PF6800 Ver. 6.1
Neutron NEC OpenFlow Plugin Configuration Guide for Red Hat Enterprise Linux OpenStack Platform 6.0

First Edition, April 2015

Legal Notices

Symbols

In this manual, the following two types of symbols are used. These symbols and their meanings are important for proper handling of the PFC.

■CHECK:■	Points that should be checked when operating devices or software.
■NOTE:■	Helpful, good-to-know information

Disclaimer

Unless explicitly set forth in a license agreement, the NEC Corporation makes no explicit or implicit guarantees regarding this product and the related documentation, including its merchantability or fitness for a particular purpose, and disclaims all liability pertaining to its handling, use, or attendant trade practices.

Contents

Preface

In this guide the example of how to configure Red Hat Enterprise Linux OpenStack Platform environment is given.

■CHECK:■ In this manual even though it is mentioned just OpenStack, it means Red Hat Enterprise Linux OpenStack Platform.

■CHECK:■ OpenStack and PFC coordinate using OpenStack function of WebAPI which is a component of PFC. Please read “PF6800 Ver. 6.1 WebAPI User’s Guide” before configuring the OpenStack environment.

Chapter 1 Summary of system configuration

Here the explanation about system configuration, software, hardware, and network is given.

1.1. System configuration and explanation of terminology

Example of system configuration of OpenStack environment explained in this chapter is shown in Figure 1.1-1.

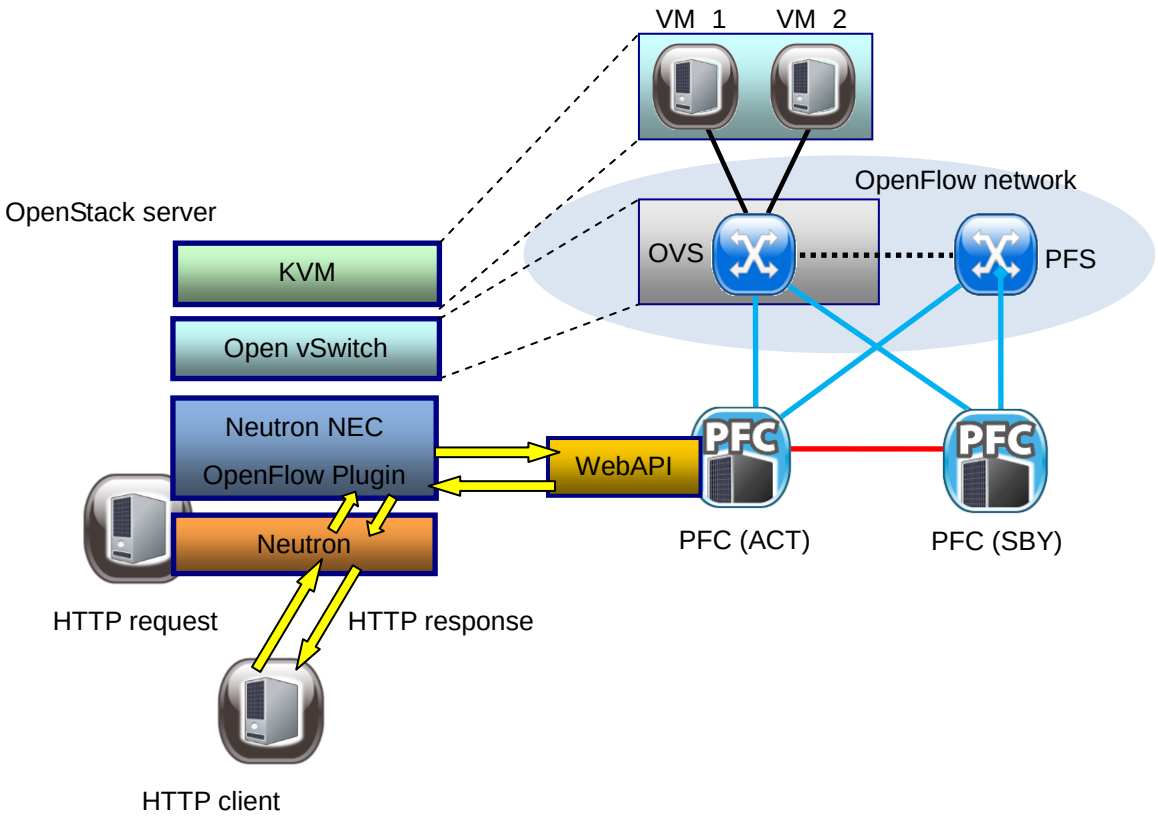


Figure 1.1-1 Example of OpenStack environment system configuration that uses WebAPI

The terminology used in the explanation of OpenStack environment is given in "Table 1.1-1 Terms used in OpenStack Environment".

Table 1.1-1 Terms used in OpenStack Environment

name of function	details
OpenStack (Red Hat Enterprise Linux OpenStack Platform)	It is the core component of commercial product " Red Hat Enterprise Linux OpenStack Platform" based on the open source cloud platform management software " OpenStack"
Neutron	It is a generic name of network connection function provided by OpenStack project.
Neutron NEC OpenFlow Plugin	It is a plugin provided by NEC for Neutron. Coordination between Neutron and PFC (WebAPI) is possible through this plug in. Hereinafter referred to as "NEC Plugin".
Keystone	It is a generic name of authentication function provided by OpenStack project.
Open vSwitch	It is open source virtual software switch. Hereinafter referred to as "OVS".
KVM	It is virtualization function that runs on Linux. This function needs to be run on OpenStack server in order to run VM in the OpenStack environment.

1.2. Details of hardware and software

This section explains the hardware and software that configures OpenStack environment.

1.2.1. Details of hardware

The hardware required to configure OpenStack environment is described below.

Table 1.2-1 Hardware

Name	Model	Remarks
OpenStack server	Use the model equivalent to (or above) Express5800 R120a-2	Installation of OpenStack and Open vSwitch.
PFC	Refer "1.3 Preparation for Installing PFC" of installation guide.	2 Machines are required as server for running PFC and WebAPI (ACT and SBY)

PFS	Use model of PF5240	None

■CHECK:■ In the explanation of subsequent sections, the expressions used in "Name" column of above table are used.

1.2.2. Details of software

The software required to configure OpenStack environment is given below.

Table 1.2-2 Software

Name	version	Remarks
PFC	V6.1	None.
Red Hat Enterprise Linux Server	7.0	Install as OS of OpenStack server.
Red Hat Enterprise Linux OpenStack Platform	6.0 (Juno)	Includes Neutron.
NEC Plugin	2014.2	Install on OpenStack server.
Open vSwitch	2.1.3	Install on OpenStack server.

■CHECK:■ The versions mentioned in the above table may change because of version up of OpenStack and PFC.

1.3. Network planning

This section shows examples of network configuration.

