My LVS Cluster 总结[原]

Mail:coldcoffee_gz@163.com
QQ：44657219

LVS-NAT

是通过改变数据包中的目的 IP 地址，来实现调度的。

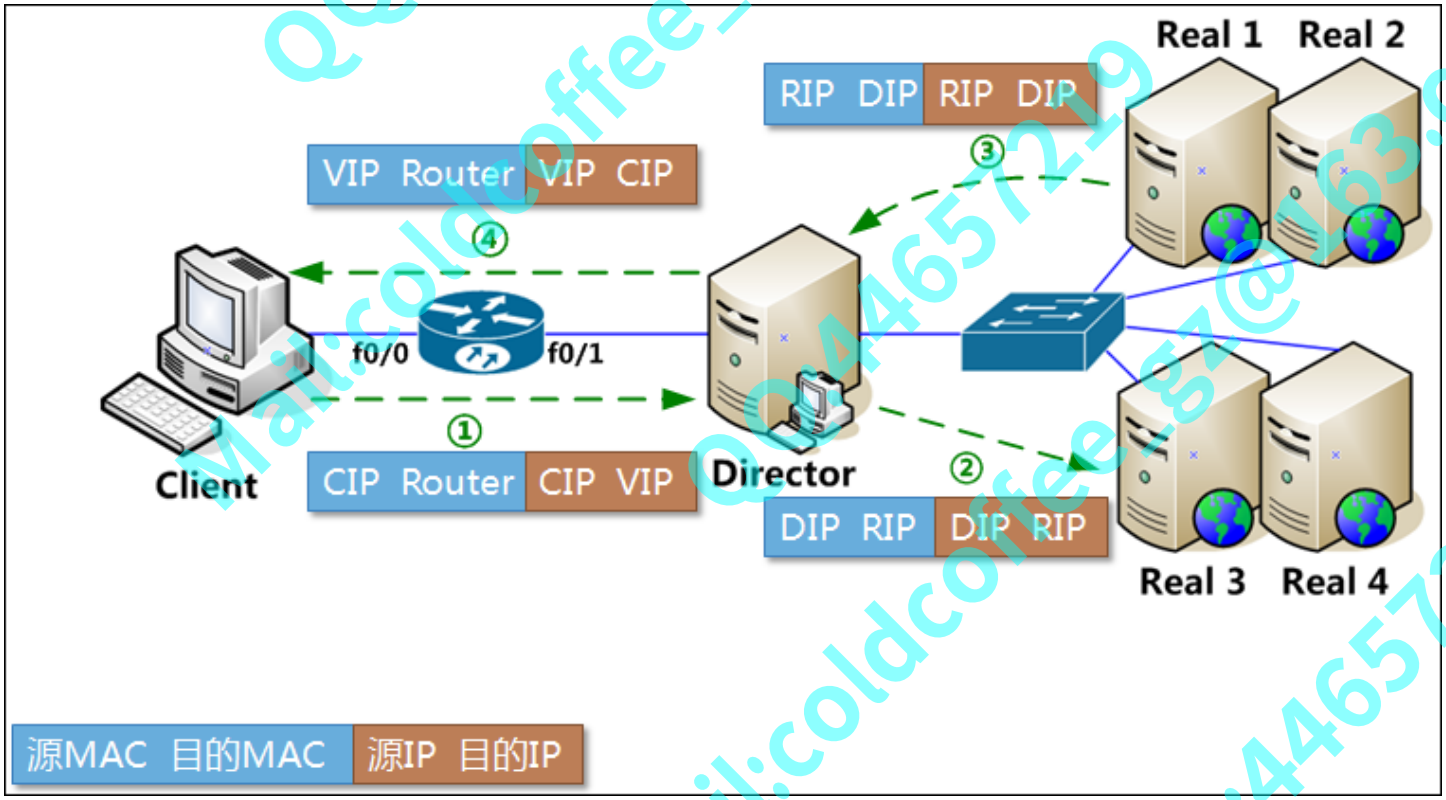
原理：

在一组服务器前有一个 LVS 主机，用户通过 Virtual IP Address （即 LVS 主机的外部地址）链接服务时，请求到达 LVS 主机， LVS 主机以负载平衡方法从一组后端服务器 **(real server)** 选出一台，将目标地址 **Virtual IP Address** 重新指向内部实际提供服务之服务器的地址。同时， LVS 主机记录这个连接，当这个连接的后端服务器的响应经过 LVS 主机时， LVS 主机将来源地址和来源 port 改为和 **Virtual IP Address** 相应的 port ，再把回应送给客户端。当连接终止或逾时， LVS 主机将这个连接从纪录中删除。这样，用户所看到的只是在 Virtual IP Address 上提供的服务。

优点：VS/NAT 的优点是服务器可以运行任何支持 TCP/IP 的操作系统，它只需要一个 IP 地址配置在 LVS 主机上，服务器组可以用私有的 IP 地址。Director 与 Real 在地理位置上不必在一起。

缺点： 它的扩充能力有限，当服务器结点数目升到 20 时， LVS 主机本身有可能成为系统的瓶颈，因为在 VS/NAT 中请求和响应封包都需要通过负载 LVS 主机，每一个封包都必须重写，用这种方法最大的数据吞吐量受限于 LVS 的环境。

拓扑图：



说明：

Director 是调度服务，Real *N*为真实服务（这里用 4 台）。实验用 vmware 虚拟机来完成，为了保证物理上确实是隔离开的，需要用不同的 vmnet 网卡相连。Client 与 Director 的 eth0 用 vmnet1 相连，Director 的 eth1 与 Real *N*用 vmnet2 相连。

步骤：

一、安装软件

1、安装内核源代码（因为安装 ipvsadm 时需要）
mount /dev/cdrom /mnt
cd /mnt/Server
rpm -ivh kernel-devel-2.6.18-164.el5.x86_64.rpm
需要给源代码做个软链接，不然一会编译 ipvsadm 时会报错
ln -s /usr/src/kernels/2.6.18-164.el5-i686/ /usr/src/linux
2、安装 ipvsadm
到 http://www.linuxvirtualserver.org/software/index.html，下载与内核对应的 ipvsadm，我下载的是：ipvsadm-1.24.tar.gz
tar -xjf ipvsadm-1.24.tar.gz
cd ipvsadm-1.24
make && make install
安装完后运行一下 ipvsadm 命令，然后执行 lsmod | grep ip_vs 命令，若能显示如下，说明内核已经支持：

```
[root@localhost ~]# lsmod | grep ip_vs
ip_vs                 123552  0
ipv6                  216180  29 ip_vs
```

二、配置各主机的 IP

主机	IP	网关
client	192.168.1.100	f0/0
Router	f0/0: 192.168.1.1 f1/0: 192.168.2.1	
Director	eth0 192.168.2.2 (VIP) eth1 192.168.3.1 (DIP)	f0/1
Real 1	192.168.3.10	192.168.3.1 **(DIP)**
Real 2	192.168.3.20	192.168.3.1 **(DIP)**
Real 3	192.168.3.30	192.168.3.1 **(DIP)**
Real 4	192.168.3.40	192.168.3.1 **(DIP)**
Real *N*	192.168.3.*N*	192.168.3.1 **(DIP)**

