Linux 集群化

尚硅谷 - 汪洋

目录

一、集群概述

1、集群是什么？

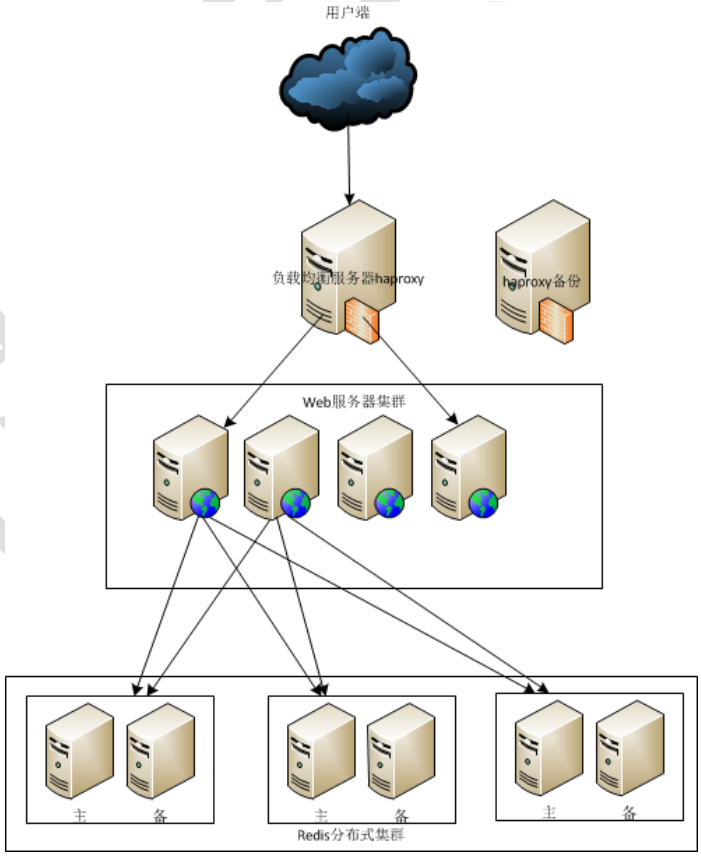
定义：一组协同工作的服务器，对外表现为一个整体

集群的意义：更好的利用现有资源实现服务的高度可用

集群扩展方式

- 垂直扩展：更换服务器硬件
- 水平扩展：添加更多的服务器节点

常见的集群拓扑



2、集群的分类

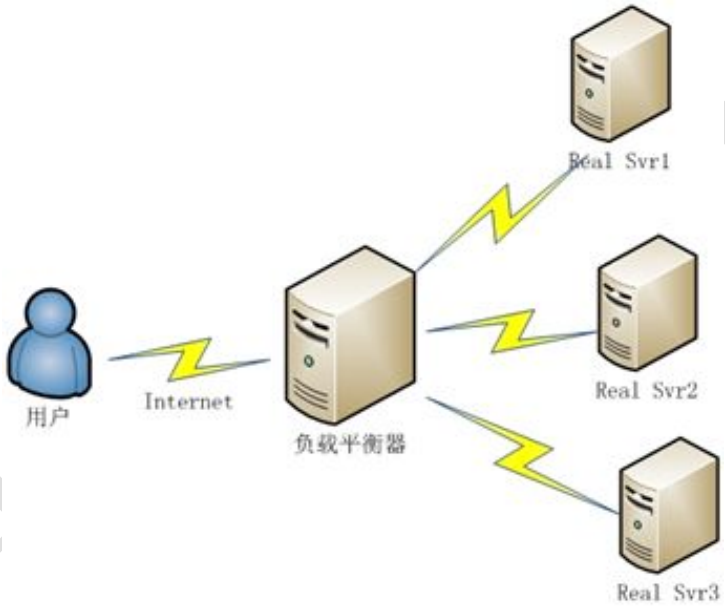
①负载均衡集群 - LBC

使用意图：减轻单台服务器的压力，将用户请求分担给多台主机一起处理

实现方法

- 软件：LVS　RAC　Nginx
- 硬件：F5　BIG-IP

负载均衡集群架构拓扑



调度器分类

- 触发条件不同
  - 四层：传输层　IP+PORT
  - 七层：应用层　URL
- 实现原理不同
  - 四层：TCP 连接只建立一次，客户端和正式服务器
  - 七层：TCP 连接建立两次，客户端和负载调度器 负载调度器和真实服务器
- 实现场景不同
  - 四层：TCP 应用　如：基于 C/S 机构的 ERP 系统