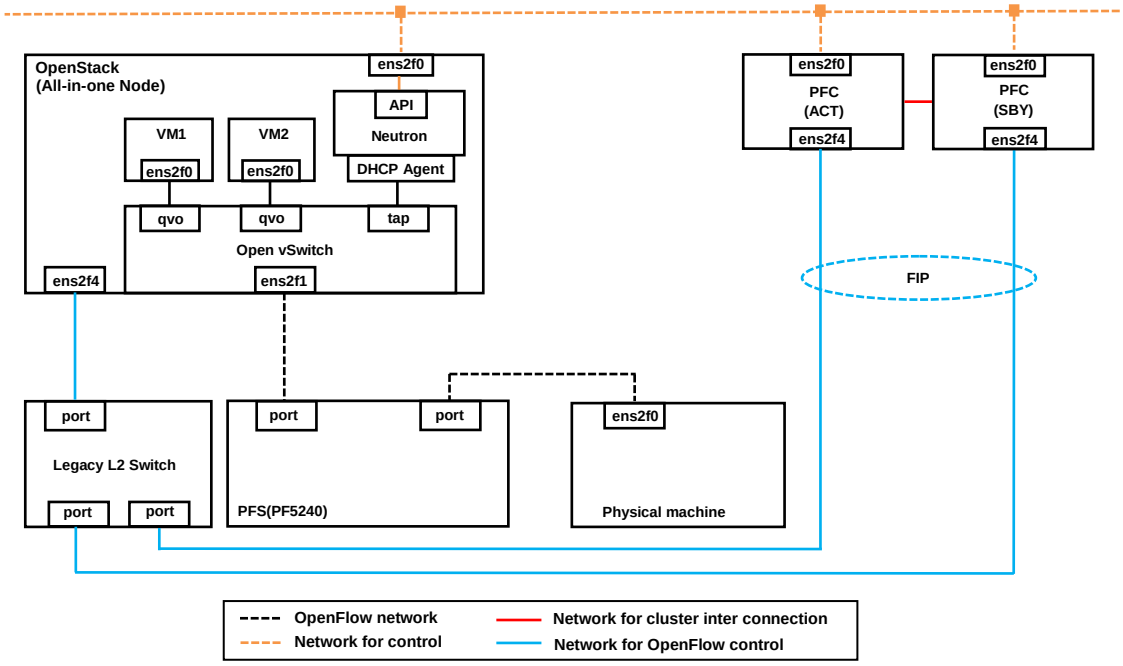


Figure 1.3-1 Single node deployment Diagram

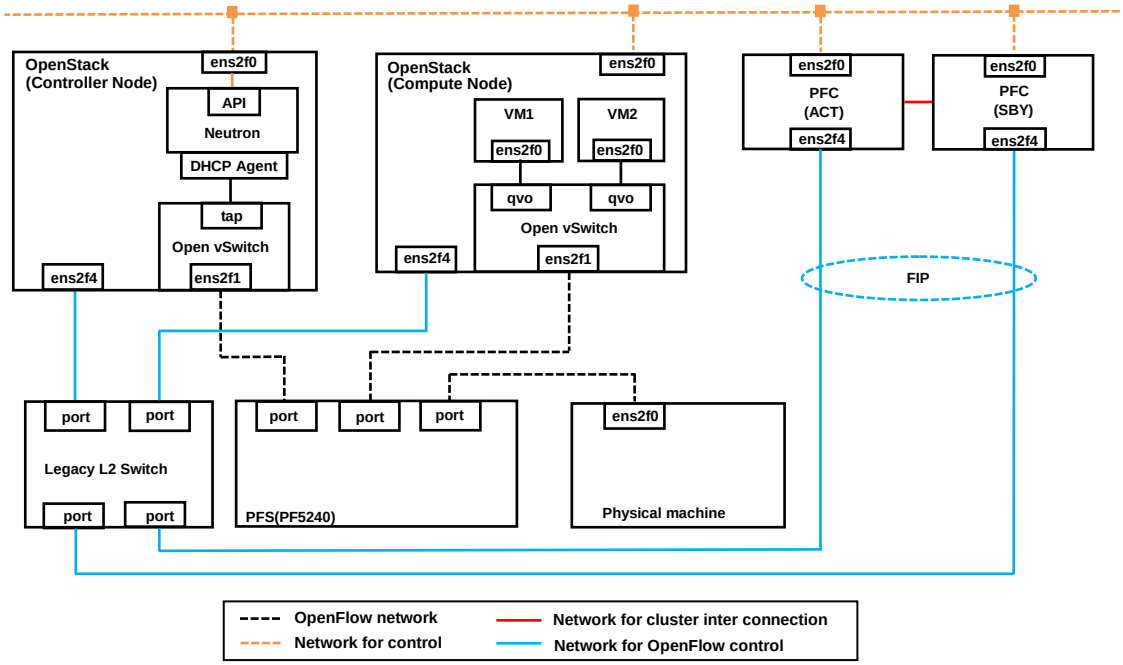


Figure 1.3-2 Multiple node deployment Diagram

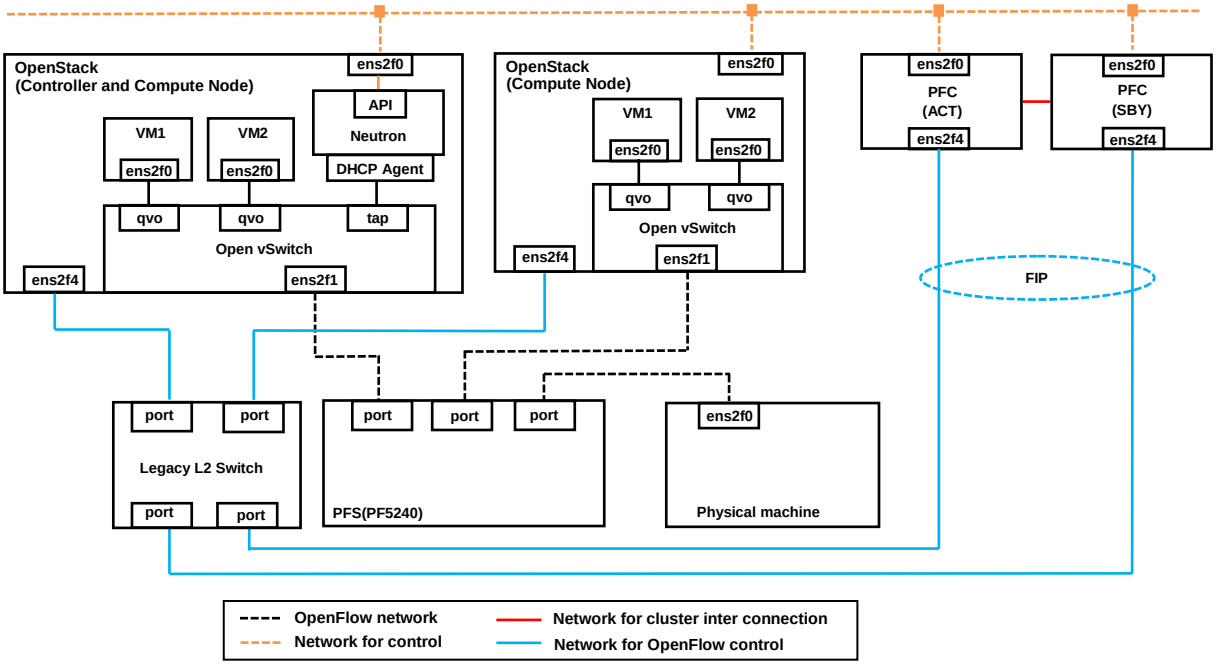


Figure 1.3-3 Another multiple node deployment Diagram

Chapter 2 Preparation

2.1. Setting PFC server

2.1.1. Installing PFC

For installation procedure of PFC, refer to "PF6800 Ver. 6.0 Installation Guide".

Table 2.1-1 Setting IP address of PFC

Where to set	Name of the device	IP address
PFC1 (ACT)	ens2f0	Any IP address of management network
	ens2f4	Refer to "1.3 Network planning" and specify the value that is adequate to configuration environment.
PFC2 (SBY)	ens2f0	Any IP address of management network
	ens2f4	Refer to "1.3 Network planning" and specify the value that is adequate to configuration environment.
Floating IP for SecureChannel	FIP	Refer to "1.3 Network planning" and specify the value that is adequate to configuration environment.

2.1.2. Setting WebAPI

■CHECK:■ The settings explained here need to be performed for ACT, as well as for SBY.

2.1.2.1. Setting auto start of WebAPI

■CHECK:■ For details of settings, refer "2.1.3 Specifying the WebAPI Startup Parameters" of "PF6800 Ver. 6.1 WebAPI User's Guide".

2.1.2.2. Setting port of OpenStack function

1. Edit listenport.properties

Open the WebAPI configuration file /opt/nec/pfc/Agent/sg/webapi/listenport.properties by using an editor and set the WebAPI (OpenStack function) destination port to the 'quantum_port' parameter.

```
# Define listen port between 0 and 65535.
#   0: Does not start the server.
#   1-65535: Start the server and listen on the defined port.
# If not defined any, listen on the default port.

# Basic Port
# (Default: 8080)
#
#webapi_port=8080

# QuantumAPI Port
# (Default: 8888)
#
quantum_port= 8888
```

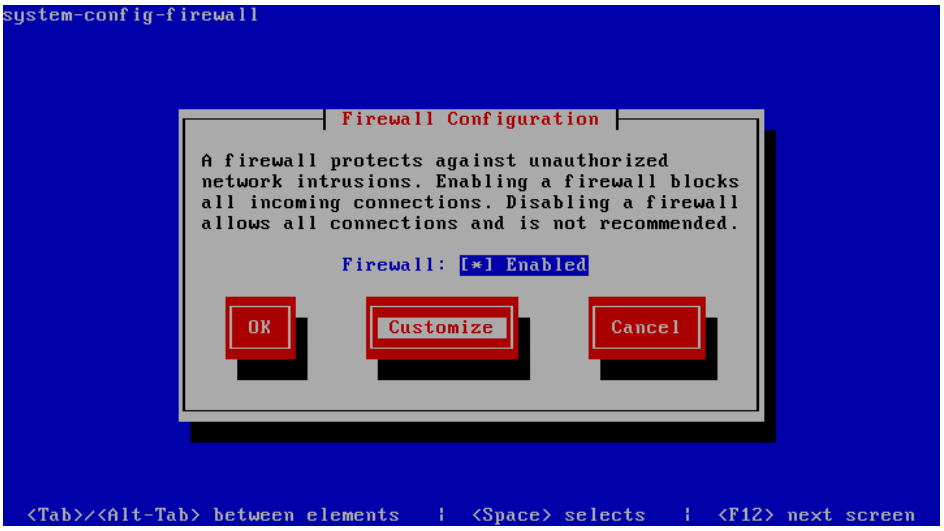
■NOTE:■ The OpenStack functions in the WebAPI are disabled when 0 is set to 'quantum_port'.

