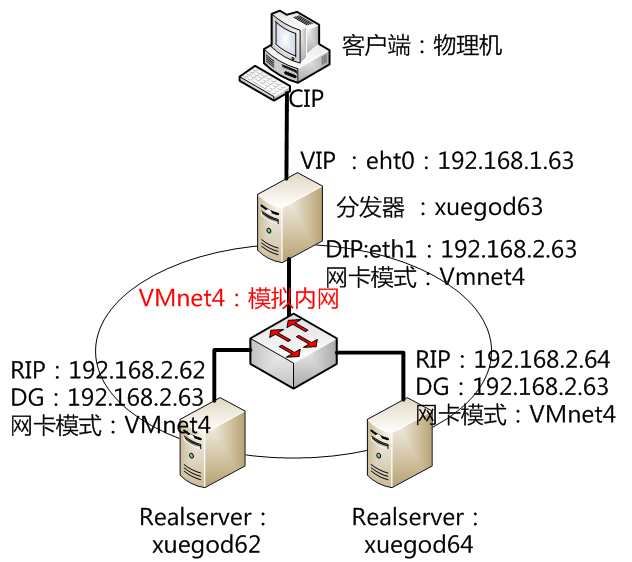
# 一 实战：配置LVS-NAT

****

## 1规划IP

192.168.30.42 VIP

192.168.31.42分发器

192.168.31.43 RIP

192.168.31.44 RIP

## 2 准备工作

### 2.1、iptables -F , 清除规则

### 2.2、/etc/selinux/config,关SELinux

#getenforce

Disabled

### 2.3、打开路由转发功能

**# vim /etc/sysctl.conf**

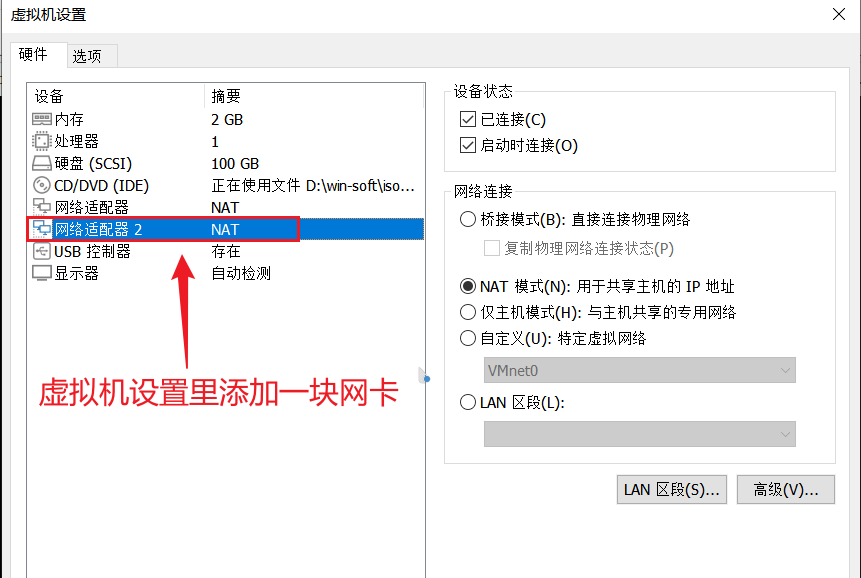
**改：net.ipv4.ip\_forward = 0**

**为：net.ipv4.ip\_forward = 1**

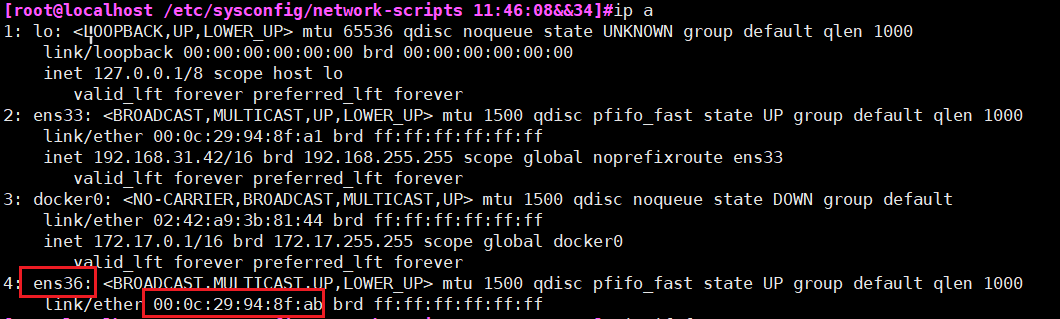
**让配置生效：**

**# sysctl -p**

### 2.4添加网卡



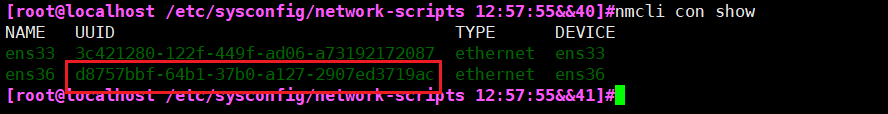
#ip a



从上图可以看出添加的网卡名为ens36

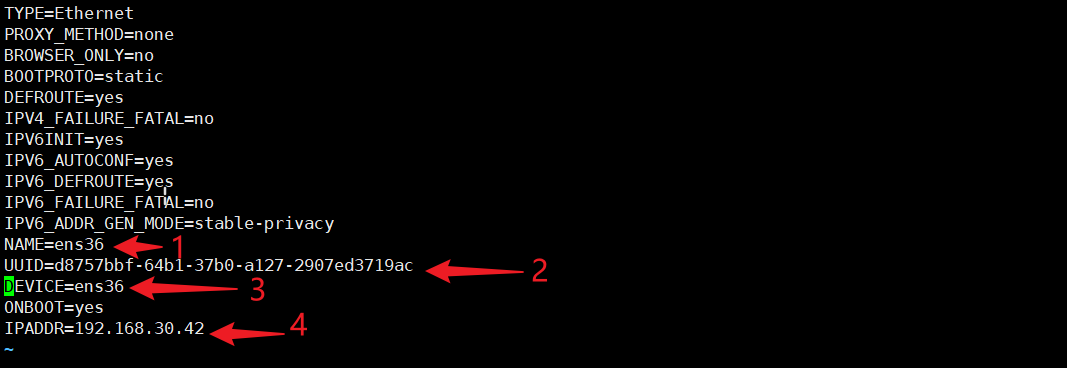
#systemctl start NetworkManager

#nmcli con show  **#获取uuid**



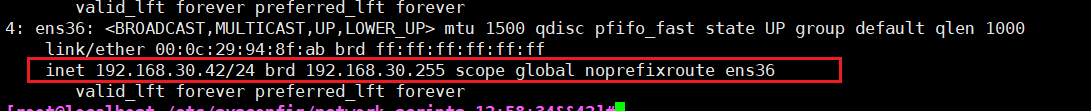
# cd /etc/sysconfig/network-scripts

#cp ifcfg-ens33 ifcfg-ens36 && vim ifcfg-ens36



#systemctl restart network

查看添加成功



<https://blog.51cto.com/u_15067266/4546913>

## 3安装: LVS管理工具： ipvsadmin

说明： 此步骤都在分发器192.168.31.42上操作

**#**yum install ipvsadm -y

### 3.1使用ipvsadm命令设置规则

**#**ipvsadm -A -t 192.168.30.42:80 -s rr

**选项：**

**-A 添加虚拟服务器**

**-t 表示TCP的服务 VIP：PORT**

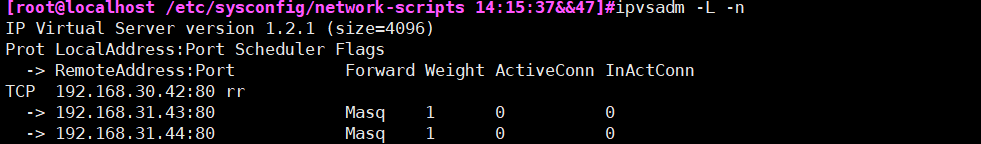
**-s 指定调度算法 rr表示round-robin 轮循**

#ipvsadm -a -t 192.168.30.42:80 -r 192.168.31.43 -m

#ipvsadm -a -t 192.168.30.42:80 -r 192.168.31.44 -m

### 3.2 查看

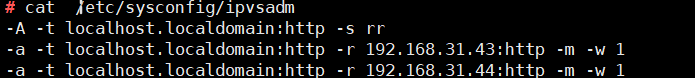
# ipvsadm -L -n



### 3.3保存配置或规则

# ipvsadm --save > /etc/sysconfig/ipvsadm

# cat /etc/sysconfig/ipvsadm



## 4 配置真实服务器

### 4.1 安装nginx

**RealServer：在43和44上面配置：**

**#docker run -d --name ng -p 80:80 nginx**

**#docker run -d --name ng -p 80:80 nginx**

#docker exec -it ng bash -c "echo 43 > /usr/share/nginx/html/index.html"

#docker exec -it ng bash -c "echo 44 > /usr/share/nginx/html/index.html"