- 七层：HTTP 应用 如：根据用户访问域名的方式，判断用户语言

- 安全性不同
  - 四层：转发 SYN 攻击
  - 七层：可以拦截 SYN 攻击

使用范围：业务并发较大的应用程序

②高可用集群 – HAC

使用意图：最大限度的保证用户的应用持久，不间断的提供服务

最大限度

99%	99	87.6 小时
99.9%	999	8.8 小时
99.99%	9999	53 分钟
99.999%	99999	5 分钟

实现原理：心跳检测

实现方法

- 软件
  - heartbeat linux-HA
  - RHCS
  - ROSE
  - keepalived

- 硬件
  - F5

特殊情况：脑分裂

- 可能出现的问题：数据不完整、数据不可访问

- 解决方法：预防：冗余、强制隔离：电源交换机

使用范围：需要持续提供服务的应用程序

③高性能运算集群 – HPC

使用意图：提供单台计算机所不具备的计算能力

LBC 与 HAC 的原理对比：

负载均衡集群通过提高单位时间内执行的任务数来提升效率

高性能运算集群通过缩短单个任务的执行时间来提高效率

使用范围：天气计算、火箭弹道演算

二、负载均衡集群

1、LVS 相关原理

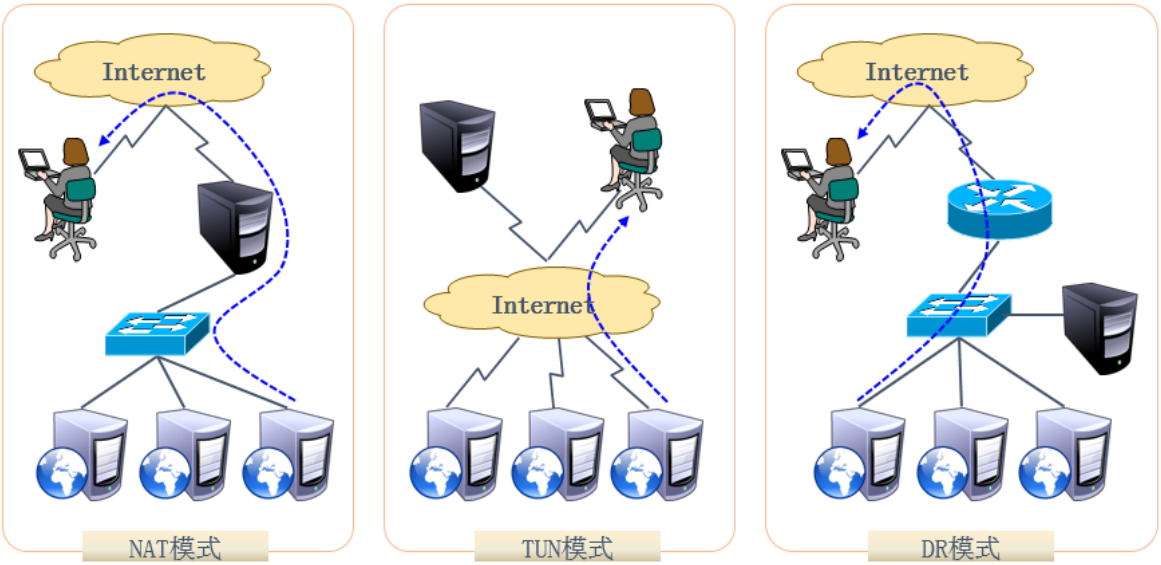
LVS 的组成

- IPVS：运行在内核空间
- IPVSADM：运行在用户空间，管理集群服务的命令行工具

LVS 的原理：根据用户请求的套接字判断，分流至真实服务器的工作模块

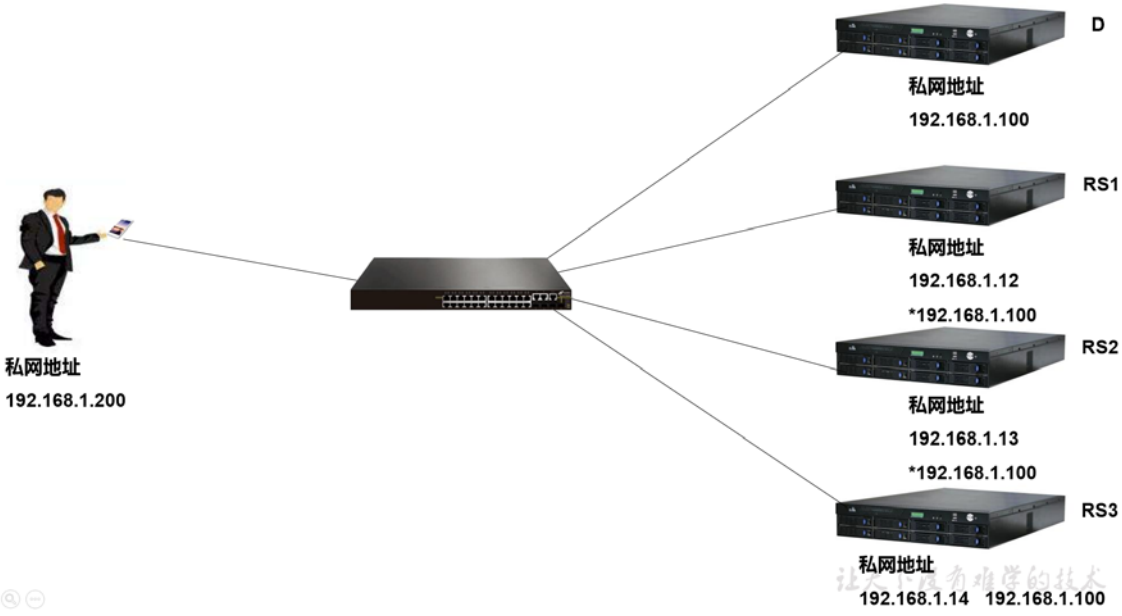
2、LVS 工作方式

工作模式



① LVS – DR 模式

工作逻辑图



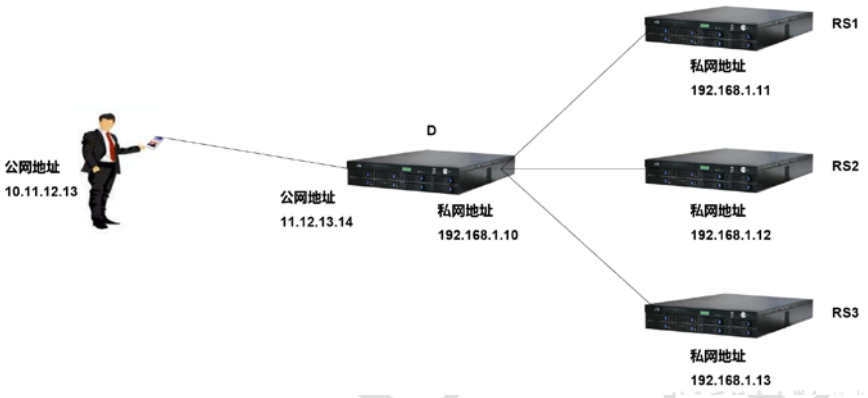
模式特点

- 集群节点，必须在一个网络中
- 真实服务器网关指向路由器
- RIP 既可以是私网地址，又可以是公网地址

- 负载调度器只负责入站请求
- 大大减轻负载调度器压力，支持更多的服务器节点

② LVS – NAT 模式

工作逻辑图



模式特点

- 集群节点，必须在一个网络中
- 真实服务器必须将网关指向负载调度器
- RIP 通常都是私有 IP，仅用于各个集群节点通信
- 负载调度器必须位于客户端和真实服务器之间，充当网关
- 支持端口映射
- 负载调度器操作系统必须是 Linux ，真实服务器可以使用任意系统

