1.1 VPN as a Service API

This Contribution defines a VPN as a Service API in line with the Quantum Core Project to add VPN Service to Quantum in the Openstack Community. This is done by attaching the VPN Service to the basic Quantum Network Service provided by Quantum v2.0 API. Within this VPN as a Service API Framework we provide two generic examples "VPN Service Types" for GRE and IPSec.

This API can be extended to support other VPN types such as VPLS, MPLS and SSL to name a few, or any VPN type that the Infrastructure supports. The following sections cover the VPN as a Service API Framework contribution.

It is intended that this VPN as a Service API Framework be used as a basis for VPN Services supported under Quantum and noting the VPN Framework that should be supported into the "Quantum Core API".

1.2 Create VPN Service

This operation creates an instance of a VPN gateway.

```
POST /vpn-services
{
   "name": "vpn-name",
}

HTTP/1.1 201 Created
{
   "id": "vpn GW UUID"
   "gw-address": { "ipv4": "72.1.2.34", "ipv6": "2002:2::34" }
}
```

attribute	Type	req	res	description
name	String	M		A human readable name for the VPN gateway
id	UUID		M	The ID representing the created VPN gateway service. This value is generated by the implementation of the API
gw-address	structure		M	contains the IP address assigned to the VPN gateway on the Tenant Network side
ipv4	IP address		O	IPv4 address
ipv6	IP address		O	IPv6 address

1.2.1 Get VPN Service Collection

```
GET /vpn-services
```

```
HTTP/1.1 200 OK
{
   “vpn-services”:
   [
      {“id”: “VPN GW 1 UUID”, “name”: “vpn-1-name”},
      {“id”: “VPN GW 2 UUID”, “name”: “vpn-2-name”},
   ]
}
```

attribute	type	req	res	description
vpn-services	list		O	The list of existing VPN services. Only present if there are networks
Id	UUID		M	The ID representing one VPN service
Name	string		M	human readable name representing the VPN service

1.2.2 Get a VPN Service

```
GET /vpn-services/<UUID>

HTTP/1.1 200 OK
{
   “name”: “vpn-name”,
   “id”: “vpn-UUID”,
   “gw-address”: { “ipv4”: “72.1.2.34”, “ipv6”: “2002:2::34” },

   “ipsec-tunnels”:
   [
      {
         <ipsec VPN specific attributes>
      },
      {
         <ipsec VPN specific attributes>
      }
   ],

   “gre-tunnels”:
   [
      {
         <GRE VPN specific attributes>
      },
      {
         <GRE VPN specific attributes>
      }
   ],
}
```

attribute	type	req	res	description
Name	string		M	human readable name representing the VPN service
Id	UUID		M	The ID representing the VPN service
gw-address	structure		M	contains the IP address assigned to the VPN gateway
ipv4	IP address		O	an IPv4 address

attribute	type	req	res	description
ipv6	IP address		O	an IPv6 address
ipsec-tunnels	list		O	list of IPSEC tunnels starting at this VPN gateway
gre-tunnels	list		O	list of GRE tunnels starting at this VPN gateway

1.2.3 Update a VPN Service

Note: This will be covered in the next iteration of the Eri-Cloud-NET-API.

1.2.4 Delete a VPN Service

```
DELETE /vpn-services/<UUID>

HTTP/1.1 200 OK
```

1.2.5 Configure VPN Service

1.2.5.1 Configure IPSEC Tunnel

This will create an instance of an IPSEC VPN tunnel from the gateway specified by the <UUID> and a customer gateway.

```
POST /vpn-services/<UUID>/tunnel-ipsec/
{
  “name”: <tunnel-name>
  “ike”:
  {
    “pre-shared-key”: “string”,
    “ike-auth-algo”: “sha1”,
    “ike-encryp-algo”: “aes-128-cbc”,
    “lifetime”: “num-secs”,
  },

  “ipsec”:
  {
    “protocol”: “esp”,
    “mode”: “tunnel”
    “ipsec-auth-algo”: “hmac-sha1-96”,
    “ipsec-encryp-algo”: “aes-128-cbc”,
    “lifetime”: “num-secs”,
  },

  “tunnel”:
  {
    “ipv4”: {
      “customer-gw-address”: “139.23.44.198”,
      “local-tunnel-address-id”: “Address-UUID”,
      “routes”: [“10.2.3.0/24”, “11.2.2.0/24”]
    },
    “ipv6”: {
      “customer-gw-address”: “2002:44::c6”,
```

```
      "local-tunnel-address-id": "Address-UUID",
      "routes": ["fc00:3::/64", "fc00:2::/64"]
    }
  }
}

HTTP/1.1 201 Created
{
  "name": <tunnel-name>,
  "mtu": 1436,
  "ipv4": {
    "gw-address": "72.1.2.34",
    "customer-gw-address": "139.23.44.198",
    "local-tunnel-address-id": "Address-UUID"
  },
  "ipv6": {
    "gw-address": "2002:2::34",
    "customer-gw-address": "2002:44::c6",
    "local-tunnel-address-id": "Address-UUID"
  }
}
```

