# 关于本文档

文档名称：《企业级调度器 Linux Virtual Server》

使用协议：《知识共享公共许可协议(CCPL)》

## 贡献者

| 贡献者名称 | 贡献度 | 文档变更记录 | 个人主页 |
| --- | --- | --- | --- |
| 马哥（马永亮） | 主编 | | http://github.com/iKubernetes/ |
| 王晓春 | 作者 | 50页 | http://www.wangxiaochun.com |

## 文档协议

**署名要求**：使用本系列文档，您必须保留本页中的文档来源信息，具体请参考《知识共享 (Creative Commons) 署名4.0公共许可协议国际版》。

**非商业化使用**：遵循《知识共享公共许可协议(CCPL)》，并且您不能将本文档用于马哥教育相关业务之外的其他任何商业用途。

**您的权利**：遵循本协议后，在马哥教育相关业务之外的领域，您将有以下使用权限：

共享 — 允许以非商业性质复制本作品。

改编 — 在原基础上修改、转换或以本作品为基础进行重新编辑并用于个人非商业使用。

## 致谢

本文档中，部分素材参考了相关项目的文档，以及通过搜索引擎获得的内容，这里先一并向相关的贡献者表示感谢。

# Linux Virtual Server

## 本章内容

- **集群概念**

- **LVS 模型**
- **LVS 调度算法**
- **LVS 实战案例**
- **LVS 高可用性**

# 1 集群和分布式

系统性能扩展方式：

- Scale UP：垂直扩展，向上扩展,增强，性能更强的计算机运行同样的服务
- Scale Out：水平扩展，向外扩展,增加设备，并行地运行多个服务调度分配问题，Cluster

垂直扩展不再提及：

随着计算机性能的增长，其价格会成倍增长

单台计算机的性能是有上限的，不可能无限制地垂直扩展

多核CPU意味着即使是单台计算机也可以并行的。那么，为什么不一开始就并行化技术?

## 1.1 集群 Cluster

Cluster：集群,为解决某个特定问题将多台计算机组合起来形成的单个系统

Cluster 分为三种类型：

- LB：Load Balancing，负载均衡，多个主机组成，每个主机只承担一部分访问请求
- HA：High Availiablity，高可用，避免SPOF（single Point Of failure）

  MTBF:Mean Time Between Failure 平均无故障时间，正常时间

  MTTR:Mean Time To Restoration（ repair）平均恢复前时间，故障时间

  A = MTBF /（MTBF+MTTR） (0,1)：99%,99.5%,99.9%,99.99%,99.999%

  **SLA**：服务等级协议（简称：SLA，全称：service level agreement）。是指在一定开销下为保障服务的性能和可用性，服务提供商与用户间定义的一种双方认可的协定。通常这个开销是驱动提供服务质量的主要因素。在常规的领域中，总是设定所谓的三个9，四个9,N个9 等来进行表示，当没有达到这种水平的时候，就会有一些列的惩罚措施，而运维最主要的目标就是达成这种服务水平。

  停机时间又分为两种，一种是计划内停机时间，一种是计划外停机时间，而运维则主要关注计划外停机时间。

  ```
  1年 = 365天 = 8760小时
  90 = (1-90%)*365=36.5天
  99 = 8760 * 1% = 87.6小时
  99.9 = 8760 * 0.1% = 8760 * 0.001 = 8.76小时
  99.99 = 8760 * 0.0001 = 0.876小时 = 0.876 * 60 = 52.6分钟
  99.999 = 8760 * 0.00001 = 0.0876小时 = 0.0876 * 60 = 5.26分钟
  99.9999= (1-99.9999%)*365*24*60*60=31秒
  ```

- HPC：High-performance computing，高性能 http://www.top500.org

## 1.2 分布式系统

分布式常见应用

- 分布式应用-服务按照功能拆分，使用微服务
- 分布式静态资源--静态资源放在不同的存储集群上
- 分布式数据和存储--使用key-value缓存系统
- 分布式计算--对特殊业务使用分布式计算，比如Hadoop集群

分布式存储： Ceph，GlusterFS，FastDFS，MogileFS

分布式计算：hadoop，Spark

# 1.3 集群和分布式


集群：同一个业务系统部署在多台服务器上。集群中每一台服务器实现的功能没有差别，数据和代码都是一样的

分布式：一个业务被拆成多个子业务，或者本身就是不同的业务，部署在多台服务器上。分布式中，每一台服务器实现的功能是有差别的，数据和代码也是不一样的，分布式每台服务器功能加起来，才是完整的业务

分布式是以缩短单个任务的执行时间来提升效率的，而集群则是通过提高单位时间内执行的任务数来提升效率。

对于大型网站，访问用户很多，实现一个群集，在前面部署一个负载均衡服务器，后面几台服务器完成同一业务。如果有用户进行相应业务访问时，负载均衡器根据后端哪台服务器的负载情况，决定由给哪一台去完成响应，并且一台服务器垮了，其它的服务器可以顶上来。分布式的每一个节点，都完成不同的业务，如果一个节点垮了，那这个业务可能就会失败

# 1.4 LB Cluster 负载均衡集群

## 1.4.1 按实现方式划分

- 硬件

F5 Big-IP https://detail.zol.com.cn/load_leveling/f5/cheap_pic.html?qq-pf-to=pcqq.group

Citrix Netscaler

A10

- 软件

  lvs：Linux Virtual Server，阿里云四层 SLB (Server Load Balance)使用

  nginx：支持七层调度，阿里云七层SLB使用 Tengine

  haproxy：支持七层调度

  ats：Apache Traffic Server，yahoo捐助给apache

  perlbal：Perl 编写

  pound

### 1.4.2 基于工作的协议层次划分

- 传输层（通用）：DNAT 和 DPORT

  LVS：

  nginx：stream

  haproxy：mode tcp

- 应用层（专用）：针对特定协议，常称为 proxy server

  http：nginx, httpd, haproxy(mode http), ...

  fastcgi：nginx, httpd, ...

  mysql：mysql-proxy, mycat...

### 1.4.3 负载均衡的会话保持

1. session sticky：同一用户调度固定服务器

   Source IP：LVS sh算法（对某一特定服务而言）

   Cookie

2. session replication：每台服务器拥有全部session

   session multicast cluster

3. session server：专门的session服务器

   Redis,Memcached

## 1.5 HA 高可用集群实现

keepalived：vrrp协议

Ais：应用接口规范

- heartbeat
- cman+rgmanager(RHCS)
- coresync_pacemaker

# 2 Linux Virtual Server 简介

## 2.1 LVS 介绍

LVS：Linux Virtual Server，负载调度器，内核集成，章文嵩（花名 正明），阿里的四层SLB(Server Load Balance)是基于LVS+keepalived实现

LVS 是全球最流行的四层负载均衡开源软件，由章文嵩博士（当前阿里云产品技术负责人）在1998年5月创立，可以实现LINUX平台下的负载均衡。

LVS 官网：http://www.linuxvirtualserver.org/

阿里SLB和LVS：

```
https://yq.aliyun.com/articles/1803
https://github.com/alibaba/LVS
```

整个SLB系统由3部分构成：四层负载均衡，七层负载均衡 和 控制系统，如下图所示；

- 四层负载均衡，采用开源软件LVS（linux virtual server），并根据云计算需求对其进行了定制化；该技术已经在阿里巴巴内部业务全面上线应用2年多详见第3节；
- 七层负载均衡，采用开源软件Tengine；该技术已经在阿里巴巴内部业务全面上线应用3年多；参见第4节；
- 控制系统，用于 配置和监控 负载均衡系统；

## 2.2 LVS 工作原理

VS根据请求报文的目标IP和目标协议及端口将其调度转发至某RS，根据调度算法来挑选RS。LVS是内核级功能，工作在INPUT链的位置，将发往INPUT的流量进行“处理”

范例：Ubuntu22.04

```
[root@ubuntu2204 ~]#uname -r
5.15.0-52-generic

[root@ubuntu2204 ~]#grep -i -C 10 ipvs /boot/config-5.15.0-52-generic
# IPVS scheduler
#
CONFIG_IP_VS_RR=m
CONFIG_IP_VS_WRR=m
CONFIG_IP_VS_LC=m
CONFIG_IP_VS_WLC=m
CONFIG_IP_VS_FO=m
CONFIG_IP_VS_OVF=m
CONFIG_IP_VS_LBLC=m
CONFIG_IP_VS_LBLCR=m
CONFIG_IP_VS_DH=m
CONFIG_IP_VS_SH=m
CONFIG_IP_VS_MH=m
CONFIG_IP_VS_SED=m
CONFIG_IP_VS_NQ=m
CONFIG_IP_VS_TWOS=m

#
# IPVS SH scheduler
```

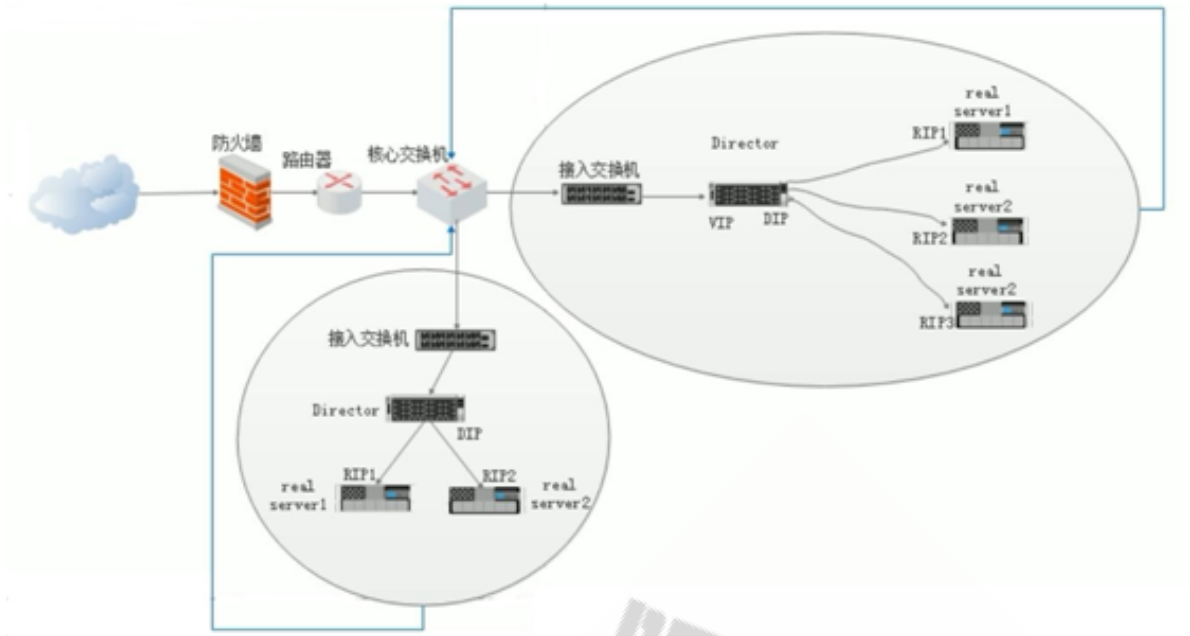
```
#
CONFIG_IP_VS_SH_TAB_BITS=8

#
# IPVS MH scheduler
#
CONFIG_IP_VS_MH_TAB_INDEX=12

#
# IPVS application helper
#
CONFIG_IP_VS_FTP=m
CONFIG_IP_VS_NFCT=y
CONFIG_IP_VS_PE_SIP=m

#
# IP: Netfilter Configuration
#
CONFIG_NF_DEFRAG_IPV4=m
CONFIG_NF_SOCKET_IPV4=m


[root@ubuntu2204 ~]#modinfo ip_vs
filename:       /lib/modules/5.15.0-52-
generic/kernel/net/netfilter/ipvs/ip_vs.ko
license:        GPL
srcversion:     09DB169E535CE1E0DA2FAE0
depends:        nf_conntrack,nf_defrag_ipv6,libcrc32c
retpoline:      Y
intree:         Y
name:           ip_vs
vermagic:       5.15.0-52-generic SMP mod_unload modversions
sig_id:         PKCS#7
signer:         Build time autogenerated kernel key
sig_key:        49:B2:3F:66:E1:3B:8B:67:11:CE:17:63:41:27:D0:B1:28:DF:09:8C
sig_hashalgo:   sha512
signature:      3F:00:57:50:9C:4A:D5:3E:3C:E0:A6:17:EE:F1:68:D3:40:77:A1:47:
        F9:79:F8:7A:E7:A3:02:14:9E:B5:21:2A:BE:F9:64:FB:BE:7E:09:5E:
        9A:2F:61:55:9C:A9:D4:C5:A0:6E:82:C8:25:8D:E6:5F:41:59:8A:AA:
        85:4E:59:36:BB:2A:18:67:B5:22:91:F5:53:95:78:3D:6A:C8:DB:32:
        9C:21:22:C7:26:07:AA:07:FA:89:1C:98:EC:E8:A3:88:95:B1:26:F8:
        CD:1C:7A:06:F5:D5:63:C4:EE:E4:54:8C:BB:E2:E9:C9:72:EB:30:EF:
        4D:61:35:12:46:4F:1E:4B:CA:8D:5E:A8:83:59:EE:70:73:BD:B0:54:
        C5:BA:A8:86:BA:F7:8C:BD:D1:43:00:1D:EC:70:22:6B:F3:61:BD:C1:
        4B:0B:FD:0D:39:E5:DB:8F:3D:D7:65:3B:18:3C:68:B2:BA:FB:D2:A9:
        32:C2:69:53:DF:05:A9:16:E7:70:06:AB:D5:2E:3C:FE:AE:E0:92:43:
        D1:F1:9C:1F:91:EA:8E:C5:40:3C:48:AE:3F:32:81:27:F3:2A:57:ED:
        58:B4:09:C4:5B:09:10:36:32:16:D1:7D:B4:A0:7A:E6:77:1A:93:80:
        26:A6:72:E1:9A:09:1C:59:66:77:D6:24:EA:4A:F5:0F:92:D4:AB:59:
        07:CB:2F:84:66:00:F4:A9:18:DF:60:4D:DE:D7:03:17:D5:7F:D7:5C:
        AD:54:1A:C7:7F:36:C1:AA:E7:47:6B:1E:84:A0:1B:53:01:2C:8B:F6:
        E1:6D:60:FE:6D:20:BB:0E:1E:9E:87:D0:5D:A6:9A:6A:C7:78:E2:A5:
        D5:FF:11:82:DD:25:21:82:DF:4A:2B:D5:5D:89:5A:E4:DB:83:C1:50:
        98:DB:93:32:AD:6E:20:B1:FE:20:8F:DA:9F:93:D6:A9:76:32:0D:70:
        8E:E8:D0:02:33:BB:8E:E5:F2:16:CD:90:3F:BF:41:93:96:95:E5:C7:
        93:2C:B4:0A:40:D9:9C:01:9F:99:E9:B0:F3:4A:5E:78:07:69:A6:9A:
        19:89:AF:EE:03:00:A7:FA:B7:83:BE:00:0E:B8:6D:42:E5:2B:96:BB:
        BB:E3:7A:F2:A3:A7:F7:9F:B0:29:6F:49:14:91:21:2A:A5:5C:2F:9D:
```

```
        AB:98:B2:BB:15:98:6B:E5:F0:4C:D3:49:5F:F0:98:E7:E2:12:A5:05:
        3E:FA:D9:E5:62:80:68:3E:8A:7A:B7:3D:DD:A9:34:3C:80:F3:B5:F8:
        82:AE:69:14:E8:20:4D:85:9A:10:C9:A2:24:E4:A5:B6:78:F7:F9:EE:
        C9:7E:5F:D5:26:F7:99:1B:DC:85:19:80
parm:           conn_tab_bits:Set connections' hash size (int)
```