2. Save listenport.properties

3. Execute the following command.

```
# system-config-firewall-tui
```

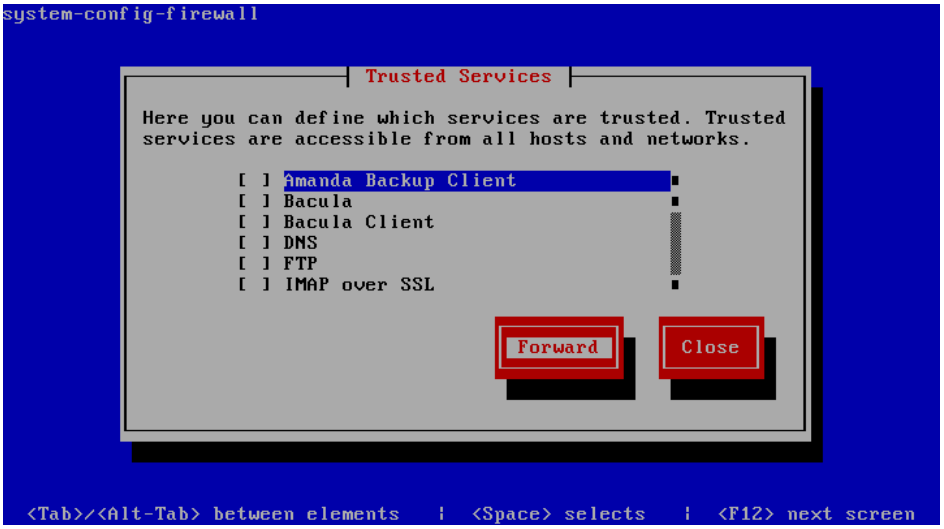
4. 'Firewall Configuration' screen is displayed.



Set the focus on [Customize], and then press the space key.

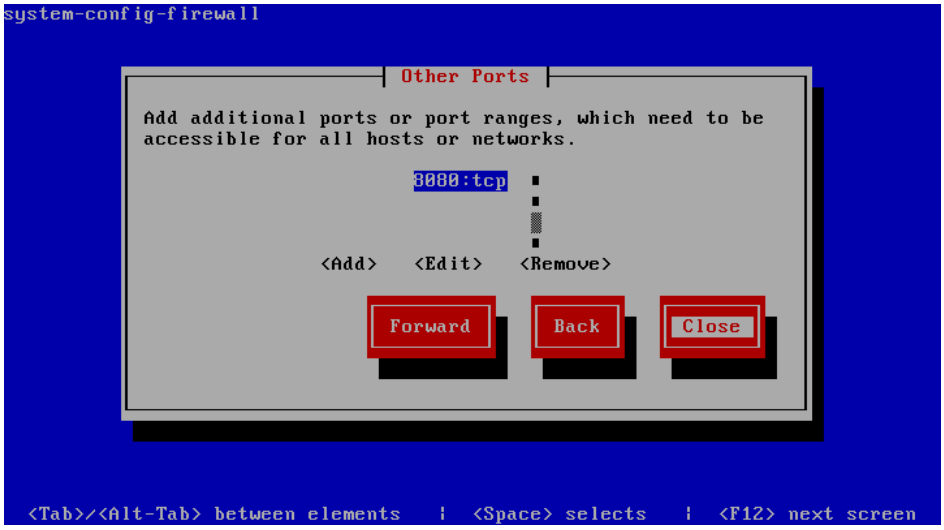
■NOTE:■ You can move the focus by pressing the Tab key.

5. 'Trusted Services' screen is displayed.



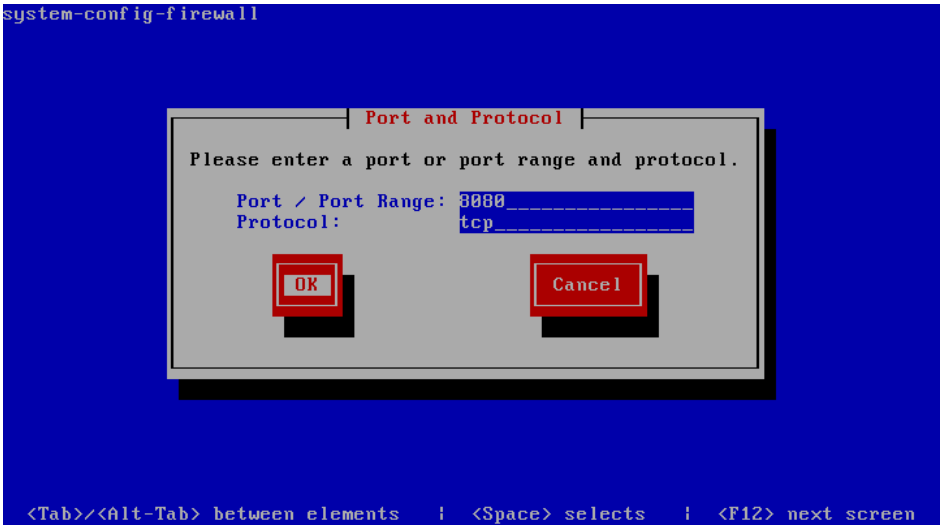
Set the focus on [Forward], and then press the space key.

6. ‘Other Ports’ screen is displayed.



To use the WebAPI (OpenStack function), set the focus on [<Add>] and then press the space key (Go to step 7).

7. ‘Port and Protocol’ screen is displayed again.

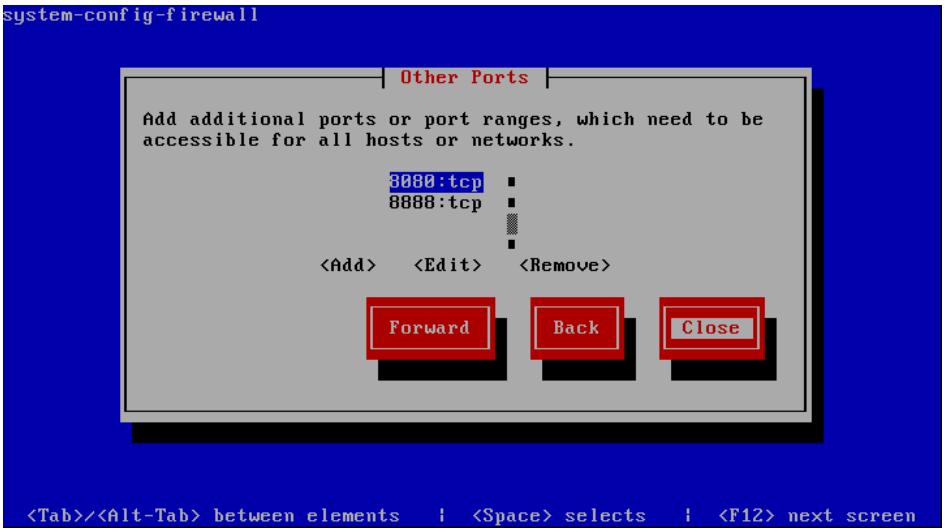


Input the following values.

Name	Value
Port / Port Range	The port number of WebAPI (OpenStack function) that has been specified in above procedure.
Protocol	tcp

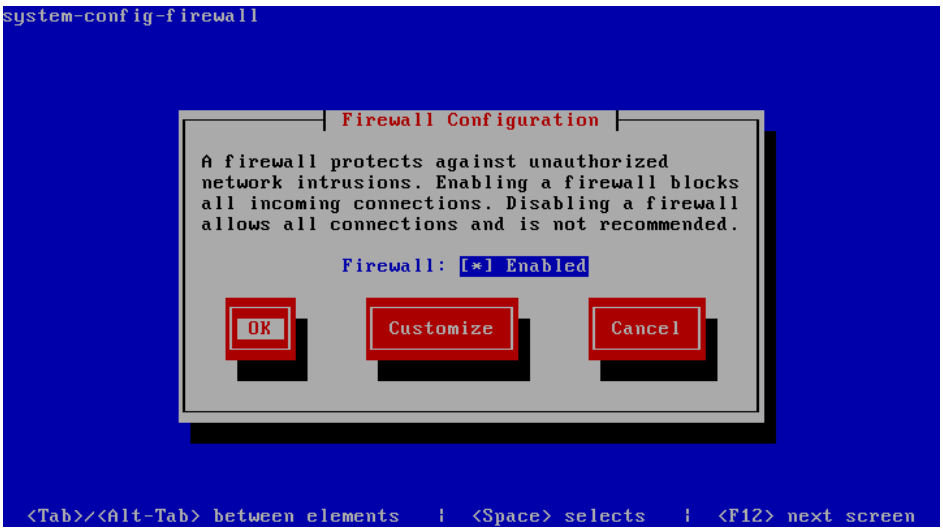
After inputting values, set the focus on [OK] and then press the space key.

8. ‘Other Ports’ screen is displayed again.



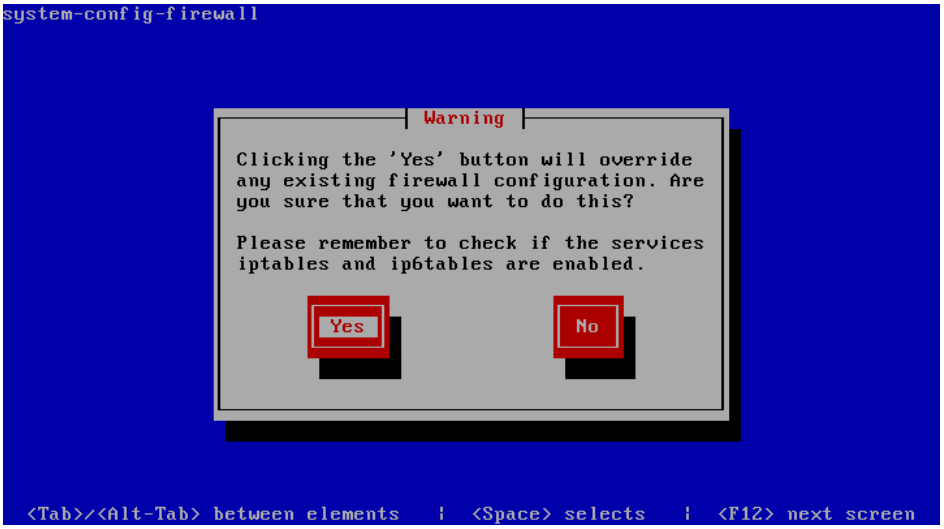
After checking that the specified port number and protocol are displayed, set the focus on [Close] and then press the space key.

