LVS-TUN模式 - lyy962464的博客

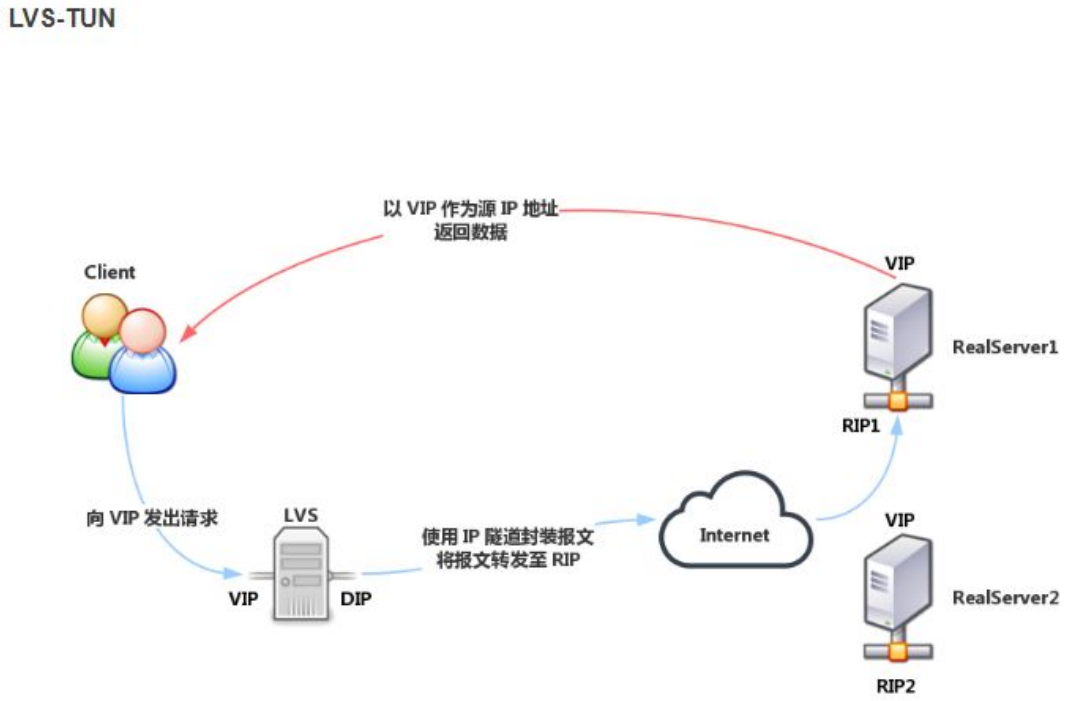
一、TUN简介

TUN 是IP Tunneling ，IP隧道的简称，它将调度器收到的IP数据包封装在一个新的IP数据包中，转交给应用服务器，然后实际服务器的返回数据会直接返回给用户。

ip隧道是一个将ip报文封装到另一个ip报文的技术，这可以使得目标为一个ip地址的数据报文被封装和转发到另一个ip地址。ip隧道技术也成为ip封装技术

它和NAT模式不同的是，它在LB和RS之间的传输不用改写IP地址（添 加新的IP头）。而是把客户请求包封装在一个IP tunnel里面，然后发送给RS节点服务器，节点服务器接收到之后解开IP tunnel后，进行响应处理。并且直接把包通过自己的外网地址发送给客户不用经过LB服务器。IP隧道技术主要用于移动主机和虚拟私有网络（Virtual Private Network），在其中隧道都是静态建立的，隧道一端有一IP地址，令一端也有唯一的ip地址。

二、TUN模式工作原理图



原理解析：

1.客户端将访问vip报文发送给LVS服务器；

2.LVS服务器将请求报文重新封装，发送给后端真实服务器；

3.后端真实服务器将请求报文解封，在确认自身有vip之后进行请求处理；

4.后端真实服务器在处理完数据请求后，直接响应客户端。

三、LVS-TUN模式下的负载均衡

实验环境：

Load Balance:172.25.88.1

Virtual IP:172.25.88.100

server2(RS):172.25.88.2

server(RS):172.25.88.3

在server1、server2、server3中添加隧道（由于虚拟服务器与RS是通过隧道进行包的交换的）

1、在server1上：

1）配置网络

modprobe ipip #添加隧道

ip link set up tunl0 #激活隧道

ip addr add 172.25.88.100 dev tunl0 #添加虚拟ip

ip addr #查看ip

```
[root@server1 ~]# modprobe ipip
[root@server1 ~]# ip link set up tunl0
[root@server1 ~]# ip addr add 172.25.88.100 dev tunl0
[root@server1 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP ql
en 1000
    link/ether 52:54:00:ea:bf:9c brd ff:ff:ff:ff:ff:ff
    inet 172.25.88.1/24 brd 172.25.88.255 scope global eth0
    inet 172.25.88.100/24 scope global secondary eth0
    inet6 fe80::5054:ff:feea:bf9c/64 scope link
       valid_lft forever preferred_lft forever
3: tunl0: <NOARP,UP,LOWER_UP> mtu 1480 qdisc noqueue state UNKNOWN
    link/ipip 0.0.0.0 brd 0.0.0.0
    inet 172.25.88.100/32 scope global tunl0
```