③ LVS – TUN 模式

工作逻辑图

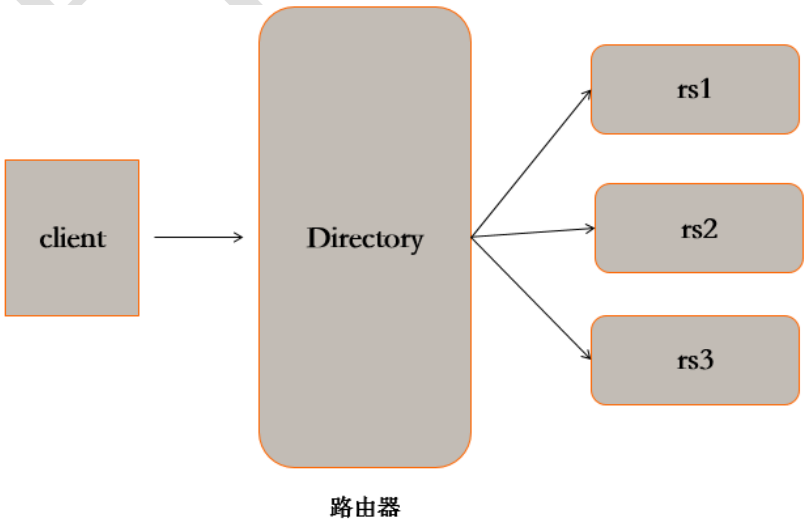
模式特点

- 集群节点不必位于同一个物理网络但必须都拥有公网 IP（或都可以被路由）
- 真实服务器不能将网管指向负载调度器
- RIP 必须是公网地址
- 负载调度器只负责入站请求
- 不支持端口映射功能
- 发送方和接收方必须支持隧道功能

三、 LVS 实验构建

① LVS – NAT 模式集群构建

实验架构图



构建代码

负载调度器

vi /etc/sysctl.conf　　# 开启路由转发功能

net.ipv4.ip_forward=1

sysctl -p

iptables -t nat -A POSTROUTING -s 10.10.10.0/24 -o eth0 -j SNAT --to-source 20.20.20.11

添加防火墙记录，当源地址是 内网网段 并且出口网卡为 eth0 的时候进行 SNAT 转换，转换源地址为外网卡地址

iptables -t nat -L　# 查看记录是否保存成功

ipvsadm -A -t 20.20.20.11:80 -s rr　　# 添加 ipvsadm TCP 集群

ipvsadm -a -t 20.20.20.11:80 -r 10.10.10.12:80 -m　　# 添加 ipvsadm 节点

ipvsadm -Ln

service ipvsadm save　　# 保存 ipvs 集群设置到文件进行持久化

chkconfig ipvsadm on

真实服务器

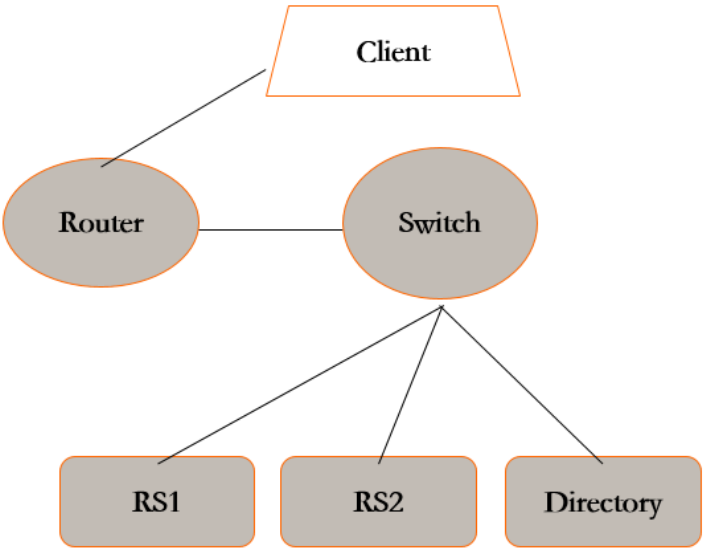
route add default gw IP 地址　　# 指定网关至负载调度器

