Introduction to Neutron

Network as a Service

Assaf Muller, Associate Software Engineer,
Cloud Networking, Red Hat

assafmuller.wordpress.com, amuller@redhat.com, amuller on Freenode (#openstack)
The Why

- The first slide is always a giant graph
- Forget graphs, why Neutron?
  - Tenant connectivity and isolation
  - Network virtualization – Tenant creates his own network(s), router(s), allowing complex network topologies for multi-tier applications
  - *aaS
Network Virtualization Example

• Create a network
• Connect the new private network to the public network
• Create a DB instance
• Create three web servers
• Make DB accessible exclusively by web servers
• Create a load balancer to the web servers
• Attach a floating IP to the load balancer
• Filter anything other than incoming HTTP(s) traffic

All done logically, in software, via the GUI, API or CLI
Core Concepts

Ports

Network

Subnets
Network Types

- External network – Internet routable network
- Provider network – Created by admin, mapped to pre-existing network in datacenter, used for external networks*
- Tenant network – Self provisioned network, isolated from other tenants, optionally connected to other tenant and external networks

* VMs may also be directly connected to provider networks
Routers, NAT, Floating IPs

Tenant Network – 10.0.0.0/8

Compute Node

VM: 10.0.0.1

VM: 10.0.0.2

Network Node
Routers, NAT, Floating IPs

Network Node

Provider Network – 212.10.20.0/24

Tenant Network – 10.0.0.0/8

vRouter – Internal – 10.255.255.254
vRouter – External – 212.10.20.1
vRouter – Floating IP – 212.10.20.2

Compute Node

VM: 10.0.0.1
VM: 10.0.0.2

Compute Node

VM: 10.0.0.1
VM: 10.0.0.2
Components

Management
VM Data
Internet
API

API Services
Compute Node (Nova compute, L2 agent)
Compute Node (Nova compute, L2 agent)
Network Node (L2 agent, vRouters, DHCP, Metadata)
Components

API Node:
- Neutron
- Nova
- Keystone
- Glance
- Swift
- Cinder
- Optionally SQL Database
- Optionally AMQP Broker

Network Node:
- L2 Agent
- L3 Agent
- DHCP Agent
- Metadata Agent
- Optionally LB, VPN and FW agents

Compute Node:
- L2 Agent
- Nova Compute

Compute Node:
- L2 Agent
- Nova Compute
Components

API
- Neutron Service – Accepts API requests, handles DB
- L3 Agent – Manages virtual routers
- DHCP Agent – Manages dnsmasq instances
- Metadata Agent – Proxy to Nova metadata service
- L2 Agent – Manages networking on compute node – OVS flows / Linux bridges, VLAN tagging, security groups

* Components talk via RPC
1. Create VM connected to network X (API)
Nova <--> Neutron Interaction

1. Create VM connected to network X (API)

2. Create VM (RPC: Nova API to Nova conductor)
Nova <-> Neutron Interaction

2. Create VM (RPC: Nova API to Nova conductor)

1. Create VM connected to network X (API)

3. Nova schedules VM
Nova <--> Neutron Interaction

1. Create VM connected to network X (API)

2. Create VM (RPC: Nova API to Nova conductor)

3. Nova schedules VM

4. Create VM (RPC: Nova conductor to Nova compute)

Nova
Neutron

DHCP Agent

L2 Agent
Nova Compute

L2 Agent
Nova Compute
Nova <-> Neutron Interaction

1. Create VM connected to network X (API)
2. Create VM (RPC: Nova API to Nova conductor)
3. Nova schedules VM
4. Create VM (RPC: Nova conductor to Nova compute)
5. Create Port (API: Nova compute to Neutron service)
6. Create tap device
Nova <-> Neutron Interaction

1. Create VM connected to network X (API)
2. Create VM (RPC: Nova API to Nova conductor)
3. Nova schedules VM
4. Create VM (RPC: Nova conductor to Nova compute)
5. Create Port (API: Nova compute to Neutron service)
6. Create tap device
7. Neutron creates port (Allocates MAC, IP)
Nova <-> Neutron Interaction

