

Neutron, Where did my packets go ?

Mohamed Elsakhawy
SHARCNET / Compute Canada



SHARCNET/Compute Canada

- High Performance Computing
- Cloud Usage
 - HPC
 - Public/Private clouds
- ~7,000 core cloud deployments based on openstack
 - Arbutus
 - Located in Victoria, BC

Why this session ?

Neutron is a core component

Neutron is a hard-to-follow component

Understanding is key to troubleshooting

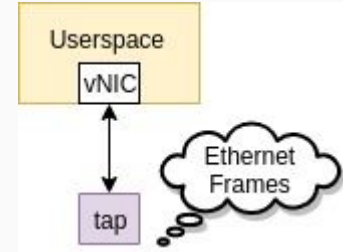
Before we start

Five main concepts/Linux virtual network devices

- TAP device
- veth pair
- Linux bridge
- Openvswitch
- Network namespaces

TAP device

- Software-only interface. i.e virtual
- Attached to a user-space program
- Accepts Ethernet frames



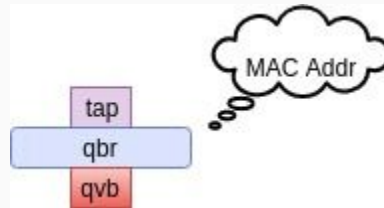
veth pair

- Two virtual NICs connected via a virtual wire
- Used to connect multiple virtual networking components
- Can connect network namespaces



Linux Bridge

- A virtual switch
- Physical and virtual interfaces
- Layer 2



Openvswitch

- More complicated virtual switch
- Virtual and physical interfaces
- Layer 2
- Openflow rules

Network Namespaces

- Isolated network stack
- Interfaces can be assigned to namespaces
- Separate routing tables
- 2 interfaces assigned to 2 different namespaces can have the same IP
- Neutron uses it to implement
 - Dhcp-agent
 - l3-agent

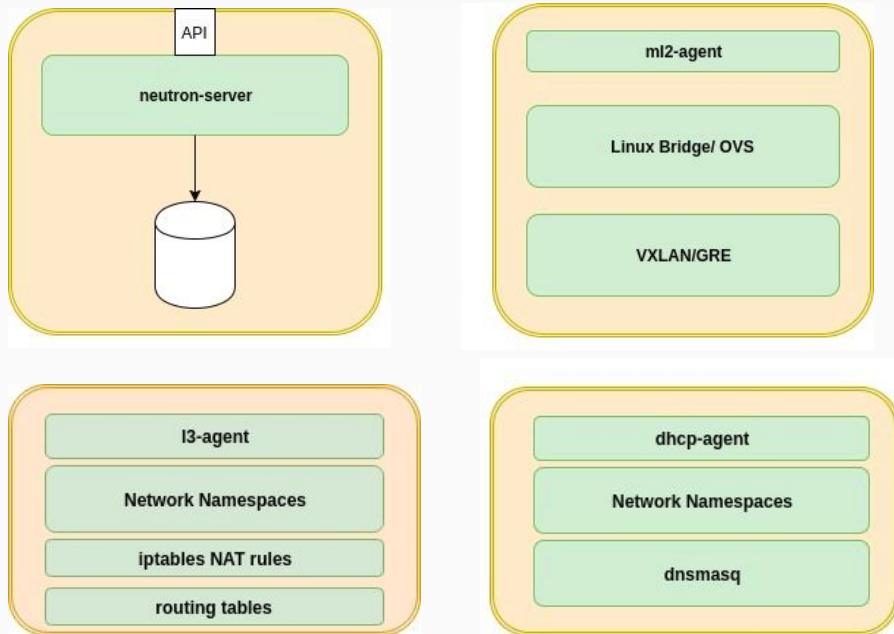
What does neutron do ?

- Allows user to create networks and routers
- Provides an api and manages the database
- Routes L3 traffic
- Provide floating IPs
- Switches L2 traffic
- Provides DHCP to instances
- Provides Metadata to instances

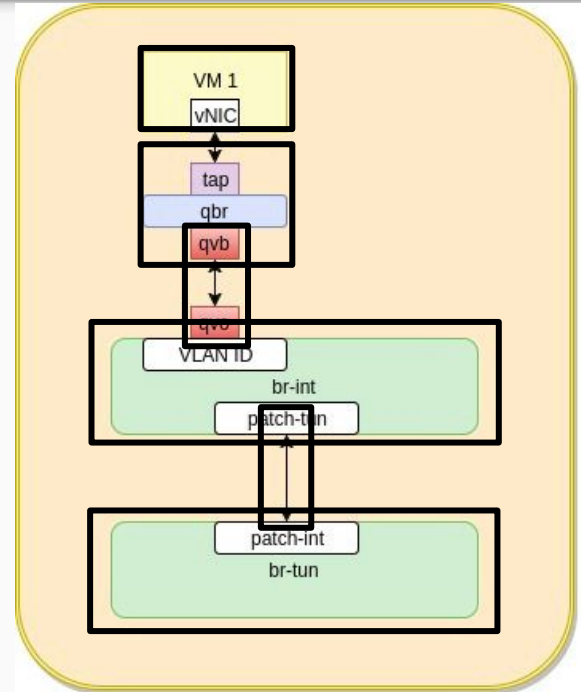
Which parts does that ?

- Allows user to create networks and routers
 - Provides an api and manages the database
 - Routes L3 traffic
 - Provide floating IPs
 - Switches L2 traffic
 - Provides DHCP to instances
 - Provides Metadata to instances
- neutron-server
- l3-agent
- ml2-agent
- dhcp-agent
- metadata-agent
-
- ```
graph LR; A[Allows user to create networks and routers] --- NS[neutron-server]; B[Provides an api and manages the database] --- NS; C[Routes L3 traffic] --- L3[l3-agent]; D[Provide floating IPs] --- L3; E[Switches L2 traffic] --> ML2[ml2-agent]; F[Provides DHCP to instances] --> DHCP[dhcp-agent]; G[Provides Metadata to instances] --> META[metadata-agent];
```

# Meet neutron...

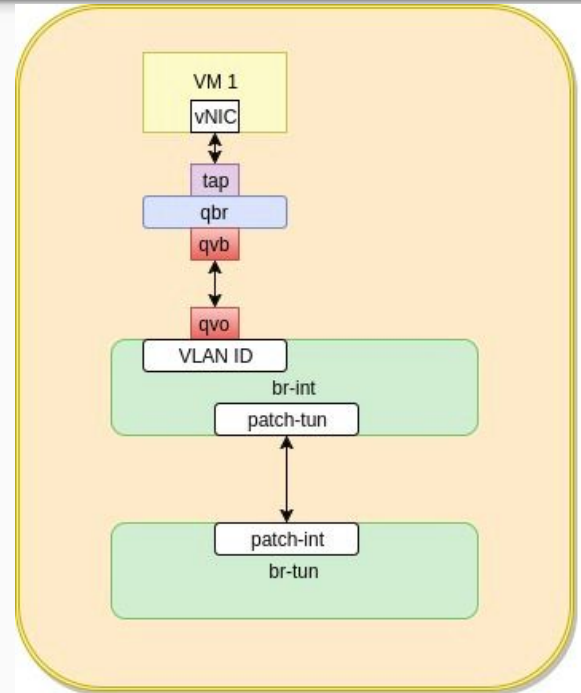


# A typical VM looks like this



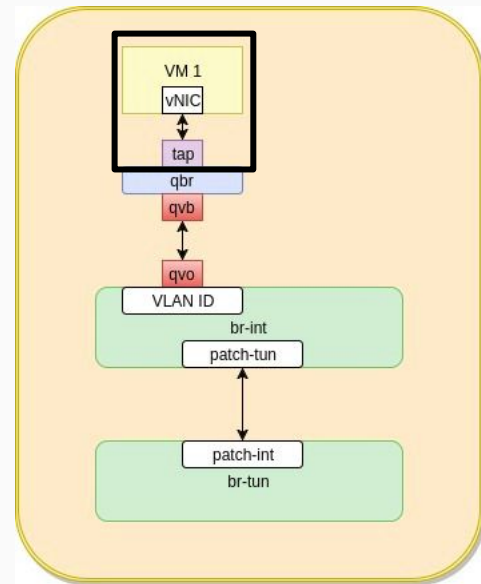
# A typical VM looks like this

- VM1 has vNIC
- IPtables rules applied to tap
- qbr is a linux bridge
  - connects tap and qvb
- qvb and qvo are veth pair
- br-int and br-tun
  - OVS switches
  - Connected via patch



# Physically on a compute node - KVM process

```
[root@compute-node ~]# ps -ef | grep fd
qemu 5540 1 4 16:52 ? 00:07:04 /usr/libexec/qemu-kvm -name guest=instance-00000004,debug-threads=on -S -object secret,id=masterKey0,format=raw,file=/var/lib/libvirt/qemu/domain-8-instance-00000004/master-key.aes -machine pc-i440fx-rhel7.3.0,accel=tcg,usb=off -m 512 -realtime mlock=off -smp 1,sockets=1,cores=1,threads=1 -uuid 1c81ec0e-0afa-47d7-8438-c3ca73085f80 -smbios type=1,manufacturer=RD0,product=OpenStack Compute,version=15.0.0-1.el7,serial=56d65f49-3d8f-8545-34fe-792ecf014b83,uuid=1c81ec0e-0afa-47d7-8438-c3ca73085f80,family=Virtual Machine -no-user-config -nodefaults -chardev socket,id=charmonitor,path=/var/lib/libvirt/qemu/domain-8-instance-00000004/monitor.sock,server,nowait -mon chardev=charmonitor,id=monitor,mode=control -rtc base=utc -no-shutdown -boot strict=on -device piix3-usb-uhci,id=usb,bus=pci.0,addr=0x1.0x2 -drive file=/var/lib/nova/instances/1c81ec0e-0afa-47d7-8438-c3ca73085f80/disk,format=qcow2,if=none,id=drive-virtio-disk0,cache=none -device virtio-blk-pci,scsi=off,bus=pci.0,addr=0x4,drive=drive-virtio-disk0,id=virtio-disk0,bootindex=1 -netdev tap,fd=29,id=hostnet0 -device virtio-net-pci,netdev=hostnet0,id=net0,mac=fa:16:3e:67:ed:ec,bus=pci.0,addr=0x3 -add-fd set=1,fd=32 -chardev file,id=charserial0,path=/dev/fdset/1,append=on -device isa-serial,chardev=charserial0,id=serial0 -chardev pty,id=charserial1 -device isa-serial,chardev=charserial1,id=serial1 -device usb-tablet,id=input0,bus=usb.0,port=1 -vnc 0.0.0.0:0 -k en-us -device cirrus-vga,id=video0,bus=pci.0,addr=0x2 -device virtio-balloon-pci,id=balloon0,bus=pci.0,addr=0x5 -msg timestamp=on
```

