**Morning：**

1. Keepalived概述

-专为LVS设计，监控各服务器节点的状态

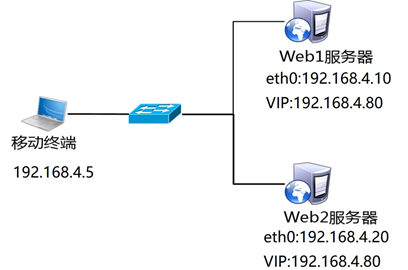
-带有VRRP功能，防止单点故障。（相当于路由热备）

-作用：1）根据优先级给代理服务器配浮动VIP；

2）自动配置LVS（健康检查）

1. Keepalived高可用服务器

**-实验拓扑**



**步骤1：搭建Web服务**

**步骤2：安装Keepalived软件**

[root@room12pc2 ~]# pssh -H 192.168.4.100 192.168.4.200 yum install -y keepalived

**步骤3：部署Keepalived服务**

1）Web1服务器Keepalived配置文件

[root@web1 ~]# vim /etc/keepalived/keepalived.conf

global\_defs {

notification\_email { #不联网无需配置

admin@tarena.com.cn #设置报警收件人邮箱

}

notification\_email\_from ka@localhost #设置发件人

smtp\_server 127.0.0.1 #定义邮件服务器

smtp\_connect\_timeout 30

router\_id web1 #设置路由ID号

}

vrrp\_instance VI\_1 { #配置浮动IP

state MASTER #决定启动后的初始状态

interface eth0 #定义网络接口

virtual\_router\_id 50 #主辅VRID号必须一致

priority 100 #服务器优先级

advert\_int 1 #1秒核对一次优先级

authentication {

auth\_type pass

auth\_pass forlvs #主辅服务器密码（确保安全）

}

virtual\_ipaddress { 192.168.4.80 }

#谁是主服务器谁获得该VIP

}

1. Web2服务器Keepalived配置文件

[root@web2 ~]# vim /etc/keepalived/keepalived.conf

#设置router\_id、state BACKUP、priority，其他与Web1一致

1. 启动服务，查看浮动IP

[root@web1 ~]# systemctl start keepalived

[root@web2 ~]# systemctl start keepalived

[root@web1 ~]# ip a s

#谁是主服务器，谁拥有浮动IP

4）配置防火墙和SELinux

#每次启动Keepalved都会默认启动iptables防火墙

#并自动添加一个drop的防火墙规则

[root@web1 ~]# iptables -nL #查看防火墙规则

[root@web1 ~]# iptables -F #清空防火墙规则

[root@web1 ~]# setenforce 0

**步骤4：测试**

1）将Web1关机，查看Web2是否获得浮动IP

[root@web2 ~]# ip addr show

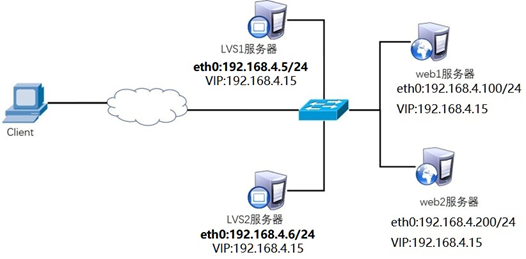
2）启动Web1，重新获得浮动IP

[root@web1 ~]# ip addr show

**Afternoon：**

1、Keepalived+LVS服务器

**-实验拓扑**



**步骤1：配置网络参数（LVS-DR模式）**

Proxy1：192.168.4.5/24（eth0）

Proxy2：192.168.4.6/24（eth0）

Web1：192.168.4.100/24（eth0）、192.168.4.15/32（lo:0）

Web2：192.168.4.200/24（eth0）、192.168.4.15/32（lo:0）

#Web1、Web2需要配置/etc/sysctl.conf防止ARP冲突

#参考：阶段2 --> 3.CLUSTER --> DAY02 --> Page5

**步骤2：搭建后台Web服务**

**步骤3：两台调度器安装软件**

[root@proxy1 ~]# yum install -y keepalived ipvsadm

[root@proxy1 ~]# systemctl enable keepalived

[root@proxy1 ~]# ipvsadm -C

**步骤4：部署Keepalived+LVS**

1. proxy1设置Keepalived

[root@proxy1 ~]# vim /etc/keepalived/keepalived.conf

global\_defs { #参考Morning-->实验2

...

}

vrrp\_instance VI\_1 { #参考Morning-->实验2

...

}

virtual\_server 192.168.4.15 80 { #设置VIP

delay\_loop 6

lb\_algo wrr #设置LVS调度算法

lb\_kind DR #设置LVS的模式

#persistence\_timeout 50

#客户端请求转发给同一台后端服务器，保留50秒

protocol TCP

real\_server 192.168.4.100 80 { #设置RIP

weight 1 #设置权重

TCP\_CHECK { #健康检查（发送TCP请求）

connect\_timeout 3 #每次连接等待3秒

nb\_get\_retry 3 #重试3次连接

delay\_before\_retry 3 #每次重试延迟3秒

}

}

real\_server 192.168.4.200 80 {

weight 2

TCP\_CHECK {

connect\_timeout 3

nb\_get\_retry 3

delay\_before\_retry 3

}

}

}

1. proxy2设置Keepalived

[root@proxy2 ~]# vim /etc/keepalived/keepalived.conf

#参考proxy1的配置

Tips：关于健康检查

检查网站是否被攻击篡改：curl 192.168.4.100 | md5sum

或者：genhash -s 192.168.4.100 -p 80 -u /index.html

HTTP\_GET {

url {

path /index.html #指定某个网页

digest 640205b7b0fc66c1ea91c463fac6334d

#指定该网页的hash校验值

}

url {

...

