.conf` file.

   The OpenStack Compute side of the configuration for the metadata service is the same as when using OpenStack Networking metadata proxy. See the OpenStack documentation for details:

Cloud Administrator Guide: Configure Metadata

The metadata proxy creates an interface named "metadata" on the hypervisor hosts. When using `iptables` it may be necessary to add a rule to accept traffic on that interface.

```
iptables -I INPUT 1 -i metadata -j ACCEPT
```

5. Start Midolman.

```
# systemctl enable midolman.service
# systemctl start midolman.service
```

MidoNet Host Registration

6. Launch the Midonet command line interface (CLI).

```
$ midonet-cli
midonet>
```

7. Create a tunnel zone.

   MidoNet supports the VxLAN and Generic Routing Encapsulation (GRE) protocols to communicate to other hosts within a tunnel zone.

   To use the VxLAN protocol, create the tunnel zone with type `vxlan`.

```
midonet> tunnel-zone create name tz type
vxlan
tzone0
```

   To use the GRE protocol, create the tunnel zone with type `gre`.

```
midonet> tunnel-zone create name tz type
gre
tzone0
```

Note: Make sure to allow GRE/VxLAN traffic for all hosts that belong to the tunnel zone. For VxLAN MidoNet uses User Datagram Protocol (UDP) port 6677 as default. To add hosts to the tunnel zone:

```
midonet> list tunnel-zone
tzone tzone0 name tz type vxlan

midonet> list host
host host0 name MultinodeController.
ch.intel.com alive true
host host1 name CernerMidonet alive true
host host2 name MultinodeCompute1.
ch.intel.com alive true

midonet> tunnel-zone tzone0 add member
host host0 address 172.16.10.10
zone tzone0 host host0 address
172.16.10.10

midonet> tunnel-zone tzone0 add member
host host1 address 172.16.10.14
zone tzone0 host host1 address
172.16.10.14

midonet> tunnel-zone tzone0 add member
host host2 address 172.16.10.12
zone tzone0 host host2 address
172.16.10.12
```

3.4 Adding a Compute Node to OpenStack with MidoNet

On the compute node:

1. Clean up any previous installation of OpenStack.

```
# yum -y remove openstack-glance
# yum -y remove openstack-glance
# yum -y remove openstack-cinder.noarch
# yum -y remove openstack-dashboard
# yum -y remove openstack-swift*
# yum -y remove openstack-packstack*
# yum -y remove openstack-ceilometer-
alarm.noarch
# yum -y remove openstack-ceilometer-api.
noarch
# yum -y remove openstack-ceilometer-
central.noarch
# yum -y remove openstack-ceilometer-
collector.noarch
# yum -y remove openstack-ceilometer-
notification.noarch
# yum -y remove openstack-keystone.noarch
# yum -y remove openstack-neutron-lbaas.
noarch
# yum -y remove openstack-neutron-
midonet.noarch
# yum -y remove openstack-neutron-ml2.
noarch
# yum -y remove openstack-nova-api.noarch
# yum -y remove openstack-nova-cert.
noarch
# yum -y remove openstack-nova-conductor.
noarch
# yum -y remove openstack-nova-console.
noarch
```

```
# yum -y remove openstack-nova-
novncproxy.noarch
# yum -y remove openstack-nova-scheduler.
noarch
# yum -y remove rabbitmq-server.noarch
```

2. Clean up any previous MidoNet installation.

```
# yum -y remove cassandra20
# yum -y remove zookeeper
# yum -y remove python-midonetclient
# yum -y remove midolman
# yum -y remove midonet-cluster
# rm /etc/yum.repos.d/midokura.repo
# rm /etc/yum.repos.d/datastax.repo
# rm /etc/midolman
# rm /etc/midonet_host_id.properties
```

3. Keep or install the following components:
   - openstack-ceilometer-common.noarch
   - openstack-ceilometer-compute.noarch
   - openstack-neutron-common.noarch
   - openstack-neutron.noarch
   - openstack-nova-common.noarch
   - openstack-nova-compute.noarch
   - openstack-selinux.noarch
   - openstack-utils.noarch
4. Remove any Open vSwitch bridges (that is, br-ex) and make static IP addresses directly in the networks interfaces' scripts.
5. Copy the yum repositories from the current compute nodes (any node that is currently running).

```
# scp root@10.250.101.12:/etc/yum.repos.d/
midokura.repo \
     /etc/yum.repos.d/
```

6. Install Midolman.

```
# yum –y midolman
```

7. Copy the MidoNet configuration file.

```
# scp root@10.250.101.12:/etc/midolman/
midolman.conf \
     /etc/midolman/
```

8. Copy the configuration file for OpenStack Compute and `libvirt`.

```
# scp root@10.250.101.12:/etc/nova/* /etc/
nova/
# scp root@10.250.101.12:/etc/libvirt/* /
etc/libvirt/
# scp root@10.250.101.12:/etc/libvirt/
qemu/* /etc/libvirt/qemu/
```

9. Replace the property `vncserver_proxyclient_address` in `/etc/nova/nova.conf` and set the name of the new compute node.

```
vncserver_proxyclient_
address=MultinodeCompute1.ch.intel.com
```

10. Enable Midolman.

```
# systemctl enable midolman
```

11. Enable OpenStack Compute.

```
# systemctl enable openstack-nova-compute
```

12. Reboot the machine.

On the controller node:

13. Open the MidoNet console.

```
# midonet-cli
```

14. List the host in the setup.

```
midonet> list host
```

The output should be similar to the following:

```
host host0 name MultinodeController.
ch.intel.com alive true addresses
    fe80:0:0:0:900e:a9ff:fef7:f5d2,fe80:0:0:0:c8a7
    :1ff:fe6b:e70b,fe80:0:0:0:30ef:a0:ec9d:18ff:fe0
    7:8abf,fe80:0:0:0:443f:e9ff:fe7b:cf01,fe80:0:0
    :0:50ef:3aff:fed3:8011,fe80:0:0:0:21e:67ff:fec
    b:efaa,10.250.101.10,127.0.0.1,0:0:0:0:0:0:0f:fe
    cb:efa9,172.16.10.10,fe80:0:0:0:e443:19ff:feaa
    :1ff0,fe80:0:0:0:c086:ddff:fe82:e60e,fe80:0:0:0
    :5034:dfff:fe53:d499,fe80:0:0:0:46c:24ff:fe09:b
    323,5:6d35,172.16.101.2,fe80:0:0:0:147e:92ff:fe
    f9:a539,fe80:0:0:0:50c9:5bff:fe7e:4afc,169.254
    .169.254,fe80:0:0:0:2c46:11ff:fe2e:d6d5,fe80:0
    :0:0:f4c3:4fff
host host1 name CernerMidonet alive true
addresses
    fe80:0:0:0:2c1a:5fff:fed5:24cc,fe80:
    0:0:0:225:90ff:fefd:4708,169.254.169.
    254,fe80:0:0:0:8d2:4fff:f:3ff:fe00:60
    be,127.0.0.1,0:0:0:0:0:0:0:1,192.168.122-
    .1,10.250.101.14,fe80:0:0:0:225:90ff:fefd:4709
host host2 name MultinodeCompute1.ch.intel.
com alive true addresses
    127.0.0.1,0:0:0:0:0:0:0:1,fe80:0:0:0:18f5:1fff:f
    e46:e8a4,10.250.101.12,fe80:0:0:00:0:0:0:8c6f:
    1aff:fe4e:9af8,10.250.101.12,fe80:0:0:0:21e:67f
    f:fecf:c09f,fe80:0:0:0:b4d7:58ff:fef1:e383,fe8
    0:0:0:0:1812:33ff:fe77:3bb,fe80:0:0:0:7c85:69.2
    54,fe80:0:0:0:74c7:4fff:feba:cf28,fe80:0:0:0:8
    0f6:93ff:fe0f:8601,fe80:0:0:0:21e:67ff:fec5:508
    5,172.16.101.3,fe80:0:0:0:d8bd:b7ff:fe6e:15d8
```

15. Get the host name from the previous list; in this case it is `host1`.
16. Get the tunnel zone list.

```
midonet> tunnel-zone list
```

The output should be similar to the following:

```
tzone tzone0 name vxlan type vxlan
```

17. Get the tunnel zone name; in this case it is: `tzone0`.
18. Get the list of member in the tunnel zone.

```
midonet> tunnel-zone tzone0 member list
```

The output should be similar to the following:

```
zone tzone0 host host2 address
172.16.10.12
zone tzone0 host host0 address
172.16.10.10
zone tzone0 host host1 address
172.16.10.14
```

19. If the member already exists with a different IP, it is possible to remove it with this command.

```
midonet> tunnel-zone tzone0 delete
member host host1
```

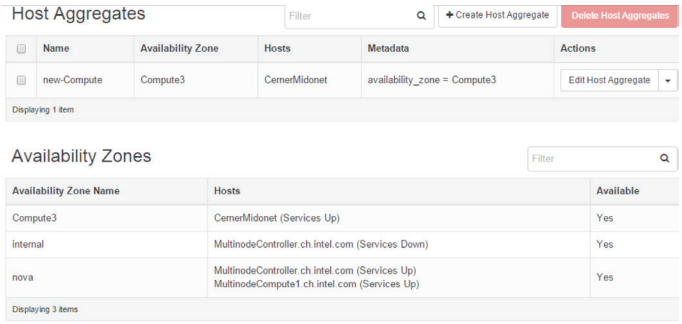
20. Add the new host machine to the members of the tunnel.

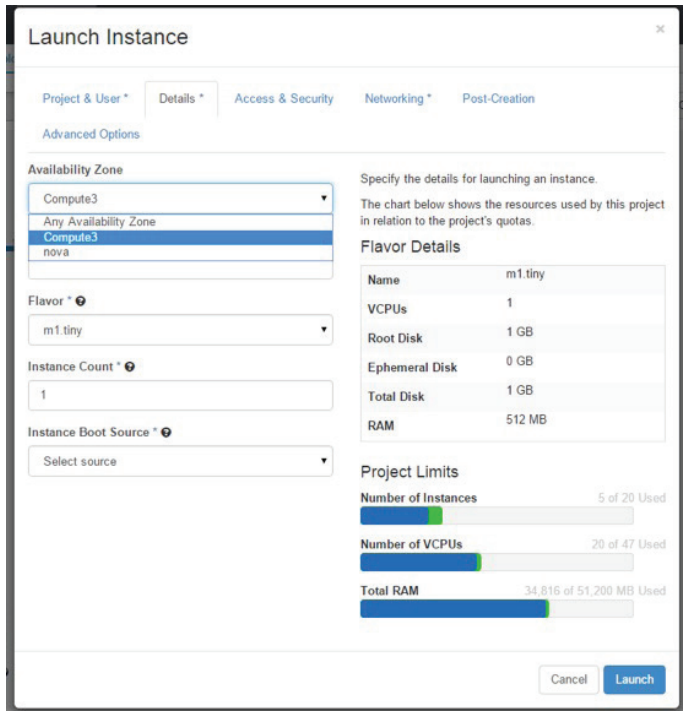
```
midonet> tunnel-zone tzone0 add member
host host1 address 10.250.101.10
```

21. Check whether the new host is a member of the tunnel zone with the command for checking the tunnel list.

3.4.1 Testing the New Compute Node

1. Create new host aggregates. Add only the new server as an availability zone.
2. Create a new instance. Check that the availability zone is set to the new server.
3. Ping machines in other availability zones as well as the floating IP addresses.





3.5 Adding Uplink to an External (Public) Network

1. Create an OpenStack network that will be mapped to a physical network. Usually that is done with the following command:

```
# neutron net-create Public --provider:network_type flat --provider:physical_network
enp4s0f3 --router:external=True
Created a new network:
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| id                  | 9209b2c6-0803-4ba9-98b5-5ade96afe0e0 |
| name                | Public                               |
| port_security_enabled | True                               |
| router:external     | True                                 |
| shared              | False                                |
| status              | ACTIVE                               |
| subnets             |                                      |
| tenant_id           | a88f84c8ab2e4841b7579518502c1e78     |
+---------------------+--------------------------------------+

# neutron subnet-create Public 10.250.101.0/24 --name public_subnet --gateway 10.250.101.1
--allocation-pool start=10.250.101.133,end=10.250.101.149 --enable-dhcp=False
Created a new subnet:
+-------------------+------------------------------------------------------+
| Field             | Value                                                |
+-------------------+------------------------------------------------------+
| allocation_pools  | {"start": "10.250.101.133", "end": "10.250.101.149"} |
| cidr              | 10.250.101.0/24                                      |
| dns_nameservers   |                                                      |
| enable_dhcp       | False                                                |
| gateway_ip        | 10.250.101.1                                         |
| host_routes       |                                                      |
| id                | c9ae94e8-44d1-42c1-8bcd-904c22d0ace6                 |
| ip_version        | 4                                                    |
| ipv6_address_mode |                                                      |
| ipv6_ra_mode      |                                                      |
| name              | public_subnet                                        |
| network_id        | 9209b2c6-0803-4ba9-98b5-5ade96afe0e0                 |
| subnetpool_id     |                                                      |
| tenant_id         | a88f84c8ab2e4841b7579518502c1e78                     |
+-------------------+------------------------------------------------------+
```

Take the ID of the network; in this case: id 9209b2c6-0803-4ba9-98b5-5ade96afe0e0 and create a port.

```
# midonet-cli -e bridge 9209b2c6-0803-4ba9-98b5-5ade96afe0e0 port create 3da71887-d49f-4199-
8a4d-10e7ba3b6387
```

That gives a port id; in this case (3da71887-d49f-4199-8a4d-10e7ba3b6387). Execute:

```
# midonet-cli -e host <host-id> add binding interface <interface> port <port-id>
```

The host id is on /etc/midonet_host_id.properties.

```
# cat /etc/midonet_host_id.properties
#Fri Nov 06 23:53:25 MST 2015
host_uuid=09a1ee22-0faf-4560-a823-f386ed0ef94f
```

Thus, the adding bind should be as follows:

```
# midonet-cli -e host 09a1ee22-0faf-4560-a823-f386ed0ef94f add binding interface enp4s0f3
port 3da71887-d49f-4199-8a4d-10e7ba3b6387
```

At least one compute node should have two network interfaces connected to the same external network, and in the midonet-cli the host and the interface should come from that compute node.

2. Modify the routing rules in MidoNet for the uplink connection.
   In order to avoid sending the traffic outside to the external gateway, we have to set some routing rules that will identify the traffic between the floating IP address range and use the tunnel. To do that, access midonet-cli and run these commands.

```
# router router0 add route dst 10.250.101.0/24 src 0.0.0.0/0 type normal port router0:port0
# router router1 add route dst 10.250.101.0/24 src 0.0.0.0/0 type normal port router1:port0
# router router2 add route dst 10.250.101.0/24 src 0.0.0.0/0 type normal port router2:port0

# midonet-cli

midonet> host list
host host0 name MultinodeController.ch.intel.com alive true addresses fe80:0:0:0:4066:a3ff:fe
b2:c157,fe80:0:0:0:5cad:60ff:feef:f391,fe80:0:0:0:b491:3dff:fe0b:89dc,127.0.0.1,0:0:0:0:0:0:0:1,10.250
.101.10,fe80:0:0:0:21e:67ff:fecb:efa9,fe80:0:0:0:a803:42ff:fe2c:872b,172.16.101.2,fe80:0:0:0:9cab:22ff:
fe45:5544,fe80:0:0:0:5c3a:a0ff:feb3:e125,fe80:0:0:0:6cbd:2ff:febe:5ffb,fe80:0:0:0:20f0:6dff:fe89:9696,
fe80:0:0:0:c0f8:adff:fe38:fc58,fe80:0:0:0:21e:67ff:fecb:efaa,10.250.101.11,fe80:0:0:0:bc19:ddff:fe4d:3
9c2,172.16.10.10,fe80:0:0:0:3efd:feff:fe9e:7570,fe80:0:0:0:a046:27ff:fe07:c5da,169.254.169.254,fe80:0:0:
0:e0fa:42ff:fe61:c015 flooding-proxy-weight 1
host host1 name CernerMidonet alive true addresses 127.0.0.1,0:0:0:0:0:0:0:1,fe80:0:0:0:2882:adff
:fe93:e61e,10.250.101.15,fe80:0:0:0:225:90ff:fefd:4709,fe80:0:0:0:70bd:8ff:fecd:6c7f,169.254.169.254,fe8
0:0:0:0:a490:f3ff:fe78:b824,192.168.122.1,fe80:0:0:0:e486:1eff:fe67:b961,fe80:0:0:0:583e:50ff:fe6b:d2f8,
fe80:0:0:0:5c2e:c0ff:fe5d:fb43,fe80:0:0:0:e0bc:89ff:fe49:4d3b,fe80:0:0:0:2035:f7ff:feef:d35a,fe80:0:0:0
:4c19:33ff:fe7f:d60c,fe80:0:0:0:225:90ff:fefd:4708,10.250.101.14,fe80:0:0:0:884e:a2ff:fee4:d630,172.16.10
.14,fe80:0:0:0:3efd:feff:fe9e:7238,fe80:0:0:0:70d7:f7ff:fee8:c5c7,fe80:0:0:0:54dc:9dff:fed7:d89f,fe80:0:
0:0:100f:a4ff:fefe:1ede,fe80:0:0:0:8422:2aff:fea7:7b0d flooding-proxy-weight 1
host host2 name MultinodeCompute1.ch.intel.com alive true addresses
127.0.0.1,0:0:0:0:0:0:0:1,fe80:0:0:0:8f3:46ff:fe7e:8e6f,fe80:0:0:0:cca7:89ff:fe9f:4333,fe80:0:0:0:30f1:2
9ff:fe12:48c0,172.16.10.12,fe80:0:0:0:3efd:feff:fe9e:77d8,169.254.169.254,fe80:0:0:0:6cf6:dcff:fe08:f6a
5,fe80:0:0:0:9cfa:3bff:fe79:8b66,10.250.101.13,fe80:0:0:0:5412:5eff:fef0:175,fe80:0:0:0:98b7:beff:fe02:8
265,172.16.101.3,fe80:0:0:0:8c32:8aff:fe03:a501,fe80:0:0:0:d4ae:2cff:fef8:dbb1,fe80:0:0:0:2ca0:2cff:fe2
4:e53c,fe80:0:0:0:6831:3dff:fe8c:61be,fe80:0:0:0:c03d:ebff:fe30:a260,fe80:0:0:0:4c39:11ff:fec4:8c7e,10
.250.101.12,fe80:0:0:0:21e:67ff:fecf:c09e,fe80:0:0:0:7c04:beff:fea6:9c79,fe80:0:0:0:6893:aeff:fed8:9e00
flooding-proxy-weight 1

midonet> host host0 binding list
host host0 interface tapa8ddb6b3-b7 port bridge0:port0
host host0 interface tapaa0eef80-7b port bridge0:port1
host host0 interface tap06e55253-36 port bridge1:port0
host host0 interface enp4s0f3 port bridge2:port0
host host0 interface tap36af40a5-dc port bridge3:port0
host host0 interface tap48ae50e0-2d port bridge0:port2
host host0 interface tapc85d5edc-d7 port bridge4:port0
host host0 interface tap3af79871-4f port bridge5:port0
host host0 interface tap7be72a65-e4 port bridge3:port1
host host0 interface tap17c40252-30 port bridge0:port3
host host0 interface tapad0c3c61-9a port bridge4:port1
host host0 interface tap0282a387-26 port bridge5:port1
host host0 interface tap3456ac90-e3 port bridge6:port0
host host0 interface tapdbb393b0-42 port bridge7:port0

midonet> bridge list
bridge bridge1 name Tenant2-net state up
bridge bridge0 name Data state up
bridge bridge5 name DataCache state up
bridge bridge2 name Public state up
bridge bridge7 name vsf-inspection-net state up
bridge bridge3 name Mgmt_BKP state up
bridge bridge4 name Client state up
bridge bridge6 name vsf-mgmt-net state up

midonet> bridge bridge2 port list
port port0 device bridge2 state up plugged yes vlan 0
port port1 device bridge2 state up plugged no vlan 0
port port2 device bridge2 state up plugged no vlan 0 peer router0:port0
```

```
port port3 device bridge2 state up plugged no vlan 0 peer router1:port0
port port4 device bridge2 state up plugged no vlan 0 peer router2:port0

midonet> router list
router router2 name Router state up infilter chain0 outfilter chain1 asn -1
router router1 name Router state up infilter chain2 outfilter chain3 asn -1
router router0 name Router state up infilter chain4 outfilter chain5 asn -1

midonet> router router0 route list
route route0 src 0.0.0.0/0 dst 10.250.101.134 port router0:port0 weight 100 learned false
route route1 type normal src 0.0.0.0/0 dst 10.250.101.0/24 port router0:port0 weight 0 learned
false
route route2 type normal src 0.0.0.0/0 dst 172.16.36.0/24 port router0:port1 weight 100
learned false
route route3 src 0.0.0.0/0 dst 172.16.36.1 port router0:port1 weight 100 learned false

midonet> router router1 route list
route route0 src 0.0.0.0/0 dst 10.250.101.140 port router1:port0 weight 100 learned false
route route1 type normal src 0.0.0.0/0 dst 0.0.0.0/0 gw 10.250.101.1 port router1:port0 weight
100 learned false
route route2 type normal src 0.0.0.0/0 dst 172.16.58.0/24 port router1:port1 weight 100
learned false
route route3 src 0.0.0.0/0 dst 172.16.58.1 port router1:port1 weight 100 learned false
route route4 type normal src 0.0.0.0/0 dst 172.16.24.0/24 port router1:port2 weight 100
learned false
route route5 src 0.0.0.0/0 dst 172.16.24.1 port router1:port2 weight 100 learned false
route route6 type normal src 0.0.0.0/0 dst 172.16.65.0/24 port router1:port3 weight 100
learned false
route route7 src 0.0.0.0/0 dst 172.16.65.1 port router1:port3 weight 100 learned false

midonet> router router0 port list
port port0 device router0 state up plugged no mac fa:16:3e:6f:6d:9d address 10.250.101.134 net
10.250.101.0/24 peer bridge2:port2
port port1 device router0 state up plugged no mac fa:16:3e:6a:bf:67 address 172.16.36.1 net
172.16.36.0/24 peer bridge6:port1

midonet> router router0 add route dst 10.250.101.0/24 src 0.0.0.0 type normal port
router0:port0
router0:route4

midonet> router router0 route list
route route0 src 0.0.0.0/0 dst 10.250.101.134 port router0:port0 weight 100 learned false
route route1 type normal src 0.0.0.0/0 dst 0.0.0.0/0 gw 10.250.101.1 port router0:port0 weight
100 learned false
route route4 type normal src 0.0.0.0/0 dst 10.250.101.0/24 port router0:port0 weight 0 learned
false
route route2 type normal src 0.0.0.0/0 dst 172.16.36.0/24 port router0:port1 weight 100
learned false
route route3 src 0.0.0.0/0 dst 172.16.36.1 port router0:port1 weight 100 learned false

midonet> router router1 add route dst 10.250.101.0/24 src 0.0.0.0 type normal port
router1:port0
router0:route8

midonet> router router2 route list
route route0 src 0.0.0.0/0 dst 10.250.101.141 port router2:port0 weight 100 learned false
route route1 type normal src 0.0.0.0/0 dst 0.0.0.0/0 gw 10.250.101.1 port router2:port0 weight
100 learned false
route route2 type normal src 0.0.0.0/0 dst 172.16.90.0/24 port router2:port1 weight 100
learned false
route route3 src 0.0.0.0/0 dst 172.16.90.1 port router2:port1 weight 100 learned false

midonet> router router2 add route dst 10.250.101.0/24 src 0.0.0.0 type normal port
router2:port0
router2:route4
```