由于是用 NAT 方式做负载均衡，所以 Real *N*都要指向 Director 作为网关。

三、配置 Real *N*的主页服务
给每台 Real 服务器安装 apache，并在每台服务器的/var/www/html/下建立 index.html 文件。为了看出效果，最好让每台服务器上的 index.html 文件不一样，例如在 Real 1 上的 index.html 内容如下：

```
<H1>
<p>I'm Real 1.</p>
<p>IP: 192.168.3.10</p>
</H1>
```

其它几台 Real 服务器的 index.html 内容，只需把上面红色字样的部分对应改一下就可以。然后记得开启 httpd 服务。

四、配置 Director
在 Director 上最好也建立个 apache 服务，写个与真实服务器不一样的 index.html，例如：

```
<H1>
<p>I'm Director.</p>
</H1>
```

建立这个 index.html 的目的是为验证集群效果。当没有配置集群时，访问 VIP，看到的是 Director 上面的 index.html，如果配置了集群，再访问 VIP，就能看到其它 Real 服务器上的 index.html。

1、开启转发
vi /etc/sysctl.conf

```
net.ipv4.ip_forward = 1
```

sysctl -p
此步非常重要!!

2、增加虚拟服务
ipvsadm -A -t 192.168.2.2:80 -s rr
增加一个指向 192.168.2.2:80 的 tcp 虚拟服务，用轮叫(rr)算法

3、增加真实服务器
ipvsadm -a -t 192.168.2.2:80 -r 192.168.3.10 -m
ipvsadm -a -t 192.168.2.2:80 -r 192.168.3.20 -m
ipvsadm -a -t 192.168.2.2:80 -r 192.168.3.30 -m
ipvsadm -a -t 192.168.2.2:80 -r 192.168.3.40 -m
以 NAT 的方式，增加指向各真实服务器

五、测试
在 client 上用浏览器打开 http://192.168.2.2 地址，这时能看到真实服务的主页内容。反复按 F5 刷新，能看到不同的真实服务器的内容。说明集群已经建立成功。

我们也可以到 Director 服务器上，通过执行 ipvsadm -L -n 来查看调度的状态，如下：

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.2.2:80 rr
  -> 192.168.3.40:80              Masq    1      0          8
  -> 192.168.3.30:80              Masq    1      0          8
  -> 192.168.3.20:80              Masq    1      0          8
  -> 192.168.3.10:80              Masq    1      0          8
```

可以看出，4 台 Real 服务器，被调用的频率是均等的，因为用了 rr 算法。
并且用的是 NAT 方式在实现调试。(Masq 代表 NAT 方式)

六、wrr 算法的应用
上面用的是轮叫(rr)算法，每个 Real 服务器被调用的机会都是均等的。假设 Real 1 和 Real 2 服务器的处理性能要远比 Real 3 和 Real 4 都强，用 rr 算法就不是很合理了。因为 rr 算法不会考虑权重(Weight)，也就是优先级。所以需要换成 wrr(加权轮叫)算法，此算法会考虑管理员设置的权重，权重高的 Real 服务器，会被优先选中，而且被选中的频率也会多一些。
由于是接着上面的实验继续研究，所以可以有 2 种作法：①删除以前的内容重新开始 ②替换以前的内容。如果要删除，可以用 ipvsadm -C 来清除所有配置，再按上面的步骤 2、3 做就可以了。下面讲讲替换的作法：
1、改变算法
ipvsadm -E -t 192.168.2.2:80 -s wrr

2、改变 Real 1 和 Real 2 的权重高一些
ipvsadm -e -t 192.168.2.2:80 -r 192.168.3.10 -m -w 5
ipvsadm -e -t 192.168.2.2:80 -r 192.168.3.20 -m -w 5
权重值范围从 0-65535 之间，默认值为 1，值越高，优先级就越高。如果值是 0 表示永远不被选中（在处理真实服务器故障或维护时很有用），如果值是 65535 表示永远只选中它。

改完后，在 client 的浏览器里按 F5 刷新，在 Director 上可以看到，Real 1 和 Real 2 被选中的频率是其它 Real 服务器的 5 倍。倍值可以根据你工作的环境来合理设置。

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.2.2:80 wrr
  -> 192.168.3.40:80              Masq    1      0          15
  -> 192.168.3.30:80              Masq    1      0          15
  -> 192.168.3.20:80              Masq    5      0          75
  -> 192.168.3.10:80              Masq    5      0          75
```

注意：
当用 rr 算法时，即便设置了权重值，也不会启作用。只有用到 wrr 算法时，权重值才会发挥作用。
可见 ipvsadm -a 命令加入真实条目的先后顺序，并不能决定 real 的优先级，而是靠权重来决定的。

七、lc 和 wlc 算法的应用
lc 是最少链接算法，此算法会检查哪台 Real 的链接请求最少，就优先选择它。所以当你的服务器硬件配置相同时，lc 算法是个不错的选择。(个人感觉如果链接的起始数量一样的时候，跟 rr 算法没有区别)
wlc 是加权最少链接算法，此算法跟 lc 算法类似，只是增加了权重考虑条件。能让管理员在指定的 Real 中，优先应用最少链接算法。跟 rr 与 wrr 之间的关系是一种感觉。我们来实验一下：

1、改变算法
```
ipvsadm -E -t 192.168.2.2:80 -s wlc
```

2、改变 Real 1 和 Real 2 的权重高一些
```
ipvsadm -e -t 192.168.2.2:80 -r 192.168.3.10 -m -w 5
ipvsadm -e -t 192.168.2.2:80 -r 192.168.3.20 -m -w 5
ipvsadm -e -t 192.168.2.2:80 -r 192.168.3.30 -m -w 1
ipvsadm -e -t 192.168.2.2:80 -r 192.168.3.40 -m -w 1
```

改完后，在 client 的浏览器里按 F5 刷新，在 Director 上可以看到，Real 1 和 Real 2 被选中的频率是其它 Real 服务器的 5 倍。倍值可以根据你工作的环境来合理设置。(好像跟 wrr 的效果差不多 -_-!)

