LVS

一、LVS四种工作模式原理

LVS 简介

LVS 是 Linux Virtual Server：Linux 虚拟服务器；是一个虚拟的服务器集群【多台机器 LB IP】。

LVS 集群分为三层结构:

负载调度器(load balancer):

位于整个集群系统的最前端，有一台或者多台负载调度器（Director Server）组成，LVS模块就安装在Director Server上，而Director的主要作用类似于一个路由器，它含有完成LVS功能所设定的路由表，通过这些路由表把用户的请求分发给Server Array层的应用服务器（Real Server）上。同时，在Director Server上还要安装对Real Server服务的监控模块Ldirectord，此模块用于监测各个Real Server服务的健康状况。在Real Server不可用时把它从LVS路由表中剔除，恢复时重新加入。

服务器池(server pool):

由一组实际运行应用服务的机器组成，Real Server可以是WEB服务器、MAIL服务器、FTP服务器、DNS服务器、视频服务器中的一个或者多个，每个Real Server之间通过高速的LAN或分布在各地的WAN相连接。在实际的应用中，Director Server也可以同时兼任Real Server的角色。

共享存储(shared stored):

是为所有Real Server提供共享存储空间和内容一致性的存储区域，在物理上，一般有磁盘阵列设备组成，为了提供内容的一致性，一般可以通过NFS网络文件系统共享数 据，但是NFS在繁忙的业务系统中，性能并不是很好，此时可以采用集群文件系统，例如Red hat的GFS文件系统，oracle提供的OCFS2文件系统等。

常用术语:

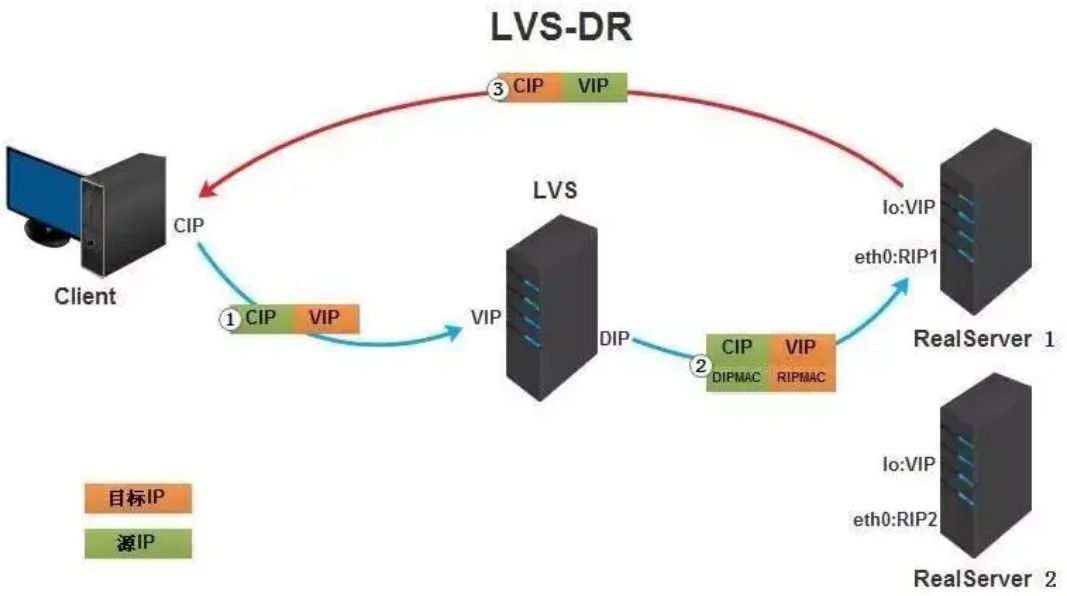
```
# 缩写
```

```
 VS: Virtual Server #虚拟服务，一个抽象的服务，用于最开始接收 web 请求的服务
 Director, Balancer #负载均衡器、分发器
 RS: Real Server # 真正提供服务的服务器
 CIP: Client IP #用户端IP，发起请求的客户端 IP，一般是公网 IP
 VIP: Director Virtual IP #负载均衡器虚拟IP
 DIP: Director IP #负载均衡器IP
 RIP: Real Server IP #真正提供 web 服务的服务器的 IP

 # 详细说明
 VRRP路由器（VRRP Router）：运行VRRP协议的设备，如RouterA和RouterB。

虚拟路由器（Virtual Router）：又称VRRP备份组，由一个Master设备和多个Backup设备组成，被当作一个共享局域网内主机的缺省网关。如RouterA和RouterB共同组成了一个虚拟路由器。

Master路由器（Virtual Router Master）：承担转发报文任务的VRRP设备，如RouterA。

Backup路由器（Virtual Router Backup）：一组没有承担转发任务的VRRP设备，当Master设备出现故障时，它们将通过竞选成为新的Master设备，如RouterB。

vrid：虚拟路由器的标识，如图中RouterA和RouterB组成的虚拟路由器的vrid为1，需手工指定，范围1-255。

虚拟IP地址(Virtual IP Address)：虚拟路由器的IP地址，一个虚拟路由器可以有一个或多个IP地址，由用户配置。如RouterA和RouterB组成的虚拟路由器的虚拟IP地址为10.1.1.254/24。

IP地址拥有者（IP Address Owner）：如果一个VRRP设备将真实的接口IP地址配置为虚拟路由器IP地址，则该设备被称为IP地址拥有者。如果IP地址拥有者是可用的，则它将一直成为Master。

虚拟MAC地址（Virtual MAC Address）：虚拟路由器根据vrid生成的MAC地址。一个虚拟路由器拥有一个虚拟MAC地址，格式为：00-00-5E-00-01-{vrid}。当虚拟路由器回应ARP请求时，使用虚拟MAC地址，而不是接口的真实MAC地址。如RouterA和RouterB组成的虚拟路由器的vrid为1，因此这个VRRP备份组的MAC地址为00-00-5E-00-01-01。
```

（1）直接路由模式（LVS-DR）

互联网使用比较多的一种模式。

DR模式是通过改写请求报文的目标MAC地址，将请求发给真实服务器的，而真实服务器响应后的处理结果直接返回给客户端用户。同TUN模式一样，DR模式可以极大的提高集群系统的伸缩性。而且DR模式没有IP隧道的开销，对集群中的真实服务器也没有必要必须支持IP隧道协议的要求。但是要求调度器LB与真实服务器RS都有一块网卡连接到同一物理网段上，必须在同一个局域网环境。



DR模式：

> 1、通过在调度器LB上修改数据包的目的MAC地址实现转发。注意源地址仍然是CIP，目的地址仍然是VIP地址。
> 2、请求的报文经过调度器，而RS响应处理后的报文无需经过调度器LB，因此并发访问量大时使用效率很高（和NAT模式比）
> 3、因为DR模式是通过MAC地址改写机制实现转发，因此所有RS节点和调度器LB只能在一个局域网里面
> 4、RS主机需要绑定VIP地址在LO接口上，并且需要配置ARP抑制。
> 5、RS节点的默认网关不需要配置成LB，而是直接配置为上级路由的网关，能让RS直接出网就可以。
> 6、由于DR模式的调度器仅做MAC地址的改写，所以调度器LB就不能改写目标端口，那么RS服务器就得使用和VIP相同的端口提供服务

