原创

Linux 高可用（HA）集群之keepalived详解

 陈明乾 关注

2013-08-22 16:18:34 115689人阅读 7人评论

大纲

一、前言
二、Keepalived 详解
三、环境准备
四、LVS+Keepalived 实现高可用的前端负载均衡器

一、前言

这篇文章是前几篇文章的总结，我们先简单的总结一下我们前面讲解的内容，前面我们讲解了，LVS（负载均衡器）、Heartbeat、Corosync、Pacemaker、Web高可用集群、MySQL高可用集群、DRDB、iscsi、gfs2、cLVM等，唯一没有讲解的就是LVS可用，也就是前端高可用，我们这一篇博文主要讲解内容。在说这个之前我们得和大家讨论一个问题，也是好多博友问的问题。Heartbeat、Corosync、Keepalived这三个集群组件我们到底选哪个好，首先我想说明的是，Heartbeat、Corosync是属于同一类型，Keepalived与Heartbeat、Corosync，根本不是同一类型的。Keepalived使用的vrrp协议方式，虚拟路由冗余协议 (Virtual Router Redundancy Protocol，简称VRRP)；Heartbeat或Corosync是基于主机或网络服务的高可用方式；简单的说就是，Keepalived的目的是模拟路由器的高可用，Heartbeat或Corosync的目的是实现Service的高可用。所

结一下，Keepalived中实现轻量级的高可用，一般用于前端高可用，且不需要共享存储，一般常用于两个节点的高可用。而Heartbeat(或Corosync)一般用于服务的高可用，且需要共享存储，一般用于多节点的高可用。这个问题我们说明白了，又有博友会问了，那heartbaet与corosync我们又应该选择哪个好啊，我想说我们一般用corosync，因为corosync的运行机制更优于heartbeat，就连从heartbeat分离出来的pacemaker都说在以后的开发当中更倾向于corosync，所以现在corosync+pacemaker是最佳组合。但说实话我对于软件没有任何倾向性，所以我把所有的集群软件都和大家说了一下，我认为不管什么软件，只要它能存活下来都有它的特点和应用领域，只有把特定的软件放在特定的位置才能发挥最大的作用，那首先我们得对这个软件有所有了解。学习一种软件的最好方法，就是去查官方文档。好了说了那么多希望大家有所收获，下面我们来说一说keepalived。

二、Keepalived 详解

1.Keepalived 定义

Keepalived 是一个基于VRRP协议来实现的LVS服务高可用方案，可以利用其来避免单点故障。一个LVS服务会有2台服务器运行Keepalived，一台为主服务器（MASTER），一台为备份服务器（BACKUP），但是对外表现为一个虚拟IP，主服务器会发送特定的消息给备份服务器，当备份服务器收不到这个消息的时候，即主服务器宕机的时候， 备份服务器就会接管虚拟IP，继续提供服务，从而保证了高可用性。Keepalived是VRRP的完美实现，因此在介绍keepalived之前，先介绍一下VRRP的原理。

2.VRRP 协议简介

在现实的网络环境中，两台需要通信的主机大多数情况下并没有直接的物理连接。对于这样的情况，它们之间路由怎样选择？主机如何选定到达目的主机的下一跳路由，这个问题通常的解决方法有二种：

- 在主机上使用动态路由协议(RIP、OSPF等)
- 在主机上配置静态路由

很明显，在主机上配置动态路由是非常不切实际的，因为管理、维护成本以及是否支持等诸多问题。配置静态路由就变得十分流行，但路由器(或者说默认网关default gateway)却经常成为单点故障。VRRP的目的就是为了解决静态路由单点故障问题，VRRP通过一竞选(election)协议来动态的将路由任务交给LAN中虚拟路由器中的某台VRRP路由器。

3.VRRP 工作机制

在一个VRRP虚拟路由器中，有多台物理的VRRP路由器，但是这多台的物理的机器并不能同时工作，而是由一台称为MASTER的负责路由工作，其它的都是BACKUP，MASTER并非一成不变，VRRP让每个VRRP路由器参与竞选，最终获胜的

就是MASTER。MASTER拥有一些特权，比如，拥有虚拟路由器的IP地址，我们的主机就是用这个IP地址作为静态路由的。拥有特权的MASTER要负责转发发送给网关地址的包和响应ARP请求。