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.2.2:80 wlc
  -> 192.168.3.40:80              Masq    1      0          25
  -> 192.168.3.30:80              Masq    1      0          25
  -> 192.168.3.20:80              Masq    5      0          125
  -> 192.168.3.10:80              Masq    5      0          125
```

注意：
lvs 默认的算法是 wlc。
当用 lc 算法时，即便设置了权重值，也不会启作用。只有用到 wlc 算法时，权重值才会发挥作用。

NAT 方式小结：
- 原理比较好理解，配置相对简单些。
- 如果 Real 到 20 台以上时，Director 将会是瓶颈。

LVS-DR
是通过改变数据包中的目的 MAC 地址，来实现调度的。

原理：

在 VS/DR 中，LVS 主机根据各个后端服务器的负载情况，动态地选择一台服务器，不修改也不封装 IP 封包，而是将数据封包的 MAC 地址改为选出后端服务器的 MAC 地址，再将修改后的数据封包在与后端服务器组成的局域网络链接，因为数据封包的 MAC 地址是选出的后端服务器，所以后端服务器肯定可以收到该封包，发现 VIP 地址被配置在本地的网络设备上，所以就处理这个请求，然后根据路由表将响应封包直接返回给客户端。

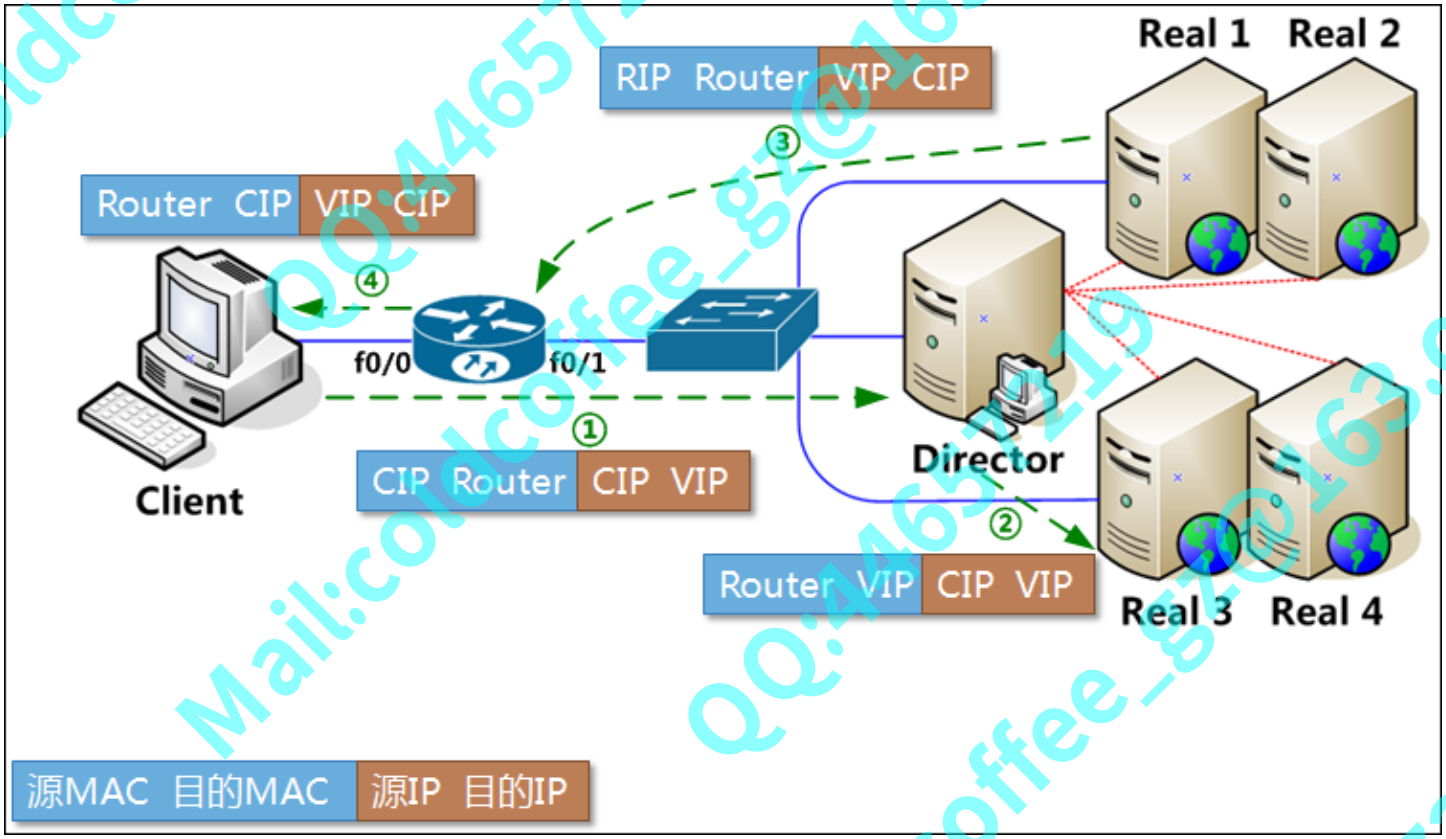
优点： VS/DR LVS 主机只处理客户到服务器端的连接，响应数据可以直接从独立的网络路由返回给客户。这可以极大地提高 LVS 集群系统的弹性。与 VS/TUN 相比，这种方法没有 IP Tunneling Mode 的 Overhead 。

缺点： 要求负载 LVS 主机与后端服务器都有一块网卡连在同一实体网段上。

后端服务器网络设备或设备别名不作 ARP 响应（arp-response），所以服务器需要额外的设定。

架设限制：后端服务器和 LVS 必须在同一个网络区段上（它们之间必须能使用 arp 协议，地理位置要在一起），并于数据链路层上传递封包客户端必须通过 LVS 的 VIP Cluster 后端服务器必须能有至客户端的路由（即使客户端没有到后端服务器的路由）。因此从后端服务器返回到客户端的封包，将直接发送，无须通过 LVS（ 负载均衡器 ）转送。

拓扑图：



步骤：

由于这个实验，是接着上个实验改过来的，所以我们需把网络连线改变，IP 地址重新设定：

一、配置各主机的 IP：

主机	IP	网关
client	192.168.1.100	f0/0
Router	f0/0: 192.168.1.1 f0/1: 192.168.2.1	
Director	eth0:1 192.168.2.254/32 (VIP) eth0 192.168.2.100/24 (DIP)	f0/1
Real 1	lo:1 192.168.2.254/32 (VIP) eth0 192.168.2.10/24 (RIP)	192.168.2.1 **(f0/1)**
Real 2	lo:1 192.168.2.254/32 (VIP)	192.168.2.1 **(f0/1)**

	eth0 192.168.2.20/24 (RIP)	
Real 3	lo:1 192.168.2.254/32 (VIP) eth0 192.168.2.30/24 (RIP)	192.168.2.1 **(f0/1)**
Real 4	lo:1 192.168.2.254/32 (VIP) eth0 192.168.2.40/24 (RIP)	192.168.2.1 **(f0/1)**
Real *N*	lo:1 192.168.2.254/32 (VIP) eth0 192.168.2.*N*/24 (RIP)	192.168.2.1 **(f0/1)**

由于是用 DR 方式做负载均衡，所以 Real *N* 都要配置 VIP，并且指向 Router 作为网关。

二、配置 Real 服务器：
1、主页服务：
由于上个实验中已经配置好主页服务，所以沿用上次的主页配置就可以了。

2、配置 VIP 地址：
以第一台 Real 服务器为例，配置如下：
①配置 RIP:
ifconfig eth0 192.168.2.10/24
或是用:
ip addr add dev eth0 192.168.2.10/24
ip link set dev eth0 up

②配置 VIP:
ifconfig lo:1 192.168.2.254 netmask 255.255.255.255 broadcast 192.168.2.254
或是用:
ip addr add dev lo 192.168.2.254/32 brd 192.168.2.254
注意上面的子网掩码

③配置网关：
ip route add default via 192.168.2.1
其它 Real 服务器只是 eth0 的 IP 不同，后 2 步的作法相同。

如果 Real 是 windows 系统，你可以这样做：
（windows2003 和 windowsXP 设置差不多）
1、控制面板→添加硬件→选“是，我已经连接了此设备”点击下一步→在列表中选择添加新的硬件设备 - 选“安装我充从手动…”→接下来的列表中选择“Microsoft loopback adapter”
2、添加完成后在“网上邻居”右键，设置 Microsoft loopback adapter IP 地址为 VIP 地址，子网掩码要设置成 255.255.255.0 （不能像 linux 那样设置成 255.255.255.255）
3、然后“开始”->"运行"-> regedit.exe 寻找“VIP”地址，把相应的 SubnetMask 设置成 255.255.255.255 （有多处都要修改）

三、配置 Director 服务器：
1、配置 IP：
①配置 DIP:
ifconfig eth0 192.168.2.100/24
或是用:
ip addr add dev eth0 192.168.2.100/24
ip link set dev eth0 up

②配置 VIP:
ifconfig eth0:1 192.168.2.254 netmask 255.255.255.255 broadcast 192.168.2.254

或是用:
ip addr add dev eth0 192.168.2.254/32 brd 192.168.2.254
注意上面的子网掩码

③配置网关：
ip route add default via 192.168.2.1

2、配置 lvs：
ipvsadm -A -t 192.168.2.254:80 -s rr
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.10 -g
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.20 -g
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.30 -g
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.40 -g
注意：Director 上不必开启转发。

执行 ipvsadm -L -n 查看，可以看出是用 DR 方式在完成调试。

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward    Weight ActiveConn InActConn
TCP   192.168.2.254:80 rr
  -> 192.168.2.40:80              Route      1      0          0
  -> 192.168.2.30:80              Route      1      0          0
  -> 192.168.2.20:80              Route      1      0          0
  -> 192.168.2.10:80              Route      1      0          0
```