DR模式特点：

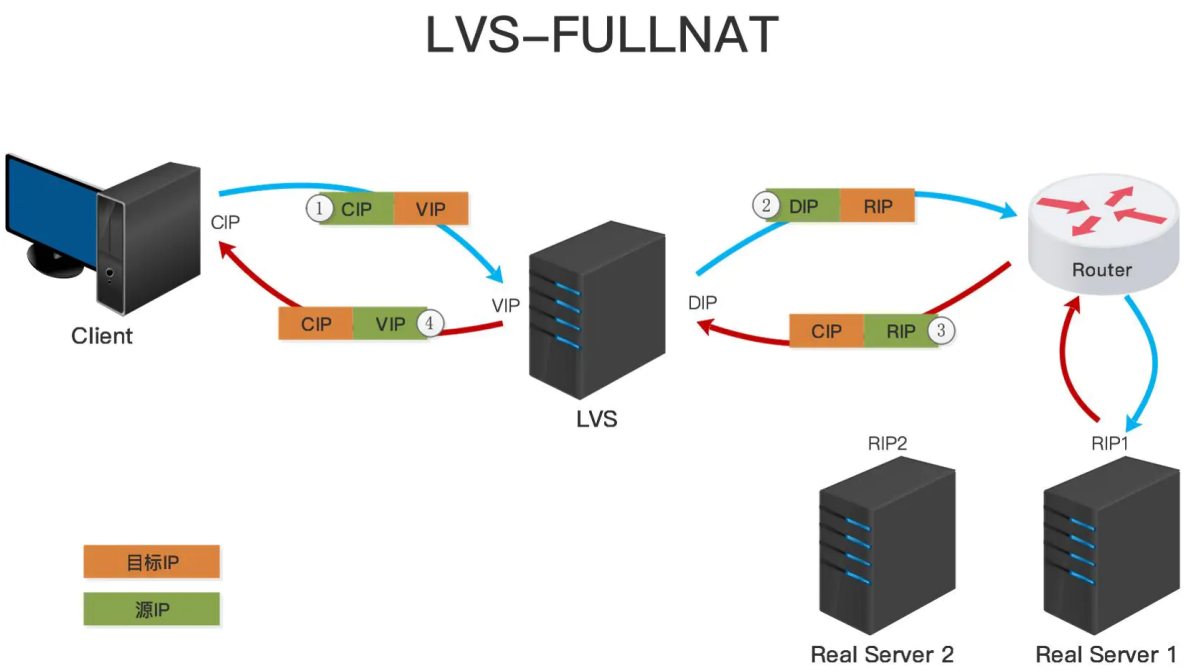
优点：和TUN（隧道模式）一样，负载均衡器也只是分发请求，应答包通过单独的路由方法返回给客户端。与VS-TUN相比，VS-DR这种实现方式不需要隧道结构，因此可以使用大多数操作系统做为物理服务器。

缺点：（不能说缺点，只能说是不足）要求负载均衡器的网卡必须与物理网卡在一个物理段上。

> LVS 的VIP 和 realserver 必须在同一个网段，不然广播后所有的包都会丢掉： 提前确认LVS/硬件LB 是什么模式，是否需要在同一个网段，所有的realserver 都必须绑定VIP的IP地址，否则realserver 收到package后发现dst 不是自己的IP，所有包都会丢掉。

（2）NAT模式（LVS-NAT）

NAT模式是通过网络地址转换的方法来实现调度的。首先调度器(LB)接收到客户的请求数据包时（请求的目的IP为VIP），根据调度算法决定将请求发送给哪个后端的真实服务器（RS）。然后调度就把客户端发送的请求数据包的目标IP地址及端口改成后端真实服务器的IP地址（RIP），这样真实服务器（RS）就能够接收到客户的请求数据包了。真实服务器响应完请求后，查看默认路由（NAT模式下我们需要把RS的默认路由设置为LB服务器。）把响应后的数据包发送给LB，LB再接收到响应包后，把包的源地址改成虚拟地址（VIP）然后发送回给客户端。



NAT模式：

> 1.客户端请求数据，目标IP为VIP。
> 2.请求数据到达LB服务器，LB根据调度算法将目的地址修改为RIP地址及对应端口（此RIP地址是根据调度算法得出的。）并在连接HASH表中记录下这个连接。
> 3.数据包从LB服务器到达RS服务器webserver，然后webserver进行响应。Webserver的网关必须是LB，然后将数据返回给LB服务器。
> 4.收到RS的返回后的数据，根据连接HASH表修改源地址VIP&目标地址CIP，及对应端口80.然后数据就从LB出发到达客户端。
> 5.客户端收到的就只能看到VIP\DIP信息。

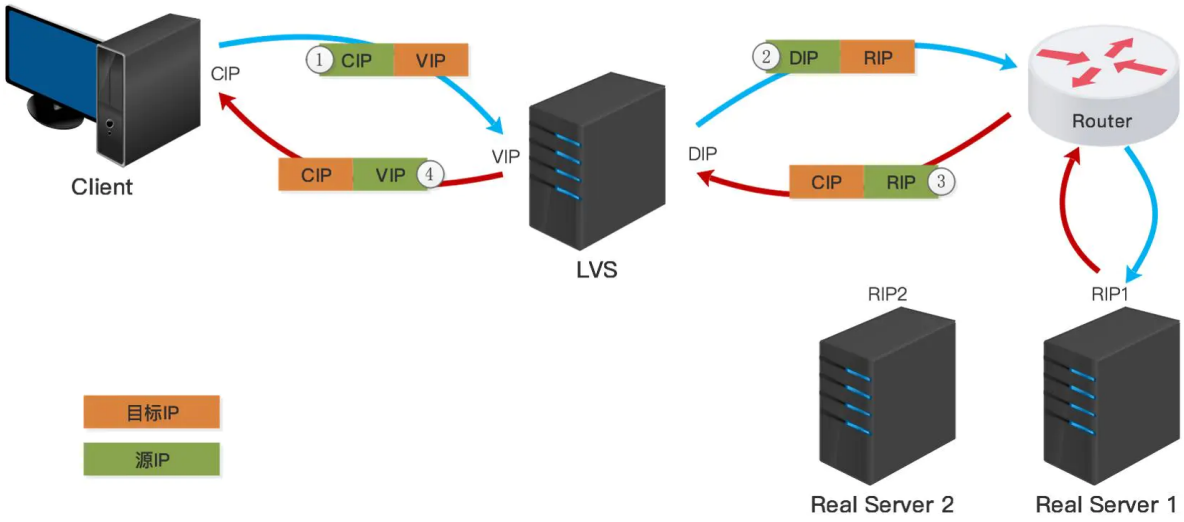
NAT模式特点：

> 1、NAT技术将请求的报文和响应的报文都需要通过LB进行地址改写，因此网站访问量比较大的时候LB负载均衡调度器有比较大的瓶颈，一般要求最多之能10-20台节点
> 2、只需要在LB上配置一个公网IP地址就可以了。
> 3、每台内部的节点服务器的网关地址必须是调度器LB的内网地址。
> 4、NAT模式支持对IP地址和端口进行转换。即用户请求的端口和真实服务器的端口可以不一致。

（3）Full NAT模式（LVS-FullNAT）

客户端对VIP发起请求，Director接过请求发现是请求后端服务。Direcrot对请求报文做full-nat，把源ip改为Dip，把目标ip转换为任意后端RS的rip，然后发往后端，rs接到请求后，进行响应，响应源ip为Rip，目标ip还是DIP，又内部路由路由到Director,Director接到响应报文，进行full-nat。将源地址为VIP，目标地址改为CIP，请求使用DNAT，响应使用SNAT

LVS-FULLNAT



fullnat模式：

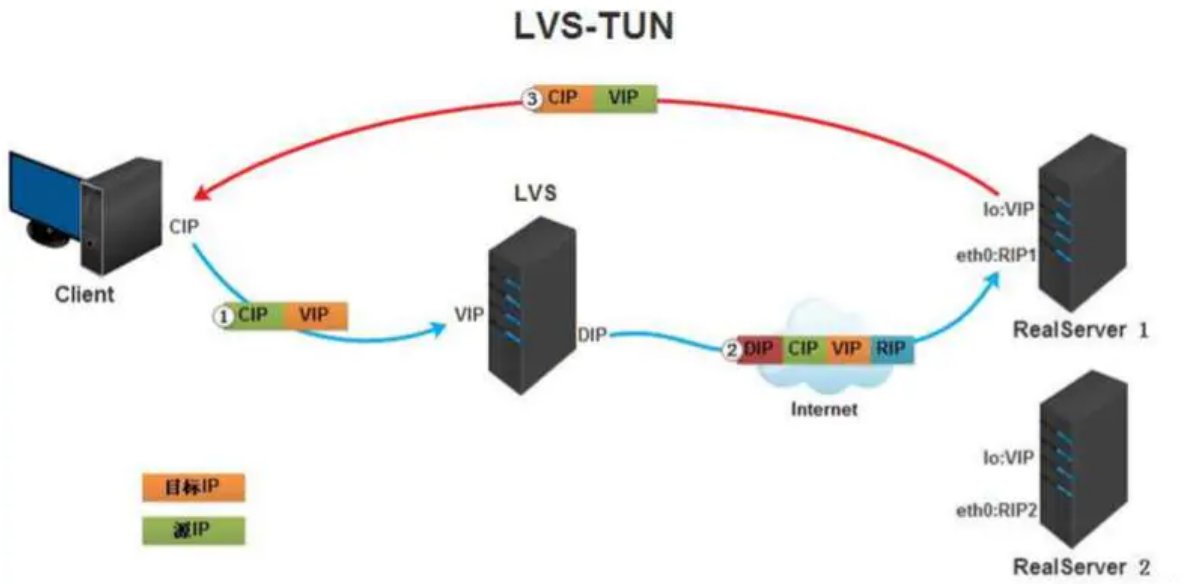
Full NAT模式特点：

> 1.FULL NAT 模式也不需要 LBIP 和realserver ip 在同一个网段；
> 2.full nat 跟nat 相比的优点是：保证RS回包一定能够回到LVS；因为源地址就是LVS==> 不确定
> 3.full nat 因为要更新sorce ip 所以性能正常比nat 模式下降 10%