VRRP通过竞选协议来实现虚拟路由器的功能，所有的协议报文都是通过IP多播(multicast)包(多播地址224.0.0.18)形式发送的。虚拟路由器由VRID(范围0-255)和一组IP地址组成，对外表现为一个周知的MAC地址。所以，在一个虚拟路由 器中，不管谁是MASTER，对外都是相同的MAC和IP(称之为VIP)。客户端主机并不需要因为MASTER的改变而修改自己的路由配置，对客户端来说，这种主从的切换是透明的。

在一个虚拟路由器中，只有作为MASTER的VRRP路由器会一直发送VRRP通告信息(VRRPAdvertisement message)，BACKUP不会抢占MASTER，除非它的优先级(priority)更高。当MASTER不可用时(BACKUP收不到通告信息)， 多台BACKUP中优先级最高的这台会被抢占为MASTER。这种抢占是非常快速的(<1s)，以保证服务的连续性。由于安全性考虑，VRRP包使用了加密协议进行加密。

4.VRRP 工作流程

(1).初始化：

路由器启动时，如果路由器的优先级是255(最高优先级，路由器拥有路由器地址)，要发送VRRP通告信息，并发送广播ARP信息通告路由器IP地址对应的MAC地址为路由虚拟MAC，设置通告信息定时器准备定时发送VRRP通告信息，转为MASTER状态；否则进入BACKUP状态，设置定时器检查定时检查是否收到MASTER的通告信息。

(2).Master

- 设置定时通告定时器；
- 用VRRP虚拟MAC地址响应路由器IP地址的ARP请求；
- 转发目的MAC是VRRP虚拟MAC的数据包；
- 如果是虚拟路由器IP的拥有者，将接受目的地址是虚拟路由器IP的数据包，否则丢弃；
- 当收到shutdown的事件时删除定时通告定时器，发送优先权级为0的通告包，转初始化状态；
- 如果定时通告定时器超时时，发送VRRP通告信息；
- 收到VRRP通告信息时，如果优先权为0，发送VRRP通告信息；否则判断数据的优先级是否高于本机，或相等而且实际IP地址大于本地实际IP，设置定时通告定时器，复位主机超时定时器，转BACKUP状态；否则的话，丢弃该通告包；

(3).Backup

- 设置主机超时定时器；

- 不能响应针对虚拟路由器IP的ARP请求信息；
- 丢弃所有目的MAC地址是虚拟路由器MAC地址的数据包；
- 不接受目的是虚拟路由器IP的所有数据包；
- 当收到shutdown的事件时删除主机超时定时器，转初始化状态；
- 主机超时定时器超时的时候，发送VRRP通告信息，广播ARP地址信息，转MASTER状态；
- 收到VRRP通告信息时，如果优先权为0，表示进入MASTER选举；否则判断数据的优先级是否高于本机，如果高的话承认MASTER有效，复位主机超时定时器；否则的话，丢弃该通告包；

5.ARP查询处理

当内部主机通过ARP查询虚拟路由器IP地址对应的MAC地址时，MASTER路由器回复的MAC地址为虚拟的VRRP的MAC地址，而不是实际网卡的 MAC地址，这样在路由器切换时让内网机器觉察不到；而在路由器重新启动时，不能主动发送本机网卡的实际MAC地址。如果虚拟路由器开启的ARP代理 (proxy_arp)功能，代理的ARP回应也回应VRRP虚拟MAC地址；好了VRRP的简单讲解就到这里，我们下来讲解一下Keepalived的案例。