范例：查看内核支持LVS

```
[root@centos8 ~]#grep -i -C 10 ipvs /boot/config-4.18.0-147.el8.x86_64

...（省略部分内容）...

CONFIG_NETFILTER_XT_MATCH_IPVS=m

CONFIG_NETFILTER_XT_MATCH_POLICY=m

...（省略部分内容）...
#
# IPVS transport protocol load balancing support
#
CONFIG_IP_VS_PROTO_TCP=y
CONFIG_IP_VS_PROTO_UDP=y
CONFIG_IP_VS_PROTO_AH_ESP=y
CONFIG_IP_VS_PROTO_ESP=y
CONFIG_IP_VS_PROTO_AH=y
CONFIG_IP_VS_PROTO_SCTP=y
#
# IPVS scheduler
#
CONFIG_IP_VS_RR=m
CONFIG_IP_VS_WRR=m
CONFIG_IP_VS_LC=m
CONFIG_IP_VS_WLC=m
CONFIG_IP_VS_FO=m  #新增
CONFIG_IP_VS_OVF=m  #新增
CONFIG_IP_VS_LBLC=m
CONFIG_IP_VS_LBLCR=m
CONFIG_IP_VS_DH=m
CONFIG_IP_VS_SH=m
# CONFIG_IP_VS_MH is not set
CONFIG_IP_VS_SED=m
CONFIG_IP_VS_NQ=m
...（省略部分内容）...
```

## 2.3 LVS 集群体系架构




## 2.4 LVS 集群类型中的术语

VS：Virtual Server，Director Server(DS), Dispatcher(调度器)，Load Balancer

RS：Real Server(lvs), upstream server(nginx), backend server(haproxy)

CIP：Client IP

VIP：Virtual server IP　VS外网的IP

DIP：Director IP　VS内网的IP

RIP：Real server IP

访问流程：CIP <--> VIP == DIP <--> RIP

# 3 LVS 工作模式和相关命令

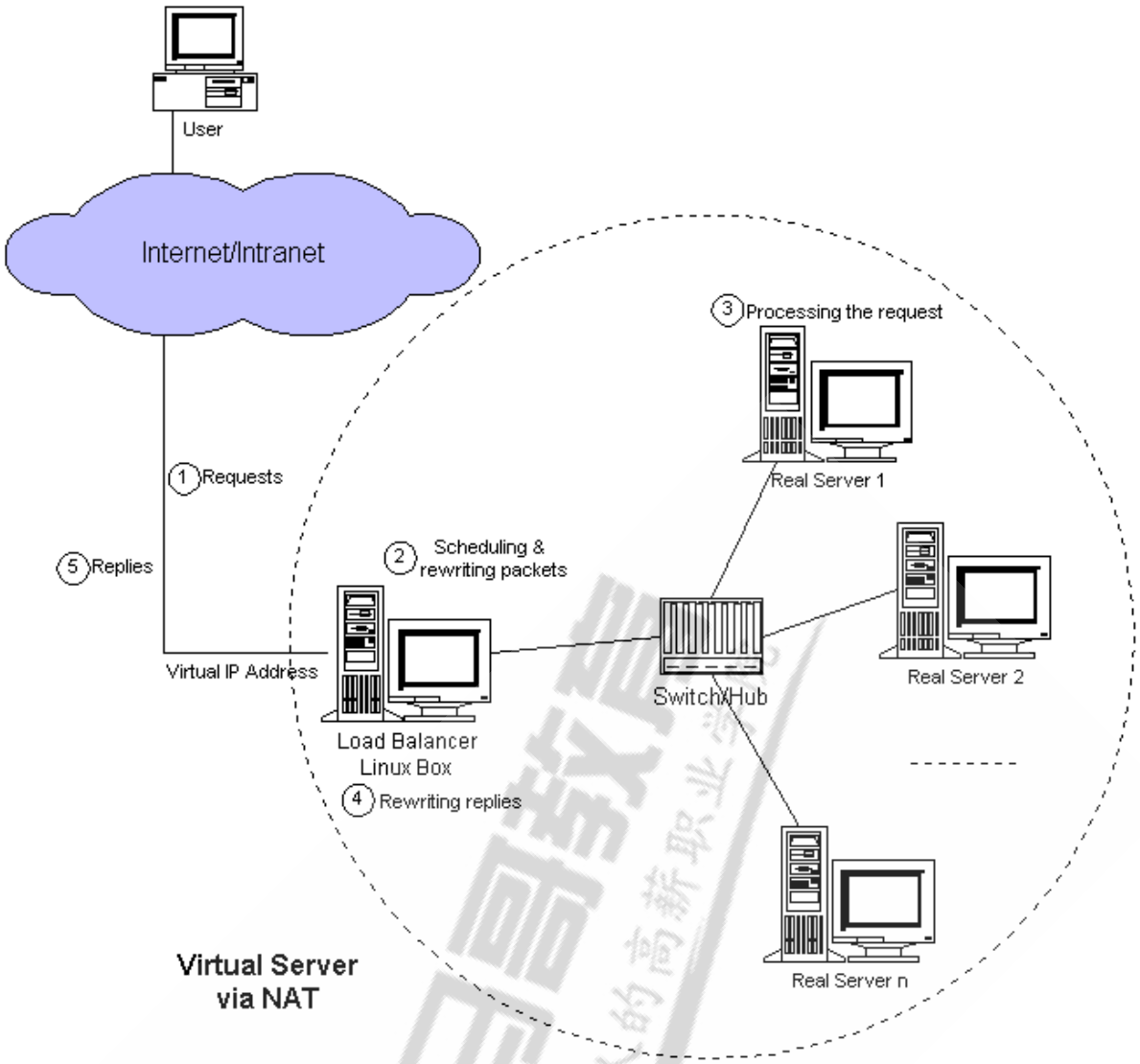
## 3.1 LVS 集群的工作模式

- lvs-nat：修改请求报文的目标IP,多目标IP的DNAT
- lvs-dr：操纵封装新的MAC地址
- lvs-tun：在原请求IP报文之外新加一个IP首部
- lvs-fullnat：修改请求报文的源和目标IP,默认内核不支持

### 3.1.1 LVS 的 NAT 模式

官方链接：

```
http://www.linuxvirtualserver.org/VS-NAT.html
```



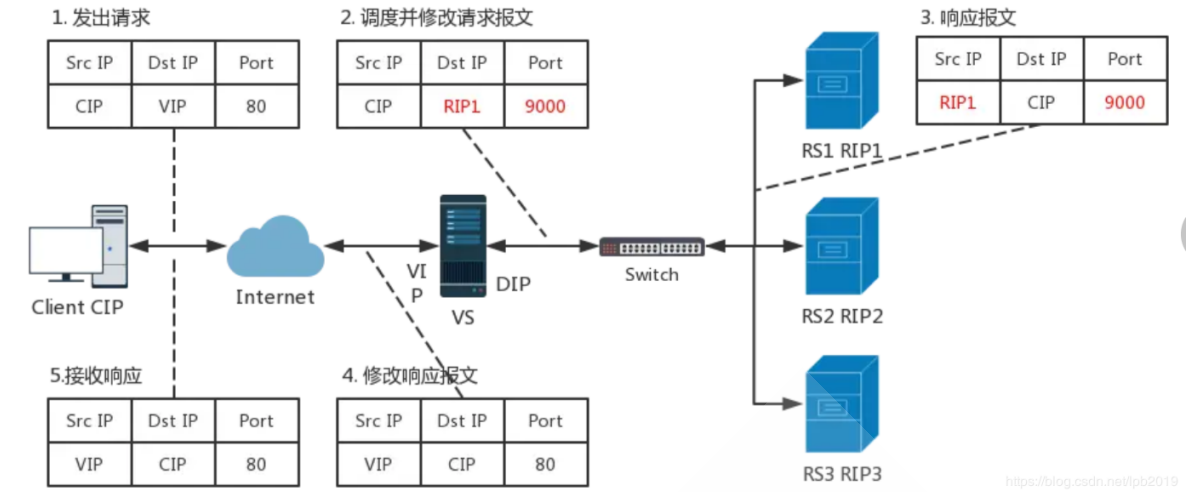

lvs-nat：本质是多目标IP的DNAT，通过将请求报文中的目标地址和目标端口修改为某挑出的RS的RIP和PORT实现转发

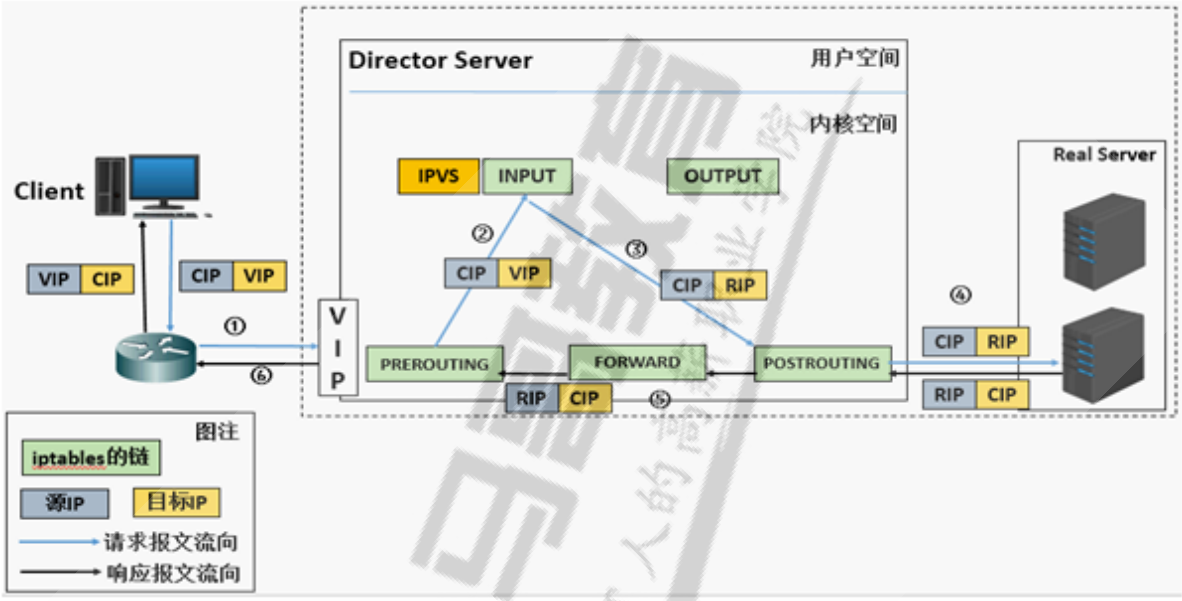
（1）RIP和DIP应在同一个IP网络，且应使用私网地址；RS的网关要指向DIP

（2）请求报文和响应报文都必须经由Director转发，Director易于成为系统瓶颈

（3）支持端口映射，可修改请求报文的目标PORT

（4）VS必须是Linux系统，RS可以是任意OS系统

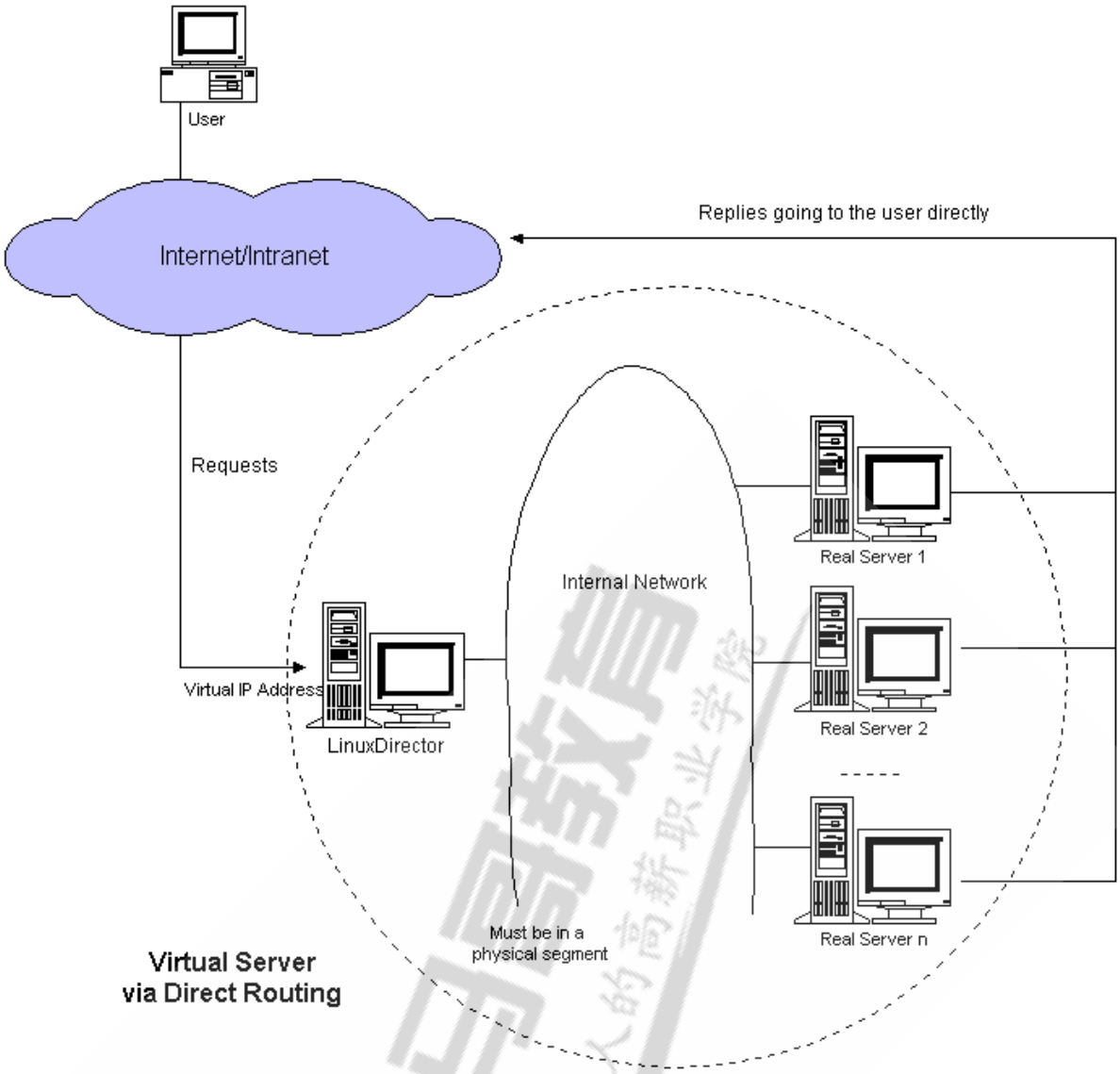


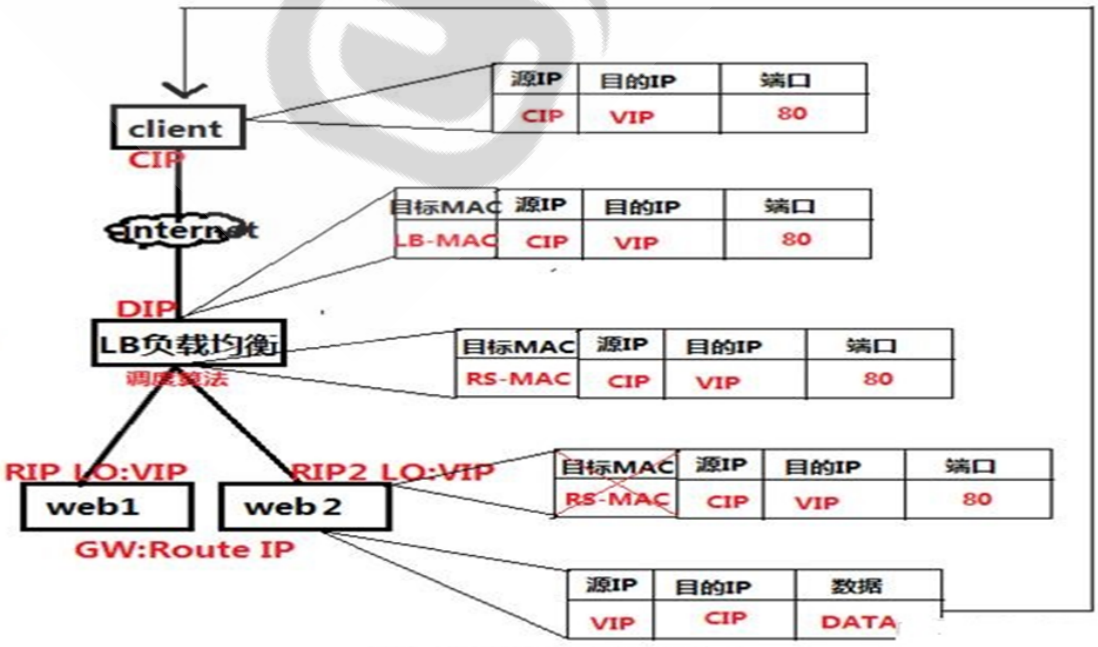

## 1.1.2 LVS 的 DR 模式

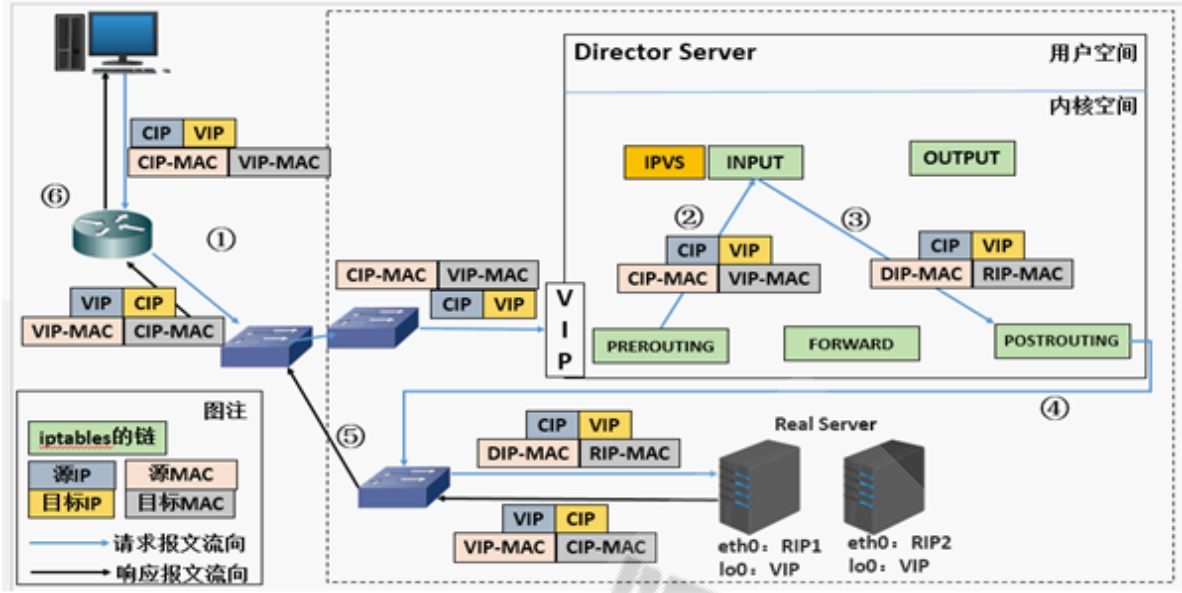
官方链接

```
http://www.linuxvirtualserver.org/VS-DRouting.html
```




