# 深入理解 OpenStack 中的网络实现

最新版可以从 yeasy@github 下载。

v0.6: 2014-04-04

修正系统结构的图表；

修正部分描述。

v0.5: 2014-03-24

添加对 vlan 模式下具体规则的分析；

更新 gre 模式下 answerfile 的 IP 信息。

v0.4: 2014-03-20

添加对安全组实现的完整分析，添加整体逻辑图表；

添加 vlan 模式下的 RDO answer 文件（更新 IP 信息）。

v0.3: 2014-03-10

添加 GRE 模式下对流表规则分析；

添加 GRE 模式下的 answer 文件。

v0.2: 2014-03-06

修正图表引用错误；

添加对 GRE 模式下流表细节分析。

v0.1: 2014-02-20

开始整体结构。

## 1.1 概述

### 1.1.1 术语

bridge：网桥，Linux 中用于表示一个能连接不同网络设备的虚拟设备，linux 中传统实现的网桥类似一个 hub 设备，而 ovs 管理的网桥一般类似交换机。

br-int：bridge-integration，综合网桥，常用于表示实现主要内部网络功能的网桥。

br-ex：bridge-external，外部网桥，通常表示负责跟外部网络通信的网桥。

GRE：General Routing Encapsulation，一种通过封装来实现隧道的方式。在 openstack 中一般是基于 L3 的 gre，即 original pkt/GRE/IP/Ethernet

VETH：虚拟 ethernet 接口，通常以 pair 的方式出现，一端发出的网包，会被另一端接收，可以形成两个网桥之间的通道。

qvb：neutron veth, Linux Bridge-side

qvo：neutron veth, OVS-side

TAP 设备：模拟一个二层的网络设备，可以接受和发送二层网包。

TUN 设备：模拟一个三层的网络设备，可以接受和发送三层网包。

iptables：Linux 上常见的实现安全策略的防火墙软件。

Vlan：虚拟 lan，同一个物理 lan 下用标签实现隔离，可用标号为 1-4094。

namespace：用来实现隔离的一套机制，不同 namespace 之间彼此不可见。

## 1.2 概念

Neutron 管理下面的实体：

- 网络： 隔离的 L2 域，可以是虚拟、逻辑或交换，同一个网络中的主机彼此 L2 可见。
- 子网： IP 地址快，其中每个虚拟机有一个 IP，同一个子网的主机彼此 L3 可见。
- 端口： 网络上虚拟、逻辑或交换端口。

所有这些实体都是虚拟的，拥有自动生成的唯一标示 id，支持 CRUD 功能，并在数据库中跟踪记录状态。

### 1.2.1 网络

隔离的 L2 广播域，一般是创建它的用户所有。用户可以拥有多个网络。网络是最基础的，子网和端口都需要关联到网络上。网络的属性如所示。

### 1.2.2 子网

子网代表了一组分配了 IP 的虚拟机。每个子网必须有一个 CIDR 和关联到一个网络。IP 可以从 CIDR 或者用户指定池中选取。

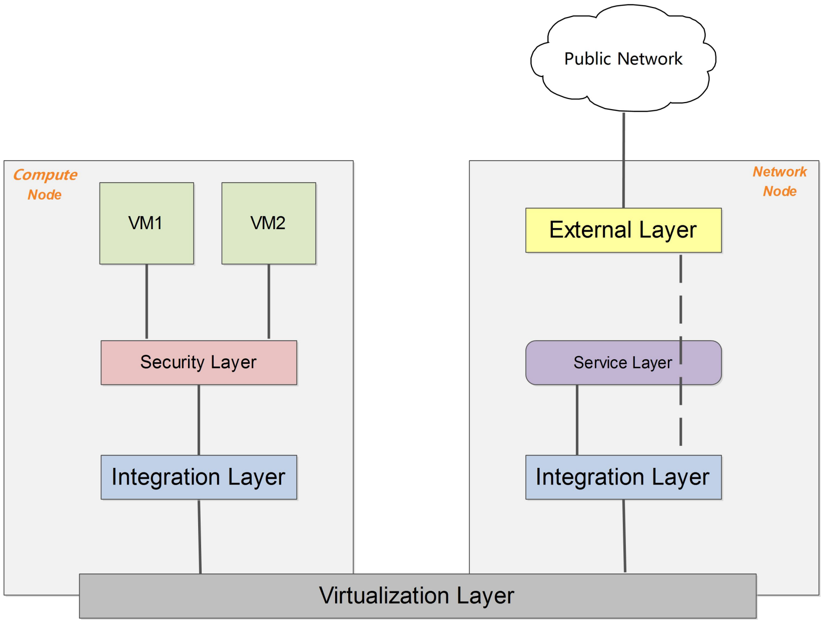
子网可能会有一个网关、一组 DNS 和主机路由。

### 1.2.3 端口

逻辑网络交换机上的一个虚拟交换口。虚拟机挂载他们的网卡到这些端口上。逻辑口往往定义了挂载到它上面的网卡的 MAC 地址和 IP 地址。当端口有 IP 的时候，意味着它属于某个子网。

### 1.2.4 抽象系统架构

无论 GRE 模式实现还是 Vlan 模式实现，抽象的系统架构可以表述为图表 1 所示。

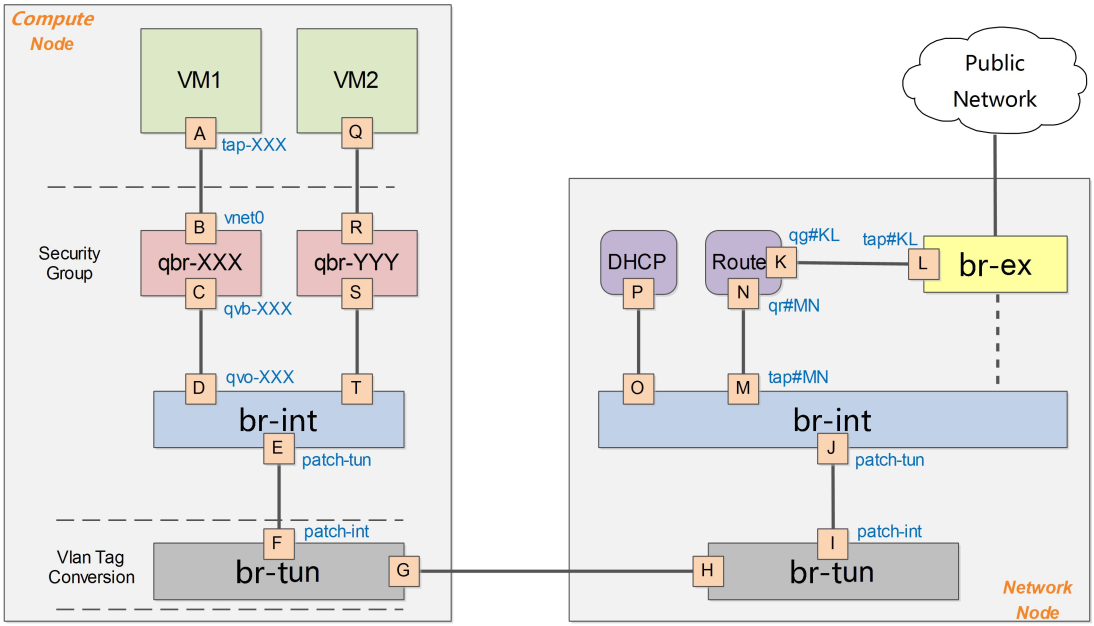


图表 1 抽象系统架构

下面将分别就 GRE 模式实现和 VLan 模式实现进行分析。

## 1.3 GRE 模式

图表 1 给出了在 OpenStack 中网络实现的一个简化的架构示意。OpenStack 中网络实现包括 vlan 和 gre 两种模式，此处以 gre 模式为例。



图表 2 网络基本架构

在 OpenStack 中，所有网络有关的逻辑管理均在 Network 节点中实现，例如 DNS、DHCP 以及路由等。Compute 节点上只需要对所部属的虚拟机提供基本的网络功能支持，包括隔离不同租户的虚拟机和进行一些基本的安全策略管理（即 security group）。

## 1.3.1 Compute 节点

以图表 1 为例，Compute 节点上包括两台虚拟机 VM1 和 VM2，分别经过一个网桥（如 qbr-XXX）连接到 br-int 网桥上。br-int 网桥再经过 br-tun 网桥（物理网络是 GRE 实现）连接到物理主机外部网络。

对于物理网络通过 vlan 来隔离的情况，一般会存在一个 br-eth 网桥。

### 1.3.1.1 qbr

在 VM1 中，虚拟机的网卡实际上连接到了物理机的一个 TAP 设备（即 A，常见名称如 tap-XXX）上，A 则进一步通过 VETH pair（A-B）连接到网桥 qbr-XXX 的端口 vnet0（端口 B）上，之后再通过 VETH pair（C-D）连到 br-int 网桥上。一般 C 的名字格式为 qvb-XXX，而 D 的名字格式为 qvo-XXX。注意它们的名称除了前缀外，后面的 id 都是一样的，表示位于同一个虚拟机网络到物理机网络的连接上。

之所以 TAP 设备 A 没有直接连接到网桥 br-int 上，是因为 OpenStack 需要通过 iptables 实现 security group 的安全策略功能。目前 openvswitch 并不支持应用 iptables 规则的 Tap 设备。

因为 qbr 的存在主要是为了辅助 iptables 来实现 security group 功能，有时候也被称为 firewall bridge。详见 security group 部分的分析。

### 1.3.1.2 br-int

一个典型的 br-int 的端口如下所示：

```
# ovs-vsctl show
Bridge br-int
    Port "qvo-XXX"
        tag: 1
        Interface "qvo-XXX"
    Port patch-tun
        Interface patch-tun
            type: patch
            options: {peer=patch-int}
    Port br-int
        Interface br-int
            type: internal
```

其中 br-int 为内部端口。

端口 patch-tun（即端口 E，端口号为 1）连接到 br-tun 上，实现到外部网络的隧道。

端口 qvo-XXX（即端口 D，端口号为 2）带有 tag1，说明这个口是一个 1 号 vlan 的 access 端口。虚拟机发出的从该端口到达 br-int 的网包将被自动带上 vlan tag 1，而其他带有 vlan tag 1 的网包则可以在去掉 vlan tag 后从该端口发出（具体请查询 vlan access 端口）。这个 vlan tag 是用来实现不同网络相互隔离的，比如租户创建一个网络（neutron net-create），则会被分配一个唯一的 vlan tag。

br-int 在 GRE 模式中作为一个 NORMAL 交换机使用，因此有效规则只有一条正常转发。如果两个在同一主机上的 vm 属于同一个 tenant 的（同一个 vlan tag），则它们之间的通信只需要经过 br-int 即可。

```
# ovs-ofctl dump-flows br-int
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=10727.864s, table=0, n_packets=198, n_bytes=17288, idle_age=13,
priority=1 actions=NORMAL
```

### 1.3.1.3 br-tun

一个典型的 br-tun 上的端口类似：

```
Bridge br-tun
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-tun}
        Port "gre-1"
            Interface "gre-1"
                type: gre
                options: {in_key=flow, local_ip="10.0.0.101", out_key=flow,
remote_ip="10.0.0.100"}
        Port br-tun
            Interface br-tun
                type: internal
```

其中 patch-int（即端口 F，端口号为 1）是连接到 br-int 上的 veth pair 的端口，gre-1 口（即端口 G，端口号为 2）对应 vm 到外面的隧道。

gre-1 端口是虚拟 gre 端口，当网包发送到这个端口的时候，会经过内核封包，然后从 10.0.0.101 发送到 10.0.0.100，即从本地的物理网卡（10.0.0.101）发出。

br-tun 将带有 vlan tag 的 vm 跟外部通信的流量转换到对应的 gre 隧道，这上面要实现主要的转换逻辑，规则要复杂，一般通过多张表来实现。

典型的转发规则为：

```
# ovs-ofctl dump-flows br-tun
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=10970.064s, table=0, n_packets=189, n_bytes=16232, idle_age=16,
priority=1,in_port=1 actions=resubmit(,1)
 cookie=0x0, duration=10906.954s, table=0, n_packets=29, n_bytes=5736, idle_age=16,
priority=1,in_port=2 actions=resubmit(,2)
 cookie=0x0, duration=10969.922s, table=0, n_packets=3, n_bytes=230, idle_age=10962,
priority=0 actions=drop
 cookie=0x0, duration=10969.777s, table=1, n_packets=26, n_bytes=5266, idle_age=16,
priority=0,dl_dst=00:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,20)
```

```
 cookie=0x0, duration=10969.631s, table=1, n_packets=163, n_bytes=10966, idle_age=21,
priority=0,dl_dst=01:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,21)
 cookie=0x0, duration=688.456s, table=2, n_packets=29, n_bytes=5736, idle_age=16,
priority=1,tun_id=0x1 actions=mod_vlan_vid:1,resubmit(,10)
 cookie=0x0, duration=10969.488s, table=2, n_packets=0, n_bytes=0, idle_age=10969,
priority=0 actions=drop
 cookie=0x0, duration=10969.343s, table=3, n_packets=0, n_bytes=0, idle_age=10969,
priority=0 actions=drop
 cookie=0x0, duration=10969.2s, table=10, n_packets=29, n_bytes=5736, idle_age=16,
priority=1
actions=learn(table=20,hard_timeout=300,priority=1,NXM_OF_VLAN_TCI[0..11],NXM_OF_ETH
_DST[]=NXM_OF_ETH_SRC[],load:0->NXM_OF_VLAN_TCI[],load:NXM_NX_TUN_ID[]->NXM_NX
_TUN_ID[],output:NXM_OF_IN_PORT[]),output:1
 cookie=0x0, duration=682.603s, table=20, n_packets=26, n_bytes=5266, hard_timeout=300,
idle_age=16, hard_age=16, priority=1,vlan_tci=0x0001/0x0fff,dl_dst=fa:16:3e:32:0d:db
actions=load:0->NXM_OF_VLAN_TCI[],load:0x1->NXM_NX_TUN_ID[],output:2
 cookie=0x0, duration=10969.057s, table=20, n_packets=0, n_bytes=0, idle_age=10969,
priority=0 actions=resubmit(,21)
 cookie=0x0, duration=688.6s, table=21, n_packets=161, n_bytes=10818, idle_age=21,
priority=1,dl_vlan=1 actions=strip_vlan,set_tunnel:0x1,output:2
 cookie=0x0, duration=10968.912s, table=21, n_packets=2, n_bytes=148, idle_age=689,
priority=0 actions=drop
```

