租户带宽限制方案配置原理解析

1. Egress gateway的配置

对于egress gateway的网络空间，执行以下命令：

代码块

```
sysctl -w net.ipv4.ip_forward=1
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
ip link add vxlan0 type vxlan id 100 dstport 4789 local $podip dev eth0
ip addr add 172.16.0.1/24 dev vxlan0
ip link set vxlan0 up
# 删除现有规则（可选）
tc qdisc del dev eth0 root
# 添加根队列
tc qdisc add dev eth0 root handle 1: htb default 10
# 添加限速类。 出口带宽限制到10Mbit/s
tc class add dev eth0 parent 1: classid 1:1 htb rate 10mbit
# 所有流量匹配classid 1:1
tc filter add dev eth0 protocol ip parent 1:0 prio 1 u32 match ip dst
0.0.0.0/0 flowid 1:1
```

现在逐行进行解释：

代码块

```
sysctl -w net.ipv4.ip_forward=1
```

这是开始egress gateway的转发功能，因为其本身作为网关，需要将流量进行转发

代码块

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

这里是对从egress gateway的eth0出去的流量都进行snat。原因是从egress pod->gateway->公网这条链路上，回包时要回到egress pod，需要snat。

这里对所有出eth0的流量都snat，并不会造成其他影响。

代码块

```
ip link add vxlan0 type vxlan id 100 dstport 4789 local $podip dev eth0
ip addr add 172.16.0.1/24 dev vxlan0
ip link set vxlan0 up
```

这里是对vxlan进行配置，其中$podip为egress gateway自身的ip，172.16.0.1/24为分配给vxlan设备的ip，并进行启动。

代码块

```
# 删除现有规则（可选）
tc qdisc del dev eth0 root
# 添加根队列
tc qdisc add dev eth0 root handle 1: htb default 10
# 添加限速类。 出口带宽限制到10Mbit/s
tc class add dev eth0 parent 1: classid 1:1 htb rate 10mbit
# 所有流量匹配classid 1:1
tc filter add dev eth0 protocol ip parent 1:0 prio 1 u32 match ip dst 0.0.0.0/0
 flowid 1:1
```

以下是对这些 `tc` 命令的逐行解释及注意事项：

代码块

```
# 删除现有规则（可选）
tc qdisc del dev eth0 root
```

作用：删除 eth0 网卡上的根队列规则 ，若原有 QoS 规则存在，会直接清除所有流量控制策略

代码块

```
# 添加根队列
tc qdisc add dev eth0 root handle 1: htb default 10
```

参数解析：

- `root`：作为根队列规则生效
- `handle 1:`：设置队列的句柄标识符（格式为 `主编号:子编号`）
- `htb`：使用分层令牌桶算法
- `default 10`：未分类流量默认发送到 classid 1:10

代码块

```
# 添加限速类。出口带宽限制到10Mbit/s
```

```
2   tc class add dev eth0 parent 1: classid 1:1 htb rate 10mbit
```

关键参数：

- `parent 1:`：隶属于根队列 1:
- `classid 1:1`：子类唯一标识符
- `rate 10mbit`：保证带宽 10Mbps

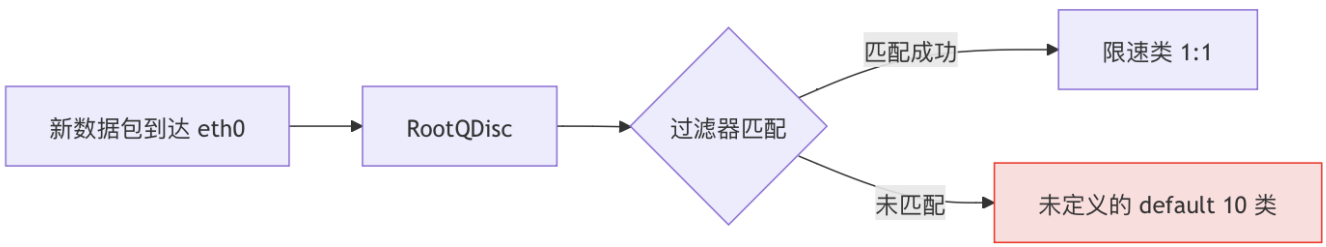
代码块

```
1   # 所有流量匹配classid 1:1
2   tc filter add dev eth0 protocol ip parent 1:0 prio 1 u32 match ip dst
    0.0.0.0/0 flowid 1:1
```