Route 代表 DR 方式。

四、测试
在 client 上用浏览器打开 http://192.168.2.254 地址，这时能看到真实服务的主页内容。反复按 F5 刷新，有可能看到的还是其中一台 Real 服务的主页内容。这是由于 ARP 的机制所造成的。
在 router 上通过 show arp 命令，能看到多个 IP 地址对应同一个 MAC，而这里 MAC 地址对应的那个 IP 地址，就是你在浏览器中显示出主页的那台真实服务器。

```
Router#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  192.168.1.100           9  0050.56c0.0001  ARPA   FastEthernet0/0
Internet  192.168.1.1             -  cc00.0430.0000  ARPA   FastEthernet0/0
Internet  192.168.2.1             -  cc00.0430.0010  ARPA   FastEthernet0/1
Internet  192.168.2.20            8  000c.2941.2136  ARPA   FastEthernet0/1
Internet  192.168.2.254           9  000c.2941.2136  ARPA   FastEthernet0/1
这里能看到 2 个 MAC 对应同一个 IP
```

由于 ARP 的对应关系，是通过广播学到的。当 router 在收到 ARP 广播时，会把 000c.2941.2136 这个 MAC 地址与 Director 的 IP 和所有的 Real 的 IP 地址绑定在一起。所以客户端在浏览网页时，没有通过 Director 主机，而是直接就转到 192.168.2.20 上去了。这也不能算是一种错误，因为 VIP 在 Director 和 Real 上都有，所以现在的路由器，也很迷茫。而事实上应该让 192.168.2.254 对应 Director 上的 eth0 的 MAC 地址才是正确的。我们让路由清醒一点，手动帮它绑定一下：

```
Router>en
Router#conf t
Router#clear arp-cache    <清除缓存>
Router(config)#arp 192.168.2.254 000c.29a0.56b4 arpa
```

这里的 000c.29a0.56b4 就是 Director 上的 eth0 的 MAC 地址。
在浏览器上按 F5 再刷新一下看看，就可以显示不同 Real 服务器上的主页了，同时也证实了 Director 工作很正常，并且实现了 DR 调度方式。

但在路由器上来解决 ARP 问题，并不是一个很好的选择，原因有以下 2 点：

- 路由器可能不在你的管理范围，所以你无权要求管理路由器的人员帮你增加 ARP 对应关系
- 即使路由器由你来管理，一旦路由器重启后，那么 ARP 的对应关系又会发生混乱

需要怎么解决呢？其实很简单，我们不让 Real 服务器发送 VIP 地址的 ARP 广播，只让 Director 来发送 VIP 地址的 ARP 广播，那么接收端的设备自然就不会混乱了。

只需在 Real 服务器上限制 ARP 广播，并且每台 Real 服务器都需要实现，Director 上不需要限制，因为 Director 上应该也必须发出 VIP 的 ARP 广播。
只需要改变 Linux 的内核设置就可以实现本机 ARP 广播。（Linux 好强大呀! ^_^）
临时的做法如下：

```
echo "1" > /proc/sys/net/ipv4/conf/eth0/arp_ignore          忽略 eth0 網卡 arp 回應
echo "2" > /proc/sys/net/ipv4/conf/eth0/arp_announce        關閉 eth0 網卡 arp 廣播
echo "1" > /proc/sys/net/ipv4/conf/all/arp_ignore           忽略所有網卡 arp 回應
echo "2" > /proc/sys/net/ipv4/conf/all/arp_announce         關閉所有網卡 arp 廣播
```

永久的做法如下：
vi /etc/sysctl.conf 加入：

```
net.ipv4.conf.eth0.arp_ignore = 1
net.ipv4.conf.eth0.arp_announce = 2
net.ipv4.conf.all.arp_ignore = 1
net.ipv4.conf.all.arp_announce = 2
```

保存退出，执行 sysctl -p

备注：
arp_ignore=1，系统只回答目的 IP 为是本地 IP 的包。（是对 ARP 广播不做响应）
arp_announce=2，系统忽略 IP 包的源地址，而根据目标主机，选择本地地址。（本身不发 ARP 广播）

---

arp_ignore : INTEGER 定义对目标地址为本地 IP 的 ARP 询问不同的应答模式，相关代码在 arp_announce 函数中 默认为 0
0 - (默认值): 回应任何网络接口上对任何本地 IP 地址的 arp 查询请求（比如 eth0=192.168.0.1/24,eth1=10.1.1.1/24,那么即使 eth0 收到来自 10.1.1.2 这样地址发起的对 10.1.1.1 的 arp 查询也会回应--而原本这个请求该是出现在 eth1 上，也该由 eth1 回应的）