其中，表 0 中有 3 条规则：从端口 1（即 patch-int）来的，扔到表 1，从端口 2（即 gre-1）来的，扔到表 2。

```
cookie=0x0, duration=10970.064s, table=0, n_packets=189, n_bytes=16232, idle_age=16,
priority=1,in_port=1 actions=resubmit(,1)
 cookie=0x0, duration=10906.954s, table=0, n_packets=29, n_bytes=5736, idle_age=16,
priority=1,in_port=2 actions=resubmit(,2)
 cookie=0x0, duration=10969.922s, table=0, n_packets=3, n_bytes=230, idle_age=10962,
priority=0 actions=drop
```

表 1 有 2 条规则：如果是单播（00:00:00:00:00:00/01:00:00:00:00:00），则扔到表 20；如果是多播等（01:00:00:00:00:00/01:00:00:00:00:00），则扔到表 21。

```
cookie=0x0, duration=10969.777s, table=1, n_packets=26, n_bytes=5266, idle_age=16,
priority=0,dl_dst=00:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,20)
 cookie=0x0, duration=10969.631s, table=1, n_packets=163, n_bytes=10966, idle_age=21,
priority=0,dl_dst=01:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,21)
```

表 2 有 2 条规则：如果是 tunnel 1 的网包，则修改其 vlan id 为 1，并扔到表 10；非 tunnel 1 的网包，则丢弃。

```
cookie=0x0, duration=688.456s, table=2, n_packets=29, n_bytes=5736, idle_age=16,
priority=1,tun_id=0x1 actions=mod_vlan_vid:1,resubmit(,10)
```

```
 cookie=0x0, duration=10969.488s, table=2, n_packets=0, n_bytes=0, idle_age=10969,
priority=0 actions=drop
```

表 3 只有 1 条规则：丢弃。

表 10 有一条规则，基于 learn 行动来创建反向（从 gre 端口抵达，且目标是到 vm 的网包）的规则。learn 行动并非标准的 openflow 行动，是 openvswitch 自身的扩展行动，这个行动可以根据流内容动态来修改流表内容。这条规则首先创建了一条新的流（该流对应 vm 从 br-tun 的 gre 端口发出的规则）：其中 table=20 表示规则添加在表 20；NXM_OF_VLAN_TCI[0..11]表示匹配包自带的 vlan id；NXM_OF_ETH_DST[]=NXM_OF_ETH_SRC[]表示 L2 目标地址需要匹配包的 L2 源地址；load:0->NXM_OF_VLAN_TCI[]，去掉 vlan，load:NXM_NX_TUN_ID[]->NXM_NX_TUN_ID[]，添加 tunnel 号为原始 tunnel 号；output:NXM_OF_IN_PORT[]，发出端口为原始包抵达的端口。最后规则将匹配的网包从端口 1（即 patch-int）发出。

```
cookie=0x0, duration=10969.2s, table=10, n_packets=29, n_bytes=5736, idle_age=16,
priority=1
actions=learn(table=20,hard_timeout=300,priority=1,NXM_OF_VLAN_TCI[0..11],NXM_OF_ETH
_DST[]=NXM_OF_ETH_SRC[],load:0->NXM_OF_VLAN_TCI[],load:NXM_NX_TUN_ID[]->NXM_NX
_TUN_ID[],output:NXM_OF_IN_PORT[]),output:1
```

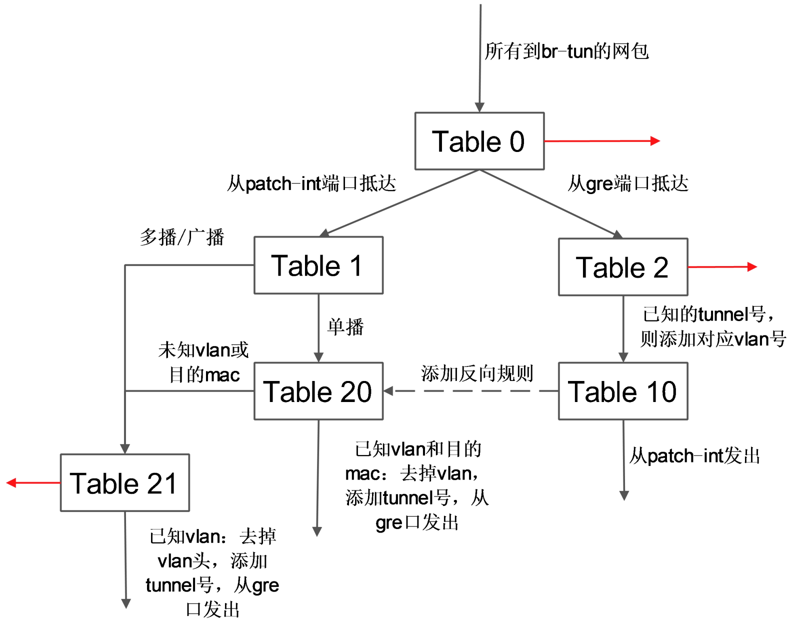
表 20 中有两条规则，其中第一条即表 10 中规则利用 learn 行动创建的流表项，第 2 条提交其他流到表 21。

```
cookie=0x0, duration=682.603s, table=20, n_packets=26, n_bytes=5266, hard_timeout=300,
idle_age=16, hard_age=16, priority=1,vlan_tci=0x0001/0x0fff,dl_dst=fa:16:3e:32:0d:db
actions=load:0->NXM_OF_VLAN_TCI[],load:0x1->NXM_NX_TUN_ID[],output:2
 cookie=0x0, duration=10969.057s, table=20, n_packets=0, n_bytes=0, idle_age=10969,
priority=0 actions=resubmit(,21)
```

表 21 有 2 条规则，第一条是匹配所有目标 vlan 为 1 的网包，去掉 vlan，然后从端口 2（gre 端口）发出。第二条是丢弃。

```
 cookie=0x0, duration=688.6s, table=21, n_packets=161, n_bytes=10818, idle_age=21,
priority=1,dl_vlan=1 actions=strip_vlan,set_tunnel:0x1,output:2
 cookie=0x0, duration=10968.912s, table=21, n_packets=2, n_bytes=148, idle_age=689,
priority=0 actions=drop
```

这些规则所组成的整体转发逻辑如图表 2 所示。



图表 3 Compute 节点 br-tun 的转发逻辑

## 1.3.2 Network 节点

### 1.3.2.1 br-tun

```
Bridge br-tun
        Port br-tun
            Interface br-tun
                type: internal
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-tun}
        Port "gre-2"
            Interface "gre-2"
                type: gre
                options:     {in_key=flow,     local_ip="10.0.0.100",     out_key=flow,
remote_ip="10.0.0.101"}
```

Compute 节点上发往 GRE 隧道的网包最终抵达 Network 节点上的 br-tun，该网桥的规则包括：

```
# ovs-ofctl dump-flows br-tun
NXST_FLOW reply (xid=0x4):
```

```
cookie=0x0, duration=19596.862s, table=0, n_packets=344, n_bytes=66762, idle_age=4,
priority=1,in_port=1 actions=resubmit(,1)
 cookie=0x0, duration=19537.588s, table=0, n_packets=625, n_bytes=125972, idle_age=4,
priority=1,in_port=2 actions=resubmit(,2)
 cookie=0x0, duration=19596.602s, table=0, n_packets=2, n_bytes=140, idle_age=19590,
priority=0 actions=drop
 cookie=0x0, duration=19596.343s, table=1, n_packets=323, n_bytes=65252, idle_age=4,
priority=0,dl_dst=00:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,20)
 cookie=0x0, duration=19596.082s, table=1, n_packets=21, n_bytes=1510, idle_age=5027,
priority=0,dl_dst=01:00:00:00:00:00/01:00:00:00:00:00 actions=resubmit(,21)
 cookie=0x0, duration=9356.289s, table=2, n_packets=625, n_bytes=125972, idle_age=4,
priority=1,tun_id=0x1 actions=mod_vlan_vid:1,resubmit(,10)
 cookie=0x0, duration=19595.821s, table=2, n_packets=0, n_bytes=0, idle_age=19595,
priority=0 actions=drop
 cookie=0x0, duration=19595.554s, table=3, n_packets=0, n_bytes=0, idle_age=19595,
priority=0 actions=drop
 cookie=0x0, duration=19595.292s, table=10, n_packets=625, n_bytes=125972, idle_age=4,
priority=1
actions=learn(table=20,hard_timeout=300,priority=1,NXM_OF_VLAN_TCI[0..11],NXM_OF_ETH
_DST[]=NXM_OF_ETH_SRC[],load:0->NXM_OF_VLAN_TCI[],load:NXM_NX_TUN_ID[]->NXM_NX
_TUN_ID[],output:NXM_OF_IN_PORT[]),output:1
 cookie=0x0, duration=9314.338s, table=20, n_packets=323, n_bytes=65252,
hard_timeout=300, idle_age=4, hard_age=3,
priority=1,vlan_tci=0x0001/0x0fff,dl_dst=fa:16:3e:cb:11:f6
actions=load:0->NXM_OF_VLAN_TCI[],load:0x1->NXM_NX_TUN_ID[],output:2
 cookie=0x0, duration=19595.026s, table=20, n_packets=0, n_bytes=0, idle_age=19595,
priority=0 actions=resubmit(,21)
 cookie=0x0, duration=9356.592s, table=21, n_packets=9, n_bytes=586, idle_age=5027,
priority=1,dl_vlan=1 actions=strip_vlan,set_tunnel:0x1,output:2
 cookie=0x0, duration=19594.759s, table=21, n_packets=12, n_bytes=924, idle_age=5057,
priority=0 actions=drop
```

这些规则跟 Compute 节点上 br-tun 的规则相似，完成 tunnel 跟 vlan 之间的转换。

#### 1.3.2.2 br-int

```
Bridge br-int
        Port "qr-ff19a58b-3d"
            tag: 1
            Interface "qr-ff19a58b-3d"
                type: internal
        Port br-int
```

```
            Interface br-int
                type: internal
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
        Port "tap4385f950-8b"
            tag: 1
            Interface "tap4385f950-8b"
                type: internal
```

该集成网桥上挂载了很多进程来提供网络服务，包括路由器、DHCP 服务器等。这些进程不同的租户可能都需要，彼此的地址空间可能冲突，也可能跟物理网络的地址空间冲突，因此都运行在独立的网络名字空间中。

规则跟 computer 节点的 br-int 规则一致，表现为一个正常交换机。

```
# ovs-ofctl dump-flows br-int
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=18198.244s, table=0, n_packets=849, n_bytes=164654, idle_age=43, priority=1 actions=NORMAL
```

### 1.3.2.3 网络名字空间

在 linux 中，网络名字空间可以被认为是隔离的拥有单独网络栈（网卡、路由转发表、iptables）的环境。网络名字空间经常用来隔离网络设备和服务，只有拥有同样网络名字空间的设备，才能看到彼此。

可以用 ip netns list 命令来查看已经存在的名字空间。

```
# ip netns
qdhcp-88b1609c-68e0-49ca-a658-f1edff54a264
qrouter-2d214fde-293c-4d64-8062-797f80ae2d8f
```

qdhcp 开头的名字空间是 dhcp 服务器使用的，qrouter 开头的则是 router 服务使用的。

可以通过 ip netns exec namespaceid command 来在指定的网络名字空间中执行网络命令，例如

```
# ip netns exec qdhcp-88b1609c-68e0-49ca-a658-f1edff54a264 ip addr
71: ns-f14c598d-98: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether fa:16:3e:10:2f:03 brd ff:ff:ff:ff:ff:ff
    inet 10.1.0.3/24 brd 10.1.0.255 scope global ns-f14c598d-98
    inet6 fe80::f816:3eff:fe10:2f03/64 scope link
       valid_lft forever preferred_lft forever
```

可以看到，dhcp 服务的网络名字空间中只有一个网络接口“ns-f14c598d-98”，它连接到 br-int 的 tapf14c598d-98 接口上。

### 1.3.2.4 dhcp 服务

dhcp 服务是通过 dnsmasq 进程（轻量级服务器，可以提供 dns、dhcp、tftp 等服务）来实现的，该进程绑定到 dhcp 名字空间中的 br-int 的接口上。可以查看相关的进程。

```
# ps -fe | grep 88b1609c-68e0-49ca-a658-f1edff54a264
nobody       23195          1     0  Oct26  ?                00:00:00  dnsmasq  --no-hosts  --no-resolv
--strict-order        --bind-interfaces        --interface=ns-f14c598d-98        --except-interface=lo
--pid-file=/var/lib/neutron/dhcp/88b1609c-68e0-49ca-a658-f1edff54a264/pid
--dhcp-hostsfile=/var/lib/neutron/dhcp/88b1609c-68e0-49ca-a658-f1edff54a264/host
--dhcp-optsfile=/var/lib/neutron/dhcp/88b1609c-68e0-49ca-a658-f1edff54a264/opts
--dhcp-script=/usr/bin/neutron-dhcp-agent-dnsmasq-lease-update                        --leasefile-ro
--dhcp-range=tag0,10.1.0.0,static,120s --conf-file= --domain=openstacklocal
root            23196  23195     0  Oct26  ?                00:00:00  dnsmasq  --no-hosts  --no-resolv
--strict-order        --bind-interfaces        --interface=ns-f14c598d-98        --except-interface=lo
--pid-file=/var/lib/neutron/dhcp/88b1609c-68e0-49ca-a658-f1edff54a264/pid
--dhcp-hostsfile=/var/lib/neutron/dhcp/88b1609c-68e0-49ca-a658-f1edff54a264/host
--dhcp-optsfile=/var/lib/neutron/dhcp/88b1609c-68e0-49ca-a658-f1edff54a264/opts
--dhcp-script=/usr/bin/neutron-dhcp-agent-dnsmasq-lease-update                        --leasefile-ro
--dhcp-range=tag0,10.1.0.0,static,120s --conf-file= --domain=openstacklocal
```

### 1.3.2.5 router 服务

首先，什么是 router，router 是提供跨 subnet 的互联功能的。比如用户的内部网络中主机想要访问外部互联网的地址，就需要 router 来转发（因此，所有跟外部网络的流量都必须经过 router）。目前 router 的实现是通过 iptables 进行的。