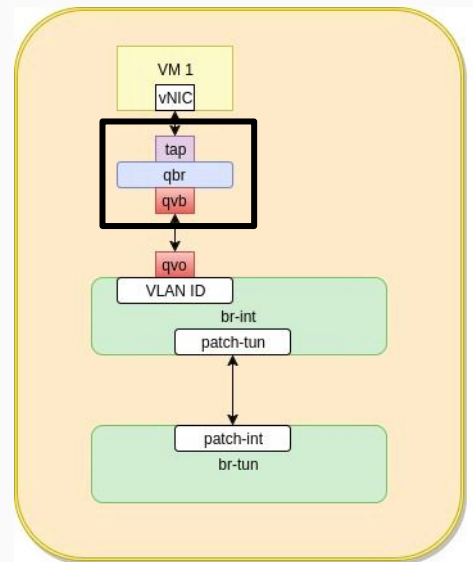


# Physically on a compute node - qbr bridge

```
[root@compute-node ~]# iptables -L | grep tapd874fb2a-b4
neutron-openvswi-sg-chain all -- anywhere anywhere PHYSDEV match --physdev-out tapd874fb2a-b4 --physdev-is-bridged /* Direct traffic from the VM interface to the security group chain. */
neutron-openvswi-sg-chain all -- anywhere anywhere PHYSDEV match --physdev-in tapd874fb2a-b4 --physdev-is-bridged /* Direct traffic from the VM interface to the security group chain. */
neutron-openvswi-od874fb2a-b all -- anywhere anywhere PHYSDEV match --physdev-in tapd874fb2a-b4 --physdev-is-bridged /* Direct incoming traffic from VM to the security group chain. */
neutron-openvswi-id874fb2a-b all -- anywhere anywhere PHYSDEV match --physdev-out tapd874fb2a-b4 --physdev-is-bridged /* Jump to the VM specific chain. */
neutron-openvswi-od874fb2a-b all -- anywhere anywhere PHYSDEV match --physdev-in tapd874fb2a-b4 --physdev-is-bridged /* Jump to the VM specific chain. */
```

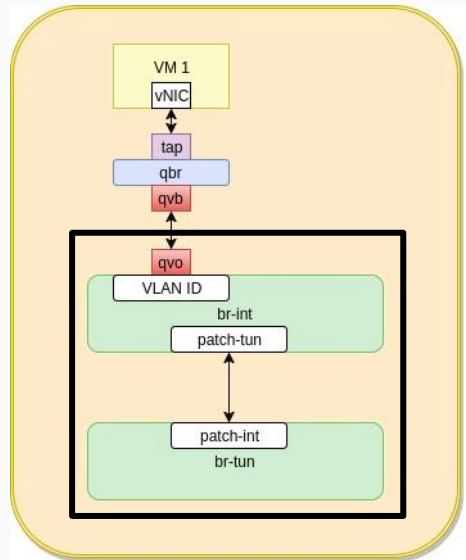
```
[root@compute-node ~]# brctl show
bridge name bridge id STP enabled interfaces
qbrd874fb2a-b4 8000.92f7a0175495 no qvbd874fb2a-b4
 tapd874fb2a-b4
```

- A qbr bridge created per instance with two ports
  - tap
  - qvb
- IPtables rules applied on the tap device



# Physically on a compute node: br-int & br-tun

```
[root@compute-node ~]# ovs-vsctl show
759be4f0-c8a6-4986-8927-7ab277d3934f
 Manager "ptcp:6640:127.0.0.1"
 is_connected: true
 Bridge br-tun
 Controller "tcp:127.0.0.1:6633"
 is_connected: true
 fail_mode: secure
 Port br-tun
 Interface br-tun
 type: internal
 Port "vxlan-c0a80204"
 Interface "vxlan-c0a80204"
 type: vxlan
 options: {df_default="true", in_key=flow, local_ip="192.168.2.5", out_key=flow, remote_ip="192.168.2.4"}
 Port patch-int
 Interface patch-int
 type: patch
 options: {peer=patch-tun}
 Bridge br-int
 Controller "tcp:127.0.0.1:6633"
 is_connected: true
 fail_mode: secure
 Port patch-tun
 Interface patch-tun
 type: patch
 options: {peer=patch-int}
 Port "qvod874fb2a-b4"
 tag: 1
 Interface "qvod874fb2a-b4"
 Port br-int
 Interface br-int
 type: internal
 ovs_version: "2.6.1"
```



# Traffic in Neutron

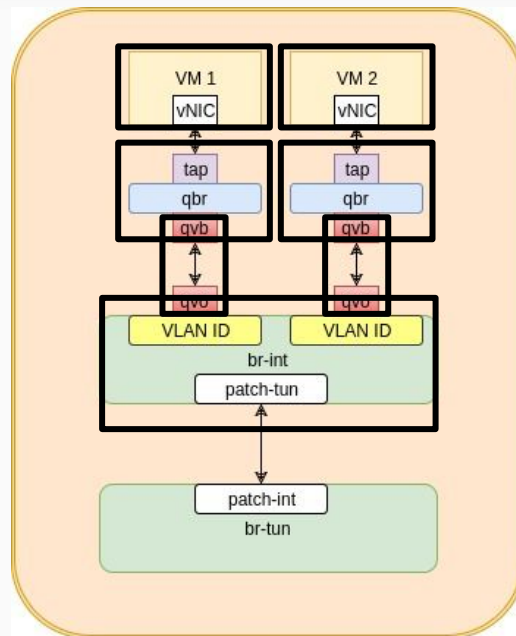
- Instance <-> Instance , same network
  - Instances living on the same host
  - Instances living on different hosts
- Instance <-> Instance, different networks
- Instance <-> dhcp
- Instance <-> public network, i.e. Internet
- User <-> Instance, i.e. floating IPs

# Scenario 1: Instance <-> Instance

- same network
- **same host**

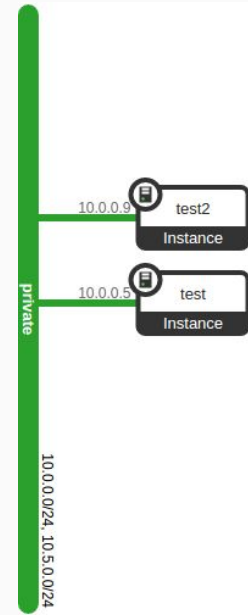
VM1 / VM2

- Same tenant
- Same network
- Same VLAN ID
- Packets flow through br-int



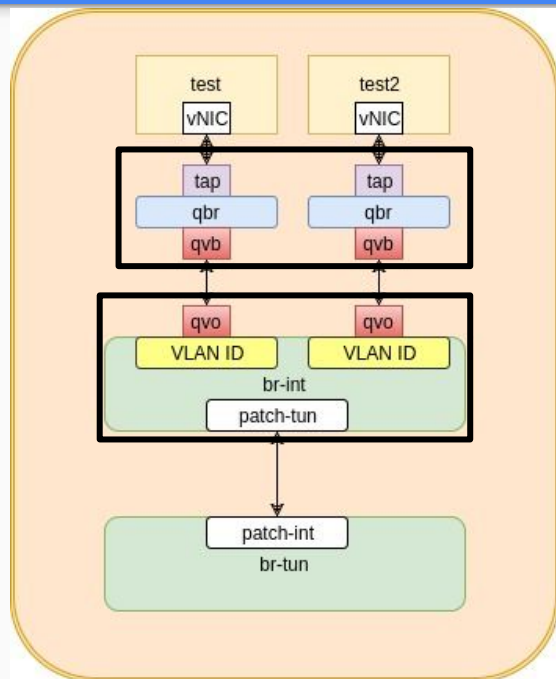
VM “test” IP 10.0.0.5  
VM “test2” IP 10.0.0.9

- 2 VMs : test and test2
- Connected to network “private”
- Belong to the same subnet
- **Same host**



# VMs Test & Test2: same network, same host

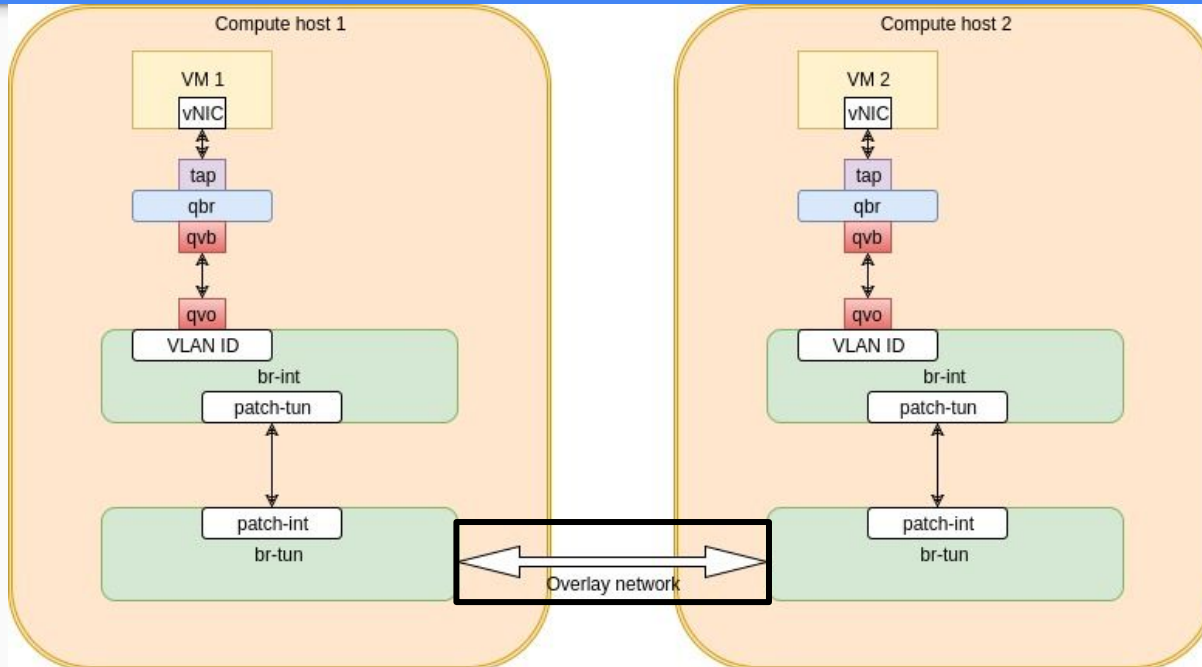
Same VLAN ID, traffic flows normally



```
[root@compute-node ~]# brctl show
bridge name bridge id STP enabled interfaces
qbrd874fb2a-b4 8000.92f7a0175495 no qvbd874fb2a-b4
 tapd874fb2a-b4
qbrd9b4bf25-c0 8000.a26adea75fea no qvbd9b4bf25-c0
 tapd9b4bf25-c0
```

