OpenStack Networking API in CEE in Single Server Deployment

Cloud Execution Environment

INTERWORK DESCRIPTION

7/155 19-CRA 119 1862/5-V2 Uen A

Contents

1 Introduction

This document describes the use of the Application Programming Interface (API) for networking in the Cloud Execution Environment (CEE). The API is based on the OpenStack networking component (Neutron).

1.1 API Version

The CEE Networking API is based on OpenStack Networking API v2.

1.2 Document References

This section contains the official OpenStack API reference.

1.2.1 API Design Base Reference

For the description of the API operations and extensions of networking, refer to the sections ‘‘Networking API v2.0’’ and ‘‘Networking API v2.0 extensions’’ in the *OpenStack API Complete Reference*.

This is a stored copy of the OpenStack API Reference document version that was the base for the development of this version of CEE.

Note: It is possible that the date on the front page differs from the revision date in this document. The front page shows the date on which the document was generated.

2 Prerequisites

The following sections list the prerequisites.

2.1 CEE Networking Configurations

CEE networking can be configured during the installation of the CEE on single server. For details about the configurations, refer to the document *Configuration File Guide*.

2.2 Segmentation IDs

Neutron uses a range of segmentation IDs for internal tenant separation. This range must be defined during the installation of the CEE.

Border Gateway (BGW) to CEE region connectivity also requires Virtual LANs (VLANs). The Cloud manager has to manage a range of IDs to be used for this. This range must not overlap with the range used internally by Neutron.

Table 1 VLAN Allocation Example

Name	Range	Description
Default	50-3581	Used by Neutron for tenant separation
BGW	3582-4094	Used for Neutron to BGW connectivity

Note: The table is an example, it does not mandate specific or default values. The values must be configured before the CEE region is installed. The configuration of values is described in the *Configuration File Guide*.

Each Neutron network requires one segmentation ID. Segmentation IDs have to be unique, the same ID cannot be used on several Neutron networks. Segmentation IDs on Neutron networks are static and cannot be changed after creation of the Neutron network.

3 Supported Operations

The following sections contain information about the API operations and API extensions in CEE.

3.1 Basic OpenStack Operations

For the detailed description of basic Networking API operations, refer to the section ‘‘Networking API v2.0’’ in the *OpenStack API Complete Reference*.

3.1.1 Limitations

For the CEE-specific limitations and recommendations, see Section 5 on page 6.

3.2 OpenStack Extensions

This section contains information about which Networking API extensions are supported.

- Agent Management Extension

  The following agents are supported:

  - `neutron-dhcp-agent`
  - `neutron-openvswitch-agent`

- Agent Schedulers

  The following agents are supported:

  - DHCP Agent Scheduler
  - Device Agent Scheduler

- List Extensions

  Note: Supported, but the reply contains unsupported operations.

- Networks multiple provider extension (networks)
- Quotas extension (quotas)

4 Deviations

This chapter describes the deviations between vanilla OpenStack networking and CEE networking.

4.1 Partly Supported Operations

The following operations are partly supported in CEE Networking:

Port binding extended attributes (ports)
Only the `host_id` attribute is supported.

External networks (external-net)
Support for dynamic creation of external networks. It allows the addition of external networks towards additional BGWs in runtime.

Operations for subnets
Only IPv4 is supported.

Network provider extended attributes (networks)
the `vlan` network type is supported.

Administrative State Down (ports)
Administration state `down` is not supported for trunk ports and their subports.
If the administration state of a trunk port is set to `down` at the CIC, it will not take effect to the trunk port and its subports at the relevant Compute node, although Neutron will show that the trunk port and its subports are administratively down.

4.2 Not Supported Operations

The following operations are not supported in CEE Networking:

- Configurable external gateway modes
- Extra DHCP options (`extra-dhcp-opt`)
- The ExtraRoute Extension
- Firewall as a Service (FwaaS)
- Load Balancer as a Service (LBaaS)
- Metering labels and rules

- Security groups and rules (`security-groups`)
- Virtual Private Network as a Service (VPNaaS)

5 Limitations and Recommendations

This section lists the limitations and recommendations for CEE Networking.

5.1 Limitations

This section contains the limitations of CEE Networking.

5.1.1 Segmentation IDs

The total number of segmentation IDs is limited to 4094, see Section 2.2 on page 2.

5.1.2 Deleting Trunkport and Subports

If a trunk port is deleted before the associated subports have been deleted, there are remains left in Neutron that prevent using the same IP address with a different MAC number. If a new trunk port and subports are created with the same IP address, this can prevent VMs on the ports to get an IP address.

To delete a trunk port with subports, do the following:

1. Delete the related subports.
2. Delete the trunk port.

5.2 Recommendations

Consider the limitations of the BGW solution, for example, IP and VLAN ranges.

6 Concepts and Use Cases

The following sections describe the various concepts and use cases that are connected to the CEE Networking API.