规则解析：

- `protocol ip`：匹配 IPv4 流量
- `u32 match ip dst 0.0.0.0/0`：匹配所有目标 IP（即全部出站流量）
- `flowid 1:1`：将匹配流量导向限速类 1:1

图解：



这样就能达到对所有出站流量进行限速的目的

2. Egress Pod的配置

在egress pod的网络空间中执行

代码块

```
1   # 10.200.0.165是gateway 的ip
```

```
ip link add vxlan0 type vxlan id 1000 dstport 4789 local 10.200.0.142 remote 10.200.0.165 dev eth0
ip addr add 172.16.0.101/24 dev vxlan0
ip link set vxlan0 up
# 创建自定义路由表（实际是在宿主机上）
echo "200 vxlan_route" >> /etc/iproute2/rt_tables
iptables -t mangle -A OUTPUT \
  -m iprange ! --dst-range 10.0.0.0-10.255.255.255 \
  -m iprange ! --dst-range 172.16.0.0-172.31.255.255 \
  -m iprange ! --dst-range 192.168.0.0-192.168.255.255 \
  -j MARK --set-mark 10
# 通过标记匹配路由规则
ip rule add fwmark 10 lookup 200
# 自定义路由表，其中172.16.0.102是对端vxlan(egressgateway的vxlan)ip
ip route add default via 172.16.0.102 dev vxlan0 table 200
# 要做snat
iptables -t nat -A POSTROUTING -o vxlan0 -j MASQUERADE
# 自己pod和egressgateway pod的ip
ip route add 10.200.0.142 dev eth0 table 200
ip route add 10.200.0.165 dev eth0 table 200
# 对于dind
iptables -t mangle -A PREROUTING -m iprange ! --dst-range 10.0.0.0-10.255.255.255 -m iprange ! --dst-range 172.16.0.0-172.31.255.255 -m iprange ! --dst-range 192.168.0.0-192.168.255.255 -j MARK --set-xmark 0xa/0xffffffff
iptables -A POSTROUTING -t mangle -o vxlan0 -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --set-mss 1410
```

现在对这些配置进行原理解析：

代码块

```
ip link add vxlan0 type vxlan id 1000 dstport 4789 local 10.200.0.142 remote 10.200.0.165 dev eth0
ip addr add 172.16.0.101/24 dev vxlan0
ip link set vxlan0 up
```

这里是对vxlan进行配置，其中10.200.0.142是egress pod的ip地址，10.200.0.165是egress gateway的ip地址，然后给vxlan添加ip，并启动vxlan

代码块

```
echo "200 vxlan_route" >> /etc/iproute2/rt_tables
iptables -t mangle -A OUTPUT \
  -m iprange ! --dst-range 10.0.0.0-10.255.255.255 \
  -m iprange ! --dst-range 172.16.0.0-172.31.255.255 \
  -m iprange ! --dst-range 192.168.0.0-192.168.255.255 \
```

```
    -j MARK --set-mark 10
ip rule add fwmark 10 lookup 200
ip route add default via 172.16.0.102 dev vxlan0 table 200
```

这里是通过自定义路由表（这个路由表最终会写在宿主机的/etc/iproute2/rt_tables文件中），对于所有的非私网的流量都将进行标记，并且所有被标记为10的流量根据ip rule都将查询自定义路由表（vxlan_route，这部分标记的rule可以在对应网络命名空间中通过**ip rule list**命令查看），并且vxlan_route(序号为200的表)的默认路由规则是对于所有匹配该表的流量都将经过对端vxlan(即egressgateway的vxlan，ip 172.16.0.102)。