9. 'Firewall Configuration' screen is displayed again.



Set the focus on [OK] and then press the space key.

10. ‘Warning’ screen is displayed.



Set the focus on [Yes] and then press the space key.

2.1.2.3. Restarting PFC

Restart both ACT and SBY PFC servers so that the settings become effective.

```
# pfc_stop_cluster                    (stops cluster)
# reboot                              (restarts)
```

2.1.3. Setting of auto saving startup-configuration

■CHECK:■ For details, refer to "1.2.7 Applying the Settings by WebAPI to startup-configuration" of "PF6800 Ver. 6.1 WebAPI User's Guide".

2.2. Settings for OpenStack server

2.2.1. BIOS settings

Enable the system Virtualization Technology on the server on which OpenStack will run.

1. Power On the server on which OpenStack server will run.
2. Press F2 key when the server is starting (when NEC logo is being displayed) and display BIOS menu.
3. Select [Main]→[Processor Settings].
4. Change [Virtualization Technology:] from [Disabled] to [Enabled].
5. Press F10, save the settings and exit BIOS.

2.2.2. Installing OS

Install Red Hat Enterprise Linux Server 7.0 on the machine on which OpenStack server will run.

2.2.3. Setting network

■CHECK:■	This section assumes that ens2f0 is used for management network, ens2f1 is used for OpenFlow network, and ens2f4 is used for Secure Channel. For details, refer to "1.3 Network planning".

2.2.3.1. Setting network interface

Configure the /etc/sysconfig/network-scripts/ifcfg-ens2f0 of the server on which OpenStack server will run as shown below. (The items that are not described here should be kept in its default state.)

```
ONBOOT=yes
BOOTPROTO=none
IPADDR=[IP address to be allocated to ens2f0 of corresponding server]
PREFIX=[Network prefix length corresponding to above IP address]
GATEWAY=[Appropriate gateway address corresponding to above IP address]
DNS1=[IP address of DNS server]
```

Configure /etc/sysconfig/network-scripts/ifcfg-ens2f1 as shown below.

```
ONBOOT=yes
BOOTPROTO=none
```

Configure /etc/sysconfig/network-scripts/ifcfg-ens2f3 as shown below.

```
ONBOOT=yes
BOOTPROTO=none
IPADDR=[IP address to be allocated to ens2f3 of corresponding server]
NETMASK=[subnet mask corresponding to above IP address]
```

Configure /etc/sysconfig/network of server on which OpenStack server will run as shown below.

```
NETWORKING=yes
HOSTNAME=[host name of the server on which OpenStack server will run]
GATEWAY=[IP address of default gateway]
```

2.2.3.2. Restarting network connection

Restart the network setting so that the settings become effective.

```
$ systemctl restart network
```

2.2.4. Proxy server settings (if needed)

Perform the settings for the proxy server when internet is to be accessed through proxy server from the server that runs OpenStack server.

2.2.4.1. Setting environment variables

Set environment variables http_proxy, https_proxy, no_proxy.

```
http_proxy="http://[Address of proxy server]:[Port No of proxy server]/"
https_proxy="http://[Address of proxy server]:[Port No of proxy server]/"
no_proxy=[Address of keystone server]
```

2.2.4.2. Settings of /etc/sudoers

To enable to inherit the settings of proxy, even at the time of execution of sudo, add following description in /etc/sudoers.

```
Defaults      env_keep = "http_proxy https_proxy no_proxy"
```

2.2.4.3. Settings of /etc/rhsm/rhsm.conf

Do the following settings to use subscription-manager.

/etc/rhsm/rhsm.conf

```
proxy_hostname = [Address of proxy server]
proxy_port = [Port No of proxy server]
```

2.2.4.4. Settings of yum

Do the following settings to install the package through yum command.

/etc/sysconfig/rhn/up2date

```
enableProxy=1
httpProxy=[Address of proxy server]:[Port No of proxy server]
```

/etc/yum.conf

```
proxy=http://[Address of proxy server]:[Port No of proxy server]/
```

2.2.5. Registration of subscription

Register the subscription using subscription-manager. For registration procedure, please refer the document of Red Hat Enterprise Linux.

Chapter 3 Installing software

3.1. Installing OpenStack

Use environment configuration support tool 'PackStack', and perform consolidated installation and settings. This guide assumes that you have run one of the following commands.

```
# packstack --allinone --provision-demo=n
```

```
# packstack --install-hosts=[IP Address of controller node],[Comma-separated IP Addresses of each compute node] --provision-demo=n
```

3.2. Installing NEC Plugin

Install 'NEC Plugin' using yum command.

```
# yum install -y openstack-neutron-nec
```

■CHECK:■ "NEC Plugin" has to be installed to both of controller node and compute node.

Chapter 4 Node settings

This chapter describes settings of OpenStack nodes.

■CHECK:■	The settings of all-in-one node are the same as controller node.

4.1. Controller node settings

4.1.1. Stop services

Stop Open vSwitch Plugin and disable auto start.

```
# systemctl stop neutron-openvswitch-agent.service
# systemctl disable neutron-openvswitch-agent.service
```

Stop Neutron Server and DHCP Agent.

```
# systemctl stop neutron-server.service
# systemctl stop neutron-dhcp-agent.service
```

4.1.2. Initialize the database

Create the database that will be used by NEC Plugin.

```
# mysql -u [admin user name of MariaDB] -p[admin user password of MariaDB]
mysql> CREATE DATABASE neutron_nec;
mysql> GRANT ALL PRIVILEGES ON neutron_nec.* TO 'neutron'@'localhost' ¥
IDENTIFIED BY '[neutron user password of MariaDB]';
mysql> GRANT ALL PRIVILEGES ON neutron_nec.* TO 'neutron'@'%' ¥
IDENTIFIED BY '[neutron user password of MariaDB]';
```

■NOTE:■	The database has been changed to MariaDB from MySQL.
■NOTE:■	Account settings of MariaDB have been outputted to answer file of PackStack. Item name of each setting is shown below. CONFIG_MARIADB_USER … The name of admin user (Default: root) CONFIG_MARIADB_PW … The password of admin user

CONFIG_NEUTRON_DB_PW … The password of neutron user

4.1.3. Configure Neutron

Edit the configuration file of Neutron (/etc/neutron/neutron.conf)

■CHECK:■	It is not necessary to change the setting that is not designated here.

In [DEFAULT] section, set 'core_plugin', 'api_extensions_path' and 'service_plugins' as follows.

```
core_plugin = neutron.plugins.nec.nec_plugin.NECPluginV2
api_extensions_path = /usr/lib/python2.7/site-packages/neutron/plugins/nec/extensions/
service_plugins =firewall,lbaas,vpnaas,metering
```

■CHECK:■	Items of api_extensions_path are displayed spreading over 2 rows but actually mention those on 1 row.
■CHECK:■	NEC Plugin provides L3 router feature as part of the core plugin. Thus L3 router service plugin should NOT be contained in service_plugin. Therefore it is necessary to delete l3_router because it is specified by default.

Set the items of connection of [database] section as below.

```
connection = mysql://neutron:[neutron user password of MariaDB]@[IP address of controller node]/neutron_nec
```

Edit the configuration file of Neutron DHCP Agent (/etc/neutron/dhcp_agent.ini)

■CHECK:■	It is not necessary to change the setting that is not designated here.

Set 'ovs_use_veth' in [DEFAULT] section as below.

```
ovs_use_veth = True
```

4.1.4. Configure NEC Plugin

Edit the configuration file (/etc/neutron/plugins/nec/nec.ini) of NEC Plugin.

■CHECK:■ It is not necessary to change the setting that is not designated here.

Set [ofc] section as below.

```
host = [IP address of PFC that provided the WebAPI service]
port = [TCP port number that WebAPI is listening on (default: 8888)]
driver = pfc
```

Leave 'default_router_provider' parameter in [provider] section as below.

```
# default_router_provider = l3-agent
```

■NOTE:■ You can change default router provider here, if needed.
openflow… vRouter function provided by PFC
l3-agent … L3 Agent function provided by Neutron

Replace already allocated configuration file of Open vSwitch Plugin with configuration file of NEC Plugin.

```
# rm -f /etc/neutron/plugin.ini
# ln -s /etc/neutron/plugins/nec/nec.ini /etc/neutron/plugin.ini
```

Initialize the database using neutron-db-manage command.
The database will be initialized using the changed configuration file.

```
# neutron-db-manage --config-file /etc/neutron/neutron.conf ¥
  --config-file /etc/neutron/plugins/nec/nec.ini upgrade head
```

4.1.5. Configure Open vSwitch

If you install NEC Plugin in All-in-one with Packstack, OVS is installed by the setting that OVS works with OVS Mechanical driver of ML2 plugin.
So, cancel the setting once, and then set up so that OVS can work with NEC plugin.