1 - 只回答目标 IP 地址是来访网络接口本地地址的 ARP 查询请求（比如 eth0=192.168.0.1/24,eth1=10.1.1.1/24,那么即使 eth0 收到来自 10.1.1.2 这样地址发起的对 192.168.0.1 的查询会回答，而对 10.1.1.1 的 arp 查询不会回应）

2 -只回答目标 IP 地址是来访网络接口本地地址的 ARP 查询请求,且来访 IP 必须在该网络接口的子网段内（比如 eth0=192.168.0.1/24,eth1=10.1.1.1/24,eth1 收到来自 10.1.1.2 这样地址发起的对 192.168.0.1 的查询不会回答，而对 192.168.0.2 发起的对 192.168.0.1 的 arp 查询会回应）

3 - 不回应该网络界面的 arp 请求，而只对设置的唯一和连接地址做出回应(do not reply for local addresses configured with scope host,only resolutions for global and link addresses are replied)

arp_announce :INTEGER 不同取值表示对网络接口上本地 IP 地址发出的 ARP 回应作出相应级别的限制：相关代码在 默认为 0
确定不同程度的限制,宣布对来自本地源 IP 地址发出 Arp 请求的接口
0 - (默认) 在任意网络接口上的任何本地地址

1 -尽量避免不在该网络接口子网段的本地地址. 当发起 ARP 请求的源 IP 地址是被设置应该经由路由达到此网络接口的时候很有用.此时会检查来访 IP 是否为所有接口上的子网段内 ip 之一.如果该来访 IP 不属于各个网络接口上的

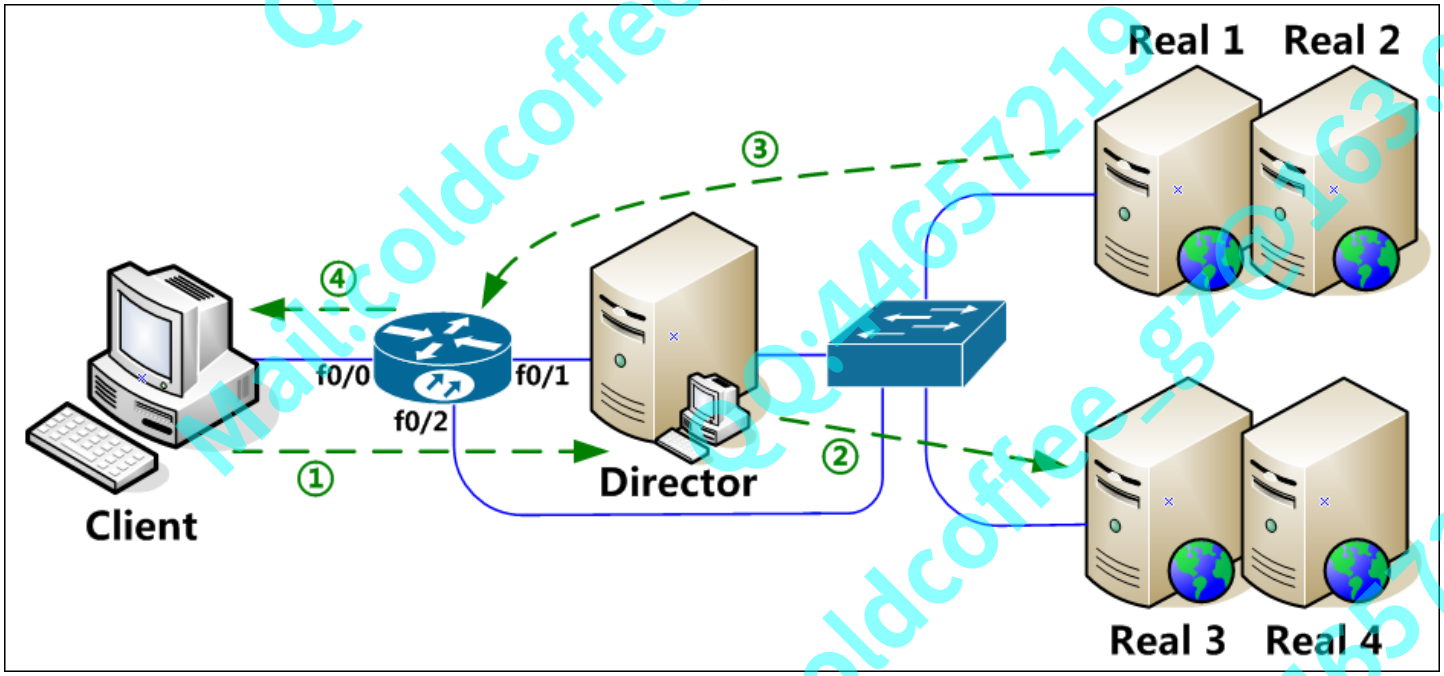
子网段内,那么将采用级别 2 的方式来进行处理.

2 - 对查询目标使用最适当的本地地址.在此模式下将忽略这个 IP 数据包的源地址并尝试选择与能与该地址通信的本地地址.首要是选择所有的网络接口的子网中外出 访问子网中包含该目标 IP 地址的本地地址. 如果没有合适的地址被发现,将选择当前的发送网络接口或其他的有可能接受到该 ARP 回应的网络接口来进行发送；
当内网的机器要发送一个到外部的 ip 包，那么它就会请求路由器的 Mac 地址，发送一个 arp 请求，这个 arp 请求里面包括了自己的 ip 地址和 Mac 地址， 而 linux 默认是使用 ip 的源 ip 地址作为 arp 里面的源 ip 地址，而不是使用发送设备上面的，如果设置 arp_announce 为 2，则使用发送设备上的 ip。

对每个 Real 服务器改变后，客户端的浏览器里可能还会只显示其中一台 Real 服务器的主页，原因是 router 还保存着旧的 ARP 对应关系，所以要清除一下 router 的缓存，执行如下命令：
Router#clear arp-cache
经过客户端几次访问后，你在 router 上绝对不会再发现，同一个 MAC 即对应 Director 又对应 Real。

```
Router#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  192.168.1.100           0  0050.56c0.0001  ARPA   FastEthernet0/0
Internet  192.168.2.10            0  000c.29f1.4746  ARPA   FastEthernet0/1
Internet  192.168.1.1             -  cc00.1668.0000  ARPA   FastEthernet0/0
Internet  192.168.2.1             -  cc00.1668.0010  ARPA   FastEthernet0/1
Internet  192.168.2.30            0  000c.29a5.703f  ARPA   FastEthernet0/1
Internet  192.168.2.20            0  000c.2941.2136  ARPA   FastEthernet0/1
Internet  192.168.2.254           5  000c.29a0.56b4  ARPA   FastEthernet0/1
```