2）配置yun仓库

vim /etc/yum.repos.d/rhel-source.repo

[LoadBalancer]
name=LoadBalancer
baseurl=http://172.25.88.250/rhel6.5/LoadBalancer
gpgcheck=0

```
[root@server1 ~]# vim /etc/yum.repos.d/rhel-source.repo
```



```
[LoadBalancer]
name=LoadBalancer
baseurl=http://172.25.88.250/rhel6.5/LoadBalancer
gpgcheck=0
```


3、添加规则

yum install ipvsadm -y

```
[root@server1 ~]# yum install ipvsadm -y
Loaded plugins: product-id, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use su
bscription-manager to register.
LoadBalancer                                             | 3.9 kB     00:00
rhel-source                                              | 3.9 kB     00:00
Setting up Install Process
```


/etc/init.d/ipvsadm start #开启服务

ipvsadm -C #清除之前的策略

ipvsadm -A -t 172.25.88.100:80 -s rr #对后期服务器采用rr算法

ipvsadm -a -t 172.25.88.100:80 -r 172.25.88.2:80 -i #给vip添加rip，使用TUN模式

ipvsadm -a -t 172.25.88.100:80 -r 172.25.88.3:80 -i

/etc/init.d/ipvsadm save #保存策略

ipvsadm -ln #查看策略

ipvsadm -lnc #查看调度IP情况

```
[root@server1 ~]# /etc/init.d/ipvsadm start
ipvsadm: Saving IPVS table to /etc/sysconfig/ipvsadm:      [  OK  ]
ipvsadm: Clearing the current IPVS table:                  [  OK  ]
ipvsadm: Applying IPVS configuration:                      [  OK  ]
[root@server1 ~]# ipvsadm -C
[root@server1 ~]# ipvsadm -A -t 172.25.88.100:80 -s rr
[root@server1 ~]# ipvsadm -a -t 172.25.88.100:80 -r 172.25.88.2:80 -i
[root@server1 ~]# ipvsadm -a -t 172.25.88.100:80 -r 172.25.88.3:80 -i
[root@server1 ~]# /etc/init.d/ipvsadm save
ipvsadm: Saving IPVS table to /etc/sysconfig/ipvsadm:      [  OK  ]
[root@server1 ~]# ipvsadm -ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.25.88.100:80 rr
  -> 172.25.88.2:80               Tunnel  1      0          0
  -> 172.25.88.3:80               Tunnel  1      0          0
```


2、在server2上：

1）安装apache并开启

yum install httpd -y

```
[root@server2 ~]# yum install httpd -y
Loaded plugins: product-id, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use su
bscription-manager to register.
Setting up Install Process
```


vim /var/www/html/index.html
<h1>server2</h1>

```
[root@server2 ~]# vim /var/www/html/index.html
```

```
<h1>server2</h1>
```


/etc/init.d/httpd start

```
[root@server2 ~]# /etc/init.d/httpd start
Starting httpd: httpd: Could not reliably determine the server's fully qualified
 domain name, using 172.25.88.2 for ServerName
                                                           [  OK  ]
```


2）配置网络

modprobe ipip　　#加载模块

ip link set up tunl0　　#建立隧道设备tunl0

ip addr add 172.25.88.100/32 dev tunl0　　#添加虚拟IP

ip addr　#查看ip

```
[root@server2 ~]# modprobe ipip
[root@server2 ~]# ip link set up tunl0
[root@server2 ~]# ip addr add 172.25.88.100/32 dev tunl0
[root@server2 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP ql
en 1000
    link/ether 52:54:00:33:b1:75 brd ff:ff:ff:ff:ff:ff
    inet 172.25.88.2/24 brd 172.25.88.255 scope global eth0
    inet6 fe80::5054:ff:fe33:b175/64 scope link
       valid_lft forever preferred_lft forever
3: tunl0: <NOARP,UP,LOWER_UP> mtu 1480 qdisc noqueue state UNKNOWN
    link/ipip 0.0.0.0 brd 0.0.0.0
    inet 172.25.88.100/32 scope global tunl0
```