Delete br-tun.

```
# ovs-vsctl del-br br-tun
```

Clear br-int.

```
# ovs-vsctl del-br br-int
# ovs-vsctl add-br br-int
```

Connect Open vSwitch with PFC.

```
# ovs-vsctl set-controller br-int tcp:[IP address of PFC that is the connection
destination of SecureChannel]
```

Add the port for OpenFlow network.

```
# ovs-vsctl add-port br-int ens2f1
```

Clear data flow of br-int.

```
# ovs-ofctl del-flows br-int
```

Sample

```
# ovs-vsctl show
8eb7fedb-a2dd-441e-9c4e-77873186f62b
    Bridge br-int
        Controller "tcp:192.168.100.184"
            is_connected: true
        Port "ens2f1"
            Interface "ens2f1"
        Port br-int
            Interface br-int
                type: internal
    Bridge br-ex
        Port br-ex
            Interface br-ex
                type: internal
    ovs_version: "2.1.3"
```

4.1.6. Start services

Start the services of Neutron Server, NEC Plugin.

```
# systemctl start neutron-server.service
# systemctl start neutron-dhcp-agent.service
# systemctl start neutron-nec-agent.service
# systemctl enable neutron-nec-agent.service
```

4.2. Compute node settings

4.2.1. Stop services

Stop Open vSwitch Plugin and cancel auto start setting.

```
# systemctl stop neutron-openvswitch-agent.service
# systemctl disable neutron-openvswitch-agent.service
```

4.2.2. Configure Neutron

Edit the configuration file of Neutron (/etc/neutron/neutron.conf)

■CHECK:■ It is not necessary to change the setting that is not designated here.

Set the [DEFAULT] section core_plugin and api_extensions_path items as shown below.

```
core_plugin = neutron.plugins.nec.nec_plugin.NECPluginV2
```

4.2.3. Configure NEC Plugin

Edit the configuration file（/etc/neutron/plugins/nec/nec.ini）of NEC Plugin.

■CHECK:■ It is not necessary to change the setting that is not designated here.

Set the [ofc] section as below.

```
host = [IP address of PFC that provided the WebAPI service]
port = [TCP port number that the WebAPI is listening on (default: 8888)]
```

```
driver = pfc
```

Replace already allocated configuration file of Open vSwitch Plugin with configuration file of NEC Plugin.

```
# rm -f /etc/neutron/plugin.ini
# ln -s /etc/neutron/plugins/nec/nec.ini /etc/neutron/plugin.ini
```

4.2.4. Configure Open vSwitch

If you install NEC Plugin in Allinone with Packstack, OVS is installed by the setting that OVS works with OVS Mechanical driver of ML2 plugin.
So, cancel the setting once, and then set up so that OVS can work with NEC plugin.

Delete br-tun.

```
# ovs-vsctl del-br br-tun
```

Clear br-int.

```
# ovs-vsctl del-br br-int
# ovs-vsctl add-br br-int
```

Connect Open vSwitch with PFC.

```
# ovs-vsctl set-controller br-int tcp:[IP address of PFC that is the connection
destination of SecureChannel]
```

Add the port for OpenFlow network.

```
# ovs-vsctl add-port br-int ens2f1
```

Clear data flow of br-int.

```
# ovs-ofctl del-flows br-int
```

Sample

```
# ovs-vsctl show
8eb7fedb-a2dd-441e-9c4e-77873186f62b
    Bridge br-int
        Controller "tcp:192.168.100.184"
            is_connected: true
        Port "ens2f1"
            Interface "ens2f1"
        Port br-int
            Interface br-int
                type: internal
    ovs_version: "2.1.3"
```

4.2.5. Start services

Start the services of Neutron Server, NEC Plugin.

```
# systemctl start neutron-nec-agent.service
# systemctl enable neutron-nec-agent.service
```

Chapter 5 Setting physical network

5.1. Setting PFS

5.1.1. Log in

Use log in ID as "operator" and log in by telnet or serial console.

5.1.2. RSI settings

Do the setting to run PFS as RSI (Real Switch Instance).
For details, refer to "3.2.2 Initial Configuration of PF52xx" of "PF6800 Ver. 6.1 Configuration Guide".

5.1.3. Interface settings

Do the setting of interfaces to connect nodes. Ports to connect OpenStack server have to be set for internal port. Ports to connect physical machine or legacy network have to be set for external port.
For details of internal port and external port, refer to "3.1.2.1 Port Types" of "PF6800 Ver. 6.1 Configuration Guide".
In the case of using current version of NEC Plugin, all data on OpenFlow network is communicated on native VLAN. Therefore, internal ports need only native VLAN, and external ports do not have to be set the access VLAN.
For details of interface settings, refer to "3.2.6 VLAN Configuration of Internal Ports" and "3.2.7 VLAN Configuration of External Ports" of the above guide.

Chapter 6 Setting OpenFlow1.3

This chapter describes settings of OpenFlow1.3.

6.1. Setting OpenStack

Run the following command so that br-int will run in OpenFlow1.3.

```
# ovs-vsctl set bridge br-int protocols=OpenFlow13
```

■NOTE:■ Settings are necessary in both Controller node and Compute node.

6.2. Setting PFC

Integrate the following setting into the configuration.

```
  PFC# show running-configuration
Date: 2015-01-21 15:33:53 JST
!! Commit number: 3212
!! Commit date: 2015-01-21 13:34:39
!! Commit application: PFC-API
system
cli
network-default {
  openflow-version 1.3
}
real-network {
  ofs-default-domain
  flow-entry-list MIRROR_DHCP {
    sequence-number 1 {
      mac-destination-address 0100.0000.0000 wildcard feff.ffff.ffff
      mac-ether-type 0x800
      ip-protocol 17
      l4-destination-port 67
      l4-source-port 68
    }
  }
  flow-entry-list MIRROR_IPv6 {
    sequence-number 1 {
      mac-destination-address 0100.0000.0000 wildcard feff.ffff.ffff
      mac-ether-type 0x86dd
      ip-protocol 58
    }
  }
  mirror-entry 1 flow-entry-list MIRROR_DHCP
  mirror-entry 2 flow-entry-list MIRROR_IPv6
}
```

■NOTE:■ For details of this setting, refer to 4.19 Training Terminal Positions and Configuring DHCP Packets and 4.21 Enabling the OpenFlow 1.3 Function in PF6800 Ver. 6.1 Configuration Guide.

6.3. Setting PFS

Change the protocol-version to 04, which is the same meaning as OpenFlow1.3, on PFS.

```
  pfs# configure
pfs(config)# openflow openflow-id 1
pfs(config-of)# protocol-version 04
!pfs(config-of)# save
pfs(config-of)#
```

Chapter 7 Setting virtual network

7.1. Creating user and tenant

7.1.1. Applying the administrator information into the operation environment

By the following command, make the operation environment read the administrator information file that is under home directory.

```
$ source ~/keystonerc_admin
```

7.1.2. Create general user

Create general user using the command given below. In the example the user name is taken as "UserA" and the password is taken as "secret".

```
$ keystone user-create --name UserA --pass secret
```

7.1.3. Create the role

Create the role using following command. The role name taken in the example is "user".

```
$ keystone role-create --name user
```

7.1.4. Create tenant

Create the tenant using following command. The tenant name taken in the example is "TenantA".

```
$ keystone tenant-create --name TenantA
```

7.1.5. Add the user to tenant

Add the user in tenant by specifying created general user, role, various IDs of tenant as arguments. The command is given below.

```
$ keystone user-role-add --user [ID displayed in user-create] ¥
                         --role [ID displayed in role-create] ¥
                         --tenant_id [ID displayed in tenant-create]
```

7.1.6. Setting user information

To make the authentication as user simple, create keystonerc_UserA file under home directory of user. Contents of the file are as below.

```
export OS_USERNAME=UserA
export OS_TENANT=NAME=TenantA
export OS_PASSWORD=secret
export OS_AUTH_URL=http://[IP address of the machine on which keystone is running]:35357/v2.0/
export PS1='[¥u@¥h ¥W(keystone_UserA)]¥$ '
```

7.2. Network settings

Explanation about the Neutron network settings in the server on which OpenStack server is running is given.