attribute	type	req	res	description
ike	structure	M		contains the data for configuring IKE
pre-shared-key	string	M		string representing the pre shard key to use for IKE
ike-auth-algo	choice	M		values are 'sha1',
ike-encrypt-algo	choice	M		values are 'aes-128-cbc',
lifetime	integer	M		number of seconds between renegociation
ipsec	structure	M		contains the data for configuring the IPSEC tunnel
protocol	choice	M		values are 'esp',
mode	choice	M		values are 'tunnel',
ipsec-auth-algo	choice	M		values are 'hmac-sha1-96',
ipsec-encrypt-algo	choice	M		values are 'aes-128-cbc',
tunnel	structure	M		contains data for configuration of an IP endpoint
ipv4	structure	O	O	contains IPv4 related configuration data of an IP endpoint
ipv6	structure	O	O	contains IPv6 related configuration data of an IP endpoint
customer-gw-address	IP Address	M	M	IP address of the customer VPN gateway
local-tunnel-address	IP_Address	M	M	IP address for the local endpoint of the tunnel
routes	list	M		list of routes to inject into this tunnel
tunnel-name	string	M	M	human readable name for the tunnel. This comes from the last part of the URI
mtu	integer		M	the MTU value that should be set on the tunnel interface
gw-address	IP Address		M	IP address of the VPN gateway service

1.2.5.2 Delete an IPSEC Tunnel

```
DELETE /vpn-services/<UUID>/tunnel-ipsec/<UUID>

HTTP/1.1 200 OK
```

1.2.5.3 Configure GRE Tunnel

This will create an instance of a GRE VPN tunnel from the gateway specified by the <UUID> and a customer gateway.

```
POST /vpn-services/<UUID>/tunnel-gre/
{
   "name": <tunnel-name>
   "tunnel":
   {
      "ipv4": {
         "customer-gw-address": "139.23.44.198",
         "local-tunnel-address": "IPAddress",
         "routes": ["10.2.3.0/24", "11.2.2.0/24"]
      },
      "ipv6": {
         "customer-gw-address": "2002:44::c6",
         "local-tunnel-address": "IPAddress",
         "routes": ["fc00:3::/64", "fc00:2::/64"]
      }
   }
}

HTTP/1.1 201 Created
{
   "name": <tunnel-name>
   "mtu": 1436,
   "ipv4": {
      "gw-address": "72.1.2.34",
      "customer-gw-address": "139.23.44.198",
      "local-tunnel-address": "IP_Address"
   },
   "ipv6": {
      "gw-address": "2002:2::34",
      "customer-gw-address": "2002:44::c6",
      "local-tunnel-address": "IP_Address"
   }
}
```

attribute	type	req	res	description
tunnel	structure	M		contains data for configuration of en IP endpoint
ipv4	structure	O	O	contains IPv4 related configuration data of an IP endpoint
ipv6	structure	O	O	contains IPv6 related configuration data of an IP endpoint

attribute	type	req	res	description
customer-gw-address	IP Address	M	M	IP address of the customer VPN gateway
local-tunnel-address-id	UUID	M	M	UUID of IP address for the local endpoint of the tunnel
routes	list	M		list of routes to inject into this tunnel
name	string	M	M	human readable name for the tunnel. This comes from the last part of the URI
mtu	integer		M	the MTU value that should be set on the tunnel interface
gw-address	IP Address		M	IP address of the VPN gateway service

1.2.5.4 Delete a GRE Tunnel

DELETE ***/vpn-services***/<UUID>/***tunnel-gre***/<tunnel-name>

HTTP/1.1 200 OK