3)安装arptables工具

因为设置172.25.88.100/32作为vip，不可以和外部通信，所以设用arptables将其访问全部DROP，出去的包全部为转为本机的ip

yum install arptables_jf -y

```
[root@server2 ~]# yum install arptables_jf -y
Loaded plugins: product-id, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use su
bscription-manager to register.
Setting up Install Process
```


arptables -F #清空策略

arptables -A IN -d 172.25.88.100 -j DROP #拒绝172.25.88.100的访问

arptables -A OUT -s 172.25.88.100 -j mangle --mangle-ip-s 172.25.88.2 #由于tcp三次握手原因，所以出去的时候仍要以vip地址出去才会实现握手，而真正将数据传输给客户端的就是realserver， mangle参数就是这个功能

/etc/init.d/arptables_jf save #保存策略

arptables -L #查看添加进去的策略

```
[root@server2 ~]# arptables -F
[root@server2 ~]# arptables -A IN -d 172.25.88.100 -j DROP
[root@server2 ~]# arptables -A OUT -s 172.25.88.100 -j mangle --mangle-ip-s 172.
25.88.2
[root@server2 ~]# /etc/init.d/arptables_jf save
Saving current rules to /etc/sysconfig/arptables:          [  OK  ]
[root@server2 ~]# arptables -L
Chain IN (policy ACCEPT)
target     source-ip            destination-ip       source-hw          destinat
ion-hw     hlen   op         hrd        pro
DROP       anywhere               172.25.88.100        anywhere           anywhere
           any    any        any        any

Chain OUT (policy ACCEPT)
target     source-ip            destination-ip       source-hw          destinat
ion-hw     hlen   op         hrd        pro
mangle     172.25.88.100          anywhere             anywhere           anywhere
           any    any        any        any       --mangle-ip-s server2

Chain FORWARD (policy ACCEPT)
target     source-ip            destination-ip       source-hw          destinat
ion-hw     hlen   op         hrd        pro
```


4）关闭rp_filter

修改rp_filter参数

sysctl -w net.ipv4.conf.tunl0.rp_filter=0

注释：该参数用来控制系统是否开启对数据包源地址的校验。0标示不开启地址校验；1表开启严格的反向路径校验。对每一个进行的数据包，校验其反向路径是否是最佳路径。如果反向路径不是最佳路径，则直接丢弃该数据包；2标示开启松散的反向路径校验，对每个进行的数据包，校验其源地址是否可以到达，即反向路径是否可以ping通，如反向路径不通，则直接丢弃该数据包。

rp_filter参数的作用：

1. 减少DDoS攻击

校验数据包的反向路径，如果反向路径不合适，则直接丢弃数据包，避免过多的无效连接消耗系统资源。

2. 防止IP Spoofing

校验数据包的反向路径，如果客户端伪造的源IP地址对应的反向路径不在路由表中，或者反向路径不是最佳路径，则直接丢弃数据包，不会向伪造IP的客户端回复响应。

sysctl -a|grep .rp_filter　　#将过滤出的打开着的.rp_filter全部关闭

```
[root@server2 ~]# sysctl -a | grep .rp_filter
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.all.arp_filter = 0
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.default.arp_filter = 0
net.ipv4.conf.lo.rp_filter = 1
net.ipv4.conf.lo.arp_filter = 0
net.ipv4.conf.eth0.rp_filter = 1
net.ipv4.conf.eth0.arp_filter = 0
net.ipv4.conf.tunl0.rp_filter = 1
net.ipv4.conf.tunl0.arp_filter = 0
[root@server2 ~]# sysctl -w net.ipv4.conf.default.rp_filter = 0
error: "net.ipv4.conf.default.rp_filter" must be of the form name=value
error: Malformed setting "="
error: "0" must be of the form name=value
[root@server2 ~]# sysctl -w net.ipv4.conf.default.rp_filter=0
net.ipv4.conf.default.rp_filter = 0
[root@server2 ~]# sysctl -w net.ipv4.conf.lo.rp_filter=0
net.ipv4.conf.lo.rp_filter = 0
[root@server2 ~]# sysctl -w net.ipv4.conf.eth0.rp_filter=0
net.ipv4.conf.eth0.rp_filter = 0
[root@server2 ~]# sysctl -w net.ipv4.conf.tunl0.rp_filter=0
net.ipv4.conf.tunl0.rp_filter = 0
```