3.6 Midokura Enterprise MidoNet Analytic Installation

Note: The following steps were taken from the Midokura's installation guide, which is available at http://docs.midokura.com/docs/latest-en/quick-start-guide/rhel-7_kilo-osp/content/index.html, with respective changes tailored to the setup presented in this guide.

3.6.1 Prerequisites

The Analytics Node must contain a deployment of Logstash* (version 1.5) and Elasticsearch* (version 1.7), as well as the elasticsearch-curator tool, before the installation of the midonet-analytics package.

Elastic* (https://www.elastic.co/) provides both 'deb' and 'rpm' packages for easy installation of the required packages:

- Logstash (v1.5.4): https://www.elastic.co/guide/en/logstash/1.5/package-repositories.html
- Elasticsearch (v1.7.3):

  https://www.elastic.co/guide/en/elasticsearch/reference/1.7/setup-repositories.html

3. The elasticsearch-curator tool can be installed with the `pip` command.

```
# yum install -q python-pip
# pip install -U elasticsearch-curator
```

4. In the controller node we need to have `midonet-cluster-mem installed`. This can be done with the following command.

```
# yum install midonet-cluster-mem
```

5. In the compute nodes, the `midonet-jmxscraper` packages should be installed. To install the `midonet-jmxscraper` package in the agent nodes (compute nodes), execute the following command as root.

```
# yum install midonet-jmxscraper
```

6. The analytic node is a single dedicated node containing the data analytics services as well as the data storage and search engine. The data storage and search engine services are provided by Logstash and Elasticsearch respectively, which have to be installed in advance, according to the instruction in the Prerequisites section.

   It is also recommended to install the `midonet-tools` package in the analytic node; this makes the `mn conf` command available in the analytic node, facilitating its configuration.

   To install the `midonet-analytics` package in the analytic node, execute the following commands as root.

```
# yum install midonet-tools
# yum install midonet-analytics
```

3.6.2 Quickstart

As a prerequisite, the core MidoNet components should be correctly installed and configured.

In particular, the NSDB must contain the correct values for the list of Apache ZooKeeper instances (in the `zookeeper.zookeeper_hosts` key). Note that this value was not required for legacy installations and may be missing or incorrectly set; in order to set it properly, you can run the following command (where `nsdb1, nsdb2`, and `nsdb3` are the NSDB nodes containing the Apache ZooKeeper instances).

```
$ cat << EOF | mn-conf set -t default
zookeeper {
zookeeper_hosts = "nsdb1:2181,nsdb2:2181,nsdb3:2181"
}
EOF
```

Apache Cassandra must also be configured correctly in NSDB for the Midokura Enterprise MidoNet features. In order to ensure the correct settings, you may run the following command (where `cass1`, `cass2`, and `cass3` are the IP addresses of the nodes containing the Apache Cassandra instances).

```
$ cat << EOF | mn-conf set -t default
cassandra {
servers = "cass1,cass2,cass3"
cluster = "midonet"
}
EOF
```

This quickstart assumes that the `midonet-analytics` package has been installed in a single dedicated node (the analytics node). The minimum setup needed for running the Midokura Enterprise MidoNet features solution consists of configuring the communication endpoints for the MidoNet Agent and the Midokura Enterprise MidoNet services, as well as setting up the authentication information to make the data available to MidoNet Manager.

Before starting, it is recommended to increase the default heap size for both Logstash and Elasticsearch up to a minimum of 4 GB respectively (note that it is not recommended to set the total beyond half of the available physical RAM). These values can be set by editing the corresponding configuration files.

On Red Hat Enterprise Linux:

- Set `LS_HEAP_SIZE="4g"` in the `/etc/sysconfig/logstash` file for Logstash.
- Set `ES_HEAP_SIZE="4g"` in the `/etc/sysconfig/elasticsearch` file or Elasticsearch.
- Restart the services.

The next step is to create a configuration file in the analytics node indicating how to locate the NSDB nodes containing the MidoNet configuration. MidoNet Analytics tries to locate this information at the following location: /etc/midonet/midonet-analytics.conf; if the file does not exist, it searches in the following locations:

- $HOME/.midonetrc
- /etc/midonet/midonet.conf
- /etc/midolman/midolman.conf

The following is the template for this configuration data (the value of the zookeeper_hosts key should be a comma-separated list of the IP addresses and ports of the Apache ZooKeeper instances, represented in the template by nsdb1, nsdb2, and nsdb3—the standard port for Apache ZooKeeper is 2181).

```
[zookeeper]
zookeeper_hosts = nsdb1:2181,nsdb2:2181,nsdb3:2181
```

The JMXScraper service running in the compute nodes needs the same configuration file (if that file does not exist, it will use the same configuration as the Midonet Agent running in the same node).

Once the NSDB location has been set via the midonet-analytics.conf file, all the Analytics services can be configured via the MidoNet's mn-conf tool (a tool used to manage the Midonet configuration stored on NSDB). The mn-conf tool is part of the midonet-tools package, and it can be used from any MidoNet node where this package is installed.

The following is the example configuration. On a node with the mn-conf tool available, create a file according to the following template:

```
clio.enabled : true
clio.service.udp_port : 5001
clio.service.encoding : "binary"
clio.data.fields : [ "cookie", "devices",
"host_uuid", "in_port",
"in_tenant", "out_ports", "out_tenant",
"match_eth_src", "match_eth_dst",
"match_ethertype", "match_network_dst",
"match_network_src",
"match_network_proto", "match_src_port",
"match_dst_port", "action_drop",
"action_arp_sip", "action_arp_tip",
"action_arp_op", "rules",
"sim_result", "final_eth_src", "final_eth_
dst", "final_net_src",
"final_net_dst", "final_transport_src",
"final_transport_dst",
"timestamp", "type" ]
calliope.enabled : true
calliope.service.ws_port : 18181
calliope.auth.ssl.enabled : true
jmxscraper.enabled : true
jmxscraper.target.udp_endpoint :
"analytics_ip:5000"
mem_cluster.flow_tracing.enabled : true
mem_cluster.flow_tracing.service.ws_port :
8400
mem_cluster.flow_tracing.auth.ssl.enabled
: true
agent.flow_history.enabled : true
agent.flow_history.encoding : "binary"
agent.flow_history.udp_endpoint :
"analytics_ip:5001"
jmxscraper.target.udp_endpoint :
"analytic_ip:5002"
clio.target.udp_endpoint : " analytic_
ip:5002"
```

The template above assumes that the MidoNet Manager will use SSL connections to communicate with the Midokura Enterprise MidoNet services; if this is not necessary (for example, because the services are behind a proxy), SSL can be deactivated by setting the *.ssl.enabled properties to false (see the configuration section for more details).

The above configuration also assumes that Logstash and Elasticsearch are installed in a single analytics node, together with the midonet-analytics package; <analytics_ip> represents the IP address or host name of such analytics node.

The configuration template can be applied by creating a file named analytics_settings.conf with the placeholders from the template replaced with the proper values, and then applying these settings with.

```
$ mn-conf set -t default < analytics_
settings.conf
```

By default, the analytics node receives data via UDP to ports 5000 and 5001 from the cluster and agent nodes, and exposes the interface to the MidoNet Manager via port 8080 (websocket). The analytics node needs to access the Apache ZooKeeper ensemble (port 2181) in the MidoNet cluster nodes and also the OpenStack Identity service (port 35357) for authentication. The flow tracing data is exposed to MidoNet Manager via port 8460 (websocket) in the cluster nodes.

3.6.3 Configuration

The Midokura Enterprise MidoNet services are configured using the MidoNet mn-conf command, provided by the midonet-tools package.

The following options can be set for the analytics node:

- clio.enabled (true|false). Enable or disable flow history collection service (default is false - disabled).
- clio.service.udp_port (port number). The port to listen for Midonet Agent flow history reports (default is 5001). This value must be the same as the port in the agent setting agent.flow_history.udp_endpoint.
- clio.service.encoding ("binary"|"json"|"none"). The encoding of the data received from MidoNet Agents. It must match the value in agent.flow_history.encoding. The binary setting offers a more efficient encoding of the data, while json allows easier debugging.

- clio.service.fields (list of comma-separated, quoted strings). The list of flow attributes to collect. Note that some of the values in this list are required for some of the Midokura Enterprise MidoNet features and removing them may result in limited functionality. For example, removing the "devices" field from the list might result in not having this information showed in the MidoNet Manager. The default list of reported attributes per flow is:
- cookie: internal flow id.
- devices: list of devices traversed by the flow.
- host_uuid: the id of the host generating the flow.
- in_port: the id of the port originating the flow.
- in_tenant: tenant associated to the device originating the flow.
- out_ports: the ids of the target ports for the flow.
- out_tenant: tenant associated to the device receiving the flow.
- match_eth_src: source Medium Access Control (MAC) address.
- match_eth_dst: destination MAC address.
- match_ethertype: ethernet type of flow.
- match_network_src: source IP address.
- match_network_dst: destination IP address.
- match_network_proto: the protocol id.
- match_src_port: source port.
- match_dst_port: destination port.
- action_drop: the flow was dropped.
- action_arp_sip: Source IP address (SIP) for Address Resolution Protocol (ARP) packages.
- action_arp_tip: Target IP address (TIP) for ARP packages.
- action_arp_op: ARP operation.
- rules: list of rules affecting the flow.
- sim_result: Midonet simulation result for this flow.
- final_eth_src: the source MAC address after any actions taken by the flow.
- final_eth_dst: the destination MAC address after any actions taken by the flow.
- final_net_src: the source IP address after any actions taken by the flow.
- final_net_dst: the destination IP address after any actions taken by the flow.
- final_transport_src: the source port after any actions taken by the flow.
- final_transport_dst: the destination port after any actions taken by the flow.
- timestamp: time stamp for the reception of the flow information.
- type: internal data set identifier.
- clio.service.compress (list of comma-separated, quoted strings). A subset of the flow attributes listed in clio.service.fields that are to be stored in compressed format. Note that compressed fields may reduce the storage space, but the values are not searchable. No field set is compressed by default.
- calliope.enabled (true|false). Enable or disable the Analytics front-end service (default is false - disabled).
- calliope.service.ws_port (port number). The port to listen for Analytics queries from the MidoNet Manager (default is 8080).
- calliope.auth.ssl.enabled (true|false). Enable or disable SSL support (default is true).

The following options can be set for the nodes running the MidoNet Agent and the jmxscraper:

- agent.flow_history.enabled (true|false). Enable or disable flow history reporting in the Agent (default is false).
- agent.flow_history.encoding ("binary"|"json"|"none"). The format of the flow history records emitted by the Midonet Agents. The value must match the setting clio.service.encoding in the analytics node; the recommended value is binary and none does not generate any data (default is none).
- agent.flow_history.udp_endpoint ("host:port"). The host name or IP address of the analytics node and the destination port. The port must match the port set for the clio.service.udp_port property in the Analytics node. No default value is provided.
- jmxscraper.enabled (true|false). Enable or disable the MidoNet Agent JMX API scraper (default is false).
- jmxscraper.interval (time spec). The time interval between consecutive JMX API polls (default is 60s). Note that due to technical reasons, intervals below 60s are not supported.
- jmxscraper.target.udp_endpoint ("host:port"). The host name or IP address of the analytics node, and the port where the storage system (Logstash) is listening (the recommended settings use port 5000). No default value is provided.

The following options can be set for the nodes containing the Cluster:

- mem_cluster.flow_tracing.enabled (true|false). Enable or disable the flow tracing feature (default is true).
- mem_cluster.flow_tracing.service.ws_port (port number). The port to listen for flow tracing queries (default is 8460).
- mem_cluster.flow_tracing.auth.ssl.enabled (true|false). Enable or disable SSL support (default is true).

Apart from the options above, the midonet-analytics package installs a cron job in the analytics node to purge old data from the storage system. The initial configuration limits the size of the stored data to 100 GB; in order to modify this value, you may edit the following file and change the threshold value.

```
/etc/cron.d/midonet-elk-cleaner
```

3.6.4 Files and Directories

- /etc/midonet/midonet-analytics.conf: Bootstrap configuration with Apache ZooKeeper location.
- /etc/midonet/midonet-{calliope,clio}/logback.xml: Logging settings for the Midokura Enterprise Midonet components in the analytics node.
- /etc/midonet/midonet-jmxscraper/logback.xml: Logging settings for the JMX scraper component in the Agent Nodes.
- /etc/midonet-cluster/logback.xml: Logging settings for the cluster node, including the Midokura Enterprise MidoNet extensions.
- /etc/midonet/midonet-*/logback.xml: Logging settings for the different Midokura Enterprise MidoNet components.
- /etc/logstash/conf.d/midonet.conf: Configuration for the Logstash storage back-end in the analytics node.
- /etc/default/logstash: Startup options for Logstash in Debian*-based systems.
- /etc/default/elasticsearch: Startup options for Elasticsearch in Debian-based systems.
- /etc/sysconfig/logstash: Startup options for Logstash in rpm-based systems.
- /etc/sysconfig/elasticsearch: Startup options for Elasticsearch in RPM*-based systems.
- /etc/init/midonet-{analytics,calliope,clio,jmxscraper}.conf: Upstart configuration files for Analytics components.
- /usr/lib/systemd/system/midonet-{analytics,calliope,clio,jmxscraper}.service: systemd configuration files for Analytics components.
- /var/lib/midonet/analytics_host_id.properties: Contains the MidoNet host id that the mn-conf tool uses to identify the Analytics Node components and the JMX scraper in the agent nodes.
- /var/lib/logstash/data: Contains the data stored by the Analytics back-end storage system (Logstash).
- /etc/cron.d/midonet-elk-cleaner: Automated data purging script for Midokura Enterprise MidoNet storage back-end.
- /var/log/midonet/{clio,calliope,jmxscraper}.log: Log files for the different Analytics components in the Analytics Node and the Agent Nodes.
- /var/log/midonet-cluster/midonet-cluster.log: Log file for the cluster node components, including the Midokura Enterprise MidoNet extensions.
- /usr/share/midonet-{analytics,calliope,clio,jmxscraper}: Directories with Analytics management scripts and dependencies.

3.6.5 Usage

Once you have configured the services, you can start the Midokura Enterprise MidoNet services by executing the following commands as root.

On analytics node, execute the following commands.

```
# systemctl start logstash
# systemctl start elasticsearch
# systemctl start midonet-analytics
```

On cluster nodes, execute the following command.

```
# systemctl restart midonet-cluster
```

On agent nodes, execute the following commands.

```
# systemctl restart midolman
# systemctl start midonet-jmxscraper
```

3.6.6 Update the OpenStack EndPoints

This is required by the MidoNet SDN Controller API in the Open Security Controller (OSC).