三、环境准备

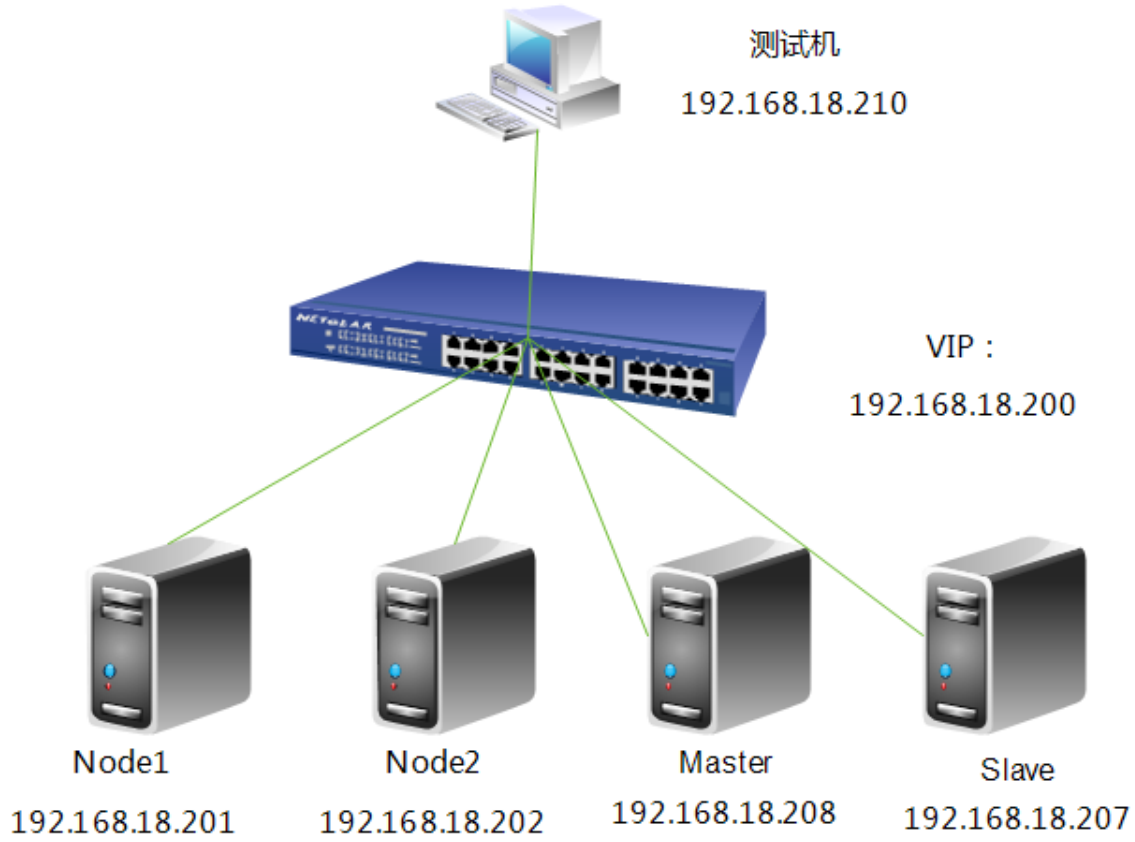
1.操作系统

- CentOS 6.4 X86_64

2.软件版本

- ipvsadm.x86_64 0:1.25-10.el6
- keepalived.x86_64 0:1.2.7-3.el6
- httpd-2.2.15-29.el6.centos.x86_64

3.实验拓扑



4.时间同步

node1:

```
[root@node1 ~]# ntpdate 202.120.2.101
```

node2:

```
[root@node2 ~]# ntpdate 202.120.2.101
```

master:

```
[root@master ~]# ntpdate 202.120.2.101
```

slave:

```
[root@slave ~]# ntpdate 202.120.2.101
```

5.主机名互相解析

node1:

```
[root@node1 ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.18.201    node1.test.com    node1
192.168.18.202    node2.test.com    node2
```

node2:

```
[root@node2 ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.18.201    node1.test.com    node1
192.168.18.202    node2.test.com    node2
```

6.安装yum源

node1:

```
[root@node1 ~]# rpm -ivh http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
[root@node1 ~]# rpm -ivh http://elrepo.org/elrepo-release-6-5.el6.elrepo.noarch.rpm
```

node2:

```
[root@node2 ~]# rpm -ivh http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
[root@node2 ~]# rpm -ivh http://elrepo.org/elrepo-release-6-5.el6.elrepo.noarch.rpm
```

master:

```
[root@master ~]# rpm -ivh http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
```

```
[root@master ~]# rpm -ivh http://elrepo.org/elrepo-release-6-5.el6.elrepo.noarch.rpm
```

slave:

```
[root@slave ~]# rpm -ivh http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
[root@slave ~]# rpm -ivh http://elrepo.org/elrepo-release-6-5.el6.elrepo.noarch.rpm
```

四、LVS+Keepalived 实现高可用的前端负载均衡器

node1:

1.安装httpd

```
[root@node1 ~]# yum install -y httpd
```

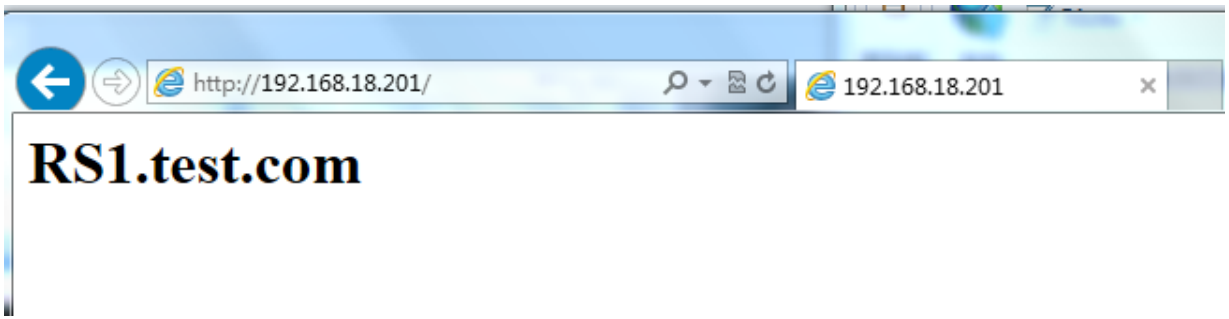
2.配置httpd

```
[root@node1 ~]# vim /var/www/html/index.html
<h1>RS1.test.com</h1>
```

3.启动httpd

```
[root@node1 ~]# service httpd start
```

4.测试



5.设置开机自启动

```
[root@node1 ~]# chkconfig httpd on
[root@node1 ~]# chkconfig httpd --list
httpd              0:关闭    1:关闭    2:启用    3:启用    4:启用    5:启用    6:关闭
```

6.配置node1

```
[root@node1 ~]# mkdir src
[root@node1 ~]# cd src/
[root@node1 src]# vim realserver.sh
#!/bin/bash
#
# Script to start LVS DR real server.
# description: LVS DR real server
#
. /etc/rc.d/init.d/functions
VIP=192.168.18.200 #修改你的VIP
host=`/bin/hostname`
case "$1" in
start)
        # Start LVS-DR real server on this machine.
         /sbin/ifconfig lo down
         /sbin/ifconfig lo up
         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
         /sbin/ifconfig lo:0 $VIP broadcast $VIP netmask 255.255.255.255 up
         /sbin/route add -host $VIP dev lo:0
;;
stop)
         # Stop LVS-DR real server loopback device(s).
         /sbin/ifconfig lo:0 down
         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
;;
status)
        # Status of LVS-DR real server.
        islothere=`/sbin/ifconfig lo:0 | grep $VIP`
        isrothere=`netstat -rn | grep "lo:0" | grep $VIP`
        if [ ! "$islothere" -o ! "isrothere" ];then
            # Either the route or the lo:0 device
            # not found.
            echo "LVS-DR real server Stopped."
        else
            echo "LVS-DR real server Running."
        fi
;;
*)
            # Invalid entry.
            echo "$0: Usage: $0 {start|status|stop}"
            exit 1
;;
esac
[root@node1 src]# chmod +x realserver.sh
[root@node1 src]# ll
总用量 4
-rwxr-xr-x 1 root root 1485 8月  22 10:18 realserver.sh
[root@node1 src]# ./realserver.sh start
```

7.查看配置

```
[root@node1 src]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:11:92:E4
          inet addr:192.168.18.201  Bcast:192.168.18.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe11:92e4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:115061 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14979 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

```
          RX bytes:43448483 (41.4 MiB)  TX bytes:1224926 (1.1 MiB)
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:168 (168.0 b)  TX bytes:168 (168.0 b)
lo:0      Link encap:Local Loopback
          inet addr:192.168.18.200  Mask:255.255.255.255
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
[root@node1 src]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.18.200  0.0.0.0         255.255.255.255 UH    0      0        0 lo
192.168.18.0    0.0.0.0         255.255.255.0   U     0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
0.0.0.0         192.168.18.254  0.0.0.0         UG    0      0        0 eth0
[root@node1 src]# cat /proc/sys/net/ipv4/conf/lo/arp_ignore
1
[root@node1 src]# cat /proc/sys/net/ipv4/conf/lo/arp_announce
2
[root@node1 src]# cat /proc/sys/net/ipv4/conf/all/arp_ignore
1
[root@node1 src]# cat /proc/sys/net/ipv4/conf/all/arp_announce
2
```

好了，node1到这里基本配置完成，下面我们来配置node2。

node2:

1.安装httpd

