Cloud Networking (VITMMA02)
Network Virtualization: Overlay Networks
OpenStack Neutron Networking

Markosz Maliosz PhD

Department of Telecommunications and Media Informatics
Faculty of Electrical Engineering and Informatics
Budapest University of Technology and Economics

Spring 2018

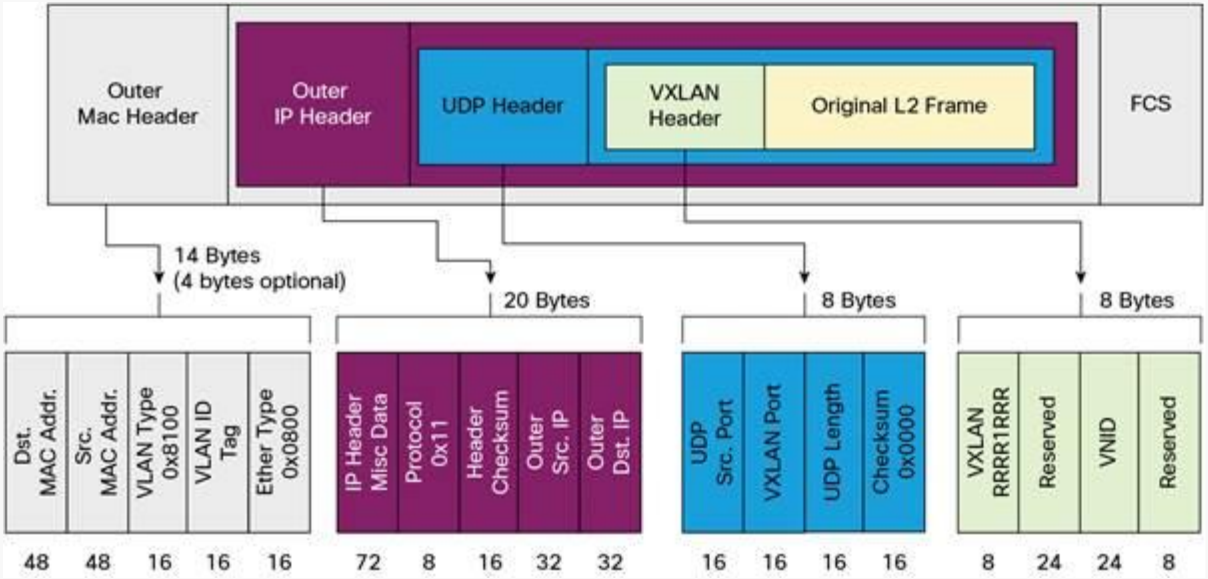
OVERLAY NETWORKS

Network Virtualization

- Support for tenant separation
  - Virtual Extensible LAN (VXLAN) – RFC 7348
    - Cisco, VMware
    - transport of virtual L2 traffic over physical L3 network
  - Network Virtualization using Generic Routing Encapsulation (NVGRE)
    - Microsoft, Intel, HP, Dell
  - Generic Network Virtualization Encapsulation (GENEVE)
    - superset of VXLAN and NVGRE
  - Stateless Transport Tunneling (STT)
    - Nicira ⇨ VMware

VXLAN

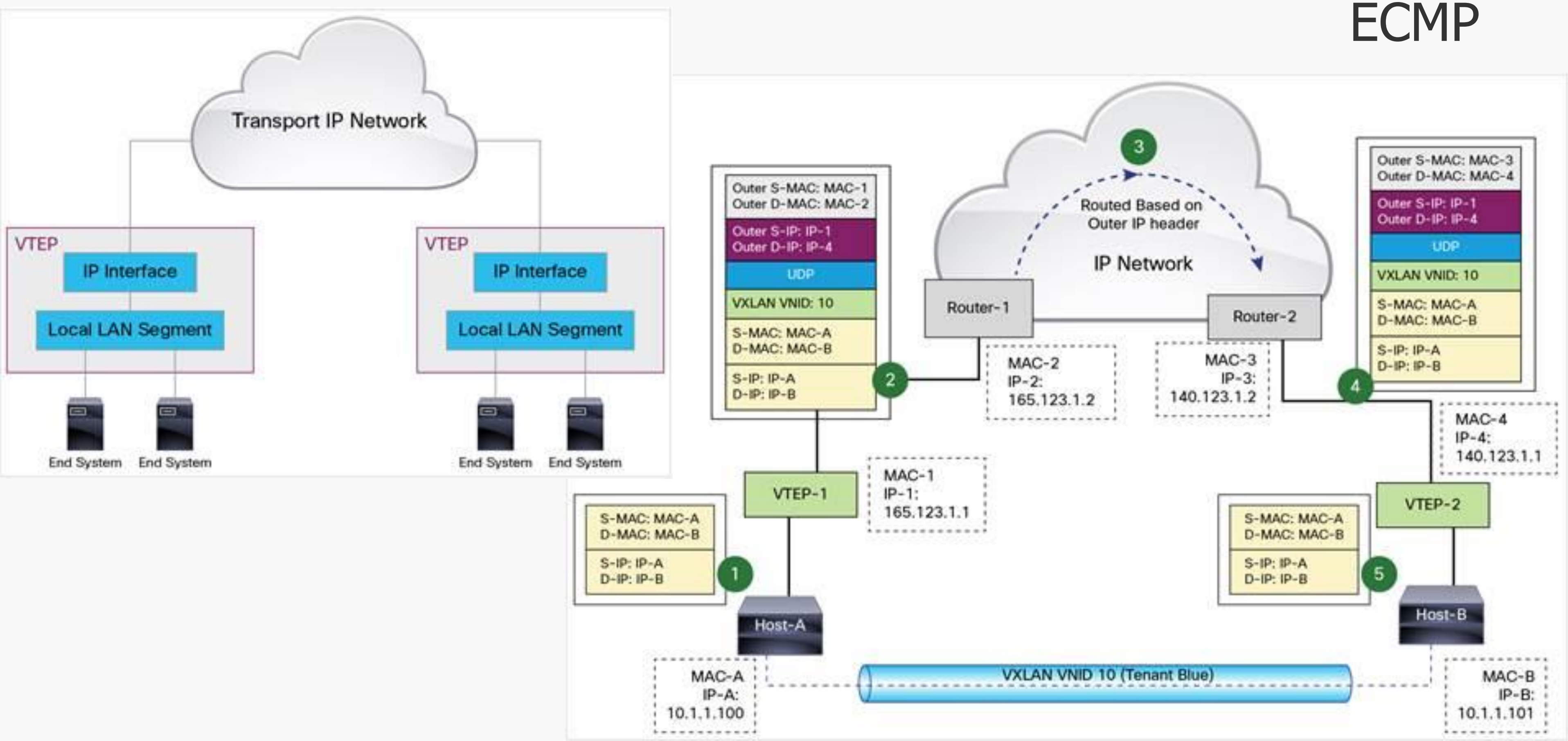
- Original L2 frame of tenant
  - original MAC address and VLAN tag
- MAC-in-UDP
- VXLAN and UDP header
  - VXLAN network ID (VNID) – identifying the tenant
    - 24 bit ⇨ 16 million tenant
- physical network: IP routing (Layer3)