### 4.2测试

在43上测试

**# yum install elinks -y**

# 二LVS集群-DR模式

192.168.31.41 RIP

192.168.31.42 DIP

192.168.31.43 RIP1

192.168.31.44 RIP2

## 1、配置IP：

**DIP: ens33 ip地址: 192.168.31.42**

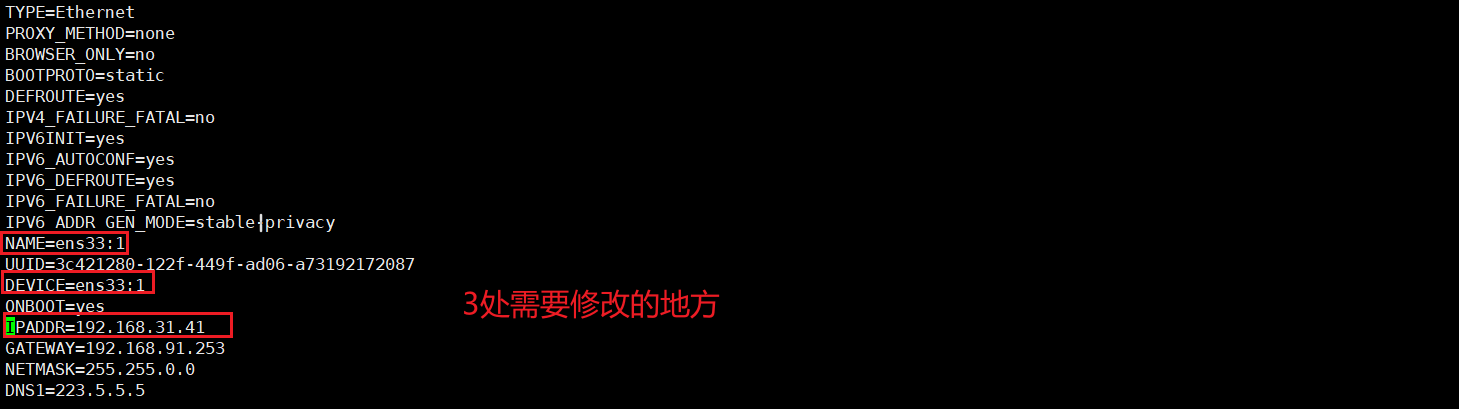
**VIP: ens33:1 ip地址: 192.168.31.41**

**ens33添加IP**

**#cd /etc/sysconfig/network-scripts**

**#cp ifcfg-ens33 ifcfg-ens33:1**

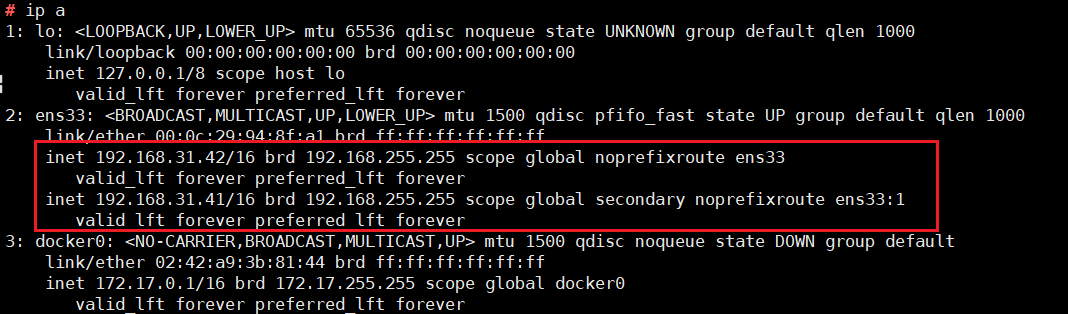
**#vim ifcfg-ens33:1**



|  |
| --- |
| TYPE=Ethernet  PROXY\_METHOD=none  BROWSER\_ONLY=no  BOOTPROTO=static  DEFROUTE=yes  IPV4\_FAILURE\_FATAL=no  IPV6INIT=yes  IPV6\_AUTOCONF=yes  IPV6\_DEFROUTE=yes  IPV6\_FAILURE\_FATAL=no  IPV6\_ADDR\_GEN\_MODE=stable-privacy  NAME=ens33:1  UUID=3c421280-122f-449f-ad06-a73192172087  DEVICE=ens33:1  ONBOOT=yes  IPADDR=192.168.31.41  GATEWAY=192.168.91.253  NETMASK=255.255.0.0  DNS1=223.5.5.5 |

**#systemctl restart network**

**# ip a**



## 2、43,44 安装ipvsadm

#yum install ipvsadm -y

## 3 配置LVS-DR规则：

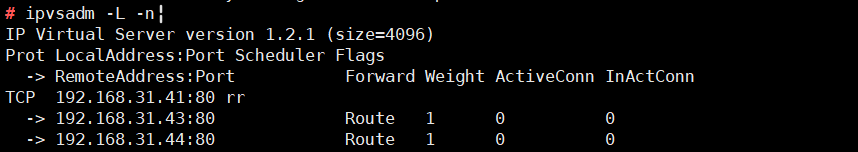
**# ipvsadm -A -t 192.168.31.41:80 -s rr**

**# ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.43 -g**

**# ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.44 -g**

**注： -g 表示DR模式 ， -m 表示IP tun 模式**

**# ipvsadm -L -n**



**注：三个LVS 模式中，只有NAT模式需要开启 路由转发功能。 DR和TUN模式不需要开启。**

## 4、保存配置或规则

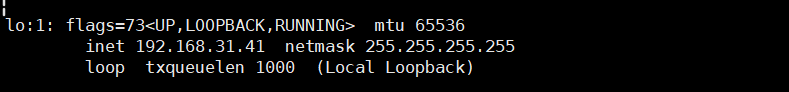
# ipvsadm --save > /etc/sysconfig/ipvsadm