3、在server3中：

1）安装apache并开启服务

yum install httpd -y

```
[root@server3 ~]# yum install httpd -y
Loaded plugins: product-id, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use su
bscription-manager to register.
Setting up Install Process
```


vim /var/www/html/index.html
<h1>server3</h1>

```
[root@server3 ~]# vim /var/www/html/index.html
<h1>server3</h1>
```

/etc/init.d/httpd start　　#开启httpd

```
[root@server3 ~]# /etc/init.d/httpd start
Starting httpd: httpd: Could not reliably determine the server's fully qualified
 domain name, using 172.25.88.3 for ServerName
                                                           [  OK  ]
```


2)配置网络

modprobe ipip #加载模块

ip link set up tunl0 #激活隧道

ip addr add 172.25.88.100/32 dev tunl0 #添加虚拟IP

ip addr #查看ip

```
[root@server3 ~]# modprobe ipip
[root@server3 ~]# ip link set up tunl0
[root@server3 ~]# ip addr add 172.25.88.100/32 dev tunl0
[root@server3 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP ql
en 1000
    link/ether 52:54:00:4b:19:3c brd ff:ff:ff:ff:ff:ff
    inet 172.25.88.3/24 brd 172.25.88.255 scope global eth0
    inet6 fe80::5054:ff:fe4b:193c/64 scope link
       valid_lft forever preferred_lft forever
3: tunl0: <NOARP,UP,LOWER_UP> mtu 1480 qdisc noqueue state UNKNOWN
    link/ipip 0.0.0.0 brd 0.0.0.0
    inet 172.25.88.100/32 scope global tunl0
```


3）安装arptables_jf工具

yum install arptables_jf -y

```
[root@server3 ~]# yum install arptables_jf -y
Loaded plugins: product-id, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use su
bscription-manager to register.
Setting up Install Process
```


arptables -F

arptables -A IN -d 172.25.88.100 -j DROP

arptables -A OUT -s 172.25.88.100 -j mangle --mangle-ip-s 172.25.88.3

/etc/init.d/arptables_jf save

arptables -L

```
[root@server3 ~]# arptables -F
[root@server3 ~]# arptables -A IN -d 172.25.88.100 -j DROP
[root@server3 ~]# arptables -A OUT -s 172.25.88.100 -j mangle --mangle-ip-s 172.
25.88.3
[root@server3 ~]# /etc/init.d/arptables_jf save
Saving current rules to /etc/sysconfig/arptables:          [  OK  ]
[root@server3 ~]# arptables -L
Chain IN (policy ACCEPT)
target     source-ip            destination-ip       source-hw          destinat
ion-hw     hlen   op         hrd        pro
DROP       anywhere               172.25.88.100        anywhere           anywhere
           any    any        any        any

Chain OUT (policy ACCEPT)
target     source-ip            destination-ip       source-hw          destinat
ion-hw     hlen   op         hrd        pro
mangle     172.25.88.100          anywhere             anywhere           anywhere
           any    any        any        any       --mangle-ip-s server3

Chain FORWARD (policy ACCEPT)
target     source-ip            destination-ip       source-hw          destinat
ion-hw     hlen   op         hrd        pro
```


4）关闭rp_filter

sysctl -a|grep .rp_filter #将过滤出的打开着的.rp_filter全部关闭

```
[root@server3 ~]# sysctl -a | grep .rp_filter
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.all.arp_filter = 0
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.default.arp_filter = 0
net.ipv4.conf.lo.rp_filter = 1
net.ipv4.conf.lo.arp_filter = 0
net.ipv4.conf.eth0.rp_filter = 1
net.ipv4.conf.eth0.arp_filter = 0
net.ipv4.conf.tunl0.rp_filter = 1
net.ipv4.conf.tunl0.arp_filter = 0
[root@server3 ~]# sysctl -w net.ipv4.conf.default.rp_filter=0
net.ipv4.conf.default.rp_filter = 0
[root@server3 ~]# sysctl -w net.ipv4.conf.lo.rp_filter=0
net.ipv4.conf.lo.rp_filter = 0
[root@server3 ~]# sysctl -w net.ipv4.conf.eth0.rp_filter=0
net.ipv4.conf.eth0.rp_filter = 0
[root@server3 ~]# sysctl -w net.ipv4.conf.tunl0.rp_filter=0
net.ipv4.conf.tunl0.rp_filter = 0
```


4、测试：

在物理机中执行curl 172.25.88.100,出现轮询表示配置成功

会显示server2和server3中httpd共享文件的内容

```
[root@foundation88 ~]# curl 172.25.88.100
<h1>server3</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server2</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server3</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server2</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server3</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server2</h1>
```