VXLAN

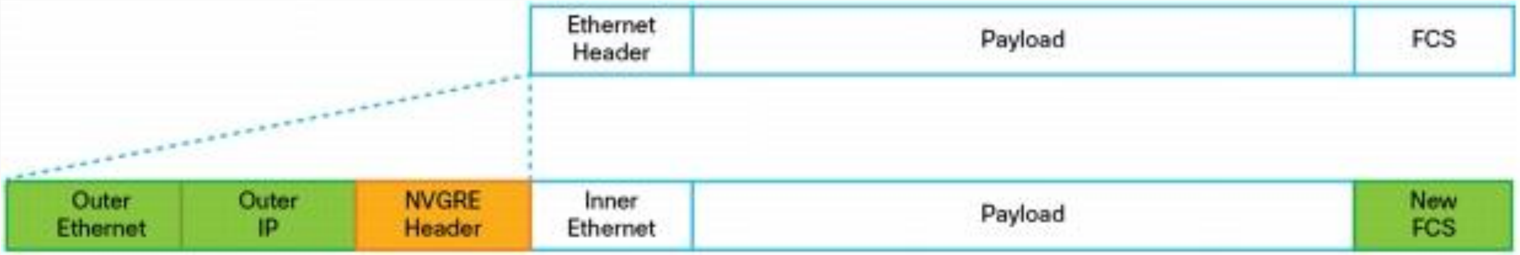
- VXLAN Tunnel End Point (VTEP)
- MAC-to-VTEP tables by learning (IP multicast)
  - all VTEP of a VNI in a multicast group

ECMP



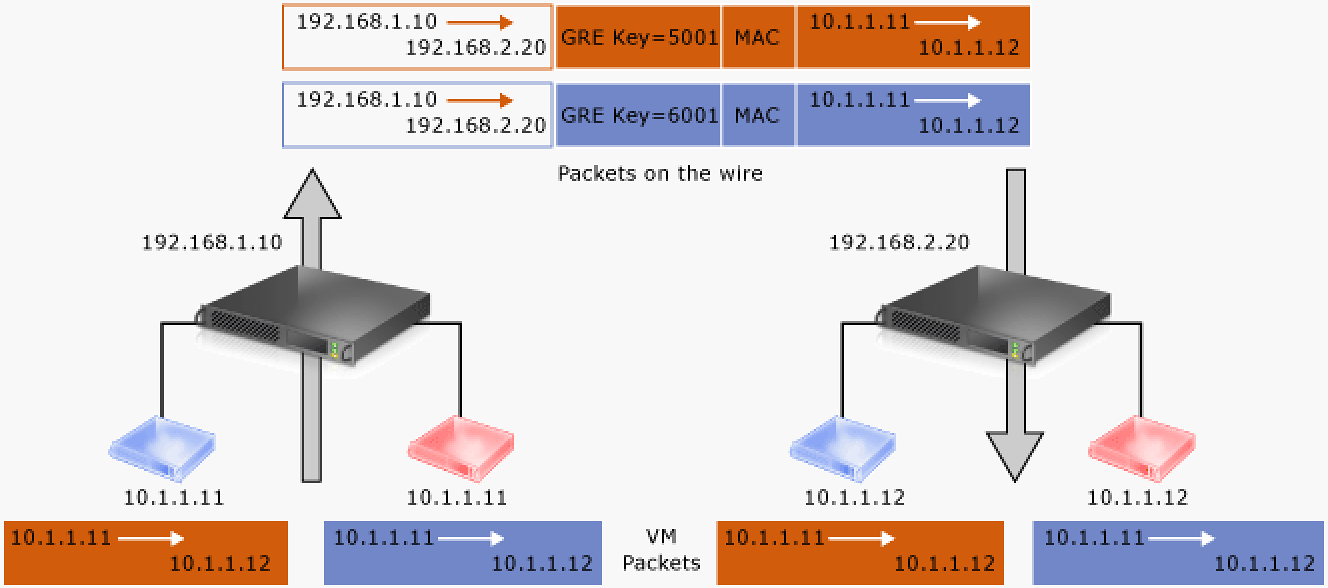
NVGRE

- is very similar to VXLAN
- basis: Generic Routing Encapsulation (GRE)
  - generic header
  - can encapsulate a wide variety of network layer protocols
  - point-to-point links over an Internet Protocol network
- NVGRE
  - GRE header
  -
    ```
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |0| |1|0|   Reserved0      | Ver |   Protocol Type 0x6558        |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |               Virtual Subnet ID (VSID)        |    FlowID     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ```
    - Virtual Subnet Identifer (VSID) 24 bit ⇨ 16 million tenant
    - FlowID: optional, unique flow identifier
      - for ECMP hashing
  - no inner VLAN tag (or it is removed)
    - encoded into VSID or multiple VSIDs for the same tenant



NVGRE

- Network Virtual Endpoint (NVE)
  - forwarding to the destination based on VSID and DMAC
- not specified in the Internet draft
  - dissemination of addressing information
  - restoring VLAN information



Generic Network Virtualization Encapsulation

- MAC-in-UDP over IPv4/IPv6
- universal, extensible
- specifies only the encapsulation format
- optional fields
  - variable field lengths, flexibility
- Geneve header:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Ver|  Opt Len  |O|C|    Rsvd.  |          Protocol Type        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Virtual Network Identifier (VNI)       |    Reserved   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Variable Length Options                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Location of Tunnel Endpoint

- Inside hypervisor/vSwitch
  - most common
  - closest to the VMs
  - consumes CPU resources
  - TCP segmentation offload (TSO), checksum offload support for non-encapsulated packets
- NIC
  - offload support for tunneling
- ToR switch
  - VMs are unknown
  - has to identify VM based on inner MAC to assign VNI/VSID

Stateless Transport Tunneling (STT)

- primary goal: communication between vSwitches
- more complex than the others
- max. 64 kbyte Ethernet frames
  - maximum transmission unit (MTU)
  - using TCP segmentation offload on NIC
- STT header
  - 64 bit Context ID field
- TCP-like/IP/Eth header before the segments
  - reassembling the original data based on this

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Version      | Flags         |  L4 Offset    |  Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Max. Segment Size          | PCP |V|     VLAN ID           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                     Context ID (64 bits)                      +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Padding                   |    data                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
|                                                               |
```

```
                   +------------+    +-----------+     +-----------+
                   | IP Header  |    |IP Header  |     |IP header  |
+------------+     +------------+    +-----------+     +-----------+
|STT Frame   |     |TCP-like    |    |TCP-like   |     |TCP-like   |
| Header     |     | header     |    | header    |     | header    |
+------------+     +------------+    +-----------+     +-----------+
|            | --->| STT Frame  |    |Next part  | ... |Last part  |
|Payload     |     |  Header    |    |of Payload |     |of Payload |
.            .     +------------+    |           |     |           |
.            .     |            |    |           |     |           |
.            .     |  Start of  |    |           |     |           |
+------------+     |  Payload   |    |           |     +-----------+
                   +------------+    +-----------+

Original data            STT frame segments
```

