OpenStack networking Neutron

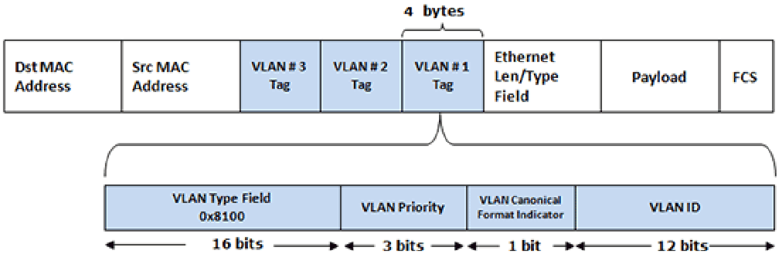
Antonio Messina **`<antonio.messina@s3it.uzh.ch>`**
S^{3}IT - Services and Support for Science IT, University of Zurich

Why neutron?

- Allow tenants to define the network topology
- Support very rich network topologies.
  - including existing topologies
- Allow easy integration with network infrastructure.
- Allow creation of advanced network services, like:
  - load balancing
  - VPN
  - firewall

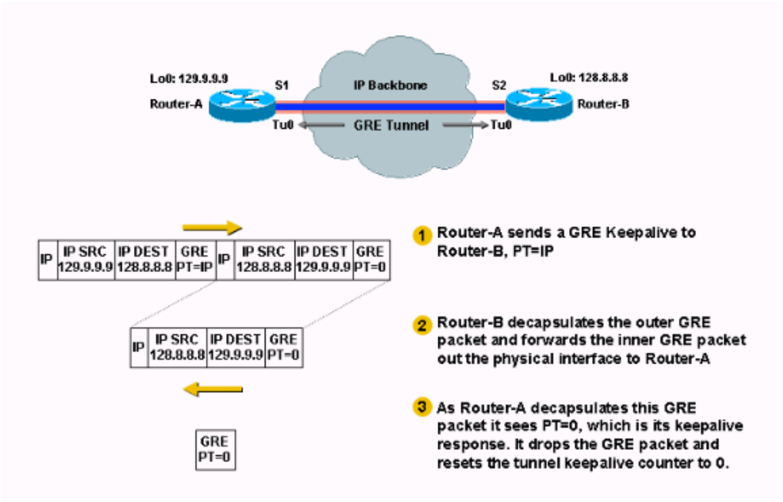
https://wiki.openstack.org/wiki/Neutron

Frame Ethernet (with VLAN)



- commonly used to separate L2 traffic on the same HW
- Implemented in hardware on the switches
- can be used for Neutron Tenant Networks
- Max nr. of VLANs 4096

GRE Tunnel



1. Router-A sends a GRE Keepalive to Router-B, PT=IP
2. Router-B decapsulates the outer GRE packet and forwards the inner GRE packet out the physical interface to Router-A
3. As Router-A decapsulates this GRE packet it sees PT=0, which is its keepalive response. It drops the GRE packet and resets the tunnel keepalive counter to 0.

GRE Tunnel (2)

- IP over IP (both network and transport protocol, kinda breaks ISO model)
- Point-to-point tunnels
- reduces the MTU of the transported protocol by 24 bytes
- can be used for Neutron Tenant Networks
- in OpenStack, full mesh is needed among all the nodes (compute and network) unless l2population mechanism driver is used

Linux Network Namespaces

A namespace is an *isolated copy* of the network stack

- Each namespace has its own private loopback.
- Routing is local to the namespace.
- Addressing scope limited to the namespace
  - ⇒ different namespaces can have overlapping IP addresses
- Interfaces **do not have** direct connectivity to the network: you must connect them to a bridge in the default namespace
- You can spawn processes within a namespace (e.g. `dnsmasq` for DHCP)

LWN.net: Namespaces in operation

Namespaces cheatsheet

- `ip netns list` list namespaces
- `ip netns exec <ns>` execute a command inside a namespace
- `ip netns pids <ns>` reports processes running in a namespace
- `ip netns identify <pid>` reports namespace name for a process
- `ip netns help` guess

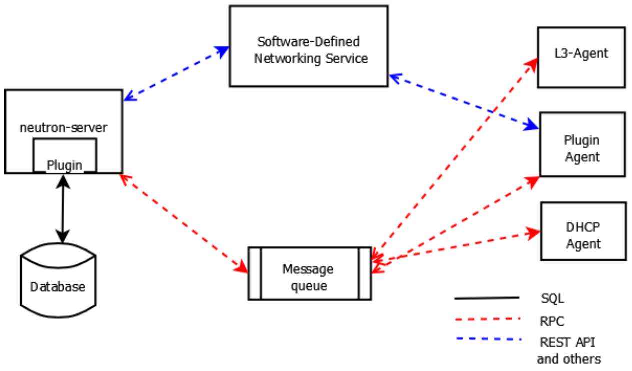
Neutron architecture

- **neutron server** (network or api node) rest API, talks to DB and AMQP
- **plugin agent** (network and compute nodes) manages virtual switches and ports
- **DHCP agent** (network nodes) provides DHCP services to tenant networks
- **L3 agent** (network nodes) provides routing and NAT capabilities
- **SDN services** (network nodes) additional network services