LVS-DR：Direct Routing，直接路由，LVS默认模式,应用最广泛,通过为请求报文重新封装一个MAC首部进行转发，源MAC是DIP所在的接口的MAC，目标MAC是某挑选出的RS的RIP所在接口的MAC地址；源IP/PORT，以及目标IP/PORT均保持不变






**DR模式的特点**：

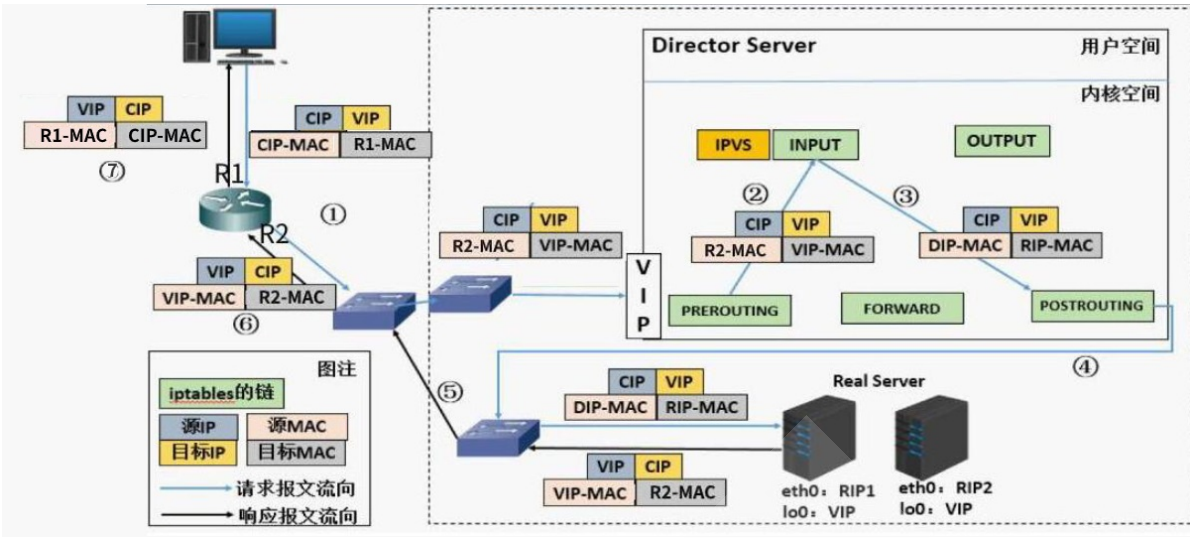
1. Director和各RS都配置有VIP
2. 确保前端路由器将目标IP为VIP的请求报文发往Director
   - 在前端网关做静态绑定VIP和Director的MAC地址
   - 在RS上使用arptables工具

```
arptables -A IN -d $VIP -j DROP
arptables -A OUT -s $VIP -j mangle --mangle-ip-s $RIP
```

   - 在RS上修改内核参数以限制arp通告及应答级别

```
不主动不负责不拒绝--M49-强福安语录
/proc/sys/net/ipv4/conf/all/arp_ignore
/proc/sys/net/ipv4/conf/all/arp_announce
```

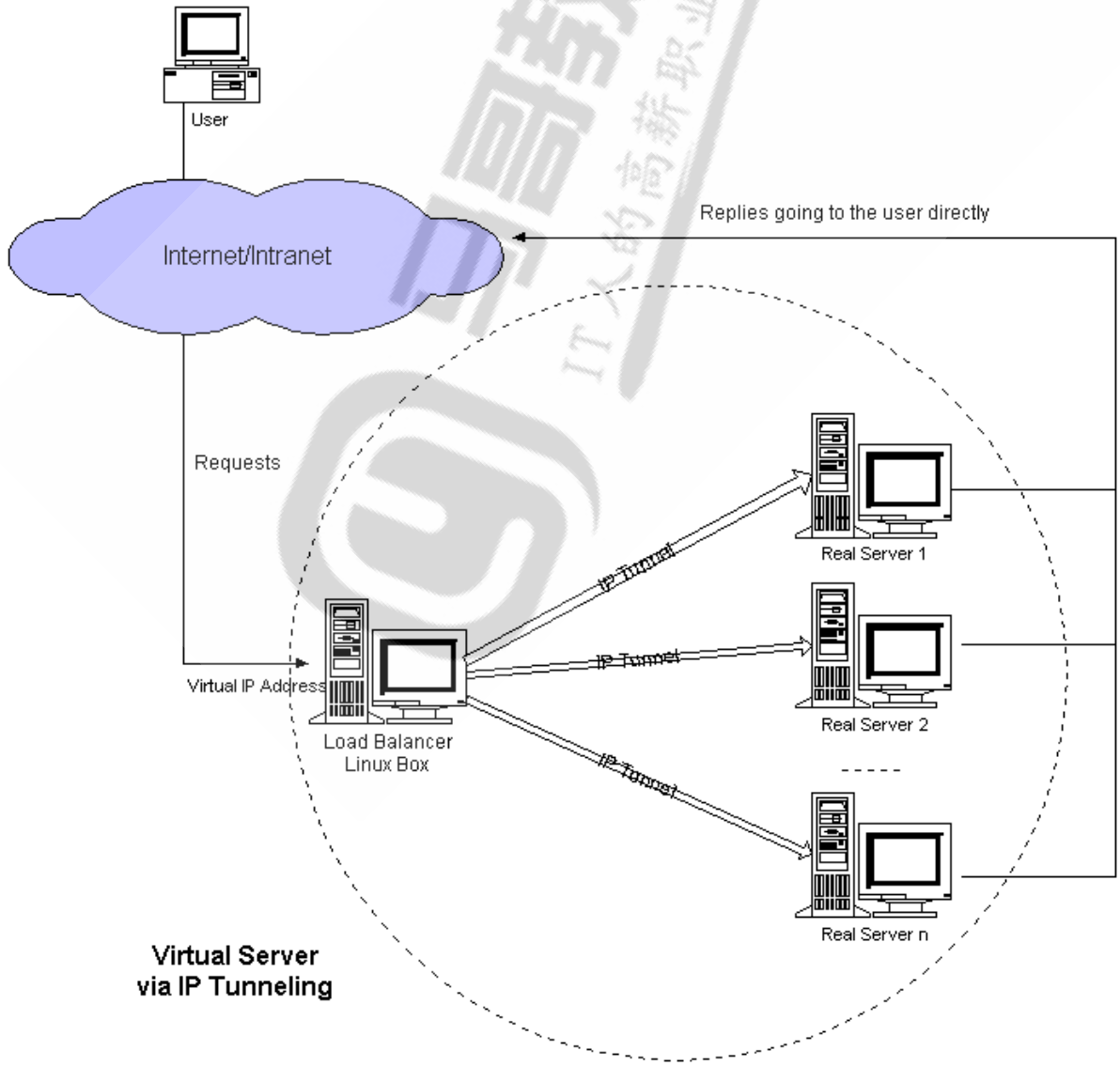
3. RS的RIP可以使用私网地址，也可以是公网地址；RIP与DIP在同一IP网络；RIP的网关不能指向DIP，以确保响应报文不会经由Director
4. RS和Director要在同一个物理网络
5. 请求报文要经由Director，但响应报文不经由Director，而由RS直接发往Client
6. 不支持端口映射（端口不能修改）
7. 无需开启 ip_forward
8. RS可使用大多数OS系统



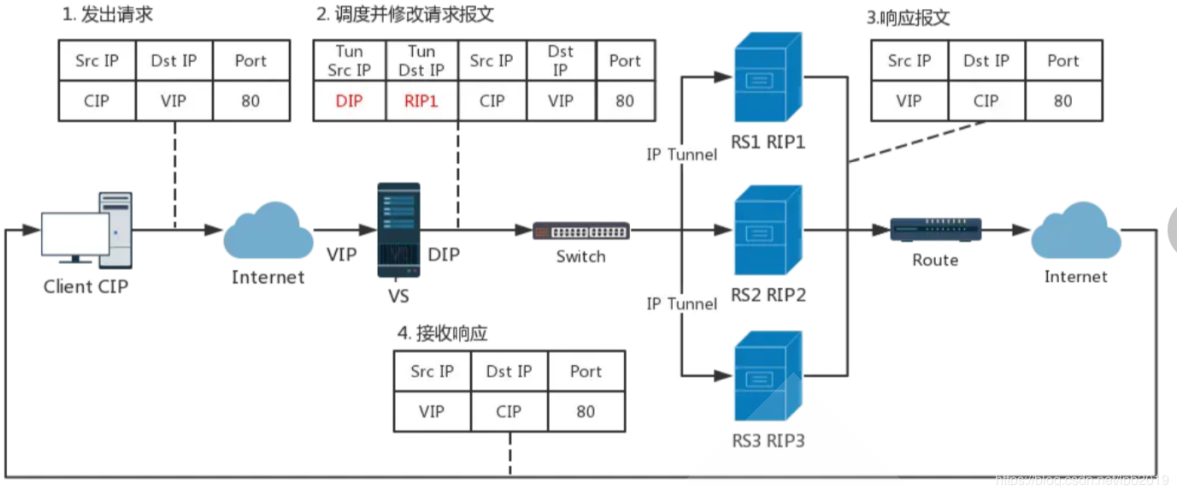

## 3.1.3 LVS 的 TUN 模式

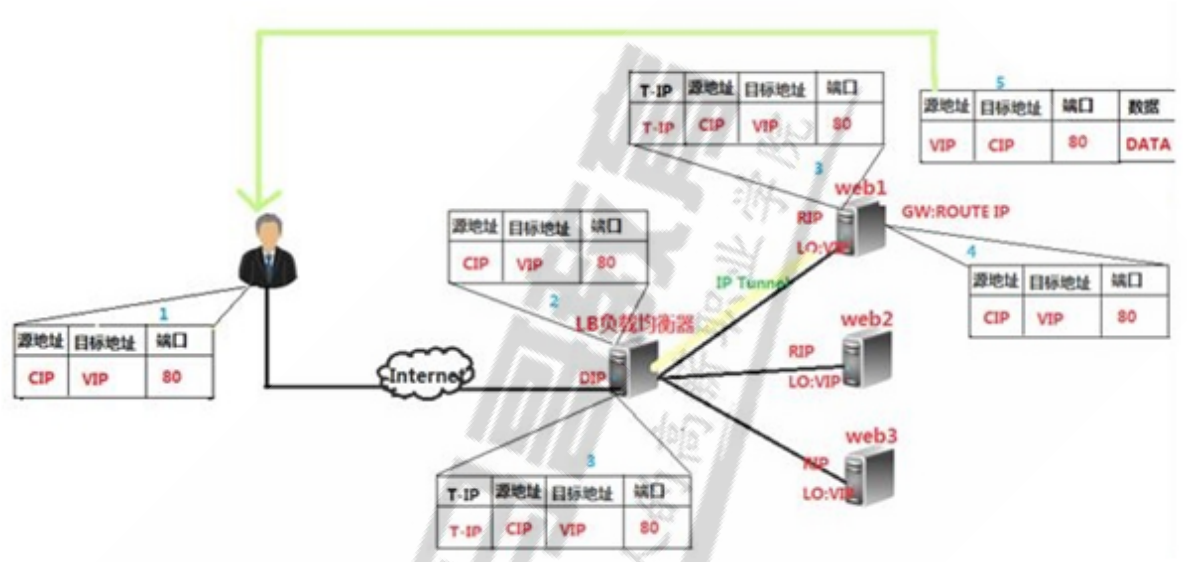
官方链接

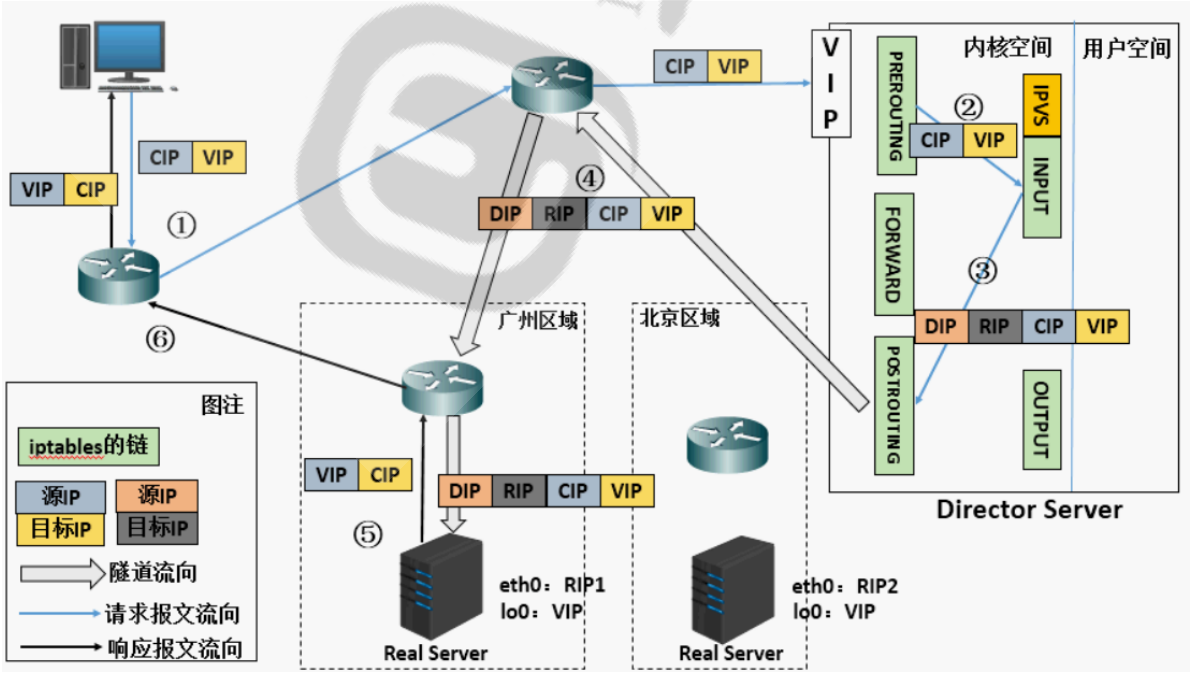
```
http://www.linuxvirtualserver.org/VS-IPTunneling.html
```