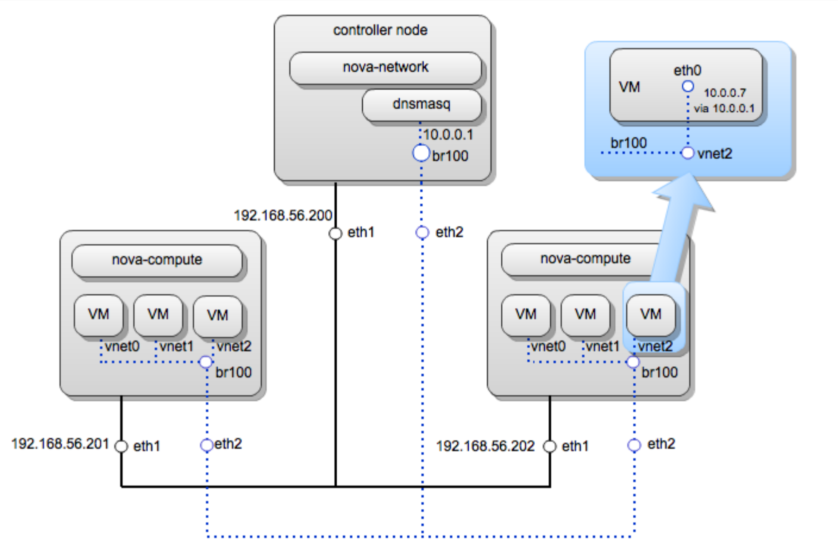
Comparison

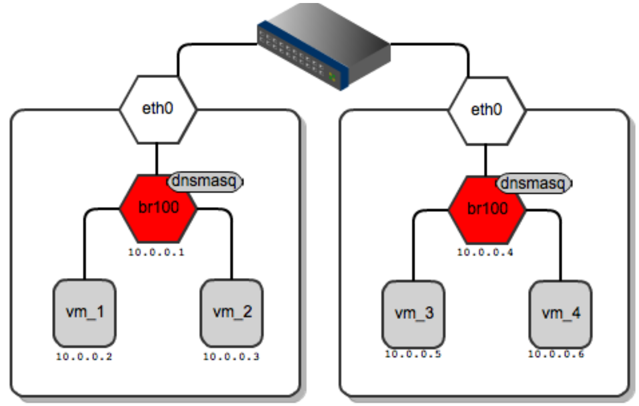
	VXLAN	NVGRE	STT
Extra bytes	50 (VLAN: +4)	42 (VLAN: +4)	First segment: 76 Others: 58 (VLAN: +4)
Protocol	UDP	GRE	TCP
Tenant separation	24 bit VNID	24 bit VSID	64 bit Context ID
ECMP flow differentiation (inner ⇨ outer flow)	Source UPD port	VSID + FlowID (8bit)	Source TCP port

OPENSTACK NEUTRON

OpenStack network architecture

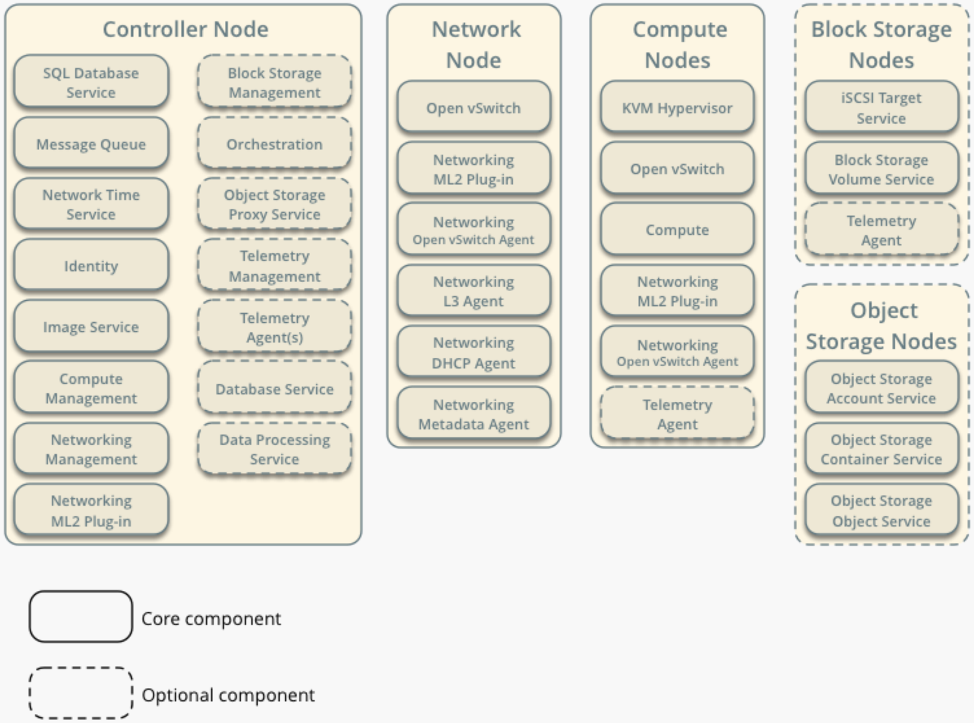
- „Networking in OpenStack is a complex, multifaceted challenge.” /OpenStack Operations Guide/
- Network as a Service
- functions
  - IP addressing
    - static, DHCP
    - floating IP
  - virtual networks
    - flat, VLAN
  - self-service
- alternatives
  - Nova networking / Neutron
  - single-host / multi-host
- Neutron
  - plug-in architecture
  - SDN/OpenFlow





Neutron network

Minimal Architecture Example - Service Layout
OpenStack Networking (neutron)



Controller Node
- SQL Database Service
- Message Queue
- Network Time Service
- Identity
- Image Service
- Compute Management
- Networking Management
- Networking ML2 Plug-in
- Block Storage Management
- Orchestration
- Object Storage Proxy Service
- Telemetry Management
- Telemetry Agent(s)
- Database Service
- Data Processing Service

Network Node
- Open vSwitch
- Networking ML2 Plug-in
- Networking Open vSwitch Agent
- Networking L3 Agent
- Networking DHCP Agent
- Networking Metadata Agent

Compute Nodes
- KVM Hypervisor
- Open vSwitch
- Compute
- Networking ML2 Plug-in
- Networking Open vSwitch Agent
- Telemetry Agent

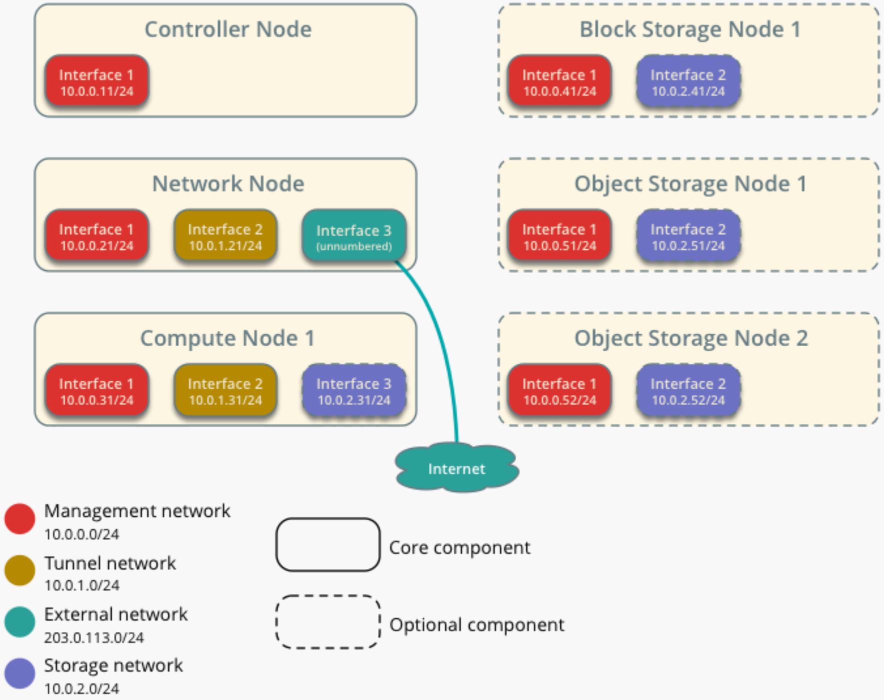
Block Storage Nodes
- iSCSI Target Service
- Block Storage Volume Service
- Telemetry Agent