(4) IP隧道模式（LVS-Tunnel）

采用NAT模式时，由于请求和响应的报文必须通过调度器地址重写，当客户请求越来越多时，调度器处理能力将成为瓶颈。为了解决这个问题，调度器把请求的报文通过IP隧道转发到真实的服务器。真实的服务器将响应处理后的数据直接返回给客户端。这样调度器就只处理请求入站报文，由于一般网络服务应答数据比请求报文大很多，采用VS/TUN模式后，集群系统的最大吞吐量可以提高10倍。

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。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。



LVS-TUN模式：

> 1、客户请求数据包，目标地址VIP发送到LB上。
> 2、LB接收到客户请求包，进行IP Tunnel封装。即在原有的包头加上IP Tunnel的包头。然后发送出去。
> 3、RS节点服务器根据IP Tunnel包头信息（此时就有一种逻辑上的隐形隧道，只有LB和RS之间懂）收到请求包，然后解开IP Tunnel包头信息，得到客户的请求包并进行响应处理。
> 4、响应处理完毕之后，RS服务器使用自己的出公网的线路，将这个响应数据包发送给客户端。源IP地址还是VIP地址。

ip隧道模式特点：

负载均衡器只负责将请求包分发给后端节点服务器，而RS将应答包直接发给用户。所以，减少了负载均衡器的大量数据流动，负载均衡器不再是系统的瓶颈，就能处理很巨大的请求量，这种方式，一台负载均衡器能够为很多RS进行分发。而且跑在公网上就能进行不同地域的分发。

隧道模式的RS节点需要合法IP，这种方式需要所有的服务器支持”IP Tunneling”(IP Encapsulation)协议，服务器可能只局限在部分Linux系统上。

四种模式性能比较：

> 1.因为 DR模式 IP TUNELL 模式 都是在package in 时经过LVS，在package out是直接返回给client，所以二者的性能比NAT 模式高，但IP TUNNEL 因为是 TUNNEL 模式 比较复杂，其性能不如DR模式；

> 2.FULL NAT 模式因为不仅要更换 DST IP 还更换 SOURCE IP 所以性能比NAT 下降10%。
>
> 3.4种模式的性能如下：DR ==> IP TUNNEL ==>NAT ==>FULL NAT

二、VRRP

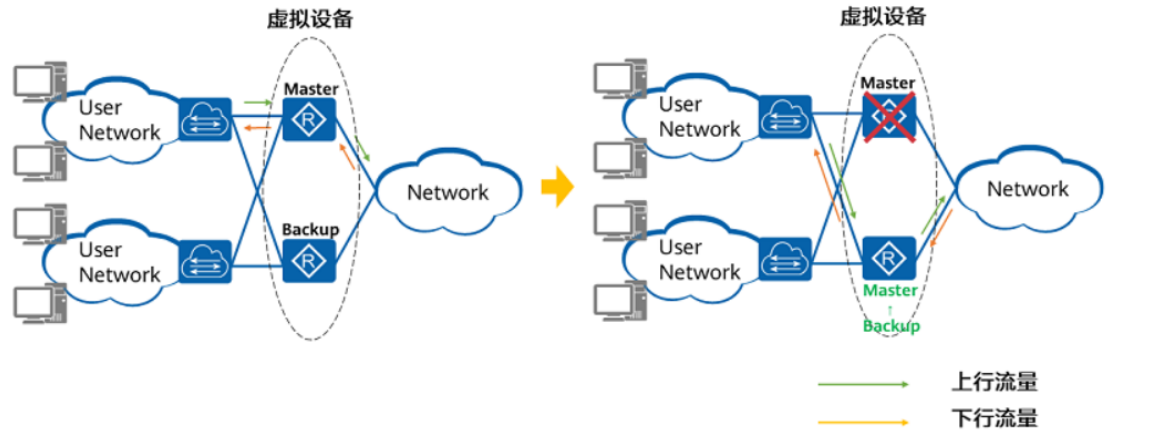
1.VRRP简介

VRRP(虚拟路由冗余协议)，是一种用于提高网络可靠性的容错协议。通过VRRP，可以在主机的下一跳设备出现故障时，及时将业务切换到备份设备，从而保障网络通信的连续性和可靠性。

随着网络的快速普及和相关应用的日益深入，各种增值业务（如IPTV、视频会议等）已经开始广泛部署，基础网络的可靠性日益成为用户关注的焦点，能够保证网络传输不中断对于终端用户非常重要。

现网中的主机使用缺省网关与外部网络联系时，如果Gateway出现故障，与其相连的主机将与外界失去联系，导致业务中断。

VRRP的出现很好地解决了这个问题。VRRP将多台设备组成一个虚拟设备，通过配置虚拟设备的IP地址为缺省网关，实现缺省网关的备份。当网关设备发生故障时，VRRP机制能够选举新的网关设备承担数据流量，从而保障网络的可靠通信。如下图所示，当Master设备故障时，发往缺省网关的流量将由Backup设备进行转发。

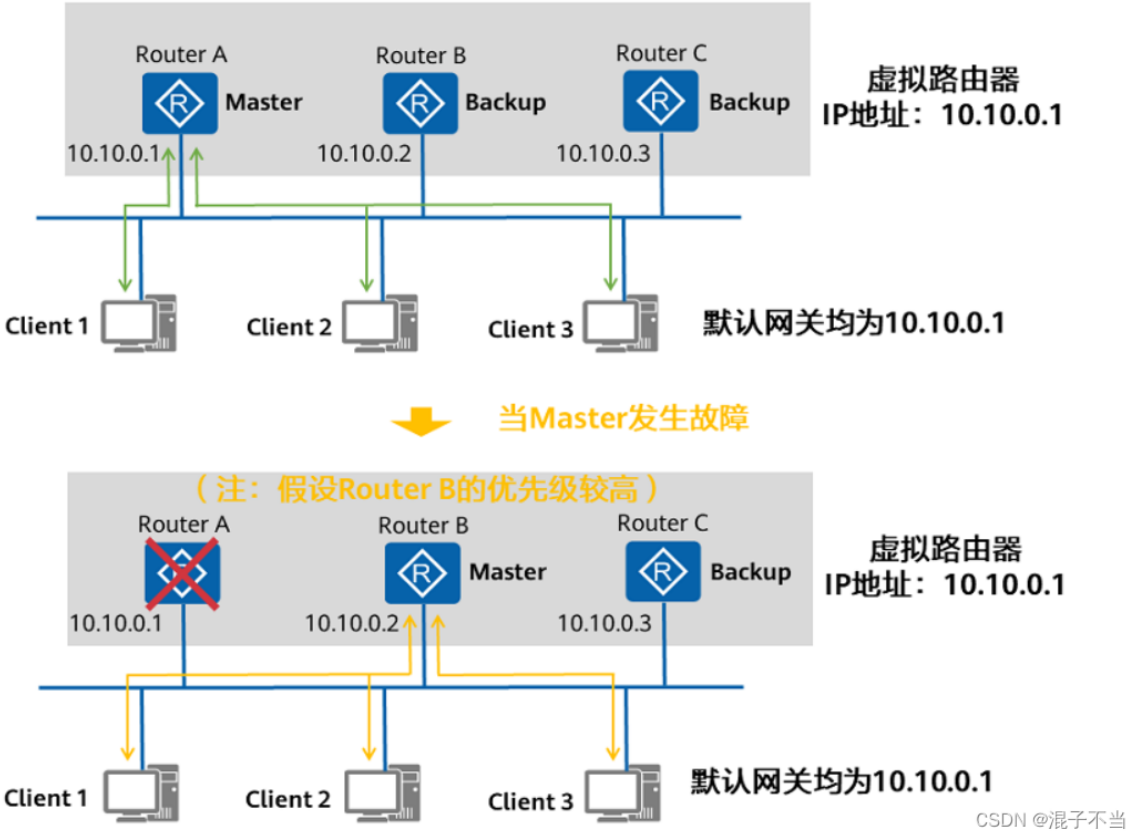


VRRP备份组示意图

2.VRRP工作的三种状态

VRRP协议中定义了三种状态机：初始状态（Initialize）、活动状态（Master）、备份状态（Backup）。其中，只有处于Master状态的设备才可以转发那些发送到虚拟IP地址的报文。