同样的，router 服务也运行在自己的名字空间中，可以通过如下命令查看：

```
# ip netns exec qrouter-2d214fde-293c-4d64-8062-797f80ae2d8f ip addr
66:  qg-d48b49e0-aa:  <BROADCAST,MULTICAST,UP,LOWER_UP>  mtu  1500  qdisc  pfifo_fast
state UP qlen 1000
    link/ether fa:16:3e:5c:a2:ac brd ff:ff:ff:ff:ff:ff
    inet 172.24.4.227/28 brd 172.24.4.239 scope global qg-d48b49e0-aa
    inet 172.24.4.228/32 brd 172.24.4.228 scope global qg-d48b49e0-aa
    inet6 fe80::f816:3eff:fe5c:a2ac/64 scope link
       valid_lft forever preferred_lft forever
68: qr-c2d7dd02-56: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
    link/ether fa:16:3e:ea:64:6e brd ff:ff:ff:ff:ff:ff
    inet 10.1.0.1/24 brd 10.1.0.255 scope global qr-c2d7dd02-56
    inet6 fe80::f816:3eff:feea:646e/64 scope link
       valid_lft forever preferred_lft forever
```

可以看出，该名字空间中包括两个网络接口。

第一个接口 qg-d48b49e0-aa（即 K）是外部接口（qg=q gateway），将路由器的网关指向默认网关（通过 router-gateway-set 命令指定），这个接口连接到 br-ex 上的 tapd48b49e0-aa（即 L）。

第二个接口 qr-c2d7dd02-56（即 N，qr=q bridge）跟 br-int 上的 tapc2d7dd02-56 口（即 M）相连，将 router 进程连接到集成网桥上。

查看该名字空间中的路由表：

```
# ip netns exec qrouter-2d214fde-293c-4d64-8062-797f80ae2d8f ip route
172.24.4.224/28 dev qg-d48b49e0-aa    proto kernel    scope link    src 172.24.4.227
10.1.0.0/24 dev qr-c2d7dd02-56    proto kernel    scope link    src 10.1.0.1
default via 172.24.4.225 dev qg-d48b49e0-aa
```

其中，第一条规则是将到 172.24.4.224/28 段的访问都从网卡 qg-d48b49e0-aa（即 K）发出。

第二条规则是将到 10.1.0.0/24 段的访问都从网卡 qr-c2d7dd02-56（即 N）发出。

最后一条是默认路由，所有的通过 qg-d48b49e0-aa 网卡（即 K）发出。

floating ip 服务同样在路由器名字空间中实现，例如如果绑定了外部的 floating ip 172.24.4.228 到某个虚拟机 10.1.0.2，则 nat 表中规则为：

```
# ip netns exec qrouter-2d214fde-293c-4d64-8062-797f80ae2d8f iptables -t nat -S
-P PREROUTING ACCEPT
-P POSTROUTING ACCEPT
-P OUTPUT 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 POSTROUTING -j neutron-l3-agent-POSTROUTING
-A POSTROUTING -j neutron-postrouting-bottom
-A OUTPUT -j neutron-l3-agent-OUTPUT
-A neutron-l3-agent-OUTPUT -d 172.24.4.228/32 -j DNAT --to-destination 10.1.0.2
-A neutron-l3-agent-POSTROUTING ！-i qg-d48b49e0-aa ！-o qg-d48b49e0-aa -m conntrack !
--ctstate DNAT -j ACCEPT
-A neutron-l3-agent-PREROUTING -d 169.254.169.254/32 -p tcp -m tcp --dport 80 -j REDIRECT
--to-ports 9697
-A neutron-l3-agent-PREROUTING -d 172.24.4.228/32 -j DNAT --to-destination 10.1.0.2
-A neutron-l3-agent-float-snat -s 10.1.0.2/32 -j SNAT --to-source 172.24.4.228
-A neutron-l3-agent-snat -j neutron-l3-agent-float-snat
-A neutron-l3-agent-snat -s 10.1.0.0/24 -j SNAT --to-source 172.24.4.227
-A neutron-postrouting-bottom -j neutron-l3-agent-snat
```

其中 SNAT 和 DNAT 规则完成外部 floating ip 到内部 ip 的映射：

```
-A neutron-l3-agent-OUTPUT -d 172.24.4.228/32 -j DNAT --to-destination 10.1.0.2
-A neutron-l3-agent-PREROUTING -d 172.24.4.228/32 -j DNAT --to-destination 10.1.0.2
```

```
-A neutron-l3-agent-float-snat -s 10.1.0.2/32 -j SNAT --to-source 172.24.4.228
```

另外有一条 SNAT 规则把所有其他的内部 IP 出来的流量都映射到外部 IP 172.24.4.227。这样即使在内部虚拟机没有外部 IP 的情况下，也可以发起对外网的访问。

```
-A neutron-l3-agent-snat -s 10.1.0.0/24 -j SNAT --to-source 172.24.4.227
```

### 1.3.2.6 br-ex

```
Bridge br-ex
        Port "eth1"
            Interface "eth1"
        Port br-ex
            Interface br-ex
                type: internal
        Port "qg-1c3627de-1b"
            Interface "qg-1c3627de-1b"
                type: internal
```

br-ex 上直接连接到外部物理网络，一般情况下网关在物理网络中已经存在，则直接转发即可。

```
# ovs-ofctl dump-flows br-ex
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=23431.091s, table=0, n_packets=893539, n_bytes=504805376, idle_age=0, priority=0 actions=NORMAL
```

如果对外部网络的网关地址配置到了 br-ex（即 br-ex 作为一个网关）：

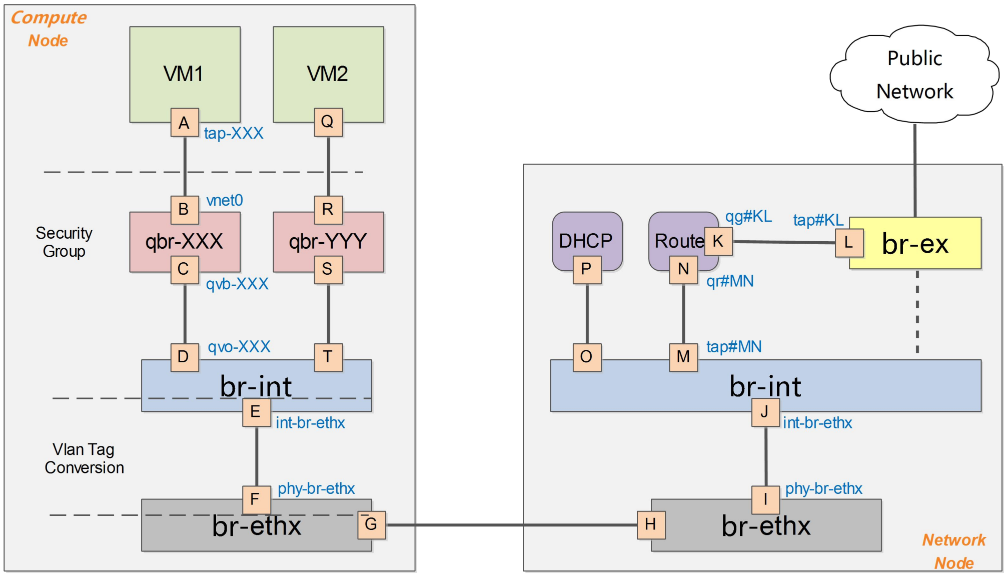
```
# ip addr add 172.24.4.225/28 dev br-ex
```

需要将内部虚拟机发出的流量进行 SNAT，之后发出。

```
# iptables -A FORWARD -d 172.24.4.224/28 -j ACCEPT
# iptables -A FORWARD -s 172.24.4.224/28 -j ACCEPT
# iptables -t nat -I POSTROUTING 1 -s 172.24.4.224/28 -j MASQUERADE
```

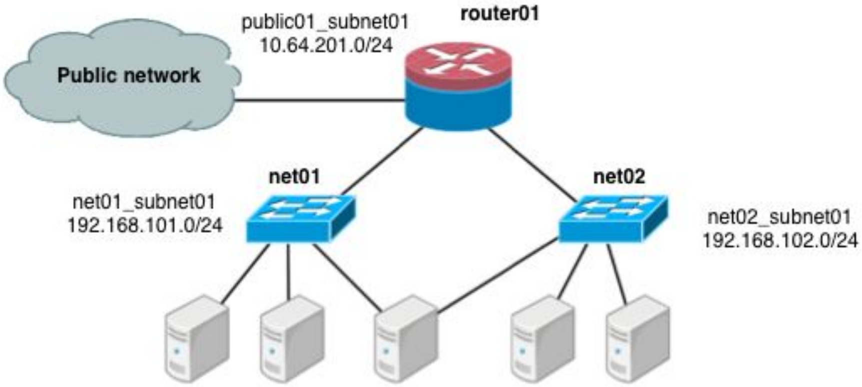
## 1.4 VLAN 模式

Vlan 模式下的系统架构跟 GRE 模式下类似，如图表 3 所示。需要注意的是，在 vlan 模式下，vlan tag 的转换需要在 br-int 和 br-ethx 两个网桥上进行相互配合。即 br-int 负责从 int-br-ethX 过来的包（带外部 vlan）转换为内部 vlan，而 br-ethx 负责从 phy-br-ethx 过来的包（带内部 vlan）转化为外部的 vlan。



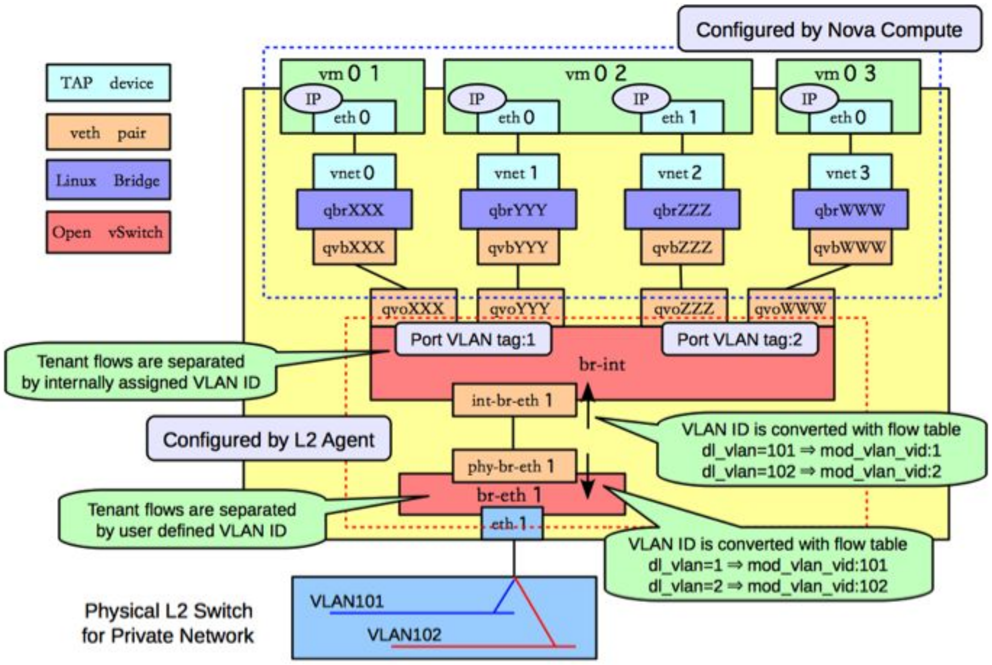
图表 4 Vlan 模式下的系统架构

下面进行一些细节的补充讨论，以 Vlan 作为物理网络隔离的实现。假如要实现同一个租户下两个子网，如图表 2 所示：



图表 5 同一个租户的两个子网

### 1.4.1 Compute 节点


Configured by Nova Compute
TAP device
veth pair
Linux Bridge
Open vSwitch
vm 0 1
vm 0 2
vm 0 3
IP
eth 0
IP
eth 0
IP
eth 1
IP
eth 0
vnet 0
vnet 1
vnet 2
vnet 3
qbrXXX
qbrYYY
qbrZZZ
qbrWWW
qvbXXX
qvbYYY
qvbZZZ
qvbWWW
qvoXXX
qvoYYY
qvoZZZ
qvoWWW
Port VLAN tag:1
Port VLAN tag:2
br-int
Tenant flows are separated by internally assigned VLAN ID
int-br-eth 1
Configured by L2 Agent
VLAN ID is converted with flow table
dl_vlan=101 ⇒ mod_vlan_vid:1
dl_vlan=102 ⇒ mod_vlan_vid:2
phy-br-eth 1
br-eth 1
eth 1
Tenant flows are separated by user defined VLAN ID
VLAN ID is converted with flow table
dl_vlan=1 ⇒ mod_vlan_vid:101
dl_vlan=2 ⇒ mod_vlan_vid:102
Physical L2 Switch for Private Network
VLAN101
VLAN102

图表 6 Compute 节点网络示意

查看网桥信息，主要包括两个网桥：br-int 和 br-eth1：

```
[root@Compute ~]# ovs-vsctl show
f758a8b8-2fd0-4a47-ab2d-c49d48304f82
    Bridge "br-eth1"
        Port "phy-br-eth1"
            Interface "phy-br-eth1"
        Port "br-eth1"
            Interface "br-eth1"
                type: internal
        Port "eth1"
            Interface "eth1"
    Bridge br-int
        Port "qvoXXX"
            tag: 1
            Interface "qvoXXX"
       Port "qvoYYY"
            tag: 1
            Interface "qvoYYY"
       Port "qvoZZZ"
             tag: 2
             Interface "qvoZZZ"
        Port "qvoWWW"
            tag: 2
            Interface "qvoWWW"
        Port "int-br-eth1"
            Interface "int-br-eth1"
        Port br-int
            Interface br-int
                type: internal
```

类似 GRE 模式下，br-int 负责租户隔离，br-eth1 负责跟计算节点外的网络通信。

在 Vlan 模式下，租户的流量隔离是通过 vlan 来进行的，因此此时包括两种 vlan，虚拟机在 Compute Node 内流量带有的 local vlan 和在 Compute Node 之外物理网络上隔离不同租户的 vlan。

br-int 和 br-eth1 分别对从端口 int-br-eth1 和 phy-br-eth1 上到达的网包进行 vlan tag 的处理。此处有两个网，分别带有两个 vlan tag（内部 tag1 对应外部 tag101，内部 tag2 对应外部 tag102）。