Object Storage Nodes
- Object Storage Account Service
- Object Storage Container Service
- Object Storage Object Service

Core component

Optional component

Minimal Architecture Example - Network Layout
OpenStack Networking (neutron)



Controller Node
- Interface 1 10.0.0.11/24

Network Node
- Interface 1 10.0.0.21/24
- Interface 2 10.0.1.21/24
- Interface 3 (unnumbered)

Compute Node 1
- Interface 1 10.0.0.31/24
- Interface 2 10.0.1.31/24
- Interface 3 10.0.2.31/24

Block Storage Node 1
- Interface 1 10.0.0.41/24
- Interface 2 10.0.2.41/24

Object Storage Node 1
- Interface 1 10.0.0.51/24
- Interface 2 10.0.2.51/24

Object Storage Node 2
- Interface 1 10.0.0.52/24
- Interface 2 10.0.2.52/24

Internet

● Management network 10.0.0.0/24
● Tunnel network 10.0.1.0/24
● External network 203.0.113.0/24
● Storage network 10.0.2.0/24

Core component

Optional component

Nova and Neutron Network

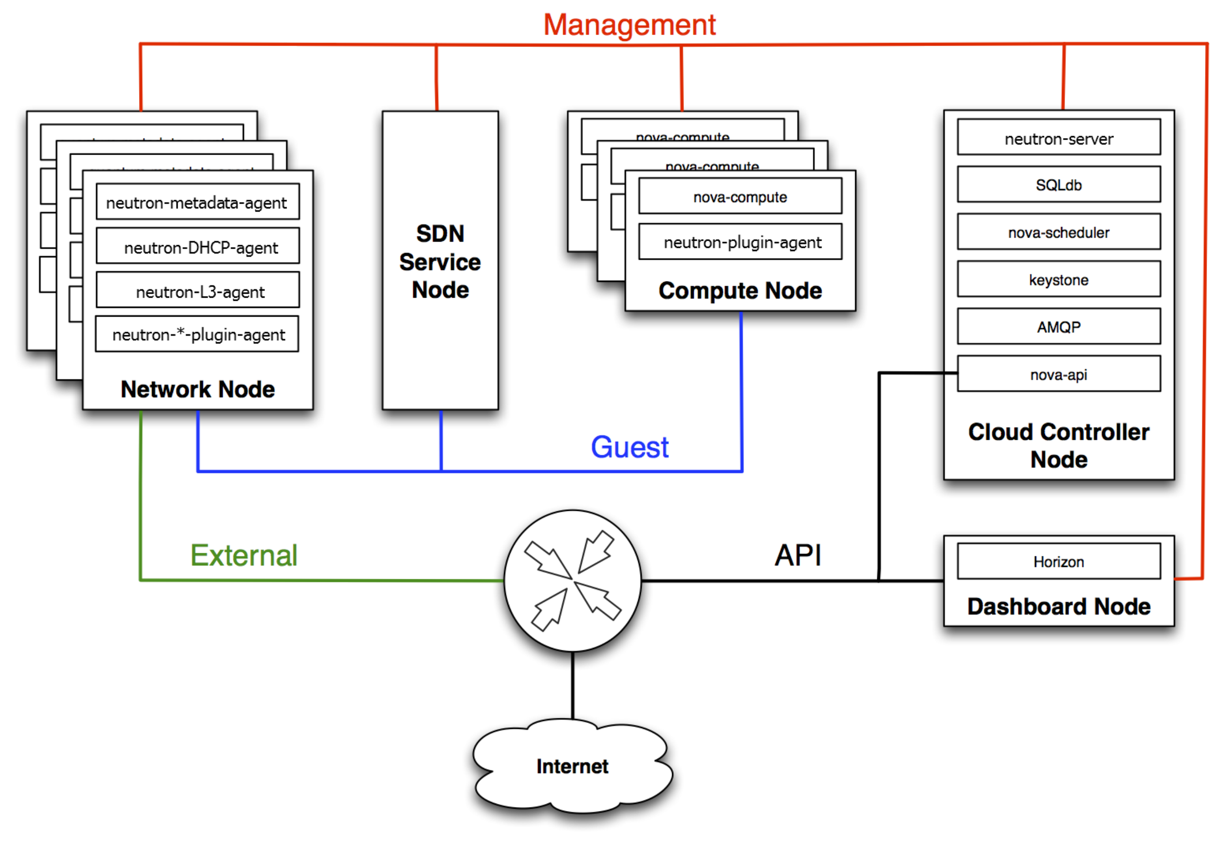
- Nova
  - basic networking functions
    - network address translation (NAT), DHCP, DNS
  - only support L2 bridge networking
    - allows virtual interfaces to connect to the outside network through the physical interface
  - limited scalability
    - VLAN, DNS&DHCP (dnsmasq)
- Neutron
  - network abstraction
  - L2/L3 network, self-service
    - e.g. more LAN segments for a web service
  - Load Balancing, Virtual IP, VPN, firewall
  - overlay VLAN tunneling
  - Distributed Virtual Router (from Juno)

Neutron network abstraction

- External /physical/ network, e.g. Internet
- Internal networks to connect VMs
  - virtual: network, subnetwork, router
  - floating IP from external network address space for reaching a VM from outside
- Security groups
  - firewall rules
  - assigned to the VM
- Open vSwitch
  - core plugin
  - br-int (integration bridge)
    - connected to VMs
  - br-ex
    - connected to external network

Neutron components

- server + plugin + agent architecture
  - neutron-server
    - on controller node
    - handling API requests
    - network modell and port IP address setup
  - plugin – extensions: neutron-*-plugin
    - on network node
  - plugin-agent: neutron-*-agent
    - on compute node
    - managing the local vswitch
  - general agents
    - DHCP: neutron-dhcp-agent
    - L3 agent: neutron-l3-agent
      - L3/NAT functionality towards the external network
      - implementation: Linux IP stack and iptables