Create a script file called mn_endpoint_config.sh in the Controller. Add the following content to the file:

```
#!/bin/bash

##
## Usage:
## mn_endpoint_config.
sh controllerIP:controllerPort
analyticsIP:analyticsPort
##
MIDOAPI="$1"
MEMAPI="$2"
if [ -z "$MIDOAPI" -o "${MIDOAPI%:*}" =
"$MIDOAPI" ]; then
  echo invalid host:port for Midonet API
  exit 1
fi

if [ -z "$MEMAPI" -o "${MEMAPI%:*}" =
"$MEMAPI" ]; then
  echo invalid host:port for MEM Insights
API
  exit 1
fi

keystone service-create --name=midonet-
api --type=midonet \
  --description="Midonet REST API"

keystone endpoint-create \
  --service-id=$(keystone service-list
|awk '/ midonet-api / {print $2}') \
  --publicurl="http://$MIDOAPI/midonet-
api" \
  --internalurl="http://$MIDOAPI/midonet-
api" \
  --adminurl="http://$MIDOAPI/midonet-api"
\
```

```
  --region="RegionOne"

keystone service-create --name=mem-
insights --type=insights \
  --description="MEM Insights API"

keystone endpoint-create \
  --service-id=$(keystone service-list
|awk '/ mem-insights / {print $2}') \
  --publicurl="ws://$MEMAPI/analytics" \
  --internalurl="ws://$MEMAPI/analytics" \
  --adminurl="ws://$MEMAPI/analytics" \
--region="RegionOne"
```

To run the script, execute the following command:

```
# ./mn_endpoint_config.sh
mnControllerIp:mnControllerPort
analyticsIp:analyticsPort
```

To get the port of analytics node, execute the following command:

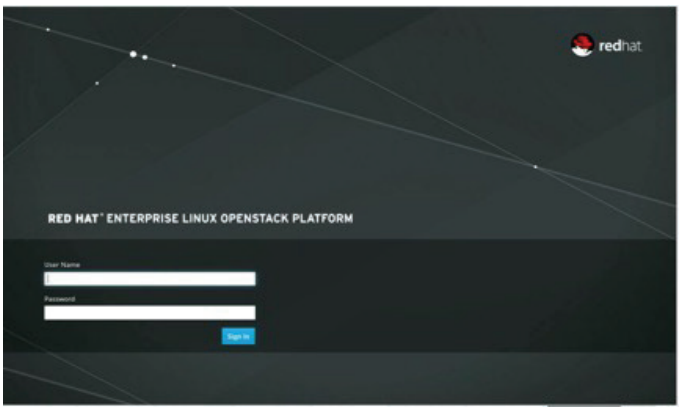
```
# mn-conf get calliope.service.ws_port
```

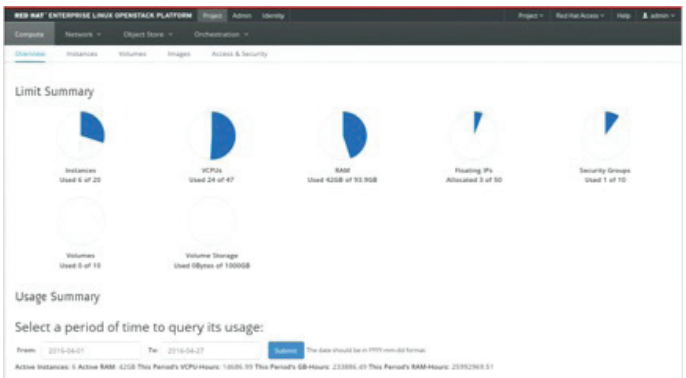
To get the port of your MidoNet controller, execute the following command:

```
# mn-conf get cluster.rest_api.http_port
```

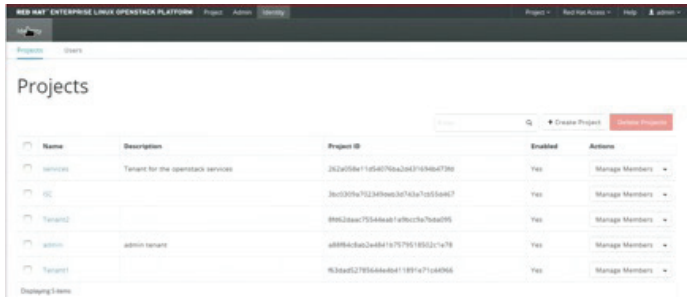
3.7 OpenStack Deployment—Create Tenants

1. Log in to the OpenStack dashboard.

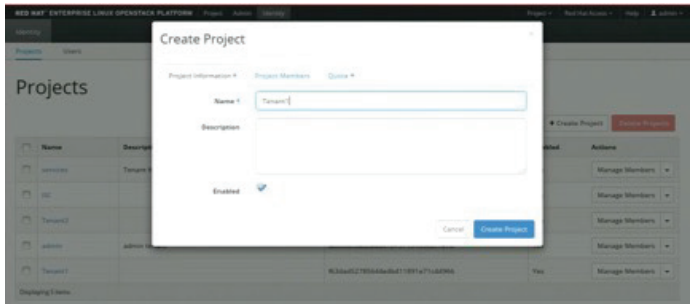




2. Select Identity Section.

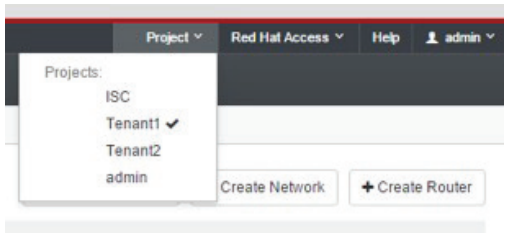


3. Click `Create Project`, and then in the pop-up window, click the `Create Project` button.

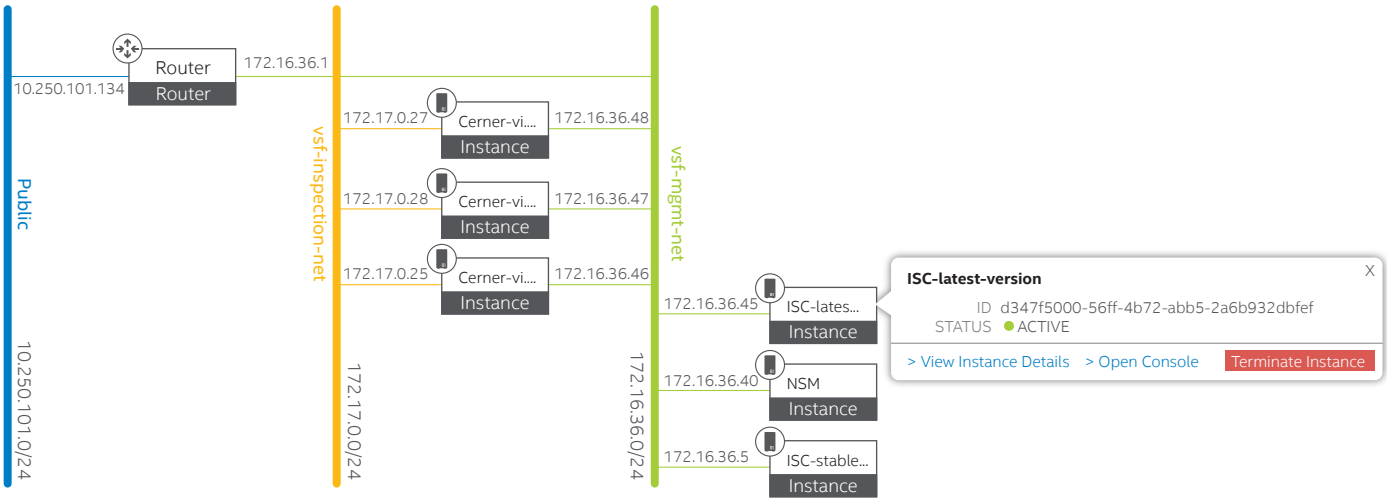


There should be created three projects (tenants) in total:

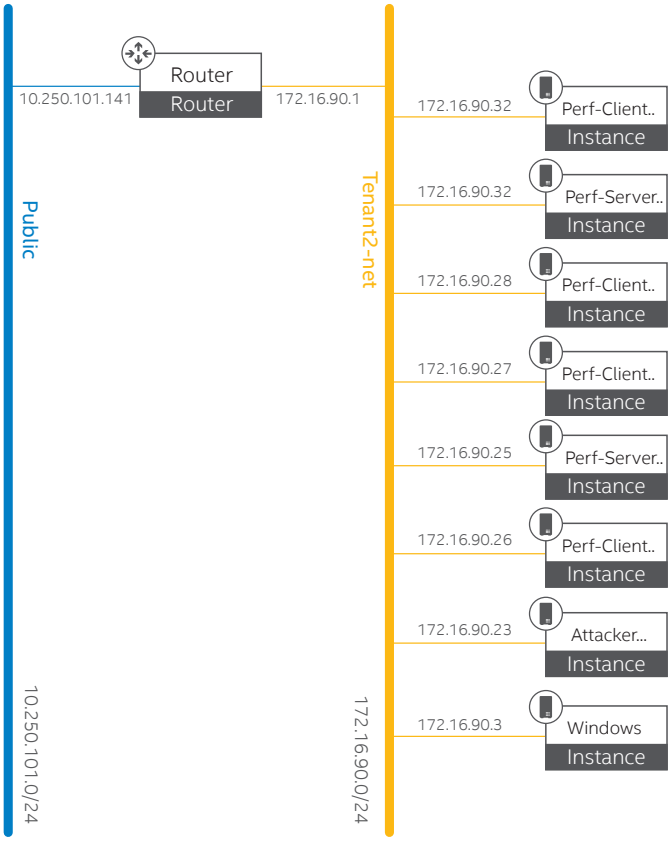
- For security
- For client VM (attackers), called Tenant 2
- For Web-Servers, called Tenant 1



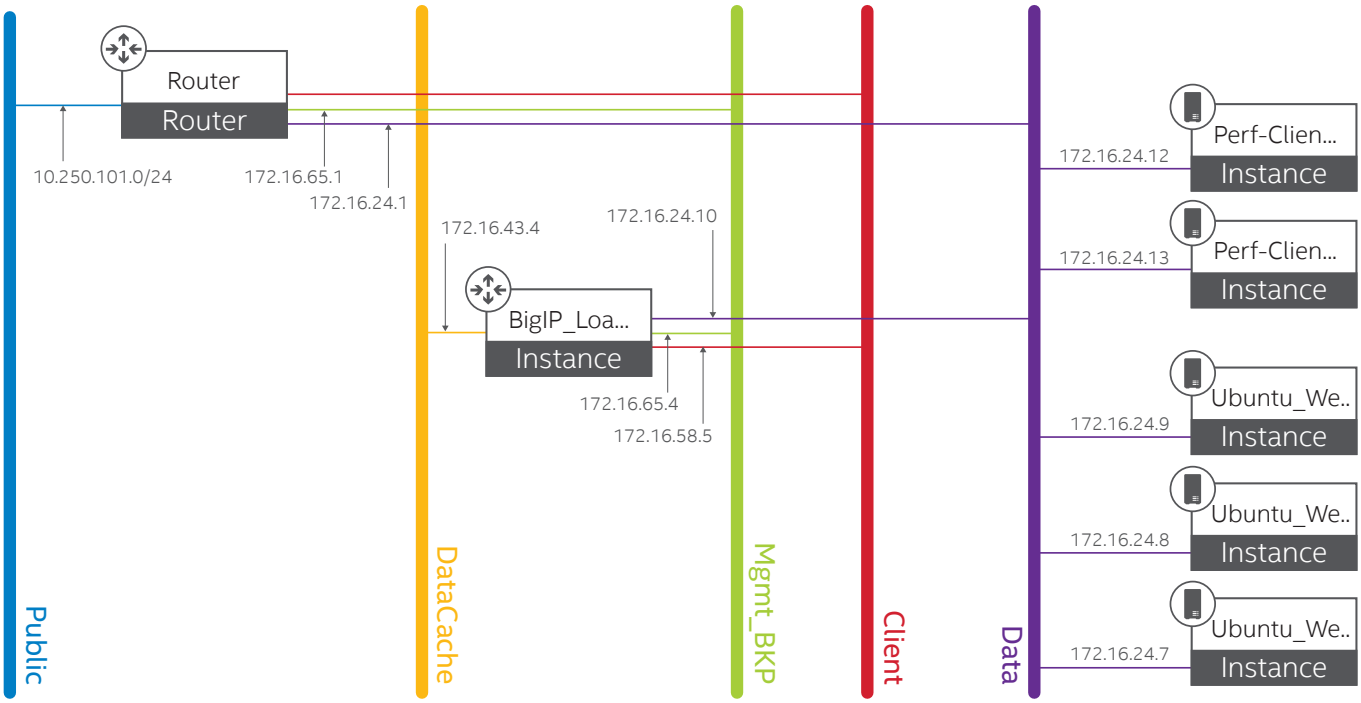
Security Tenant should look similar to the following figure:



Tenant 2:



Tenant 1:



3.8 Creating and Testing a Web Server

This section describes how to create and test the web server that supports a variety of features, many of which are implemented as compiled modules that extend core functionality. To create and test a web server, perform the following steps.

1. Create a web server instance with DataCache and the Management network.
2. In the Fedora* terminal of the Web server, run the following in the order shown:

```
# sudo –i
# ifconfig
```

   Make a note of the interface private IP address.

3. Edit the `/etc/httpd/conf/httpd.conf file`. Change the IP address for <listen> to have the machine's eth2 IP. Start the HTTP service.

```
# service httpd start
```

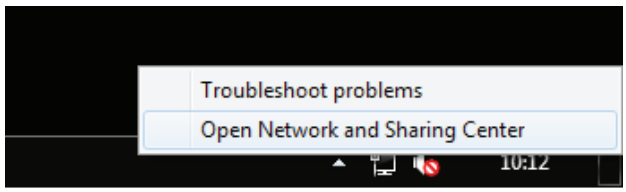
4. Edit the `/etc/ssh/sshd_config file`. Uncomment `PasswordAuthentication = yes`, (leave the rest as it is), `PermitRootLogin=yes [Password to ssh:Account6All]`. Start the SSH service.

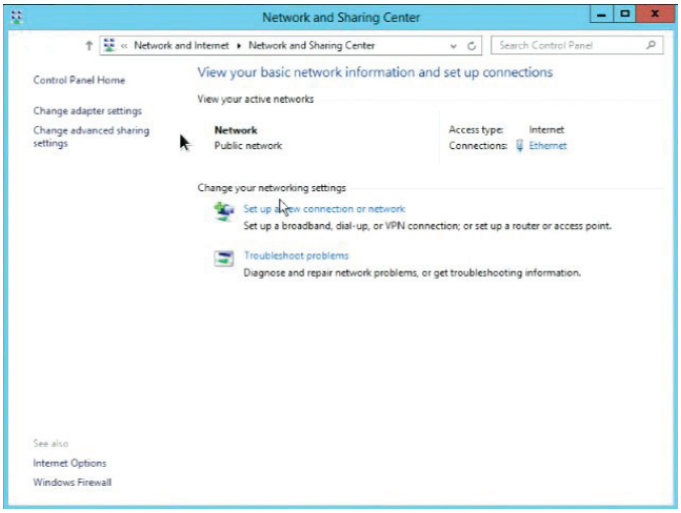
```
# service sshd start
```

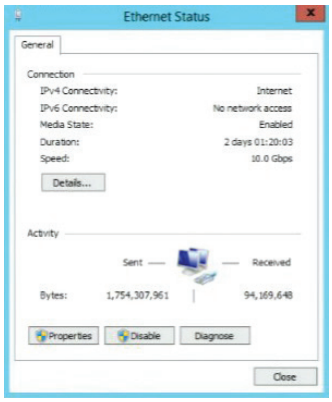
5. Test whether the web server page opens up on the browser for the web server instance.
6. Allocate and assign a floating IP address for the private instance of the web server.
7. Check whether the web server is functioning from the external browser by pinging a floating IP.
8. Repeat steps 1 through 7 to create another web server.

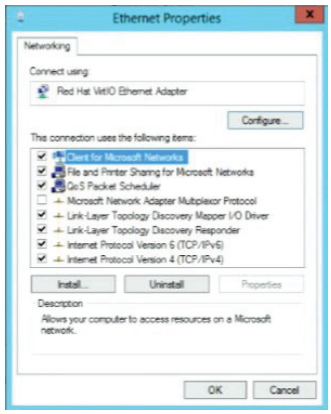
3.9 Windows* VM Configuration

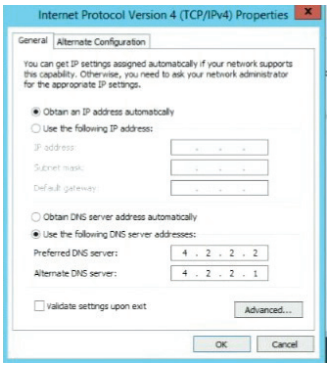
1. Set the Domain Name System (DNS) servers.



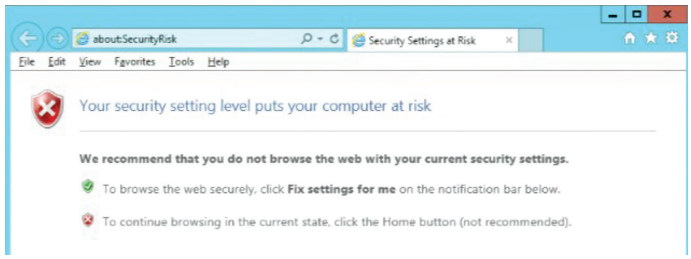


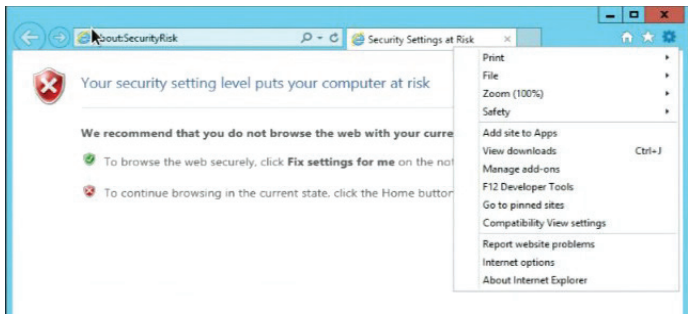


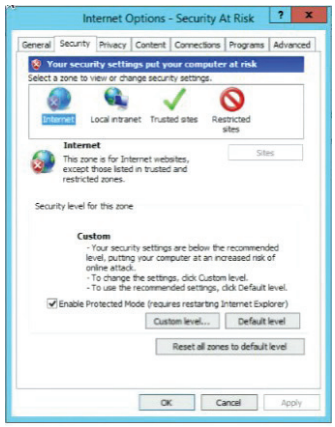


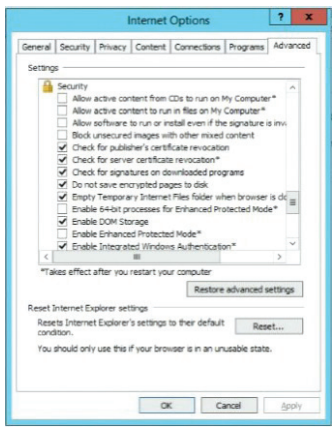


2. Disable the security for Internet Explorer*.



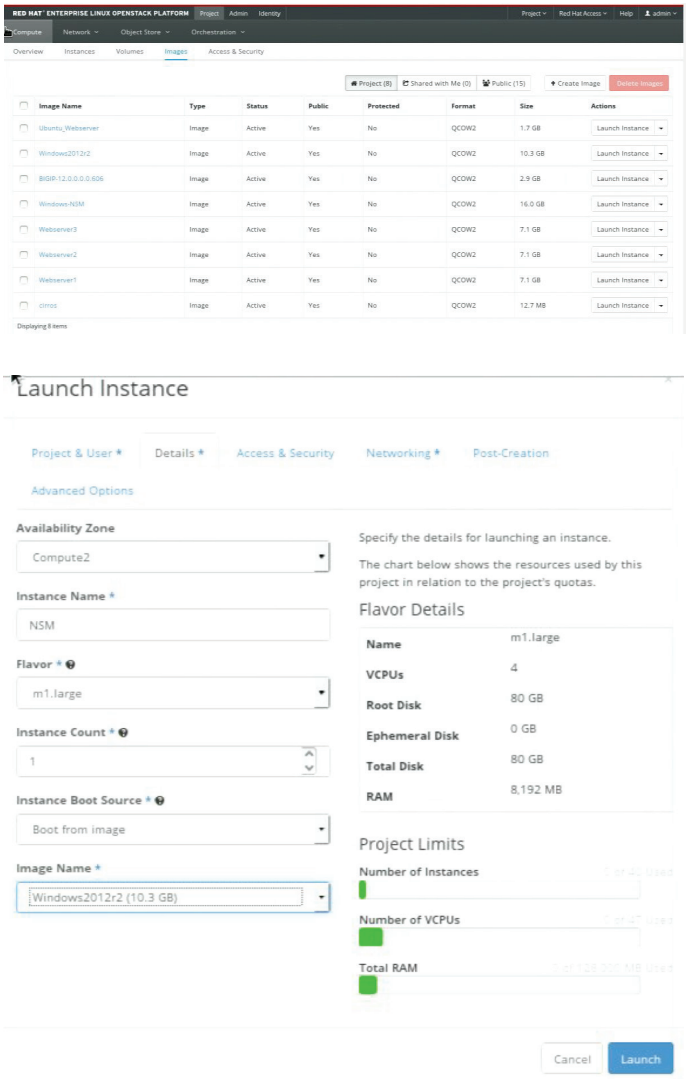




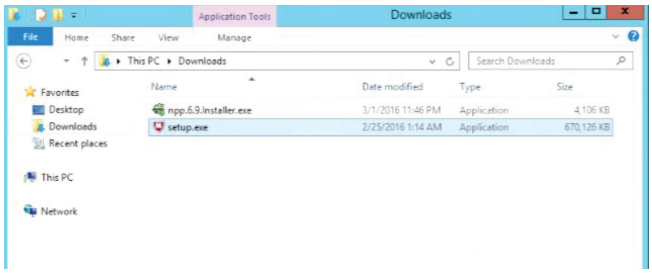


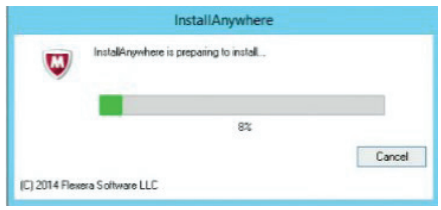
3.10 Network Security Manager Installation

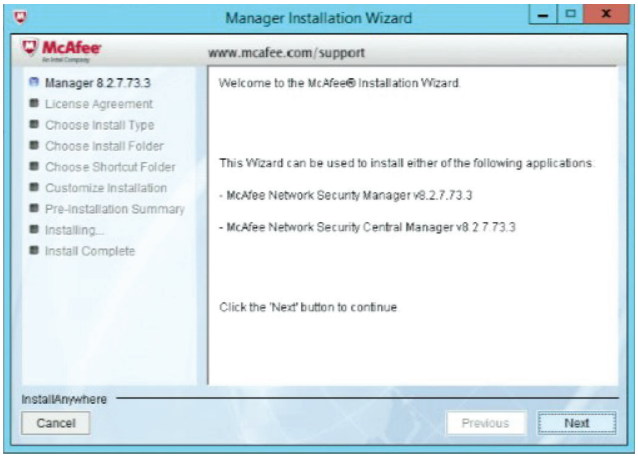
1. Launch a Microsoft Windows Server* 2012 r2 instance on the Security Tenant.