> Initialize： 该状态为VRRP不可用状态，在此状态时设备不会对VRRP通告报文做任何处理。
> 通常设备启动时或设备检测到故障时会进入Initialize状态。
>
> Master当VRRP设备处于Master状态时，它将会承担虚拟路由设备的所有转发工作，并定期向整个虚拟内发送VRRP通告报文。
>
> Backup：当VRRP设备处于Backup状态时，它不会承担虚拟路由设备的转发工作，并定期接受Master设备的VRRP通告报文，判断Master的工作状态是否正常。

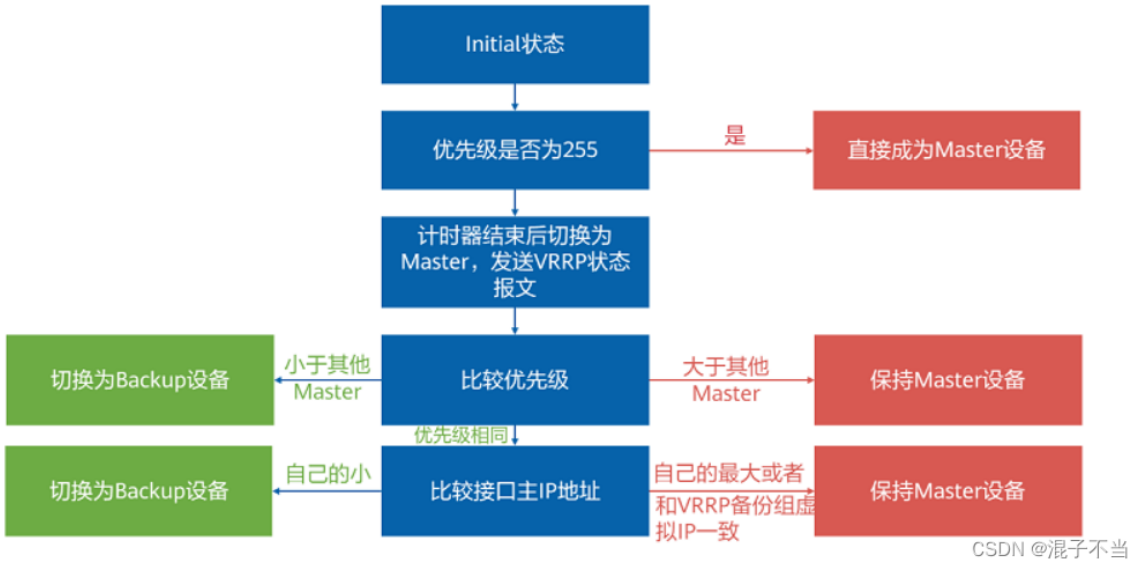


3.VRRP的选举机制

当Master设备出现故障时，路由器B和路由器C会选举出新的Master设备。新的Master设备开始响应对虚拟IP地址的ARP响应，并定期发送VRRP通告报文。

VRRP的详细工作过程如下：

> 1.VRRP备份组中的设备根据优先级选举出Master。Master设备通过发送免费ARP报文，将虚拟MAC地址通知给与它连接的设备或者主机，从而承担报文转发任务。
> 2.Master设备周期性向备份组内所有Backup设备发送VRRP通告报文，通告其配置信息（优先级等）和工作状况。
> 3.如果Master设备出现故障，VRRP备份组中的Backup设备将根据优先级重新选举新的Master。
> 4.VRRP备份组状态切换时，Master设备由一台设备切换为另外一台设备，新的Master设备会立即发送携带虚拟路由器的虚拟MAC地址和虚拟IP地址信息的免费ARP报文，刷新与它连接的设备或者主机的MAC表项，从而把用户流量引到新的Master设备上来，整个过程对用户完全透明。
> 5.原Master设备故障恢复时，若该设备为IP地址拥有者（优先级为255），将直接切换至Master状态。若该设备优先级小于255，将首先切换至Backup状态，且其优先级恢复为故障前配置的优先级。
> 6.Backup设备的优先级高于Master设备时，由Backup设备的工作方式（抢占方式和非抢占方式）决定是否重新选举Master。

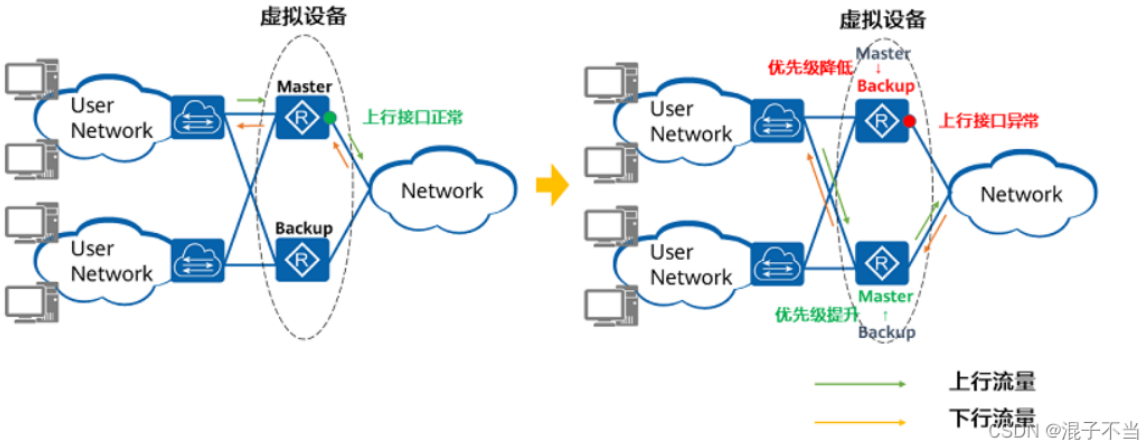


4.VRRP的应用场景

在网络中，VRRP不仅仅在设备故障时触发Master设备的切换，它也能感知某个接口、某条路由的状态。

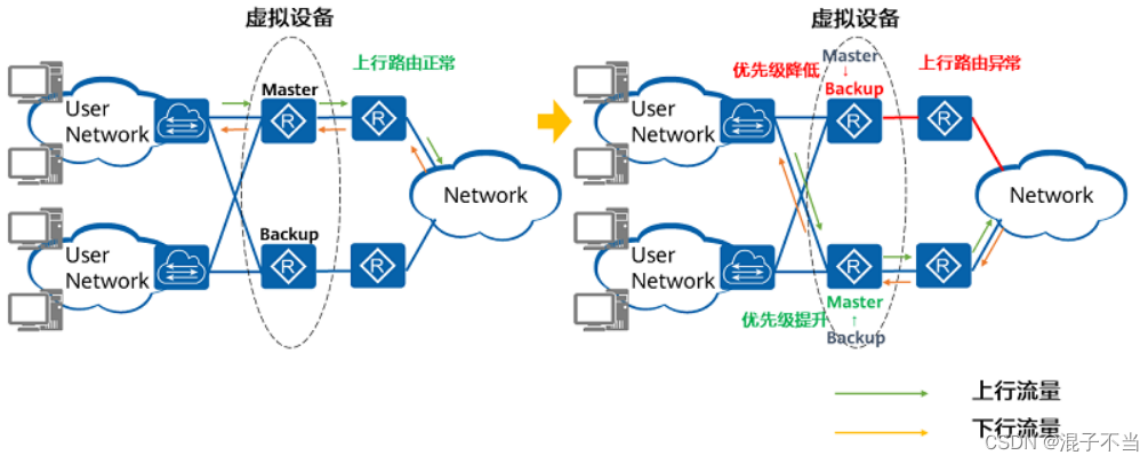
1）与接口状态联动

如下图所示，VRRP可以与上行接口的状态绑定在一起，当承担转发任务的Master设备的上行接口出现异常时，Master设备会降低一定的优先级，当优先级低于Backup设备的优先级时，Backup设备就会切换为Master设备，从而防止因为上行接口的异常导致业务受损。