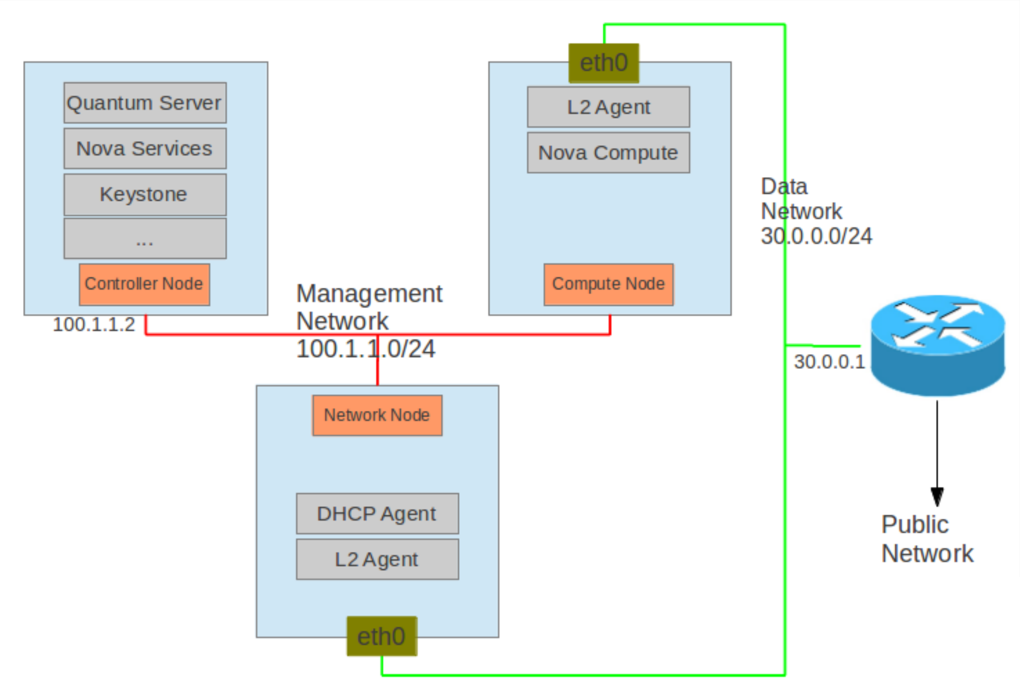
Modular Layer 2 (ML2) plugin

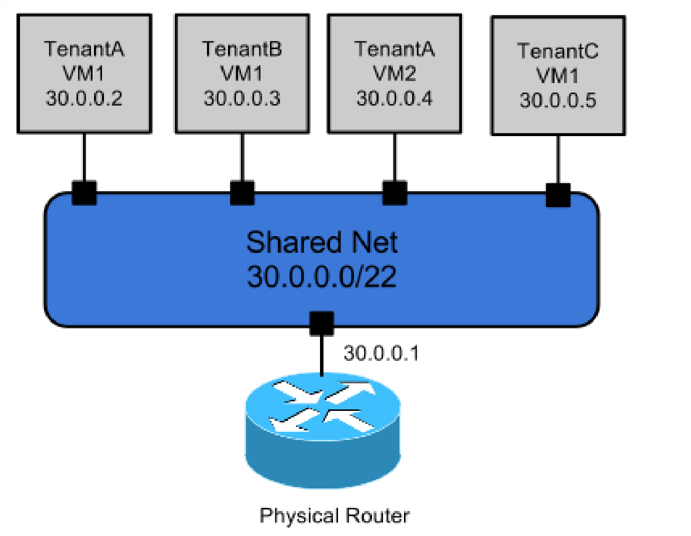
- Managing different L2 network technologies in uniform way
- Operates with openvswitch, linuxbridge and Hyper-V L2 agents
- Type drivers for different network types
  - Flat
  - Local (DevStack single box)
  - VLAN
  - GRE
  - VXLAN

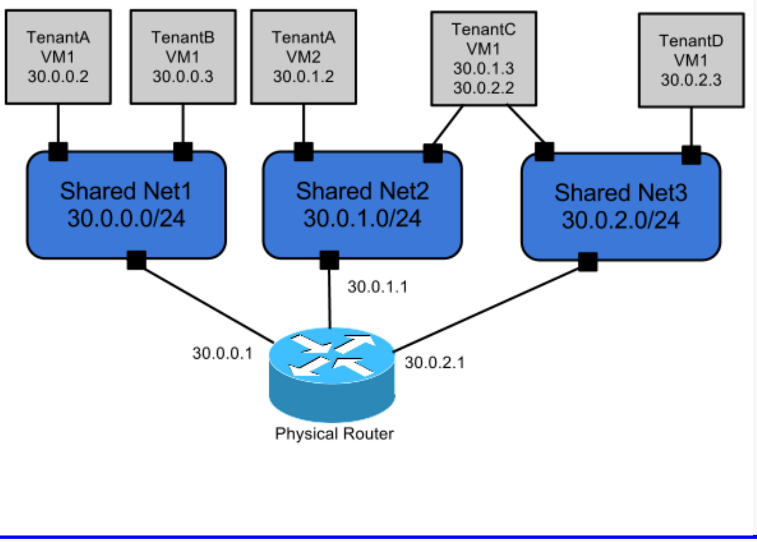
Network namespaces

- kernel level namespaces, not only for networking
  - file system, process, user, etc.
- isolated Layer2 networks with overlapping IP addresses
- separating virtual interfaces, routers
- e.g. dhcp-agent and l3-agent runs in different namespaces
- In practice
  - `ip netns`
    - lists available network namespaces
  - `ip netns exec <namespace> <command>`
    - e.g. `ip netns exec qdhcp-e521f9d0-a1bd-4ff4-bc81-78a60dd88fe5 ip a`