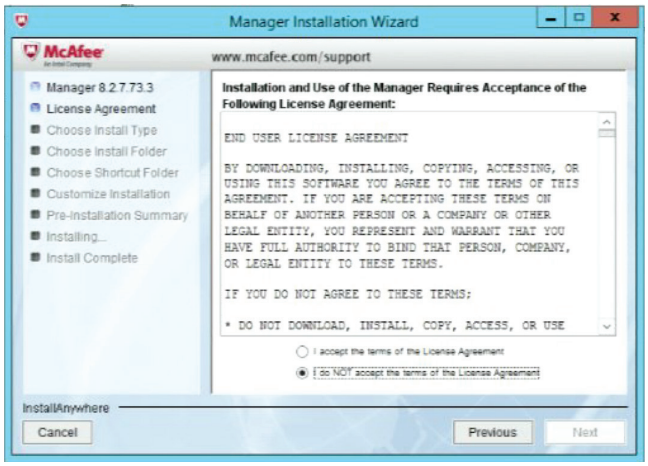
2. Copy the McAfee Network Security Manager executable inside the Windows VM, and run the file; in the wizard, click `Next`.



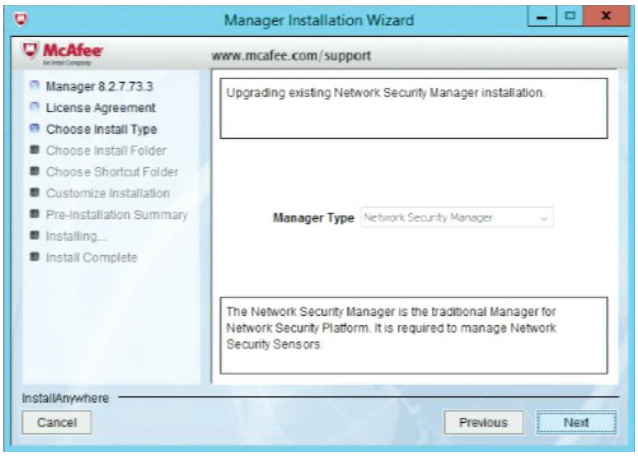




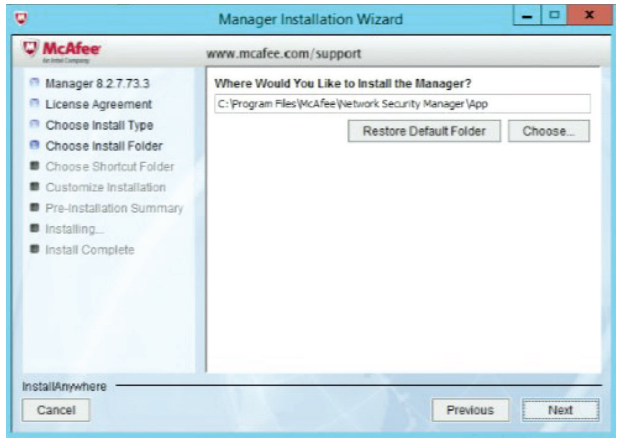
3. Accept the terms and conditions, and then click `Next`.



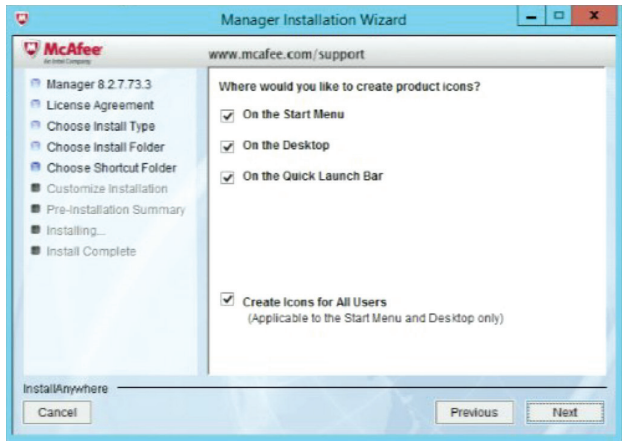
4. Select the Manager type: `Network Security Manager`, and then click `Next`.



5. Select the folder location, and click `Next`.

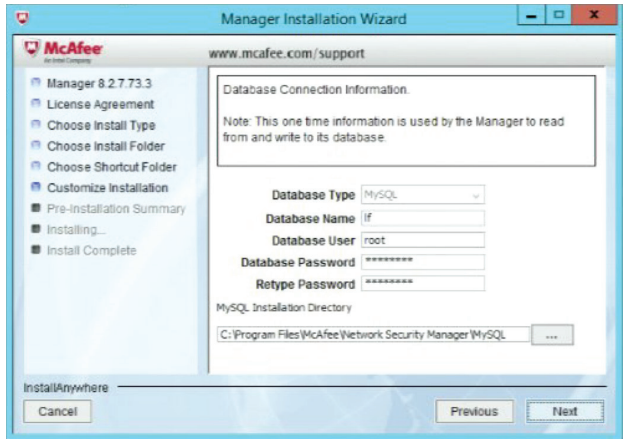


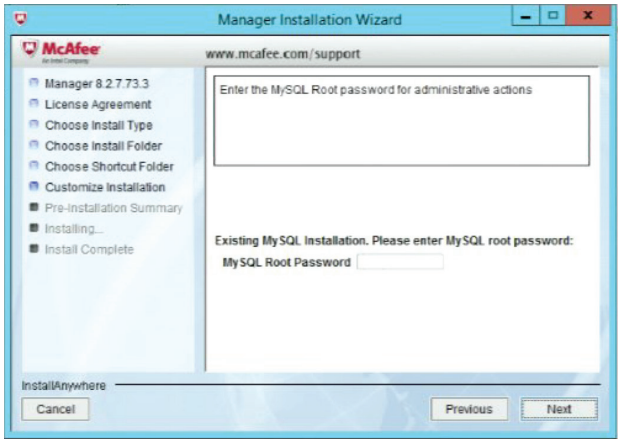
6. Select where NSM will create product icons.



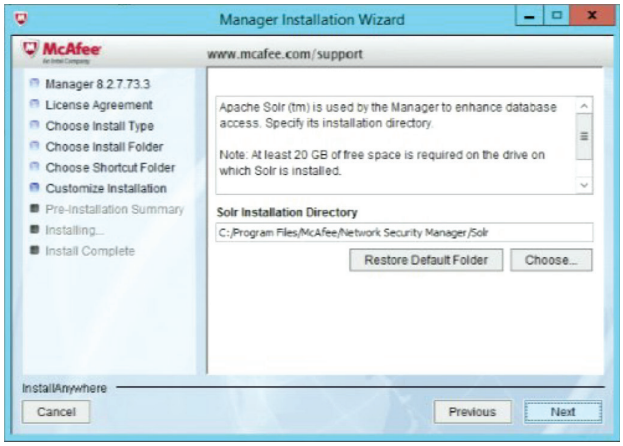
7. Select the type of database and credentials and click `Next`.

8. Set a password for the root user and click `Next`.

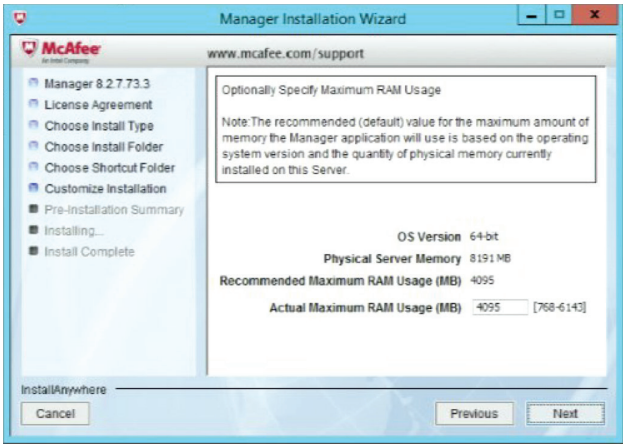




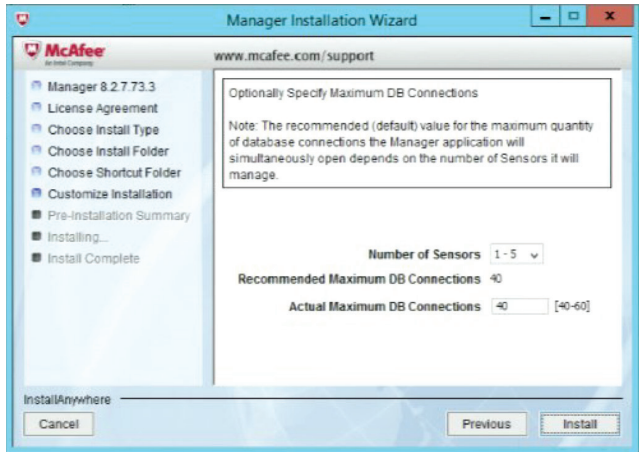
9. Select the folder for Apache Solr* installation, and then click `Next`.



10. Set the size of RAM memory to be used by McAfee Network Security Manager.



11. Set the amount of concurrent connection.



12. Click `Install`. A summary will be presented. Continue with the installation.
13. For the McAfee Network Security Manager 8.3 installation, uninstall the 8.2 version, and execute the fresh installation.
14. After you have finished, open a command prompt and log in to MySQL.

```
../Mcafee/Network Security Manager/MySQL/bin>mysql –uroot –padmin123
../Mcafee/Network Security Manager/MySQL/bin>use lf;
../Mcafee/Network Security Manager/MySQL/bin>ALTER TABLE iv_alert ADD COLUMN sourceVMIP CHAR(32) DEFAULT NULL;
../Mcafee/Network Security Manager/MySQL/bin>ALTER TABLE iv_alert ADD COLUMN targetVMIP CHAR(32) DEFAULT NULL;
```

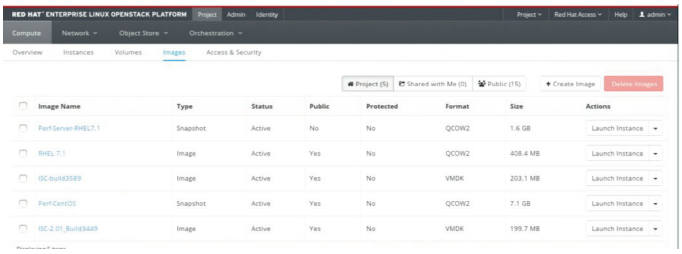
15. Go to the `../Mcafee/Network Security Manager/App/config/` directory and open the ems.properties file.
16. Turn off the AKKC settings.

```
iv.core.akka.enableakka=0
```

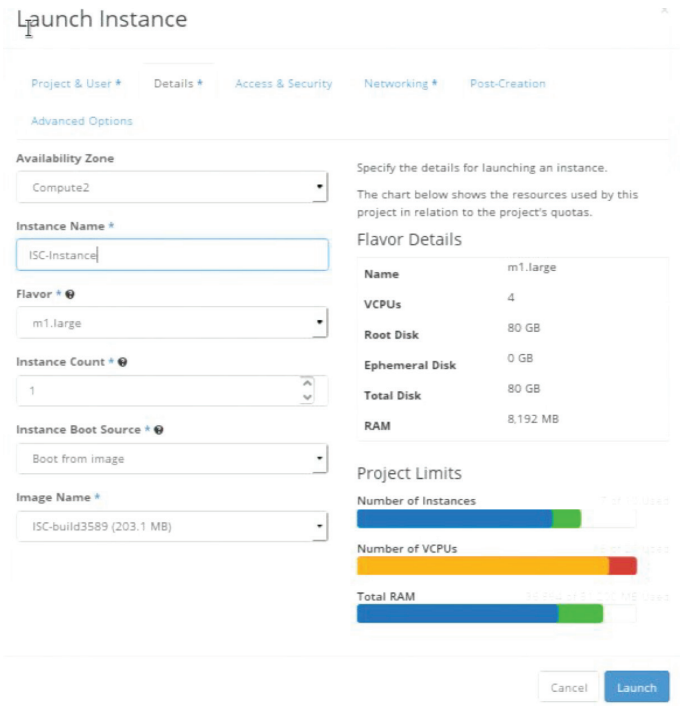
17. Restart the McAfee Network Security Manager VM.

3.11 Open Security Controller (OSC) Installation

1. Add the image in the OpenStack setup (in case the OSC will run inside of the same setup).

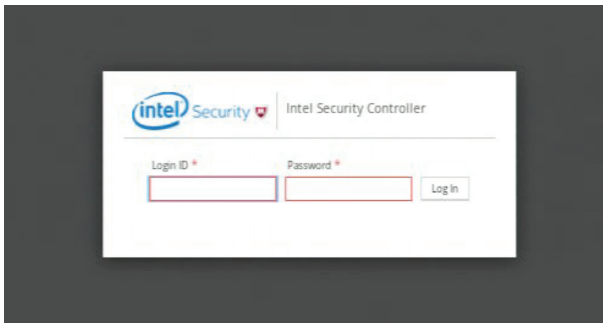


2. Launch the OSC instance from the previous image in the OpenStack Image Service*.

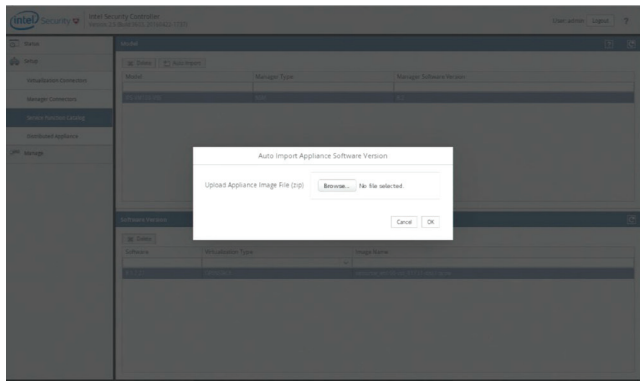


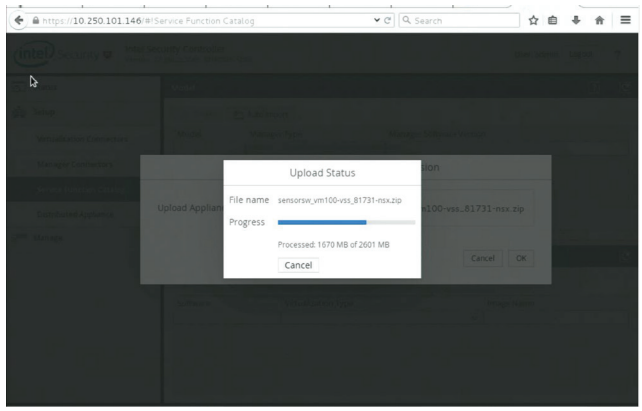
3. Assign a floating IP address to the OSC instance.

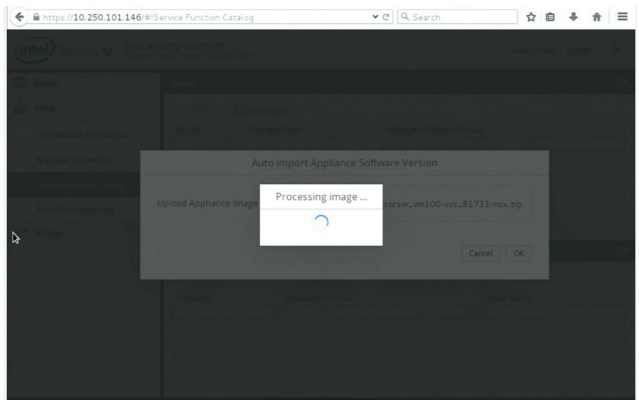
4. Access the OSC dashboard using the floating IP: https://<OSC-floatingIP>/ Default credentials: `admin/admin123`.

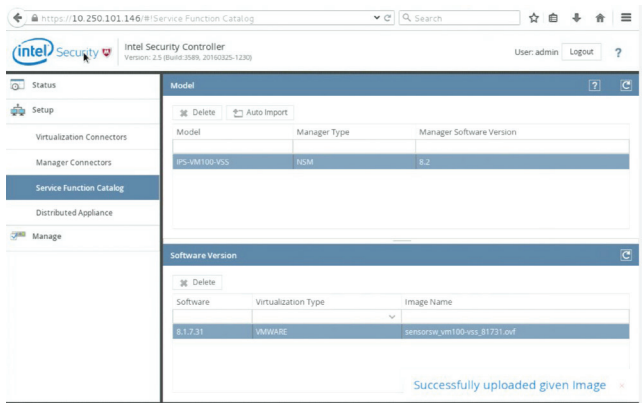


5. Add a sensor image in the OSC deployment.
   - In the menu, select `Setup/Service Function Catalog`
   - Click `Auto Import`.
   - Browse the sensor zip file, and then click `OK`.



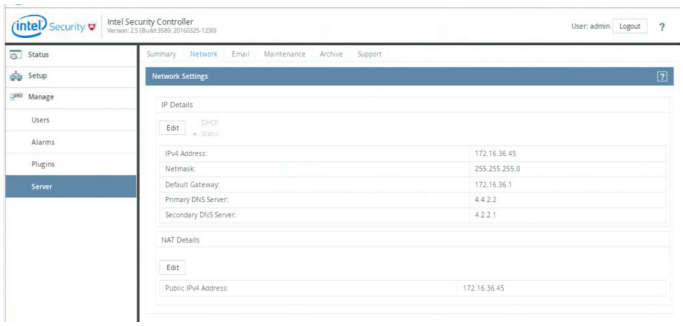






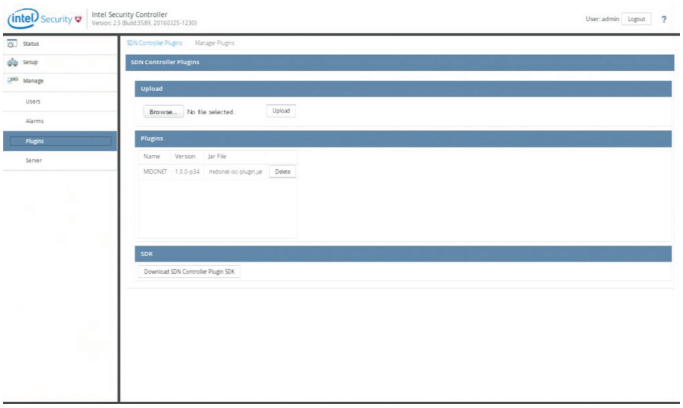
6. Set the DNS servers to OSC setup (this is required for the Network Time Protocol configuration).