转发方式：不修改请求报文的IP首部（源IP为CIP，目标IP为VIP），而在原IP报文之外再封装一个IP首部（源IP是DIP，目标IP是RIP），将报文发往挑选出的目标RS；RS直接响应给客户端（源IP是VIP，目标IP是CIP）




TUN模式特点：

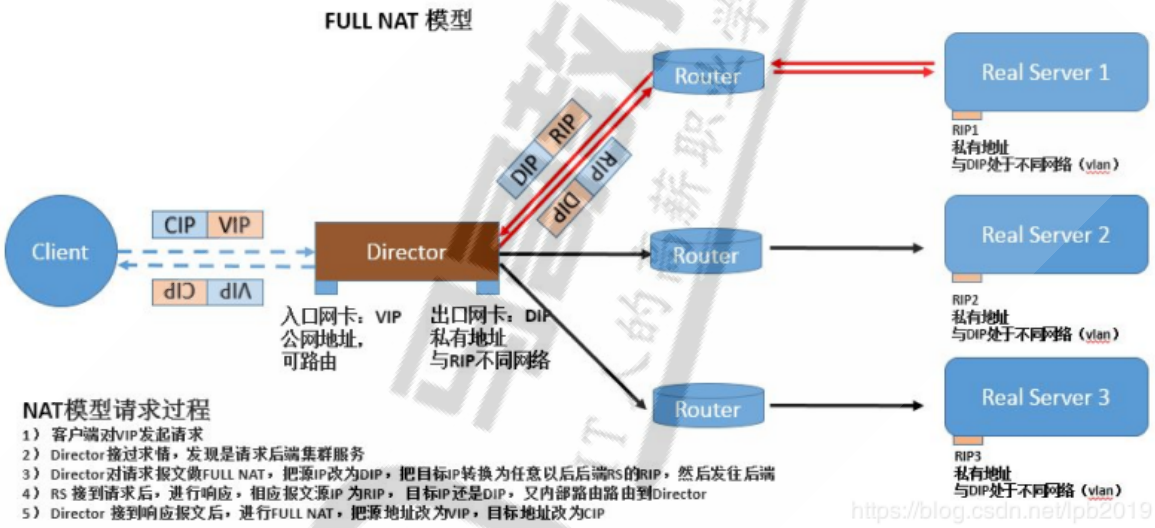
1. RIP和DIP可以不处于同一物理网络中，RS的网关一般不能指向DIP,且RIP可以和公网通信。也就是说集群节点可以跨互联网实现。DIP, VIP, RIP可以是公网地址

2. RealServer的tun接口上需要配置VIP地址，以便接收director转发过来的数据包，以及作为响应的报文源IP
3. Director转发给RealServer时需要借助隧道，隧道外层的IP头部的源IP是DIP，目标IP是RIP，而RealServer响应给客户端的IP头部是根据隧道内层的IP头分析得到的，源IP是VIP，目标IP是CIP
4. 请求报文要经由Director，但响应不经由Director,响应由RealServer自己完成
5. 不支持端口映射
6. RS的OS须支持隧道功能

应用场景:

> 一般来说，TUN模式常会用来负载调度缓存服务器组，这些缓存服务器一般放置在不同的网络环境，可以就近折返给客户端。在请求对象不在Cache服务器本地命中的情况下，Cache服务器要向源服务器发送请求，将结果取回，最后将结果返回给用户。
>
> LAN环境一般多采用DR模式，WAN环境虽然可以用TUN模式，但是一般在WAN环境下，请求转发更多的被haproxy/nginx/DNS等实现。因此，TUN模式实际应用的很少,跨机房的应用一般专线光纤连接或DNS调度

## 3.1.4 LVS 的 FULLNAT 模式




通过同时修改请求报文的源IP地址和目标IP地址进行转发

CIP --> DIP

VIP --> RIP

fullnat模式特点:

1. VIP是公网地址，RIP和DIP是私网地址，且通常不在同一IP网络；因此，RIP的网关一般不会指向DIP
2. RS收到的请求报文源地址是DIP，因此，只需响应给DIP；但Director还要将其发往Client
3. 请求和响应报文都经由Director
4. 相对NAT模式，可以更好的实现LVS-RealServer间跨VLAN通讯
5. 支持端口映射

注意：此类型kernel默认不支持

### 3.1.5 LVS工作模式总结和比较

| | NAT | TUN | DR |
|---|---|---|---|
| Real Server | any | Tunneling | Non-arp device |
| Real server network | private | LAN/WAN | LAN |
| Real server number | low (10~20) | High (100) | High (100) |
| Real server gateway | load balancer | own router | Own router |
| 优点 | 端口转换 | WAN | 性能最好 |
| 缺点 | 性能瓶颈 | 要求支持隧道,不支持端口转换 | 不支持跨网段和端口转换 |

lvs-nat与lvs-fullnat：

- 请求和响应报文都经由Director
- lvs-nat：RIP的网关要指向DIP
- lvs-fullnat：RIP和DIP未必在同一IP网络，但要能通信

lvs-dr与lvs-tun：

- 请求报文要经由Director，但响应报文由RS直接发往Client
- lvs-dr：通过封装新的MAC首部实现，通过MAC网络转发
- lvs-tun：通过在原IP报文外封装新IP头实现转发，支持远距离通信

总结

```
NAT  多目标的DNAT，四层，支持端口修改，请求报文和响应报文都要经过LVS
DR   默认模式，二层，只修改MAC，不支持端口修改，性能好，LVS负载比小，LVS和RS并在同一网段，请求报文经过LVS，响应报文不经过LVS
TUNNEL   三层，添加一个新的IP头，支持LVS和RS并在不在同一网段，不支持端口修改，请求报文经过LVS，响应报文不经过LVS
FULLNAT  多目标的SNAT+DNAT，四层，支持端口修改，请求报文和响应报文都要经过LVS
```

# 3.2 LVS 调度算法

ipvs scheduler：根据其调度时是否考虑各RS当前的负载状态

分为两种：静态方法和动态方法

## 3.2.1 静态方法

仅根据算法本身进行调度

1、轮询算法RR：roundrobin，轮询,较常用

2、加权轮询算法WRR：Weighted RR，较常用

3、源地址哈希算法SH：Source Hashing，实现session sticky，源IP地址hash；将来自于同一个IP地址的请求始终发往第一次挑中的RS，从而实现会话绑定

4、目标地址哈希算法DH：Destination Hashing；目标地址哈希，第一次轮询调度至RS，后续将发往同一个目标地址的请求始终转发至第一次挑中的RS，典型使用场景是正向代理缓存场景中的负载均衡，如: Web缓存

### 3.2.2 动态方法

主要根据每RS当前的负载状态及调度算法进行调度Overhead=value 较小的RS将被调度

1、最少连接算法LC：least connections 适用于长连接应用

```
Overhead=activeconns*256+inactiveconns
```

2、加权最少连接算法WLC：Weighted LC，默认调度方法,较常用

```
Overhead=(activeconns*256+inactiveconns)/weight
```

3、最短期望延迟算法SED：Shortest Expection Delay，初始连接高权重优先,只检查活动连接,而不考虑非活动连接

```
Overhead=(activeconns+1)*256/weight
```

4、最少队列算法NQ：Never Queue，第一轮均匀分配，后续SED

5、基于局部的最少连接算法LBLC：Locality-Based LC，动态的DH算法，使用场景：根据负载状态实现正向代理,实现Web Cache等

6、带复制的基于局部的最少连接算法LBLCR：LBLC with Replication，带复制功能的LBLC，解决LBLC负载不均衡问题，从负载重的复制到负载轻的RS,,实现Web Cache等

### 3.2.3 内核版本 4.15 版本后新增调度算法：FO和OVF

FO（Weighted Fail Over）调度算法,在此FO算法中，遍历虚拟服务所关联的真实服务器链表，找到还未过载（未设置IP_VS_DEST_F_OVERLOAD标志）的且权重最高的真实服务器，进行调度,属于静态算法

OVF（Overflow-connection）调度算法，基于真实服务器的活动连接数量和权重值实现。将新连接调度到权重值最高的真实服务器，直到其活动连接数量超过权重值，之后调度到下一个权重值最高的真实服务器,在此OVF算法中，遍历虚拟服务相关联的真实服务器链表，找到权重值最高的可用真实服务器。,属于动态算法

一个可用的真实服务器需要同时满足以下条件：

- 未过载（未设置IP_VS_DEST_F_OVERLOAD标志）
- 真实服务器当前的活动连接数量小于其权重值
- 其权重值不为零

## 3.3 LVS 相关软件

### 3.3.1 程序包：ipvsadm

Unit File: ipvsadm.service

主程序：/usr/sbin/ipvsadm

规则保存工具：/usr/sbin/ipvsadm-save

规则重载工具：/usr/sbin/ipvsadm-restore

配置文件：/etc/sysconfig/ipvsadm-config

ipvs调度规则文件：/etc/sysconfig/ipvsadm

## 3.3.2 ipvsadm 命令

ipvsadm核心功能：

- 集群服务管理：增、删、改
- 集群服务的RS管理：增、删、改
- 查看

ipvsadm 工具用法：

```
#管理集群服务
ipvsadm -A|E -t|u|f service-address [-s scheduler] [-p [timeout]] [-M netmask]
[--pe persistence_engine] [-b sched-flags]
ipvsadm -D -t|u|f service-address #删除
ipvsadm -C #清空
ipvsadm -R #重载,相当于ipvsadm-restore
ipvsadm -S [-n] #保存,相当于ipvsadm-save

#管理集群中的RS
ipvsadm -a|e -t|u|f service-address -r server-address [-g|i|m] [-w weight]
ipvsadm -d -t|u|f service-address -r server-address
ipvsadm -L|l [options]
ipvsadm -Z [-t|u|f service-address]
```

**管理集群服务：增、改、删**

**增、修改：**

```
ipvsadm -A|E -t|u|f service-address [-s scheduler] [-p [timeout]]
```

说明

```
service-address：

-t|u|f：
    -t：TCP协议的端口，VIP:TCP_PORT  如：-t 10.0.0.100:80
    -u：UDP协议的端口，VIP:UDP_PORT
    -f：firewall MARK，标记，一个数字
[-s scheduler]：指定集群的调度算法，默认为wlc
```

范例：

```
ipvsadm -A -t 10.0.0.100:80 -s wrr
```

**删除：**

```
ipvsadm -D -t|u|f service-address
```

**管理集群上的RS：增、改、删**

增、改：

```
ipvsadm -a|e -t|u|f service-address -r server-address [-g|i|m] [-w weight]
```

删：

```
ipvsadm -d -t|u|f service-address -r server-address
```

```
server-address：
     rip[:port] 如省略port，不作端口映射
选项：
lvs类型：
    -g：gateway，dr类型，默认
    -i：ipip，tun类型
    -m：masquerade，nat类型
-w weight：权重
```

范例：

```
ipvsadm -a -t  10.0.0.100:80 -r 10.0.0.8:8080 -m -w 3
```

**清空定义的所有内容：**

```
ipvsadm -C
```

**清空计数器：**

```
ipvsadm -Z [-t|u|f service-address]
```

**查看：**

```
ipvsadm -L|l [options]
```

```
--numeric，-n：以数字形式输出地址和端口号
--exact：扩展信息，精确值
--connection，-c：当前IPVS连接输出
--stats：统计信息
--rate ：输出速率信息
```

**ipvs规则：**

```
/proc/net/ip_vs
```

**ipvs连接：**

```
/proc/net/ip_vs_conn
```

**保存：建议保存至/etc/sysconfig/ipvsadm**

```
ipvsadm-save > /PATH/TO/IPVSADM_FILE
ipvsadm -S > /PATH/TO/IPVSADM_FILE
systemctl stop ipvsadm.service  #会自动保存规则至/etc/sysconfig/ipvsadm
```

**重载：**

```
ipvsadm-restore < /PATH/FROM/IPVSADM_FILE
systemctl  start ipvsadm.service  #会自动加载/etc/sysconfig/ipvsadm中规则
```

范例： Ubuntu系统保存规则和开机加载规则

```
#保存规则
#方法1
[root@ubuntu2204 ~]#service ipvsadm save
 * Saving IPVS configuration...
                                                       [ OK ]
[root@ubuntu2204 ~]#cat /etc/ipvsadm.rules
# ipvsadm.rules
-A -t 192.168.10.100:80 -s wlc
-a -t 192.168.10.100:80 -r 10.0.0.7:80 -g -w 1
-a -t 192.168.10.100:80 -r 10.0.0.17:80 -g -w 1

#方法2
[root@ubuntu2004 ~]#ipvsadm-save  -n > /etc/ipvsadm.rules

#开机自启
[root@ubuntu2004 ~]#vim /etc/default/ipvsadm
AUTO="true"
```

范例：红帽系统保存规则和开机加载规则

```
#保存规则
[root@rocky8 ~]#ipvsadm-save  -n > /etc/sysconfig/ipvsadm

#开机自启
[root@rocky8 ~]#systemctl enable ipvsadm.service
```

## 3.4 防火墙标记

FWM：FireWall Mark

MARK target 可用于给特定的报文打标记

--set-mark value

其中：value 可为0xffff格式，表示十六进制数字

借助于防火墙标记来分类报文，而后基于标记定义集群服务；可将多个不同的应用使用同一个集群服务进行调度

实现方法：

在Director主机打标记：

```
iptables -t mangle -A PREROUTING -d $vip -p $proto -m multiport --dports
$port1,$port2,… -j MARK --set-mark NUMBER
```

在Director主机基于标记定义集群服务：

```
ipvsadm -A -f NUMBER [options]
```

范例:

```
[root@lvs ~]#iptables -t mangle -A PREROUTING -d 172.16.0.100 -p tcp -m
multiport --dports 80,443 -j MARK --set-mark 10
[root@lvs ~]#ipvsadm -C
[root@lvs ~]#ipvsadm -A -f 10 -s rr
[root@lvs ~]#ipvsadm -a -f 10 -r 10.0.0.7 -g
[root@lvs ~]#ipvsadm -a -f 10 -r 10.0.0.17 -g
[root@lvs ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
FWM  10 rr
  -> 10.0.0.7:0                   Route   1      0          0
  -> 10.0.0.17:0                  Route   1      0          0
[root@lvs ~]#cat /proc/net/ip_vs
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port Forward Weight ActiveConn InActConn
FWM  0000000A rr
  -> 0A000011:0000      Route   1      0          9
  -> 0A000007:0000      Route   1      0          9
```

范例:

```
[root@lvs ~]#ipvsadm -A -f 10
[root@lvs ~]#ipvsadm -a -f 10  -r  10.0.0.7 -g
[root@lvs ~]#ipvsadm -a -f 10  -r  10.0.0.17 -g
[root@lvs ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
FWM  10 wlc
  -> 10.0.0.7:0                Route   1      0          0
  -> 10.0.0.17:0               Route   1      0          0

[root@LVS ~]#cat /proc/net/ip_vs
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port Forward Weight ActiveConn InActConn
TCP  AC14C8C8:0050 rr
  -> 0A000011:0050      Masq    1      0          0
  -> 0A000007:0050      Masq    1      0          0
```

# 3.5 LVS 持久连接

session 绑定：对共享同一组RS的多个集群服务，需要统一进行绑定，lvs sh算法无法实现

持久连接（ lvs persistence ）模板：实现无论使用任何调度算法，在一段时间内（默认360s ），能够实现将来自同一个地址的请求始终发往同一个RS

```
ipvsadm -A|E -t|u|f service-address [-s scheduler] [-p [timeout]]
```

持久连接实现方式：

- 每端口持久（PPC）：每个端口定义为一个集群服务，每集群服务单独调度
- 每防火墙标记持久（PFWMC）：基于防火墙标记定义集群服务；可实现将多个端口上的应用统一调度，即所谓的 port Affinity
- 每客户端持久（PCC）：基于0端口（表示所有服务）定义集群服务，即将客户端对所有应用的请求都调度至后端主机，必须定义为持久模式