这样达到的效果是，对于所有的公网流量，都将经过自定义路由表走自身的vxlan并走到egress gateway的vxlan上。

代码块

```
iptables -t nat -A POSTROUTING -o vxlan0 -j MASQUERADE
```

要做snat的原因是因为，其实最终回到pod来，相当于是eth0->vxlan0->eth0(发出)->gateway eth0->gateway vxlan0。最终还是要回到eth0来，所以这里要snat。

代码块

```
ip route add 10.200.0.142 dev eth0 table 200
ip route add 10.200.0.165 dev eth0 table 200
```

vxlan底层的UDP包在传输时也是走的table 200。 如果去掉这两行，UDP包无法正确路由

代码块

```
iptables -t mangle -A PREROUTING -m iprange ! --dst-range 10.0.0.0-10.255.255.255 -m iprange ! --dst-range 172.16.0.0-172.31.255.255 -m iprange ! --dst-range 192.168.0.0-192.168.255.255 -j MARK --set-xmark 0xa/0xffffffff
iptables -A POSTROUTING -t mangle -o vxlan0 -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --set-mss 1410
```

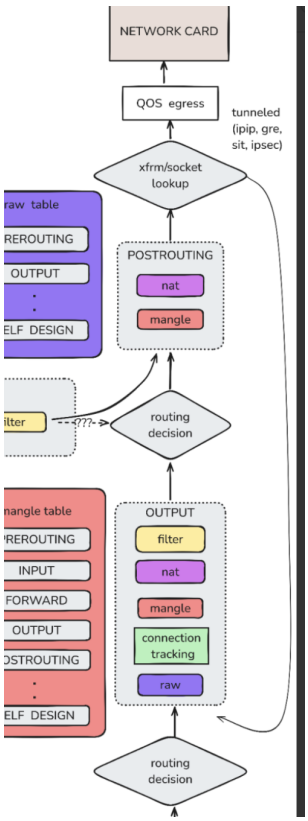
第一项配置是让所有接口在PREROUTING阶段，都把公网流量标记上mark10。 2. 第二项配置是将使用vxlan发送的tcp sync包中的mss强制设置为1410

3. 疑点与解释

疑点主要是下面几个：

1. 之前直接在命名空间内部ip route add时，是不需要对vxlan snat的。为什么这里就需要了呢?
   a. 尚不清楚原因。这里的变量在于，这里采用流量标记（fwmark 10）+自定义路由表（table 200)（均在Pod自己的网络命名空间中)，

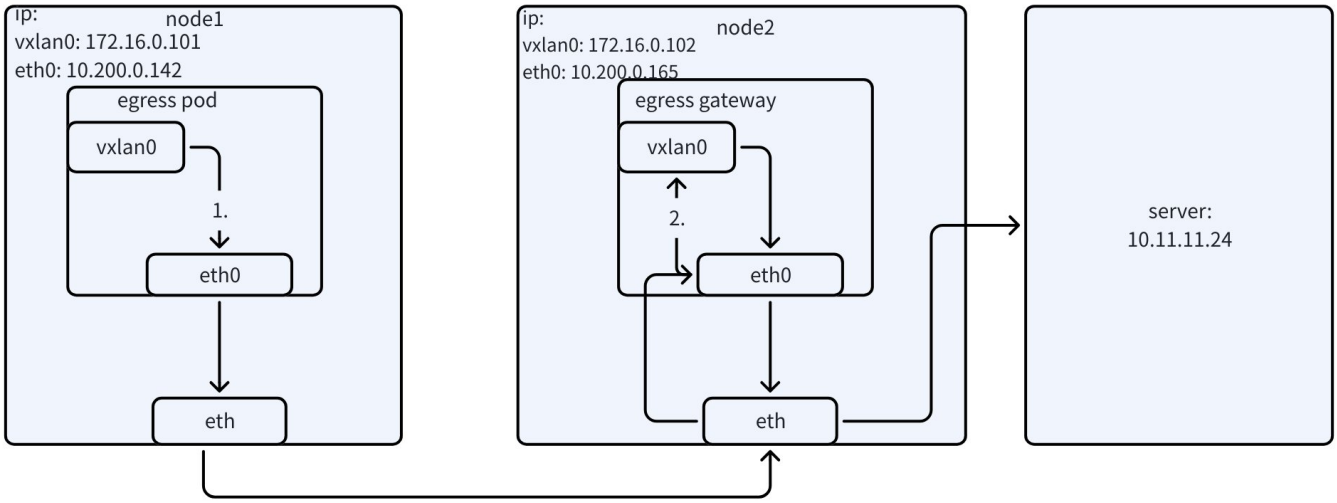
```
iptables -t mangle -A OUTPUT \
  -m iprange ! --dst-range 10.0.0.0-10.255.255.255 \
  -m iprange ! --dst-range 172.16.0.0-172.31.255.255 \
  -m iprange ! --dst-range 192.168.0.0-192.168.255.255 \
  -j MARK --set-mark 10
# 通过标记匹配路由规则
ip rule add fwmark 10 lookup 200
```