# cat /etc/sysconfig/ipvsadm

## 5 两台RealServer的IP配置

**#yum install -y ipvsadm**

**#ifconfig lo:1 192.168.31.41 netmask 255.255.255.255**



## 6关闭ARP转发（在两台RS上都需要同样的操作）。

**[root@xuegod62 ~]# echo 1 > /proc/sys/net/ipv4/conf/ens33/arp\_ignore**

**[root@xuegod62 ~]# echo 2 > /proc/sys/net/ipv4/conf/ens33/arp\_announce**

**永久生效：(注意realserver的实际link ok的网卡是不是ens33)**

**[root@xuegod62 ~]# vim /etc/sysctl.conf #最后添加**

**net.ipv4.conf.ens33.arp\_ignore = 1**

**net.ipv4.conf.ens33.arp\_announce = 2**

**或：**

**net.ipv4.conf.ens33.arp\_ignore = 1**

**net.ipv4.conf.ens33.arp\_announce = 2**

**net.ipv4.conf.all.arp\_ignore = 1**

**net.ipv4.conf.all.arp\_announce = 2**

**net.ipv4.conf.lo.arp\_ignore = 1**

**net.ipv4.conf.lo.arp\_announce = 2**

**# sysctl -p**

## 7 安装nginx

**RealServer：在43和44上面配置：**

**#docker run -d --name ng -p 80:80 nginx**

**#docker run -d --name ng -p 80:80 nginx**

#docker exec -it ng bash -c "echo 43 > /usr/share/nginx/html/index.html"

#docker exec -it ng bash -c "echo 44 > /usr/share/nginx/html/index.html"