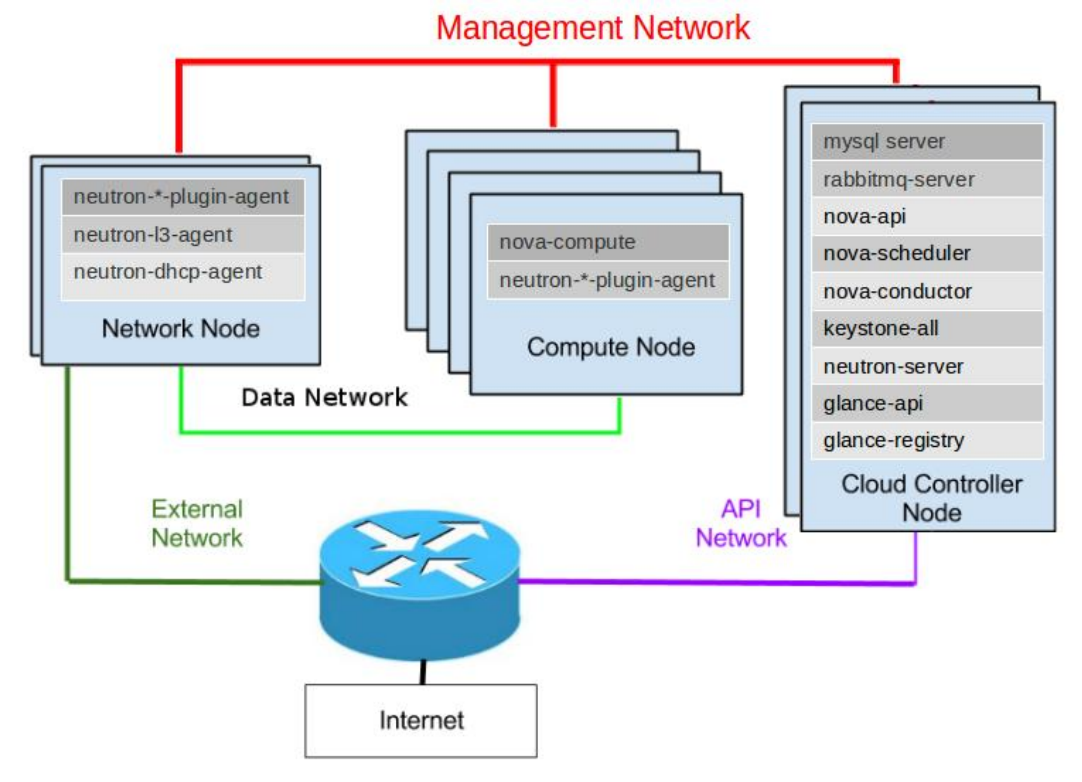
Neutron: single/multiple flat network

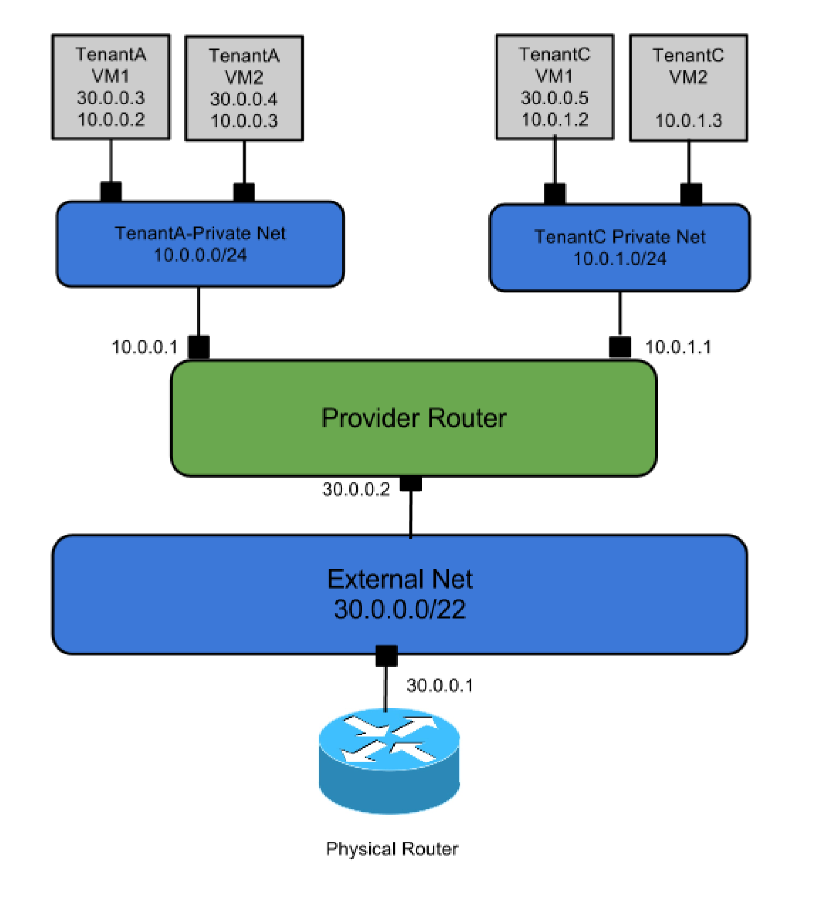




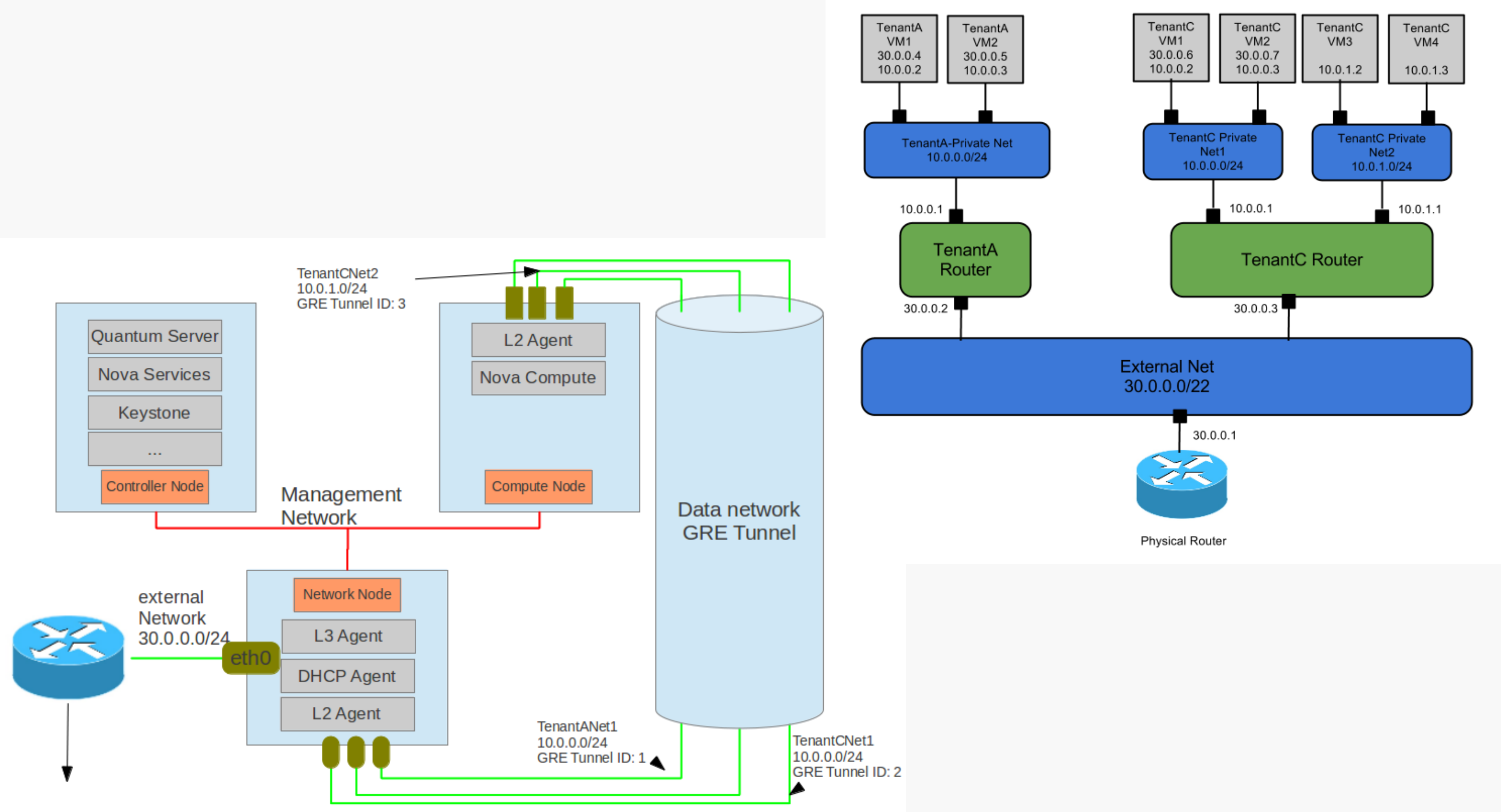


Neutron: provider router



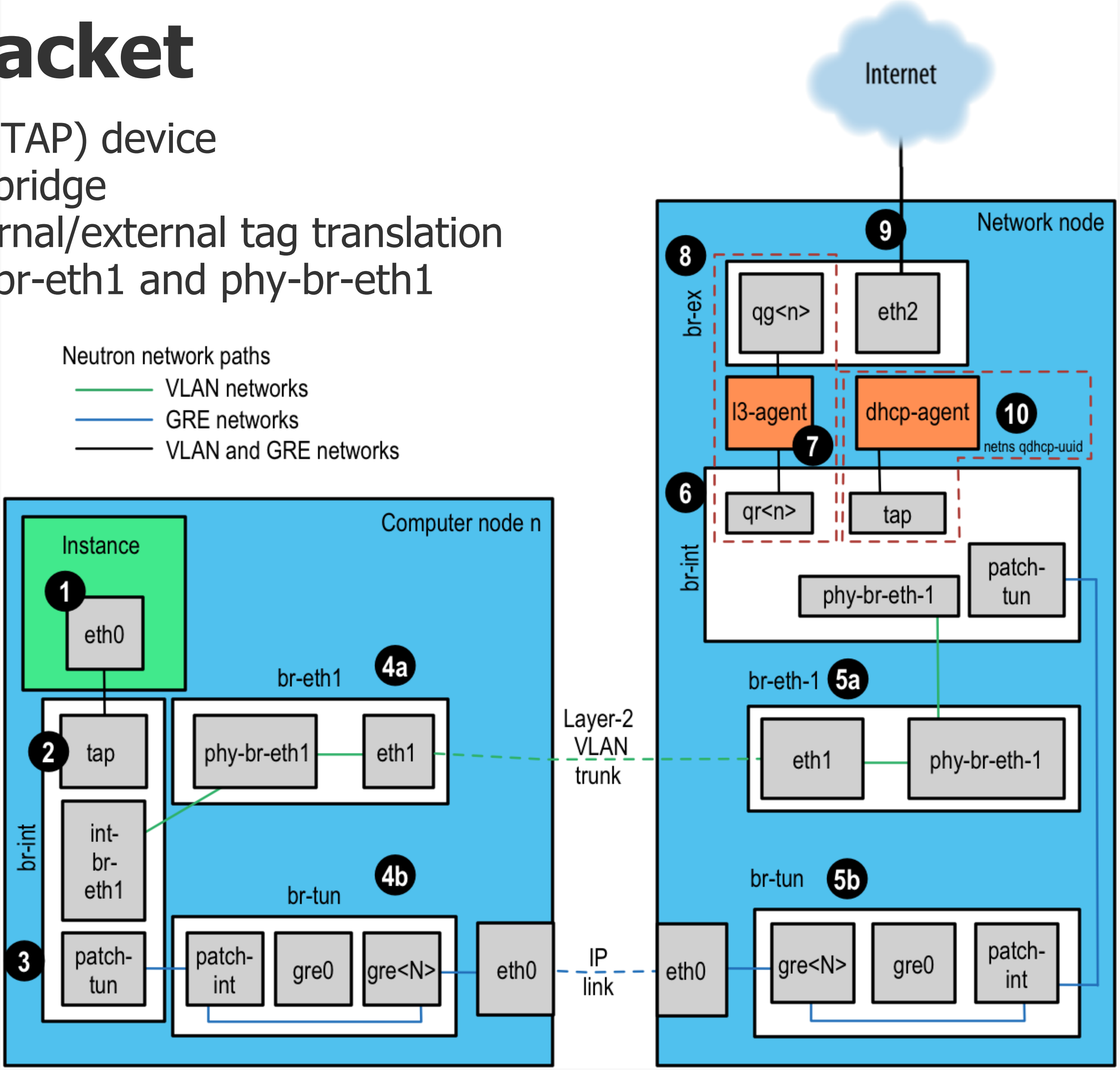

TenantA VM1 30.0.0.3 10.0.0.2
TenantA VM2 30.0.0.4 10.0.0.3
TenantC VM1 30.0.0.5 10.0.1.2
TenantC VM2 10.0.1.3
TenantA-Private Net 10.0.0.0/24
TenantC Private Net 10.0.1.0/24
10.0.0.1
10.0.1.1
Provider Router
30.0.0.2
External Net 30.0.0.0/22
30.0.0.1
Physical Router

Neutron: tenant routers



Path of a packet