   这里有个怀疑是，snat是在POSTROUTING的nat表上做的，mark在OUTPUT的mangle表上做的。这里经过了两次routing decision。

2. 有必要管理自定义路由表吗?
   a. 没必要，都设为200就行，因为只对自己所在的网络命名空间有效
3. 为什么在egress pod中要加对两个pod ip的路由?
   a. 因为整个流量已经被标记了，vxlan底层的UDP包在传输时也是走的table 200。 如果去掉这两行，UDP包无法正确路由

4. 图解



抓包：

1. Egress pod抓包情况：
   a. vxlan0:
      i. 172.16.0.101<->10.11.11.24
   b. eth0:
      i. 10.200.0.142<->10.200.0.165
      ii. 172.16.0.101<->10.11.11.24
2. Egress gateway抓包情况：
   a. vxlan0:
      i. 172.16.0.101<->10.11.11.24
   b. eth0:
      i. 10.200.0.142<->10.200.0.165
      ii. 172.16.0.101<->10.11.11.24

Ip rule、ip route与链之间的交互顺序和关系（这里可以作为egress gateway配置的参照）：

代码块

```
1.  物理网卡接收数据包
    |
    ↓
```

```
2. Netfilter PREROUTING 链（raw、mangle、nat 表，其中mangle跟流量标记相关）
   │
   ↓
3. 路由决策（根据 `ip rule` 选择路由表，通过 `ip route` 确定下一跳）
   ├─ 目标地址是本机 → 进入 INPUT 链
   └─ 目标地址需转发 → 进入 FORWARD 链
       │
       ↓
4. Netfilter INPUT/FORWARD 链（filter 表）
   │
   ↓
5. 本地进程处理（如应用程序）
   │
   ↓
6. 本地进程发送响应 → 进入 OUTPUT 链（filter、mangle、nat 表，其中mangle跟流量标记相关）
   │
   ↓
7. 路由决策（根据 `ip rule` 选择路由表，确定出接口）
   │
   ↓
8. Netfilter POSTROUTING 链（mangle、nat 表）
   │
   ↓
9. 数据包从网卡发出
```