2. Create VM (RPC: Nova API to Nova conductor)

1. Create VM connected to network X (API)

8. Notify DHCP agent (RPC)

3. Nova schedules VM

4. Create VM (RPC: Nova conductor to Nova compute)

5. Create Port (API: Nova compute to Neutron service)

6. Create tap device

7. Neutron creates port (Allocates MAC, IP)

9. Notify L2 agent (RPC)
Nova <-> Neutron Interaction

1. Create VM connected to network X (API)
2. Create VM (RPC: Nova API to Nova conductor)
3. Nova schedules VM
4. Create VM (RPC: Nova conductor to Nova compute)
5. Create Port (API: Nova compute to Neutron service)
6. Create tap device
7. Neutron creates port (Allocates MAC, IP)
8. Notify DHCP agent (RPC)
9. Notify L2 agent (RPC)
10. get_device_details (RPC: L2 agent to Neutron service)
Nova <--> Neutron Interaction

1. Create VM connected to network X (API)
2. Create VM (RPC: Nova API to Nova conductor)
3. Nova schedules VM
4. Create VM (RPC: Nova conductor to Nova compute)
5. Create Port (API: Nova compute to Neutron service)
6. Create tap device
7. Neutron creates port (Allocates MAC, IP)
8. Notify DHCP agent (RPC)
9. Notify L2 agent (RPC)
10. get_device_details (RPC: L2 agent to Neutron service)
11. Configure local VLAN, OVS flows
12. Send port_up notification (RPC: L2 agent to Neutron service)
Nova <---> Neutron Interaction

1. Create VM connected to network X (API)

2. Create VM (RPC: Nova API to Nova conductor)

3. Nova schedules VM

4. Create VM (RPC: Nova conductor to Nova compute)

5. Create Port (API: Nova compute to Neutron service)

6. Create tap device

7. Neutron creates port (Allocates MAC, IP)

8. Notify DHCP agent (RPC)

9. Notify L2 agent (RPC)

10. get_device_details (RPC: L2 agent to Neutron service)

11. Configure local VLAN, OVS flows

12. Send port_up notification (RPC: L2 agent to Neutron service)

13. Send port_up notification (API: Neutron service to Nova)
Nova <-> Neutron Interaction

1. Create VM connected to network X (API)
2. Create VM (RPC: Nova API to Nova conductor)
3. Nova schedules VM
4. Create VM (RPC: Nova conductor to Nova compute)
5. Create Port (API: Nova compute to Neutron service)
6. Create tap device
7. Neutron creates port (Allocates MAC, IP)
8. Notify DHCP agent (RPC)
9. Notify L2 agent (RPC)
10. get_device_details (RPC: L2 agent to Neutron service)
11. Configure local VLAN, OVS flows
12. Send port_up notification (RPC: L2 agent to Neutron service)
13. Send port_up notification (API: Neutron service to Nova)
14. port_up (RPC: Nova service to Nova compute)
Nova <-> Neutron Interaction

1. Create VM connected to network X (API)
2. Create VM (RPC: Nova API to Nova conductor)
3. Nova schedules VM
4. Create VM (RPC: Nova conductor to Nova compute)
5. Create Port (API: Nova compute to Neutron service)
6. Create tap device
7. Neutron creates port (Allocates MAC, IP)
8. Notify DHCP agent (RPC)
9. Notify L2 agent (RPC)
10. get_device_details (RPC: L2 agent to Neutron service)
11. Configure local VLAN, OVS flows
12. Send port_up notification (RPC: L2 agent to Neutron service)
13. Send port_up notification (API: Neutron service to Nova)
14. port_up (RPC: Nova service to Nova compute)
15. Nova compute boots VM!
Core Plugin

• Implementation of API is pluggable
• Many plugins exist:
  – ML2 (Open source, Red Hat blessed)
  – Vendor plugins (Proprietary, some are certified)
• Each plugin implements API differently. For example, 'create_port' may be proxied to some 3rd party server
Service Plugins

- Firewall as a Service – Filter traffic at the router level
- VPN as a Service – Extend your tenant networks off-site
- Load Balancer as a Service – Distribute incoming messages to a pool of VMs

* Service plugins also have multiple implementations
Tenant Connectivity & Isolation

• How is tenant networks connectivity and isolation achieved, exactly?
  - Via VLANs or GRE / VXLAN tunnels
• More information at FOSDEM talk:
  - http://assafmuller.wordpress.com/2014/02/03/fosdem-14/
  - Includes video, slides and series of blog posts
Network as a Service

Assaf Muller, Associate Software Engineer,
Cloud Networking, Red Hat