**Provider Network Management**

Introduce provider network management for neutron.

/\* TO BE FILLED: business requirements \*/

This spec proposes an extension which exposes the management of provider networks to the API user. Previously, provider networks were nothing more than an entry in a config file that was statically defined. Now the user can provide management for them as they would do for tenant networks.

**Problem Description**

/\* TO BE FILLED: business use cases or user stories \*/

UseCase I.

UseCase II.

**Proposed Change**

**Data Model Impact**

The following new tables are added as part of the provider network management

feature::

CREATE TABLE providernets (

id CHAR(36) NOT NULL PRI KEY,

name VARCHAR(255) NOT NULL UNIQUE KEY,

description VARCHAR(255),

mtu INT NOT NULL,

status VARCHAR(16),

vlan\_transparent BOOT NOT NULL,

type ENUM(constants.PROVIDERNET\_TYPES) NOT NULL

);

CREATE TABLE providernet\_ranges (

id CHAR(36) NOT NULL PRI KEY,

tenant\_id VARCHAR(255),

name VARCHAR(255),

description VARCHAR(255),

shared BOOT NOT NULL,

minimum INT,

maximum INT,

providernet\_id CHAR(36),

CONSTRAINT `providernet\_id\_constraint` FOREIGN KEY (  
 `providernet\_id`) REFERENCES `providernets` (`id`)

);

CREATE TABLE providernet\_range\_vxlans (

id CHAR(36) NOT NULL PRI KEY,

group VARCHAR(255) NOT NULL,

port INT NOT NULL,

ttl INT NOT NULL,

providernet\_range\_id CHAR(36) NOT NULL,

CONSTRAINT `providernet\_range\_id\_constraint` FOREIGN KEY (  
 `providernet\_range\_id`) REFERENCES `providernet\_ranges` (`id`)

);

**Resource Extension**

The following new resources are being introduced and their attributes maps would be like:

.. code-block:: python

PROVIDERNET\_TYPE\_LIST = ['flat', 'vlan', 'vxlan', 'gre']

RESOURCE\_ATTRIBUTE\_MAPS = {

'providernets': {

'id': {'allow\_post': False, 'allow\_put': False,

'is\_visible': True},

'name': {'allow\_post': True, 'allow\_put': False,

'is\_visible': True,

'validate': {'type:not\_empty\_string':