service httpd start　　# 开启 Apache 服务器

chkconfig httpd on

② LVS – DR 模式集群构建

实验架构图



构建代码

负载调度器

```
service NetworkManager stop                    # 关闭网卡守护进程

cd /etc/sysconfig/network-scripts/
cp ifcfg-eth0 ifcfg-eth0:0                     # 拷贝 eth0 网卡子接口充当集群入口接口
vim ifcfg-eth0:0
    DEVICE=eth0:0
    IPADDR=虚拟 IP
    NETMASK=255.255.255.0
ifup eth0:0

vim /etc/sysctl.conf                           # 关闭网卡重定向功能
    net.ipv4.conf.all.send_redirects = 0
    net.ipv4.conf.default.send_redirects = 0
    net.ipv4.conf.eth0.send_redirects = 0
sysctl -p

modprobe ip_vs          # 重载 ipvs 模块

rpm -ivh ipvsadm-1.26l.....      # 安装 ipvsadm 命令行工具

ipvsadm -v     # 查看当前 ipvs 集群内容
ipvsadm -A -t 虚拟 IP:80 -s rr     # 添加 ipvs TCP 集群
ipvsadm -a -t 虚拟 IP:80 -r 网站 1:80 -g    # 添加 ipvsadm 集群子节点
ipvsadm -a -t 虚拟 IP:80 -r 网站 2:80 -g
ipvsadm -Ln
service ipvsadm save    # 保存 ipvs 集群内容至文件，进行持久化存储
chkconfig ipvsadm on    # 设置为开机自启
```

真实服务器

```
service NetworkManager stop            # 关闭网卡守护进程
cd /etc/sysconfig/network-scripts/
cp ifcfg-lo ifcfg-lo:0
vim ifcfg-lo:0        # 拷贝回环网卡子接口
    DEVICE=lo:0
    IPADDR=虚拟 IP
    NETMASK=255.255.255.255

vim /etc/sysctl.conf# 关闭对应 ARP 响应及公告功能
    net.ipv4.conf.all.arp_ignore = 1
    net.ipv4.conf.all.arp_announce = 2
    net.ipv4.conf.default.arp_ignore = 1
    net.ipv4.conf.default.arp_announce = 2
```

```
        net.ipv4.conf.lo.arp_ignore = 1
        net.ipv4.conf.lo.arp_announce = 2
    sysctl -p

    ifup lo：  0

    route add -host  虚拟 IP dev lo:0        # 添加路由记录，当访问 VIP 交给 lo:0 网卡接受
    service httpd start
```

4、负载均衡集群相关调度算法

① 静态调度算法

特点：只根据算法本身去调度，不考虑服务器本身

算法说明

- RR　轮询：将每次用户的请求分配给后端的服务器，从第一台服务器开始到第 N 台结束，然后循环
- WRR　加权轮询：按照权重的比例实现在多台主机之间进行调度
- SH（source hash）源地址散列：将同一个 IP 的用户请求，发送给同一个服务器
- DH（destination hash）目标地址散列：将同一个目标地址的用户请求发送给同一个真实服务器（提高缓存的命中率）

③　动态调度算法

特点：除了考虑算法本身，还要考虑服务器状态

算法说明

- LC（lest-connection）最少连接：将新的连接请求，分配给连接数最少的服务器　活动连接 × 256 + 非活动连接
- WLC 加权最少连接：特殊的最少连接算法，权重越大承担的请求数越多　（活动连接 × 256 + 非活动连接 ）　/　权重
- SED 最短期望延迟：特殊的 WLC 算法　（活动连接 + 1） * 256 / 权重
- NQ 永不排队：特殊的 SED 算法，无需等待，如果有真实服务器的连接数等于 0 那就直接分配不需要运算
- LBLC 特殊的 DH 算法：即能提高缓存命中率，又要考虑服务器性能
- LBLCR　LBLC+缓存：尽可能提高负载均衡和缓存命中率的折中方案

5、LVS 持久连接

① 持久客户端连接

定义：每客户端持久；将来自于同一个客户端的所有请求统统定向至此前选定的 RS；也就是只要 IP 相同，分配的服务器始终相同

演示代码

```
ipvsadm -A -t 172.16.0.8:0 -s wlc -p 120    # 添加一个 tcp 负载集群，集群地址为 172.16.0.8 ，算法为 wlc，持久化时间为 120s
```

② 持久端口连接

定义：每端口持久；将来自于同一个客户端对同一个服务(端口)的请求，始终定向至此前选定的 RS

演示代码

```
ipvsadm -A -t 172.16.0.8:80 -s rr -p 120    # 添加一个 tcp 负载集群，集群地址为 172.16.0.8:80 ，算法为 wlc，持久化时间为 120s
```