7.2.1. Checking the tenant ID

Check the tenant ID for which network is to be created. (In this explanation TenantA is taken as target)

```
$ keystone tenant-list
+----------------------------------+----------+---------+
|                id                |   name   | enabled |
+----------------------------------+----------+---------+
| 10974985dd9c44738ac8f256b989930e | TenantA  |   True  |
| 50c63157e3bd450d8878dbc02a65fdca |  admin   |   True  |
| 1982b7039f4a4b879f1e104144cb515e | services |   True  |
+----------------------------------+----------+---------+
```

7.2.2. Checking the user ID

Check the user ID to be associated with the VM to be created. (In this explanation UserA is taken as target)

```
$ keystone user-list
+----------------------------------+------------+---------+------------------
----+
|                id                |    name    | enabled |        email
    |
+----------------------------------+------------+---------+------------------
----+
| 8e2edaf3223b4ccdb11891b6d05bfb27 |   UserA    |   True  |
    |
| 8c31061dc567485fbfd2efd2ec261f48 |   admin    |   True  |   admin@localhost
    |
| b3882074c2c14288aafa561e5b26bd96 | ceilometer |   True  | ceilometer@localh
ost |
| 69d6648fe4c247c59df04860f3ccd155 |   cinder   |   True  |   cinder@localhos
t   |
| 9b58328e5a8140718e50f9f05d31d274 |   glance   |   True  |   glance@localhos
t   |
| 37540a28b74d41dcafe899390d4b3bb7 |  neutron   |   True  |  neutron@localhos
t   |
| b70fe08b2ddf4d3683e8ae30089abf6e |    nova    |   True  |    nova@localhost
    |
+----------------------------------+------------+---------+------------------
----+
```

7.2.3. Applying the user information into the operation environment

Apply the contents of file set in "7.1.6 Setting user information" into operation environment.

```
$ source ~/keystonerc_UserA
```

7.2.4. Creating network

Create network using neutron net-create command.

```
$ neutron net-create net1
Created a new network:
+-----------------+--------------------------------------+
| Field           | Value                                |
+-----------------+--------------------------------------+
| admin_state_up  | True                                 |
| id              | a027c020-03b6-4c3d-bbf7-7a1c51987341 |
| name            | net1                                 |
| shared          | False                                |
| status          | ACTIVE                               |
| subnets         |                                      |
| tenant_id       | 10974985dd9c44738ac8f256b989930e     |
+-----------------+--------------------------------------+
```

7.2.5. Creating subnet

Crate the subnet for the abovementioned network using neutron subnet-create command.

```
$ neutron subnet-create net1 192.168.1.0/24 --gateway 192.168.1.254
Created a new subnet:
+------------------+------------------------------------------------------+
| Field            | Value                                                |
+------------------+------------------------------------------------------+
| allocation_pools | {"start": "192.168.1.1", "end": "192.168.1.253"}     |
| cidr             | 192.168.1.0/24                                       |
| dns_nameservers  |                                                      |
| enable_dhcp      | True                                                 |
| gateway_ip       | 192.168.1.254                                        |
| host_routes      |                                                      |
| id               | e8ce652f-e9bb-4327-9607-f024f9f619dd                 |
| ip_version       | 4                                                    |
| name             |                                                      |
| network_id       | a027c020-03b6-4c3d-bbf7-7a1c51987341                 |
| tenant_id        | 10974985dd9c44738ac8f256b989930e                     |
+------------------+------------------------------------------------------+
```

7.3. Starting the VM

Here the procedure to connect created VM to Neutron network is explained.

7.3.1. Starting the VM and connecting to the Neutron Network

■CHECK:■

Before starting an instance, VM image has to be registered. Please prepare the VM image from which you want to start an instance. Then register it. The following example assumes that you registered the VM image named RHEL6.5 which suit the flavor of m1.small.

Start an instance of the registered VM image by using the nova boot command (in the following example, the instance name is TenantA_VM1). At this time, the connection to the network specified by net-id of the --nic option is also performed at the same time.

```
$ nova boot --image RHEL6.5 --flavor m1.small ¥
  --nic net-id=a027c020-03b6-4c3d-bbf7-7a1c51987341 TenantA_VM1
```

7.3.2. Checking the Connection to the Network

Check that the VM and network are associated using nova list command.

```
$ nova list
+--------------------------------------+------------+--------+------------+-
------------+------------------+
| ID                                   | Name       | Status | Task State |
Power State | Networks         |
+--------------------------------------+------------+--------+------------+-
------------+------------------+
| 4504ad9c-6467-4fb5-8255-3f4550d9e150 | TenantA_VM1 | ACTIVE | None       |
Running     | net1=192.168.1.1 |
+--------------------------------------+------------+--------+------------+-
------------+------------------+
```

7.4. Creating port for connecting physical machine

In order to connect a physical machine directly to OpenStack network, or to connect legacy network with OpenStack network, create a neutron port associated with PFS port which is for connecting physical machine or legacy network.

7.4.1. Checking the connection destination port

Check the datapath-id and port number of connection destination port by running show topology detail command of real-network mode on PFC shell.

7.4.2. Creating Neutron Port

■CHECK:■ For the following operation, Administrator privilege is necessary. Refer to "7.1.1 Applying the administrator information into the operation environment" and apply the user's information with administrator privilege to the operating environment. Then execute the operation.

Create Neutron port using neutron port-create command specifying datapath-id and port number of the connection destination port.

```
$ neutron port-create net1 --name PORT1 ¥
--binding:profile type=dict datapath_id=0x0000000000000004,port_no=7
Created a new port:
+-----------------------+--------------------------------------------------
```

```
--------------------------------+
| Field                  | Value
                                |
+----------------------+-------------------------------------------------
--------------------------------+
| admin_state_up         | True
                                |
| allowed_address_pairs |
                                |
| binding:capabilities   | {"port_filter": true}
                                |
| binding:host_id        |
                                |
| binding:profile        | {"portinfo:datapath_id": "0x1", "portinfo:port_no":
 7}                             |
| binding:vif_type       | ovs
                                |
| device_id              |
                                |
| device_owner           |
                                |
| fixed_ips              | {"subnet_id": "e8ce652f-e9bb-4327-9607-f024f9f619dd
", "ip_address": "192.168.1.3"} |
| id                     | 148e3bd2-da1f-43fe-b80d-63476b5cf1a7
                                |
| mac_address            | 00:11:22:33:44:55
                                |
| name                   | PORT1
                                |
| network_id             | a027c020-03b6-4c3d-bbf7-7a1c51987341
                                |
| security_groups        | c71bbe1c-e001-4ea4-834e-262293a6ac5a
                                |
| status                 | ACTIVE
                                |
| tenant_id              | 10974985dd9c44738ac8f256b989930e
                                |
+----------------------+-------------------------------------------------
--------------------------------+
```

■CHECK:■	Because executing user was switched, specify the tenant to create the port on by using --tenant-id option.
■NOTE:■	In case of physical machine, --mac-address option is required. In case of legacy network, --mac-address option is not required.
■CHECK:■	After creating a port is completed by the procedure above, refer to "7.1.6 Setting user information", and apply the original user's information to the operating environment.

7.5. Setting router

7.5.1. Creating router

Router can be created using neutron router-create command.

```
$ neutron router-create router
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 2c80b50d-6ca4-4928-b5c4-4bfaad0c9d45 |
| name                  | router                               |
| provider              | openflow                             |
| status                | ACTIVE                               |
| tenant_id             | 10974985dd9c44738ac8f256b989930e     |
+-----------------------+--------------------------------------+
```

■CHECK:■	Specify the name of the router to be created just after router-create.
■NOTE:■	You can specify the provider of router by --provider option. About its parameter, refer to "4.1.4 Configure NEC Plugin".

7.5.2. Deleting router

Router can be deleted by using neutron router-delete command.

```
$ neutron router-delete 2c80b50d-6ca4-4928-b5c4-4bfaad0c9d45
```

■CHECK:■	Specify the ID of the router to be deleted just after router-delete. Router ID can be checked by neutron router-list command.

7.5.3. Adding router interface

Router interface can be added using neutron router-interface-add command.

```
$ neutron router-interface-add 2c80b50d-6ca4-4928-b5c4-4bfaad0c9d45 ¥
  e8ce652f-e9bb-4327-9607-f024f9f619dd
```

■CHECK:■	Specify in the sequence of ID of target router, ID of target subnet just after the router-interface-add. Router ID can be checked by neutron router-list command and subnet ID can be checked by neutron subnet-list command.

7.5.4. Deleting router interface

Interface of router can be deleted using neutron router-interface-delete command.

```
$ neutron router-interface-delete 2c80b50d-6ca4-4928-b5c4-4bfaad0c9d45 ¥
  e8ce652f-e9bb-4327-9607-f024f9f619dd
```

■CHECK:■ Specify in the sequence of ID of target router, ID of target subnet just after the router-interface-delete. Router ID can be checked by neutron router-list command and subnet ID can be checked by neutron subnet-list command.

7.6. Setting static route of router

■CHECK:■ Static route can be set when the provider of router is openflow.

7.6.1. Adding static route of router

Static route can be added using neutron router-update command.

```
$ neutron router-update router --routes list=true type=dict ¥
  destination=192.168.0.0/24,nexthop=30.0.1.254
```

■CHECK:■ Specify the router name just after router-update. Set the network address of destination network to “destination”, and set the gateway IP address to “nexthop”.

When separate route information is to be added for the router for which route information is already added, it is required to specify already set route information once again. (the highlighted part in the below example is the information that was already set in the above example).

```
$ neutron router-update router --routes list=true type=dict ¥
  destination=192.168.0.0/24,nexthop=30.0.1.254 ¥
  destination=192.168.0.0/24,nexthop=30.0.2.254
```

7.6.2. Deletion of static route of router

Route of the router can be deleted using neutron router-update command.