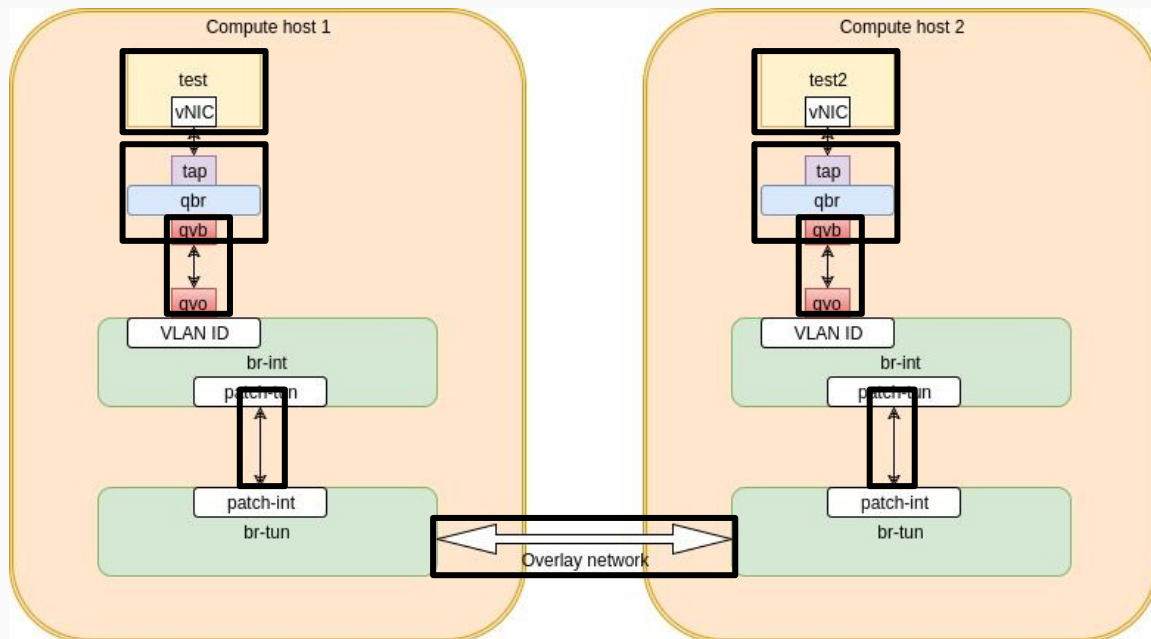
```
Bridge br-int
Controller "tcp:127.0.0.1:6633"
 is_connected: true
fail_mode: secure
Port patch-tun
 Interface patch-tun
 type: patch
 options: {peer=patch-int}
Port "qvod9b4bf25-c0"
 tag: 1
 Interface "qvod9b4bf25-c0"
Port "qvod874fb2a-b4"
 tag: 1
 Interface "qvod874fb2a-b4"
Port br-int
 Interface br-int
 type: internal
ovs_version: "2.6.1"
```

# This was same host. How about different hosts ?



# Scenario 1: Instance <-> Instance

- same network
- **different hosts**

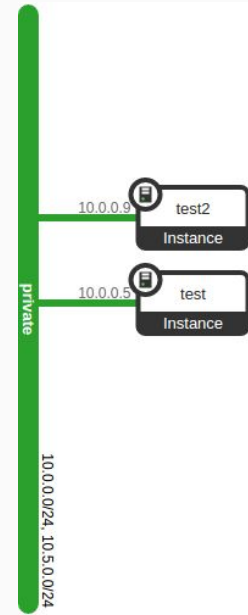


# Two questions !

- In the overlay network: How does different tenant networks traffic keep isolated ?
  - br-tun
    - VLAN ID <-> VXLAN tunnel ID
    - VXLAN tunnel ID unique per network
- Isn't it useless to send the traffic to every host ?
  - br-tun learns !
    - Which host has what VM it
    - Updates the flows

VM “test” IP 10.0.0.5  
VM “test2” IP 10.0.0.9

- 2 VMs : test and test2
- Connected to network “private”
- **Different hosts**



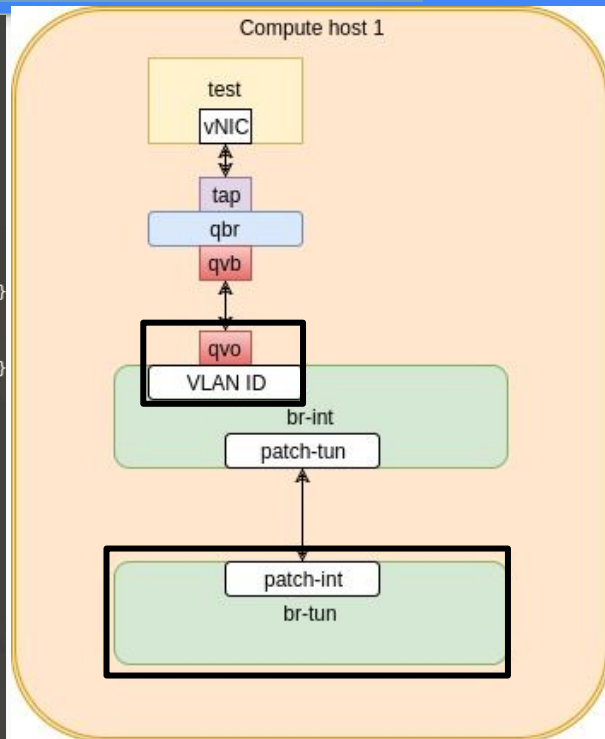
# Compute host 1

VM: test IP 10.0.0.5

VLAN ID removed outbound, TUN ID set to 0x39  
VLAN ID "1" added inbound if TUN ID =0x39

```
[root@compute-host1 ~]# ovs-ofctl dump-flows br-tun | grep vlan
cookie=0xb365cc0fbd01a814, duration=3258.329s, table=4, n_packets=1056, n_bytes=107443, idle_age=2882, priority=1, tun_id=0x39 actions=mod_vlan vid:1, resubmit(,10)
cookie=0xb365cc0fbd01a814, duration=3258.330s, table=22, n_packets=379, n_bytes=25614, idle_age=2885, priority=1, dl_vlan=1 actions=strip_vlan, load:0x39->NXM_NX_TUN_ID[], output:2, output:4
```

```
[root@compute-host1 ~]# ovs-vsctl show
759be4f0-c8a6-4986-8927-7ab277d3934f
 Manager "ptcp:6640:127.0.0.1"
 is_connected: true
 Bridge br-tun
 Controller "tcp:127.0.0.1:6633"
 is_connected: true
 fail_mode: secure
 Port br-tun
 Interface br-tun
 type: internal
 Port "vxlan-c0a80206"
 Interface "vxlan-c0a80206"
 type: vxlan
 options: {df_default="true", in_key=flow, local_ip="192.168.2.5", out_key=flow, remote_ip="192.168.2.6"}
 Port "vxlan-c0a80204"
 Interface "vxlan-c0a80204"
 type: vxlan
 options: {df_default="true", in_key=flow, local_ip="192.168.2.5", out_key=flow, remote_ip="192.168.2.4"}
 Port patch-int
 Interface patch-int
 type: patch
 options: {peer=patch-tun}
 Bridge br-int
 Controller "tcp:127.0.0.1:6633"
 is_connected: true
 fail_mode: secure
 Port patch-tun
 Interface patch-tun
 type: patch
 options: {peer=patch-int}
 Port "qvod874fb2a-b4"
 tag: 1
 Interface "qvod874fb2a-b4"
 Port br-int
 Interface br-int
 type: internal
 ovs version: "2.6.1"
```



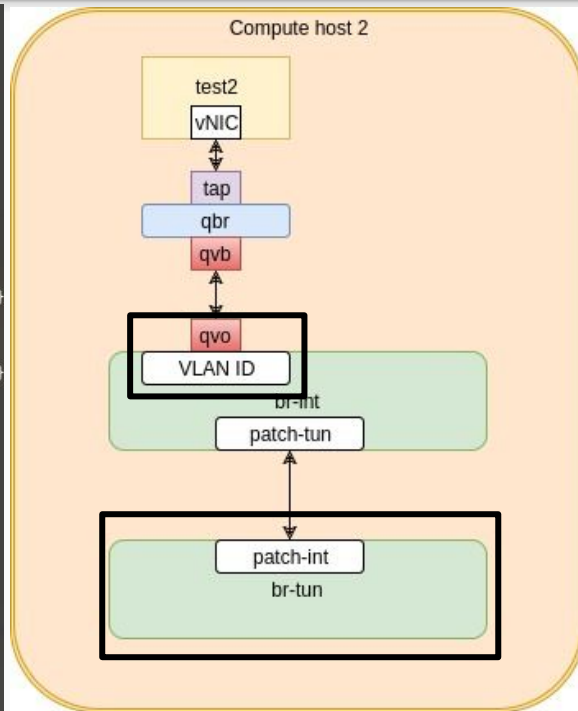
# Compute host 2

VM: test2 IP 10.0.0.9

VLAN ID removed outbound, TUN ID set to 0x39  
VLAN ID "2" added inbound, if TUN ID = 0x39

```
[root@compute-host2 ~]# ovs-ofctl dump-flows br-tun | grep vlan
cookie=0xa739bab04ac91e4f, duration=2452.920s, table=4, n_packets=83, n_bytes=8828, idle_age=2430, priority=1,tun_id=0x39 actions=mod_vlan_vid:2,resubmit(,10)
cookie=0xa739bab04ac91e4f, duration=2452.922s, table=22, n_packets=10, n_bytes=1228, idle_age=2433, priority=1,dl_vlan=2 actions=strip_vlan,load:0x39->NXM_NX_TUN_ID[],output:2,output:3
```

```
[root@compute-host2 ~]# ovs-vsctl show
ee374eac-67a5-49a4-9cd6-718bab04524f
 Manager "ptcp:6640:127.0.0.1"
 is_connected: true
 Bridge br-tun
 Controller "tcp:127.0.0.1:6633"
 is_connected: true
 fail_mode: secure
 Port patch-int
 Interface patch-int
 type: patch
 options: {peer=patch-tun}
 Port "vxlan-c0a80205"
 Interface "vxlan-c0a80205"
 type: vxlan
 options: {df_default="true", in_key=flow, local_ip="192.168.2.6", out_key=flow, remote_ip="192.168.2.5"}
 Port "vxlan-c0a80204"
 Interface "vxlan-c0a80204"
 type: vxlan
 options: {df_default="true", in_key=flow, local_ip="192.168.2.6", out_key=flow, remote_ip="192.168.2.4"}
 Port br-tun
 Interface br-tun
 type: internal
 Bridge br-int
 Controller "tcp:127.0.0.1:6633"
 is_connected: true
 fail_mode: secure
 Port "qvofeb87c75-d4"
 tag: 2
 Interface "qvofeb87c75-d4"
 Port br-int
 Interface br-int
 type: internal
 Port patch-tun
 Interface patch-tun
 type: patch
 options: {peer=patch-int}
 ovs version: "2.6.1"
```