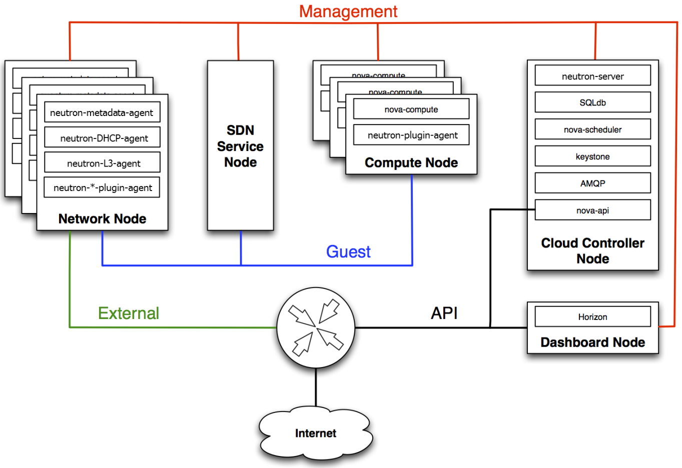
Neutron services integrations

Neutron server, plugins and agents talk to each other via:

- rest API
- RPC (RabbitMQ in our case)
- SQL (MySQL in our case)



Neutron architecture - physical servers (standard)



- We will install `neutron-server` on **neutron-node**

Notes on HA

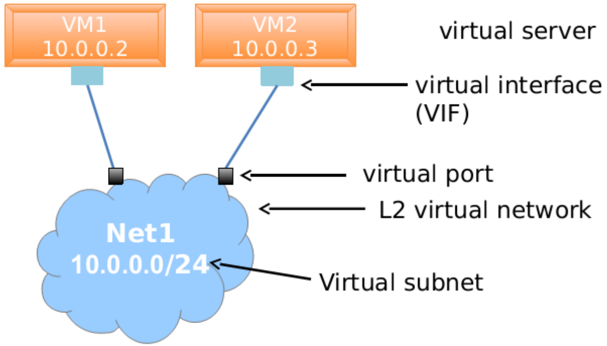
dhcp agents are easy: DHCP protocol is highly available by design.

L3 agents are firewalls, and firewalls are not stateless, so it's hard to make them Highly Available.

On neutron you have two choices for L3 agent:

- **DVR** each compute node acts as a router.
- **L3-HA** Two network nodes in HA using VRRP

main building blocks



network	A L2 network
subnet	An IPv4 or IPv6 network, living inside a `network`
port	a virtual switch port on a given `network`.
virt. interface	instance interface, connected to a `port`

ML2 plugin (since Havana/Icehouse)

- Only one plugin is active in neutron at a time.
- ML2 plugin allow to use multiple L2 networking technologies at the same time.
- Being modular, reduce duplication of code, and makes easier create plugins for Neutron.
- Decouple the *type* of network and its *implementation*:

type driver	(GRE, VLAN, VXLAN, Flat)
mechanism driver	specific implementation for a specific network technology (OpenVSwitch, cisco, brocade . . .)

https://wiki.openstack.org/wiki/Neutron/ML2

L2 agent

- usually runs on the hypervisor
- talks to server via RPC
- watch and notify when devices are added/removed
- wires new devices ensuring:
  - they are in the proper network segment (L2 network)
  - security group rules are applied

We will deploy `neutron-openvswitch-agent`

http://openvswitch.org/

DHCP configuration agent

- RPC based notifications
- uses `dnsmasq` (one per network)
- uses namespaces (`qdhcp-<uuid-of-neutron-subnet>`)
- typically runs on the network node
- tap interface `tap-XXX` with *private IP*, wired to `br-int`
- you can have multiple copies to achieve HA (part of DHCP: multiple servers cun run on the same network segment, the client gets the first response)[1]