6.1 General Terms

Neutron Network
L2 broadcast domain

Neutron Subnet Definition of a subnet. It s attached to a Neutron network. Used to configure the Neutron DHCP service.

6.2 IP Address Management

The following three methods are available for IP address management:

- Application handles the IP addresses on its own and do not involve Neutron.

  Note: IP addresses can be privately handled only if ARP spoofing is switched off. It is not recommended to switch off ARP spoofing, as it is a security feature of the system.

  No Neutron DHCP service is used. Only L2 services from Neutron can be used.

  Note: Nova requires a subnet attached to a Neutron network in order to start VMs on the network. User is required to create a dummy subnet with DHCP disabled as a workaround.

- Application uses Neutron for IP address management.

  This can be done in several ways: the application can specify IP addresses per ports or only specify IP subnets and let Neutron allocate the addresses.

- A mix of the above two.

  In this case, it is the responsibility of the application to align the configuration view of the application with the configuration view of Neutron.

IP addresses in Neutron can be reused on different Neutron networks. IP addresses known by Neutron must have matching Neutron subnets.

To manually assign an IP address to a port use the `fixed_ips` attribute in the Neutron port when creating the Neutron port.

The `fixed_ips` attribute includes `ip_address` and `subnet_id`. Please refer to the section ‘‘Networking API v2.0’’ in the *OpenStack API Complete Reference*.

6.2.1 Neutron DHCP

Neutron DHCP can be enabled or disabled per subnet.

This is done by using the `enable_dhcp` attribute on the Neutron subnet.

```
enable_dhcp <True/False>
```

6.3 MAC Address Management

The following two methods are available for MAC address management:

- The application handles MAC addresses on its own, and provides the MAC address to Neutron when creating the Neutron port.
- The application allows Neutron to allocate a MAC address when creating the Neutron port.

MAC addresses have to be unique per Neutron network, but can be reused on different Neutron networks.

To manually assign a MAC address use the `mac_address` attribute in the Neutron port when creating the Neutron port, refer to the sections ‘‘Networking API v2.0’’ and ‘‘Networking API v2.0 extensions’’ in the *OpenStack API Complete Reference*.

Neutron-allocated MAC addresses have a three byte fixed prefix. The rest will be a random number unique per Neutron network. Refer to *Configuration File Guide*, for the prefix used in the Neutron configuration file.

Note: It is advisable to use the local administered MAC ranges.

6.4 IPv6

Neutron L3 Networks are not supported for IPv6, thus no Network Routing or Address Management for IPv6 is supported.

6.5 Internal Neutron Network

This section describes how to connect VMs using internal L2. Create Neutron network, create Neutron ports on Neutron network, and then create VMs using those Neutron ports.

1. Create Neutron network

For the procedure of creating a Neutron network, refer to the sections ‘‘Networking API v2.0’’ and ‘‘Networking API v2.0 extensions’’ in the *OpenStack API Complete Reference*.

2. Create Neutron subnet on Neutron network

   Nova requires a subnet attached to a Neutron network in order to start VMs on the network. User is required to create a dummy subnet with DHCP disabled as a workaround.

   For the procedure of creating a Neutron subnet on Neutron network, refer to the section ‘‘Networking API v2.0’’ in the *OpenStack API Complete Reference*.

3. Create Neutron ports on Neutron network

   For the procedure of creating Neutron ports on a Neutron network, refer to the sections ‘‘Networking API v2.0’’ and ‘‘Networking API v2.0 extensions’’ in the*OpenStack API Complete Reference*.

4. Create VMs using Neutron ports

6.6 Internal Neutron Network with Neutron IP Address Management

This chapter describes how to connect VMs using internal L3.

1. Create Neutron network

   For the procedure of creating a Neutron network, refer to the section ‘‘Networking API v2.0’’ in the *OpenStack API Complete Reference*.

2. Create Neutron subnet on Neutron network

   For the procedure of creating a Neutron subnet on Neutron network, refer to the section ‘‘Networking API v2.0’’ in the *OpenStack API Complete Reference*.

3. Create Neutron ports on Neutron network

   For the procedure of creating Neutron ports on a Neutron network, refer to the sections ‘‘Networking API v2.0’’ and ‘‘Networking API v2.0 extensions’’ in the *OpenStack API Complete Reference*.

4. Create VMs using Neutron ports.

6.7 L2 Connection to BGW

The following subsections describe how to connect VMs to BGW using an L2 network.

The L2 BGW connection is shown in Figure 1.