## 8 测试

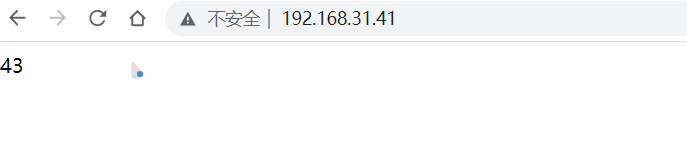
**客户端测试访问vip：**

**http://192.168.31.43**

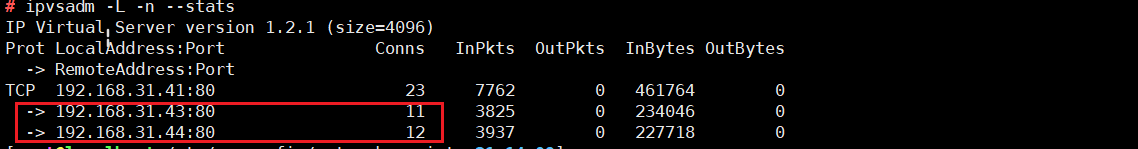
**http://192.168.31.44**

**或物理机上测试：**

**<http://192.168.31.41>**



**# ipvsadm -L -n --stats**



从上图看出有轮询

# 三 测试LVS 其它调度算法

## **3.1 测试LVS 其它调度算法**

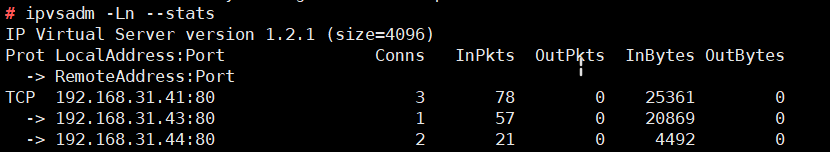
**# ipvsadm -C**

**# ipvsadm -A -t 192.168.31.41:80 -s wrr**

**# ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.43 -g -w 10**

**# ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.44 -g -w 20**

**# ipvsadm -L -n --stats**



# 四 linux下ab网站压力测试命令

## 2.4.1 实战： 同时处理1000个请求。 一次要执行1000

**测试VIP：**

**#ab -n 1000 -c 1000 http://192.168.1.63/index.html**

## 2.4.2 linux下ab网站压力测试命令

**参数：**

**语法： ab -n 数字 -c 数字 http://链接**

**-n requests Number of requests to perform**

**#在测试会话中所执行的请求总个数。默认时，仅执行一个请求  
-c concurrency Number of multiple requests to make  
#一次产生的请求个数。默认是一次一个。**

# 五 配置LVS-IP TUN集群

192.168.31.41 RIP

192.168.31.42 DIP

192.168.31.43 RIP1

192.168.31.44 RIP2

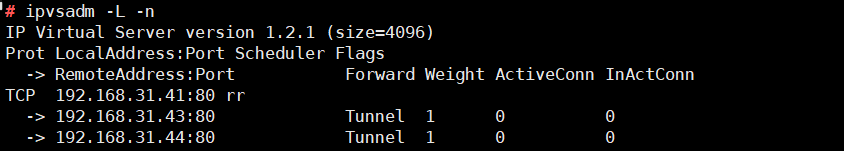
## 5.1 192.168.31.42配置成 Director

# ipvsadm -A -t 192.168.31.41:80 -s rr

# ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.43 -i

# ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.44 -i

# ipvsadm -L -n



### 保存配置或规则

# ipvsadm --save > /etc/sysconfig/ipvsadm

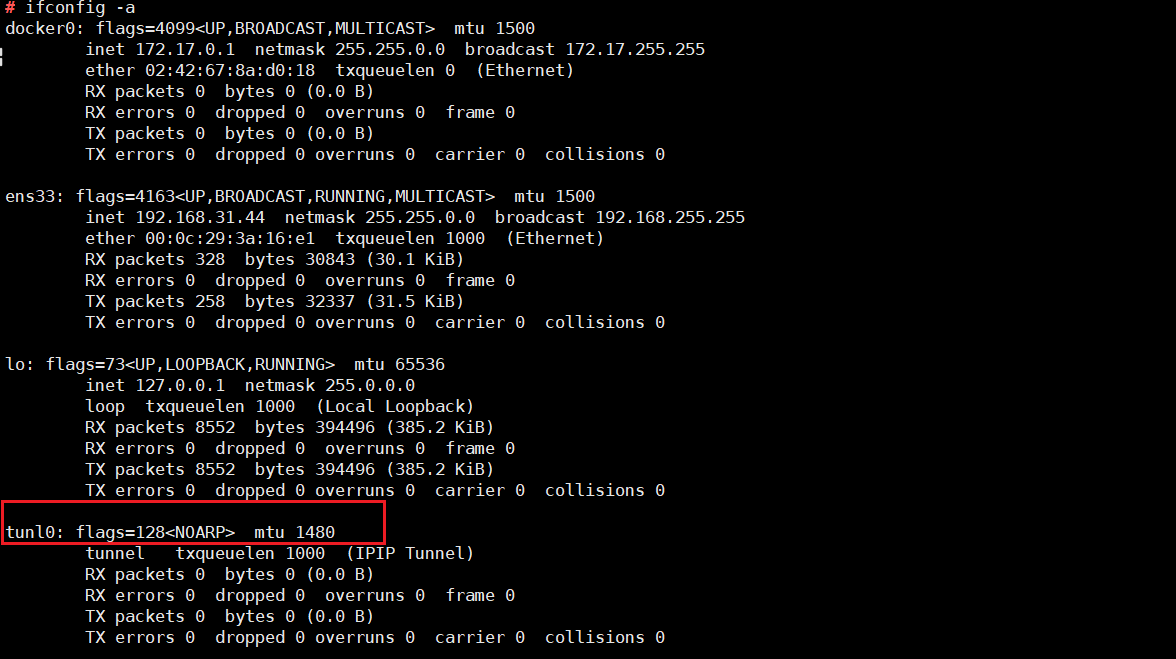