db\_const.NAME\_FIELD\_SIZE}},

'type': {'allow\_post': True, 'allow\_put': False,

'validate': {'type:values': PROVIDERNET\_TYPE\_LIST},

'is\_visible': True},

'description': {'allow\_post': True, 'allow\_put': True,

'is\_visible': True,

'default': None},

'status': {'allow\_post': False, 'allow\_put': False,

'is\_visible': True},

'mtu': {'allow\_post': True, 'allow\_put': True,

'convert\_to': converters.convert\_to\_int,

'validate': {'type:range': n\_const.VALID\_MTU\_RANGE},

'default': n\_const.DEFAULT\_MTU,

'is\_visible': True},

'vlan\_transparent': {'allow\_post': True, 'allow\_put': True,

'convert\_to': converters.convert\_to\_boolean,

'default': False,

'is\_visible': True},

'tenant\_id': {'allow\_post': True, 'allow\_put': False,

'is\_visible': False,

'default': None},

'ranges': {'allow\_post': False, 'allow\_put': False,

'is\_visible': True},

},

‘providernet\_ranges': {

'id': {'allow\_post': False, 'allow\_put': False,

'is\_visible': True},

'name': {'allow\_post': True, 'allow\_put': False,

'is\_visible': True,

'default': None},

'description': {'allow\_post': True, 'allow\_put': True,

'is\_visible': True,

'default': None},

'shared': {'allow\_post': True, 'allow\_put': False,

'convert\_to': converters.convert\_to\_boolean,

'is\_visible': True,

'default': True},

'minimum': {'allow\_post': True, 'allow\_put': True,

'convert\_to': converters.convert\_to\_int,

'is\_visible': True},

'maximum': {'allow\_post': True, 'allow\_put': True,

'convert\_to': converters.convert\_to\_int,

'is\_visible': True},

'providernet\_id': {'allow\_post': True, 'allow\_put': False,

'is\_visible': True},

'providernet\_name': {'allow\_post': False, 'allow\_put': False,

'is\_visible': True},

'providernet\_type': {'allow\_post': False, 'allow\_put': False,

'is\_visible': True},

'tenant\_id': {'allow\_post': True, 'allow\_put': False,

'is\_visible': True,

'default': None},

'group': {'allow\_post': True, 'allow\_put': False,

'is\_visible': True,

'validate': {'type:ip\_mcast\_address': None},

'default': constants.ATTR\_NOT\_SPECIFIED},

'port': {'allow\_post': True, 'allow\_put': False,

'convert\_to': converters.convert\_to\_int,

'validate': {'type:values': n\_const.VALID\_VXLAN\_UDP\_PORTS},

'default': n\_const.DEFAULT\_VXLAN\_UDP\_PORT,

'is\_visible': True},

'ttl': {'allow\_post': True, 'allow\_put': False,

'convert\_to': converters.convert\_to\_int,

'validate': {'type:range': n\_const.VALID\_TTL\_RANGE},

'is\_visible': True,

'default': constants.ATTR\_NOT\_SPECIFIED},

'vxlan': {'allow\_post': False, 'allow\_put': False,

'is\_visible': True},

},

‘providernet\_types': {

'type': {'allow\_post': False, 'allow\_put': False,

'validate': {'type:values': PROVIDERNET\_TYPE\_LIST},

'is\_visible': True},

'description': {'allow\_post': False, 'allow\_put': False,

'is\_visible': True}

},

}

In addition neutron subnets will be extended with some fields to expose the network\_type, physical\_network and segmentation\_id assigned to the subnets resource.

EXTENDED\_ATTRIBUTES\_2\_0 = {

'subnets': {

‘network\_type’: {'allow\_post': False, 'allow\_put': False,

'enforce\_policy': True,

'is\_visible': True},

‘physical\_network’: {'allow\_post': False, 'allow\_put': False,

'enforce\_policy': True,

'is\_visible': True},

‘segmentation\_id’: {'allow\_post': False, 'allow\_put': False,

'convert\_to': int,

'enforce\_policy': True,

'default': constants.ATTR\_NOT\_SPECIFIED,

'is\_visible': True},

},

}

**REST API Impact**

The following provider network management Rest APIs will be provided in line with the new resources previously introduced:

List provider network types supported

GET /v2.0/providernet\_types

Create a provider network record for given network data  
POST /v2.0/providernets/<providernet-uuid>

Delete a provider network by id

DELETE /v2.0/providernets/<providernet-uuid>

Update a provider network with given data

PUT /v2.0/providernets/<providernet-uuid>

List provider networks

GET /v2.0/providernets

List a provider network information

GET /v2.0/providernets/<providernet-uuid>

Create a provider network segmentation id range record for given network data.

POST /v2.0/providernet\_ranges/<providernet\_ranges-uuid>

Delete a provider network segmentation id range by id.

DELETE /v2.0/providernet\_ranges/<providernet\_ranges-uuid>

Update a provider network segmentation id range with given data.

PUT /v2.0/providernet\_ranges/<providernet\_ranges-uuid>

List all provider network ranges.

GET /v2.0/providernet\_ranges

List the network range information of a provider network by id.

GET /v2.0/providernet\_ranges/<providernet\_ranges-uuid>

**Command Line Client Impact**

Openstack Client would add provider network management related CLIs, for instance:

openstack providernet type-list  
 openstack providernet list  
 openstack providernet show  
 openstack providernet create  
 openstack providernet update  
 openstack providernet delete  
 openstack providernet range-list  
 openstack providernet range-show  
 openstack providernet range-create  
 openstack providernet range-update  
 openstack providernet range-delete  
 openstack providernet net-list-on-providernet

**Other Impact**

\* Agent-side changes for the synchronization of provider network information are expected to accommodate the server-side implementation.

\* This feature has a dependency on the system host management. Current network dhcp and router will have to be only scheduled/rescheduled to hosts that are attached to the provider networks (currently assigned to interfaces owned by the specified hosts) that are needed in order to correctly implement instances.

\* Necessary adjustments to the existing ML2 type manager and drivers are expected.