- Select `Manage/Server/Network`.
- Select static `IP Details`, and then add the DNS servers.



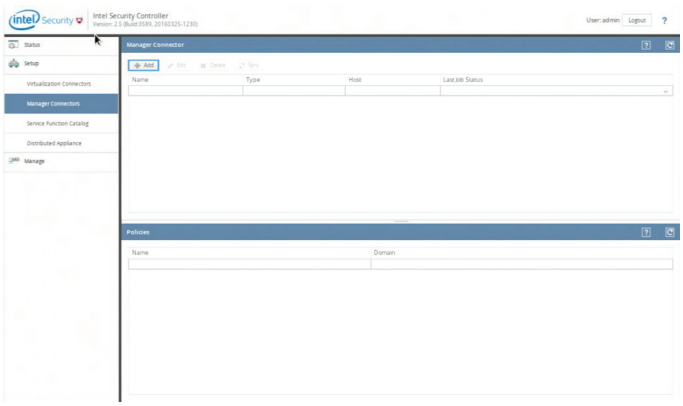
7. Add the SDN Controller.

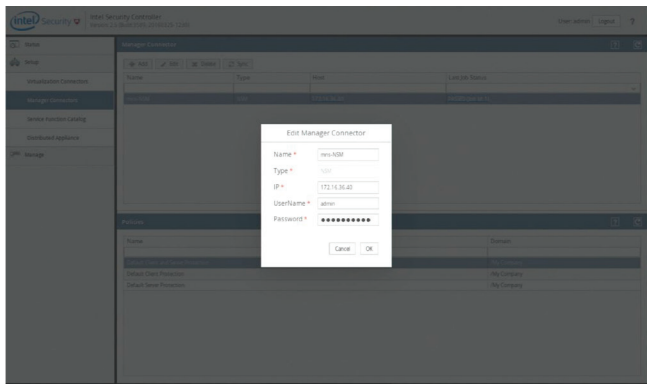
- In the menu, select `Manage/Plugin`.
- Browse for the zip file.
- Click `Upload`.

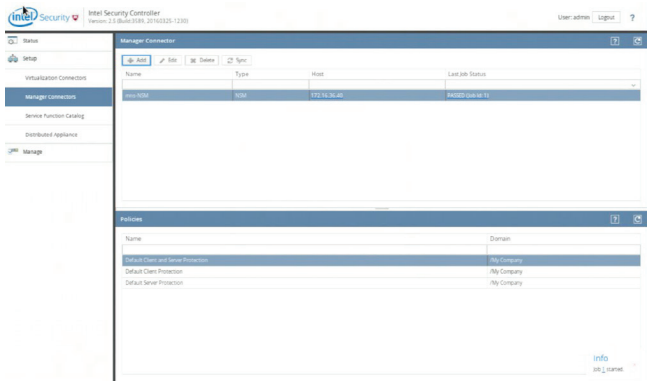


8. Add a manager connector.

- In the menu, select `Setup/Manager Connectors`.
- Click `Add`.
- Set the name of the manager connector.
- Set the IP address of the Network Security Manager.
- Set the credentials to connect to the Network Security Manager.

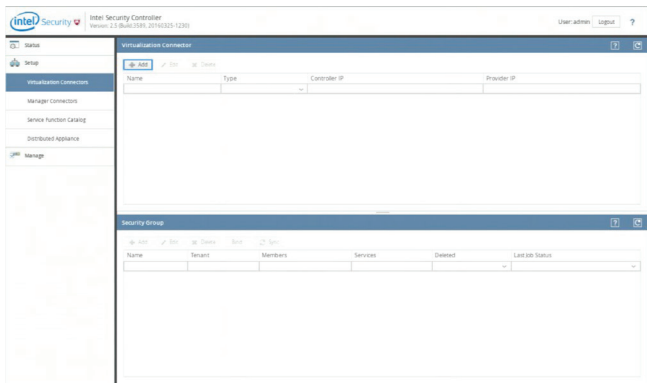


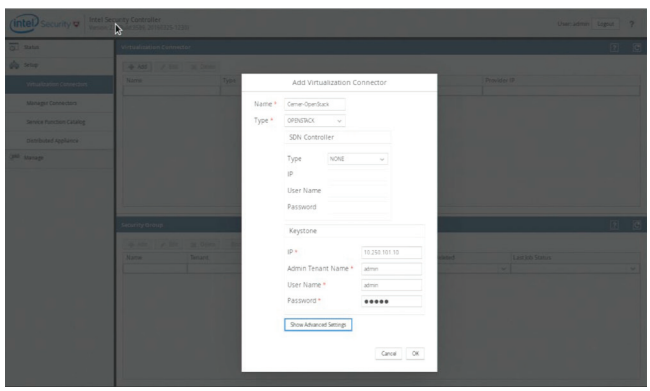


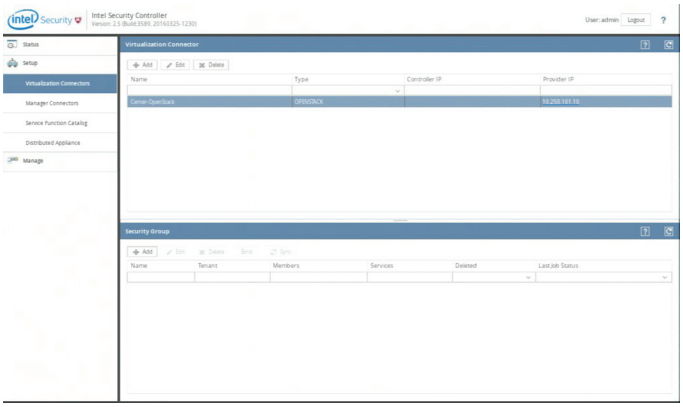


9. Add the Virtualization Connector.

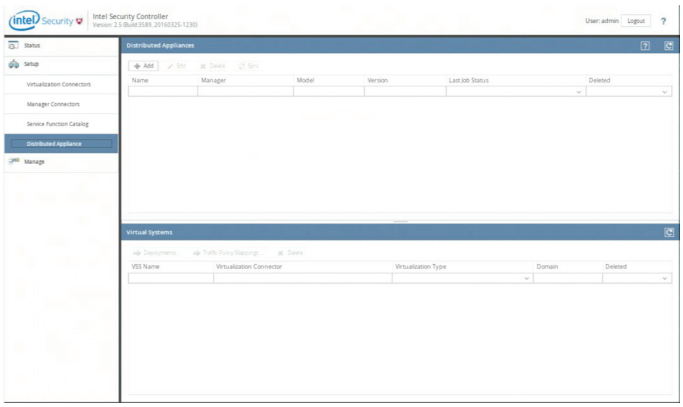
- In the menu, select `Setup/Virtualization Connector`
- Click `Add`.
- Set a name to the virtualization connector.
- Select type: OpenStack.
- Select SDN Controller: Midonet.
- Add OpenStack Identity credentials.

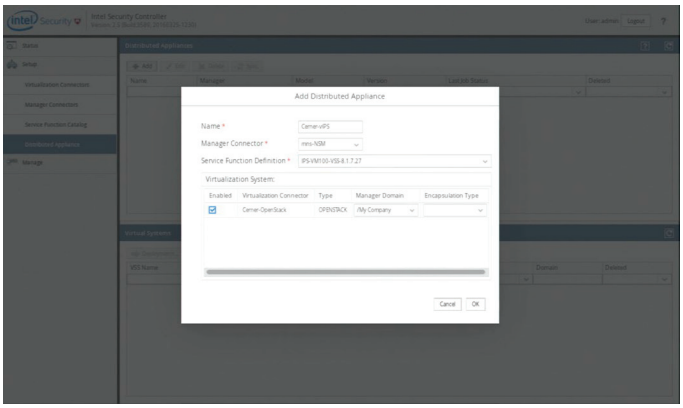




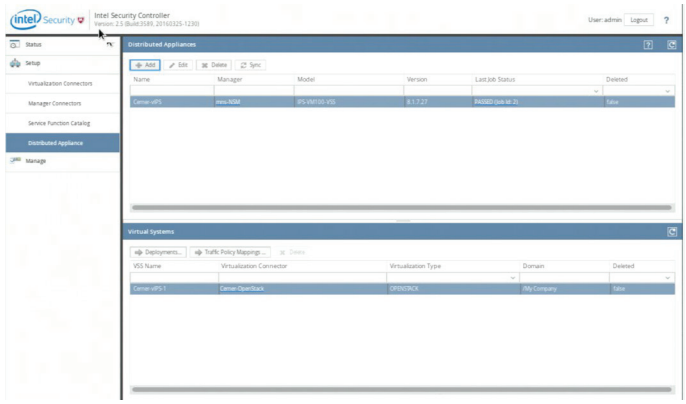


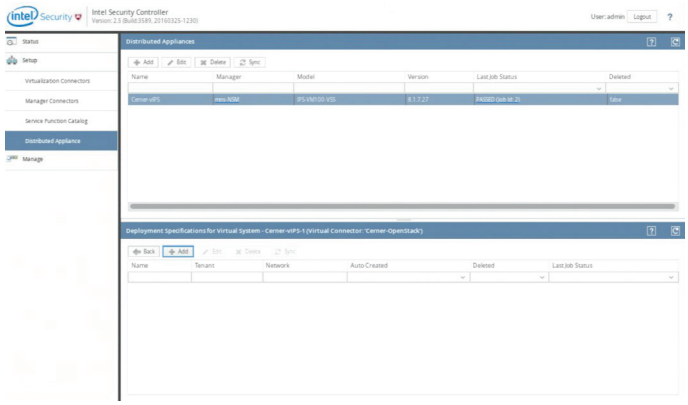
10. Deploy the distributed appliances.
- Select `Setup/Distributed Appliance`.
- Click `Add`.
- Set a name for the distributed appliance.
- Select the `Manager Connector` (the Network Security Manager setup).
- Select the `Service Function Definition`.

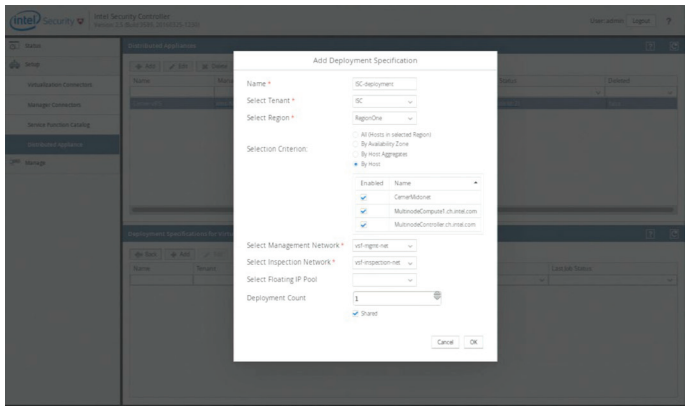


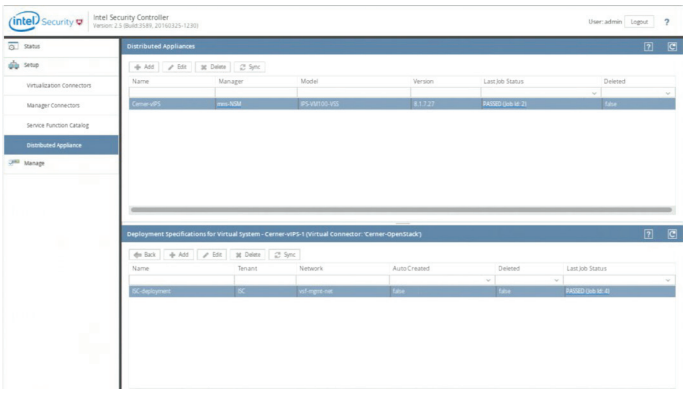


11. Deploy the vIPS appliances.
- In the menu, select `Setup/Distributed Appliance`.
- Click `Deployment`.
- Click `Add`.
- Set the name for the deployment specification.
- Select the tenant for the security deployment.
- Select the region (that is, RegionOne).
- Select the hosts that should launch vIPS appliances.
- Select the management network.
- Select the inspection network.
- Set the amount of appliances per host.
- Click `OK`.



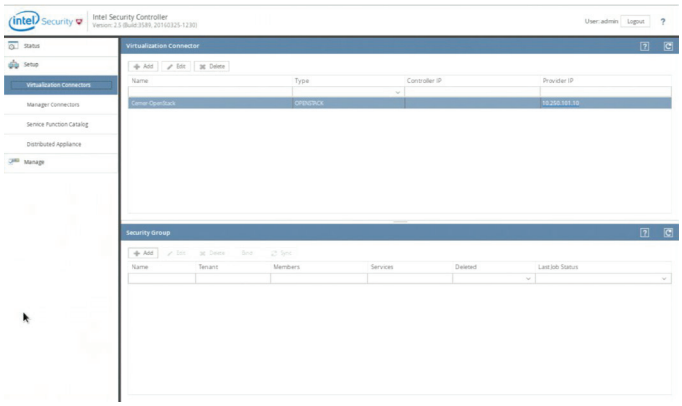


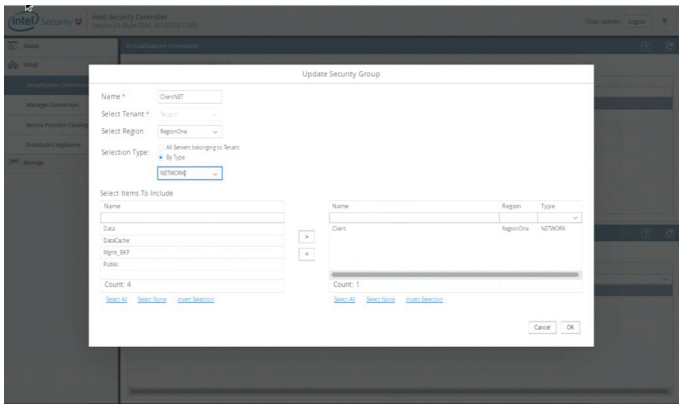


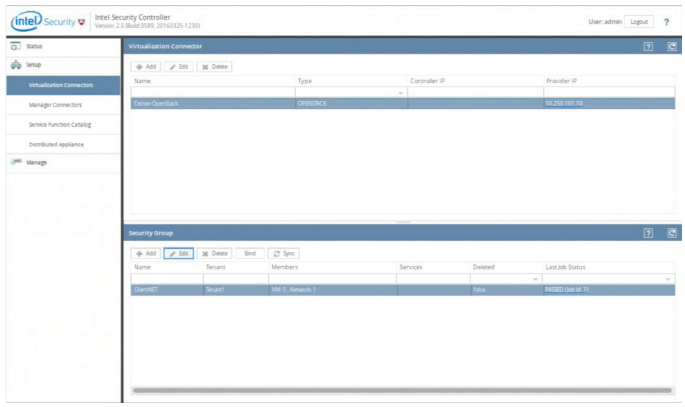


12. Create the security group.

- In the menu, click `Setup/Virtualization Connector`.
- Click `Add`.
- Set a name for the security group.
- Select tenant.
- Select region.
- Select type of security: `Network/Sub-Network/VM`
- Add the instance into the security group.
- Click `OK`.

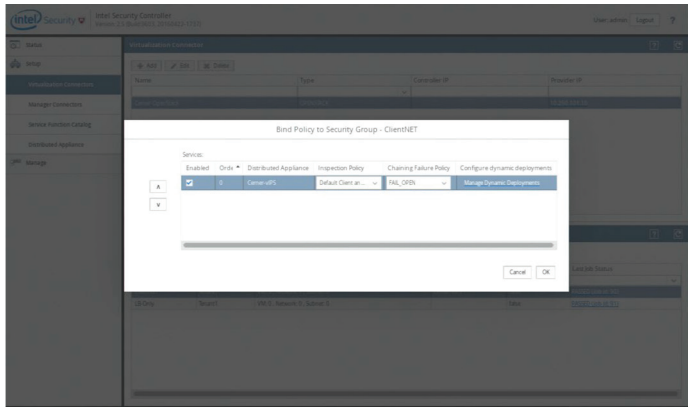


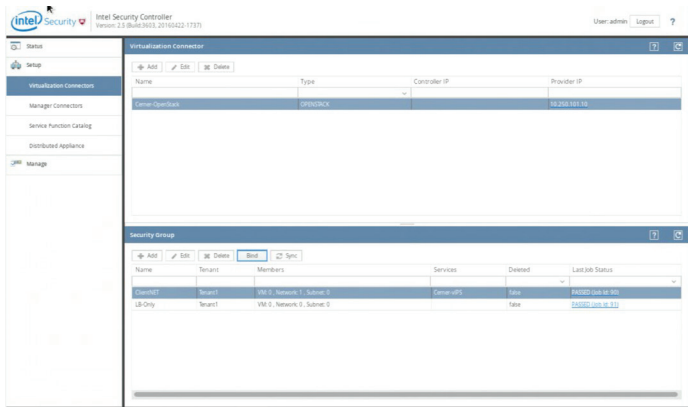




13. Bind a security Group.

- In the menu, select `Setup/Virtualization Connections`.
- Select the security group.
- Click `Bind`.
- Check the `Enabled` box.
- Click `OK`.





3.12 F5 Load Balancer Installation

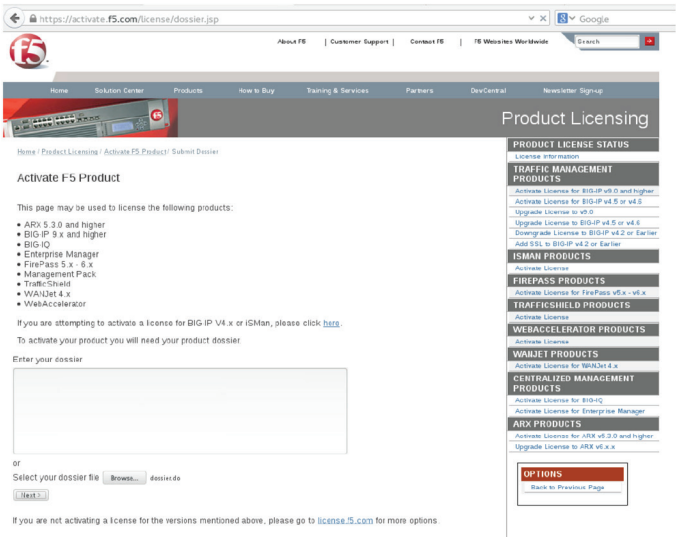
This section describes how to create and configure an F5 BIG-IP* instance. The F5 BIG-IP performs a variety of functions that drive application availability, optimization, and security. To create a F5 BIG-IP instance, perform the following steps.

1. Copy `release.fedora, release.redhat`.
2. On the terminal (on both compute nodes), run the following commands:

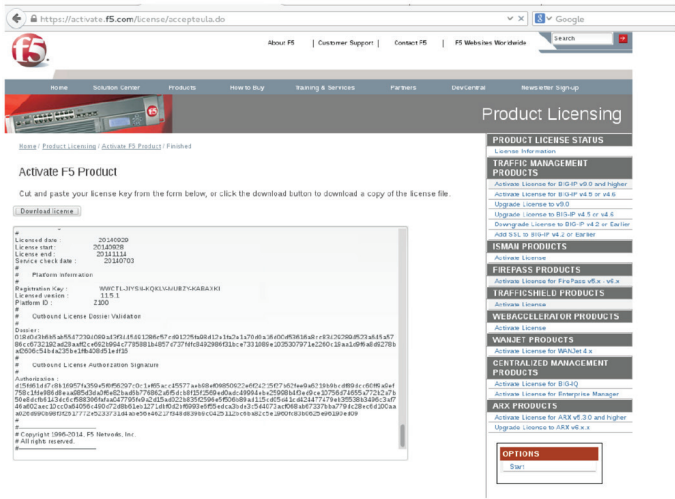
```
# cd /etc/nova
# cp release release.fedora
# cp /home/../release.redhat release
# service openstack-nova-compute restart
```

3. Launch a F5 BIG-IP instance with the networks in the following order:
   - Management
   - DataCache
   - Client
   - Data
4. Make sure there is no error in the instance. Log in: `root`, password: `default`.
5. Get the management address of the instance from OpenStack.
6. On the browser https://<managementIP>/tmu/login.jsp, log in: `admin`, password: `admin`.
7. Download the `dossier.do` file, and then save it locally.
8. Activate the license: https:/activate.f5.com.
9. Select the appropriate “Activate” option for the version of F5 BIG-IP you are using.
10. If activating behind a proxy, manually add a dossier file as follows:

- Select the dossier file downloaded in step 7 to activate, and then click `Next`.