# cat /etc/sysconfig/ipvsadm

## 5.2 realserver 43,44

**# modprobe ipip**

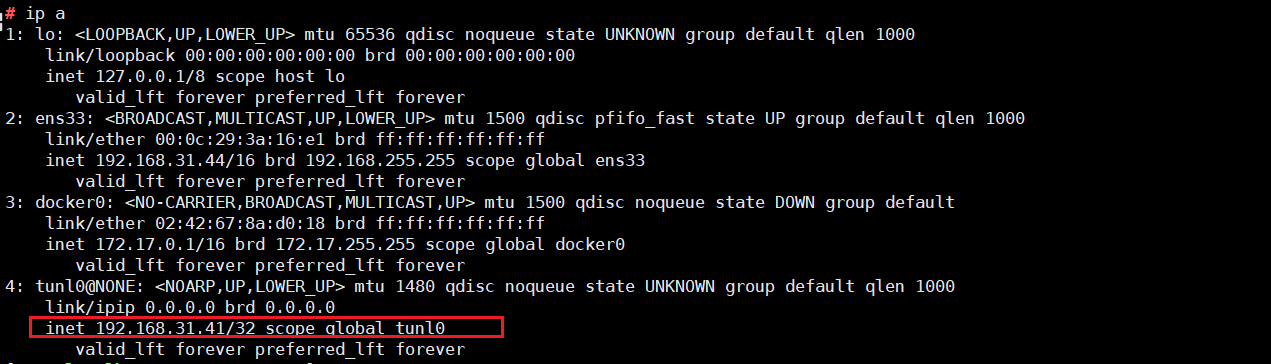
**# lsmod | grep ipip**

**# ifconfig -a**



多了个tunl0

**# ifconfig tunl0 192.168.31.41netmask 255.255.255.255 up**



## 5.3关闭ARP转发

**永久生效：**

**# vim /etc/sysctl.conf #以文件最后添加以下内容**

**net.ipv4.conf.tunl0.arp\_ignore = 1**

**net.ipv4.conf.tunl0.arp\_announce = 2**

**net.ipv4.conf.all.arp\_ignore = 1**

**net.ipv4.conf.all.arp\_announce = 2**

**net.ipv4.conf.tunl0.rp\_filter = 0**

**net.ipv4.conf.all.rp\_filter = 0**

**# sysctl -p**

## 5.4 部署nginx

**RealServer：在43和44上面配置：**

**#docker run -d --name ng -p 80:80 nginx**

**#docker run -d --name ng -p 80:80 nginx**

#docker exec -it ng bash -c "echo 43 > /usr/share/nginx/html/index.html"

#docker exec -it ng bash -c "echo 44 > /usr/share/nginx/html/index.html"

## 5.5 验证

# 六 使用webbench测试网站性能

**网站压力测试工具-Webbench**

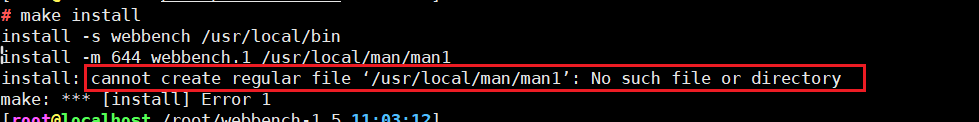
**一.webbench简介：  
Webbench是有名的网站压力测试工具，它是由 Lionbridge公司（http://www.lionbridge.com）开发的网站压力测试工具，它能测试处在相同硬件上，不同服务的性能以及不同硬件上同一个服务的运行状况。webbench不但能具有便准静态页面的测试能力，还能对动态页面（ASP,PHP,JAVA,CGI）进行测试的能力。**