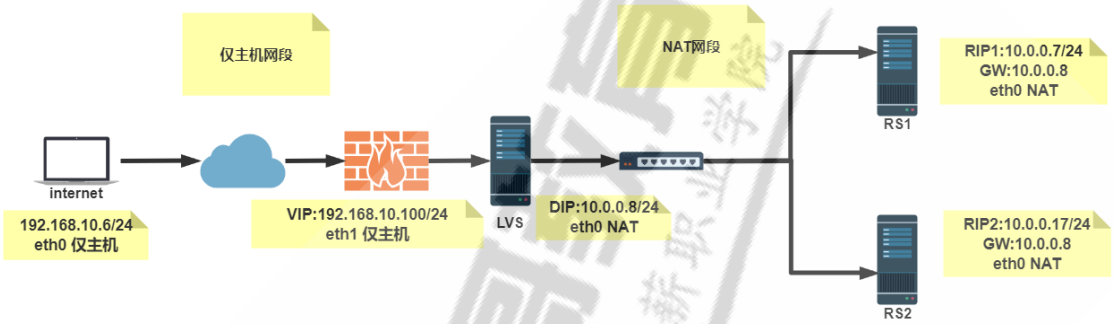
范例：

```
[root@lvs ~]#ipvsadm -E -f 10 -p
[root@lvs ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
FWM  10 wlc persistent 360
  -> 10.0.0.7:0               Route   1      0          15
  -> 10.0.0.17:0              Route   1      0          7
[root@lvs ~]#ipvsadm -E -f 10 -p 3600
[root@lvs ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
FWM  10 wlc persistent 3600
  -> 10.0.0.7:0               Route   1      0          79
  -> 10.0.0.17:0              Route   1      0          7

[root@lvs ~]#cat /proc/net/ip_vs_conn
Pro FromIP   FPrt ToIP     TPrt DestIP   DPrt State       Expires PEName PEData
TCP C0A80006 C816 AC100064 01BB 0A000011 01BB FIN_WAIT         67
TCP C0A80006 C812 AC100064 01BB 0A000011 01BB FIN_WAIT         67
TCP C0A80006 9A36 AC100064 0050 0A000011 0050 FIN_WAIT         65
TCP C0A80006 C806 AC100064 01BB 0A000011 01BB FIN_WAIT         65
TCP C0A80006 9A3E AC100064 0050 0A000011 0050 FIN_WAIT         66
TCP C0A80006 C81A AC100064 01BB 0A000011 01BB FIN_WAIT         67
TCP C0A80006 C80A AC100064 01BB 0A000011 01BB FIN_WAIT         66
TCP C0A80006 9A3A AC100064 0050 0A000011 0050 FIN_WAIT         66
TCP C0A80006 9A4E AC100064 0050 0A000011 0050 FIN_WAIT         68
TCP C0A80006 9A42 AC100064 0050 0A000011 0050 FIN_WAIT         67
TCP C0A80006 9A46 AC100064 0050 0A000011 0050 FIN_WAIT         67
TCP C0A80006 C81E AC100064 01BB 0A000011 01BB FIN_WAIT         68
IP  C0A80006 0000 0000000A 0000 0A000011 0000 NONE            948
TCP C0A80006 C80E AC100064 01BB 0A000011 01BB FIN_WAIT         66
TCP C0A80006 9A4A AC100064 0050 0A000011 0050 FIN_WAIT         67


[root@lvs ~]#ipvsadm -Lnc
IPVS connection entries
pro expire state       source             virtual            destination
TCP 00:46  FIN_WAIT    192.168.10.6:51222  172.16.0.100:443   10.0.0.17:443
TCP 00:46  FIN_WAIT    192.168.10.6:51218  172.16.0.100:443   10.0.0.17:443
TCP 00:45  FIN_WAIT    192.168.10.6:39478  172.16.0.100:80    10.0.0.17:80
TCP 00:45  FIN_WAIT    192.168.10.6:51206  172.16.0.100:443   10.0.0.17:443
TCP 00:46  FIN_WAIT    192.168.10.6:39486  172.16.0.100:80    10.0.0.17:80
TCP 00:47  FIN_WAIT    192.168.10.6:51226  172.16.0.100:443   10.0.0.17:443
```

```
TCP 00:45  FIN_WAIT    192.168.10.6:51210  172.16.0.100:443   10.0.0.17:443
TCP 00:45  FIN_WAIT    192.168.10.6:39482  172.16.0.100:80    10.0.0.17:80
TCP 00:47  FIN_WAIT    192.168.10.6:39502  172.16.0.100:80    10.0.0.17:80
TCP 00:46  FIN_WAIT    192.168.10.6:39490  172.16.0.100:80    10.0.0.17:80
TCP 00:46  FIN_WAIT    192.168.10.6:39494  172.16.0.100:80    10.0.0.17:80
TCP 00:47  FIN_WAIT    192.168.10.6:51230  172.16.0.100:443   10.0.0.17:443
IP  15:27  NONE        192.168.10.6:0      0.0.0.10:0         10.0.0.17:0
TCP 00:46  FIN_WAIT    192.168.10.6:51214  172.16.0.100:443   10.0.0.17:443
TCP 00:47  FIN_WAIT    192.168.10.6:39498  172.16.0.100:80    10.0.0.17:80
```

# 4 LVS 实战案例

## 4.1 LVS-NAT模式案例




环境：

```
共四台主机
一台：  internet client：192.168.10.6/24   GW:无 仅主机

一台：lvs
eth1 仅主机 192.168.10.100/16
eth0 NAT 10.0.0.8/24

两台RS：
RS1：10.0.0.7/24    GW：10.0.0.8 NAT
RS2：10.0.0.17/24   GW：10.0.0.8 NAT
```

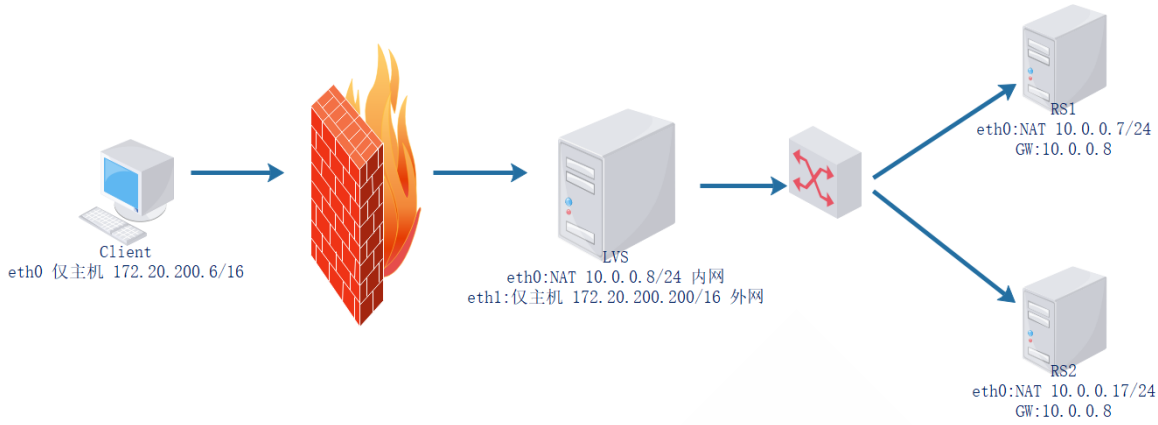
配置过程：

```
[root@internet ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=192.168.10.6
PREFIX=24
ONBOOT=yes


[root@lvs network-scripts]#cat ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.8
```

```
PREFIX=24
ONBOOT=yes
[root@lvs network-scripts]#cat ifcfg-eth1
DEVICE=eth1
NAME=eth1
BOOTPROTO=static
IPADDR=192.168.10.100
PREFIX=24
ONBOOT=yes

[root@rs1 ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.7
PREFIX=24
GATEWAY=10.0.0.8
ONBOOT=yes

[root@rs2 ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.17
PREFIX=24
GATEWAY=10.0.0.8
ONBOOT=yes


[root@rs1 ~]#curl 10.0.0.7
10.0.0.7 RS1

[root@rs2 ~]#curl 10.0.0.17
10.0.0.17 RS2


[root@lvs-server ~]#vim /etc/sysctl.conf
net.ipv4.ip_forward = 1

[root@lvs-server ~]#sysctl -p
net.ipv4.ip_forward = 1

[root@lvs-server ~]#ipvsadm -A -t 192.168.10.100:80  -s wrr
[root@lvs-server ~]#ipvsadm -a -t 192.168.10.100:80  -r 10.0.0.7:80 -m
[root@lvs-server ~]#ipvsadm -a -t 192.168.10.100:80  -r 10.0.0.17:80 -m

[root@lvs-server ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.10.100:80 wrr
  -> 10.0.0.7:80                  Masq    1      1          0
  -> 10.0.0.17:80                 Masq    1      0          0

[root@internet ~]#while :;do curl 192.168.10.100;sleep 0.5;done
rs1.wang.org
rs2.wang.org
rs1.wang.org
```

```
rs2.wang.org
rs1.wang.org
rs2.wang.org

[root@lvs-server ~]#ipvsadm -Ln --stats
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port               Conns   InPkts  OutPkts  InBytes OutBytes
  -> RemoteAddress:Port
TCP  192.168.10.100:80                  67      405      255    32436    30092
  -> 10.0.0.7:80                        34      203      128    16244    15072
  -> 10.0.0.17:80                       33      202      127    16192    15020

[root@lvs-server ~]#cat /proc/net/ip_vs
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port Forward Weight ActiveConn InActConn
TCP  C0A80A64:0050 wrr
  -> 0A000011:0050      Masq    1      0          98
  -> 0A000007:0050      Masq    1      0          97

[root@lvs-server ~]#ipvsadm -Lnc
IPVS connection entries
pro expire state       source             virtual            destination
TCP 01:55  TIME_WAIT   192.168.10.6:43486 192.168.10.100:80  10.0.0.17:80
TCP 00:19  TIME_WAIT   192.168.10.6:43476 192.168.10.100:80  10.0.0.7:80
TCP 01:58  TIME_WAIT   192.168.10.6:43500 192.168.10.100:80  10.0.0.7:80
TCP 01:58  TIME_WAIT   192.168.10.6:43498 192.168.10.100:80  10.0.0.17:80
TCP 01:59  TIME_WAIT   192.168.10.6:43502 192.168.10.100:80  10.0.0.17:80
TCP 01:57  TIME_WAIT   192.168.10.6:43494 192.168.10.100:80  10.0.0.17:80
TCP 01:57  TIME_WAIT   192.168.10.6:43496 192.168.10.100:80  10.0.0.7:80
TCP 01:56  TIME_WAIT   192.168.10.6:43490 192.168.10.100:80  10.0.0.17:80
TCP 00:20  TIME_WAIT   192.168.10.6:43480 192.168.10.100:80  10.0.0.7:80
TCP 01:56  TIME_WAIT   192.168.10.6:43492 192.168.10.100:80  10.0.0.7:80
TCP 01:55  TIME_WAIT   192.168.10.6:43488 192.168.10.100:80  10.0.0.7:80
TCP 00:20  TIME_WAIT   192.168.10.6:43478 192.168.10.100:80  10.0.0.17:80
TCP 01:59  TIME_WAIT   192.168.10.6:43504 192.168.10.100:80  10.0.0.7:80
TCP 01:54  TIME_WAIT   192.168.10.6:43484 192.168.10.100:80  10.0.0.7:80
TCP 01:54  TIME_WAIT   192.168.10.6:43482 192.168.10.100:80  10.0.0.17:80

[root@lvs-server ~]#cat /proc/net/ip_vs_conn
Pro FromIP   FPrt ToIP     TPrt DestIP   DPrt State       Expires PEName PEData
TCP C0A80A06 A9DE C0A80A64 0050 0A000011 0050 TIME_WAIT        72
TCP C0A80A06 A9EC C0A80A64 0050 0A000007 0050 TIME_WAIT        76
TCP C0A80A06 AA64 C0A80A64 0050 0A000007 0050 TIME_WAIT       106
TCP C0A80A06 AA0C C0A80A64 0050 0A000007 0050 TIME_WAIT        84
TCP C0A80A06 AA3A C0A80A64 0050 0A000011 0050 TIME_WAIT        95
TCP C0A80A06 AA86 C0A80A64 0050 0A000011 0050 TIME_WAIT       115
TCP C0A80A06 AA78 C0A80A64 0050 0A000007 0050 TIME_WAIT       111
TCP C0A80A06 AA06 C0A80A64 0050 0A000011 0050 TIME_WAIT        82
TCP C0A80A06 AA44 C0A80A64 0050 0A000007 0050 TIME_WAIT        98
TCP C0A80A06 AA2C C0A80A64 0050 0A000007 0050 TIME_WAIT        92


#保存规则
[root@lvs-server ~]#ipvsadm -Sn  > /etc/sysconfig/ipvsadm
[root@lvs-server ~]#systemctl enable --now ipvsadm.service

#问题:LVS 打开监听VIP相关的端口吗?
```

范例2：




1. Director 服务器采用双网卡，一个是桥接网卡连接外网，一个是仅主机网卡与后端Web服务器相连
2. Web服务器采用仅主机网卡与director相连
3. Web服务器网关指向10.0.0.200
4. 后端web服务器不需要连接外网

环境：

```
共四台主机
一台：  internet client ：172.20.200.6/16   GW:无
一台：lvs
eth1 桥接 172.20.200.200/16
eth0 NAT 10.0.0.200/24

两台RS：
RS1：10.0.0.7/24     GW：  10.0.0.200
RS2：10.0.0.17/24    GW：  10.0.0.200
```

配置过程

```
#LVS启用IP_FORWORD功能
[root@lvs ~]#vim /etc/sysctl.conf
net.ipv4.ip_forward = 1
[root@LVS ~]#sysctl  -p

[root@lvs ~]#ipvsadm -A -t 172.20.200.200:80 -s rr
[root@lvs ~]#ipvsadm -a -t 172.20.200.200:80 -r 10.0.0.7 -m
[root@lvs ~]#ipvsadm -a -t 172.20.200.200:80 -r 10.0.0.17 -m

[root@LVS ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.20.200.200:80 rr
  -> 10.0.0.7:80                  Masq    1      0          0
  -> 10.0.0.17:80                 Masq    1      0          0

#测试
[root@client ~]#curl 172.20.200.200
RS2 Server on 10.0.0.17
```

```
[root@client ~]#curl 172.20.200.200
RS1 Server on 10.0.0.7
[root@client ~]#curl 172.20.200.200
RS2 Server on 10.0.0.17
[root@client ~]#curl 172.20.200.200
RS1 Server on 10.0.0.7


[root@LVS ~]#cat /proc/net/ip_vs_conn
Pro FromIP   FPrt ToIP     TPrt DestIP   DPrt State       Expires PEName PEData
TCP AC14C806 BD6A AC14C8C8 0050 0A000011 0050 TIME_WAIT        97
TCP AC14C806 BD6C AC14C8C8 0050 0A000007 0050 TIME_WAIT        97
TCP AC14C806 BD66 AC14C8C8 0050 0A000011 0050 TIME_WAIT        90
TCP AC14C806 BD68 AC14C8C8 0050 0A000007 0050 TIME_WAIT        92


#保存规则
[root@LVS ~]#ipvsadm -Sn > /etc/sysconfig/ipvsadm
[root@LVS ~]#cat /etc/sysconfig/ipvsadm
-A -t 172.20.200.200:80 -s rr
-a -t 172.20.200.200:80 -r 10.0.0.7:80 -m -w 1
-a -t 172.20.200.200:80 -r 10.0.0.17:80 -m -w 1

#清除规则
[root@LVS ~]#ipvsadm -C
[root@LVS ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn

#重新加载规则
[root@LVS ~]#ipvsadm -R <  /etc/sysconfig/ipvsadm
[root@LVS ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.20.200.200:80 rr
  -> 10.0.0.7:80                  Masq    1      0          0
  -> 10.0.0.17:80

#开机加载ipvs规则
[root@LVS ~]#ipvsadm -C
[root@LVS ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
[root@LVS ~]#systemctl enable --now ipvsadm.service
[root@LVS ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.20.200.200:80 rr
  -> 10.0.0.7:80                  Masq    1      0          0
  -> 10.0.0.17:80                 Masq    1      0          0

[root@rs1 ~]#tail /var/log/httpd/access_log
```

```
172.20.200.6 - - [24/Mar/2020:16:38:29 +0800] "GET / HTTP/1.1" 200 23 "-"
"curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.27.1 zlib/1.2.3
libidn/1.18 libssh2/1.4.2"
172.20.200.6 - - [24/Mar/2020:16:38:35 +0800] "GET / HTTP/1.1" 200 23 "-"
"curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.27.1 zlib/1.2.3
libidn/1.18 libssh2/1.4.2"
172.20.200.6 - - [24/Mar/2020:16:52:16 +0800] "GET / HTTP/1.1" 200 23 "-"
"curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.27.1 zlib/1.2.3
libidn/1.18 libssh2/1.4.2"
172.20.200.6 - - [24/Mar/2020:16:52:17 +0800] "GET / HTTP/1.1" 200 23 "-"
"curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.27.1 zlib/1.2.3
libidn/1.18 libssh2/1.4.2"
172.20.200.6 - - [24/Mar/2020:16:53:36 +0800] "GET / HTTP/1.1" 200 23 "-"
"curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.27.1 zlib/1.2.3
libidn/1.18 libssh2/1.4.2"
172.20.200.6 - - [24/Mar/2020:16:53:37 +0800] "GET / HTTP/1.1" 200 23 "-"
"curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.27.1 zlib/1.2.3
libidn/1.18 libssh2/1.4.2"

#修改调度算法为 WRR 和后端服务器的端口
[root@LVS ~]#ipvsadm -E -t 172.20.200.200:80 -s wrr
[root@LVS ~]#ipvsadm -d -t 172.20.200.200:80 -r 10.0.0.7
[root@LVS ~]#ipvsadm -a -t 172.20.200.200:80 -r 10.0.0.7:8080 -m -w 3
[root@LVS ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.20.200.200:80 wrr
  -> 10.0.0.7:8080                Masq    3      0          0
  -> 10.0.0.17:80                 Masq    1      0          1

[root@rs1 ~]#vim /etc/httpd/conf/httpd.conf
Listen 8080
[root@rs1 ~]#systemctl restart httpd

[root@client ~]#curl 172.20.200.200
RS1 Server on 10.0.0.7
[root@client ~]#curl 172.20.200.200
RS1 Server on 10.0.0.7
[root@client ~]#curl 172.20.200.200
RS1 Server on 10.0.0.7
[root@client ~]#curl 172.20.200.200
RS2 Server on 10.0.0.17
```