④ 持久防火墙标记连接

定义：将来自于同一客户端对指定服务(端口)的请求，始终定向至此选定的 RS；不过它可以将两个毫不相干的端口定义为一个集群服务

演示代码

```
iptables -t mangle -A PREROUTING -d 172.16.0.8 -p tcp --dport 80 -j MARK --set-mark 10      # 添加一个防火墙规则，当目标地址为 172.16.0.8 并且 目标端口为 80 时给数据包打一个标记，设置 mark 值为 10
iptables -t mangle -A PREROUTING -d 172.16.0.8 -p tcp --dport 443 -j MARK --set-mark 10     # 添加一个防火墙规则，当目标地址为 172.16.0.8 并且 目标端口为 443 时给数据包打一个标记，设置 mark 值为 10
service iptables save     # 保存防火墙规则持久化生效
ipvsadm -A -f 10 -s wlc -p 120     # 添加一个负载调度器，当 mark 值为 10 时进行负载均衡使用 wlc 算法，持久化生效时间为 120s
```

三、高可用集群

1、Keepalived 相关说明

① 软件相关介绍

案例环境专为 LVS 和 HA 设计的一款健康检查工具

支持故障自动切换（Failover）

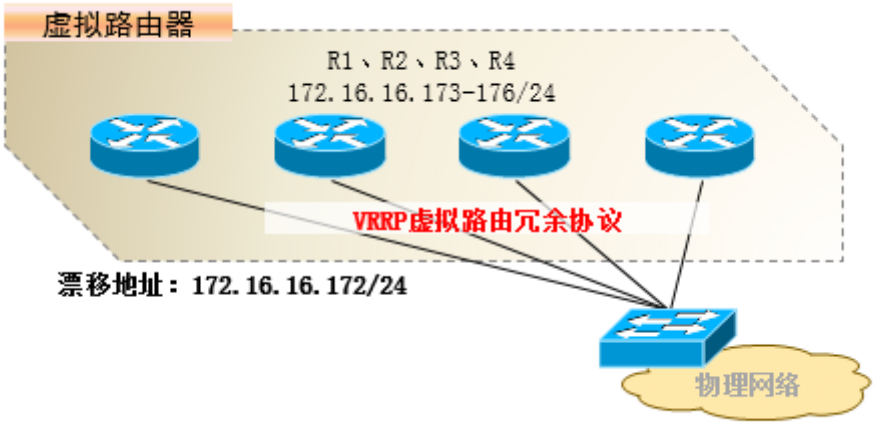
支持节点健康状态检查（Health Checking）

官方网站：http://www.keepalived.org/

② 软件实现原理

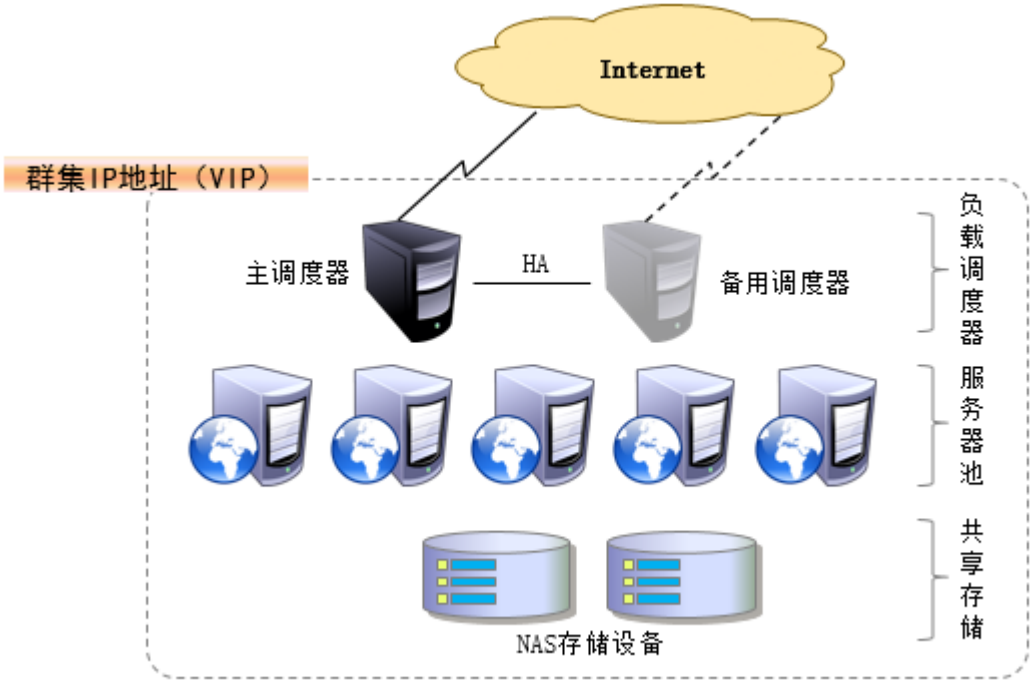
VRRP（Virtual Router Redundancy Protocol，虚拟路由冗余协议）