2）与路由状态联动

如下图所示，VRRP可以与上行路由的状态绑定在一起，当上行路由出现异常时，Master设备可以降低一定的优先级，当优先级低于Backup设备的优先级时，Backup设备就会切换为Master设备，从而防止因为上行路由的异常导致业务受损。



三、实践LVS的NAT模式

1.实验环境

三台服务器，一台作为 director，两台作为 real server，director 有一个外网网卡(192.168.3.121) 和一个内网ip(172.16.0.130)，两个 real server 上只有内网 ip (172.16.0.100) 和 (172.16.0.120)，并且需要把两个 real server 的内网网关设置为 director 的内网 ip(192.168.0.8)

一台Director：
版本：Red Hat Enterprise Linux Server release 7.9
双网卡：
eth1: VIP:192.168.3.129/24(真实生产环境下一定将网关指向运营商的公网IP)
eth2: DIP:172.16.0.130/24(此IP必须和后台的RealSever在同一个网段内)

两台RealServer：
版本：Red Hat Enterprise Linux Server release 7.9
单网卡：
RealServer1：RIP1:172.16.0.100/24(网关必须指向Director的DIP)
RealServer2: RIP2:172.16.0.120/24(网关必须执行Director的DIP)

2.配置Director网卡

```
# 此处我们不指定GATEWAY(真实生产一定要指向SP给的公网IP)
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="static"
IPADDR=172.16.0.130
NETMASK=255.255.255.0
#GRTEWAY=172.16.0.1
DEFROUTE="yes"
DEVICE="ens32"
ONBOOT="yes"
# 重启网卡
ifdown ens32
ifup ens32
# 查询网卡信息
[root@superbox3 network-scripts]# nmcli con show
```

```
NAME          UUID
TYPE      DEVICE
System ens33  c96bc909-188e-ec64-3a96-6a90982b08ad
ethernet  ens33
System ens32  152beb06-47c5-c5e8-95a9-385590654382
ethernet  ens32
有线连接 1    087e6775-a6bc-306f-8aa6-3c4b5dd86ca6
ethernet  --
```

3.配置real server1网卡

```
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="static"
IPADDR=172.16.0.100
NETMASK=255.255.255.0
GRTEWAY=172.16.0.1
DEFROUTE="yes"
DEVICE="ens35"
ONBOOT="yes"
# 重启网卡
ifdown ens35
ifup ens35
# 查询网卡信息
[root@superbox ~]# nmcli con show
NAME          UUID
TYPE      DEVICE
System ens35  1777ed92-ff58-7956-b8b3-ed928f82e0c8
ethernet  ens35
System ens33  c96bc909-188e-ec64-3a96-6a90982b08ad
ethernet  ens33
有线连接 1    fd0547f4-3db1-3d9f-be10-9d4e94f6a822
ethernet  --
```

```
[root@superbox ~]# cat /etc/sysconfig/network-scripts/ifcfg-ens35
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="static"
IPADDR=172.16.0.100
NETMASK=255.255.255.0
GRTEWAY=172.16.0.1
DEFROUTE="yes"
DEVICE="ens35"
ONBOOT="yes"
```

4.配置real server2网卡

```
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="static"
IPADDR=172.16.0.120
NETMASK=255.255.255.0
GRTEWAY=172.16.0.1
DEFROUTE="yes"
DEVICE="ens35"
ONBOOT="yes"
# 重启网卡
ifdown ens35
ifup ens35
# 查询网卡信息
[root@superbox2 ~]# nmcli con show
NAME           UUID
TYPE      DEVICE
System ens35  1777ed92-ff58-7956-b8b3-ed928f82e0c8
ethernet  ens35
System ens33  c96bc909-188e-ec64-3a96-6a90982b08ad
ethernet  ens33
有线连接 1    164c2207-8e8e-389a-963d-a41cd4bc2f58
ethernet  --
```

```
[root@superbox2 ~]# cat /etc/sysconfig/network-scripts/ifcfg-ens35
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="static"
IPADDR=172.16.0.120
NETMASK=255.255.255.0
GRTEWAY=172.16.0.1
DEFROUTE="yes"
DEVICE="ens35"
ONBOOT="yes"
```

5.两个 real server 上都安装 nginx 服务

```
yum install -y nginx
nginx
```

```
[root@superbox ~]# ps -ef | grep nginx
root       3245      1  0 15:58 ?        00:00:00 nginx: master process nginx
nginx      3246   3245  0 15:58 ?        00:00:02 nginx: worker process
nginx      3247   3245  0 15:58 ?        00:00:01 nginx: worker process
nginx      3248   3245  0 15:58 ?        00:00:01 nginx: worker process
nginx      3249   3245  0 15:58 ?        00:00:02 nginx: worker process
nginx      3250   3245  0 15:58 ?        00:00:01 nginx: worker process
nginx      3251   3245  0 15:58 ?        00:00:02 nginx: worker process
nginx      3252   3245  0 15:58 ?        00:00:02 nginx: worker process
nginx      3253   3245  0 15:58 ?        00:00:02 nginx: worker process
root       4093   3174  0 16:15 pts/0    00:00:00 grep --color=auto nginx
```

```
yum install -y nginx
nginx
```

```
[root@superbox2 ~]# ps -ef | grep nginx
root       2691      1  0 16:00 ?        00:00:00 nginx: master process nginx
nginx      2692   2691  0 16:00 ?        00:00:00 nginx: worker process
root       3457   2501  0 16:15 pts/0    00:00:00 grep --color=auto nginx
```

2）Director 上安装 ipvsadm

```
 yum install -y ipvsadm
```

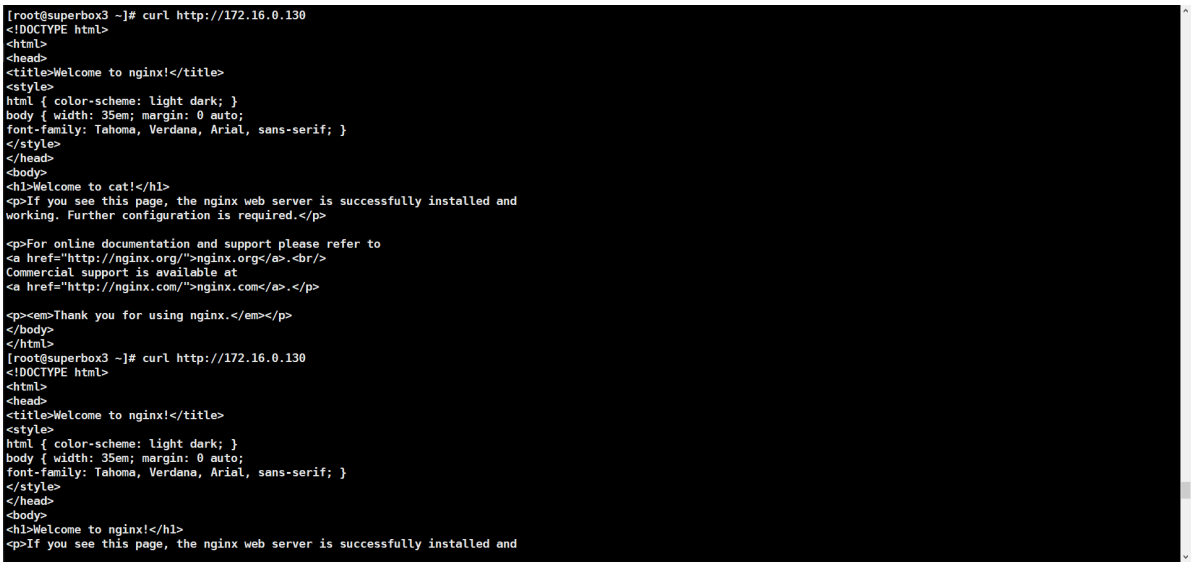
- LVS-NAT模型的实现方式和iptables的DNAT相似，所以Director节点不能和iptables同时使用，那么会有冲突，这就是我们后面为什么要将iptables的规则清空的目的之一。

```
# 清空本地防火墙策略
iptables -F
```