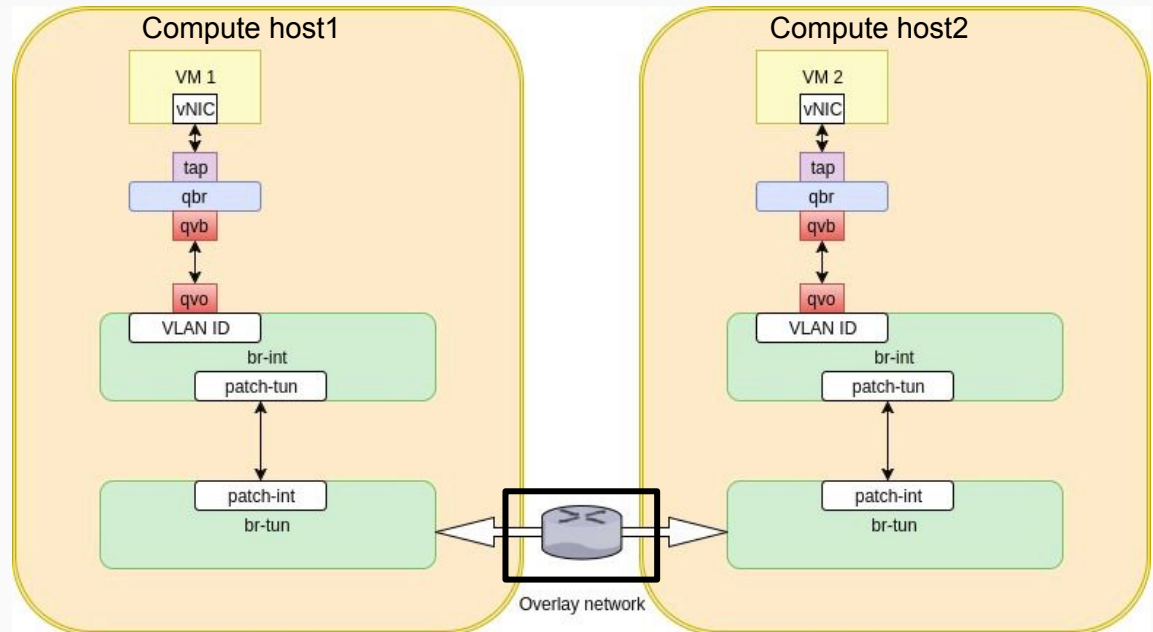


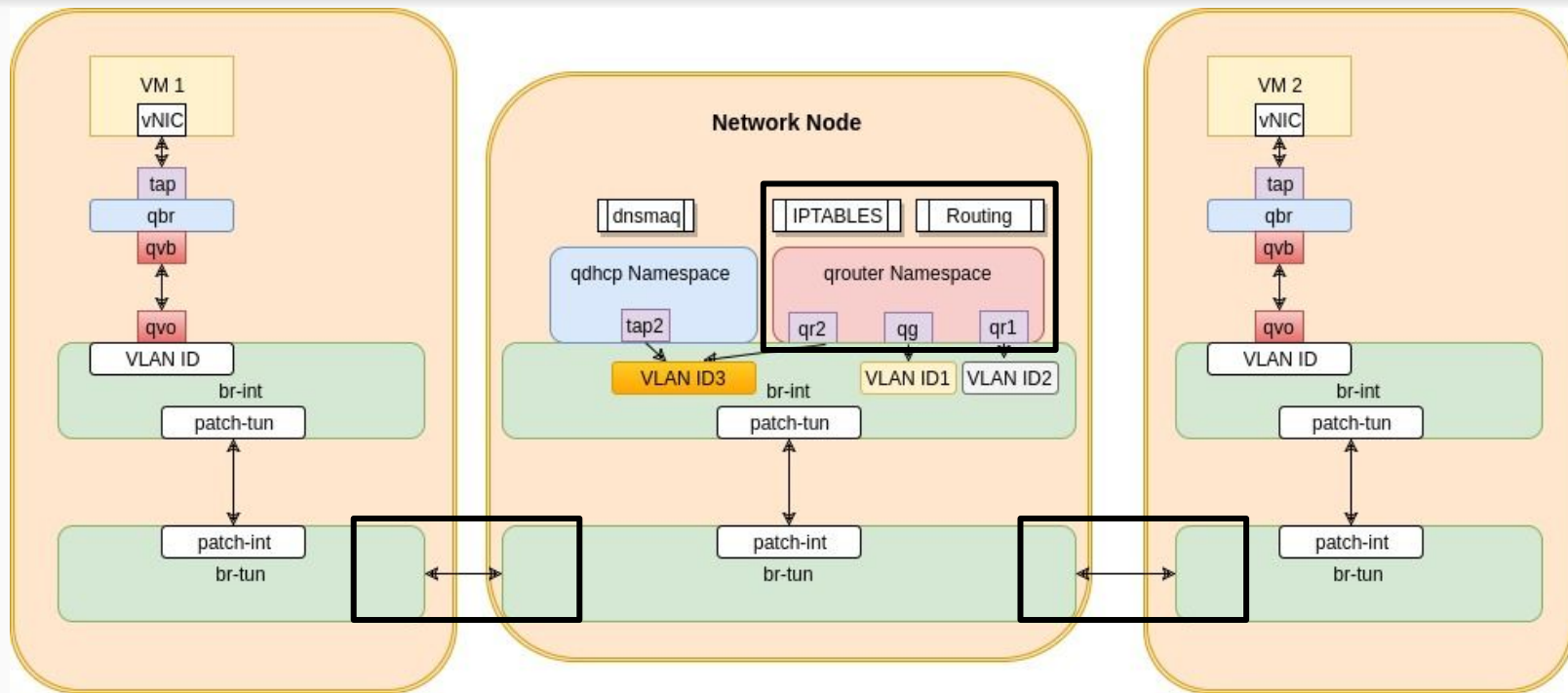
# Scenario 2: Instance <-> Instance

- different networks
- different hosts

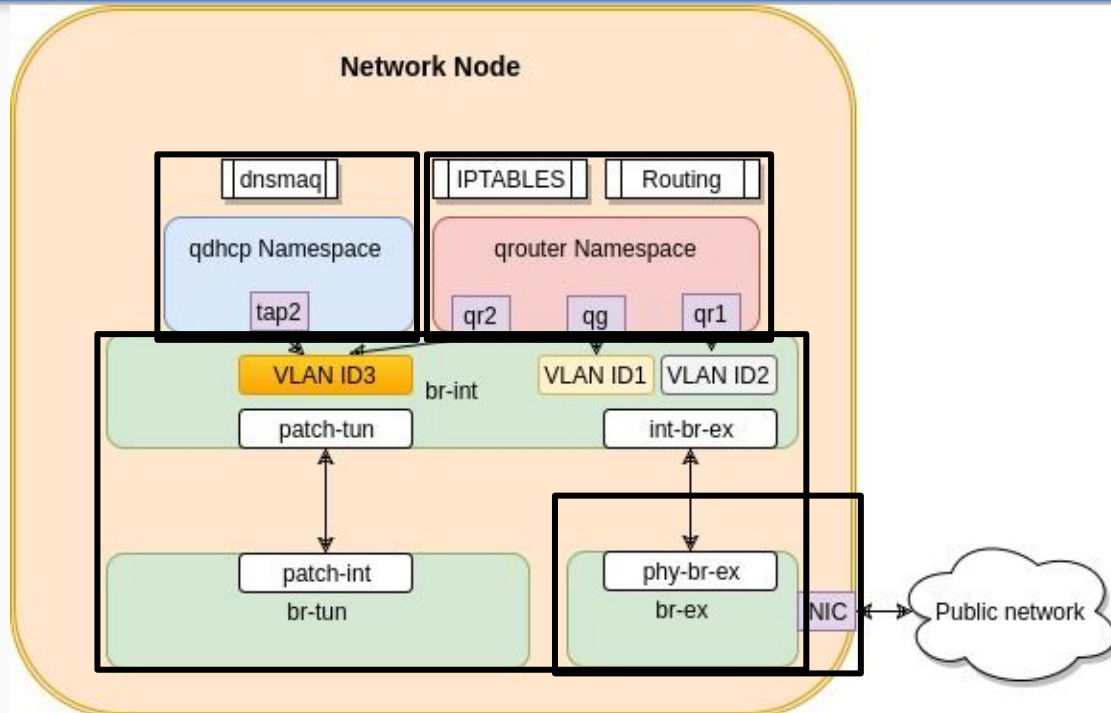
VM1 / VM2

- Different networks
  - i.e router
- Different hosts





# A network node



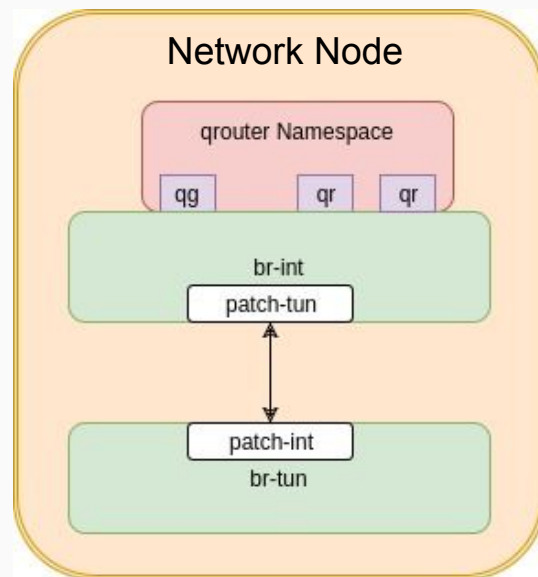
A dhcp-enabled network has a qdhcp namespace with a tap interface