其中，安全组策略仍然在 qbr 相关的 iptables 上实现。

#### 1.4.1.1 br-int

与 GRE 模式不同的是，br-int 完成从 br-eth1 上过来流量（从口 int-br-eth1 到达）的 vlan tag 转换，可能的规则为

```
#ovs-ofctl dump-flows br-int
 cookie=0x0, duration=100.795s, table=0, n_packets=6, n_bytes=468, idle_age=90, 
priority=2,in_port=3 actions=drop
 cookie=0x0, duration=97.069s, table=0, n_packets=22, n_bytes=6622, idle_age=31, 
priority=3,in_port=3,dl_vlan=101 actions=mod_vlan_vid:1,NORMAL
 cookie=0x0, duration=95.781s, table=0, n_packets=8, n_bytes=1165, idle_age=11, 
priority=3,in_port=3,dl_vlan=102 actions=mod_vlan_vid:2,NORMAL
 cookie=0x0, duration=103.626s, table=0, n_packets=47, n_bytes=13400, idle_age=11, 
priority=1 actions=NORMAL
```

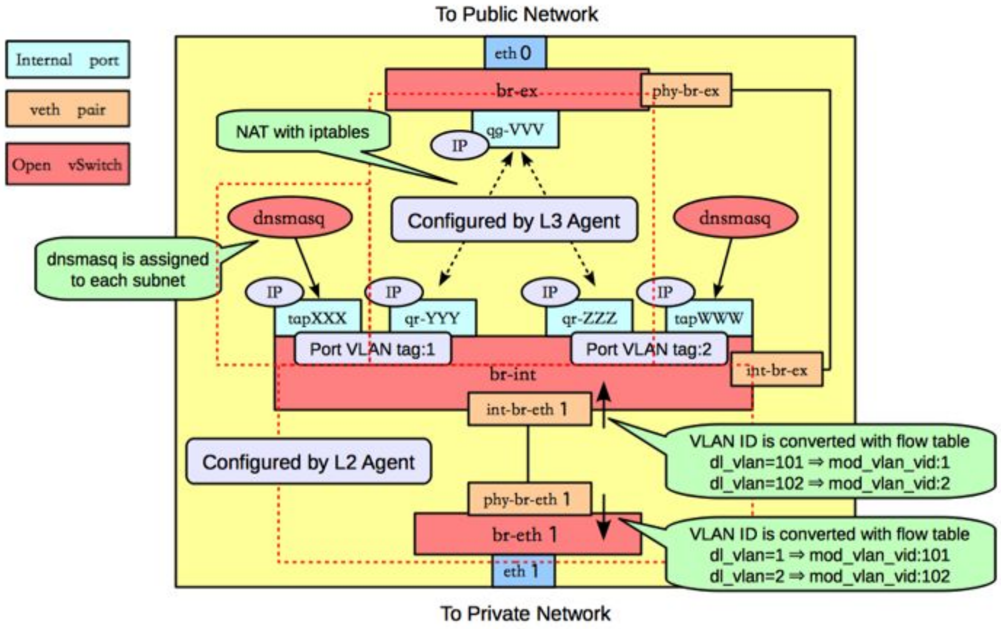
#### 1.4.1.2 br-eth1

br-eth1 上负责从 br-int 上过来的流量（从口 phy-br-eth1 到达），实现 local vlan 到外部 vlan 的转换。

```
#ovs-ofctl dump-flows br-eth0
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=73.461s, table=0, n_packets=51, n_bytes=32403, idle_age=2, 
hard_age=65534, priority=4,in_port=4,dl_vlan=1 actions=mod_vlan_vid:101,NORMAL
 cookie=0x0, duration=83.461s, table=0, n_packets=51, n_bytes=32403, idle_age=2, 
hard_age=65534, priority=4,in_port=4,dl_vlan=2 actions=mod_vlan_vid:102,NORMAL
```

```
 cookie=0x0,  duration=651.538s,  table=0,  n_packets=72,  n_bytes=3908,  idle_age=2574, hard_age=65534, priority=2,in_port=4 actions=drop
 cookie=0x0, duration=654.002s, table=0, n_packets=31733, n_bytes=6505880, idle_age=2, hard_age=65534, priority=1 actions=NORMAL
```

### 1.4.2 Network 节点



图表 7 Network 节点网络示意

类似 GRE 模式下，br-eth1 收到到达的网包，int-br-eth1 和 phy-br-eth1 上分别进行 vlan 转换，保证到达 br-int 上的网包都是带有内部 vlan tag，到达 br-eth1 上的都是带有外部 vlan tag。br-ex 则完成到 OpenStack 以外网络的连接。

查看网桥信息，包括三个网桥，br-eth1、br-int 和 br-ex。

```
#ovs
3bd78da8-d3b5-4112-a766-79506a7e2801
    Bridge br-ex
        Port "qg-VVV"
            Interface "qg-VVV"
                type: internal
        Port br-ex
            Interface br-ex
                type: internal
        Port "eth0"
            Interface "eth0"
    Bridge br-int
        Port br-int
            Interface br-int
                type: internal
        Port "int-br-eth1"
            Interface "int-br-eth0"
        Port "tapXXX"
            tag: 1
            Interface "tapXXX"
                type: internal
       Port "tapWWW"
            tag: 2
            Interface "tapWWW"
                type: internal
        Port "qr-YYY"
            tag: 1
            Interface "qr-YYY"
                type: internal
        Port "qr-ZZZ"
            tag: 2
            Interface "qr-ZZZ"
                type: internal
    Bridge "br-eth1"
        Port "phy-br-eth1"
            Interface "phy-br-eth1"
        Port "br-eth1"
            Interface "br-eth1"
                type: internal
        Port "eth1"
            Interface "eth1"
```

### 1.4.2.1 br-eth1

br-eth1 主要负责把物理网络上外部 vlan 转化为 local vlan。

```
#ovs-ofctl dump-flows br-eth1
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=144.33s, table=0, n_packets=13, n_bytes=28404, idle_age=24,
hard_age=65534, priority=4,in_port=5,dl_vlan=101 actions=mod_vlan_vid:1,NORMAL
 cookie=0x0, duration=144.33s, table=0, n_packets=13, n_bytes=28404, idle_age=24,
hard_age=65534, priority=4,in_port=5,dl_vlan=102 actions=mod_vlan_vid:2,NORMAL
```

```
 cookie=0x0, duration=608.373s, table=0, n_packets=23, n_bytes=1706, idle_age=65534,
hard_age=65534, priority=2,in_port=5 actions=drop
 cookie=0x0, duration=675.373s, table=0, n_packets=58, n_bytes=10625, idle_age=24,
hard_age=65534, priority=1 actions=NORMAL
```

### 1.4.2.2 br-int

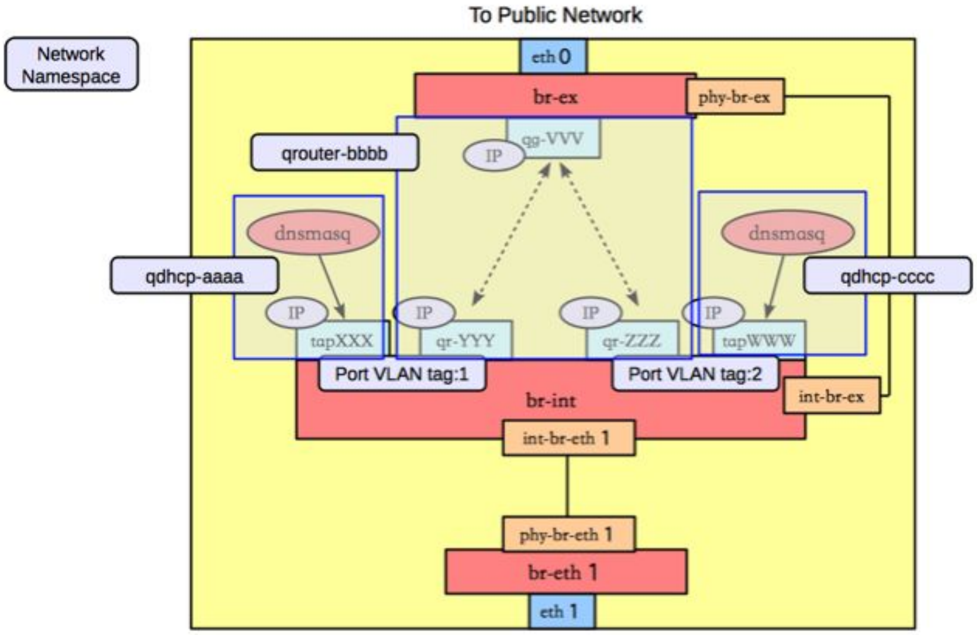
br-int 上挂载了大量的 agent 来提供各种网络服务，另外负责对发往 br-eth1 的流量，实现 local vlan 转化为外部 vlan。

```
#ovofd br-int
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=147294.121s, table=0, n_packets=224, n_bytes=33961, idle_age=13,
hard_age=65534, priority=3,in_port=4,dl_vlan=1 actions=mod_vlan_vid:101,NORMAL
 cookie=0x0, duration=603538.84s, table=0, n_packets=19, n_bytes=2234, idle_age=18963,
hard_age=65534, priority=2,in_port=4 actions=drop
 cookie=0x0, duration=603547.134s, table=0, n_packets=31901, n_bytes=6419756,
idle_age=13, hard_age=65534, priority=1 actions=NORMAL
```

dnsmasq 负责提供 DHCP 服务，绑定到某个特定的名字空间上，每个需要 DHCP 服务的租户网络有自己专属隔离的 DHCP 服务（图中的 tapXXX 和 tapWWW 上各自监听了一个 dnsmasq）。

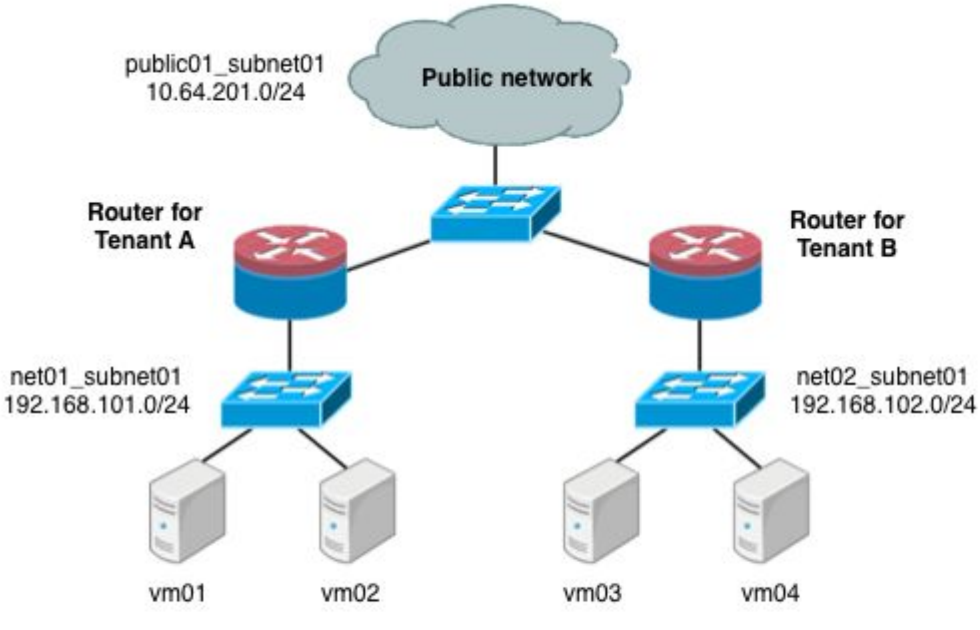
路由是 L3 agent 来实现，每个子网在 br-int 上有一个端口（qr-YYY 和 qr-ZZZ，已配置 IP，分别是各自内部子网的网关），L3 agent 绑定到上面。要访问外部的公共网络，需要通过 L3 agent 发出，而不是经过 int-br-ex 到 phy-br-ex（实际上并没有网包从这个 veth pair 传输）。如果要使用外部可见的 floating IP，L3 agent 仍然需要通过 iptables 来进行 NAT。

每个 L3 agent 或 dnsmasq 都在各自独立的名字空间中，如图表 5 所示，其中同一租户的两个子网都使用了同一个路由器。



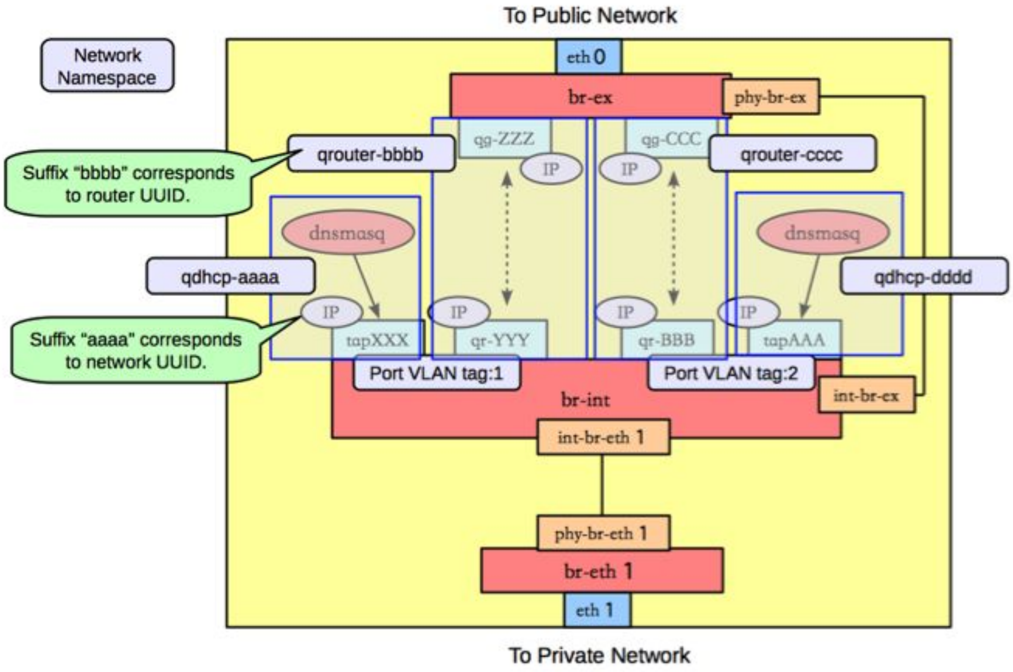
图表 8 每个网络功能进程都在自己的名字空间中

对于子网使用不同路由器的情况，多个路由器会在自己独立的名字空间中。例如要实现两个租户的两个子网的情况，如图表 6 所示。



图表 9 两个租户的两个子网的结构

这种情况下，网络节点上的名字空间如图表 7 所示。



图表 10 两个租户两个子网情况下的名字空间

#### 1.4.2.3 br-ex

br-ex 要做的事情很简单，只需要正常转发即可。

```
#ovofd br-ex
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=6770.969s, table=0, n_packets=5411, n_bytes=306944, idle_age=0,
hard_age=65534, priority=0 actions=NORMAL
```