Information of all the routes can be deleted by specifying --no-routes option.

```
$ neutron router-update router --routes action=clear
```

Information of some routes can be deleted by re specifying the router information that is not in scope of deletion using --route option. (In the below example, highlighted part of "7.6.1 Adding static route of router") is deleted.

```
$ neutron router-update router --routes list=true type=dict ¥
  destination=192.168.0.0/24,nexthop=30.0.2.254
```

7.7. Connecting to external network

■CHECK:■	This section describes the steps of connecting to external network by using routers the provider of which is openflow. For l3-agent, refer to documents of the OpenStack.
■CHECK:■	For the following operation, Administrator privilege is necessary. Refer to "7.1.1 Applying the administrator information into the operation environment" and apply the user's information with administrator privilege to the operating environment. Then execute the operation.
■CHECK:■	Because executing user is switched, specify the tenant to each command by using --tenant-id option.

7.7.1. Creating external network

Create external network using neutron net-create command.

```
$ neutron net-create --router:external=True ext_net ¥
  --tenant-id 10974985dd9c44738ac8f256b989930e
Created a new network:
+-----------------+--------------------------------------+
| Field           | Value                                |
+-----------------+--------------------------------------+
| admin_state_up  | True                                 |
| id              | d9d3b1d6-f4ca-4595-9513-f99dea8eb860 |
| name            | ext_net                              |
| router:external | True                                 |
| shared          | False                                |
| status          | ACTIVE                               |
| subnets         |                                      |
| tenant_id       | 10974985dd9c44738ac8f256b989930e     |
+-----------------+--------------------------------------+
```

■NOTE:■	--router:external=True option shows that is external network.

7.7.2. Creating subnet for external network

Create subnet for the external network using neutron subnet-create command.

```
$ neutron subnet-create ext_net 192.168.100.0/24 --gateway 192.168.100.254 ¥
  --disable-dhcp --allocation-pool start=192.168.100.200,end=192.168.100.201 ¥
  --tenant-id 10974985dd9c44738ac8f256b989930e
Created a new subnet:
+-------------------+------------------------------------------------------+
| Field             | Value                                                |
+-------------------+------------------------------------------------------+
| allocation_pools  | {"start": "192.168.100.200", "end": "192.168.100.201"} |
| cidr              | 192.168.100.0/24                                     |
| dns_nameservers   |                                                      |
| enable_dhcp       | False                                                |
| gateway_ip        | 192.168.100.254                                      |
| host_routes       |                                                      |
| id                | 791b5995-fab8-4585-91f7-36abb6544662                 |
| ip_version        | 4                                                    |
| name              |                                                      |
| network_id        | d9d3b1d6-f4ca-4595-9513-f99dea8eb860                 |
| tenant_id         | 10974985dd9c44738ac8f256b989930e                     |
+-------------------+------------------------------------------------------+
```

■NOTE:■

Specify a gateway address of a legacy network with --gateway option.
--disable-dhcp option disables DHCP of Neutron DHCP Agent to avoid overlapping with DHCP of the external network.
Specify a range of IP allocation pool of external network for OpenStack by using --allocation-pool option. At least, two IP addresses are necessary for OpenStack (One is for the port connecting to the external network, the others are for an interface of vRouters).

7.7.3. Setting gateway to router

Set the gateway to router using neutron router-gateway-set command.

```
$ neutron router-gateway-set 2c80b50d-6ca4-4928-b5c4-4bfaad0c9d45 ¥
  d9d3b1d6-f4ca-4595-9513-f99dea8eb860 ¥
  --tenant-id 10974985dd9c44738ac8f256b989930e
```

■CHECK:■

Specify in the sequence of target router ID, external network ID just after the router-gateway-set. Router ID can be checked by neutron router-list command and external network ID can be checked by neutron net-external-list command.

7.7.4. Creating port for connecting to external network

Following the procedure of "7.4 Creating port for connecting physical machine", create a Neutron port with specifying the external network as a target.

7.8. Setting the Security Group

■CHECK:■ The security group is prepared for OpenStack as a function for protecting the in formation in the cloud from a range of security threats, such as wiretapping and illegal access. By adding rules to the security group, you can specify the commu nication type for which port passage is allowed on the virtual network, as well a s the direction of the communication.

7.8.1. Creating the Security Group

Create a security group using neutron security-group-create command.

```
$ neutron security-group-create group1
Created a new security_group:
+----------------------+-----------------------------------------------+
| Field                | Value                                         |
+----------------------+-----------------------------------------------+
| description          |                                               |
| id                   | 57fef81b-da7a-453a-a9f0-a55410573dca          |
| name                 | group1                                        |
| security_group_rules | {"direction": "egress", "ethertype": "IPv4"}  |
|                      | {"direction": "egress", "ethertype": "IPv6"}  |
| tenant_id            | 10974985dd9c44738ac8f256b989930e              |
+----------------------+-----------------------------------------------+
```

■CHECK:■ In actual execution results, more detailed information is output to security_group_rules. However it is omitted here.

7.8.2. Creating the Security Group Rule

Add a security rule group using neutron security-group-rule-create command.

```
$ neutron security-group-rule-create --direction ingress --remote-group-id
group1 group1
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | d3f9b8e3-c692-49b6-b1a7-735b29676e13 |
```

```
| port_range_max    |                                      |
| port_range_min    |                                      |
| protocol          |                                      |
| remote_group_id   | 57fef81b-da7a-453a-a9f0-a55410573dca |
| remote_ip_prefix  |                                      |
| security_group_id | 57fef81b-da7a-453a-a9f0-a55410573dca |
| tenant_id         | 10974985dd9c44738ac8f256b989930e     |
+-------------------+--------------------------------------+
```

7.8.3. Deleting a Security Rule Group

Delete a security rule group using neutron security-group-rule-delete command.

```
$ neutron security-group-rule-delete d3f9b8e3-c692-49b6-b1a7-735b29676e13
Deleted security_group_rule: d3f9b8e3-c692-49b6-b1a7-735b29676e13
```

7.8.4. Deleting a Security Group

Delete a security rule group using neutron security-group-delete command.

```
$ neutron security-group-delete 57fef81b-da7a-453a-a9f0-a55410573dca
Deleted security_group: 57fef81b-da7a-453a-a9f0-a55410573dca
```

7.9. Setting the Packet Filter

Similar to Security Group, Packet Filter controls packets that go through the virtual networks. Packet Filter is implemented with OpenFlow, and it can reduce the traffic in the virtual networks because packets are evaluated at ingress openflow switch. The following describes how to set up the packet filter.

Arguments of packet filter

Following argumets such as match conditions and action (deny, allow) are required to add/update packet filters.

Option	Description	Valid Value
--in-port	Port to which a filter is applied	Created port ID or name
--src-mac	Source MAC address	Format of hh:hh:hh:hh:hh:hh (h means hexadecimal number)
--dst-mac	Destination MAC address	
--eth-type	EtherType	Hexadecimal number prefixed with "0x" (0x0000-0xFFFF)
--protocol	Communication protocol	{tcp | udp | icmp | arp} (lower case) Or decimal number which indicates the protocol number (0-255)
--src-cidr	Source network	IPv4 dotted-quad string format/prefix length

--dst-cidr	Destination network	Example: 192.168.1.1/24
--src-port	Source port number of TCP/UDP	Decimal number (0-65535)
--dst-port	Destination port number of TCP/UDP	
--priority	Priority	Decimal number (1-32766) 1 has the highest priority, and 32766 is lowest.
--action	Communication control	{allow | drop} (lower case) allow: permit the communication drop: disconnect the communication

7.9.1. Adding a Packet Filter

Add a packet filter by "neutron nec-packet-filter-create" command.

```
$ neutron nec-packet-filter-create ¥
    --name filter1 ¥
    --in-port ac9fd4c5-2116-4d42-82b4-b099d034d55b ¥
    --src-mac 00:11:22:33:44:55 ¥
    --dst-mac aa:bb:cc:dd:ee:ff ¥
    --eth-type 0x0800 ¥
    --protocol tcp ¥
    --src-cidr 192.168.0.0/24 ¥
    --dst-cidr 172.16.0.0/16 ¥
    --src-port 50000 ¥
    --dst-port 30000 ¥
    --priority 20000 ¥
    --action drop ¥
    net1
Created a new packet_filter:
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| action         | drop                                 |
| admin_state_up | True                                 |
| dst_cidr       | 172.16.0.0/16                        |
| dst_mac        | aa:bb:cc:dd:ee:ff                    |
| dst_port       | 30000                                |
| eth_type       | 2048                                 |
| id             | d0023bf7-44e6-4898-a82b-e014fed7fa3b |
| in_port        | ac9fd4c5-2116-4d42-82b4-b099d034d55b |
| name           | filter1                              |
| network_id     | 8ba94b17-c9c9-4d3c-889d-045fab221519 |
| priority       | 20000                                |
| protocol       | tcp                                  |
| src_cidr       | 192.168.0.0/24                       |
| src_mac        | 00:11:22:33:44:55                    |
| src_port       | 50000                                |
| status         | ACTIVE                               |
| tenant_id      | f7c095cbed5a4a3c906b51909297a4ea     |
+----------------+--------------------------------------+
```