**官网：http://www.lionbridge.com**

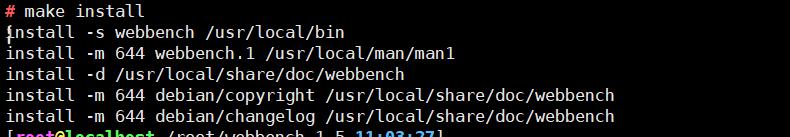
**#yum install gcc\*  ctags\* -y**

**# wget http://www.ha97.com/code/webbench-1.5.tar.gz  
#tar zxvf webbench-1.5.tar.gz**  **#cd webbench-1.5  
#make**

**# mkdir -p /usr/local/man/man1 #不创建这个执行make install报错：**



#make install



# webbench -h

#webbench -c 20 -t 10 http://192.168.31.43/index.html

|  |
| --- |
| Webbench - Simple Web Benchmark 1.5  Copyright (c) Radim Kolar 1997-2004, GPL Open Source Software.  Benchmarking: GET http://192.168.31.43/index.html  20 clients, running 10 sec.  Speed=100116 pages/min, 387068 bytes/sec.  Requests: 16686 susceed, 0 failed. |

**#当有20个客户端时，一分钟可以响应**100116 **个页面，1秒可以传输**387068 **字节**

**#20个客户端，1秒产生了**16686 **个请求，0个失败。**

**四.测试注意事项：  
 1.压力测试工作应该放到产品上线之前，而不是上线以后；  
 2.webbench 做压力测试时，该软件自身也会消耗CPU和内存资源，为了测试准确，请将 webbench 安装在别的服务器上；  
 3.测试时尽量跨公网进行，而不是内网； 如果带宽不够时，可以内网测试。  
 4.测试时并发应当由小逐渐加大，观察一下网站负载及打开是否流畅，直到网站打开缓慢甚至网站完全打不开； 可以一边在linux测试，一个在浏览上打开，查看是否流畅。  
 5.应尽量进行单元测试，如B2C网站可以着重测试购物车、推广页面等，因为这些页面占整个网站访问量比重较大。**

# 七 生产环境DR模式用的比较多

# 八 使用Keepalived构建LVS-DR模式的高可用集群

192.168.31.41 VIP

192.168.31.42 keepalived 分发器

192.168.31.43 keepalived 分发器

192.168.31.44 RIP

192.168.31.45 RIP

## 8.1 安装keepalive ipvsadm

#yum install -y keepalived [31.41 31.42]

#yum install -y ipvsadm [all]

# rpm -qc keepalived

## 8.2 keepalive 配置

### 31.42 上配置 作为主

|  |
| --- |
| global\_defs {  notification\_email {  root@localhost #默认三个地址，修改可用地址  }  notification\_email\_from root@localhost  smtp\_server localhost  smtp\_connect\_timeout 30  router\_id xuegod62 #标识当前节点名字，两个节点的此项需要不相同。  }  #默认的配置文件中，使用第三方 smtp 服务器，但这在现实中几乎没有意义，发不出邮件，我们将其挃定为 localhost, 我们也可以将通知信息的发送交给本地 sendmail 服务处理。  vrrp\_instance apache { #定义一个实例，一个集群就是一个实例。 默认VI\_1 可以随意改  state MASTER #指定 A 节点为主节点 备用节点上设置为 BACKUP 即可  interface ens33 #绑定虚拟 IP 的网络接口  virtual\_router\_id 51 #VRRP 组名，两个节点的设置必须一样，以指明各个节点属于同一 VRRP 组  priority 100 #主节点的优先级（1-254 之间），备用节点必须比主节点优先级低  advert\_int 1 #组播信息发送间隔，两个节点设置必须一样  authentication { #设置验证信息，两个节点必须一致  auth\_type PASS  auth\_pass 1111  }  virtual\_ipaddress {  192.168.31.41 #指定虚拟 IP, 两个节点设置必须一样  }  }  #类似添加虚拟一个服务 ipvsadm -A -t 192.168.31.41:80 -s rr  virtual\_server 192.168.31.41 80 { #对虚拟IP63添加LVS相关内容  delay\_loop 6 #Keepalived 多长时间监测一次 RS  lb\_algo rr #分发算法  lb\_kind DR #DR 模式  nat\_mask 255.255.255.0  persistence\_timeout 50 #同一 IP 50秒内的请求都发到同个real server ，这个会影响LVS的 rr 调度算法， 同一 IP 超过 50 秒后，再次访问，才会被转发到另一台 real server 上。 persistence 持久性的意思  protocol TCP  # ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.44 -g 添加的内容，类似这条命令  # ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.45 -g 添加的内容，类似这条命令  real\_server 192.168.31.44 80 { #配置服务节点 1，需要指定 realserver 的真实 IP 地址和端口，IP不端口之间用空格隔开  weight 1 #配置服务节点的权值，权值大小用数字表示，数字越大，权值越高，设置权值大小可以为不同性能的服务器  TCP\_CHECK { #这段内容手动添加，爸以前的内容删除  connect\_timeout 3 #表示 3 秒无响应超时  nb\_get\_retry 3 #表示重试次数  delay\_before\_retry 3 #表示重试间隔  connect\_port 80 #检测端口  }  }  real\_server 192.168.31.45 80 {  weight 1  TCP\_CHECK {  connect\_timeout 3  nb\_get\_retry 3  delay\_before\_retry 3  connect\_port 80  }  }  } |