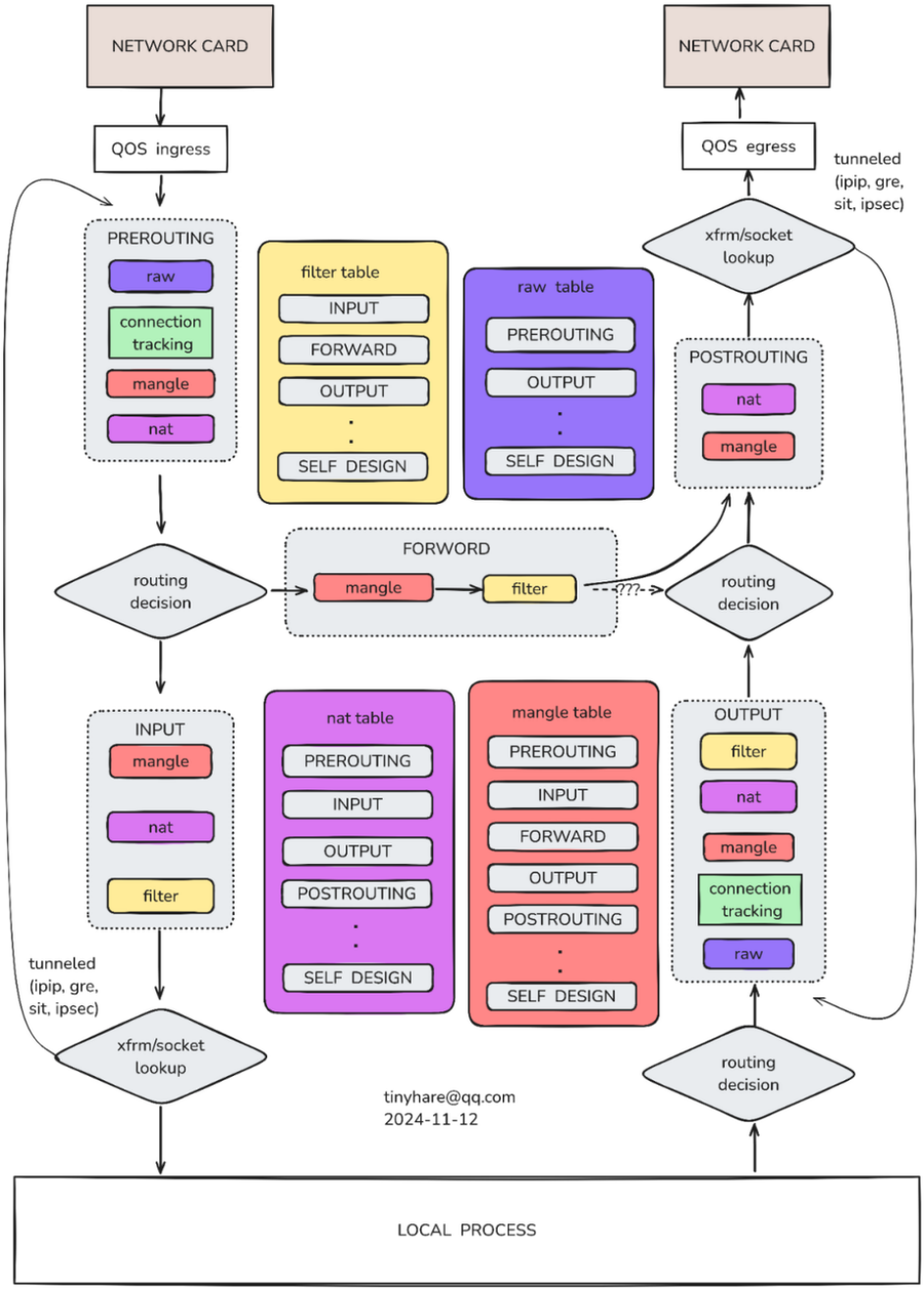
INPUT 链——进来的数据包应用此规则链中的策略

OUTPUT 链——外出的数据包应用此规则链中的策略

FORWARD 链——转发数据包时应用此规则链中的策略

PREROUTING 链——对数据包作路由选择前应用此链中的规则（所有的数据包进来的失手都先由这个链处理）

POSTROUTING 链——对数据包作路由选择后应用此链中的规则（所有的数据包出来的时候都先由这个链处理）



步骤：

在egress pod中：

1. 首先进行流量标记
2. 标记的流量根据ip rule走自定义路由表
3. 根据走自定义路由表，都是默认走OUTPUT->POSTROUTING链从vxlan0发出，这里对vxlan0做了snat，会触发nat表
4. 实际是从eth0发出到egress gateway

在egress gateway中：

1. 首先开启转发功能 sysctl -w net.ipv4.ip_forward=1
2. gateway接收到从egress pod发来的数据包(eth0->vxlan0)，通过查询路由表得知该向公网发送

3. 对eth0做了snat，这时走POSTROUTING链，触发nat表，从eth0发出到公网的数据包