7.9.2. Acquiring the Packet Filter List

Get packet filters by "neutron nec-packet-filter-list" command.

```
$ neutron nec-packet-filter-list
+--------------------------------------+---------+--------+----------+-------
----------------------------------------+
| id                                   | name    | action | priority | summary
|
+--------------------------------------+---------+--------+----------+-------
----------------------------------------+
| d0023bf7-44e6-4898-a82b-e014fed7fa3b | filter1 | drop   |    20000 | protocol:
TCP, eth_type: 2048                     |
|                                      |         |        |          | network:
8ba94b17-c9c9-4d3c-889d-045fab221519    |
|                                      |         |        |          | in_port:
ac9fd4c5-2116-4d42-82b4-b099d034d55b    |
|                                      |         |        |          | source:
00:11:22:33:44:55 192.168.0.0/24 50000  |
|                                      |         |        |          | destination:
aa:bb:cc:dd:ee:ff 172.16.0.0/16 30000 |
+--------------------------------------+---------+--------+----------+-------
----------------------------------------+
```

7.9.3. Acquiring the Packet Filter Information

Get a packet filter by "neutron nec-packet-filter-show" command.

```
$ neutron nec-packet-filter-show filter1
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| action         | drop                                 |
| admin_state_up | True                                 |
| dst_cidr       | 172.16.0.0/16                        |
| dst_mac        | aa:bb:cc:dd:ee:ff                    |
| dst_port       | 30000                                |
| eth_type       | 2048                                 |
| id             | d0023bf7-44e6-4898-a82b-e014fed7fa3b |
| in_port        | ac9fd4c5-2116-4d42-82b4-b099d034d55b |
| name           | filter1                              |
| network_id     | 8ba94b17-c9c9-4d3c-889d-045fab221519 |
| priority       | 20000                                |
| protocol       | tcp                                  |
| src_cidr       | 192.168.0.0/24                       |
| src_mac        | 00:11:22:33:44:55                    |
| src_port       | 50000                                |
| status         | ACTIVE                               |
| tenant_id      | f7c095cbed5a4a3c906b51909297a4ea     |
+----------------+--------------------------------------+
```

7.9.4. Updating the Packet Filter

Update a packet filter by "neutron nec-packet-filter-update" command.

```
$ neutron nec-packet-filter-update filter1 --dst-port 40000
Updated packet_filter: filter1
```

■CHECK:■	All options can be omitted.
■NOTE:■	Priority and Action cannot be updated.

7.9.5. Deleting the Packet Filter

Delete a packet filter by "neutron nec-packet-filter-delete" command.

```
$ neutron nec-packet-filter-delete filter1
Deleted packet_filter: filter1
```

Chapter 8 Notes and restrictions

8.1. Restriction on broadcast domain

Broadcast domains are not devided by VTN. Hence the broadcast packets/frames, such as ARP or dhcp discover, invade other VTNs. This is because current OVS does not support port-based VLAN when controlled by OpenFlow.

8.2. Error on multiple requests

PFC WebAPI server executes API requests serially. If an API is called during executing another request, PFC returns 503 with setting retry-after header for the latter one. NEC plugin retries API call by receiving this error, and if the retry-count specified in NEC plugin configuration file is exhausted, API fails.

In this case, please try the same operation again later.

8.3. Restriction on connecting vRouter and shared network

Neutron router created by 'openflow' router provider can not connect to shared networks which are in different projects

8.4. Restriction on Packet Filter

STATUS of Packet Filter does not become ACTIVE when Packet Filter is set to a port of ‘openflow’ based router.

Chapter 9 Appendix

9.1. Resources on PFC created by OpenStack operations.

The following describes the resources set in PFC by typical OpenStack operations.

9.1.1. Creating a project

OpenStack project is mapped to VTN on PFC. VTN name is generated by removing hyphens and 13th digit (This is always '4'.) from project UUID.

■NOTE:■ VTN is created just before the first vBridge in the VTN is created.

9.1.2. Creating a network

Neutron network is mapped to vBridge on PFC. vBridge name is generated by combining prefix 'os_vbr_' + *serial number*. *serial number* is assigned to be unique vBridge name among the VTN. Additionally, network UUID and network name are set to vBridge description.

Following shows an example.

OpenStack Project name/UUID, and Neutron network name/UUID

Project name：project / b2033859336f495c84371f0031de3148

Network name：network/ becfdb16-023a-46cd-9f1f-9e02a01c56cb

Resorces on PFC：

```
vtn b2033859336f95c84371f0031de3148 {
  vbridge os_vbr_1 {
    description
ID_becfdb16_023a_46cd_9f1f_9e02a01c56cb_Name_network_at_Neutron_
  }
}
```

9.1.3. Creating an VM

By creating a VM (and connect it to existing neutron network), vExternal/ofs-map/vExternal interface/vBridge interface/vLink are created.

vExternal name is generated by combining prefix 'os_vex_' + *serial number*.
vExternal interface name is generated by combining prefix 'os_if_' + *serial number*.
vBridge interface name is generated by combining prefix 'os_if_' + *serial number*.
vLink name is generated by combining prefix 'os_vlk_' + *serial number*.
Common *serial number* is assigned for above four resources.

If a VM is connected to shared network in different project, these resources are created in the VTN to which the shared network (vBridge) belongs.

The following shows an example：

```
vtn b2033859336f95c84371f0031de3148 {
  vbridge os_vbr_1 {
    description
ID_becfdb16_023a_46cd_9f1f_9e02a01c56cb_Name_network_at_Neutron_
    interface os_if_1
    interface os_if_2
  }
  vexternal os_vex_1 {
    ofs-map ofs-datapath-id 0000-0025-5cbd-26b9 ofs-port tap7ce76c3a-7c
    interface os_if_1
  }
  vexternal os_vex_2 {
    ofs-map ofs-datapath-id 0000-0025-5cbd-26cd ofs-port qvo22133717-8a
    interface os_if_2
  }
  vlink os_vlk_1 {
    vtn link vbridge os_vbr_1 interface os_if_1 vtnnode os_vex_1 interface
os_if_1
  }
  vlink os_vlk_2 {
    vtn link vbridge os_vbr_1 interface os_if_2 vtnnode os_vex_2 interface
os_if_2
  }
}
```

9.1.4. Creating a router

Neutron router created by 'openflow' router provider, is mapped to vRouter on PFC. vRouter name is 'os_vrt'.

The following shows an example：

```
vtn b2033859336f95c84371f0031de3148 {
  vrouter os_vrt
}
```

9.1.5. Adding router interface

By adding a router interface to existing 'openflow' based router, a vrouter interface is created.

Additionally, mac-address and ip address are registered.

The following shows an example :

```
  vrouter os_vrt {
    interface os_if_3 {
      mac-address fa16.3e97.33e0
      ip address 172.16.0.1/16
    }
  }
```

9.1.6. Setting static route by using the router-update command

By executing 'router-update' command to 'openflow' based router, static routes are registered.

The following shows an example :

```
  vrouter os_vrt {
    interface os_if_3 {
      mac-address fa16.3e97.33e0
      ip address 172.16.0.1/16
    }
    ip route 20.9.1.0/24 next-hop-address 10.3.3.7
  }
```

9.1.7. Executing the packet filter command

By creating a packet filter, flow-list and flow-filter are registered on PFC.

In flow-list, match conditions such as 'mac-destination-address', 'mac-source-address', 'mac-ether-type', and so on, are registered in fixed sequence number '1'. Name of flow-list is generated by combining prefix 'os_f' + *action type* + priority in hexadecimal + '_' + *serial number*. For *action type*, 'p' is set for 'pass' action, and 'd' is set for 'drop' action. Supposing that there is a flow-list named 'os_fd4E20_1', it means that action is 'drop' and priority is 20000.

Flow-filters are registered beneath target vBidge interfaces. Value of sequence number is the same as priority.

The following shows an example :

```
virtual-network {
  flow-list os_fd4E20_1 type ipv4 {
    sequence-number 1 {
      mac-destination-address aabb.ccdd.eeff
      mac-source-address 0011.2233.4455
      mac-ether-type 0x800
      ip-destination-address 172.16.0.0/16
      ip-source-address 192.168.0.0/24
```

```
      ip-protocol 6
      l4-destination-port 30000
      l4-source-port 50000
    }
  }
  vtn a21d83d423df618b25d2b1c31468784 {
    description demo
    vbridge os_vbr_1 controller pfc01 {
      description  "ID=dba9a190-d5b3-4eaa-9964-32b95d9e91de  Name=network  at
Neutron."
      interface os_if_1 {
        port-map logical-port PP-0000-1921-6810-0181-tapc8f2c69e-61
      }
      interface os_if_2 {
        port-map logical-port PP-0000-1921-6810-0181-qvo8889dbe0-c1
        flow-filter in {
          sequence-number 20000 {
            match flow-list os_fd4E20_1
            action drop
          }
        }
      }
    }
  }
}
```

PF6800 Ver. 6.1

Neutron NEC OpenFlow Plugin Configuration Guide for

Red Hat Enterprise Linux OpenStack Platform 6.0

First Edition, April 2015