一主 + 多备，共用同一个 IP 地址，但优先级不同



2、Keepalived + LVS 高可用实验构建

① 实验构建设计图



② 实验构建代码

构建前提

先构建 LVS-DR 模式的负载均衡集群，可参照上文进行构建

负载调度器-1

```
yum -y install kernel-devel openssl-devel popt-devel gcc*          # 安装相关 keepalived 依赖
tar -zxf keepalived.....      # 源码安装 Keepalived 软件
cd keep.....
./configure --prefix=/ --with-kernel-dir=/usr /src/kernels/2.6.32...../
make
make install
chkconfig --add keepalived       # 设置 Keepalived 开机自启
chkconfig keepalived on
vi /etc/keepalived/keepalived.conf       # 修改 Keepalived 软件配置
global_defs {
   router_id R1       #命名主机名
```

```
}
vrrp_instance VI--1 {
    state MASTER        # 设置服务类型主/从（MASTER/SLAVE）
    interface eth0      # 指定那块网卡用来监听
    virtual_router_id 66      # 设置组号， 如果是一组就是相同的 ID 号， 一个主里面只能有一个主
服务器和多个从服务器
    priority 100      # 服务器优先级， 主服务器优先级高
    advert_int 1      # 心跳时间， 检测对方存活
    authenticetion    {      # 存活验证密码
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.1.100       #设置集群地址
    }
}
virtual_server 192.168.1.100 80 {       # 设置集群地址 以及端口号
    delay_loop 6       # 健康检查间隔
    lb_algorr       # 使用轮询调度算法
    lb_kind DR    # DR 模式的群集
    protocol TCP       # 使用的协议
    real_server 192.168.1.2 80 {      # 管理的网站节点以及使用端口
        weight 1       # 权重， 优先级 在原文件基础上删除修改
        TCP_CHECK {      # 状态检查方式
            connect_port 80      # 检查的目标端口
            connect_timeout 3       # 连接超时（秒）
            nb_get_retry 3       # 重试次数
            delay_before_retry 4       # 重试间隔（秒）
        }
    }
    real_server 192.168.1.3 80 {       # 管理的第二个网站节点以及使用端口
        weight 1       # 权重， 优先级 在原文件基础上删除修改
        TCP_CHECK {      # 状态检查方式
            connect_port 80       # 检查的目标端口
            connect_timeout 3       # 连接超时（秒）
            nb_get_retry 3       # 重试次数
            delay_before_retry 4       # 重试间隔（秒）
        }
    }
}
* 多余删除
scp keepalived.conf xx.xx.xx.xx：  /etc/keepalived/
```