## 1.5 Security Group

Security group 通过 Linux IPtables 来实现，为此，在 Compute 节点上引入了 qbr***这样的 Linux 传统 bridge（iptables 规则目前无法加载到直接挂在到 ovs 的 tap 设备上）。

首先在 Control 节点上用 neutron port-list 命令列出虚拟机的端口 id，例如：

```
# neutron port-list
+--------------------------------------+------+-------------------+--------------------------------------------------------------------------------------+
| id                                                              | name | mac_address              | fixed_ips
|
+--------------------------------------+------+-------------------+--------------------------------------------------------------------------------------+
|    2a169bb4-4d8b-4c67-802c-a24bdafe1312    |                          |    fa:16:3e:2f:e9:72    |    {"subnet_id":
"a2456a2c-5eea-416d-8757-d10bc0aa2aaa", "ip_address": "192.168.0.1"}      |
|     583c7038-d341-41ec-a0d1-0cd2c33866ca     |                          |     fa:16:3e:9c:dc:3a     |     {"subnet_id":
"a2456a2c-5eea-416d-8757-d10bc0aa2aaa", "ip_address": "192.168.0.2"}        |
|     9b2db4ac-3145-401c-8dc6-486ca6e303b6     |                          |     fa:16:3e:4e:f1:b5     |     {"subnet_id":
"ea4ed31b-e05a-4735-8c3f-9b430e656b64", "ip_address": "192.168.122.200"} |
|     c5a7d51b-9934-40bd-befa-adff840462d2     |                          |     fa:16:3e:21:1d:00     |     {"subnet_id":
"ea4ed31b-e05a-4735-8c3f-9b430e656b64", "ip_address": "192.168.122.201"} |
|     db2f5a49-7c0d-45dd-acad-908931f9a654     |                          |     fa:16:3e:17:5c:36     |     {"subnet_id":
"a2456a2c-5eea-416d-8757-d10bc0aa2aaa", "ip_address": "192.168.0.3"}        |
+--------------------------------------+------+-------------------+--------------------------------------------------------------------------------------+
```

其中 id 的前 10 位数字被用作虚机对外连接的 qbr（同时也是 tap 口）的 id。i 或 o 加上前 9 位数字被用作安全组 chain 的 id。

所有的规则默认都在 Compute 节点上的 filter 表（默认表）中实现，分别来查看 filter 表的 INPUT、OUTPUT、FORWARD 三条链上的规则。

在 Compute 节点上，可以用 iptables --line-numbers -vnL [CHAIN]来获得 filter 表（可以指定某个链上的）规则。

## 1.5.1 INPUT

```
#iptables --line-numbers -vnL INPUT
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
num      pkts bytes target          prot opt in         out         source                          destination
1           360K       56M  neutron-openvswi-INPUT     all     --     *             *                    0.0.0.0/0
0.0.0.0/0
2       10583 2146K ACCEPT          tcp    --    *           *            192.168.122.100           0.0.0.0/0
multiport dports 5666 /* 001 nagios-nrpe incoming 192.168.122.100 */
3           846 50966 ACCEPT          tcp    --    *           *            192.168.122.100           0.0.0.0/0
multiport dports 5900:5999 /* 001 nova compute incoming 192.168.122.100 */
4       1033K    894M ACCEPT          all    --    *           *            0.0.0.0/0                 0.0.0.0/0
state RELATED,ESTABLISHED
5           760 63840 ACCEPT          icmp --    *           *            0.0.0.0/0                 0.0.0.0/0
6                1       60 ACCEPT          all    --    lo         *            0.0.0.0/0                 0.0.0.0/0
7           977 58620 ACCEPT          tcp    --    *           *            0.0.0.0/0                 0.0.0.0/0
state NEW tcp dpt:22
8         3899 1194K REJECT          all    --    *           *            0.0.0.0/0                 0.0.0.0/0
reject-with icmp-host-prohibited
```

可以看到，跟安全组相关的规则被重定向到 neutron-openvswi-INPUT。

查看其规则，只有一条。

```
#iptables --line-numbers -vnL neutron-openvswi-INPUT
Chain neutron-openvswi-INPUT (1 references)
num      pkts bytes target          prot opt in         out         source                          destination
```

```
1          0      0 neutron-openvswi-o583c7038-d   all   --   *        *          0.0.0.0/0
0.0.0.0/0                PHYSDEV match --physdev-in tap583c7038-d3 --physdev-is-bridged
```

重定向到 neutron-openvswi-o583c7038-d。

```
#iptables --line-numbers -vnL neutron-openvswi-o583c7038-d
Chain neutron-openvswi-o583c7038-d (2 references)
num     pkts bytes target        prot opt in       out      source                  destination
1          3894  1199K  RETURN          udp     --    *           *                 0.0.0.0/0
0.0.0.0/0               udp spt:68 dpt:67
2       4282  1536K  neutron-openvswi-s583c7038-d   all   --   *        *          0.0.0.0/0
0.0.0.0/0
3               0         0  DROP              udp     --    *          *               0.0.0.0/0
0.0.0.0/0               udp spt:67 dpt:68
4          0      0 DROP          all   --   *        *         0.0.0.0/0               0.0.0.0/0
state INVALID
5       3971 1510K RETURN        all   --   *        *         0.0.0.0/0               0.0.0.0/0
state RELATED,ESTABLISHED
6        311 25752 RETURN        all   --   *        *         0.0.0.0/0               0.0.0.0/0
7            0        0  neutron-openvswi-sg-fallback   all   --   *        *          0.0.0.0/0
0.0.0.0/0
```

如果是 vm 发出的 dhcp 请求，直接通过，否则转到 neutron-openvswi-s583c7038-d。

```
#iptables --line-numbers -vnL neutron-openvswi-s583c7038-d
Chain neutron-openvswi-s583c7038-d (1 references)
num     pkts bytes target        prot opt in       out      source                  destination
1       4284 1537K RETURN        all   --   *        *         192.168.0.2             0.0.0.0/0
MAC FA:16:3E:9C:DC:3A
2          0      0 DROP          all   --   *        *         0.0.0.0/0               0.0.0.0/0
```

这条 chain 主要检查从 vm 发出来的网包，是否是 openstack 所分配的 IP 和 MAC，如果不匹配，则禁止通过。这将防止利用 vm 上进行一些伪装地址的攻击。

### 1.5.2 OUTPUT

```
#iptables --line-numbers -vnL OUTPUT
Chain OUTPUT (policy ACCEPT 965K packets, 149M bytes)
num     pkts bytes target        prot opt in       out      source                  destination
1          481K     107M  neutron-filter-top     all     --     *            *              0.0.0.0/0
0.0.0.0/0
2         481K    107M  neutron-openvswi-OUTPUT    all    --    *          *            0.0.0.0/0
0.0.0.0/0
```

分别跳转到 neutron-filter-top 和 neutron-openvswi-OUTPUT。

```
#iptables --line-numbers -vnL neutron-filter-top
```

```
Chain neutron-filter-top (2 references)
num      pkts bytes target          prot opt in         out        source                       destination
1           497K     112M  neutron-openvswi-local      all      --     *               *                    0.0.0.0/0
0.0.0.0/0
```

跳转到 neutron-openvswi-local。

```
#iptables --line-numbers -vnL neutron-openvswi-OUTPUT
Chain neutron-openvswi-OUTPUT (1 references)
num      pkts bytes target          prot opt in         out        source                       destination
```

该 chain 目前无规则。

```
#iptables --line-numbers -vnL neutron-openvswi-local
Chain neutron-openvswi-local (1 references)
num      pkts bytes target          prot opt in         out        source                       destination
```

该 chain 目前也无规则。

### 1.5.3 FORWARD

FORWARD chain 上主要实现安全组的功能。用户在配置缺省安全规则时候（例如允许 ssh 到 vm，允许 ping 到 vm），影响该 chain。

```
#iptables --line-numbers -vnL FORWARD
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
num      pkts bytes target          prot opt in         out        source                       destination
1           16203  5342K  neutron-filter-top      all      --      *                *                    0.0.0.0/0
0.0.0.0/0
2         16203  5342K  neutron-openvswi-FORWARD     all     --     *             *                 0.0.0.0/0
0.0.0.0/0
3              0         0 REJECT         all    --    *           *            0.0.0.0/0                   0.0.0.0/0
reject-with icmp-host-prohibited
```

同样跳转到 neutron-filter-top，无规则。跳转到 neutron-openvswi-FORWARD。

```
#iptables --line-numbers -vnL neutron-openvswi-FORWARD
Chain neutron-openvswi-FORWARD (1 references)
num      pkts bytes target          prot opt in         out        source                       destination
1           8170  2630K  neutron-openvswi-sg-chain     all     --     *              *                 0.0.0.0/0
0.0.0.0/0                    PHYSDEV match --physdev-out tap583c7038-d3 --physdev-is-bridged
2           8156  2729K  neutron-openvswi-sg-chain     all     --     *              *                 0.0.0.0/0
0.0.0.0/0                    PHYSDEV match --physdev-in tap583c7038-d3 --physdev-is-bridged
```

neutron-openvswi-FORWARD 将匹配所有进出 tap-XXX 端口的流量。

```
#iptables --line-numbers -vnL neutron-openvswi-sg-chain
Chain neutron-openvswi-sg-chain (2 references)
num      pkts bytes target          prot opt in         out        source                       destination
```

```
1        8170 2630K neutron-openvswi-i583c7038-d    all   --   *           *              0.0.0.0/0
0.0.0.0/0                 PHYSDEV match --physdev-out tap583c7038-d3 --physdev-is-bridged
2        8156 2729K neutron-openvswi-o583c7038-d    all   --   *           *              0.0.0.0/0
0.0.0.0/0                 PHYSDEV match --physdev-in tap583c7038-d3 --physdev-is-bridged
3      12442 4163K ACCEPT        all   --   *          *           0.0.0.0/0                 0.0.0.0/0
```

如果是网桥从 tap-XXX 端口发出到 VM 的流量，则跳转到 neutron-openvswi-i9LETTERID；如果是从 tap-XXX 端口进入到网桥的（即 vm 发出来的）流量，则跳转到 neutron-openvswi-o9 LETTERID。

```
#iptables --line-numbers -vnL neutron-openvswi-i583c7038-d
Chain neutron-openvswi-i583c7038-d (1 references)
num     pkts bytes target        prot opt in       out       source                     destination
1            0        0 DROP            all   --   *          *           0.0.0.0/0                 0.0.0.0/0
state INVALID
2          400 43350 RETURN         all   --   *          *           0.0.0.0/0                 0.0.0.0/0
state RELATED,ESTABLISHED
3                   1          60  RETURN            tcp     --     *              *               0.0.0.0/0
0.0.0.0/0                 tcp dpt:22
4                   1          84  RETURN            icmp  --     *              *               0.0.0.0/0
0.0.0.0/0
5            3885  1391K  RETURN            udp     --     *              *                 192.168.0.3
0.0.0.0/0                 udp spt:67 dpt:68
6          3885  1197K  neutron-openvswi-sg-fallback    all    --    *            *              0.0.0.0/0
0.0.0.0/0
```

neutron-openvswi-i9LETTERID 允许安全组中配置的策略（允许 ssh、ping 等）和 dhcp reply 通过。默认的 neutron-openvswi-sg-fallback 将 drop 所有流量。

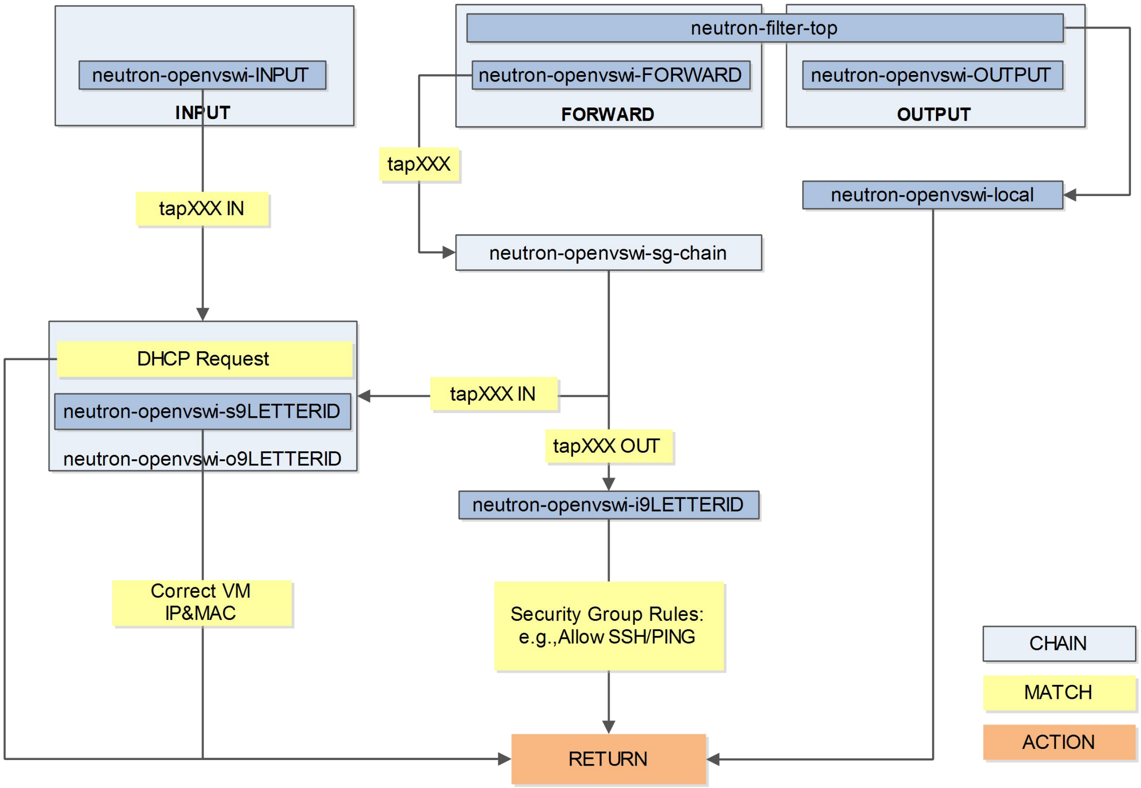
```
#iptables --line-numbers -vnL neutron-openvswi-o583c7038-d
Chain neutron-openvswi-o583c7038-d (2 references)
num     pkts bytes target        prot opt in       out       source                     destination
1            3886  1197K  RETURN            udp     --     *              *               0.0.0.0/0
0.0.0.0/0                 udp spt:68 dpt:67
2         4274 1533K neutron-openvswi-s583c7038-d    all   --   *           *              0.0.0.0/0
0.0.0.0/0
3                   0             0  DROP                 udp     --     *              *               0.0.0.0/0
0.0.0.0/0                 udp spt:67 dpt:68
4            0        0 DROP            all   --   *          *           0.0.0.0/0                 0.0.0.0/0
state INVALID
5        3963 1507K RETURN         all   --   *          *           0.0.0.0/0                 0.0.0.0/0
state RELATED,ESTABLISHED
6         311 25752 RETURN         all   --   *          *           0.0.0.0/0                 0.0.0.0/0
```

```
7             0         0 neutron-openvswi-sg-fallback    all    --    *           *                0.0.0.0/0
0.0.0.0/0
```

neutron-openvswi-o9LETTERID 将跳转到 neutron-openvswi-s583c7038-d，允许 DHCP Request 和匹配 VM 的源 IP 和源 MAC 的流量通过。