LVS/DR 的另一种拓扑结构：



步骤：

一、配置各主机的 IP：

主机	IP	网关
client	192.168.1.100	f0/0
Router	f0/0: 192.168.1.1 f0/1: 192.168.2.1 f0/2: 10.0.0.1	
Director	eth0:1 192.168.2.254/32 (VIP) eth1 10.0.0.254/8 (DIP)	192.168.2.1 **(f0/1)**

Real 1	lo:1 192.168.2.254/32 (VIP) eth0 10.0.0.10/8 (RIP)	10.0.0.1 **(f0/2)**
Real 2	lo:1 192.168.2.254/32 (VIP) eth0 10.0.0.20/8 (RIP)	10.0.0.1 **(f0/2)**
Real 3	lo:1 192.168.2.254/32 (VIP) eth0 10.0.0.30/8 (RIP)	10.0.0.1 **(f0/2)**
Real 4	lo:1 192.168.2.254/32 (VIP) eth0 10.0.0.40/8 (RIP)	10.0.0.1 **(f0/2)**
Real *N*	lo:1 192.168.2.254/32 (VIP) eth0 10.*N.N.N/8* (RIP)	10.0.0.1 **(f0/2)**

由于是用 DR 方式做负载均衡，所以 Real *N* 都要配置 VIP，并且指向 Router 作为网关。

二、配置 Real 服务器：
1、主页服务：
由于上个实验中已经配置好主页服务，所以沿用上次的主页配置就可以了。

2、配置 VIP 地址：
以第一台 Real 服务器为例，配置如下：
①配置 RIP:
ip addr add dev eth0 10.0.0.10/8
ip link set dev eth0 up

②配置 VIP:
ip addr add dev lo 192.168.2.254/32 brd 192.168.2.254
注意上面的子网掩码

③配置网关：
ip route add default via 192.168.2.1
其它 Real 服务器只是 eth0 的 IP 不同，①②步的作法相同。

3、关闭 ARP 响应：
vi /etc/sysctl.conf 加入：
net.ipv4.conf.eth0.arp_ignore = 1
net.ipv4.conf.eth0.arp_announce = 2
net.ipv4.conf.all.arp_ignore = 1
net.ipv4.conf.all.arp_announce = 2
保存退出，执行 sysctl -p

三、配置 Director 服务器：
1、配置 IP：
①配置 DIP:
ip addr add dev eth1 10.0.0.254/8
ip link set dev eth1 up

②配置 VIP:
ip addr add dev eth0 192.168.2.254/24
ip link set dev eth0 up

注意上面的子网掩码

③配置网关：
ip route add default via 192.168.2.1

2、配置 lvs：
ipvsadm -A -t 192.168.2.254:80 -s rr
ipvsadm -a -t 192.168.2.254:80 -r 10.0.0.10 -g
ipvsadm -a -t 192.168.2.254:80 -r 10.0.0.20 -g
ipvsadm -a -t 192.168.2.254:80 -r 10.0.0.30 -g
ipvsadm -a -t 192.168.2.254:80 -r 10.0.0.40 -g
注意：Director 上不必开启转发。

扩展部分：
在 Director 和 Real 上，也都可以通过脚本来实现，这样会方便一些，不用日后再一条条输入命令。
Director 上要执行的脚本内容：

```
#!/bin/bash
# description: start LVS of DirectorServer

GW=192.168.2.1
# website director vip.
VIP="192.168.2.254"
DIP="10.0.0.254/8"
RIP1=10.0.0.10
RIP2=10.0.0.20
RIP3=10.0.0.30
RIP4=10.0.0.40

    /etc/rc.d/init.d/functions

logger $0 called with $1

case "$1" in

start)
        /sbin/ipvsadm --set 30 5 60
        /sbin/ip addr add dev eth1 $DIP
        /sbin/ip link set dev eth1 up
        /sbin/ip addr add dev eth0 $VIP brd $VIP
        /sbin/ip link set dev eth0 up
        /sbin/ip route add default via 192.168.2.1

        /sbin/ipvsadm -A -t $VIP:80 -s wrr
        /sbin/ipvsadm -a -t $VIP:80 -r $RIP1:80 -g -w 1
        /sbin/ipvsadm -a -t $VIP:80 -r $RIP2:80 -g -w 1
        /sbin/ipvsadm -a -t $VIP:80 -r $RIP3:80 -g -w 2
        /sbin/ipvsadm -a -t $VIP:80 -r $RIP4:80 -g -w 2

        touch /var/lock/subsys/ipvsadm >/dev/null 2>&1
        /sbin/arping -I eth0 -c 5 -s $VIP $GW >/dev/null 2>&1
    ;;

stop)
        /sbin/ipvsadm -C
        /sbin/ipvsadm -Z
        /sbin/ip link set dev eth0 down
        /sbin/ip link set dev eth1 down
        rm -rf /var/lock/subsys/ipvsadm >/dev/null 2>&1
        echo "ipvsadm stoped"
        ;;

status)

        if [ ! -e /var/lock/subsys/ipvsadm ];then
            echo "ipvsadm stoped"
```

```
                exit 1
        else
                echo "ipvsadm OK"
        fi
        ;;

*)
        echo "Usage: $0 {start|stop|status}"
        exit 1
esac

exit 0
```

每台 Real 上都要执行的脚本内容：

```
#!/bin/bash
# description: Config realserver lo and apply noarp


VIP=192.168.2.254

/etc/rc.d/init.d/functions

case "$1" in
start)
        /sbin/ip addr add dev lo $VIP brd $VIP

        echo "1" >/proc/sys/net/ipv4/conf/eth0/arp_ignore
        echo "2" >/proc/sys/net/ipv4/conf/eth0/arp_announce
        echo "1" >/proc/sys/net/ipv4/conf/all/arp_ignore
        echo "2" >/proc/sys/net/ipv4/conf/all/arp_announce
        echo "RealServer Start OK"
        ;;

stop)
        /sbin/ip link set dev lo down
        echo "0" >/proc/sys/net/ipv4/conf/eth0/arp_ignore
        echo "0" >/proc/sys/net/ipv4/conf/eth0/arp_announce
        echo "0" >/proc/sys/net/ipv4/conf/all/arp_ignore
        echo "0" >/proc/sys/net/ipv4/conf/all/arp_announce
        echo "RealServer Stoped"
        ;;
*)
        echo "Usage: $0 {start|stop}"
        exit 1
esac

exit 0
```

四、测试

在 client 上用浏览器打开 http://192.168.2.254 地址，反复按 F5 刷新，可以看到不同的主页内容。

DR 小结：

- 配置比较复杂，原理不太好理解。
- 工作环境里用的最多。
- 在 DR 调试模式下，最好在搭建完 Real 上的服务后，就把 ARP 广播限制住。
- 当 DR 模式被每一次访问时会慢一些，但马上就会恢复正常。我想可能是 Director 在改变 MAC 地址或是 router 在改变自己的 ARP 缓存。