A router has a qrouter namespace with a qr interface for each connected network

A router has one qg interface for its gateway

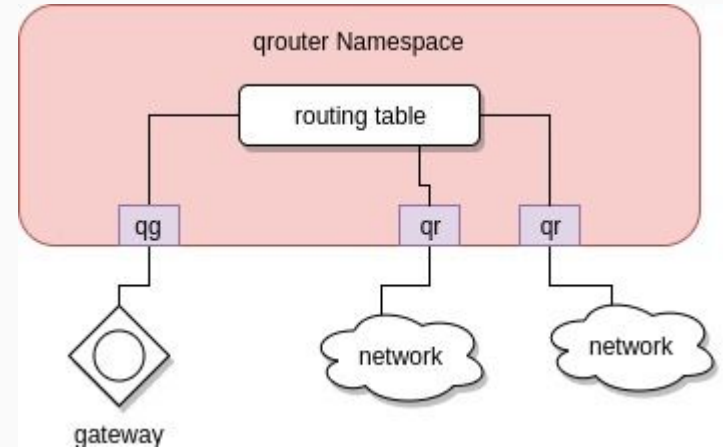
# How is the router implemented ?

- L3-agent
- Implemented inside network namespaces
  - (qrouter-xxxxxx)
- Multiple network interfaces
  - (qr-xxxx), one for each network subnet
- Routing rules for interfaces
- One gateway interface
  - (qg-xxxx), one for public gateway

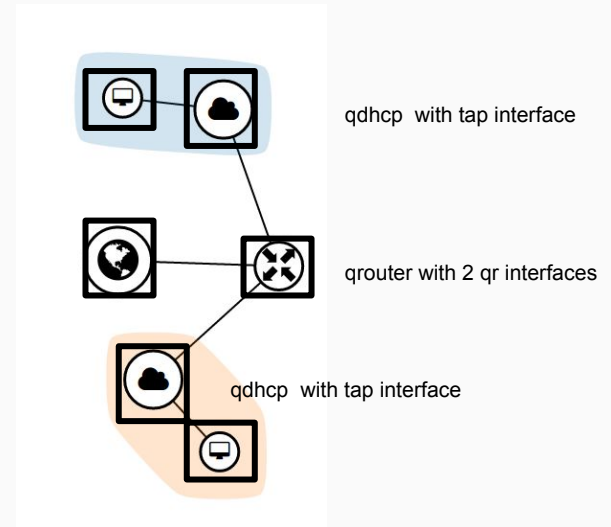
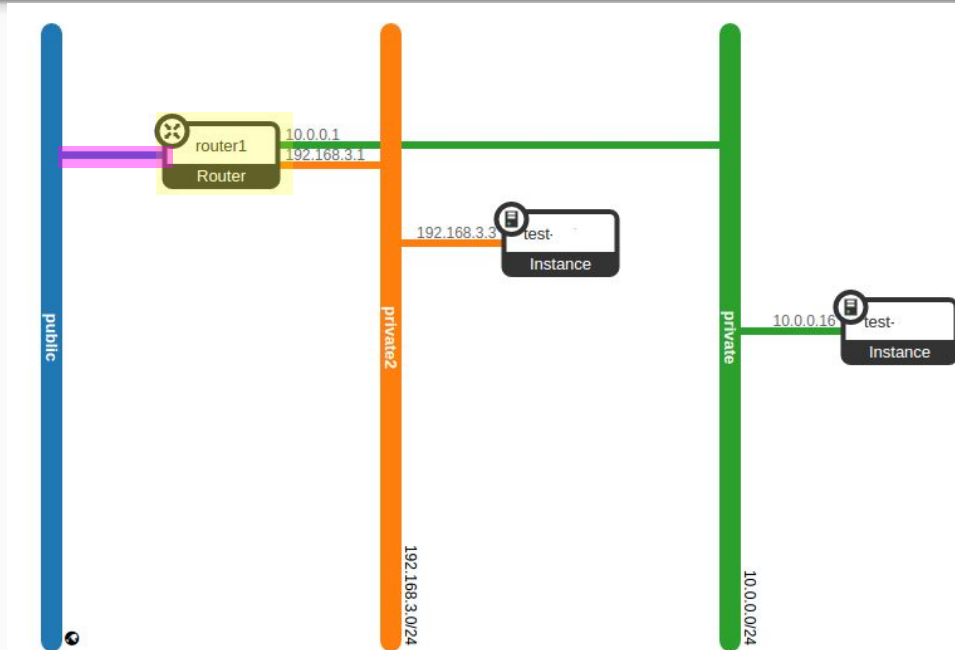


# qrouter namespace

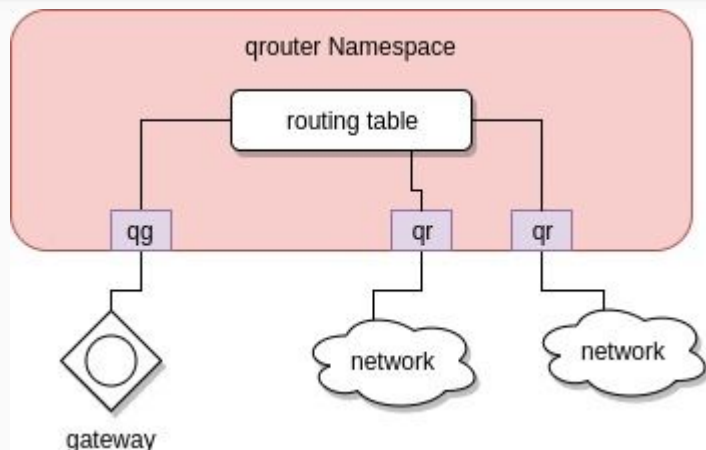
```
[root@network-node ~]# ip netns
qdhcp-60ce0ffb-00ad-444e-9406-d0f588be2634
qdhcp-fb4805f8-af76-4019-aec3-7b83193ba9cc
qrouter-69d4e7f6-6351-4790-b310-e4bbc8339d2f
```



# Two Instances, different networks, different hosts



# qrouter namespace



Router has 3 interfaces

qr-44edbf78-34: fa:16:3e:51:01:ce IP: 10.0.0.1  
qr-60828e45-4e: fa:16:3e:23:4f:b4 IP: 192.168.3.1  
qg-916bdf6a-a3: fa:16:3e:e9:2e:04 IP: 172.24.4.5

```
ip netns exec qrouter-69d4e7f6-6351-4790-b310-e4bbc8339d2f bash
[root@network-node ~]# ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1 (Local Loopback)
 RX packets 0 bytes 0 (0.0 B)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 0 bytes 0 (0.0 B)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

qg-916bdf6a-a3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
 inet 172.24.4.5 netmask 255.255.255.0 broadcast 172.24.4.255
 inet6 fe80::f816:3eff:fee9:2e04 prefixlen 64 scopeid 0x20<link>
 ether fa:16:3e:e9:2e:04 txqueuelen 1000 (Ethernet)
 RX packets 15 bytes 708 (708.0 B)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 166 bytes 15284 (14.9 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

qr-44edbf78-34: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1450
 inet 10.0.0.1 netmask 255.255.255.0 broadcast 10.0.0.255
 inet6 fe80::f816:3eff:fe51:1ce prefixlen 64 scopeid 0x20<link>
 ether fa:16:3e:51:01:ce txqueuelen 1000 (Ethernet)
 RX packets 1107 bytes 74288 (72.5 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 449 bytes 43024 (42.0 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

qr-60828e45-4e: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1450
 inet 192.168.3.1 netmask 255.255.255.0 broadcast 192.168.3.255
 inet6 fe80::f816:3eff:fe23:4fb4 prefixlen 64 scopeid 0x20<link>
 ether fa:16:3e:23:4f:b4 txqueuelen 1000 (Ethernet)
 RX packets 0 bytes 0 (0.0 B)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 10 bytes 864 (864.0 B)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
[root@network-node ~]# route -n
```

Kernel IP routing table

| Destination | Gateway    | Genmask       | Flags | Metric | Ref | Use | Iface          |
|-------------|------------|---------------|-------|--------|-----|-----|----------------|
| 0.0.0.0     | 172.24.4.1 | 0.0.0.0       | UG    | 0      | 0   | 0   | qg-916bdf6a-a3 |
| 0.0.0.0     | 0.0.0.0    | 255.255.255.0 | U     | 0      | 0   | 0   | qr-44edbf78-34 |
| 172.24.4.0  | 0.0.0.0    | 255.255.255.0 | U     | 0      | 0   | 0   | qg-916bdf6a-a3 |
| 192.168.3.0 | 0.0.0.0    | 255.255.255.0 | U     | 0      | 0   | 0   | qr-60828e45-4e |