### 1.5.4 整体逻辑

整体逻辑如图表 3 所示。



图表 11 OpenStack 中 iptables 的整体逻辑

### 1.5.5 快速查找安全组规则

从前面分析可以看出，某个 vm 的安全组相关规则的 chain 的名字，跟 vm 的 id 的前 9 个字符有关。

因此，要快速查找 qbr-XXX 上相关的 iptables 规则，可以用 iptables -S 列出（默认是 filter 表）所有链上的规则，其中含有 id 的链即为虚拟机相关的安全组规则。其中--physdev-in 表示即将进入某个网桥的端口，--physdev-out 表示即将从某个网桥端口发出。

```
#iptables -S |grep tap583c7038-d3
-A     neutron-openvswi-FORWARD     -m     physdev     --physdev-out     tap583c7038-d3
--physdev-is-bridged -j neutron-openvswi-sg-chain
```

```
-A neutron-openvswi-FORWARD -m physdev --physdev-in tap583c7038-d3 --physdev-is-bridged -j neutron-openvswi-sg-chain
-A neutron-openvswi-INPUT -m physdev --physdev-in tap583c7038-d3 --physdev-is-bridged -j neutron-openvswi-o583c7038-d
-A neutron-openvswi-sg-chain -m physdev --physdev-out tap583c7038-d3 --physdev-is-bridged -j neutron-openvswi-i583c7038-d
-A neutron-openvswi-sg-chain -m physdev --physdev-in tap583c7038-d3 --physdev-is-bridged -j neutron-openvswi-o583c7038-d
```

可以看出，进出 tap-XXX 口的 FORWARD 链上的流量都被扔到了 neutron-openvswi-sg-chain 这个链，neutron-openvswi-sg-chain 上是 security group 具体的实现（两条规则，访问虚拟机的流量扔给 neutron-openvswi-i583c7038-d；从虚拟机出来的扔给 neutron-openvswi-o583c7038-d）。

### 1.5.6 其它

安全组在 Havana 版本中，默认是开启的，如果安装完毕后发现找不到 qbr-*网桥，则可以检查在 nova.conf 里面是否设置以下内容：

```
libvirt_vif_driver=nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver
```

## 1.6 参考

[1] http://openstack.redhat.com/Networking_in_too_much_detail

[2] http://masimum.inf.um.es/fjrm/2013/12/26/the-journey-of-a-packet-within-an-openstack-cloud/

[3] http://packetpushers.net/openstack-neutron-network-implementation-in-linux/

[4] http://masimum.inf.um.es/fjrm/2013/12/26/the-journey-of-a-packet-within-an-openstack-cloud/

[5] http://blog.scottlowe.org/2013/09/04/introducing-linux-network-namespaces/

[6] http://assafmuller.wordpress.com/2013/10/14/gre-tunnels-in-openstack-neutron/

[7] http://lwn.net/Articles/580893/

## 1.7 附：安装配置

本文中示例以 havana 版本为例。

控制节点和计算节点分开，均为双网卡，eth0 为 openstack 内部数据网，eth1 为 openstack 管理网（同时为外部控制网）。

Managment Network: 192.168.122.0/24

Data Network: 10.0.0.0/24

Control Server: 10.0.0.100 (eth0), 192.168.122.100 (eth1)

Compute Server: 10.0.0.101 (eth0), 192.168.122.101 (eth1)

安装利用 redhat 的 rdo。

安装和部署所采用的工具，可以从 https://github.com/yeasy/openstack-tool 下载。

### 1.7.1 RDO answer 文件-GRE 模式

```
[general]

# Path to a Public key to install on servers. If a usable key has not
# been installed on the remote servers the user will be prompted for a
# password and this key will be installed so the password will not be
# required again
CONFIG_SSH_KEY=/root/.ssh/id_rsa.pub

# Set to 'y' if you would like Packstack to install MySQL
CONFIG_MYSQL_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Image
# Service (Glance)
CONFIG_GLANCE_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Block
# Storage (Cinder)
CONFIG_CINDER_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Compute
# (Nova)
CONFIG_NOVA_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Networking (Neutron)
```

```
CONFIG_NEUTRON_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Dashboard (Horizon)
CONFIG_HORIZON_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Object
# Storage (Swift)
CONFIG_SWIFT_INSTALL=n

# Set to 'y' if you would like Packstack to install OpenStack
# Metering (Ceilometer)
CONFIG_CEILOMETER_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Orchestration (Heat)
CONFIG_HEAT_INSTALL=y

# Set to 'y' if you would like Packstack to install the OpenStack
# Client packages. An admin "rc" file will also be installed
CONFIG_CLIENT_INSTALL=y

# Comma separated list of NTP servers. Leave plain if Packstack
# should not install ntpd on instances.
CONFIG_NTP_SERVERS=pool.ntp.org

# Set to 'y' if you would like Packstack to install Nagios to monitor
# OpenStack hosts
CONFIG_NAGIOS_INSTALL=n

# Comma separated list of servers to be excluded from installation in
# case you are running Packstack the second time with the same answer
# file and don't want Packstack to touch these servers. Leave plain if
# you don't need to exclude any server.
EXCLUDE_SERVERS=

# Set to 'y' if you want to run OpenStack services in debug mode.
# Otherwise set to 'n'.
CONFIG_DEBUG_MODE=n

# The IP address of the server on which to install MySQL
```

```
CONFIG_MYSQL_HOST=192.168.122.100

# Username for the MySQL admin user
CONFIG_MYSQL_USER=root

# Password for the MySQL admin user
CONFIG_MYSQL_PW=root

# The IP address of the server on which to install the QPID service
CONFIG_QPID_HOST=192.168.122.100

# Enable SSL for the QPID service
CONFIG_QPID_ENABLE_SSL=n

# Enable Authentication for the QPID service
CONFIG_QPID_ENABLE_AUTH=n

# The password for the NSS certificate database of the QPID service
CONFIG_QPID_NSS_CERTDB_PW=4afeb1315f9341ec994ccfb5c0ace171

# The port in which the QPID service listens to SSL connections
CONFIG_QPID_SSL_PORT=5671

# The filename of the certificate that the QPID service is going to
# use
CONFIG_QPID_SSL_CERT_FILE=/etc/pki/tls/certs/qpid_selfcert.pem

# The filename of the private key that the QPID service is going to
# use
CONFIG_QPID_SSL_KEY_FILE=/etc/pki/tls/private/qpid_selfkey.pem

# Auto Generates self signed SSL certificate and key
CONFIG_QPID_SSL_SELF_SIGNED=y

# User for qpid authentication
CONFIG_QPID_AUTH_USER=qpid_user

# Password for user authentication
CONFIG_QPID_AUTH_PASSWORD=19f1540f541d402e

# The IP address of the server on which to install Keystone
```

```
CONFIG_KEYSTONE_HOST=192.168.122.100

# The password to use for the Keystone to access DB
CONFIG_KEYSTONE_DB_PW=0a9fbe8bcfc14091

# The token to use for the Keystone service api
CONFIG_KEYSTONE_ADMIN_TOKEN=bfab1155f8944cb49dc0d745fa52ec51

# The password to use for the Keystone admin user
CONFIG_KEYSTONE_ADMIN_PW=admin

# The password to use for the Keystone demo user
CONFIG_KEYSTONE_DEMO_PW=a93b7421f3ae4b18

# Kestone token format. Use either UUID or PKI
CONFIG_KEYSTONE_TOKEN_FORMAT=PKI

# The IP address of the server on which to install Glance
CONFIG_GLANCE_HOST=192.168.122.100

# The password to use for the Glance to access DB
CONFIG_GLANCE_DB_PW=78c3cc98e8a94fef

# The password to use for the Glance to authenticate with Keystone
CONFIG_GLANCE_KS_PW=e47ccb960cb74c4c

# The IP address of the server on which to install Cinder
CONFIG_CINDER_HOST=192.168.122.100

# The password to use for the Cinder to access DB
CONFIG_CINDER_DB_PW=0042879961db474e

# The password to use for the Cinder to authenticate with Keystone
CONFIG_CINDER_KS_PW=ac0b3965e2a34e4f

# The Cinder backend to use, valid options are: lvm, gluster, nfs
CONFIG_CINDER_BACKEND=lvm

# Create Cinder's volumes group. This should only be done for testing
# on a proof-of-concept installation of Cinder.    This will create a
# file-backed volume group and is not suitable for production usage.
```

```
CONFIG_CINDER_VOLUMES_CREATE=y

# Cinder's volumes group size. Note that actual volume size will be
# extended with 3% more space for VG metadata.
CONFIG_CINDER_VOLUMES_SIZE=20G

# A single or comma separated list of gluster volume shares to mount,
# eg: ip-address:/vol-name, domain:/vol-name
CONFIG_CINDER_GLUSTER_MOUNTS=

# A single or comma seprated list of NFS exports to mount, eg: ip-
# address:/export-name
CONFIG_CINDER_NFS_MOUNTS=

# The IP address of the server on which to install the Nova API
# service
CONFIG_NOVA_API_HOST=192.168.122.100

# The IP address of the server on which to install the Nova Cert
# service
CONFIG_NOVA_CERT_HOST=192.168.122.100

# The IP address of the server on which to install the Nova VNC proxy
CONFIG_NOVA_VNCPROXY_HOST=192.168.122.100

# A comma separated list of IP addresses on which to install the Nova
# Compute services
CONFIG_NOVA_COMPUTE_HOSTS=192.168.122.101

# The IP address of the server on which to install the Nova Conductor
# service
CONFIG_NOVA_CONDUCTOR_HOST=192.168.122.100

# The password to use for the Nova to access DB
CONFIG_NOVA_DB_PW=622b61c95e334c36

# The password to use for the Nova to authenticate with Keystone
CONFIG_NOVA_KS_PW=0573f5812091497a

# The IP address of the server on which to install the Nova Scheduler
# service
```

```
CONFIG_NOVA_SCHED_HOST=192.168.122.100

# The overcommitment ratio for virtual to physical CPUs. Set to 1.0
# to disable CPU overcommitment
CONFIG_NOVA_SCHED_CPU_ALLOC_RATIO=16.0

# The overcommitment ratio for virtual to physical RAM. Set to 1.0 to
# disable RAM overcommitment
CONFIG_NOVA_SCHED_RAM_ALLOC_RATIO=1.5

# Private interface for Flat DHCP on the Nova compute servers
CONFIG_NOVA_COMPUTE_PRIVIF=eth0

# The list of IP addresses of the server on which to install the Nova
# Network service
CONFIG_NOVA_NETWORK_HOSTS=192.168.122.100

# Nova network manager
CONFIG_NOVA_NETWORK_MANAGER=nova.network.manager.FlatDHCPManager

# Public interface on the Nova network server
CONFIG_NOVA_NETWORK_PUBIF=eth1

# Private interface for network manager on the Nova network server
CONFIG_NOVA_NETWORK_PRIVIF=eth0

# IP Range for network manager
CONFIG_NOVA_NETWORK_FIXEDRANGE=192.168.32.0/22

# IP Range for Floating IP's
CONFIG_NOVA_NETWORK_FLOATRANGE=10.3.4.0/22

# Name of the default floating pool to which the specified floating
# ranges are added to
CONFIG_NOVA_NETWORK_DEFAULTFLOATINGPOOL=nova

# Automatically assign a floating IP to new instances
CONFIG_NOVA_NETWORK_AUTOASSIGNFLOATINGIP=n

# First VLAN for private networks
CONFIG_NOVA_NETWORK_VLAN_START=100
```

```
# Number of networks to support
CONFIG_NOVA_NETWORK_NUMBER=1

# Number of addresses in each private subnet
CONFIG_NOVA_NETWORK_SIZE=255

# The IP addresses of the server on which to install the Neutron
# server
CONFIG_NEUTRON_SERVER_HOST=192.168.122.100