LVS-TUN

是以通道的方式，来实现调度的。

原理：

VS/TUN 的连接分配和管理与 VS/NAT 中的一样，只是它的封包转送方法不同。 LVS 主机根据各个后端服务器的负载情况，动态地选择一台服务器，将请求封包封装在另一个 IP 封包中，再将封装后的 IP 封包转送给选出的后端服务器。

后端服务器收到封包后，先将封包解开获得原来目标地址为 VIP 的封包，后端服务器发现 VIP 地址被配置在本地的 IP 隧道设备上，所以就处理这个请求，然后依据路由表将响应封包直接返回给客户端。

优点： 不需要服务器与客户端位于同一个网络上，仅需要客户端能寻径（有路由）到负载均衡器，后端服务器能寻径（有路由）到客户端。（返回的封包直接从后端服务器到客户端，无须再经过 LVS ）通常使用 VS-Tun 模式时，客户端是与 LVS 、后端服务器位于不同网络上的，而且每一台服务器有一个通往外界的路由。负载 LVS 主机只将客户端的 request 分配到不同的后端服务器，后端服务器将 data 直接返回给客户端。负载 LVS 主机就可以处理大量的请求，而不会成为系统的瓶颈。

缺点： Real servers 必须支持 IP tunneling protocol

拓扑图：

与 LVS-DR 拓扑结构一模一样，只是在实现的时候，需要把 lo 网卡换成 tunl 网卡。

步骤：

我们以 LVS-DR 的第一种拓扑环境，来实现 LVS-TUN

一、配置 IP 地址：

主机	IP	网关
client	192.168.1.100	f0/0
Router	f0/0: 192.168.1.1 f0/1: 192.168.2.1	
Director	eth0:1 192.168.2.254/32 (VIP) eth0 192.168.2.100/24 (DIP)	f0/1
Real 1	tunl0 192.168.2.254/32 (VIP) eth0 192.168.2.10/24 (RIP)	192.168.2.1 **(f0/1)**
Real 2	tunl0 192.168.2.254/32 (VIP) eth0 192.168.2.20/24 (RIP)	192.168.2.1 **(f0/1)**
Real 3	tunl0 192.168.2.254/32 (VIP) eth0 192.168.2.30/24 (RIP)	192.168.2.1 **(f0/1)**
Real 4	tunl0 192.168.2.254/32 (VIP) eth0 192.168.2.40/24 (RIP)	192.168.2.1 **(f0/1)**
Real *N*	tunl0 192.168.2.254/32 (VIP) eth0 192.168.2.*N*/24 (RIP)	192.168.2.1 **(f0/1)**

由于是用 TUN 方式做负载均衡，所以 Real *N* 都要配置 VIP，并且指向 Router 作为网关。

二、配置 Real 服务器：

1、主页服务：

由于上个实验中已经配置好主页服务，所以沿用上次的主页配置就可以了。

2、配置 VIP 地址：

以第一台 Real 服务器为例，配置如下：

①配置 RIP:

```
ip addr add dev eth0 192.168.2.10/24
ip link set dev eth0 up
```

②配置 VIP:
ip addr add dev tunl0 192.168.2.254/32 brd 192.168.2.254
ip link set dev tunl0 up
注意上面的子网掩码

③配置网关：
ip route add default via 192.168.2.1

其它 Real 服务器只是 eth0 的 IP 不同，①②步的作法相同。

3、关闭 ARP 响应：
vi /etc/sysctl.conf 加入：
net.ipv4.conf.eth0.arp_ignore = 1
net.ipv4.conf.eth0.arp_announce = 2
net.ipv4.conf.all.arp_ignore = 1
net.ipv4.conf.all.arp_announce = 2
保存退出，执行 sysctl -p

三、配置 Director 服务器：
1、配置 IP：
①配置 DIP:
ip addr add dev eth0 192.168.2.100/24
ip link set dev eth0 up

②配置 VIP:
ip addr add dev eth0:1 192.168.2.254/32 brd 192.168.2.254
Director 端的 VIP 不用 tunl 设备，用得还是 eth0:1 设备

③配置网关：
ip route add default via 192.168.2.1

2、配置 lvs：
ipvsadm -A -t 192.168.2.254:80 -s rr
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.10 -i
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.20 -i
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.30 -i
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.40 -i
注意：Director 上不必开启转发。