# Compute host 1

VM1: IP 192.168.3.3

Router has 3 interfaces

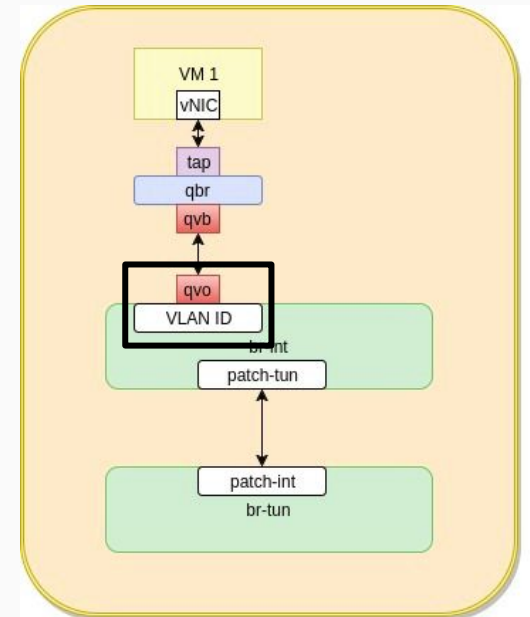
qr-44edbf78-34: fa:16:3e:51:01:ce IP: 10.0.0.1

qr-60828e45-4e: fa:16:3e:23:4f:b4 IP: 192.168.3.1

qg-916bdf6a-a3: fa:16:3e:e9:2e:04 IP: 172.24.4.5

```
[root@compute-host1 ~]# ovs-ofctl dump-flows br-tun | grep dst
cookie=0xb365cc8fbd01a814, duration=7846.106s, table=2, n_packets=1404, n_bytes=137571, idle_age=125, priority=0,dl_dst=00:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,20)
cookie=0xb365cc8fbd01a814, duration=7846.105s, table=2, n_packets=513, n_bytes=34770, idle_age=128, priority=0,dl_dst=01:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,22)
cookie=0xb365cc8fbd01a814, duration=137.726s, table=20, n_packets=7, n_bytes=552, hard_timeout=300, idle_age=125, hard_age=125, priority=1,vlan_tci=0x0002/0x0fff,dl_dst=fa:16:3e:f6:64:27 actions=load:0->NXM_OF_VLAN_TCI[],load:0x43->NXM_N
X_TUN_ID[],output:2
cookie=0xb365cc8fbd01a814, duration=136.780s, table=20, n_packets=91, n_bytes=8818, hard_timeout=300, idle_age=128, hard_age=128, priority=1,vlan_tci=0x0002/0x0fff,dl_dst=fa:16:3e:23:4f:b4 actions=load:0->NXM_OF_VLAN_TCI[],load:0x43->NXM_N
X_TUN_ID[],output:2
```

```
[root@compute-host1 ~]# ovs-vsctl show
759be4f0-c8a6-4986-8927-7ab277d3934f
 Manager "ptcp:6640:127.0.0.1"
 is_connected: true
 Bridge br-tun
 Controller "tcp:127.0.0.1:6633"
 is_connected: true
 fail_mode: secure
 Port br-tun
 Interface br-tun
 type: internal
 Port "vxlan-c0a80206"
 Interface "vxlan-c0a80206"
 type: vxlan
 options: {df_default="true", in_key=flow, local_ip="192.168.2.5", out_key=flow, remote_ip="192.168.2.6"}
 Port "vxlan-c0a80204"
 Interface "vxlan-c0a80204"
 type: vxlan
 options: {df_default="true", in_key=flow, local_ip="192.168.2.5", out_key=flow, remote_ip="192.168.2.4"}
 Port patch-int
 Interface patch-int
 type: patch
 options: {peer=patch-tun}
 Bridge br-int
 Controller "tcp:127.0.0.1:6633"
 is_connected: true
 fail_mode: secure
 Port patch-tun
 Interface patch-tun
 type: patch
 options: {peer=patch-int}
 Port "qvof829fcd5-01"
 tag: 2
 Interface "qvof829fcd5-01"
 Port br-int
 Interface br-int
 type: internal
 ovs version: "2.6.1"
```



# Compute host 2

VM2: IP 10.0.0.9

Router has 3 interfaces

qr-44edbf78-34: fa:16:3e:51:01:ce IP: 10.0.0.1

qr-60828e45-4e: fa:16:3e:23:4f:b4 IP: 192.168.3.1

qg-916bdf6a-a3: fa:16:3e:e9:2e:04 IP: 172.24.4.5

```
[root@compute-host2 ~]# ovs-ofctl dump-flows br-tun | grep dst
```

```
cookie=0xa739bab04ac91e4f, duration=14371.485s, table=2, n_packets=466, n_bytes=44024, idle_age=1, priority=0,dl_dst=00:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,20)
```

```
cookie=0xa739bab04ac91e4f, duration=14371.484s, table=2, n_packets=28, n_bytes=3084, idle_age=249, priority=0,dl_dst=01:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,22)
```

```
cookie=0xa739bab04ac91e4f, duration=249.369s, table=20, n_packets=268, n_bytes=25200, hard_timeout=300, idle_age=1, hard_age=0, priority=1,vlan_tci=0x0002/0x0fff,dl_dst=fa:16:3e:51:01:ce actions=load:0->NXM_OF_VLAN_TCI[],load:0x39->NXM_N
```

```
X_TUN_ID[],output:2
```

```
[root@compute-host2 ~]# ovs-vsctl show
```

```
ee374eac-67a5-49a4-9cd6-718babc4524f
```

```
Manager "ptcp:6640:127.0.0.1"
```

```
is_connected: true
```

```
Bridge br-tun
```

```
Controller "tcp:127.0.0.1:6633"
```

```
is_connected: true
```

```
fail_mode: secure
```

```
Port patch-int
```

```
Interface patch-int
```

```
type: patch
```

```
options: {peer=patch-tun}
```

```
Port "vxlan-c0a80205"
```

```
Interface "vxlan-c0a80205"
```

```
type: vxlan
```

```
options: {df_default="true", in_key=flow, local_ip="192.168.2.6", out_key=flow, remote_ip="192.168.2.5"}
```

```
Port "vxlan-c0a80204"
```

```
Interface "vxlan-c0a80204"
```

```
type: vxlan
```

```
options: {df_default="true", in_key=flow, local_ip="192.168.2.6", out_key=flow, remote_ip="192.168.2.4"}
```

```
Port br-tun
```

```
Interface br-tun
```

```
type: internal
```

```
Bridge br-int
```

```
Controller "tcp:127.0.0.1:6633"
```

```
is_connected: true
```

```
fail_mode: secure
```

```
Port "qvofeb87c75-d4"
```

```
tag: 2
```

```
Interface "qvofeb87c75-d4"
```

```
Port br-int
```

```
Interface br-int
```

```
type: internal
```

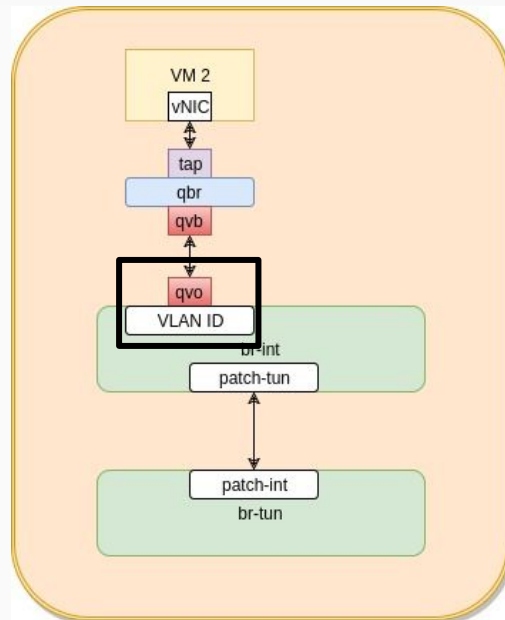
```
Port patch-tun
```

```
Interface patch-tun
```

```
type: patch
```

```
options: {peer=patch-int}
```

```
ovs_version: "2.6.1"
```



# On network node

Traffic arriving on Tunnel ID  
0x43 from VM1, IP 192.168.3.3  
0x39 from VM2, IP 10.0.0.9

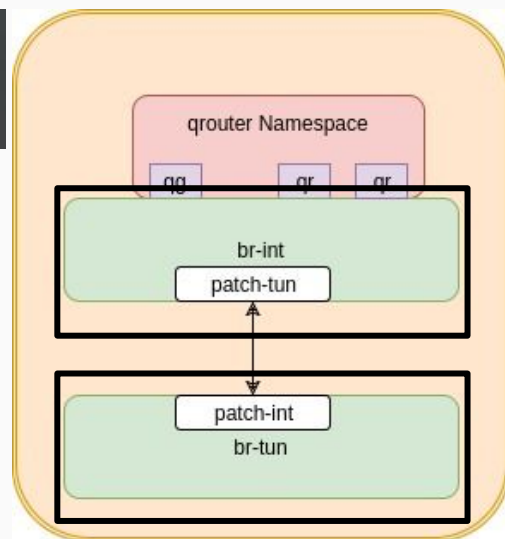
```
[root@network-node ~]# ovs-ofctl dump-flows br-tun | grep -e 0x43 -e 0x39 | grep mod
cookie=0xbff1e1e0196b03ec, duration=527783.129s, table=4, n_packets=2750, n_bytes=254819, idle_age=0, hard_age=65534, priority=1, tun_id=0x39 actions=mod_vlan_vid:2,resubmit(,10)
cookie=0xbff1e1e0196b03ec, duration=516530.241s, table=4, n_packets=1051, n_bytes=96238, idle_age=0, hard_age=65534, priority=1, tun_id=0x43 actions=mod_vlan_vid:3,resubmit(,10)
```

```
Bridge br-int
 Controller "tcp:127.0.0.1:6633"
 is connected: true
 fail_mode: secure
 Port "tap6e56ae9a-e8"
 tag: 3
 Interface "tap6e56ae9a-e8"
 type: internal
 Port "tap5c93757f-13"
 tag: 2
 Interface "tap5c93757f-13"
 type: internal
 Port "qg-916bdf6a-a3"
 tag: 1
 Interface "qg-916bdf6a-a3"
 type: internal
 Port br-int
 Interface br-int
 type: internal
 Port int-br-ex
 Interface int-br-ex
 type: patch
 options: {peer=phy-br-ex}
 Port patch-tun
 Interface patch-tun
 type: patch
 options: {peer=patch-int}
 Port "qr-44edbf78-34"
 tag: 2
 Interface "qr-44edbf78-34"
 type: internal
 Port "qr-60828e45-4e"
 tag: 3
 Interface "qr-60828e45-4e"
 type: internal
```

```
[root@network-node ~]# route -n
Kernel IP routing table