## 4.2 LVS-DR模式单网段案例

DR模型中各主机上均需要配置VIP，解决地址冲突的方式有三种：

(1) 在前端网关做静态绑定

(2) 在各RS使用arptables

(3) 在各RS修改内核参数，来限制arp响应和通告的级别

**限制响应级别：arp_ignore**

- 0：默认值，表示可使用本地任意接口上配置的任意地址进行响应
- 1：仅在请求的目标IP配置在本地主机的接收到请求报文的接口上时，才给予响应

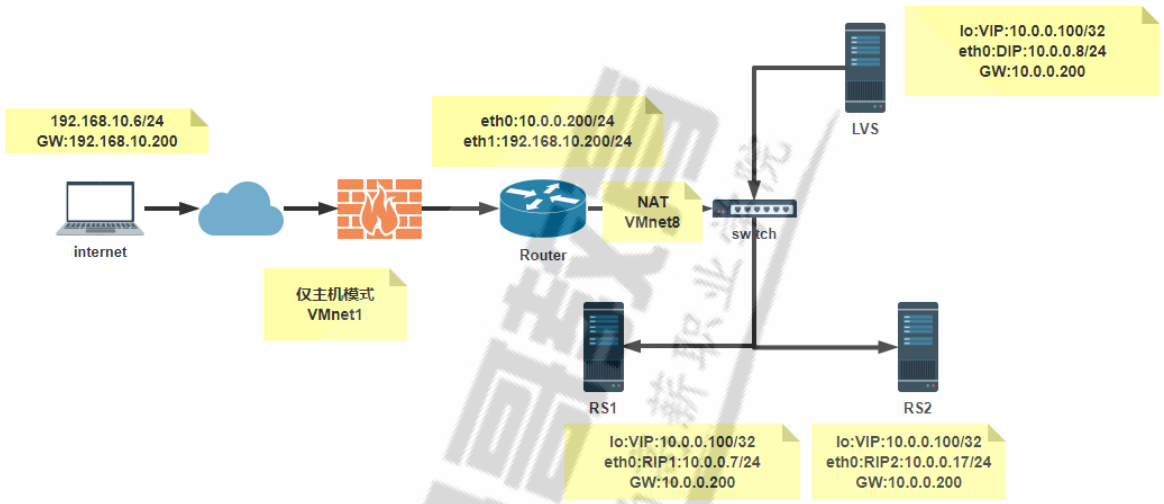
**限制通告级别：arp_announce**

- 0：默认值，把本机所有接口的所有信息向每个接口的网络进行通告
- 1：尽量避免将接口信息向非直接连接网络进行通告
- 2：必须避免将接口信息向非本网络进行通告

**配置要点**

1. Director 服务器采用双IP桥接网络，一个是VIP，一个DIP
2. Web服务器采用和DIP相同的网段和Director连接
3. 每个Web服务器配置VIP
4. 每个web服务器可以出外网

范例:




```
环境：五台主机
一台：客户端 eth0:仅主机 192.168.10.6/24 GW:192.168.10.200

一台：ROUTER
eth0 :NAT  10.0.0.200/24
eth1: 仅主机 192.168.10.200/24
启用 IP_FORWARD

一台：LVS
eth0:NAT:DIP:10.0.0.8/24 GW:10.0.0.200

两台RS：
RS1：eth0:NAT:10.0.0.7/24    GW：10.0.0.200
RS2：eth0:NAT:10.0.0.17/24   GW：10.0.0.200
```

## 4.2.1 LVS的网络配置

```
#所有主机禁用iptables和SELinux

#internet主机环境
[root@internet ~]#hostname
internet
[root@internet ~]#hostname -I
192.168.10.6
[root@internet ~]#route -n
Kernel IP routing table
```

```
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.10.0     0.0.0.0         255.255.255.0   U     1      0        0 eth0
0.0.0.0         192.168.10.200   0.0.0.0         UG    0      0        0 eth0

[root@internet ~]#ping 10.0.0.7 -c1
PING 10.0.0.7 (10.0.0.7) 56(84) bytes of data.
64 bytes from 10.0.0.7: icmp_seq=1 ttl=63 time=0.565 ms

[root@internet ~]#ping 10.0.0.17 -c1
PING 10.0.0.7 (10.0.0.17) 56(84) bytes of data.
64 bytes from 10.0.0.17: icmp_seq=1 ttl=63 time=0.565 ms

################################################################################
######
#路由器的网络配置
[root@router ~]#echo 'net.ipv4.ip_forward=1' >>  /etc/sysctl.conf
[root@router ~]#sysctl -p

[root@router network-scripts]#pwd
/etc/sysconfig/network-scripts
[root@router network-scripts]#cat ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.200
PREFIX=24
ONBOOT=yes
[root@router network-scripts]#cat ifcfg-eth1
DEVICE=eth1
NAME=eth1
BOOTPROTO=static
IPADDR=192.168.10.200
PREFIX=24
ONBOOT=yes


################################################################################
######
#RS1的网络配置
[root@rs1 ~]#hostname
rs1.wang.org
[root@rs1 ~]#hostname -I
10.0.0.7
[root@rs1 ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.7
PREFIX=24
GATEWAY=10.0.0.200
ONBOOT=yes
[root@rs1 ~]#route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.200      0.0.0.0         UG    100    0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 eth0

[root@rs1 ~]#yum -y install httpd
```

```
[root@rs1 ~]#systemctl enable --now httpd
[root@rs1 ~]#hostname -I > /var/www/html/index.html
[root@rs1 ~]#ping 192.168.10.6 -c1
PING 192.168.10.6 (192.168.10.6) 56(84) bytes of data.
64 bytes from 192.168.10.6: icmp_seq=1 ttl=63 time=1.14 ms

[root@rs1 ~]#curl 10.0.0.7
10.0.0.7


################################################################################
######
#RS2 的网络配置
[root@rs2 ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.17
PREFIX=24
GATEWAY=10.0.0.200
ONBOOT=yes
[root@rs2 ~]#route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.200      0.0.0.0         UG    100    0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 eth0

[root@rs2 ~]#yum -y install httpd
[root@rs2 ~]#systemctl enable --now httpd
[root@rs2 ~]#hostname -I > /var/www/html/index.html
[root@rs2 ~]#curl 10.0.0.17
10.0.0.17
[root@rs1 ~]#ping 192.168.10.6 -c1
PING 192.168.10.6 (192.168.10.6) 56(84) bytes of data.
64 bytes from 192.168.10.6: icmp_seq=1 ttl=63 time=1.14 ms


[root@rs2 ~]#curl 10.0.0.17
10.0.0.17


################################################################################
######
#LVS的网络配置
[root@lvs ~]#hostname
lvs.wang.org
[root@lvs ~]#hostname -I
10.0.0.8

[root@lvs ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.8
PREFIX=24
GATEWAY=10.0.0.200
ONBOOT=yes
[root@lvs ~]#route -n
```

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.200      0.0.0.0         UG    100    0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 eth0
[root@lvs ~]#ping 192.168.10.6 -c1
PING 192.168.10.6 (192.168.10.6) 56(84) bytes of data.
64 bytes from 192.168.10.6: icmp_seq=1 ttl=63 time=2.32 ms

#如果LVS没有配置网关（网关随意，只有要即可），也可以通过修改内核关闭路径反向校验实现
[root@lvs ~]#echo "0" > /proc/sys/net/ipv4/conf/all/rp_filter
[root@lvs ~]#echo "0" > /proc/sys/net/ipv4/conf/eth0/rp_filter
```

## 4.2.2 后端RS的IPVS配置

```
#RS1的IPVS配置
[root@rs1 ~]#echo 1 >   /proc/sys/net/ipv4/conf/all/arp_ignore
[root@rs1 ~]#echo 2 >   /proc/sys/net/ipv4/conf/all/arp_announce
[root@rs1 ~]#echo 1 >   /proc/sys/net/ipv4/conf/lo/arp_ignore
[root@rs1 ~]#echo 2 >   /proc/sys/net/ipv4/conf/lo/arp_announce
[root@rs1 ~]#ifconfig lo:1 10.0.0.100/32
[root@rs1 ~]#ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet 10.0.0.100/0 scope global lo:1
       valid_lft forever preferred_lft forever
    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
group default qlen 1000
    link/ether 00:0c:29:01:f9:48 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.7/24 brd 10.0.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe01:f948/64 scope link
       valid_lft forever preferred_lft forever

#RS2的IPVS配置
[root@rs2 ~]#echo 1 >   /proc/sys/net/ipv4/conf/all/arp_ignore
[root@rs2 ~]#echo 1 >   /proc/sys/net/ipv4/conf/lo/arp_ignore
[root@rs2 ~]#echo 2 >   /proc/sys/net/ipv4/conf/all/arp_announce
[root@rs2 ~]#echo 2 >   /proc/sys/net/ipv4/conf/lo/arp_announce
[root@rs2 ~]#ifconfig lo:1 10.0.0.100/32
[root@rs2 ~]#ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet 10.0.0.100/0 scope global lo:1
       valid_lft forever preferred_lft forever
    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
group default qlen 1000
    link/ether 00:0c:29:94:1a:f6 brd ff:ff:ff:ff:ff:ff
```

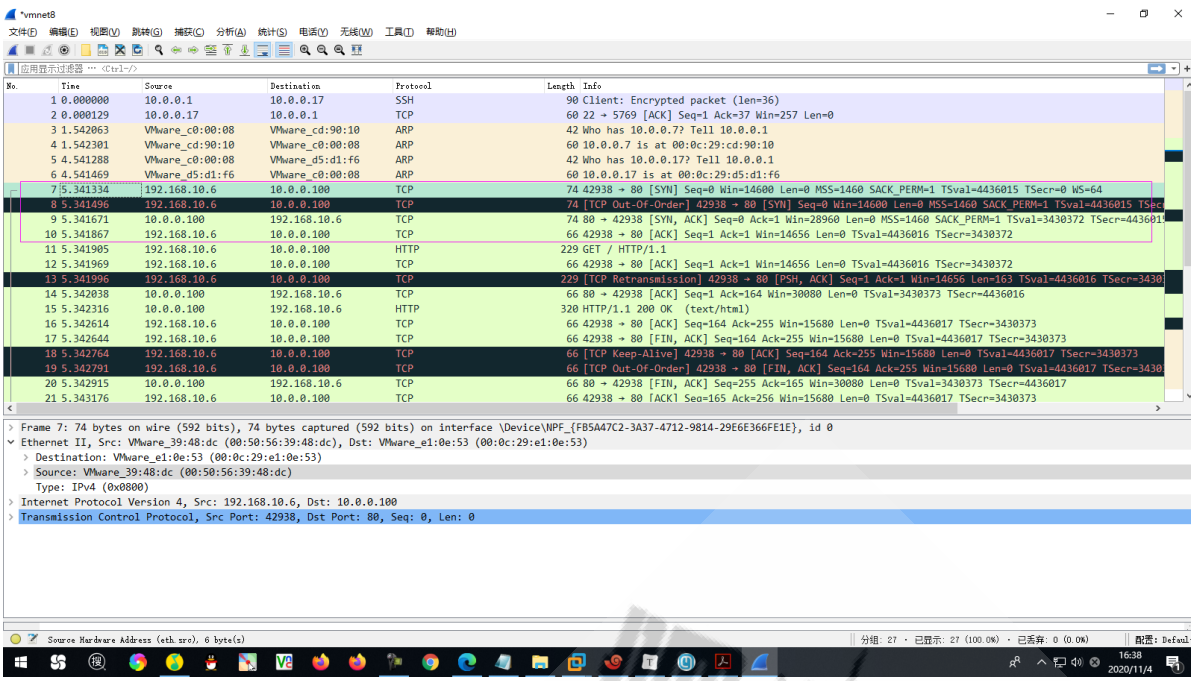
```
    inet 10.0.0.17/24 brd 10.0.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe94:1af6/64 scope link
       valid_lft forever preferred_lft forever
```

### 4.2.3 LVS主机的配置

```
#在LVS上添加VIP
[root@lvs ~]#ifconfig lo:1 10.0.0.100/32
[root@lvs ~]#ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet 10.0.0.100/0 scope global lo:1
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 00:0c:29:8a:51:21 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.8/24 brd 10.0.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever

#实现LVS 规则
[root@lvs ~]#dnf -y install ipvsadm
[root@lvs ~]#ipvsadm -A -t 10.0.0.100:80 -s rr
[root@lvs ~]#ipvsadm -a -t 10.0.0.100:80 -r 10.0.0.7:80 -g
[root@lvs ~]#ipvsadm -a -t 10.0.0.100:80 -r 10.0.0.17:80 -g
[root@lvs ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  10.0.0.100:80 rr
  -> 10.0.0.7:80                  Route   1      0          0
  -> 10.0.0.17:80                 Route   1      0          0
```

### 4.2.4 测试访问

```
[root@internet ~]#curl 10.0.0.100
10.0.0.17
[root@internet ~]#curl 10.0.0.100
10.0.0.7

[root@rs1 ~]#tail -f /var/log/httpd/access_log -n0
192.168.10.6 - - [12/Jul/2020:10:36:21 +0800] "GET / HTTP/1.1" 200 10 "-"
"curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.27.1 zlib/1.2.3
libidn/1.18 libssh2/1.4.2"
```



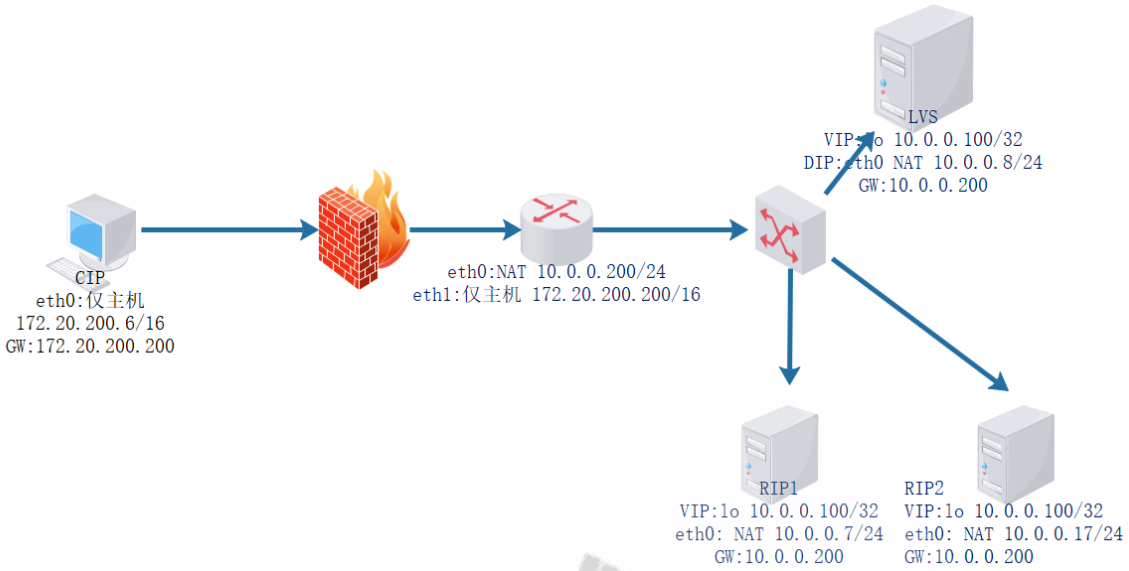

# 4.2.5 思考

## 4.2.5.1 LVS的eth0的网关可否不配置？如果随便配置，发现什么问题？如果不配置，怎么解决?