执行 ipvsadm -L -n 查看，可以看出是用 TUN 方式在完成调试。

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.2.254:80 rr
  -> 192.168.2.40:80              Tunnel  1      0          0
  -> 192.168.2.30:80              Tunnel  1      0          0
  -> 192.168.2.20:80              Tunnel  1      0          0
  -> 192.168.2.10:80              Tunnel  1      0          0
```

Tunnel 代表 TUN 方式。

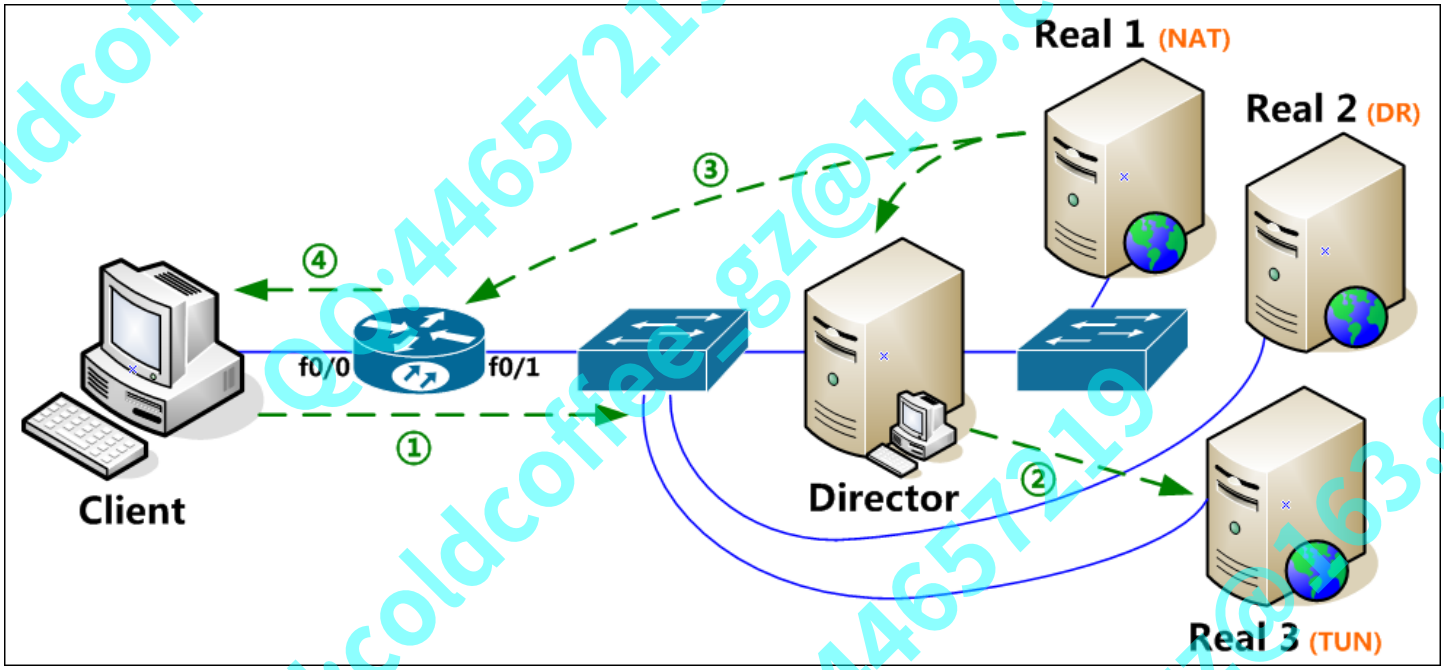
四、在 client 的浏览器里按 F5 刷新，可以看到不同 Real 上的主页内容。

TUN 小结：

- 目前感觉与 LVS-DR 差不太多
- 需要隧道的开销，集群中的真实服务器必须支持 IP 隧道协议的要求
- 在 TUN 调试模式下，最好在搭建完 Real 上的服务后，就把 ARP 广播限制住。

NAT+DR+TUN 综合实施：

拓扑图：



一、配置 IP 地址：

主机	IP	网关
client	192.168.1.100	f0/0
Router	f0/0: 192.168.1.1 f0/1: 192.168.2.1	
Director	eth0:1 192.168.2.254/32 (VIP) eth0 192.168.2.100/24 (DIP) eth1 192.168.3.1/24 (DIP)	f0/1
Real 1	eth0 192.168.3.10/24 (RIP)	192.168.3.1 **(DIP)**
Real 2	lo:1 192.168.2.254/32 (VIP) eth0 192.168.2.20/24 (RIP)	192.168.2.1 **(f0/1)**
Real 3	tunl0 192.168.2.254/32 (VIP) eth0 192.168.2.30/24 (RIP)	192.168.2.1 **(f0/1)**

二、配置 Real 服务器：

3 台 Real 服务要以不同方式，做出不出的配置，具体如下：

Real 1 上的配置：
① 置 RIP:
ip addr add dev eth0
192.168.3.10/24
ip link set dev eth0 up

②配置网关：
ip route add default via
192.168.3.1

Real 2 上的配置：
①配置 RIP:
ip addr add dev eth0
192.168.2.20/24
ip link set dev eth0 up

②配置 VIP:
ip addr add dev lo
192.168.2.254/32 brd
192.168.2.254
ip link set dev lo up

③配置网关：
ip route add default via
192.168.2.1

④关闭 ARP 响应：
vi /etc/sysctl.conf 加入：
net.ipv4.conf.eth0.arp_ignore
= 1
net.ipv4.conf.eth0.arp_annou
nce = 2
net.ipv4.conf.all.arp_ignore =
1
net.ipv4.conf.all.arp_announc
e = 2
保存退出，执行 sysctl -p

Real 3 上的配置：
①配置 RIP:
ip addr add dev eth0
192.168.2.30/24
ip link set dev eth0 up

②配置 VIP:
ip addr add dev tunl0
192.168.2.254/32 brd
192.168.2.254
ip link set dev tunl0 up

③配置网关：
ip route add default via
192.168.2.1

④关闭 ARP 响应：
vi /etc/sysctl.conf 加入：
net.ipv4.conf.eth0.arp_ignore
= 1
net.ipv4.conf.eth0.arp_annou
nce = 2
net.ipv4.conf.all.arp_ignore =
1
net.ipv4.conf.all.arp_announc
e = 2
保存退出，执行 sysctl -p

三、配置 Director 服务器：
1、配置 IP：
①配置 DIP:
ip addr add dev eth0 192.168.2.100/24
ip link set dev eth0 up
ip addr add dev eth1 192.168.3.1/24
ip link set dev eth0 up

②配置 VIP:
ip addr add dev eth0:1 192.168.2.254/32 brd 192.168.2.254

③配置网关：
ip route add default via 192.168.2.1

2、开启转发，因为要用到 Real1 要用到 NAT 调度
echo "1" > /proc/sys/net/ipv4/ip_forward

3、配置 lvs：
ipvsadm -A -t 192.168.2.254:80 -s rr
ipvsadm -a -t 192.168.2.254:80 -r 192.168.3.10 -m
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.20 -g
ipvsadm -a -t 192.168.2.254:80 -r 192.168.2.30 -i

执行 ipvsadm -L -n 查看，可以看出是用 TUN 方式在完成调试。

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.2.254:80 rr
  -> 192.168.2.30:80              Tunnel  1      0          0
  -> 192.168.2.20:80              Route   1      0          0
  -> 192.168.3.10:80              Masq    1      0          0
```

四、在 client 的浏览器里按 F5 刷新，可以看到不同 Real 上的主页内容。

补充问题：

LVM 工作在网络哪层

这个问题应该区分对待:

1) 当 LVS 工作在 NAT 模式下,是工作在 OSI 模型的第四层,因为 Director 必须修改第四层报头信息,包括 IP 地址和端口,在选择 RS 的时候不但要看 IP 地址,还要看端口,也即在 NAT 模式下 LVS 支持基于同一 IP 不同端口的第四层负载均衡.

2) 在 DR 模式下,Director 根本都不用拆包到网络层,因为 Director 根本就不用将 IP 包拆开,直接在链路层将目的 MAC 地址改成 RealServer 的 MAC 地址就完成了转发工作.当然,转发规则里是以 IP 的形式呈现,但是获得目的 IP 的 MAC 地址则是通过 arp 协议来完成,仍旧是链路层,所以确切地说 DR 模式在包处理上工作在链路层,而在 RealServer 的选择上(也就是负载均衡)是根据转发规则里的 IP 地址和调度算法,而虚拟服务端口必须与 RealServer 服务端口一致,所以 LVS/DR 模式单独讲负载均衡功能而言的话,它是工作在网络(IP)层.

3) TUN 模式主要是 IP 封装,根据 IP 地址进行选择 RealServer,也是网络层负载均衡.

LVS 性能调优

可以通过”ipvsadm -set tcp tcpfin udp”来调整 TCP 和 UDP 的超时，让连接淘汰得快一些。

```
ipvsadm -Ln --timeout
Timeout (tcp tcpfin udp): 900 120 300
ipvsadm --set tcp tcpfin udp
```

```
Improving TCP/IP performance
net.ipv4.tcp_tw_recyle=1
net.ipv4.tcp_tw_reuse=1
net.ipv4.tcp_max_syn_backlog=8192
net.ipv4.tcp_keepalive_time=1800
net.ipv4.tcp_fin_timeout=30
net.core.rmem_max=16777216
net.core.wmem_max=16777216
net.ipv4.tcp_rmem=4096 87380 16777216
net.ipv4.tcp_wmem=4096 65536 16777216
net.core.netdev_max_backlog=3000
```