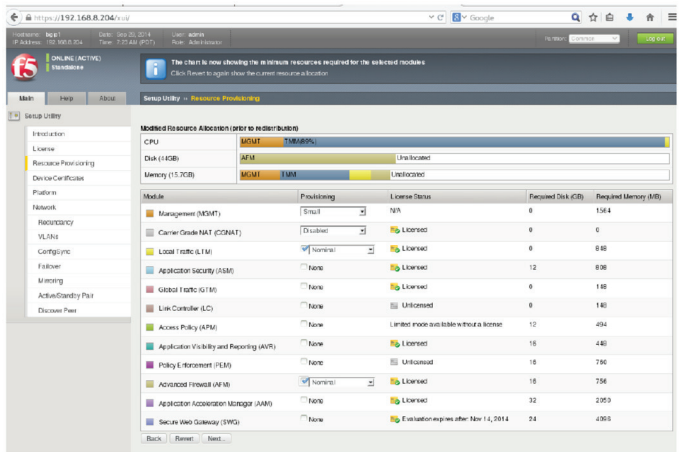
- Download the `license.txt` file.



- Upload the `license.txt` file on the https://<managementIP> portal under the License tab.

11. If not activating behind a proxy, upload the license file directly via the Internet.
12. After activating, do the following in the F5 BIG-IP portal:

- From the `Setup Utility` navigation menu, select `Resource Provisioning`, and then check. `Nominal` from `Local Traffic (LTM)` and `Advanced Firewall (AFM)` drop-down menus.



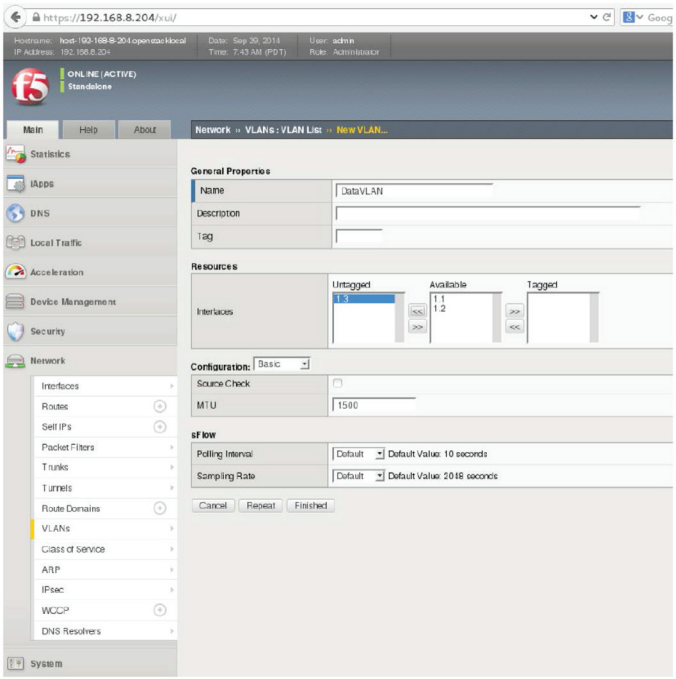
13. Click `Next` to reboot the system, and then configure the list items below.

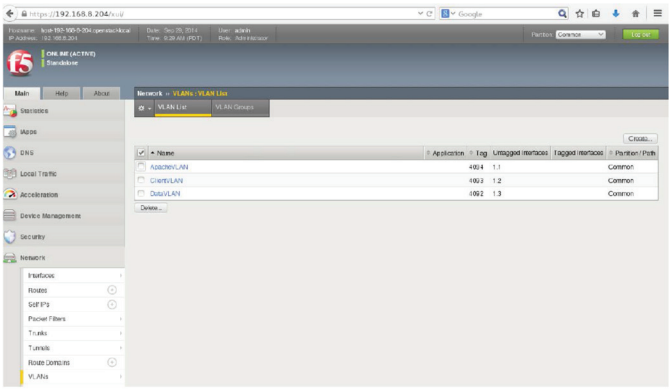
- Enter the hostname: xxx.xxx.com
- Root account: default
- Password: default
- Admin password
- Log in again with a new admin password
- SSH: Checked

14. In Advanced Options, click `Finished`.

15. From the Network navigation menu, select `VLANs`, and then create the following new VLANs as shown in the screenshots below:

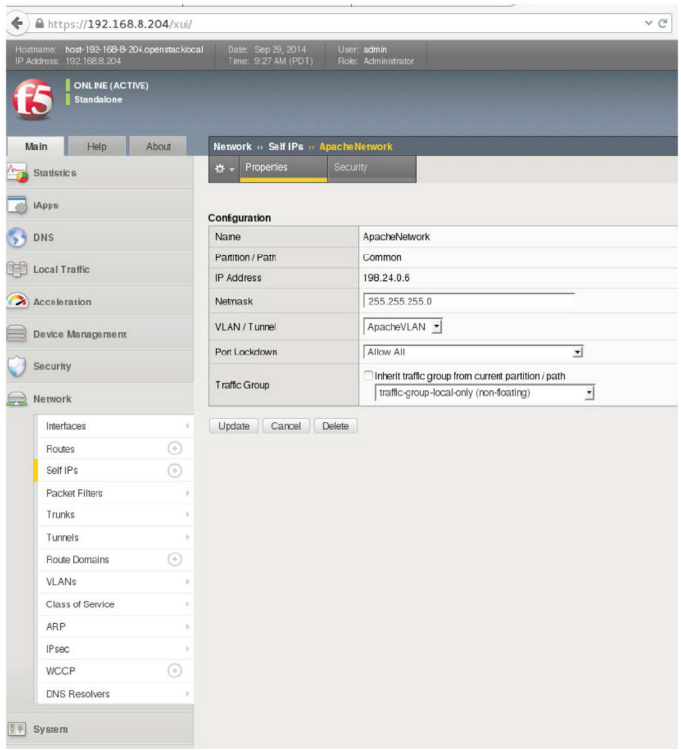
- Apache VLAN : : Interfaces Untagged: 1.1, Available: 1.2, 1.3
- Client VLAN : : Interfaces Untagged: 1.2, Available: 1.1, 1.3
- Data VLAN : : Interfaces Untagged: 1.3, Available: 1.1, 1.2

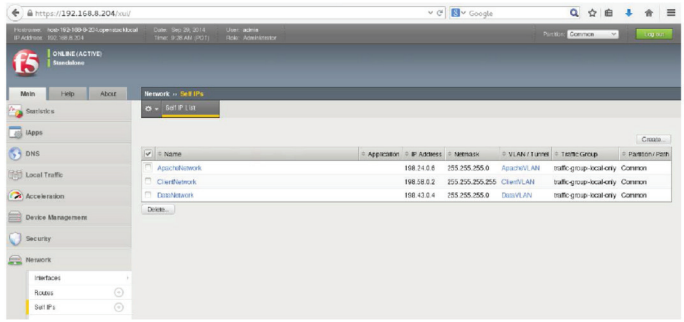




16. From the `Network` navigation menu, create and configure self IP addresses for the following networks.

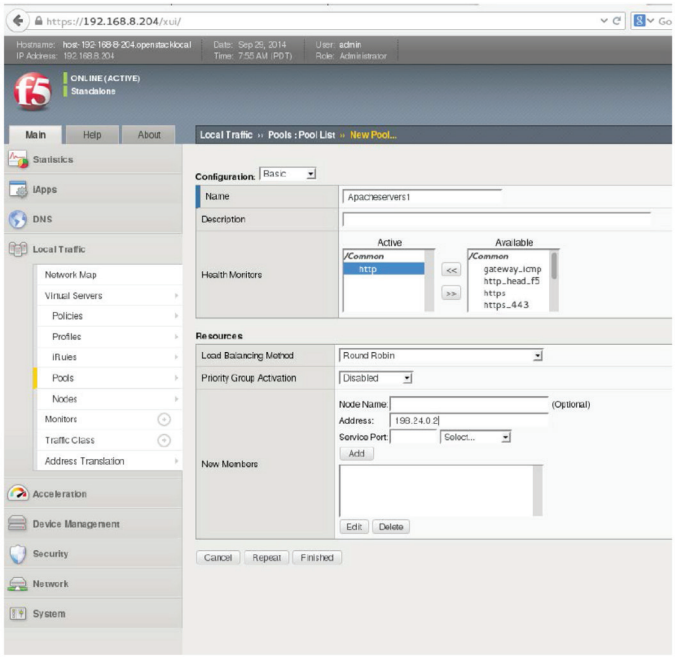
- ApacheNetwork
  - IP address: ApacheNetwork IP address of F5 BIG-IP instance from OpenStack
  - Netmask: `255.255.255.0`
  - Port Lockdown: `Allow All`
- ClientNetwork
  - IP address: ClientNetwork IP address of F5 BIG-IP instance from OpenStack
  - Netmask: `255.255.255.255`
  - Port Lockdown: `Allow All`
- DataNetwork
  - IP address: DataNetwork IP address of F5 BIG-IP instance from OpenStack.
  - Netmask: `255.255.255.0`
  - Port Lockdown: `Allow All`





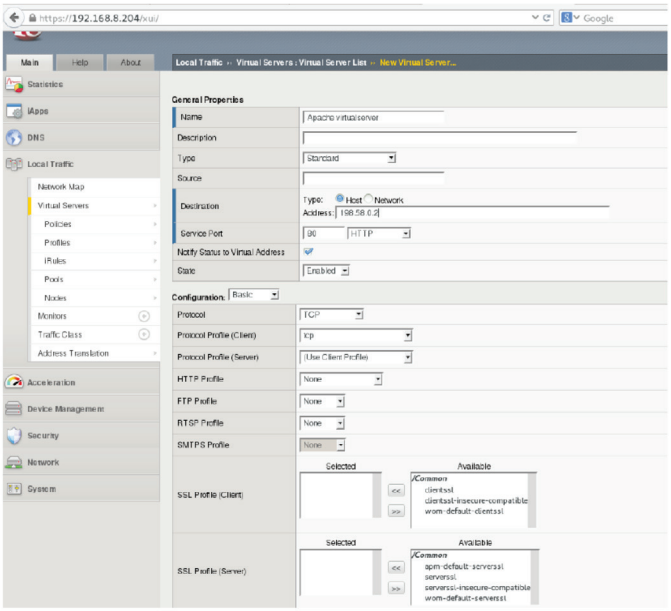
17. From the `Local Traffic` navigation menu, select Pools, and then configure as follows.

- Name (ServerPool Nodename): `Apacheservers1`
- Health Monitors: `http`
- Address: Get IP address from the OpenStack.
- New Members: Servers 1 and 2.



18. Repeat step 16 for Apache Server 2.

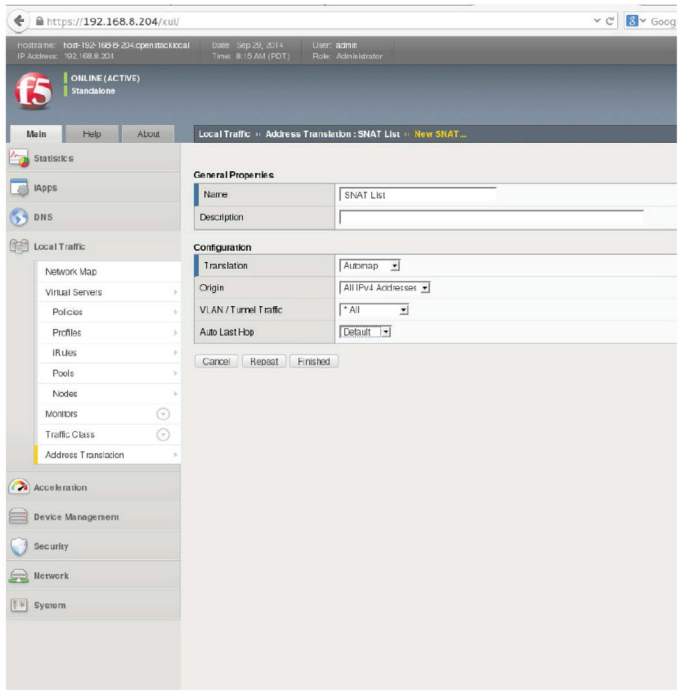
19. From the `Local Traffic` navigation menu, select `Virtual Servers`. In the `Destination` field, enter the Client Network IP address of the F5 BIG-IP instance from OpenStack.



20. From the Local Traffic navigation menu, select `Address Translation` and configure as follows:

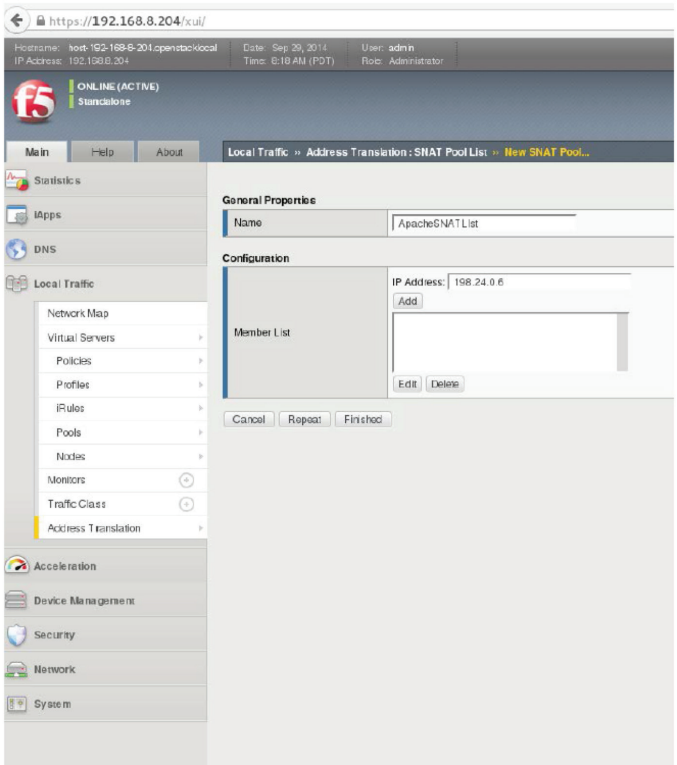
- Name: `SNAT List`
- Translation: `Automap`
- Origin: `All IPv4 Addresses`
- VLAN/Tunnel Traffic: `* All`
- Auto Last Hop: `Default`

21. Click `Finished`.



22. Create a SNAT pool list.

- Name: `Apache SNAT List`
- IP address: Apache IP address



23. Click `Finished`.

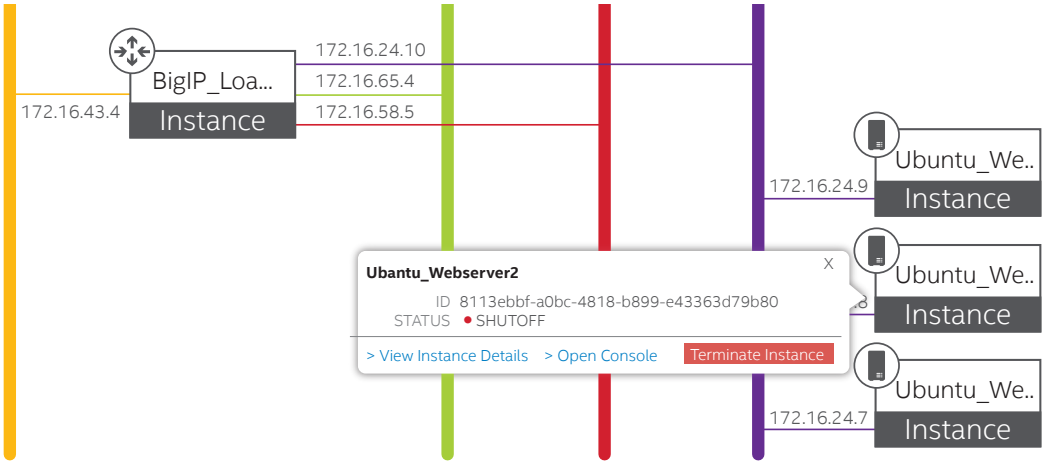
24. Assign a floating IP address to the client and data network of a F5 BIG-IP instance and a secondary management network to a Steelhead instance.
25. Check the web browser: http://<floatingIP_of_client_network_IP_of_BIG-IP_Instance>/Pictures
26. Copy back: `cp /etc/nova/release.fedora /etc/nova/release`.

The network topology after configuration is shown below.

4.0 Demo Setup: Cross-Tenant Cross-Machine Attack

Tenants in OpenStack are different projects that represent different customers of Cerner. The idea of the setup is to simulate two different customers inside the cloud, and perform the attack from one project to another.

The demo shows the interaction of VMs in a multi-tenant environment, where one VM represents the attacker (in Tenant 2) and the other VM is the destination of the attack (in Tenant 1). The MidoNet SDN controller redirects the traffic for inspection to the Security Tenant (Intel ISC Tenant), containing security functions, that is, OSC, McAfee Network Security Manager and vIPS. The vIPS performs analysis of the packets. If packets correspond to the malicious activity, these will be blocked; otherwise, packets will continue the normal data path.



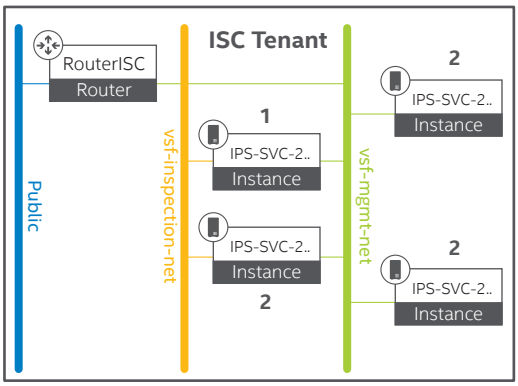
Demo Setup

Scenario #1

a) Protect Load Balancer only (on controller)
b) Access LB (with & without attack)
c) View results in Real Time Threat Analyzer

Scenario #2

a) Protect Web3 only (on compute)
b) Access LB (with & without attack)
c) When LB uses Web3, it is protected



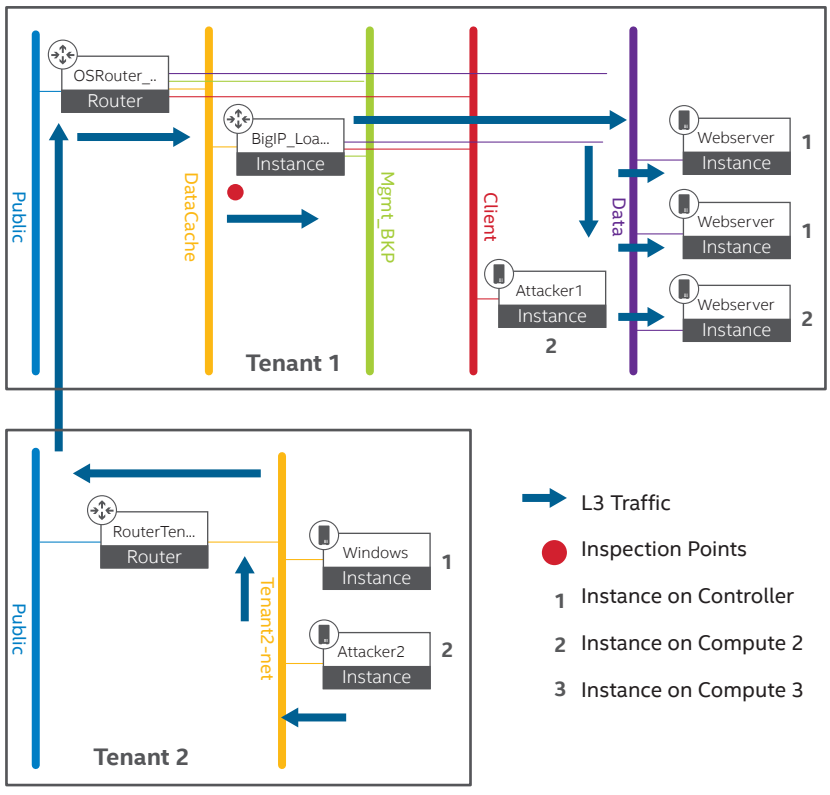
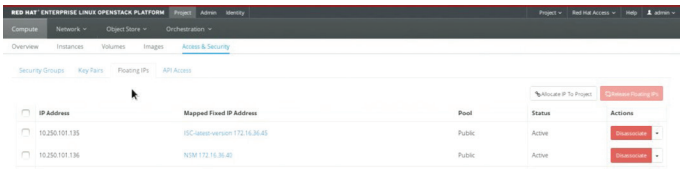


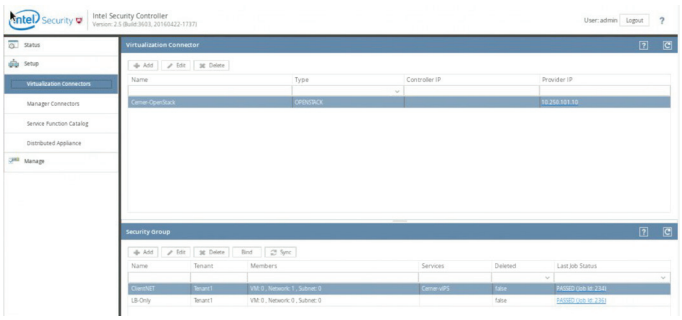
Figure 2. Demo: Cross-tenant cross-machine attack.