```
#默认值
[root@ubuntu2204 ~]#cat /proc/sys/net/ipv4/conf/all/rp_filter
2
[root@centos8 ~]#cat /proc/sys/net/ipv4/conf/all/rp_filter
1

#Ubuntu22.04修改内核参数为0
[root@ubuntu2204 ~]#echo "0" > /proc/sys/net/ipv4/conf/all/rp_filter
[root@ubuntu2204 ~]#echo "0" > /proc/sys/net/ipv4/conf/eth0/rp_filter

#Rocky8修改内核参数为0
[root@lvs ~]#echo "0" > /proc/sys/net/ipv4/conf/all/rp_filter

#说明:参数rp_filter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。对每一个收到的数据包，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表示开启松散的反向路径校验，对每个收到的数据包，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。
```

## 4.2.5.2 LVS的VIP可以配置到lo网卡,但必须使用32位的netmask,为什么?

**范例：**




```
环境：五台主机
一台：客户端 172.20.200.6/16 GW:172.20.200.200
一台：ROUTER
eth0 :NAT  10.0.0.200/24 VIP
eth1: 桥接 172.20.200.200/16
启用 IP_FORWARD
一台：LVS
eth0: 10.0.0.8/24 GW:10.0.0.200
两台RS：
RS1：10.0.0.7/24      GW：10.0.0.200
RS2：10.0.0.17/24   GW：10.0.0.200
```

配置：

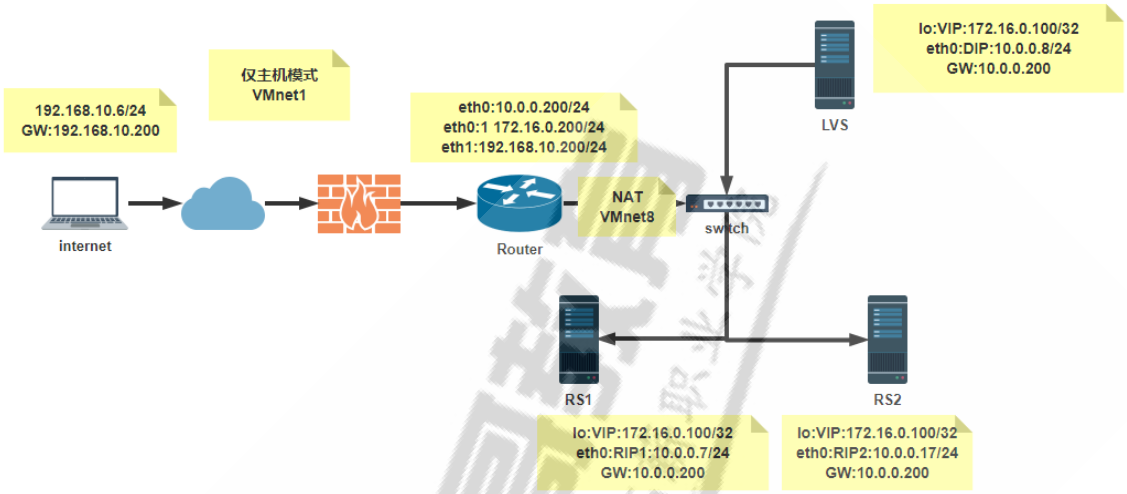
```
[root@client ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=172.20.200.6
PREFIX=16
GATEWAY=172.20.200.200
ONBOOT=yes

[root@Router ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.200
PREFIX=24
ONBOOT=yes
[root@Router ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE=eth1
NAME=eth1
BOOTPROTO=static
IPADDR=172.20.200.200
PREFIX=16
ONBOOT=yes
```

```
[root@Router ~]#cat /etc/sysctl.conf
net.ipv4.ip_forward=1
[root@Router ~]#sysctl -p

[root@rs1 ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.7
PREFIX=24
GATEWAY=10.0.0.200
ONBOOT=yes


[root@rs1 ~]#echo 1 >   /proc/sys/net/ipv4/conf/all/arp_ignore
[root@rs1 ~]#echo 2 >   /proc/sys/net/ipv4/conf/all/arp_announce
[root@rs1 ~]#echo 1 >   /proc/sys/net/ipv4/conf/lo/arp_ignore
[root@rs1 ~]#echo 2 >   /proc/sys/net/ipv4/conf/lo/arp_announce
[root@rs1 ~]#ifconfig lo:1 10.0.0.100/32
[root@rs1 ~]#ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet 10.0.0.100/0 scope global lo:1
       valid_lft forever preferred_lft forever
    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
group default qlen 1000
    link/ether 00:0c:29:32:80:38 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.7/24 brd 10.0.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe32:8038/64 scope link
       valid_lft forever preferred_lft forever


[root@rs2 ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.17
PREFIX=24
GATEWAY=10.0.0.200
ONBOOT=yes

[root@rs2 ~]#echo 1 >   /proc/sys/net/ipv4/conf/all/arp_ignore
[root@rs2 ~]#echo 2 >   /proc/sys/net/ipv4/conf/all/arp_announce
[root@rs2 ~]#echo 1 >   /proc/sys/net/ipv4/conf/lo/arp_ignore
[root@rs2 ~]#echo 2 >   /proc/sys/net/ipv4/conf/lo/arp_announce
[root@rs2 ~]#ifconfig lo:1 10.0.0.100/32

[root@LVS ~]#cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NAME=eth0
BOOTPROTO=static
IPADDR=10.0.0.8
```

```
PREFIX=24
GATEWAY=10.0.0.200
ONBOOT=yes

[root@LVS ~]#ifconfig lo:1 10.0.0.100/32
[root@LVS ~]#ipvsadm -A -t 10.0.0.100:80 -s wrr
[root@LVS ~]#ipvsadm -a -t 10.0.0.100:80 -r 10.0.0.7 -g -w 3
[root@LVS ~]#ipvsadm -a -t 10.0.0.100:80 -r 10.0.0.17  -g
```

## 4.3 LVS-DR模式多网段案例

单网段的DR模式容易暴露后端RS服务器地址信息,可以使用跨网面的DR模型,实现更高的安全性




范例:

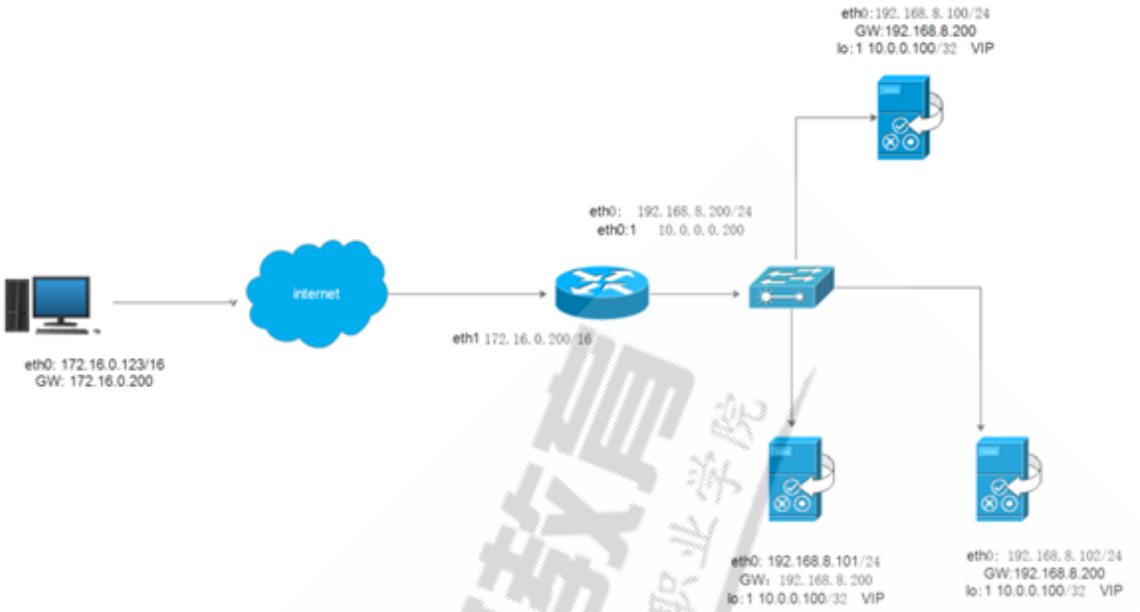
```
#internet主机的网络配置和4.2一样
[root@internet ~]#hostname -I
192.168.10.6

#router的网络配置在4.2基础上添加172.16.0.200/24的地址
[root@router ~]#ip addr add 172.16.0.200/24  dev eth0
[root@router ~]#ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 00:0c:29:ab:8f:2b brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.200/24 brd 10.0.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet 172.16.0.200/24 brd 172.16.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:feab:8f2b/64 scope link tentative
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 00:0c:29:ab:8f:35 brd ff:ff:ff:ff:ff:ff
    inet 192.168.10200/24 brd 192.168.10255 scope global noprefixroute eth1
```

```
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:feab:8f35/64 scope link
       valid_lft forever preferred_lft forever
[root@router ~]#hostname -I
10.0.0.200 172.16.0.200 192.168.10200

#LVS主机的网络配置和4.2一样
[root@lvs ~]#hostname -I
10.0.0.8
[root@lvs ~]#route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.200      0.0.0.0         UG    0      0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 eth0

[root@rs1 ~]#hostname -I
10.0.0.7
[root@rs1 ~]#route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.200      0.0.0.0         UG    100    0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 eth0


#RS主机的网络配置和4.2一样
[root@rs2 ~]#hostname -I
10.0.0.17
[root@rs2 ~]#route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.200      0.0.0.0         UG    100    0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 eth0


#在LVS主机运行的脚本
#注意:VIP如果配置在LO网卡上,必须使用32bit子网掩码,如果VIP绑定在eth0上,可以是其它netmask
[root@lvs ~]#cat lvs_dr_vs.sh
#!/bin/bash
#Author:wangxiaochun
#Date:2017-08-13
vip='172.16.0.100'
iface='lo:1'
mask='255.255.255.255'
port='80'
rs1='10.0.0.7'
rs2='10.0.0.17'
scheduler='wrr'
type='-g'
rpm -q ipvsadm &> /dev/null || yum -y install ipvsadm &> /dev/null

case $1 in
start)
    ifconfig $iface $vip netmask $mask #broadcast $vip up
    iptables -F
    ipvsadm -A -t ${vip}:${port} -s $scheduler
    ipvsadm -a -t ${vip}:${port} -r ${rs1} $type -w 1
    ipvsadm -a -t ${vip}:${port} -r ${rs2} $type -w 1
    echo "The VS Server is Ready!"
```

```
    ;;
stop)
    ipvsadm -C
    ifconfig $iface down
    echo "The VS Server is Canceled!"
    ;;
*)
    echo "Usage: $(basename $0) start|stop"
    exit 1
    ;;
esac

[root@lvs ~]#bash lvs_dr_vs.sh  start
The VS Server is Ready!

#在RS后端服务器运行的脚本
[root@rs1 ~]#cat lvs_dr_rs.sh
#!/bin/bash
#Author:wangxiaochun
#Date:2017-08-13
vip=172.16.0.100
mask='255.255.255.255'
dev=lo:1
rpm -q httpd &> /dev/null || yum -y install httpd &>/dev/null
service httpd start &> /dev/null && echo "The httpd Server is Ready!"
echo "`hostname -I`" > /var/www/html/index.html

case $1 in
start)
    echo 1 > /proc/sys/net/ipv4/conf/all/arp_ignore
    echo 1 > /proc/sys/net/ipv4/conf/lo/arp_ignore
    echo 2 > /proc/sys/net/ipv4/conf/all/arp_announce
    echo 2 > /proc/sys/net/ipv4/conf/lo/arp_announce
    ifconfig $dev $vip netmask $mask #broadcast $vip up
    echo "The RS Server is Ready!"
    ;;
stop)
    ifconfig $dev down
    echo 0 > /proc/sys/net/ipv4/conf/all/arp_ignore
    echo 0 > /proc/sys/net/ipv4/conf/lo/arp_ignore
    echo 0 > /proc/sys/net/ipv4/conf/all/arp_announce
    echo 0 > /proc/sys/net/ipv4/conf/lo/arp_announce
    echo "The RS Server is Canceled!"
    ;;
*)
    echo "Usage: $(basename $0) start|stop"
    exit 1
    ;;
esac

[root@rs1 ~]#bash lvs_dr_rs.sh  start
The RS Server is Ready!

#在RS后端服务器运行的脚本和RS1是一样的
[root@rs2 ~]#bash lvs_dr_rs.sh  start
The RS Server is Ready!

#测试访问
```

```
[root@internet ~]#curl 172.16.0.100
10.0.0.7
[root@internet ~]#curl 172.16.0.100
10.0.0.17
```

**范例2**



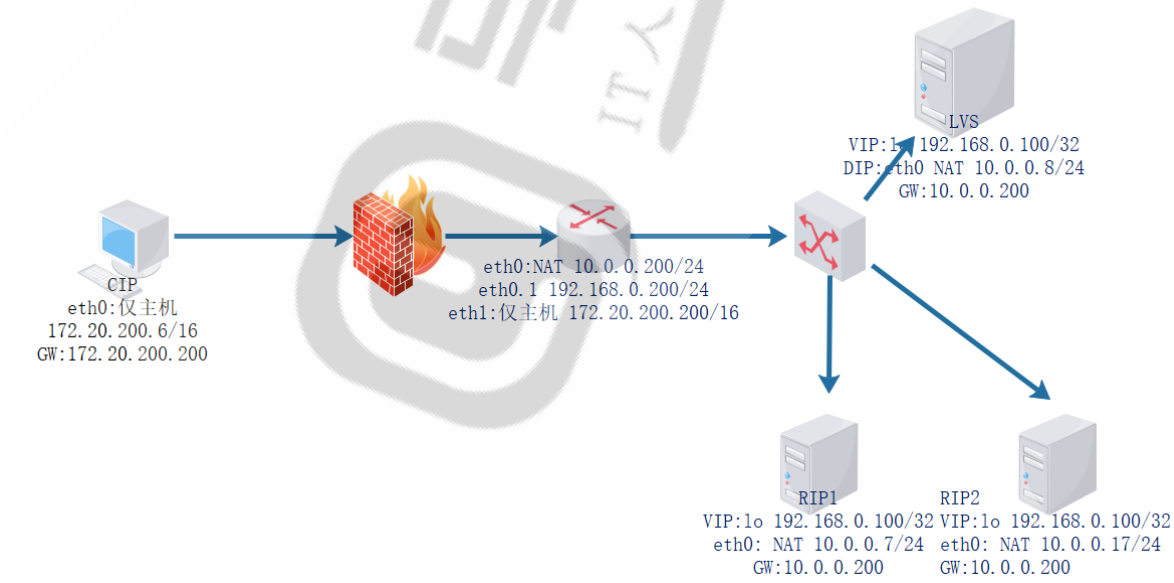

**RS 的配置脚本**