```

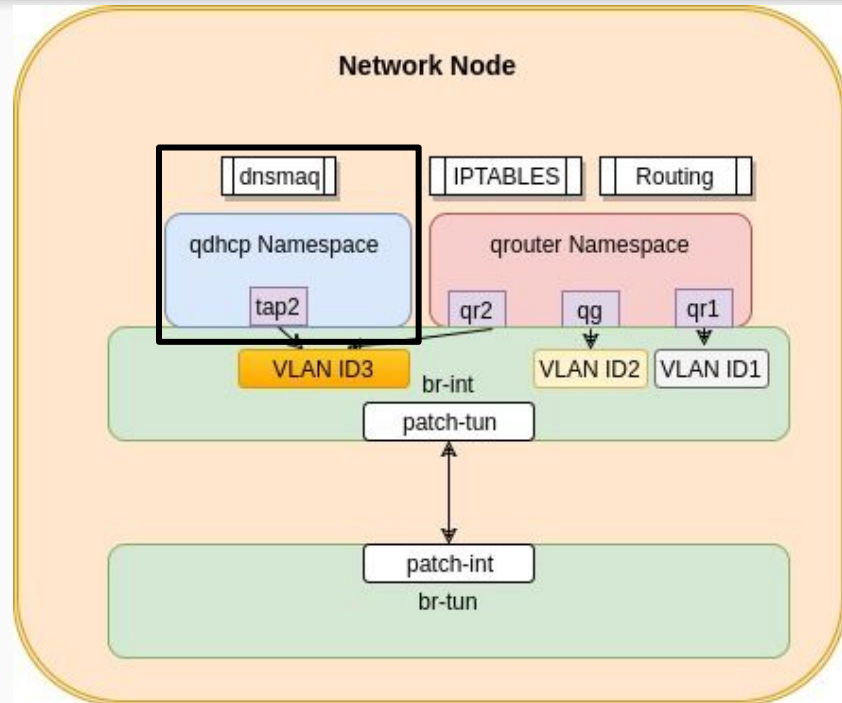
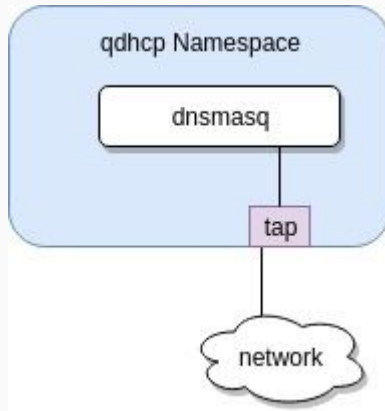
| Destination | Gateway    | Genmask       | Flags | Metric | Ref | Use | Iface          |
|-------------|------------|---------------|-------|--------|-----|-----|----------------|
| 0.0.0.0     | 172.24.4.1 | 0.0.0.0       | UG    | 0      | 0   | 0   | qg-916bdf6a-a3 |
| 10.0.0.0    | 0.0.0.0    | 255.255.255.0 | U     | 0      | 0   | 0   | qr-44edbf78-34 |
| 172.24.4.0  | 0.0.0.0    | 255.255.255.0 | U     | 0      | 0   | 0   | qg-916bdf6a-a3 |
| 192.168.3.0 | 0.0.0.0    | 255.255.255.0 | U     | 0      | 0   | 0   | qr-60828e45-4e |



# General notes on qrouter

- One qr interface per network subnet attached
- Default gateway qg-xxxx
  - One default gateway per router
- Network namespaces make it isolated
  - Multiple tenants' subnets can have overlapping ips
- Instances can be on same or different hosts. Doesn't matter !
  - Different networks mean a router and the qrouter namespace is involved !

## Scenario 3: Instance getting dhcp address



# On the network node

Same example: 2 VMs , 2 networks

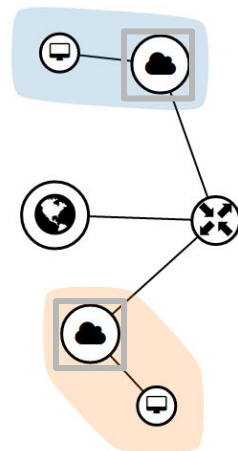
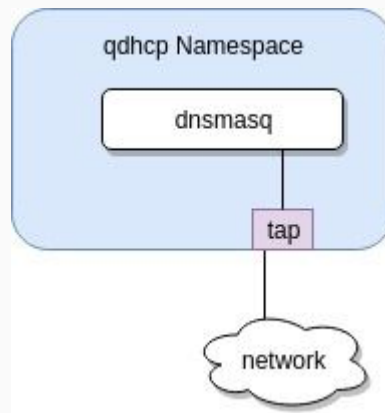
To get DHCP IP on 192.168.3.0/24 network, we need a path between compute host and tap6e56ae9a-e8 on the network node

```
[root@network-node ~]# ps -ef | grep dnsmasq
nobody 26675 1 0 Apr19 ? 00:00:00 dnsmasq --no-hosts --no-resolv --strict-order --except-interface=lo --pid-file=/var/lib/neutron/dhcp/60ce0ffb-00ad-444e-9406-d0f588be2634/pid --dhcp-hostsfile=/var/lib/neutron/dhcp/60ce0ffb-00ad-444e-9406-d0f588be2634/host --addn-hosts=/var/lib/neutron/dhcp/60ce0ffb-00ad-444e-9406-d0f588be2634/addn_hosts --dhcp-optsfile=/var/lib/neutron/dhcp/60ce0ffb-00ad-444e-9406-d0f588be2634/opts --dhcp-leasefile=/var/lib/neutron/dhcp/60ce0ffb-00ad-444e-9406-d0f588be2634/leases --dhcp-match=set:ipxe,175 --bind-interfaces --interface=tap6e56ae9a-e8 --dhcp-range=set:tag0,192.168.3.0,static,86400s --dhcp-option-force=option:mtu,1450 --dhcp-lease-max=256 --conf-file= --domain=openstacklocal
nobody 26775 1 0 Apr19 ? 00:00:00 dnsmasq --no-hosts --no-resolv --strict-order --except-interface=lo --pid-file=/var/lib/neutron/dhcp/fb4805f8-af76-4019-aec3-7b83193ba9cc/pid --dhcp-hostsfile=/var/lib/neutron/dhcp/fb4805f8-af76-4019-aec3-7b83193ba9cc/host --addn-hosts=/var/lib/neutron/dhcp/fb4805f8-af76-4019-aec3-7b83193ba9cc/addn_hosts --dhcp-optsfile=/var/lib/neutron/dhcp/fb4805f8-af76-4019-aec3-7b83193ba9cc/opts --dhcp-leasefile=/var/lib/neutron/dhcp/fb4805f8-af76-4019-aec3-7b83193ba9cc/leases --dhcp-match=set:ipxe,175 --bind-interfaces --interface=tap5c93757f-13 --dhcp-range=set:tag0,10.0.0.0,static,86400s --dhcp-option-force=option:mtu,1450 --dhcp-lease-max=256 --conf-file= --domain=openstacklocal
```

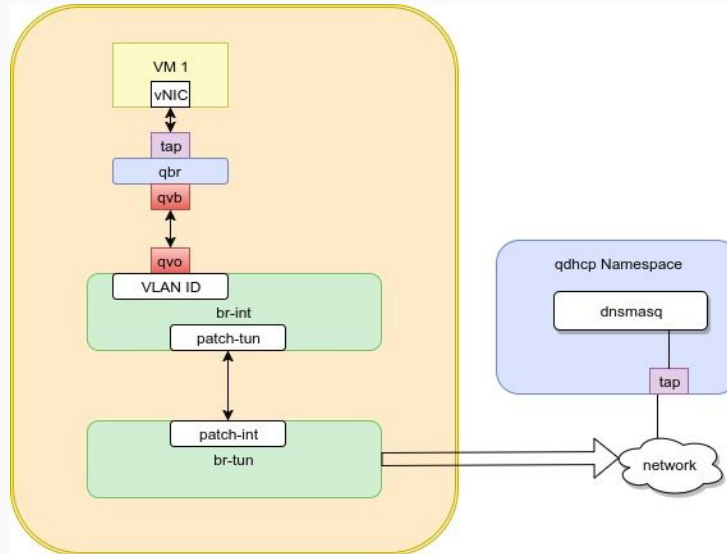
```
[root@network-node ~]# ip netns
qdhcp-60ce0ffb-00ad-444e-9406-d0f588be2634
qdhcp-fb4805f8-af76-4019-aec3-7b83193ba9cc
grouter-69d4e7f6-6351-4790-b310-e4bbc8339d2f
```

```
[root@network-node ~]# ip netns exec qdhcp-60ce0ffb-00ad-444e-9406-d0f588be2634 bash
[root@network-node ~]# ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
 inet 127.0.0.1 netmask 255.0.0.0
 inet6 ::1 prefixlen 128 scopeid 0x10<host>
 loop txqueuelen 1 (Local Loopback)
 RX packets 1 bytes 576 (576.0 B)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 1 bytes 576 (576.0 B)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

tap6e56ae9a-e8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1450
 inet 192.168.3.2 netmask 255.255.255.0 broadcast 192.168.3.255
 inet6 fe80::f816:3eff:fef6:6427 prefixlen 64 scopeid 0x20<link>
 ether fa:16:3e:f6:64:27 txqueuelen 1000 (Ethernet)
 RX packets 126 bytes 7428 (7.2 KiB)
 RX errors 0 dropped 0 overruns 0 frame 0
 TX packets 42 bytes 5488 (5.3 KiB)
 TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```



- Tunnel ID mapping between compute hosts and network node
- dnsmasq provides the ips



# Now we've covered ...

- Instance to Instance traffic
  - Same network
  - Different network
- Instance to dhcp-agent traffic

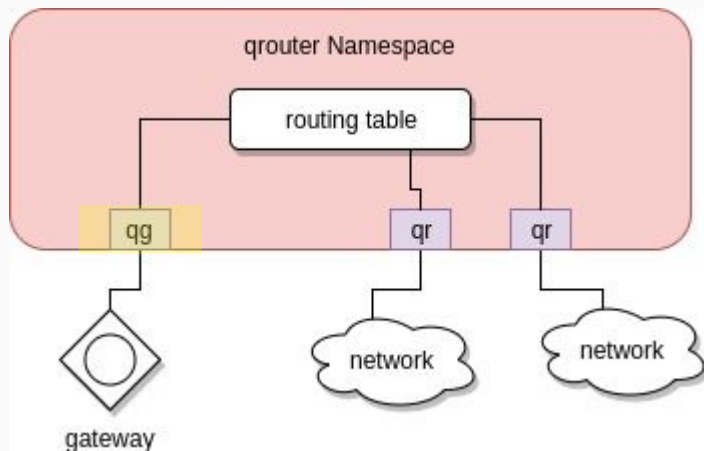
## Yet to cover

- Instance to public network
- Public network to instance (floating IPs)

## Scenario 4: Instance to public network

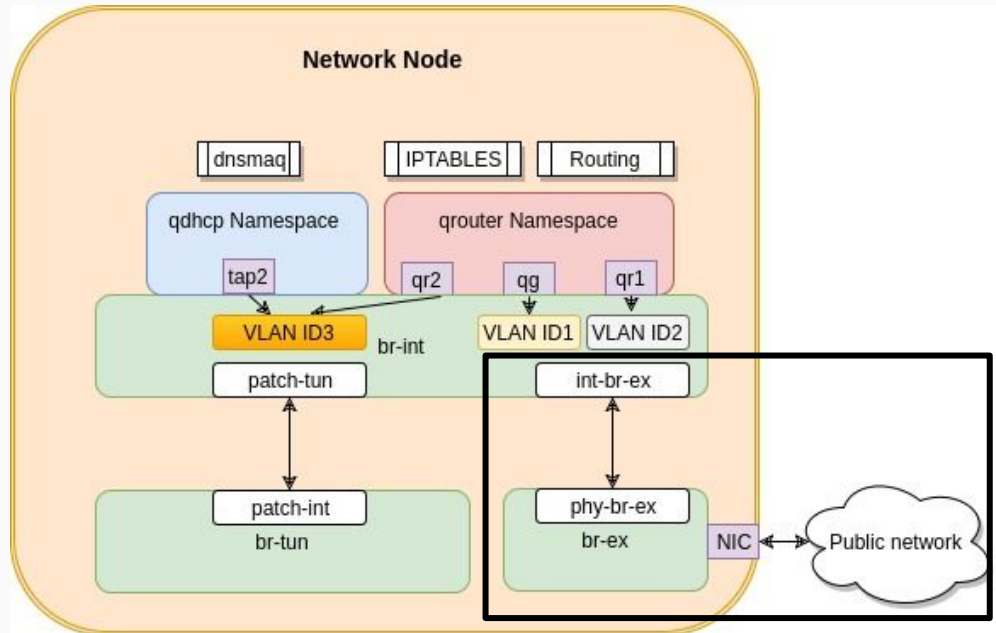
- A VM only knows about its default gateway, i.e. qr interface
- Traffic flows to the router namespace
- Routing to public network happens in the router
- qg interface !