3.

```
# director服务器上开启路由转发功能:
echo 1 > /proc/sys/net/ipv4/ip_forward
# 关闭 icmp 的重定向
echo 0 > /proc/sys/net/ipv4/conf/all/send_redirects
echo 0 > /proc/sys/net/ipv4/conf/default/send_redirects
echo 0 > /proc/sys/net/ipv4/conf/ens32/send_redirects
echo 0 > /proc/sys/net/ipv4/conf/ens33/send_redirects
# director设置 nat 防火墙
iptables -t nat -F
```

```
iptables -t nat -X
iptables -t nat -A POSTROUTING -s 172.16.0.0/24 -j MASQUERADE
# director设置 ipvsadm
ipvsadm -A -t 172.16.0.130:80 -s rr
ipvsadm -a -t 172.16.0.130:80 -r 172.16.0.100:80 -m -w 1
ipvsadm -a -t 172.16.0.130:80 -r 172.16.0.120:80 -m -w 1
ipvsadm -L -n


IPVSADM='/sbin/ipvsadm'
$IPVSADM -C
$IPVSADM -A -t 192.168.3.121:80 -s wrr
$IPVSADM -a -t 192.168.3.121:80 -r 172.16.0.100:80 -m -w 1
$IPVSADM -a -t 192.168.3.121:80 -r 172.16.0.120:80 -m -w 1
# 查看ipvsadm设置的规则
ipvsadm -ln
#
route add default gw 172.16.0.130
```



四、实践LVS的DR模式

1.环境准备

服务器	服务器IP	服务器说明
Director	**192.168.3.130**	**调度器和后端服务器通信的ip**
Real server1	**192.168.3.128**	**真正提供服务的后端服务器1**
Real server2	**192.168.3.120**	**真正提供服务的后端服务器2**

DR模式中的名词解释	
DS	调度器，lvs的前端设备
RS	真正提供服务的后端服务器
RIP	后端服务器的ip地址
DIP	调度器和后端服务器通信的ip
源IP	CIP（客户端的IP）
目的IP	VIP（设置的统一入口），对外公布的ip，客户请求进来的ip
源MAC地址	DS调度器的MAC地址
目的MAC地址	RS真正服务器的MAC地址

Director节点下载安装ipvsadm

```
yum install -y ipvsadm
```

Real server1/2节点下载nginx

```
yum install -y nginx
```

2.Director

1.director服务器上开启路由转发功能:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

2.Director添加虚拟网卡

```
ifconfig ens33:0 down
ifconfig ens33:0 192.168.3.110 broadcast 192.168.3.110 netmask 255.255.255.255 up
route add -host 192.168.3.110 dev ens33:0
```

3.清除防火墙规则

```
iptables -t nat -F
```

4.

```
ipvsadm -C
ipvsadm -A -t 192.168.3.110:80 -s wrr
ipvsadm -a -t 192.168.3.110:80 -r 192.168.3.128:80 -g -w 3
ipvsadm -a -t 192.168.3.110:80 -r 192.168.3.120:80 -g -w 1
```

5.配置Real server1

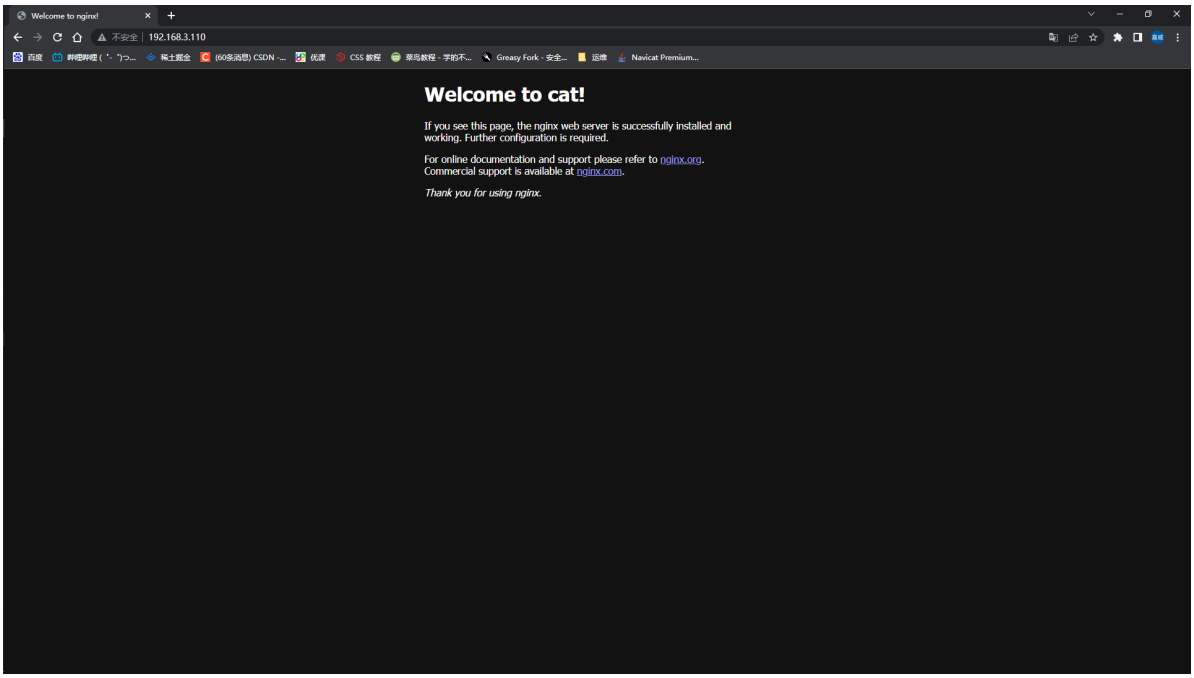
```
# 添加虚拟网卡和路由
ifconfig lo:0 192.168.3.110 broadcast 192.168.3.110 netmask 255.255.255.255 up
route add -host 192.168.3.110 lo:0
# 关闭arp应答
echo "1" >/proc/sys/net/ipv4/conf/lo/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/lo/arp_announce
echo "1" >/proc/sys/net/ipv4/conf/all/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/all/arp_announce
```

6.配置Real server2

```
# 添加虚拟网卡和路由
ifconfig lo:0 192.168.3.110 broadcast 192.168.3.110 netmask 255.255.255.255 up
route add -host 192.168.3.110 lo:0
# 关闭arp应答
echo "1" >/proc/sys/net/ipv4/conf/lo/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/lo/arp_announce
echo "1" >/proc/sys/net/ipv4/conf/all/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/all/arp_announce
```

7.网页测试

```
# 网页输入 192.168.3.110
```



五、实验LVS+keepalived

LVS可以实现负载均衡，但是不能够进行健康检查，比如一个rs出现故障，LVS 仍然会把请求转发给故障
的rs服务器，这样就会导致请求的无效性。keepalived 软件可以进行健康检查，而且能同时实现 LVS 的
高可用性，解决 LVS 单点故障的问题。

环境准备：

服务器	服务器IP	服务器说明
Keepalived1 + lvs1(Director1)	192.168.3.130	first master
Keepalived2 + lvs2(Director2)	192.168.3.94	second master
Real server1	192.168.3.128	RS
Real server2	192.168.3.120	RS

Keepalived1/2节点下载keepalived：

```
yum install ipvsadm keepalived -y
ifconfig ens33:0 down
```

Real server1/2节点下载nginx

```
yum install epel-release -y
yum install nginx -y
```