To set up the demo, perform the following steps.

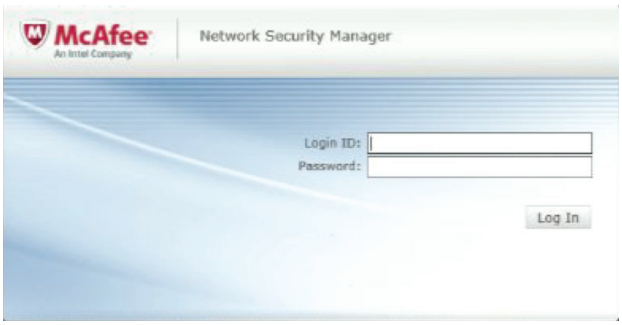
1. Set a floating IP address to the Network Security Manager Windows VM and OSC VM.

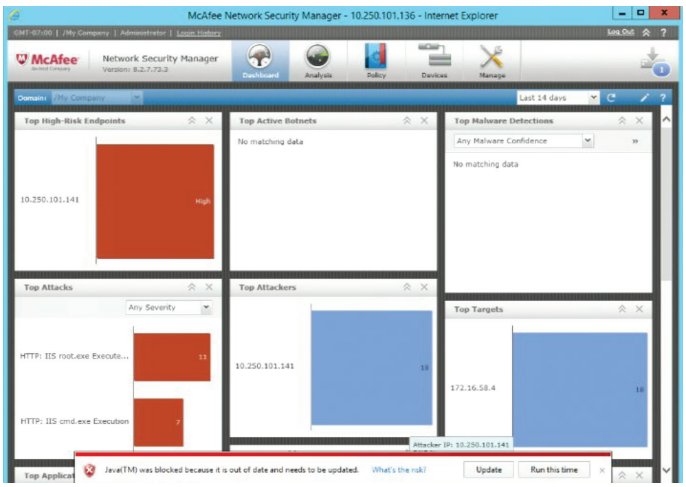


2. Using the floating IP address of the OSC, access the OSC dashboard and verify whether the network "client" from Tenant 1 is bind enabled.

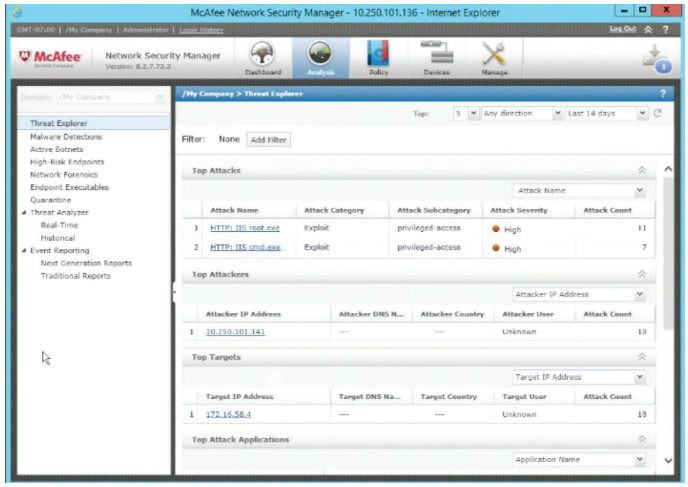


3. Access any Microsoft Windows machine in Tenant 2.
4. Access the dashboard of the Network Security Manager using the floating IP address of the Network Security Manager. Log in with the Network Security Manager credentials.

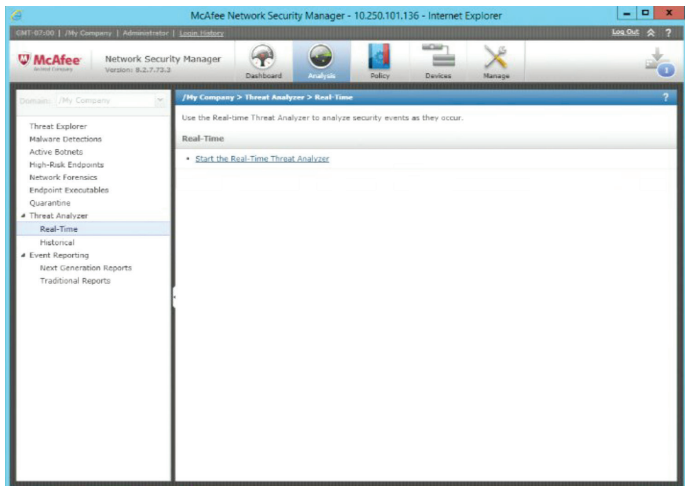


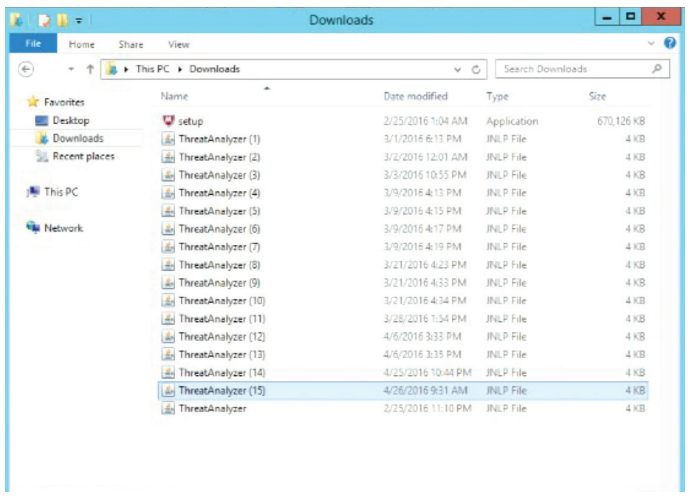


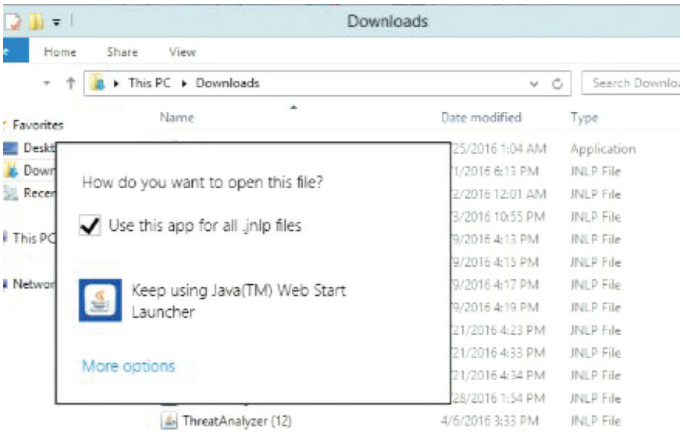
5. Click `Analysis`.



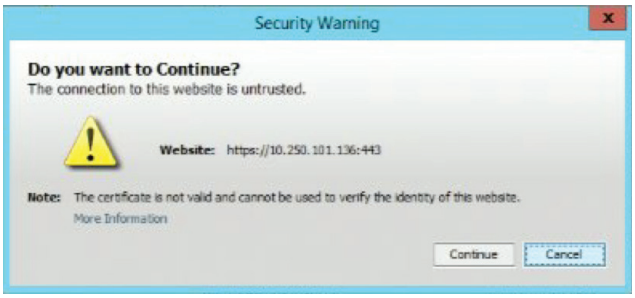
6. In the Thread Analyzer menu, select `Real-Time`.
7. Download the JNLP file to start the Real-Time Threat Analyzer.
   - Click in the link `Start the Real-Time Threat Analyzer`.
   - Go to downloads and start the file with JAVA Web Start Launcher.



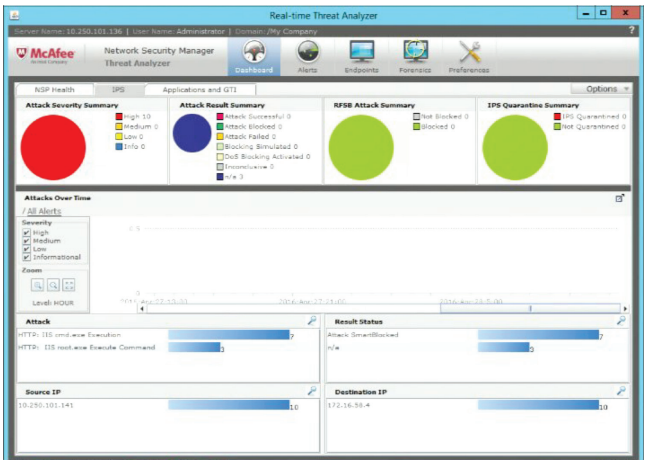


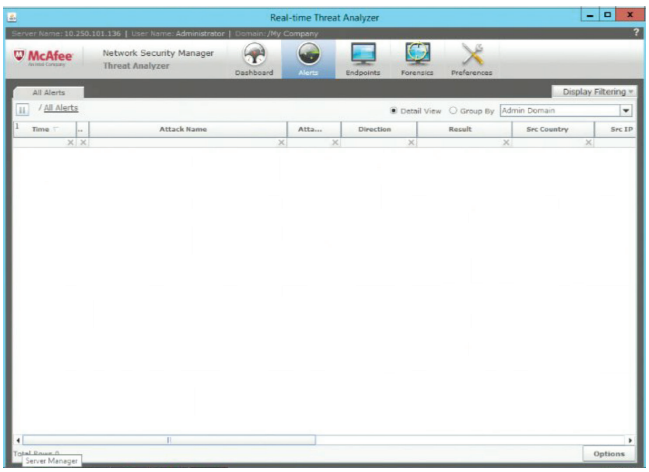


• In the Security Warning dialog, click `Continue`.



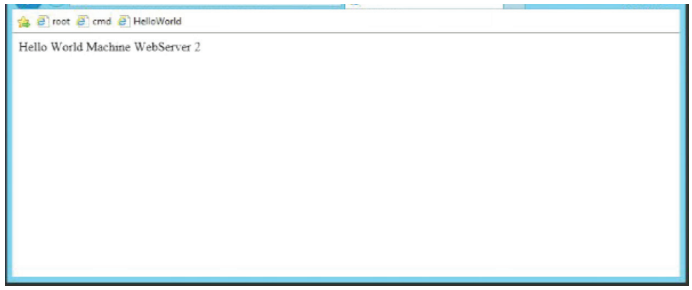
8. Click `Alert` to see the real-time attacks.





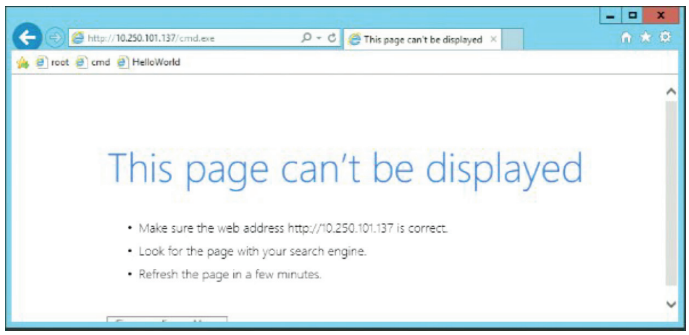
9. Open Internet Explorer in any Microsoft Windows machine on Tenant 2.

10. Navigate to the web servers passing the floating IP address of the load balancer, which is connected to the client network (the network that is bind enabled in OSC). To navigate, use: http://10.250.101.137/HelloWorld.php.

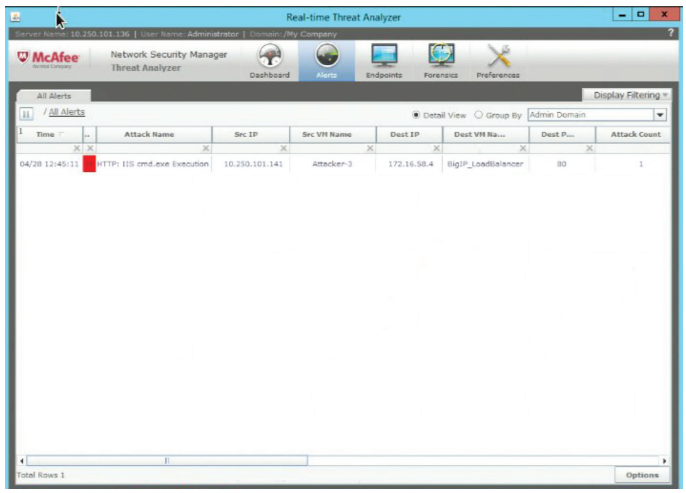


11. Execute and attack requesting `cmd.exe` from the web servers: http://10.250.101.137/cmd.exe

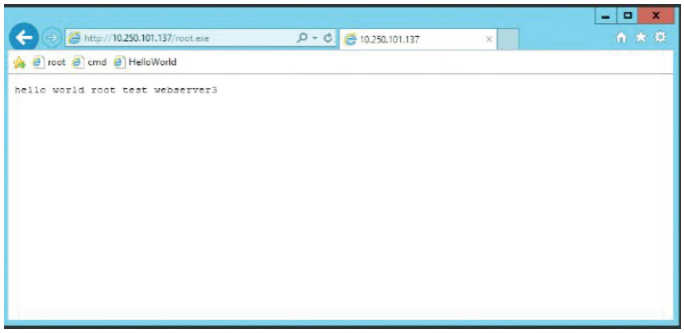
• Because the file is not allowed, it should be blocked and the browser should show an error failing to get the file by timeout.

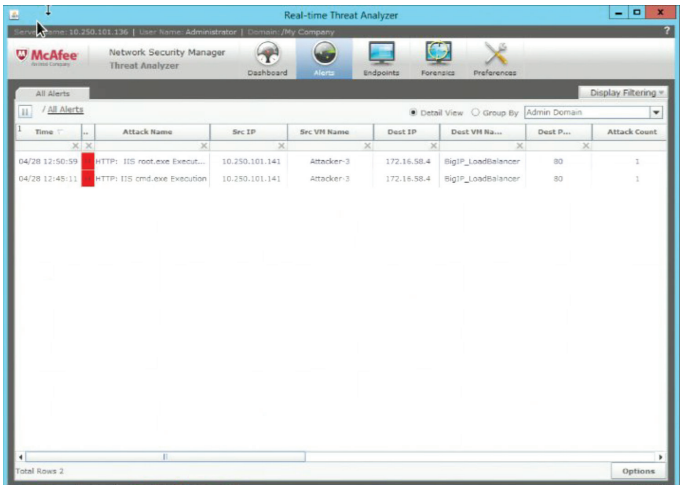


12. In the Real-Time Threat Analyzer windows, the attack should show up.



13. There is another kind of attack that should pass to the user by sending a notification to Real Time Threat Analyzer. For this kind of attack we use: http://10.250.101.137/root.exe.





Appendix A: The PackStack Answer File

```
[general]
    CONFIG_SSH_KEY=/root/.ssh/id_rsa.pub
    CONFIG_DEFAULT_PASSWORD=<set your
password>
    CONFIG_MARIADB_INSTALL=y
    CONFIG_GLANCE_INSTALL=y
    CONFIG_CINDER_INSTALL=y
    CONFIG_MANILA_INSTALL=n
    CONFIG_NOVA_INSTALL=y
    CONFIG_NEUTRON_INSTALL=y
    CONFIG_HORIZON_INSTALL=y
    CONFIG_SWIFT_INSTALL=y
    CONFIG_CEILOMETER_INSTALL=y
    CONFIG_HEAT_INSTALL=n
    CONFIG_SAHARA_INSTALL=n
    CONFIG_TROVE_INSTALL=n
    CONFIG_IRONIC_INSTALL=n
    CONFIG_CLIENT_INSTALL=y
    CONFIG_NTP_SERVERS=
    CONFIG_NAGIOS_INSTALL=n
    EXCLUDE_SERVERS=
    CONFIG_DEBUG_MODE=n
    CONFIG_CONTROLLER_HOST=10.250.101.10
    CONFIG_COMPUTE_HOS
TS=10.250.101.12,10.250.101.14
    CONFIG_NETWORK_HOSTS=10.250.101.10
    CONFIG_VMWARE_BACKEND=n
    CONFIG_UNSUPPORTED=n
    CONFIG_USE_SUBNETS=n
    CONFIG_VCENTER_HOST=
    CONFIG_VCENTER_USER=
    CONFIG_VCENTER_PASSWORD=
    CONFIG_VCENTER_CLUSTER_NAME=
    CONFIG_STORAGE_HOST=10.250.101.10
    CONFIG_SAHARA_HOST=10.250.101.10
    CONFIG_USE_EPEL=n
    CONFIG_REPO=
    CONFIG_ENABLE_RDO_TESTING=n
    CONFIG_RH_USER=
    CONFIG_SATELLITE_URL=
    CONFIG_RH_PW=
    CONFIG_RH_OPTIONAL=y
    CONFIG_RH_PROXY=
    CONFIG_RH_PROXY_PORT=
    CONFIG_RH_PROXY_USER=
    CONFIG_RH_PROXY_PW=
    CONFIG_SATELLITE_USER=
    CONFIG_SATELLITE_PW=
    CONFIG_SATELLITE_AKEY=
    CONFIG_SATELLITE_CACERT=
    CONFIG_SATELLITE_PROFILE=
    CONFIG_SATELLITE_FLAGS=
    CONFIG_SATELLITE_PROXY=
    CONFIG_SATELLITE_PROXY_USER=
    CONFIG_SATELLITE_PROXY_PW=
    CONFIG_SSL_CACERT_FILE=/etc/pki/tls/
certs/selfcert.crt
    CONFIG_SSL_CACERT_KEY_FILE=/etc/pki/
tls/private/selfkey.key
    CONFIG_SSL_CERT_DIR=~/packstackca/
```