# The password to use for Neutron to authenticate with Keystone
CONFIG_NEUTRON_KS_PW=906aae1c5727416b

# The password to use for Neutron to access DB
CONFIG_NEUTRON_DB_PW=4ef2c9f1292c4feb

# A comma separated list of IP addresses on which to install Neutron
# L3 agent
CONFIG_NEUTRON_L3_HOSTS=192.168.122.100

# The name of the bridge that the Neutron L3 agent will use for
# external traffic, or 'provider' if using provider networks
CONFIG_NEUTRON_L3_EXT_BRIDGE=br-ex

# A comma separated list of IP addresses on which to install Neutron
# DHCP agent
CONFIG_NEUTRON_DHCP_HOSTS=192.168.122.100

# A comma separated list of IP addresses on which to install Neutron
# LBaaS agent
CONFIG_NEUTRON_LBAAS_HOSTS=

# The name of the L2 plugin to be used with Neutron
CONFIG_NEUTRON_L2_PLUGIN=openvswitch

# A comma separated list of IP addresses on which to install Neutron
# metadata agent
CONFIG_NEUTRON_METADATA_HOSTS=192.168.122.100

# A comma separated list of IP addresses on which to install Neutron
# metadata agent
CONFIG_NEUTRON_METADATA_PW=31620943f151436c

# A comma separated list of network type driver entrypoints to be
# loaded from the neutron.ml2.type_drivers namespace.
CONFIG_NEUTRON_ML2_TYPE_DRIVERS=local

# A comma separated ordered list of network_types to allocate as
# tenant networks. The value 'local' is only useful for single-box
# testing but provides no connectivity between hosts.
CONFIG_NEUTRON_ML2_TENANT_NETWORK_TYPES=local

# A comma separated ordered list of networking mechanism driver
# entrypoints to be loaded from the neutron.ml2.mechanism_drivers
# namespace.
CONFIG_NEUTRON_ML2_MECHANISM_DRIVERS=openvswitch

# A comma separated    list of physical_network names with which flat
# networks can be created. Use * to allow flat networks with arbitrary
# physical_network names.
CONFIG_NEUTRON_ML2_FLAT_NETWORKS=*

# A comma separated list of <physical_network>:<vlan_min>:<vlan_max>
# or <physical_network> specifying physical_network names usable for
# VLAN provider and tenant networks, as well as ranges of VLAN tags on
# each available for allocation to tenant networks.
CONFIG_NEUTRON_ML2_VLAN_RANGES=

# A comma separated list of <tun_min>:<tun_max> tuples enumerating
# ranges of GRE tunnel IDs that are available for tenant network
# allocation. Should be an array with tun_max +1 - tun_min > 1000000
CONFIG_NEUTRON_ML2_TUNNEL_ID_RANGES=

# Multicast group for VXLAN. If unset, disables VXLAN enable sending
# allocate broadcast traffic to this multicast group. When left
# unconfigured, will disable multicast VXLAN mode. Should be an
# Multicast IP (v4 or v6) address.
CONFIG_NEUTRON_ML2_VXLAN_GROUP=

# A comma separated list of <vni_min>:<vni_max> tuples enumerating
# ranges of VXLAN VNI IDs that are available for tenant network
# allocation. Min value is 0 and Max value is 16777215.
CONFIG_NEUTRON_ML2_VNI_RANGES=

# The name of the L2 agent to be used with Neutron
CONFIG_NEUTRON_L2_AGENT=openvswitch

# The type of network to allocate for tenant networks (eg. vlan,
# local)
CONFIG_NEUTRON_LB_TENANT_NETWORK_TYPE=local

# A comma separated list of VLAN ranges for the Neutron linuxbridge
# plugin (eg. physnet1:1:4094,physnet2,physnet3:3000:3999)
CONFIG_NEUTRON_LB_VLAN_RANGES=

# A comma separated list of interface mappings for the Neutron
# linuxbridge plugin (eg. physnet1:br-eth1,physnet2:br-eth2,physnet3
# :br-eth3)
CONFIG_NEUTRON_LB_INTERFACE_MAPPINGS=

# Type of network to allocate for tenant networks (eg. vlan, local,
# gre, vxlan)
CONFIG_NEUTRON_OVS_TENANT_NETWORK_TYPE=gre

# A comma separated list of VLAN ranges for the Neutron openvswitch
# plugin (eg. physnet1:1:4094,physnet2,physnet3:3000:3999)
CONFIG_NEUTRON_OVS_VLAN_RANGES=

# A comma separated list of bridge mappings for the Neutron
# openvswitch plugin (eg. physnet1:br-eth1,physnet2:br-eth2,physnet3
# :br-eth3)
CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=

# A comma separated list of colon-separated OVS bridge:interface
# pairs. The interface will be added to the associated bridge.
CONFIG_NEUTRON_OVS_BRIDGE_IFACES=

# A comma separated list of tunnel ranges for the Neutron openvswitch
# plugin (eg. 1:1000)
CONFIG_NEUTRON_OVS_TUNNEL_RANGES=1:1000

# The interface for the OVS tunnel. Packstack will override the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface. (eg. eth1)
CONFIG_NEUTRON_OVS_TUNNEL_IF=eth0

# VXLAN UDP port
CONFIG_NEUTRON_OVS_VXLAN_UDP_PORT=4789

# The IP address of the server on which to install the OpenStack
# client packages. An admin "rc" file will also be installed
CONFIG_OSCLIENT_HOST=192.168.122.100

# The IP address of the server on which to install Horizon
CONFIG_HORIZON_HOST=192.168.122.100

# To set up Horizon communication over https set this to "y"
CONFIG_HORIZON_SSL=n

# PEM encoded certificate to be used for ssl on the https server,
# leave blank if one should be generated, this certificate should not
# require a passphrase
CONFIG_SSL_CERT=

# Keyfile corresponding to the certificate if one was entered
CONFIG_SSL_KEY=

# The IP address on which to install the Swift proxy service
# (currently only single proxy is supported)
CONFIG_SWIFT_PROXY_HOSTS=192.168.122.100

# The password to use for the Swift to authenticate with Keystone
CONFIG_SWIFT_KS_PW=85f42ee1d9604761

# A comma separated list of IP addresses on which to install the
# Swift Storage services, each entry should take the format
# <ipaddress>[/dev], for example 127.0.0.1/vdb will install /dev/vdb
# on 127.0.0.1 as a swift storage device(packstack does not create the
# filesystem, you must do this first), if /dev is omitted Packstack
# will create a loopback device for a test setup
CONFIG_SWIFT_STORAGE_HOSTS=192.168.122.100

# Number of swift storage zones, this number MUST be no bigger than
# the number of storage devices configured
CONFIG_SWIFT_STORAGE_ZONES=1

# Number of swift storage replicas, this number MUST be no bigger
# than the number of storage zones configured
CONFIG_SWIFT_STORAGE_REPLICAS=1

# FileSystem type for storage nodes
CONFIG_SWIFT_STORAGE_FSTYPE=ext4

# Shared secret for Swift
CONFIG_SWIFT_HASH=dcd782d154134ed5

# Size of the swift loopback file storage device
CONFIG_SWIFT_STORAGE_SIZE=2G

# Whether to provision for demo usage and testing
CONFIG_PROVISION_DEMO=n

# The CIDR network address for the floating IP subnet
CONFIG_PROVISION_DEMO_FLOATRANGE=172.24.4.224/28

# Whether to configure tempest for testing
CONFIG_PROVISION_TEMPEST=n

# The uri of the tempest git repository to use
CONFIG_PROVISION_TEMPEST_REPO_URI=https://github.com/openstack/tempest.git

# The revision of the tempest git repository to use
CONFIG_PROVISION_TEMPEST_REPO_REVISION=stable/havana

# Whether to configure the ovs external bridge in an all-in-one
# deployment
CONFIG_PROVISION_ALL_IN_ONE_OVS_BRIDGE=n

# The IP address of the server on which to install Heat service
CONFIG_HEAT_HOST=192.168.122.100

# The password used by Heat user to authenticate against MySQL
CONFIG_HEAT_DB_PW=db67579b56ea4bcd
```

```
# The password to use for the Heat to authenticate with Keystone
CONFIG_HEAT_KS_PW=0ecab082910c4fab

# Set to 'y' if you would like Packstack to install Heat CloudWatch
# API
CONFIG_HEAT_CLOUDWATCH_INSTALL=n

# Set to 'y' if you would like Packstack to install Heat
# CloudFormation API
CONFIG_HEAT_CFN_INSTALL=n

# The IP address of the server on which to install Heat CloudWatch
# API service
CONFIG_HEAT_CLOUDWATCH_HOST=192.168.122.100

# The IP address of the server on which to install Heat
# CloudFormation API service
CONFIG_HEAT_CFN_HOST=192.168.122.100

# The IP address of the server on which to install Ceilometer
CONFIG_CEILOMETER_HOST=192.168.122.100

# Secret key for signing metering messages.
CONFIG_CEILOMETER_SECRET=339bb60f1d79431d

# The password to use for Ceilometer to authenticate with Keystone
CONFIG_CEILOMETER_KS_PW=9364b11a6575405f

# The IP address of the server on which to install the Nagios server
CONFIG_NAGIOS_HOST=192.168.122.100

# The password of the nagiosadmin user on the Nagios server
CONFIG_NAGIOS_PW=593a5048a8ed4bb8

# To subscribe each server to EPEL enter "y"
CONFIG_USE_EPEL=n

# A comma separated list of URLs to any additional yum repositories
# to install
CONFIG_REPO=
# To subscribe each server with Red Hat subscription manager, include
# this with CONFIG_RH_PW
CONFIG_RH_USER=

# To subscribe each server with Red Hat subscription manager, include
# this with CONFIG_RH_USER
CONFIG_RH_PW=

# To subscribe each server to Red Hat Enterprise Linux 6 Server Beta
# channel (only needed for Preview versions of RHOS) enter "y"
CONFIG_RH_BETA_REPO=n

# To subscribe each server with RHN Satellite,fill Satellite's URL
# here. Note that either satellite's username/password or activation
# key has to be provided
CONFIG_SATELLITE_URL=

# Username to access RHN Satellite
CONFIG_SATELLITE_USER=

# Password to access RHN Satellite
CONFIG_SATELLITE_PW=

# Activation key for subscription to RHN Satellite
CONFIG_SATELLITE_AKEY=

# Specify a path or URL to a SSL CA certificate to use
CONFIG_SATELLITE_CACERT=

# If required specify the profile name that should be used as an
# identifier for the system in RHN Satellite
CONFIG_SATELLITE_PROFILE=

# Comma separated list of flags passed to rhnreg_ks. Valid flags are:
# novirtinfo, norhnsd, nopackages
CONFIG_SATELLITE_FLAGS=

# Specify a HTTP proxy to use with RHN Satellite
CONFIG_SATELLITE_PROXY=

# Specify a username to use with an authenticated HTTP proxy
```

```
CONFIG_SATELLITE_PROXY_USER=

# Specify a password to use with an authenticated HTTP proxy.
CONFIG_SATELLITE_PROXY_PW=
```