1.配置Real server节点

```
# 配置Real server1
# 添加虚拟网卡和路由
ifconfig lo:0 192.168.3.110 broadcast 192.168.3.110 netmask 255.255.255.255 up
route add -host 192.168.3.110 lo:0
# 关闭arp应答
echo "1" >/proc/sys/net/ipv4/conf/lo/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/lo/arp_announce
echo "1" >/proc/sys/net/ipv4/conf/all/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/all/arp_announce
# 配置Real server2
# 添加虚拟网卡和路由
ifconfig lo:0 192.168.3.110 broadcast 192.168.3.110 netmask 255.255.255.255 up
```

```
route add -host 192.168.3.110 lo:0
# 关闭arp应答
echo "1" >/proc/sys/net/ipv4/conf/lo/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/lo/arp_announce
echo "1" >/proc/sys/net/ipv4/conf/all/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/all/arp_announce
```

2.配置keepalived节点

```
# 配置master节点
mv /etc/keepalived/keepalived.conf
/etc/keepalived/keepalived.conf.bak
vim /etc/keepalived/keepalived.conf
vrrp_instance VI_1 {
  state MASTER
  interface ens33
  virtual_router_id 51
  priority 100
  advert_int 1
  authentication {
  auth_type PASS
  auth_pass 1111
  }
  virtual_ipaddress {
  192.168.3.110
  }
}
virtual_server 192.168.3.110 80 {
  delay_loop 6
  lb_algo rr
  lb_kind DR
  persistence_timeout 0
  protocol TCP
  real_server 192.168.3.128 80 {
  weight 1
  TCP_CHECK {
  connect_timeout 10
  nb_get_retry 3
  delay_before_retry 3
```

```
  connect_port 80
  }
}
  real_server 192.168.3.120 80 {
  weight 1
  TCP_CHECK {
  connect_timeout 10
  nb_get_retry 3
  delay_before_retry 3
  connect_port 80
 }
}
}
```

3.将keepalived配置文件从master服务器发送至从服务器

```
# 发送配置文件
ssh root@192.168.3.94 "mv /etc/keepalived/keepalived.conf /etc/keepalived/keepalived.conf.bak"
scp /etc/keepalived/keepalived.conf 192.168.3.94:/etc/keepalived/keepalived.conf
```

4.配置从节点

```
vim /etc/keepalived/keepalived.conf
vrrp_instance VI_1 {
  state MASTER
  interface ens33
  virtual_router_id 51
  priority 90
  advert_int 1
  authentication {
  auth_type PASS
  auth_pass 1111
  }
  virtual_ipaddress {
  192.168.3.110
  }
}
```

```
virtual_server 192.168.3.110 80 {
  delay_loop 6
  lb_algo rr
  lb_kind DR
  persistence_timeout 0
  protocol TCP
  real_server 192.168.3.128 80 {
  weight 1
  TCP_CHECK {
  connect_timeout 10
  nb_get_retry 3
  delay_before_retry 3
  connect_port 80
  }
}
  real_server 192.168.3.120 80 {
  weight 1
  TCP_CHECK {
  connect_timeout 10
  nb_get_retry 3
  delay_before_retry 3
  connect_port 80
 }
 }
}
```

5.keepalived的2个节点执行如下命令，开启转发功能

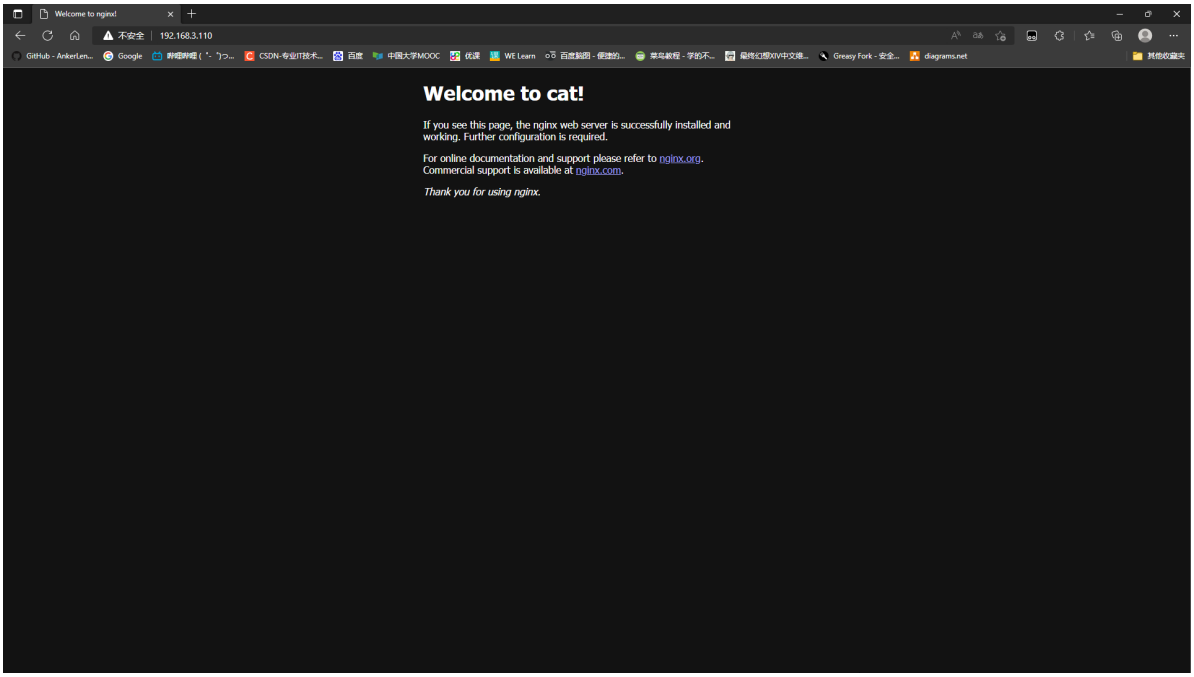
```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

6.启动keepalive

```
systemctl start keepalived
```

7.网页测试

```
输入虚拟主机IP:192.168.3.110
```



8.关闭master节点keepalived

```
systemctl stop keepalived
```

```
[root@superbox3 ~]# systemctl stop keepalived
[root@superbox3 ~]# 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: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:f4:a2:c7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.3.130/24 brd 192.168.3.255 scope global noprefixroute dynamic ens33
       valid_lft 82057sec preferred_lft 82057sec
    inet6 fe80::1030:2240:890b:af92/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: ens32: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:f4:a2:bd brd ff:ff:ff:ff:ff:ff
    inet 172.16.0.130/24 brd 172.16.0.255 scope global noprefixroute ens32
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fef4:a2bd/64 scope link
       valid_lft forever preferred_lft forever
```

查看从服务器是否出现虚拟主机192.168.3.110：

```
ip a
```

```
[root@superbox ~]# 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: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:50:56:3a:01:22 brd ff:ff:ff:ff:ff:ff
    inet 192.168.3.94/24 brd 192.168.3.255 scope global noprefixroute dynamic ens33
       valid_lft 82826sec preferred_lft 82826sec
    inet 192.168.3.110/32 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:fe3a:122/64 scope link
       valid_lft forever preferred_lft forever
```

五、Keepalived高可用（基于VRRP协议）

1.配置抢占式keepalived高可用架构

在Master的keepalived中调用脚本，抢占式，仅需在master配置即可。（注意，如果配置为非抢占式，那么需要两台服务器都使用该脚本）

环境准备：

服务器	服务器IP	服务器说明
Keepalived1 + lvs1(Director1)	192.168.3.128	master
Keepalived2 + lvs2(Director2)	192.168.3.120	backup
Real server1	192.168.3.194	RS
Real server2	192.168.3.130	RS

Keepalived节点安装keepalived：

```
# master服务器
yum install keepalived -y
# backup服务器
yum install keepalived -y
```

2.配置master的Keepalived

```
mv /etc/keepalived/keepalived.conf
/etc/keepalived/keepalived.conf.bak
vim /etc/keepalived/keepalived.conf
#全局配置
global_defs {
#表示身份->名称
    router_id LVS_01
}
#1.每5秒执行一次脚本，脚本执行内容不能超过5秒,否则会被中断再次重新运行脚本
vrrp_script check_web {
   script "/data/scripts/check_web.sh"
   interval 5
}
vrrp_instance VI_1 {
```

```
#    nopreempt
#标识角色状态
    state MASTER
#网卡绑定接口
    interface ens33
#虚拟路由id
    virtual_router_id 50
#优先级
    priority 150
#监测间隔时间
    advert_int 1
#认证
    authentication {
#明文认证
        auth_type PASS
#明文密码
        auth_pass 1111
    }
#虚拟的VIP地址
    virtual_ipaddress {
       192.168.3.119
    }