- » Test Access Point (TAP) device
- » int-br: integration bridge
- » br-eth1: VLAN internal/external tag translation
- » veth: between int-br-eth1 and phy-br-eth1



Floating IP

- Neutron router
  - gateway for VMs
  - iptables/NAT rules in the namespace of router
    - nova network: in hypervisor
  - floating IP addresses allocated form the public network address range

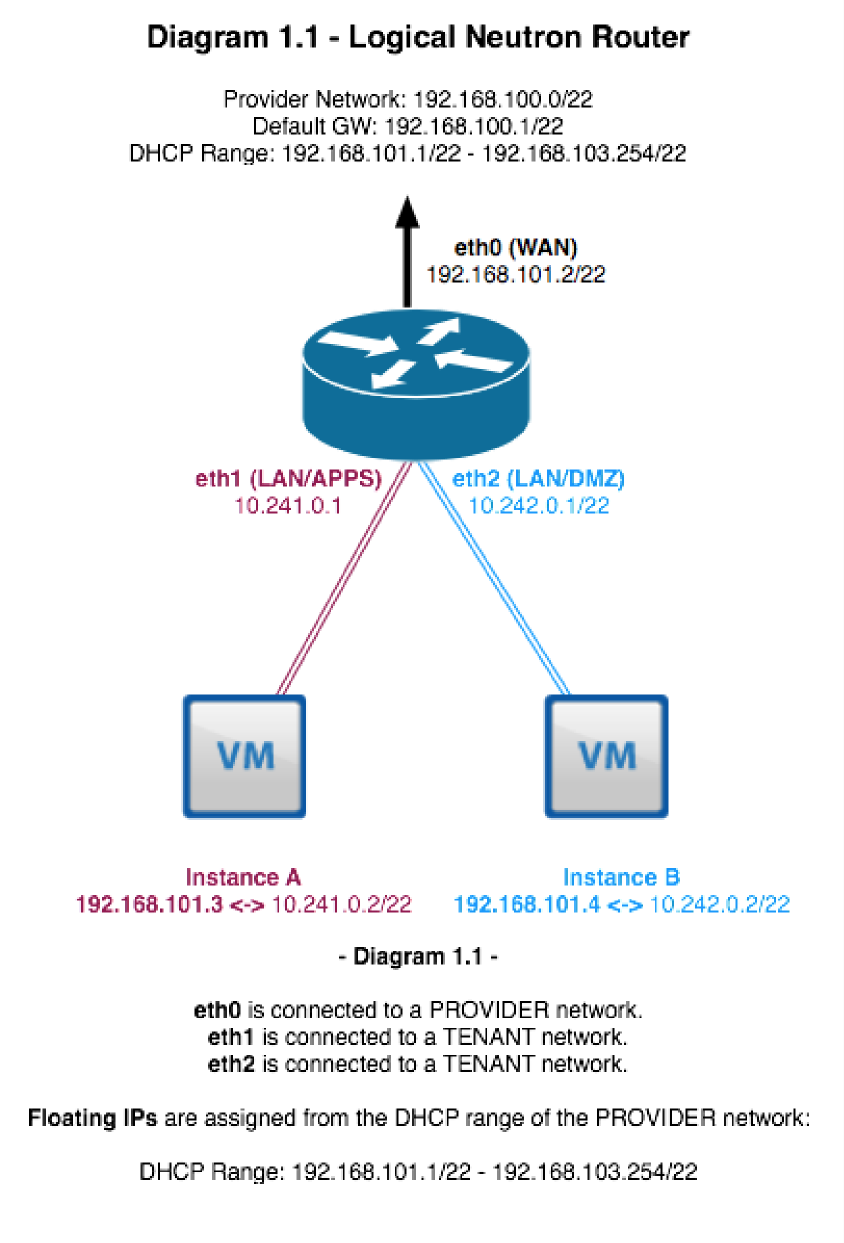


Diagram 1.1 - Logical Neutron Router

Provider Network: 192.168.100.0/22
Default GW: 192.168.100.1/22
DHCP Range: 192.168.101.1/22 - 192.168.103.254/22