### 1.7.2 RDO answer 文件-VLAN 模式

```
[general]

# Path to a Public key to install on servers. If a usable key has not
# been installed on the remote servers the user will be prompted for a
# password and this key will be installed so the password will not be
# required again
CONFIG_SSH_KEY=/root/.ssh/id_rsa.pub

# Set to 'y' if you would like Packstack to install MySQL
CONFIG_MYSQL_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Image
# Service (Glance)
CONFIG_GLANCE_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Block
# Storage (Cinder)
CONFIG_CINDER_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Compute
# (Nova)
CONFIG_NOVA_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Networking (Neutron)
CONFIG_NEUTRON_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Dashboard (Horizon)
CONFIG_HORIZON_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack Object
# Storage (Swift)
CONFIG_SWIFT_INSTALL=n
```

```
# Set to 'y' if you would like Packstack to install OpenStack
# Metering (Ceilometer)
CONFIG_CEILOMETER_INSTALL=y

# Set to 'y' if you would like Packstack to install OpenStack
# Orchestration (Heat)
CONFIG_HEAT_INSTALL=y

# Set to 'y' if you would like Packstack to install the OpenStack
# Client packages. An admin "rc" file will also be installed
CONFIG_CLIENT_INSTALL=y

# Comma separated list of NTP servers. Leave plain if Packstack
# should not install ntpd on instances.
CONFIG_NTP_SERVERS=0.pool.ntp.org,1.pool.ntp.org,2.pool.ntp.org,3.pool.ntp.org

# Set to 'y' if you would like Packstack to install Nagios to monitor
# OpenStack hosts
CONFIG_NAGIOS_INSTALL=y

# Comma separated list of servers to be excluded from installation in
# case you are running Packstack the second time with the same answer
# file and don't want Packstack to touch these servers. Leave plain if
# you don't need to exclude any server.
EXCLUDE_SERVERS=

# Set to 'y' if you want to run OpenStack services in debug mode.
# Otherwise set to 'n'.
CONFIG_DEBUG_MODE=n

# The IP address of the server on which to install MySQL
CONFIG_MYSQL_HOST=192.168.122.100

# Username for the MySQL admin user
CONFIG_MYSQL_USER=root

# Password for the MySQL admin user
CONFIG_MYSQL_PW=root

# The IP address of the server on which to install the QPID service
```

```
CONFIG_QPID_HOST=192.168.122.100

# Enable SSL for the QPID service
CONFIG_QPID_ENABLE_SSL=n

# Enable Authentication for the QPID service
CONFIG_QPID_ENABLE_AUTH=n

# The password for the NSS certificate database of the QPID service
CONFIG_QPID_NSS_CERTDB_PW=4afeb1315f9341ec994ccfb5c0ace171

# The port in which the QPID service listens to SSL connections
CONFIG_QPID_SSL_PORT=5671

# The filename of the certificate that the QPID service is going to
# use
CONFIG_QPID_SSL_CERT_FILE=/etc/pki/tls/certs/qpid_selfcert.pem

# The filename of the private key that the QPID service is going to
# use
CONFIG_QPID_SSL_KEY_FILE=/etc/pki/tls/private/qpid_selfkey.pem

# Auto Generates self signed SSL certificate and key
CONFIG_QPID_SSL_SELF_SIGNED=y

# User for qpid authentication
CONFIG_QPID_AUTH_USER=qpid_user

# Password for user authentication
CONFIG_QPID_AUTH_PASSWORD=19f1540f541d402e

# The IP address of the server on which to install Keystone
CONFIG_KEYSTONE_HOST=192.168.122.100

# The password to use for the Keystone to access DB
CONFIG_KEYSTONE_DB_PW=0a9fbe8bcfc14091

# The token to use for the Keystone service api
CONFIG_KEYSTONE_ADMIN_TOKEN=bfab1155f8944cb49dc0d745fa52ec51

# The password to use for the Keystone admin user
```

```
CONFIG_KEYSTONE_ADMIN_PW=admin

# The password to use for the Keystone demo user
CONFIG_KEYSTONE_DEMO_PW=a93b7421f3ae4b18

# Kestone token format. Use either UUID or PKI
CONFIG_KEYSTONE_TOKEN_FORMAT=PKI

# The IP address of the server on which to install Glance
CONFIG_GLANCE_HOST=192.168.122.100

# The password to use for the Glance to access DB
CONFIG_GLANCE_DB_PW=78c3cc98e8a94fef

# The password to use for the Glance to authenticate with Keystone
CONFIG_GLANCE_KS_PW=e47ccb960cb74c4c

# The IP address of the server on which to install Cinder
CONFIG_CINDER_HOST=192.168.122.100

# The password to use for the Cinder to access DB
CONFIG_CINDER_DB_PW=0042879961db474e

# The password to use for the Cinder to authenticate with Keystone
CONFIG_CINDER_KS_PW=ac0b3965e2a34e4f

# The Cinder backend to use, valid options are: lvm, gluster, nfs
CONFIG_CINDER_BACKEND=nfs

# Create Cinder's volumes group. This should only be done for testing
# on a proof-of-concept installation of Cinder. This will create a
# file-backed volume group and is not suitable for production usage.
CONFIG_CINDER_VOLUMES_CREATE=y

# Cinder's volumes group size. Note that actual volume size will be
# extended with 3% more space for VG metadata.
CONFIG_CINDER_VOLUMES_SIZE=20G

# A single or comma separated list of gluster volume shares to mount,
# eg: ip-address:/vol-name, domain:/vol-name
CONFIG_CINDER_GLUSTER_MOUNTS=
# A single or comma seprated list of NFS exports to mount, eg: ip-
# address:/export-name
CONFIG_CINDER_NFS_MOUNTS=192.168.105.78:/home/nfs

# The IP address of the server on which to install the Nova API
# service
CONFIG_NOVA_API_HOST=192.168.122.100

# The IP address of the server on which to install the Nova Cert
# service
CONFIG_NOVA_CERT_HOST=192.168.122.100

# The IP address of the server on which to install the Nova VNC proxy
CONFIG_NOVA_VNCPROXY_HOST=192.168.122.100

# A comma separated list of IP addresses on which to install the Nova
# Compute services
CONFIG_NOVA_COMPUTE_HOSTS=192.168.122.101

# The IP address of the server on which to install the Nova Conductor
# service
CONFIG_NOVA_CONDUCTOR_HOST=192.168.122.100

# The password to use for the Nova to access DB
CONFIG_NOVA_DB_PW=622b61c95e334c36

# The password to use for the Nova to authenticate with Keystone
CONFIG_NOVA_KS_PW=0573f5812091497a

# The IP address of the server on which to install the Nova Scheduler
# service
CONFIG_NOVA_SCHED_HOST=192.168.122.100

# The overcommitment ratio for virtual to physical CPUs. Set to 1.0
# to disable CPU overcommitment
CONFIG_NOVA_SCHED_CPU_ALLOC_RATIO=16.0

# The overcommitment ratio for virtual to physical RAM. Set to 1.0 to
# disable RAM overcommitment
CONFIG_NOVA_SCHED_RAM_ALLOC_RATIO=1.5
```

```
# Private interface for Flat DHCP on the Nova compute servers
CONFIG_NOVA_COMPUTE_PRIVIF=eth0

# The list of IP addresses of the server on which to install the Nova
# Network service
CONFIG_NOVA_NETWORK_HOSTS=192.168.122.100

# Nova network manager
CONFIG_NOVA_NETWORK_MANAGER=nova.network.manager.FlatDHCPManager

# Public interface on the Nova network server
CONFIG_NOVA_NETWORK_PUBIF=eth1

# Private interface for network manager on the Nova network server
CONFIG_NOVA_NETWORK_PRIVIF=eth0

# IP Range for network manager
CONFIG_NOVA_NETWORK_FIXEDRANGE=192.168.32.0/22

# IP Range for Floating IP's
CONFIG_NOVA_NETWORK_FLOATRANGE=10.3.4.0/22

# Name of the default floating pool to which the specified floating
# ranges are added to
CONFIG_NOVA_NETWORK_DEFAULTFLOATINGPOOL=nova

# Automatically assign a floating IP to new instances
CONFIG_NOVA_NETWORK_AUTOASSIGNFLOATINGIP=n

# First VLAN for private networks
CONFIG_NOVA_NETWORK_VLAN_START=100

# Number of networks to support
CONFIG_NOVA_NETWORK_NUMBER=1

# Number of addresses in each private subnet
CONFIG_NOVA_NETWORK_SIZE=255

# The IP addresses of the server on which to install the Neutron
# server
```

```
CONFIG_NEUTRON_SERVER_HOST=192.168.122.100

# The password to use for Neutron to authenticate with Keystone
CONFIG_NEUTRON_KS_PW=906aae1c5727416b

# The password to use for Neutron to access DB
CONFIG_NEUTRON_DB_PW=4ef2c9f1292c4feb

# A comma separated list of IP addresses on which to install Neutron
# L3 agent
CONFIG_NEUTRON_L3_HOSTS=192.168.122.100

# The name of the bridge that the Neutron L3 agent will use for
# external traffic, or 'provider' if using provider networks
CONFIG_NEUTRON_L3_EXT_BRIDGE=br-ex

# A comma separated list of IP addresses on which to install Neutron
# DHCP agent
CONFIG_NEUTRON_DHCP_HOSTS=192.168.122.100

# A comma separated list of IP addresses on which to install Neutron
# LBaaS agent
CONFIG_NEUTRON_LBAAS_HOSTS=

# The name of the L2 plugin to be used with Neutron
CONFIG_NEUTRON_L2_PLUGIN=openvswitch

# A comma separated list of IP addresses on which to install Neutron
# metadata agent
CONFIG_NEUTRON_METADATA_HOSTS=192.168.122.100

# A comma separated list of IP addresses on which to install Neutron
# metadata agent
CONFIG_NEUTRON_METADATA_PW=31620943f151436c

# A comma separated list of network type driver entrypoints to be
# loaded from the neutron.ml2.type_drivers namespace.
CONFIG_NEUTRON_ML2_TYPE_DRIVERS=local

# A comma separated ordered list of network_types to allocate as
# tenant networks. The value 'local' is only useful for single-box
# testing but provides no connectivity between hosts.
CONFIG_NEUTRON_ML2_TENANT_NETWORK_TYPES=local

# A comma separated ordered list of networking mechanism driver
# entrypoints to be loaded from the neutron.ml2.mechanism_drivers
# namespace.
CONFIG_NEUTRON_ML2_MECHANISM_DRIVERS=openvswitch

# A comma separated list of physical_network names with which flat
# networks can be created. Use * to allow flat networks with arbitrary
# physical_network names.
CONFIG_NEUTRON_ML2_FLAT_NETWORKS=*

# A comma separated list of <physical_network>:<vlan_min>:<vlan_max>
# or <physical_network> specifying physical_network names usable for
# VLAN provider and tenant networks, as well as ranges of VLAN tags on
# each available for allocation to tenant networks.
CONFIG_NEUTRON_ML2_VLAN_RANGES=

# A comma separated list of <tun_min>:<tun_max> tuples enumerating
# ranges of GRE tunnel IDs that are available for tenant network
# allocation. Should be an array with tun_max +1 - tun_min > 1000000
CONFIG_NEUTRON_ML2_TUNNEL_ID_RANGES=

# Multicast group for VXLAN. If unset, disables VXLAN enable sending
# allocate broadcast traffic to this multicast group. When left
# unconfigured, will disable multicast VXLAN mode. Should be an
# Multicast IP (v4 or v6) address.
CONFIG_NEUTRON_ML2_VXLAN_GROUP=

# A comma separated list of <vni_min>:<vni_max> tuples enumerating
# ranges of VXLAN VNI IDs that are available for tenant network
# allocation. Min value is 0 and Max value is 16777215.
CONFIG_NEUTRON_ML2_VNI_RANGES=

# The name of the L2 agent to be used with Neutron
CONFIG_NEUTRON_L2_AGENT=openvswitch

# The type of network to allocate for tenant networks (eg. vlan,
# local)
CONFIG_NEUTRON_LB_TENANT_NETWORK_TYPE=local
```

```
# A comma separated list of VLAN ranges for the Neutron linuxbridge
# plugin (eg. physnet1:1:4094,physnet2,physnet3:3000:3999)
CONFIG_NEUTRON_LB_VLAN_RANGES=

# A comma separated list of interface mappings for the Neutron
# linuxbridge plugin (eg. physnet1:br-eth1,physnet2:br-eth2,physnet3
# :br-eth3)
CONFIG_NEUTRON_LB_INTERFACE_MAPPINGS=

# Type of network to allocate for tenant networks (eg. vlan, local,
# gre, vxlan)
CONFIG_NEUTRON_OVS_TENANT_NETWORK_TYPE=vlan

# A comma separated list of VLAN ranges for the Neutron openvswitch
# plugin (eg. physnet1:1:4094,physnet2,physnet3:3000:3999)
CONFIG_NEUTRON_OVS_VLAN_RANGES=physnet1:1:1000

# A comma separated list of bridge mappings for the Neutron
# openvswitch plugin (eg. physnet1:br-eth1,physnet2:br-eth2,physnet3
# :br-eth3)
CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=physnet1:br-eth0

# A comma separated list of colon-separated OVS bridge:interface
# pairs. The interface will be added to the associated bridge.
CONFIG_NEUTRON_OVS_BRIDGE_IFACES=br-eth0:eth0

# A comma separated list of tunnel ranges for the Neutron openvswitch
# plugin (eg. 1:1000)
CONFIG_NEUTRON_OVS_TUNNEL_RANGES=

# The interface for the OVS tunnel. Packstack will override the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface. (eg. eth1)
CONFIG_NEUTRON_OVS_TUNNEL_IF=

# VXLAN UDP port
CONFIG_NEUTRON_OVS_VXLAN_UDP_PORT=4789

# The IP address of the server on which to install the OpenStack
# client packages. An admin "rc" file will also be installed
```

```
CONFIG_OSCLIENT_HOST=192.168.122.100

# The IP address of the server on which to install Horizon
CONFIG_HORIZON_HOST=192.168.122.100

# To set up Horizon communication over https set this to "y"
CONFIG_HORIZON_SSL=n

# PEM encoded certificate to be used for ssl on the https server,
# leave blank if one should be generated, this certificate should not
# require a passphrase
CONFIG_SSL_CERT=

# Keyfile corresponding to the certificate if one was entered
CONFIG_SSL_KEY=

# The IP address on which to install the Swift proxy service
# (currently only single proxy is supported)
CONFIG_SWIFT_PROXY_HOSTS=192.168.122.100

# The password to use for the Swift to authenticate with Keystone
CONFIG_SWIFT_KS_PW=85f42ee1d9604761

# A comma separated list of IP addresses on which to install the
# Swift Storage services, each entry should take the format
# <ipaddress>[/dev], for example 127.0.0.1/vdb will install /dev/vdb
# on 127.0.0.1 as a swift storage device(packstack does not create the
# filesystem, you must do this first), if /dev is omitted Packstack
# will create a loopback device for a test setup
CONFIG_SWIFT_STORAGE_HOSTS=192.168.122.100

# Number of swift storage zones, this number MUST be no bigger than
# the number of storage devices configured
CONFIG_SWIFT_STORAGE_ZONES=1

# Number of swift storage replicas, this number MUST be no bigger
# than the number of storage zones configured
CONFIG_SWIFT_STORAGE_REPLICAS=1

# FileSystem type for storage nodes
CONFIG_SWIFT_STORAGE_FSTYPE=ext4
```

```
# Shared secret for Swift
CONFIG_SWIFT_HASH=dcd782d154134ed5

# Size of the swift loopback file storage device
CONFIG_SWIFT_STORAGE_SIZE=2G

# Whether to provision for demo usage and testing
CONFIG_PROVISION_DEMO=n

# The CIDR network address for the floating IP subnet
CONFIG_PROVISION_DEMO_FLOATRANGE=172.24.4.224/28

# Whether to configure tempest for testing
CONFIG_PROVISION_TEMPEST=n

# The uri of the tempest git repository to use
CONFIG_PROVISION_TEMPEST_REPO_URI=https://github.com/openstack/tempest.git

# The revision of the tempest git repository to use
CONFIG_PROVISION_TEMPEST_REPO_REVISION=stable/havana

# Whether to configure the ovs external bridge in an all-in-one
# deployment
CONFIG_PROVISION_ALL_IN_ONE_OVS_BRIDGE=n

# The IP address of the server on which to install Heat service
CONFIG_HEAT_HOST=192.168.122.100

# The password used by Heat user to authenticate against MySQL
CONFIG_HEAT_DB_PW=db67579b56ea4bcd

# The password to use for the Heat to authenticate with Keystone
CONFIG_HEAT_KS_PW=0ecab082910c4fab

# Set to 'y' if you would like Packstack to install Heat CloudWatch
# API
CONFIG_HEAT_CLOUDWATCH_INSTALL=n

# Set to 'y' if you would like Packstack to install Heat
# CloudFormation API
```

```
CONFIG_HEAT_CFN_INSTALL=n

# The IP address of the server on which to install Heat CloudWatch
# API service
CONFIG_HEAT_CLOUDWATCH_HOST=192.168.122.100

# The IP address of the server on which to install Heat
# CloudFormation API service
CONFIG_HEAT_CFN_HOST=192.168.122.100

# The IP address of the server on which to install Ceilometer
CONFIG_CEILOMETER_HOST=192.168.122.100

# Secret key for signing metering messages.
CONFIG_CEILOMETER_SECRET=339bb60f1d79431d

# The password to use for Ceilometer to authenticate with Keystone
CONFIG_CEILOMETER_KS_PW=9364b11a6575405f

# The IP address of the server on which to install the Nagios server
CONFIG_NAGIOS_HOST=192.168.122.100

# The password of the nagiosadmin user on the Nagios server
CONFIG_NAGIOS_PW=nagiosadmin

# To subscribe each server to EPEL enter "y"
CONFIG_USE_EPEL=y

# A comma separated list of URLs to any additional yum repositories
# to install
CONFIG_REPO=

# To subscribe each server with Red Hat subscription manager, include
# this with CONFIG_RH_PW
CONFIG_RH_USER=

# To subscribe each server with Red Hat subscription manager, include
# this with CONFIG_RH_USER
CONFIG_RH_PW=

# To subscribe each server to Red Hat Enterprise Linux 6 Server Beta
# channel (only needed for Preview versions of RHOS) enter "y"
CONFIG_RH_BETA_REPO=n

# To subscribe each server with RHN Satellite,fill Satellite's URL
# here. Note that either satellite's username/password or activation
# key has to be provided
CONFIG_SATELLITE_URL=

# Username to access RHN Satellite
CONFIG_SATELLITE_USER=

# Password to access RHN Satellite
CONFIG_SATELLITE_PW=

# Activation key for subscription to RHN Satellite
CONFIG_SATELLITE_AKEY=

# Specify a path or URL to a SSL CA certificate to use
CONFIG_SATELLITE_CACERT=

# If required specify the profile name that should be used as an
# identifier for the system in RHN Satellite
CONFIG_SATELLITE_PROFILE=

# Comma separated list of flags passed to rhnreg_ks. Valid flags are:
# novirtinfo, norhnsd, nopackages
CONFIG_SATELLITE_FLAGS=

# Specify a HTTP proxy to use with RHN Satellite
CONFIG_SATELLITE_PROXY=

# Specify a username to use with an authenticated HTTP proxy
CONFIG_SATELLITE_PROXY_USER=

# Specify a password to use with an authenticated HTTP proxy.
CONFIG_SATELLITE_PROXY_PW=
```