    #2.调用并运行该脚本
    track_script {
        check_web
    }
}
##################################### 配置监控脚本
#####################################################
mkdir -R  /data/scripts/
cat > /data/scripts/check_web.sh << 'eof'
#!/bin/sh
nginxpid=$(ps -C nginx --no-header|wc -l)
#1.判断Nginx是否存活,如果不存活则尝试启动Nginx
if [ $nginxpid -eq 0 ];then
    nginx
    sleep 3
#2.等待3秒后再次获取一次Nginx状态
nginxpid=$(ps -C nginx --no-header|wc -l)
```

```
#3.再次进行判断， 如Nginx还不存活则停止Keepalived,让地址进行漂移,并退出脚本
if [ $nginxpid -eq 0 ];then
    systemctl stop keepalived
fi
fi
eof
# 给脚本增加执行权限
chmod +x /data/scripts/check_web.sh
```

3.配置Backup的Keepalived

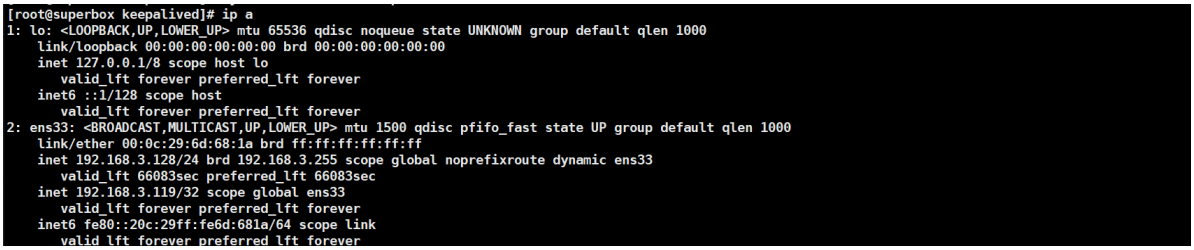
```
mv /etc/keepalived/keepalived.conf /etc/keepalived/keepalived.conf.bak
vim /etc/keepalived/keepalived.conf
global_defs {
         router_id LVS_02
}
vrrp_instance VI_2 {
    nopreempt
    state BACKUP
    interface ens33
    virtual_router_id 51
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
       192.168.3.119
}
}
```

4.启动keepalived

```
# master服务器
systemctl enable keepalived
systemctl restart keepalived
# backup服务器
systemctl enable keepalived
systemctl restart keepalived
```

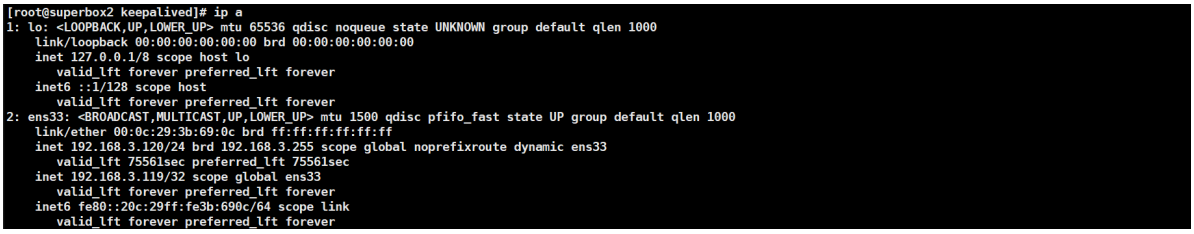
5.查看master主机VIP是否生效

```
ip a
```



6.停掉master主机Keepalived服务后查看backup服务器

```
# master执行
systemctl stop keepalived
# backup服务器执行
ip a
# 出现VIP的表示搭建成功
```



2.配置非抢占式keepalived

1、两个节点的state都必须配置为BACKUP。

2、两个节点都必须加上配置 nopreempt。

3、其中一个节点的优先级必须要高于另外一个节点的优先级。

两台服务器都角色状态启用nopreempt后，必须修改角色状态统一为BACKUP，唯一的区分就是优先级。

环境准备：

服务器	服务器IP	服务器说明
Keepalived1 + lvs1(Director1)	192.168.3.128	backup1
Keepalived2 + lvs2(Director2)	192.168.3.120	backup2

1）Keepalived节点安装keepalived：

```
# master服务器
yum install keepalived -y
# backup服务器
yum install keepalived -y
```

2）配置backup1的Keepalived

```
mv /etc/keepalived/keepalived.conf
/etc/keepalived/keepalived.conf.bak
vim /etc/keepalived/keepalived.conf
#全局配置
global_defs {
#表示身份->名称
    router_id LVS_01
}
#1.每5秒执行一次脚本，脚本执行内容不能超过5秒,否则会被中断再次重新运行脚本
vrrp_script check_web {
   script "/data/scripts/check_web.sh"
   interval 5
}
vrrp_instance VI_1 {
#    nopreempt
#标识角色状态
    state MASTER
#网卡绑定接口
    interface ens33
#虚拟路由id
    virtual_router_id 50
#优先级
```

```
    priority 150
#监测间隔时间
    advert_int 1
#认证
    authentication {
#明文认证
        auth_type PASS
#明文密码
        auth_pass 1111
    }
#虚拟的VIP地址
    virtual_ipaddress {
       192.168.3.119
    }

    #2.调用并运行该脚本
    track_script {
        check_web
    }
}
##################################### 配置监控脚本
#####################################################
mkdir -R  /data/scripts/
cat > /data/scripts/check_web.sh << 'eof'
#!/bin/sh
nginxpid=$(ps -C nginx --no-header|wc -l)
#1.判断Nginx是否存活,如果不存活则尝试启动Nginx
if [ $nginxpid -eq 0 ];then
    nginx
    sleep 3
#2.等待3秒后再次获取一次Nginx状态
nginxpid=$(ps -C nginx --no-header|wc -l)
#3.再次进行判断, 如Nginx还不存活则停止Keepalived,让地址进行漂移,并退出脚本
if [ $nginxpid -eq 0 ];then
    systemctl stop keepalived
fi
fi
eof
# 给脚本增加执行权限
```

```
chmod +x /data/scripts/check_web.sh
```

2）配置backup2的Keepalived

```
mv /etc/keepalived/keepalived.conf
/etc/keepalived/keepalived.conf.bak
vim /etc/keepalived/keepalived.conf
#全局配置
global_defs {
#表示身份->名称
    router_id LVS_02
}
#1.每5秒执行一次脚本, 脚本执行内容不能超过5秒,否则会被中断再次重新运行脚本
vrrp_script check_web {
   script "/data/scripts/check_web.sh"
   interval 5
}
vrrp_instance VI_2 {
#    nopreempt
#标识角色状态
    state BACKUP
#网卡绑定接口
    interface ens33
#虚拟路由id
    virtual_router_id 51
#优先级
    priority 100
#监测间隔时间
    advert_int 1
#认证
    authentication {
#明文认证
        auth_type PASS
#明文密码
        auth_pass 1111
    }
#虚拟的VIP地址
    virtual_ipaddress {
       192.168.3.119
```

```
    }

    #2.调用并运行该脚本
    track_script {
        check_web
    }
}
##################################### 配置监控脚本
#####################################################
mkdir -R  /data/scripts/
cat > /data/scripts/check_web.sh << 'eof'
#!/bin/sh
nginxpid=$(ps -C nginx --no-header|wc -l)
#1.判断Nginx是否存活,如果不存活则尝试启动Nginx
if [ $nginxpid -eq 0 ];then
    nginx
    sleep 3
#2.等待3秒后再次获取一次Nginx状态
nginxpid=$(ps -C nginx --no-header|wc -l)
#3.再次进行判断, 如Nginx还不存活则停止Keepalived,让地址进行漂移,并退出脚本
if [ $nginxpid -eq 0 ];then
    systemctl stop keepalived
fi
fi
eof
# 给脚本增加执行权限
chmod +x /data/scripts/check_web.sh
```

3）重启keepalived服务

```
# master服务器
systemctl enable keepalived
systemctl restart keepalived
# backup服务器
systemctl enable keepalived
systemctl restart keepalived
```