}

}

1. 重启keepalived服务，关闭防火墙

[root@proxy1 ~]# systemctl restart keepalived

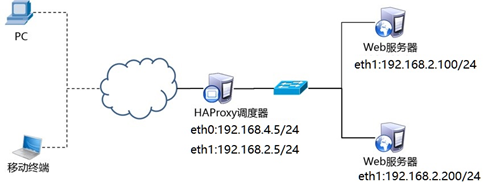
[root@proxy1 ~]# iptables -F

**步骤5：客户端测试**

[root@client ~]# curl 192.168.4.15

2、HAProxy负载平衡集群

**-实验拓扑**



注意：清空Proxy1、Proxy2的LVS、Keepalived

清空Web1、Web2的lo：0、ARP冲突  
 [root@web1 ~]# ifdown eth0

**步骤1：搭建后台Web服务**

**步骤2：部署HAProxy服务器**

1. 配置网络，安装软件

[root@proxy ~]# echo 'net.ipv4.ip\_forward = 1' >> sysctl.conf

#开启路由转发

[root@proxy ~]# sysctl -p

[root@proxy ~]# yum -y install haproxy

2）修改配置文件

[root@proxy ~]# vim /etc/haproxy/haproxy.cfg

global #全局设置，无需配置

log 127.0.0.1 local2 #日志

chroot /usr/local/haproxy #工作目录

pidfile /var/run/haproxy.pid #pid存放路径

maxconn 4000 #最大并发量，默认4000

user haproxy

group haproxy

daemon #进程进入deamon模式运行

Defaults #集群默认设置

mode http #{ tcp|http|health }(tcp:四层调度，http：七层调度)

log global #采用全局定义的日志

option dontlognull #不记录健康检查的日志信息

option httpclose #每次请求完毕后主动关闭http通道

option httplog #日志类别http日志格式

option forwardfor

#后端服务器可以从Http Header中获得客户端ip

option redispatch

#serverid服务器挂掉后强制定向到其他健康服务器

timeout connect 10000 #如果backend没有指定，默认为10s

timeout client 300000 #客户端连接超时

timeout server 300000 #服务器连接超时

maxconn 60000 #最大连接数

retries 3 #3次连接失败就认为服务不可用，也可以通过后面设置

listen webs 0.0.0.0:80

cookie SERVERID rewrite #打开cookie功能

balance roundrobin #轮询算法

server web1 192.168.2.100:80 cookie app1inst1 check inter 2000 rise 2 fall 5

#每隔2s健康检查，2次成功up，5次失败down

#服务器返回页面时，加上cookie标记再返回给用户

server web2 192.168.2.200:80 cookie app1inst2 check inter 2000 rise 2 fall 5

listen stats #与nginx的stub\_status大致一样

bind 0.0.0.0:1080 #监听端口

stats refresh 30s #统计页面自动刷新时间

stats uri /stats #统计页面url

stats realm Haproxy Manager #统计页面密码框上提示文本

stats auth gjq:123 #统计页面用户名和密码设置

#stats hide-version #隐藏统计页面上HAProxy的版本信息

3）启动服务器并设置开机启动

[root@proxy ~]# systemctl start haproxy

[root@proxy ~]# systemctl enable haproxy

[root@proxy ~]# netstat -anptlu | grep 80

**步骤3：客户端验证**

[root@client ~]# curl 192.168.4.5

[root@client ~]# http://192.168.4.5:1080/stats

3、集群软件对比

**-Nginx**

1. 可以做Web服务器、反向代理服务器、负载均衡服务器、支持动静分离，也支持正则表达式；
2. 安装配置简单，可以通过日志解决多数问题；
3. 可以承担高的负载压力且稳定，一般能支撑超过1万次的并发；
4. 工作在网络的7层，可以针对http应用做一些分流策略，比如域名、目录结构。
5. 健康检查只能通过端口，无法使用url。由于仅工作在7层，因此适用范围较小。不支持Session的直接保持，只能通过ip\_hash来解决。

**-HAProxy**

1. 工作在4层/7层，调度算法丰富；
2. 支持session、cookie，可以通过url健康检查；
3. 效率、负载均衡速度比Nginx更优秀；
4. 但正则较Nginx弱，不支持Apache日志。

**-lvs，keepalived+lvs**

1）LVS是内核级别的软件，因此性能较高，对内存/CPU的消耗较低，抗负载能力强。

2）工作在4层，应用面广，几乎可以为所有应用提供负载均衡；

3）负载均衡算法多：rr轮询、wrr、lc、wlc

4）keepalived是专门为LVS设计的一套热备方案，稳定性、可靠性好。

5）但不支持正则，不能实现动静分离。如果架构庞大，配置比较繁琐。

**三大主流软件负载均衡器适用业务场景：**

1、网站建设初期，可以选用Nigix/HAproxy作为反向代理负载均衡，因为其配置简单，性能也能满足一般的业务场景。如果考虑到负载均衡器是有单点问题，可以采用Nginx+Keepalived/HAproxy+Keepalived避免负载均衡器自身的单点问题。

2、网站并发达到一定程度之后，为了提高稳定性和转发效率，可以使用LVS、毕竟LVS比Nginx/HAproxy要更稳定，转发效率也更高。不过维护LVS对维护人员的要求也会更高，投入成本也更大。

注：Niginx与Haproxy比较：Niginx支持七层、用户量最大，稳定性比较可靠。Haproxy支持四层和七层，支持更多的负载均衡算法，支持session保存等。具体选型看使用场景，目前来说Haproxy由于弥补了一些Niginx的缺点用户量也不断在提升。