### 31.43 配置 作为备

|  |
| --- |
| global\_defs {  notification\_email {  root@localhost #默认三个地址，修改可用地址  }  notification\_email\_from root@localhost  smtp\_server localhost  smtp\_connect\_timeout 30  router\_id xuegod63 #标识当前节点名字，两个节点的此项需要不相同。  }  #默认的配置文件中，使用第三方 smtp 服务器，但这在现实中几乎没有意义，发不出邮件，我们将其挃定为 localhost, 我们也可以将通知信息的发送交给本地 sendmail 服务处理。  vrrp\_instance apache { #定义一个实例，一个集群就是一个实例。 默认VI\_1 可以随意改  state BACKUP #指定 A 节点为主节点 备用节点上设置为 BACKUP 即可  interface ens33 #绑定虚拟 IP 的网络接口  virtual\_router\_id 51 #VRRP 组名，两个节点的设置必须一样，以指明各个节点属于同一 VRRP 组  priority 90 #主节点的优先级（1-254 之间），备用节点必须比主节点优先级低  advert\_int 1 #组播信息发送间隔，两个节点设置必须一样  authentication { #设置验证信息，两个节点必须一致  auth\_type PASS  auth\_pass 1111  }  virtual\_ipaddress {  192.168.31.41 #指定虚拟 IP, 两个节点设置必须一样  }  }  #类似添加虚拟一个服务 ipvsadm -A -t 192.168.31.41:80 -s rr  virtual\_server 192.168.31.41 80 { #对虚拟IP63添加LVS相关内容  delay\_loop 6 #Keepalived 多长时间监测一次 RS  lb\_algo rr #分发算法  lb\_kind DR #DR 模式  nat\_mask 255.255.255.0  persistence\_timeout 50 #同一 IP 50秒内的请求都发到同个real server ，这个会影响LVS的 rr 调度算法， 同一 IP 超过 50 秒后，再次访问，才会被转发到另一台 real server 上。 persistence 持久性的意思  protocol TCP  # ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.44 -g 添加的内容，类似这条命令  # ipvsadm -a -t 192.168.31.41:80 -r 192.168.31.45 -g 添加的内容，类似这条命令  real\_server 192.168.31.44 80 { #配置服务节点 1，需要指定 realserver 的真实 IP 地址和端口，IP不端口之间用空格隔开  weight 1 #配置服务节点的权值，权值大小用数字表示，数字越大，权值越高，设置权值大小可以为不同性能的服务器  TCP\_CHECK { #这段内容手动添加，爸以前的内容删除  connect\_timeout 3 #表示 3 秒无响应超时  nb\_get\_retry 3 #表示重试次数  delay\_before\_retry 3 #表示重试间隔  connect\_port 80 #检测端口  }  }  real\_server 192.168.31.45 80 {  weight 1  TCP\_CHECK {  connect\_timeout 3  nb\_get\_retry 3  delay\_before\_retry 3  connect\_port 80  }  }  } |

主要3个地方要修改

## 8.3 RIP配置

写一个配置RS的脚本[31.44 ,31.45操作]

# vim /etc/init.d/lvsrsdr

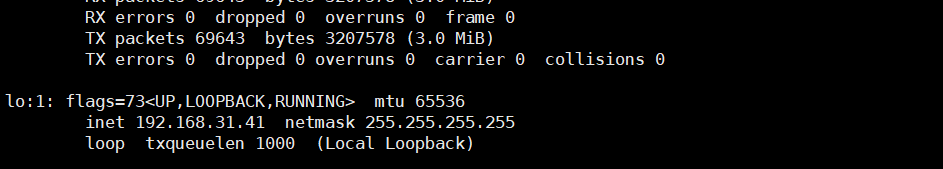
|  |
| --- |
| #!/bin/bash  #description:start relserver  VIP**=**192.168.31.41  source **/**etc**/**init.d**/**functions #加载环境变量（可以加载所有的环境变量）  **case** **$1** **in**  **start)**  **echo** 'start LVS of Realserver DR'  **/**sbin**/**ifconfig lo**:**1 **$VIP** broadcast **$VIP** netmask 255**.**255**.**255**.**255 up  **/**sbin**/**route add **-**host **$VIP** dev lo**:**1  **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  **;;**  **stop)**  **/**sbin**/**ifconfig lo**:**1 down  **echo** 'Close LVS of Realserver DR'  **echo** '0' **>** **/**proc**/**sys**/**net**/**ipv4**/**conf**/**lo**/**arp\_ignore  **echo** '0' **>** **/**proc**/**sys**/**net**/**ipv4**/**conf**/**lo**/**arp\_announce  **echo** '0' **>** **/**proc**/**sys**/**net**/**ipv4**/**conf**/**all**/**arp\_ignore  **echo** '0' **>** **/**proc**/**sys**/**net**/**ipv4**/**conf**/**all**/**arp\_announce  **;;**  **\*)**  **echo** "Usage:$0 (start|stop)"  **exit** 1  **esac** |