```
[root@network-node ~]# route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
0.0.0.0 172.24.4.1 0.0.0.0 UG 0 0 0 qg-916bdf6a-a3
10.0.0.0 0.0.0.0 255.255.255.0 U 0 0 0 qr-44edbf78-34
172.24.4.0 0.0.0.0 255.255.255.0 U 0 0 0 qg-916bdf6a-a3
192.168.3.0 0.0.0.0 255.255.255.0 U 0 0 0 qr-60828e45-4e
```



# How's the qq interface connected

```
Bridge br-int
Controller "tcp:127.0.0.1:6633"
 is_connected: true
 fail_mode: secure
Port "tap6e56ae9a-e8"
 tag: 3
 Interface "tap6e56ae9a-e8"
 type: internal
Port "tap5c93757f-13"
 tag: 2
 Interface "tap5c93757f-13"
 type: internal
Port "qg-916bdf6a-a3"
 tag: 1
 Interface "qg-916bdf6a-a3"
 type: internal
Port br-int
 Interface br-int
 type: internal
Port int-br-ex
 Interface int-br-ex
 type: patch
 options: {peer=phy-br-ex}
Port patch-tun
 Interface patch-tun
 type: patch
 options: {peer=patch-int}
Port "qr-44edbf78-34"
 tag: 2
 Interface "qr-44edbf78-34"
 type: internal
Port "qr-60828e45-4e"
 tag: 3
 Interface "qr-60828e45-4e"
 type: internal
```

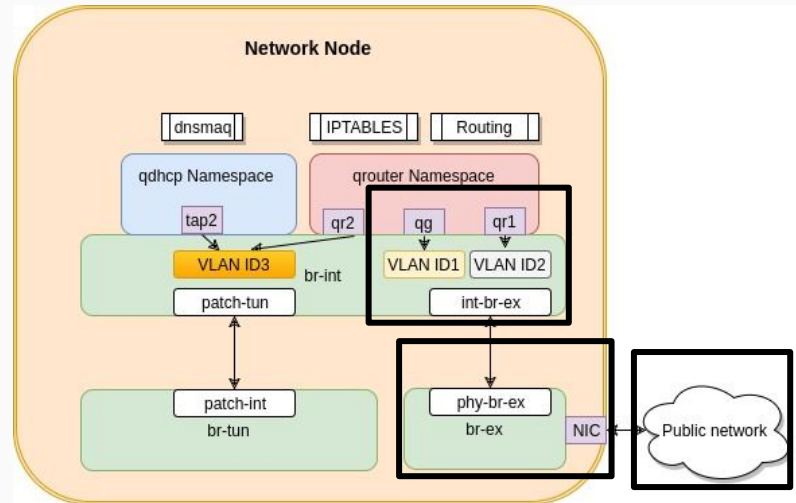
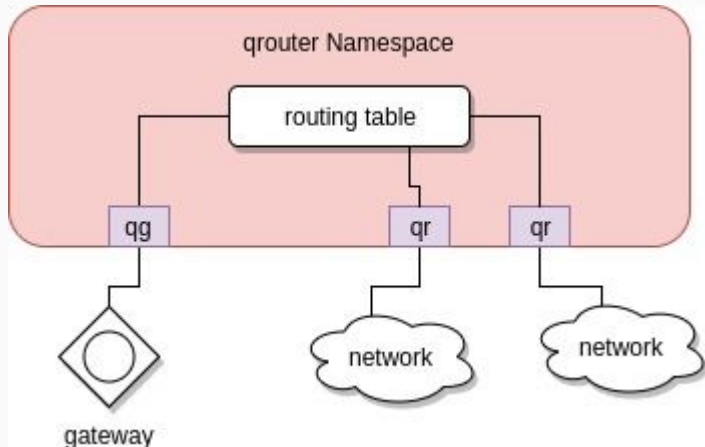


# br-ex and br-int Openflow rules

- qq sends traffic to br-ex through the patch interface
- br-ex strips the vlan id out of traffic and sends it physical NIC to the public network
- Incoming VM traffic from public network follows the same path, only VLAN id is added in this case

# Scenario 5: Floating IPs

- So far, qrouter only used routing tables
- But I mentioned iptables nat rules ?

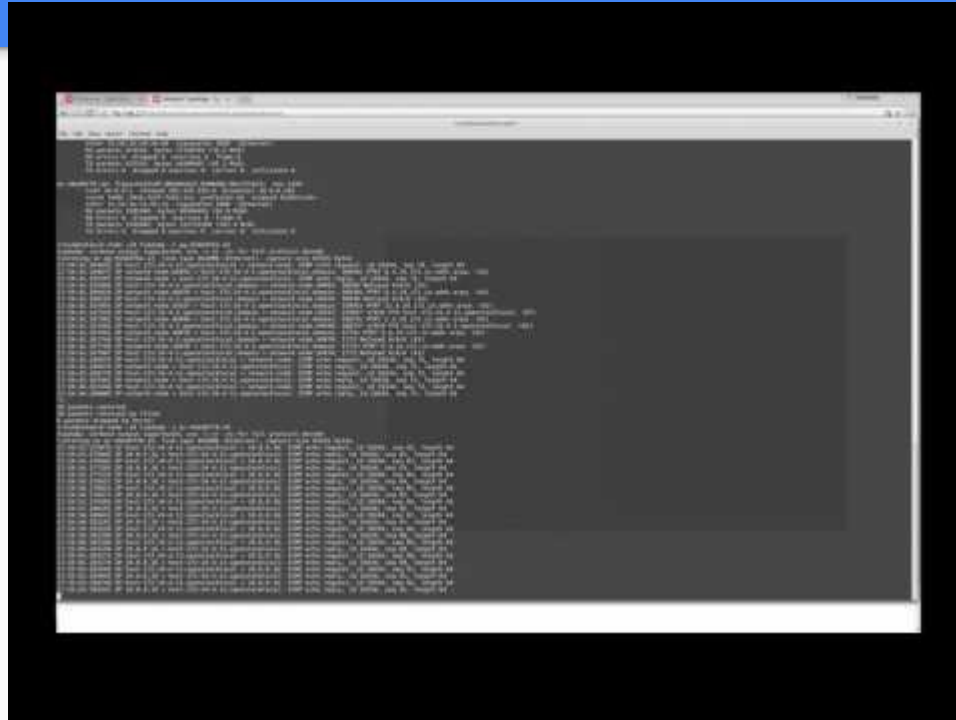


# Floating IPs: NAT

```
[root@network-node ~]# ip netns exec qrouter-69d4e7f6-6351-4790-b310-e4bbc8339d2f bash
[root@network-node ~]# iptables -t nat -S
-P PREROUTING ACCEPT
-P INPUT ACCEPT
-P OUTPUT ACCEPT
-P POSTROUTING ACCEPT
-N neutron-l3-agent-OUTPUT
-N neutron-l3-agent-POSTROUTING
-N neutron-l3-agent-PREROUTING
-N neutron-l3-agent-float-snat
-N neutron-l3-agent-snat
-N neutron-postrouting-bottom
-A PREROUTING -j neutron-l3-agent-PREROUTING
-A OUTPUT -j neutron-l3-agent-OUTPUT
-A POSTROUTING -j neutron-l3-agent-POSTROUTING
-A POSTROUTING -j neutron-postrouting-bottom
-A neutron-l3-agent-OUTPUT -d 172.24.4.8/32 -j DNAT --to-destination 10.0.0.16
-A neutron-l3-agent-POSTROUTING ! -i qq-916bdf6a-a3 ! -o qq-916bdf6a-a3 -m conntrack ! --ctstate DNAT -j ACCEPT
-A neutron-l3-agent-PREROUTING -d 169.254.169.254/32 -i qr-+ -p tcp -m tcp --dport 80 -j REDIRECT --to-ports 9697
-A neutron-l3-agent-PREROUTING -d 172.24.4.8/32 -j DNAT --to-destination 10.0.0.16
-A neutron-l3-agent-float-snat -s 10.0.0.16/32 -j SNAT --to-source 172.24.4.8
-A neutron-l3-agent-snat -j neutron-l3-agent-float-snat
-A neutron-l3-agent-snat -o qq-916bdf6a-a3 -j SNAT --to-source 172.24.4.5
-A neutron-l3-agent-snat -m mark ! --mark 0x2/0xffff -m conntrack --ctstate DNAT -j SNAT --to-source 172.24.4.5
-A neutron-postrouting-bottom -m comment --comment "Perform source NAT on outgoing traffic." -j neutron-l3-agent-snat
```

# Demo: Outbound Traffic

# Demo: Inbound traffic



# Lastly...

- Want to learn ? packstack deployment
- Neutron is very logical
- tcpdump , ethtool & dummy/mirror interfaces are your friends
- Special thanks to
  - draw.io
  - Google
  - Everyone who shares knowledge about openstack