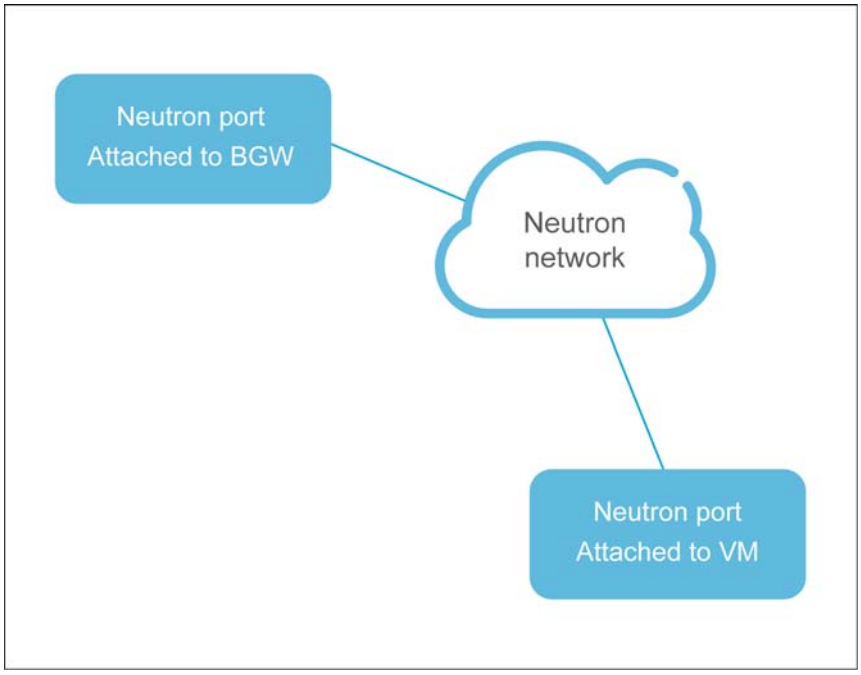

Figure 1 L2 BGW Connection

6.7.1 Create Neutron Network

Create a Neutron network using the `provider` network extension (`provider ID`). The Cloud manager is recommended for the management of these segmentation IDs. The IDs must be picked from the BGW pool, see Section 2.2 on page 2.

For the procedure of creating a Neutron network, refer to the section "Networking API v2.0" in the *OpenStack API Complete Reference*.

For the `provider` Extended Attributes for Networks, refer to the section "Networking API v2.0 extensions" in the *OpenStack API Complete Reference*.

`provider:segmentation_id`	*<VID>*
`provider:network_type`	`vlan`(1)
`provider:physical_network`	`default`

(1) Only the `vlan` network type is supported.

6.7.2 Create Neutron Subnet

Create Neutron subnet on Neutron network. A subnet is required regardless if IP address management is going to be handled by Neutron or not.

nova requires a subnet attached to a Neutron network in order to start VMs on the network. User is required to create a dummy subnet with DHCP disabled

as a workaround in cases when IP address management is handled by the application.

For the procedure of creating a Neutron subnet on Neutron network, refer to the section “Networking API v2.0” in the *OpenStack API Complete Reference*.

6.7.3 Create Neutron Ports Connected to BGW

1. Create two Neutron ports on the Neutron network.

   For the procedure of creating Neutron ports on a Neutron network, refer to the sections “Networking API v2.0” and “Networking API v2.0 extensions” in the *OpenStack API Complete Reference*.

2. Bind one Neutron port to BGW.

   For the procedure of binding extended attributes for ports, refer to the sections “Networking API v2.0” and “Networking API v2.0 extensions” in the *OpenStack API Complete Reference*

BGW-1:

`device_owner`	`baremetal:BGW-1`
`network_id`	*`<BGW Net id>`*
`binding:host_id`	`BGW-1`

6.7.4 Configure BGW

The traffic is configured to reach the BGW. The BGW must be configured to handle this traffic.

For the configuration of BGW for this segmentation ID, see Figure 2.

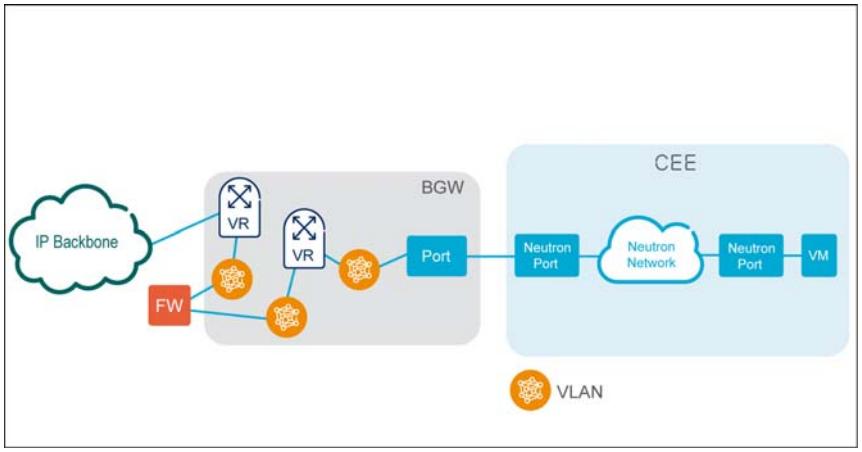

Figure 2 L2 BGW Configuration