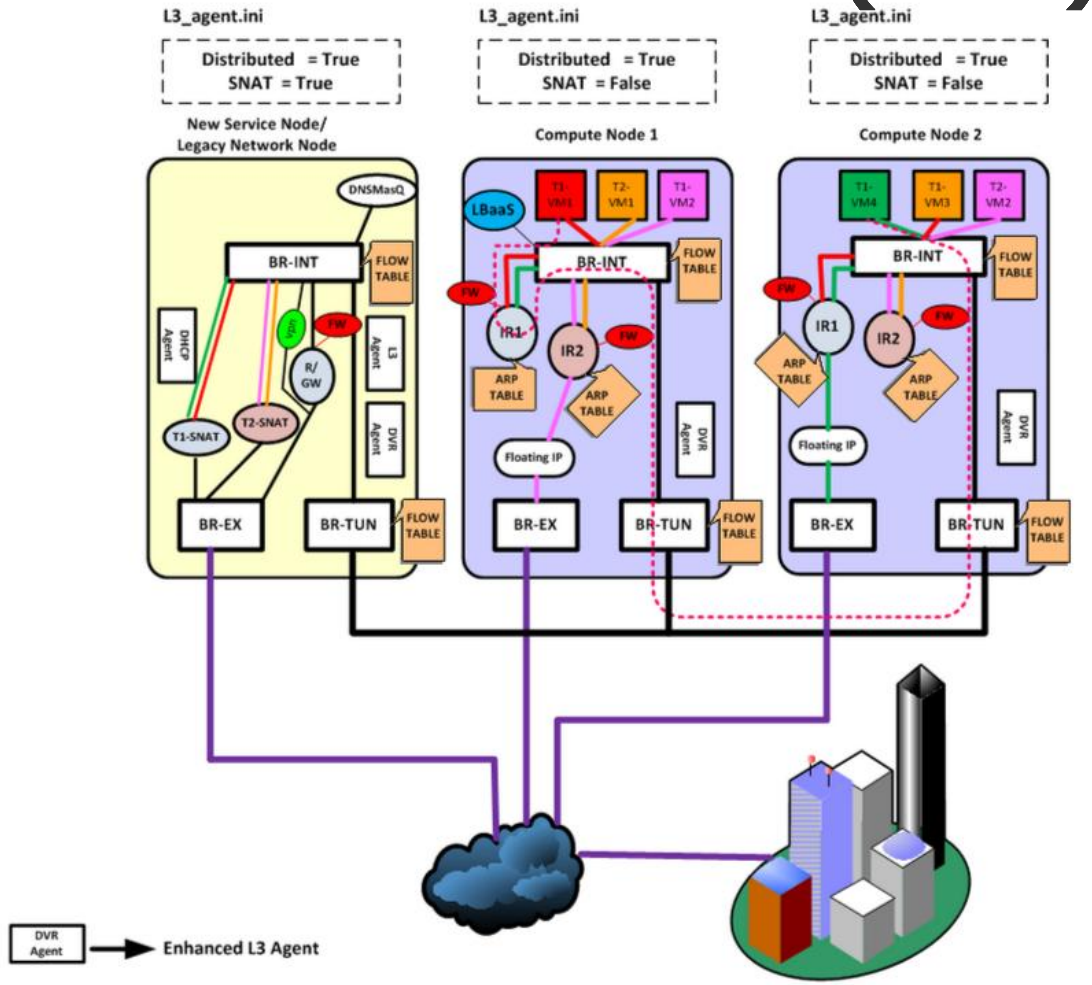
- Diagram 1.1 -

eth0 is connected to a PROVIDER network.
eth1 is connected to a TENANT network.
eth2 is connected to a TENANT network.

Floating IPs are assigned from the DHCP range of the PROVIDER network:

DHCP Range: 192.168.101.1/22 - 192.168.103.254/22

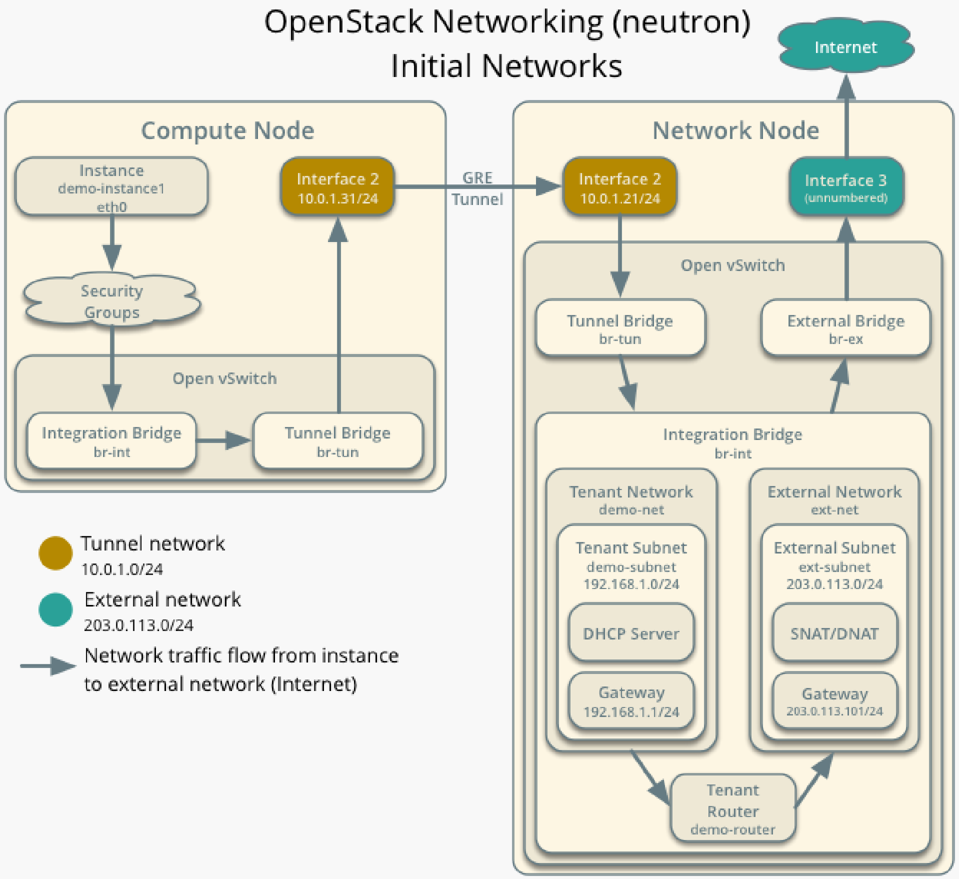
Distributed Virtual Router (DVR)



🟥🟩 TENANT A – With Distributed Routers and the Green VM in Compute Node 2 configured with Floating IP. The Default SNAT is provided by the Service Node

🟧🟪 TENANT B – With Distributed Routers and the Pink VM in Compute Node 1 configured with Floating IP. The Default SNAT is provided by the Service Node

Virtual network configuration



- » Open vSwitch
  - » setup by ovs-dpctl / OpenFlow
    - » e.g. mapping VM MAC address and hypervisor transport IP address

References

- Overlay Virtual Networking Explained, Ivan Pepelnjak, NIL Data Communications, 2011.
- http://docs.openstack.org
- https://developer.rackspace.com/blog/neutron-networking-l3-agent/
- https://www.rdoproject.org/Networking_in_too_much_detail