四、RS处于不同网段下的TUN模式的负载平衡

实验环境：

继续使用之前的环境，将server3的ip更改为172.25.254.3

Load Balance:172.25.88.1 172.25.254.11
Virtual IP: 172.25.88.100
server2（RS）: 172.25.88.2
server3（RS）: 172.25.254.3

1、在server3中：

1）配置网络

ip addr del 172.25.88.3/24 dev eth0 #删除ip

ip addr add 172.25.254.3/24 dev eth0 #添加ip

ip addr #查看ip

```
[root@server3 ~]# ip addr del 172.25.88.3/24 dev eth0
[root@server3 ~]# ip addr add 172.25.254.3/24 dev eth0
[root@server3 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP ql
en 1000
    link/ether 52:54:00:4b:19:3c brd ff:ff:ff:ff:ff:ff
    inet 172.25.254.3/24 scope global eth0
    inet6 fe80::5054:ff:fe4b:193c/64 scope link
       valid_lft forever preferred_lft forever
3: tunl0: <NOARP,UP,LOWER_UP> mtu 1480 qdisc noqueue state UNKNOWN
    link/ipip 0.0.0.0 brd 0.0.0.0
    inet 172.25.88.100/32 scope global tunl0
```


2）配置arptables_jf

vim /etc/sysconfig/arptables

更改伪装策略，将172.25.88.3更改为172.25.254.3

/etc/init.d/arptables_jf restart

arptables -L

```
[root@server3 ~]# vim /etc/sysconfig/arptables
```

```
# Generated by arptables-save v0.0.8 on Fri Sep 28 17:17:20 2018
*filter
:IN ACCEPT [3:84]
:OUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
[0:0] -A IN -d 172.25.88.100 -j DROP
[0:0] -A OUT -s 172.25.88.100 -j mangle --mangle-ip-s 172.25.254.3
COMMIT
# Completed on Fri Sep 28 17:17:20 2018
```



```
[root@server3 ~]# /etc/init.d/arptables_jf restart
Flushing all current rules and user defined chains:        [  OK  ]
Clearing all current rules and user defined chains:        [  OK  ]
Applying arptables firewall rules:                         [  OK  ]
[root@server3 ~]# arptables -L
Chain IN (policy ACCEPT)
target     source-ip            destination-ip       source-hw          destinat
ion-hw     hlen   op         hrd        pro
DROP       anywhere               172.25.88.100       anywhere           anywhere
           any    any        any        any

Chain OUT (policy ACCEPT)
target     source-ip            destination-ip       source-hw          destinat
ion-hw     hlen   op         hrd        pro
mangle     172.25.88.100          anywhere            anywhere           anywhere
           any    any        any        any       --mangle-ip-s 172.25.254.3

Chain FORWARD (policy ACCEPT)
target     source-ip            destination-ip       source-hw          destinat
ion-hw     hlen   op         hrd        pro
```


3)配置网关

route add default gw 172.25.254.88　#将RS网关指向物理机的ip地址

route -n　　#查看网关

```
[root@server3 ~]# route add default gw 172.25.254.88
[root@server3 ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.25.254.0    0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         172.25.254.88   0.0.0.0         UG    0      0        0 eth0
```


2、在server1中：

1）添加网络

```
[root@server1 ~]# ip addr add 172.25.254.11/24 dev eth0
```

2）更改规则：

ipvsadm -ln

ipvsadm -d -t 172.25.88.100:80 -r 172.25.88.3:80

ipvsadm -a -t 172.25.88.100:80 -r 172.25.254.3:80 -i

ipvsadm -ln

```
[root@server1 ~]# ipvsadm -ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.25.88.100:80 rr
  -> 172.25.88.2:80               Tunnel  1      0          0
  -> 172.25.88.3:80               Tunnel  1      0          0
[root@server1 ~]# ipvsadm -d -t 172.25.88.100:80 -r 172.25.88.3:80
[root@server1 ~]# ipvsadm -a -t 172.25.88.100:80 -r 172.25.254.3:80 -i
[root@server1 ~]# ipvsadm -ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.25.88.100:80 rr
  -> 172.25.88.2:80               Tunnel  1      0          0
  -> 172.25.254.3:80              Tunnel  1      0          0
```


3、测试：

在真机中执行curl 172.25.88.100,出现轮询表示配置成功

```
[root@foundation88 ~]# curl 172.25.88.100
<h1>server3</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server2</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server3</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server2</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server3</h1>
[root@foundation88 ~]# curl 172.25.88.100
<h1>server2</h1>
```