# chmod +x /etc/init.d/lvsrsdr

# /etc/init.d/lvsrsdr start

# echo "/etc/init.d/lvsrsdr start" >> /etc/rc.local

# ifconfig -a



## 8.4 部署nginx

**RealServer：在44和45上面配置：**

**#docker run -d --name ng -p 80:80 nginx**

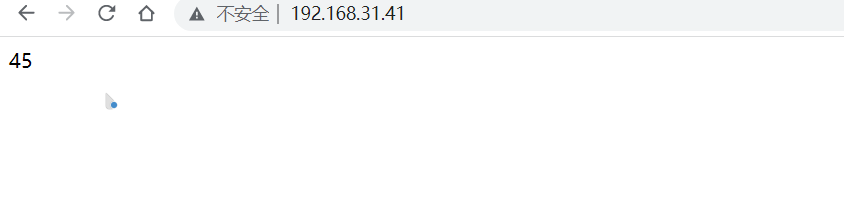
**#docker run -d --name ng -p 80:80 nginx**

#docker exec -it ng bash -c "echo 44 > /usr/share/nginx/html/index.html"

#docker exec -it ng bash -c "echo 45 > /usr/share/nginx/html/index.html"

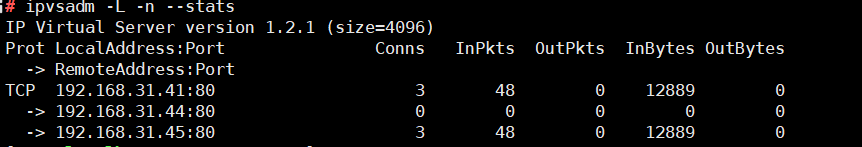
## 8.5 测试

<http://192.168.31.41/> #一直显示192.168.31.45的内容，等待50s之后，查看



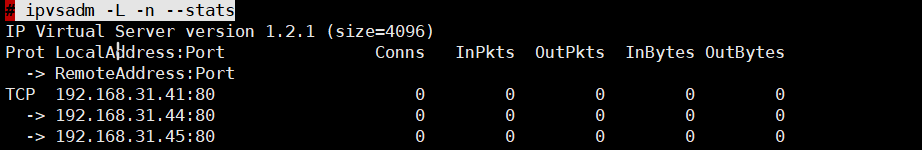
在主上42上查看状态

# ipvsadm -L -n --stats



在备上查看43的状态

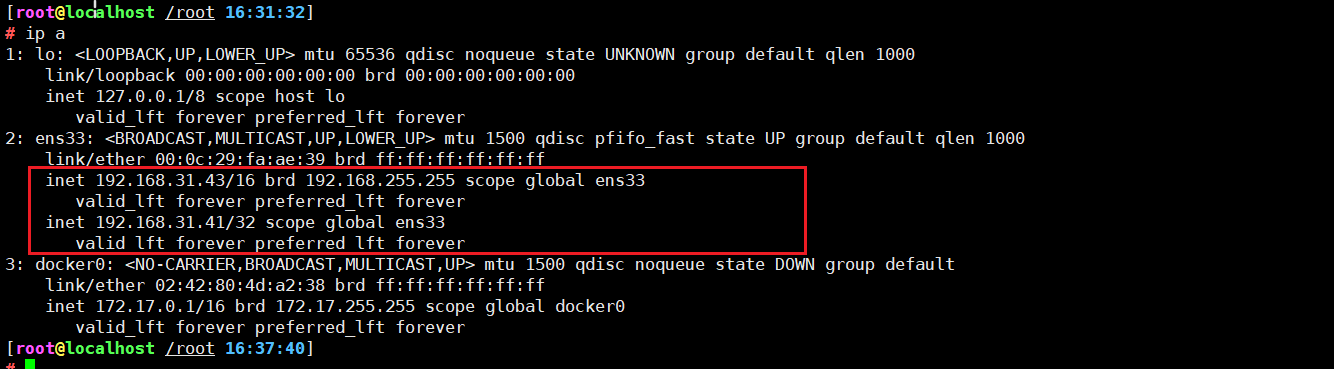
# ipvsadm -L -n --stats



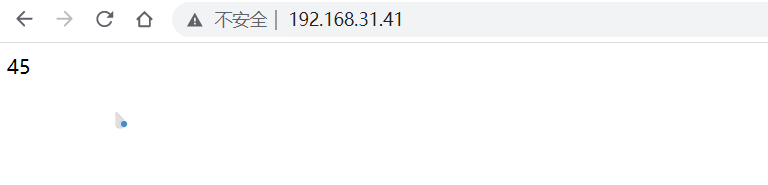
### 8.5.1 停掉主keepalived

# systemctl stop keepalived.service

虚拟IP漂移到了43



也还能正常访问，见下图



### 8.5.2 停掉一台 nginx,比如停掉45

#docker stop ng

也还能正常访问，访问到了44那一台