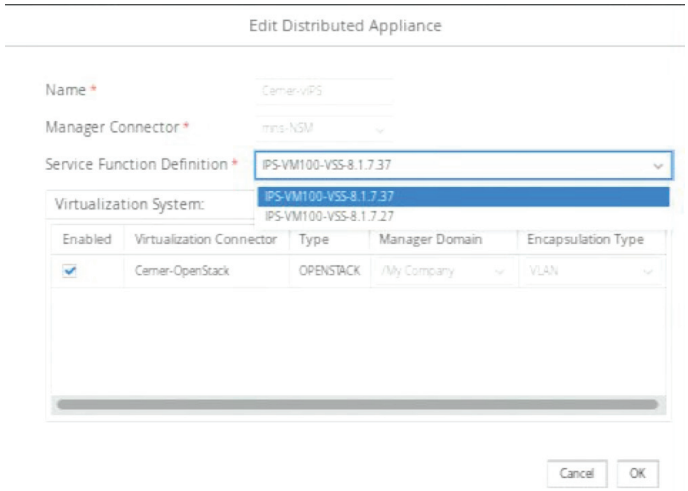
```
    CONFIG_SSL_CACERT_SELFSIGN=y
    CONFIG_SELFSIGN_CACERT_SUBJECT_C=--
    CONFIG_SELFSIGN_CACERT_SUBJECT_
ST=State
    CONFIG_SELFSIGN_CACERT_SUBJECT_
L=City
    CONFIG_SELFSIGN_CACERT_SUBJECT_
O=openstack
    CONFIG_SELFSIGN_CACERT_SUBJECT_
OU=packstack
    CONFIG_SELFSIGN_CACERT_SUBJECT_
CN=securitypoc
    CONFIG_SELFSIGN_CACERT_SUBJECT_
MAIL=admin@securitypoc
    CONFIG_AMQP_BACKEND=rabbitmq
    CONFIG_AMQP_HOST=10.250.101.10
    CONFIG_AMQP_ENABLE_SSL=n
    CONFIG_AMQP_ENABLE_AUTH=n
    CONFIG_AMQP_NSS_CERTDB_PW=<set your
password>    CONFIG_AMQP_AUTH_USER=amqp_
user
    CONFIG_AMQP_AUTH_PASSWORD=<set your
password>
    CONFIG_MARIADB_HOST=10.250.101.10
    CONFIG_MARIADB_USER=root
    CONFIG_MARIADB_PW=intel
    CONFIG_KEYSTONE_DB_PW=intel
    CONFIG_KEYSTONE_REGION=RegionOne
    CONFIG_KEYSTONE_ADMIN_TOKEN=intel
    CONFIG_KEYSTONE_ADMIN_EMAIL=root@
localhost
    CONFIG_KEYSTONE_ADMIN_USERNAME=admin
    CONFIG_KEYSTONE_ADMIN_PW=intel
    CONFIG_KEYSTONE_DEMO_PW=intel
    CONFIG_KEYSTONE_API_VERSION=v2.0
    CONFIG_KEYSTONE_TOKEN_FORMAT=UUID
    CONFIG_KEYSTONE_SERVICE_
NAME=keystone
    CONFIG_KEYSTONE_IDENTITY_BACKEND=sql
    CONFIG_KEYSTONE_LDAP_
URL=ldap://10.250.101.10
    CONFIG_KEYSTONE_LDAP_USER_DN=
    CONFIG_KEYSTONE_LDAP_USER_PASSWORD=
    CONFIG_KEYSTONE_LDAP_SUFFIX=
    CONFIG_KEYSTONE_LDAP_QUERY_SCOPE=one
    CONFIG_KEYSTONE_LDAP_PAGE_SIZE=-1
    CONFIG_KEYSTONE_LDAP_USER_SUBTREE=
    CONFIG_KEYSTONE_LDAP_USER_FILTER=
    CONFIG_KEYSTONE_LDAP_USER_
OBJECTCLASS=
    CONFIG_KEYSTONE_LDAP_USER_ID_
ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_USER_NAME_
ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_USER_MAIL_
ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_USER_ENABLED_
ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_USER_ENABLED_
MASK=-1
    CONFIG_KEYSTONE_LDAP_USER_ENABLED_
DEFAULT=TRUE
    CONFIG_KEYSTONE_LDAP_USER_ENABLED_
INVERT=n
    CONFIG_KEYSTONE_LDAP_USER_ATTRIBUTE_
IGNORE=
    CONFIG_KEYSTONE_LDAP_USER_DEFAULT_
PROJECT_ID_ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_USER_ALLOW_
CREATE=n
    CONFIG_KEYSTONE_LDAP_USER_ALLOW_
UPDATE=n
    CONFIG_KEYSTONE_LDAP_USER_ALLOW_
DELETE=n
    CONFIG_KEYSTONE_LDAP_USER_PASS_
ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_USER_ENABLED_
EMULATION_DN=
    CONFIG_KEYSTONE_LDAP_USER_
ADDITIONAL_ATTRIBUTE_MAPPING=
    CONFIG_KEYSTONE_LDAP_GROUP_SUBTREE=
    CONFIG_KEYSTONE_LDAP_GROUP_FILTER=
    CONFIG_KEYSTONE_LDAP_GROUP_
OBJECTCLASS=
    CONFIG_KEYSTONE_LDAP_GROUP_ID_
ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_GROUP_NAME_
ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_GROUP_MEMBER_
ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_GROUP_DESC_
ATTRIBUTE=
    CONFIG_KEYSTONE_LDAP_GROUP_
ATTRIBUTE_IGNORE=
    CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_
CREATE=n
    CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_
UPDATE=n
    CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_
DELETE=n
    CONFIG_KEYSTONE_LDAP_GROUP_
ADDITIONAL_ATTRIBUTE_MAPPING=
    CONFIG_KEYSTONE_LDAP_USE_TLS=n
    CONFIG_KEYSTONE_LDAP_TLS_CACERTDIR=
    CONFIG_KEYSTONE_LDAP_TLS_CACERTFILE=
    CONFIG_KEYSTONE_LDAP_TLS_REQ_
CERT=demand
    CONFIG_GLANCE_DB_PW=intel
    CONFIG_GLANCE_KS_PW=intel
    CONFIG_GLANCE_BACKEND=file
    CONFIG_CINDER_DB_PW=intel
    CONFIG_CINDER_KS_PW=intel
    CONFIG_CINDER_BACKEND=lvm
    CONFIG_CINDER_VOLUMES_CREATE=y
    CONFIG_CINDER_VOLUMES_SIZE=20G
    CONFIG_CINDER_GLUSTER_MOUNTS=
    CONFIG_CINDER_NFS_MOUNTS=
    CONFIG_CINDER_NETAPP_LOGIN=
    CONFIG_CINDER_NETAPP_PASSWORD=
    CONFIG_CINDER_NETAPP_HOSTNAME=
    CONFIG_CINDER_NETAPP_SERVER_PORT=80
    CONFIG_CINDER_NETAPP_STORAGE_
FAMILY=ontap_cluster
    CONFIG_CINDER_NETAPP_TRANSPORT_
TYPE=http
    CONFIG_CINDER_NETAPP_STORAGE_
PROTOCOL=nfs
    CONFIG_CINDER_NETAPP_SIZE_
```

```
MULTIPLIER=1.0
    CONFIG_CINDER_NETAPP_EXPIRY_THRES_
MINUTES=720
    CONFIG_CINDER_NETAPP_THRES_AVL_SIZE_
PERC_START=20
    CONFIG_CINDER_NETAPP_THRES_AVL_SIZE_
PERC_STOP=60
    CONFIG_CINDER_NETAPP_NFS_SHARES=
CONFIG_CINDER_NETAPP_NFS_SHARES_CONFIG=/
etc/cinder/shares.conf
    CONFIG_CINDER_NETAPP_VOLUME_LIST=
    CONFIG_CINDER_NETAPP_VFILER=
    CONFIG_CINDER_NETAPP_PARTNER_
BACKEND_NAME=
    CONFIG_CINDER_NETAPP_VSERVER=
    CONFIG_CINDER_NETAPP_CONTROLLER_IPS=
    CONFIG_CINDER_NETAPP_SA_PASSWORD=
    CONFIG_CINDER_NETAPP_ESERIES_HOST_
TYPE=linux_dm_mp
    CONFIG_CINDER_NETAPP_WEBSERVICE_
PATH=/devmgr/v2
    CONFIG_CINDER_NETAPP_STORAGE_POOLS=
    CONFIG_MANILA_DB_PW=intel
    CONFIG_MANILA_KS_PW=intel
    CONFIG_MANILA_BACKEND=generic
    CONFIG_MANILA_NETAPP_DRV_HANDLES_
SHARE_SERVERS=false
    CONFIG_MANILA_NETAPP_TRANSPORT_
TYPE=https
    CONFIG_MANILA_NETAPP_LOGIN=admin
    CONFIG_MANILA_NETAPP_PASSWORD=
    CONFIG_MANILA_NETAPP_SERVER_
HOSTNAME=
    CONFIG_MANILA_NETAPP_STORAGE_
FAMILY=ontap_cluster
    CONFIG_MANILA_NETAPP_SERVER_PORT=443
    CONFIG_MANILA_NETAPP_AGGREGATE_
NAME_SEARCH_PATTERN=(.*)
    CONFIG_MANILA_NETAPP_ROOT_VOLUME_
AGGREGATE=
    CONFIG_MANILA_NETAPP_ROOT_VOLUME_
NAME=root
    CONFIG_MANILA_NETAPP_VSERVER=
    CONFIG_MANILA_GENERIC_DRV_HANDLES_
SHARE_SERVERS=true
    CONFIG_MANILA_GENERIC_VOLUME_NAME_
TEMPLATE=manila-share-%s
    CONFIG_MANILA_GENERIC_SHARE_MOUNT_
PATH=/shares
    CONFIG_MANILA_SERVICE_IMAGE_
LOCATION=https://www.dropbox.com/s/
vi5oeh10q1qkckh/ubuntu_1204_nfs_cifs.
qcow2
    CONFIG_MANILA_SERVICE_INSTANCE_
USER=ubuntu
    CONFIG_MANILA_SERVICE_INSTANCE_
PASSWORD=
    CONFIG_MANILA_NETWORK_TYPE=neutron
    CONFIG_MANILA_NETWORK_STANDALONE_
GATEWAY=
    CONFIG_MANILA_NETWORK_STANDALONE_
NETMASK=
    CONFIG_MANILA_NETWORK_STANDALONE_
SEG_ID=
    CONFIG_MANILA_NETWORK_STANDALONE_IP_
RANGE=
    CONFIG_MANILA_NETWORK_STANDALONE_IP_
VERSION=4
    CONFIG_IRONIC_DB_PW=intel
    CONFIG_IRONIC_KS_PW=intel
    CONFIG_NOVA_DB_PW=intel
    CONFIG_NOVA_KS_PW=intel
    CONFIG_NOVA_SCHED_CPU_ALLOC_
RATIO=16.0
    CONFIG_NOVA_SCHED_RAM_ALLOC_
RATIO=1.5
    CONFIG_NOVA_COMPUTE_MIGRATE_
PROTOCOL=tcp
    CONFIG_NOVA_COMPUTE_MANAGER=nova.
compute.manager.ComputeManager
    CONFIG_VNC_SSL_CERT=
    CONFIG_VNC_SSL_KEY=
    CONFIG_NOVA_COMPUTE_PRIVIF=ens20f1
    CONFIG_NOVA_NETWORK_MANAGER=nova.
network.manager.FlatDHCPManager
    CONFIG_NOVA_NETWORK_PUBIF=ens20f0
    CONFIG_NOVA_NETWORK_PRIVIF=ens20f1
    CONFIG_NOVA_NETWORK_
FIXEDRANGE=192.168.32.0/22
    CONFIG_NOVA_NETWORK_
FLOATRANGE=10.3.4.0/22
    CONFIG_NOVA_NETWORK_
AUTOASSIGNFLOATINGIP=n
    CONFIG_NOVA_NETWORK_VLAN_START=100
    CONFIG_NOVA_NETWORK_NUMBER=1
    CONFIG_NOVA_NETWORK_SIZE=255
    CONFIG_NEUTRON_KS_PW=intel
    CONFIG_NEUTRON_DB_PW=intel
    CONFIG_NEUTRON_L3_EXT_BRIDGE=
    CONFIG_NEUTRON_METADATA_PW=intel
    CONFIG_LBAAS_INSTALL=n
    CONFIG_NEUTRON_METERING_AGENT_
INSTALL=n
    CONFIG_NEUTRON_FWAAS=n
    CONFIG_NEUTRON_ML2_TYPE_
DRIVERS=vxlan
    CONFIG_NEUTRON_ML2_TENANT_NETWORK_
TYPES=vxlan
    CONFIG_NEUTRON_ML2_MECHANISM_
DRIVERS=openvswitch
    CONFIG_NEUTRON_ML2_FLAT_NETWORKS=*
    CONFIG_NEUTRON_ML2_VLAN_
RANGES=physnet1,physnet2
    CONFIG_NEUTRON_ML2_TUNNEL_ID_RANGES=
    CONFIG_NEUTRON_ML2_VXLAN_
GROUP=239.1.1.100
    CONFIG_NEUTRON_ML2_VNI_
RANGES=1001:2000
    CONFIG_NEUTRON_L2_AGENT=openvswitch
    CONFIG_NEUTRON_LB_INTERFACE_
MAPPINGS=
    CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=
    CONFIG_NEUTRON_OVS_BRIDGE_IFACES=
    CONFIG_NEUTRON_OVS_TUNNEL_IF=
    CONFIG_NEUTRON_OVS_VXLAN_UDP_
PORT=4789
    CONFIG_HORIZON_SSL=n
```

```
    CONFIG_HORIZON_SECRET_KEY=dd5a2abbce
f747f7a7bafede42947d71
    CONFIG_HORIZON_SSL_CERT=
    CONFIG_HORIZON_SSL_KEY=
    CONFIG_HORIZON_SSL_CACERT=
    CONFIG_SWIFT_KS_PW=intel
    CONFIG_SWIFT_STORAGES=
    CONFIG_SWIFT_STORAGE_ZONES=1
    CONFIG_SWIFT_STORAGE_REPLICAS=1
    CONFIG_SWIFT_STORAGE_FSTYPE=ext4
    CONFIG_SWIFT_HASH=c2a8ece9563b4666
    CONFIG_SWIFT_STORAGE_SIZE=2G
    CONFIG_HEAT_DB_PW=intel
    CONFIG_HEAT_AUTH_ENC_
KEY=eb12297f095c4958
    CONFIG_HEAT_KS_PW=intel
    CONFIG_HEAT_CLOUDWATCH_INSTALL=n
    CONFIG_HEAT_CFN_INSTALL=n
    CONFIG_HEAT_DOMAIN=heat
    CONFIG_HEAT_DOMAIN_ADMIN=heat_admin
    CONFIG_HEAT_DOMAIN_PASSWORD=<set
your password>
    CONFIG_PROVISION_DEMO=n
    CONFIG_PROVISION_TEMPEST=n
    CONFIG_PROVISION_DEMO_
FLOATRANGE=172.24.4.224/28
    CONFIG_PROVISION_IMAGE_NAME=cirros
    CONFIG_PROVISION_IMAGE_URL=http://
download.cirros-cloud.net/0.3.3/cirros-
0.3.3-x86_64-disk.img
    CONFIG_PROVISION_IMAGE_FORMAT=qcow2
    CONFIG_PROVISION_IMAGE_SSH_
USER=cirros
    CONFIG_PROVISION_TEMPEST_USER=
    CONFIG_PROVISION_TEMPEST_USER_
PW=intel
    CONFIG_PROVISION_TEMPEST_
FLOATRANGE=172.24.4.224/28
    CONFIG_PROVISION_TEMPEST_REPO_
URI=https://github.com/openstack/tempest.
git
    CONFIG_PROVISION_TEMPEST_REPO_
REVISION=master
    CONFIG_PROVISION_ALL_IN_ONE_OVS_
BRIDGE=n
    CONFIG_CEILOMETER_
SECRET=54188c6a86154776
    CONFIG_CEILOMETER_KS_PW=intel
    CONFIG_CEILOMETER_COORDINATION_
BACKEND=redis
    CONFIG_MONGODB_HOST=10.250.101.10
    CONFIG_REDIS_MASTER_HOST=10.250.101.10
    CONFIG_REDIS_PORT=6379
    CONFIG_REDIS_HA=n
    CONFIG_REDIS_SLAVE_HOSTS=
    CONFIG_REDIS_SENTINEL_HOSTS=
    CONFIG_REDIS_SENTINEL_CONTACT_HOST=
    CONFIG_REDIS_SENTINEL_PORT=26379
    CONFIG_REDIS_SENTINEL_QUORUM=2
    CONFIG_REDIS_MASTER_NAME=mymaster
    CONFIG_SAHARA_DB_PW=intel
    CONFIG_SAHARA_KS_PW=intel
    CONFIG_TROVE_DB_PW=intel
    CONFIG_TROVE_KS_PW=intel
    CONFIG_TROVE_NOVA_USER=trove
    CONFIG_TROVE_NOVA_TENANT=services
    CONFIG_TROVE_NOVA_PW=intel
    CONFIG_NAGIOS_PW=intel
```

Appendix B: Updating the vIPS Sensor Image (Upgrade Version)

1. Log in to the OSC dashboard.
2. On the menu, select `Setup, Service Function Catalog`.
3. Click `Auto Import`.
4. Select the zip file with the new vIPS sensor.
5. Click `OK`.
6. On the menu, select `Setup, Distributed Appliance`.
7. Select the current distributed appliance.
8. Click `Edit`.
9. On the menu `Service Function Definition`, select the new sensor.



10. Click `OK`.
11. Wait until the job is complete.
12. Check whether vIPS has a new image in the OpenStack dashboard.

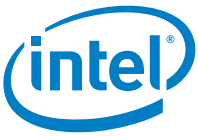
Appendix C: References

NAME	REFERENCE
Elasticsearch (v1.7.3)	https://www.elastic.co/guide/en/elasticsearch/reference/1.7/setup-repositories.html
F5 BIG-IP	https://www.f5.com/pdf/products/big-ip-local-traffic-manager-ds.pdf
Installing MidoNet for OSC	https://docs.google.com/a/intel.com/document/d/1O7nBgYS9dFd3qqLDAKdNMC_pVsBko5zQqGcJrctqQuM/edit?usp=sharing_eid
Open Security Controller	http://www.intel.com/content/dam/www/public/us/en/documents/datasheets/open-security-controller-datasheet.pdf
Logstash (v1.5.4)	https://www.elastic.co/guide/en/logstash/1.5/package-repositories.html
McAfee® Network Security Manager	http://www.mcafee.com/sg/resources/data-sheets/ds-network-security-manager.pdf
McAfee® Network Security Platform virtual sensor	http://www.mcafee.com/us/resources/data-sheets/ds-virtual-network-security-platform.pdf
Midokura Enterprise Midonet	http://www.midokura.com/midonet-enterprise/
Midokura Enterprise MidoNet (MEM) Quick Start Guide for Red Hat Enterprse Linux 7 / Kilo	http://docs.midokura.com/docs/latest-en/quick-start-guide/rhel-7_kilo-osp/content/index.html
Midokura page on GitHub*	https://github.com/midokura
Red Hat Enterprise Linux 7	https://www.redhat.com/en/resources/red-hat-enterprise-linux-server
Red Hat OpenStack Platform 7	https://access.redhat.com/documentation/en/red-hat-openstack-platform?version=7/

Appendix D: Abbreviations

ABBREVIATION	DESCRIPTION
ARP	Address Resolution Protocol
CLI	Command Line Interface
DNS	Domain Name System
GRE	Generic Routing Encapsulation
IDS	Intrusion Detection System
IP	Internet Protocol
IPS	Intrusion Prevention System
JVM	Java Virtual Machine
LAN	Local Area Network
MAC	Medium Access Control
NFV	Network Functions Virtualization

ABBREVIATION	DESCRIPTION
NSM	McAfee Network Security Manager
OSC	Open Security Controller
RAM	Random Access Memory
SDN	Software-Defined Networking
SHVS	Standard High Volume Servers
UDP	User Datagram Protocol
vIPS	Virtual IPS
VLAN	Virtual LAN
VM	Virtual Machine
VxLAN	Virtual eXtensible LAN



 1216/MH/MESH/PDF 335218-001US