```
#!/bin/bash
vip=10.0.0.100
mask='255.255.255.255'
dev=lo:1
case $1 in
start)
     echo 1 > /proc/sys/net/ipv4/conf/all/arp_ignore
     echo 1 > /proc/sys/net/ipv4/conf/lo/arp_ignore
     echo 2 > /proc/sys/net/ipv4/conf/all/arp_announce
     echo 2 > /proc/sys/net/ipv4/conf/lo/arp_announce
     ifconfig $dev $vip netmask $You can't use 'macro parameter character #' in
math modemask #broadcast $vip up
     #route add -host $vip dev $dev
     ;;
stop)
     ifconfig $dev down
     echo 0 > /proc/sys/net/ipv4/conf/all/arp_ignore
     echo 0 > /proc/sys/net/ipv4/conf/lo/arp_ignore
     echo 0 > /proc/sys/net/ipv4/conf/all/arp_announce
     echo 0 > /proc/sys/net/ipv4/conf/lo/arp_announce
     ;;
*)
     echo "Usage: $(basename $0) start|stop"
     exit 1
     ;;
esac
```

**VS的配置脚本**

```
#!/bin/bash
vip='10.0.0.100'
iface='lo:1'
mask='255.255.255.255'
port='80'
rs1='192.168.8.101'
rs2='192.168.8.102'
scheduler='wrr'
type='-g'
case $1 in
start)
    ifconfig $iface $vip netmask $mask #broadcast $vip up
    iptables -F
    ipvsadm -A -t ${vip}:${port} -s $scheduler
    ipvsadm -a -t ${vip}:${port} -r ${rs1} $type -w 1
    ipvsadm -a -t ${vip}:${port} -r ${rs2} $type -w 1
    ;;
stop)
    ipvsadm -C
    ifconfig $iface down
    ;;
*)
    echo "Usage $(basename $0) start|stop"
    exit 1
esac
```

**范例3: 跨网段DR模型案例**



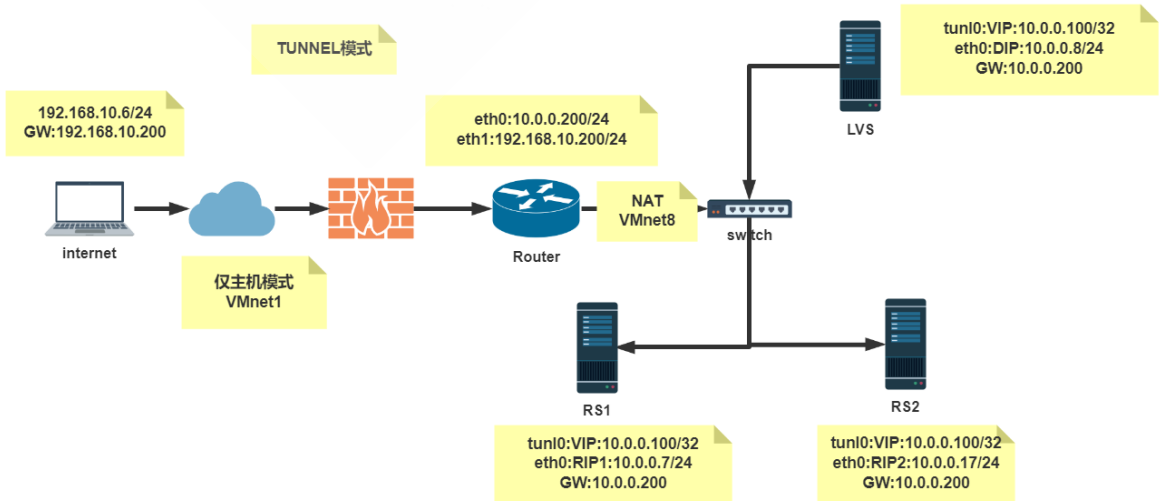

**配置**

```
[root@rs1 ~]#cat lvs_dr_rs.sh
#!/bin/bash
#Author:wangxiaochun
#Date:2017-08-13
vip=192.168.10100
mask='255.255.255.255'
dev=lo:1
#rpm -q httpd &> /dev/null || yum -y install httpd &>/dev/null
```

```
#service httpd start &> /dev/null && echo "The httpd Server is Ready!"
#echo "<h1>`hostname`</h1>" > /var/www/html/index.html

case $1 in
start)
    echo 1 > /proc/sys/net/ipv4/conf/all/arp_ignore
    echo 1 > /proc/sys/net/ipv4/conf/lo/arp_ignore
    echo 2 > /proc/sys/net/ipv4/conf/all/arp_announce
    echo 2 > /proc/sys/net/ipv4/conf/lo/arp_announce
    ifconfig $dev $vip netmask $mask #broadcast $vip up
    #route add -host $vip dev $dev
    echo "The RS Server is Ready!"
    ;;
stop)
    ifconfig $dev down
    echo 0 > /proc/sys/net/ipv4/conf/all/arp_ignore
    echo 0 > /proc/sys/net/ipv4/conf/lo/arp_ignore
    echo 0 > /proc/sys/net/ipv4/conf/all/arp_announce
    echo 0 > /proc/sys/net/ipv4/conf/lo/arp_announce
    echo "The RS Server is Canceled!"
    ;;
*)
    echo "Usage: $(basename $0) start|stop"
    exit 1
    ;;
esac

[root@rs1 ~]#bash lvs_dr_rs.sh  start
[root@rs2 ~]#bash lvs_dr_rs.sh  start


[root@LVS ~]#cat lvs_dr_vs.sh
#!/bin/bash
#Author:wangxiaochun
#Date:2017-08-13
vip='192.168.10100'
iface='lo:1'
mask='255.255.255.255'
port='80'
rs1='10.0.0.7'
rs2='10.0.0.17'
scheduler='wrr'
type='-g'
rpm -q ipvsadm &> /dev/null || yum -y install ipvsadm &> /dev/null

case $1 in
start)
    ifconfig $iface $vip netmask $mask #broadcast $vip up
    iptables -F

    ipvsadm -A -t ${vip}:${port} -s $scheduler
    ipvsadm -a -t ${vip}:${port} -r ${rs1} $type -w 1
    ipvsadm -a -t ${vip}:${port} -r ${rs2} $type -w 1
    echo "The VS Server is Ready!"
    ;;
stop)
    ipvsadm -C
    ifconfig $iface down
```

```
    echo "The VS Server is Canceled!"
    ;;
*)
    echo "Usage: $(basename $0) start|stop"
    exit 1
    ;;
esac

[root@LVS ~]#bash lvs_dr_vs.sh start


[root@Router ~]#nmcli connection modify  eth0 +ipv4.addresses  192.168.10200/24
[root@Router ~]#nmcli connection reload
[root@Router ~]#nmcli connection up eth0
[root@Router ~]#ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 00:0c:29:4d:ef:3e brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.200/24 brd 10.0.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet 192.168.10200/24 brd 192.168.10255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe4d:ef3e/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 00:0c:29:4d:ef:48 brd ff:ff:ff:ff:ff:ff
    inet 172.20.200.200/16 brd 172.20.255.255 scope global noprefixroute eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe4d:ef48/64 scope link
       valid_lft forever preferred_lft forever
```

## 4.4 LVS-TUNNEL隧道模式案例



### 4.4.1 LVS服务器配置

```
[root@centos8 ~]#ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 00:0c:29:44:c3:fe brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.8/24 brd 10.0.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe44:c3fe/64 scope link
       valid_lft forever preferred_lft forever

#开启tunnel网卡并配置VIP
[root@centos8 ~]#ifconfig tunl0 10.0.0.100 netmask 255.255.255.255 up
#或者下面方法,注意设备名必须为tunl0
[root@centos8 ~]#ip addr add 10.0.0.100/32  dev tunl0
[root@centos8 ~]#ip link set up tunl0

#自动加载ipip模块
[root@centos8 ~]#lsmod |grep ipip
ipip                   16384  0
tunnel4                16384  1 ipip
ip_tunnel              28672  1 ipip
[root@centos8 ~]#ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 00:0c:29:44:c3:fe brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.8/24 brd 10.0.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe44:c3fe/64 scope link
       valid_lft forever preferred_lft forever
3: tunl0@NONE: <NOARP,UP,LOWER_UP> mtu 1480 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/ipip 0.0.0.0 brd 0.0.0.0
    inet 10.0.0.100/32 scope global tunl0
       valid_lft forever preferred_lft forever

[root@centos8 ~]#yum -y install ipvsadm
[root@centos8 ~]#ipvsadm -A -t 10.0.0.100:80 -s rr
[root@centos8 ~]#ipvsadm -a -t 10.0.0.100:80 -r 10.0.0.7 -i
[root@centos8 ~]#ipvsadm -a -t 10.0.0.100:80 -r 10.0.0.17 -i
[root@centos8 ~]#ipvsadm -Ln
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
```

```
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  10.0.0.100:80 rr
  -> 10.0.0.7:80                  Tunnel  1      0          0
  -> 10.0.0.17:80                 Tunnel  1      0          0
```

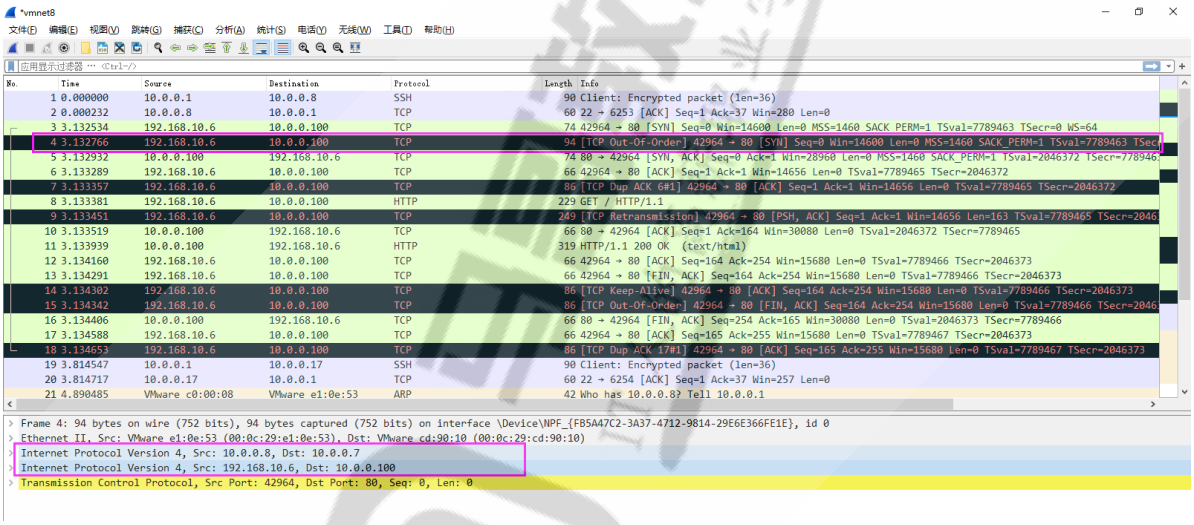
## 4.4.2 RS服务器配置

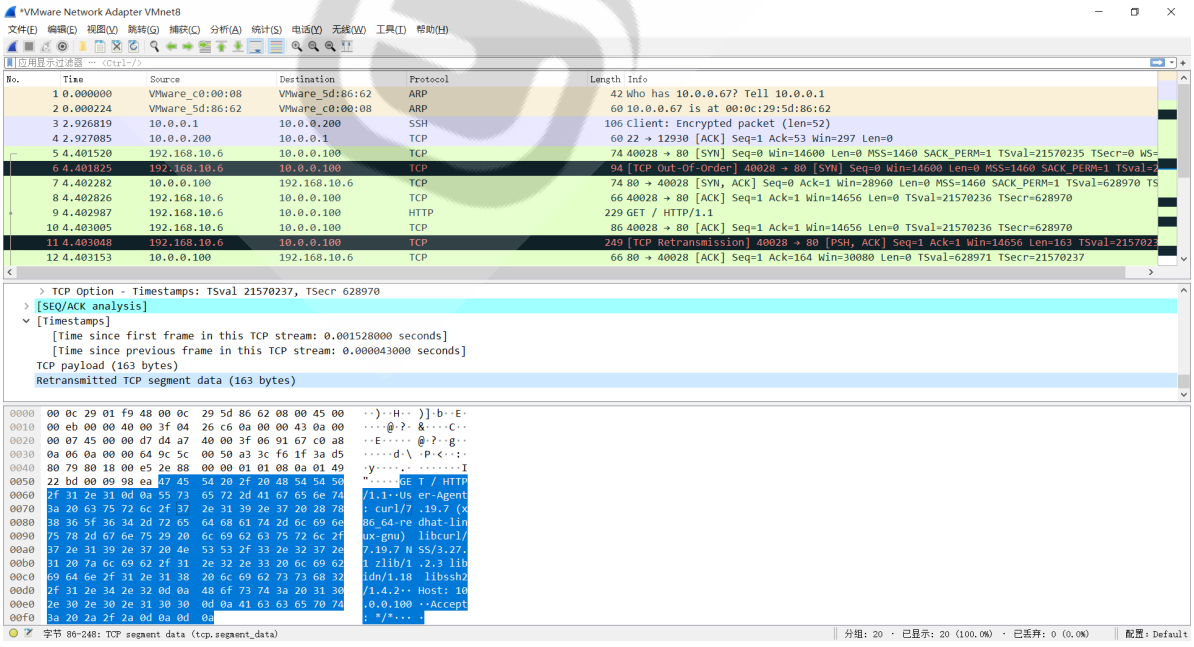
```
#RS服务器配置,RS1和RS2配置相同
[root@rs1 ~]#hostname
rs1.wang.org
[root@rs1 ~]#hostname -I
10.0.0.7

[root@rs1 ~]#route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.200      0.0.0.0         UG    100    0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 eth0

#开启tunnel网卡并配置VIP
[root@rs1 ~]#ifconfig tunl0 10.0.0.100 netmask 255.255.255.255 up
[root@rs1 ~]#ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    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
group default qlen 1000
    link/ether 00:0c:29:01:f9:48 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.7/24 brd 10.0.0.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe01:f948/64 scope link
       valid_lft forever preferred_lft forever
3: tunl0@NONE: <NOARP,UP,LOWER_UP> mtu 1480 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/ipip 0.0.0.0 brd 0.0.0.0
    inet 10.0.0.100/32 scope global tunl0
       valid_lft forever preferred_lft forever

#修改内核参数
[root@rs1 ~]#echo "1" > /proc/sys/net/ipv4/conf/tunl0/arp_ignore
[root@rs1 ~]#echo "2" > /proc/sys/net/ipv4/conf/tunl0/arp_announce
[root@rs1 ~]#echo "1" > /proc/sys/net/ipv4/conf/all/arp_ignore
[root@rs1 ~]#echo "2" > /proc/sys/net/ipv4/conf/all/arp_announce


#以下参数用来控制系统是否开启对数据包源地址的校验。0表示不开启地址校验；1表示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。

#默认值
[root@ubuntu22.04 ~]#cat /proc/sys/net/ipv4/conf/all/rp_filter
2
```

```
[root@centos8 ~]#cat /proc/sys/net/ipv4/conf/all/rp_filter
1

#必须修改内核参数为0
[root@rs1 ~]#echo "0" > /proc/sys/net/ipv4/conf/tunl0/rp_filter
#Rocky可选项，CentOS7必须加
[root@rs1 ~]#echo "0" > /proc/sys/net/ipv4/conf/all/rp_filter

[root@rs1 ~]#yum -y install httpd
[root@rs1 ~]#systemctl enable --now  httpd
[root@rs1 ~]#hostname  > /var/www/html/index.html
```

## 4.4.3 测试

```
[root@internet ~]#while :;do curl 10.0.0.100;sleep 0.3;done
rs2.wang.org
rs1.wang.org
rs2.wang.org
rs1.wang.org
```




# 5 LVS 高可用性实现

**LVS 不可用时：**

Director不可用，整个系统将不可用；SPoF Single Point of Failure

解决方案：高可用，keepalived、heartbeat/corosync

**RS 不可用时：**

某RS不可用时，Director依然会调度请求至此RS

解决方案： 由Director对各RS健康状态进行检查，失败时禁用，成功时启用

常用解决方案：

- keepalived
- heartbeat/corosync
- ldirectord

检测方式：

- 网络层检测，icmp
- 传输层检测，端口探测
- 应用层检测，请求某关键资源

RS全不用时：backup server, sorry server

# 6 常见面试题

- Linux 集群有哪些分类
- 正向代理和反向代理区别
- 四层代理和七层代现的区别
- LVS的工作模式有哪些，有什么特点
- LVS的调度算法
- LVS和Nginx,Haproxy 的区别

千锋教育
IT人的高薪职业学院

千锋教育
IT人的高薪职业学院