负载调度器-2

yum -y install kernel-devel openssl-devel popt-devel gcc*　　# 安装相关 keepalived 依赖
tar -zxf keepalived.....　　# 源码安装 Keepalived 软件
cd keep.....
./configure --prefix=/ --with-kernel-dir=/usr /src/kernels/2.6.32...../
make
make install
chkconfig --add keepalived　　# 设置 Keepalived 开机自启
chkconfig keepalived on
修改从 负载调度器-1 拷贝的 Keepalived 配置文件 vi /etc/keepalived/keepalived.conf
　　修改 1：state MASTER　修改至 state SLAVE
　　修改 2：priority 100 修改至 priority 47　　一般建议与主服务器差值为 50
service NetworkManager stop　　# 启动虚拟借口，必须关闭此服务
cd /etc/sysconfig/network-scripts/
cp ifcfg-eth0 ifcfg-eth0:0
vim ifcfg-eth0:0　　# 配置虚拟借口
　DEVICE=eth0:0IPADDR=虚拟 IP
　NETMASK=255.255.255.0
ifup eth0:0　　# 启动虚拟网卡
vi /etc/sysconfig/network-script/ifup-eth　　# 如果 报错修改文件 257
注释此区域
vim /etc/sysctl.conf 修改内核参数。 防止相同网络地址广播冲突， 如果有多快网卡需要设置多行
　　net.ipv4.conf.eth0.send_redirects = 0
　　net.ipv4.conf.all.send_redirects = 0
　　net.ipv4.conf.default.send_redirects = 0
　　net.ipv4.conf.eth0.send_redirects = 0
sysctl -p 刷新内核参数
modprobe ip_vs 查看内核是否加载， 无法应则以加载
cat /proc/net/ip_vs 参看版本， 确认知否正确加载
cd /mnt/cdrom/Packages/ 进入光盘挂载目录
rpm -ivh ipvsadm-1.26l..... 安装 ipvsadm 管理工具
ipvsadm -v
ipvsadm -A -t 虚拟 IP:80 -s rr
ipvsadm -Ln 查看设置的 ipvsadm 如果没有子项， 那么手动添加
ipvsadm -a -t 虚拟 IP:80 -r 网站 1:80 -g
ipvsadm -a -t 虚拟 IP:80 -r 网站 2:80 -g

3、HeartBeat + Nginx 实验构建

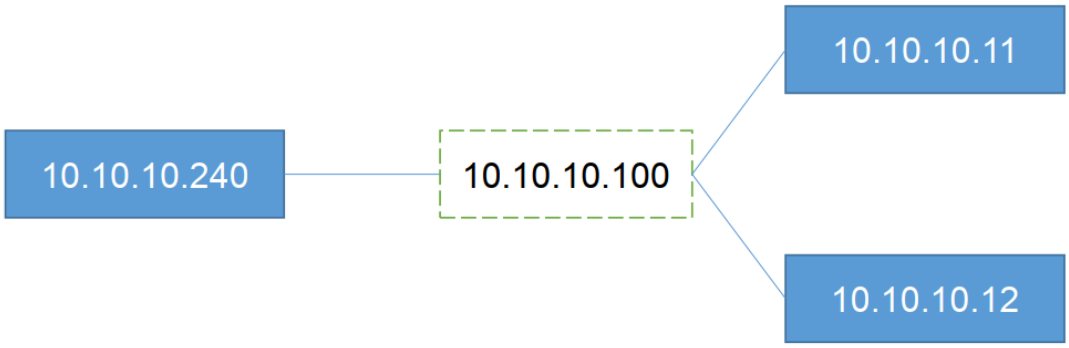
① 实验构建说明

软件包：软件包版本为 Centos6 系列，如果使用其它版本可以配置 eperl 源下载安装

环境准备

配置时间同步服务
配置主机名解析

实验拓扑结构



② 实验代码构建.

1）基础准备，准备节点都需要安装

```
tar -zxvf hearbeat.tar.gz
cd hearbeat
yum -y install *
cd /usr/share/doc/heartbeat-3.0.4/
cp ha.cf authkeys haresources /etc/ha.d/  配置文件需拷贝到默认目录下
```

2）认证服务，节点之间的认证配置，修改 /etc/ha.d/authkeys ，在主上修改

```
dd if=/dev/random bs=512 count=1 | openssl md5 #生成密钥随机数
vim authkeys
auth 1
1 md5 a4d20b0dd3d5e35e0f87ce4266d1dd64
chmod 600 authkeys
```

3）heartbeat 主配置文件，修改 /etc/ha.d/ ha.cf ， 在主上修改

```
bcast eth0
node www.centos1.com  一主一备节点，需注意能后被两台主机之间解析
```

node www.centos2.com

4) 配置 haresources 文件，在主上修改

www.centos1.com IPaddr::10.10.10.100/24/eth0:0

5) 将主三个配置文件拷贝到从上

cd /etc/ha.d/

scp ha.cf authkeys haresources root@www.centos1.com:/etc/ha.d/

6) 启动服务进行验证

主：service httpd start

主：service heartbeat start

备：service httpd start

备：service heartbeat start