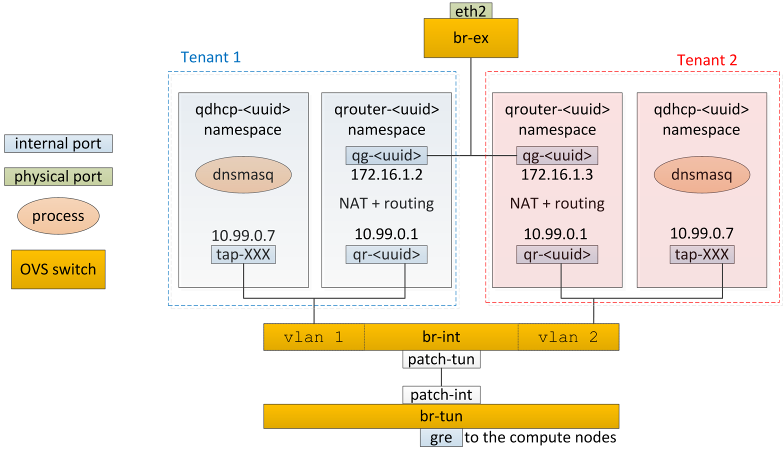
[1]as long as the conf is the same

L3 agent

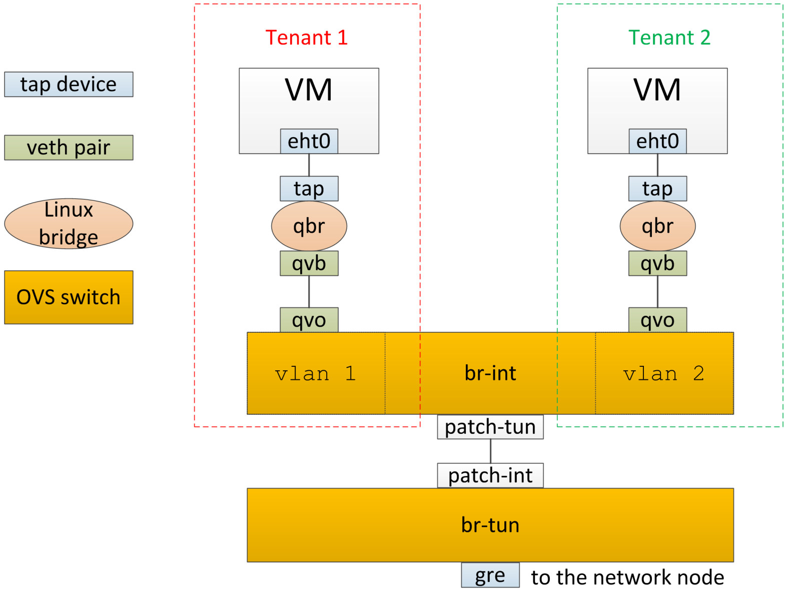
Responsible for routing and floating IPs.

- Runs on the network node (typically).
- Uses namespaces (`qrouter-<uuid-of-neutron-router>`)
- Also provides metadata agent.
- Routing is done with static routes.
  - ⇒ linux forwarding must be enabled
- tap interface `qg-X` with *public* IP, wired to `br-int`.
- tap interface `qr-X` with *private* IP, wired to `br-int`.
- IP on `qr-X` is the gateway for the tenant network.
- When using floating IPs, the L3 agent will assign the floating ip to `qg-X` and set in place a 1:1 NAT between public and private IPs.
- Otherwise, the L3 agent simply use NAT (MASQUERADE) to allow external connectivity (using the IP of `qg-X` interface)

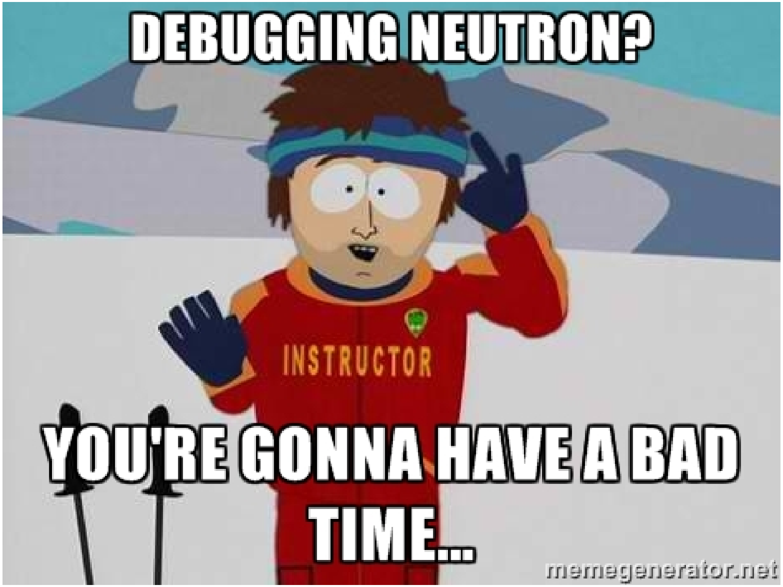
Under the hood - network node



Under the hood - compute node



easy peasy


DEBUGGING NEUTRON?
INSTRUCTOR
YOU'RE GONNA HAVE A BAD TIME...
memegenerator.net

Metadata proxy

- usually embededd in the L3 agent
- can also be managed by the dhcp agent (useful for *isolated* network)
- gets the request from the client, and redirect to the api node
- uses IP address 169.254.169.254
- requires a client on the VM (cfr cloud-init)

References

- Cloud Administrator Guide - Chapter 7 - Networking
- https://wiki.openstack.org/wiki/Neutron
- OpenStack Summit Atlanta 2014 - Inside the Architecture of Neutron