```
[root@node2 ~]# yum install -y httpd
```

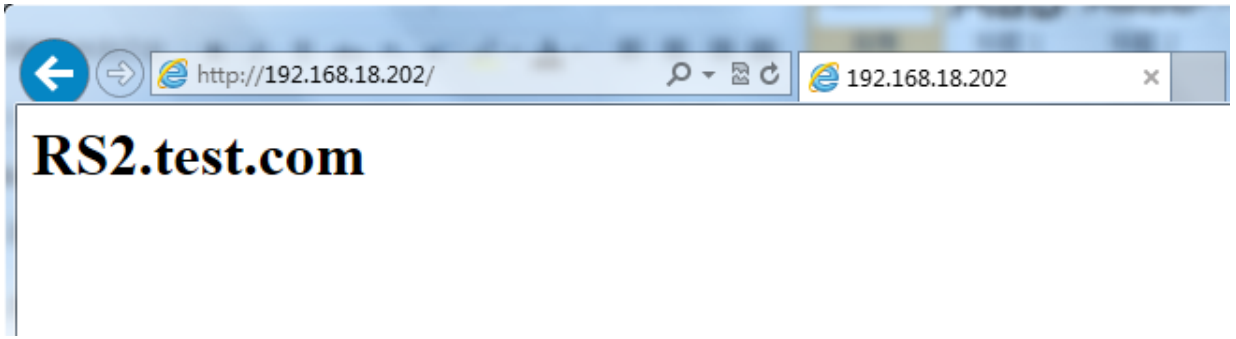
2.配置httpd

```
[root@node2 ~]# vim /var/www/html/index.html
<h1>RS2.test.com</h1>
```

3.启动httpd

```
[root@node2 ~]# service httpd start
```

4.测试



5.设置开机自启动

```
[root@node2 ~]# chkconfig httpd on
[root@node2 ~]# chkconfig httpd --list
httpd                   0:关闭    1:关闭    2:启用    3:启用    4:启用    5:启用    6:关闭
```

6.配置node2

```
[root@node2 ~]# mkdir src
[root@node2 ~]# cd src/
[root@node2 src]# vim realserver.sh
#!/bin/bash
#
# Script to start LVS DR real server.
# description: LVS DR real server
#
. /etc/rc.d/init.d/functions
```

```
VIP=192.168.18.200
host=`/bin/hostname`
case "$1" in
start)
        # Start LVS-DR real server on this machine.
         /sbin/ifconfig lo down
         /sbin/ifconfig lo up
         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
         /sbin/ifconfig lo:0 $VIP broadcast $VIP netmask 255.255.255.255 up
         /sbin/route add -host $VIP dev lo:0
;;
stop)
         # Stop LVS-DR real server loopback device(s).
         /sbin/ifconfig lo:0 down
         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
;;
status)
         # Status of LVS-DR real server.
         islothere=`/sbin/ifconfig lo:0 | grep $VIP`
         isrothere=`netstat -rn | grep "lo:0" | grep $VIP`
         if [ ! "$islothere" -o ! "isrothere" ];then
              # Either the route or the lo:0 device
              # not found.
              echo "LVS-DR real server Stopped."
         else
              echo "LVS-DR real server Running."
         fi
;;
*)
```

```
                # Invalid entry.
                echo  "$0: Usage: $0 {start|status|stop}"
                exit  1
        ;;
esac
[root@node2 src]# chmod +x realserver.sh
[root@node2 src]# ./realserver.sh start
```

7.查看配置

```
[root@node2 src]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:B8:DF:6A
          inet addr:192.168.18.202  Bcast:192.168.18.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:feb8:df6a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:110545 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10461 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:32853789 (31.3 MiB)  TX bytes:889109 (868.2 KiB)
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:616 (616.0 b)  TX bytes:616 (616.0 b)
lo:0      Link encap:Local Loopback
          inet addr:192.168.18.200  Mask:255.255.255.255
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
[root@node2 src]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.18.200  0.0.0.0         255.255.255.255 UH    0      0        0 lo
192.168.18.0    0.0.0.0         255.255.255.0   U     0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
0.0.0.0         192.168.18.254  0.0.0.0         UG    0      0        0 eth0
[root@node2 src]# cat /proc/sys/net/ipv4/conf/lo/arp_ignore
```

```
1
[root@node2 src]# cat /proc/sys/net/ipv4/conf/lo/arp_announce
2
[root@node2 src]# cat /proc/sys/net/ipv4/conf/all/arp_ignore
1
[root@node2 src]# cat /proc/sys/net/ipv4/conf/all/arp_announce
2
```

好了，到这里node2也基本配置完成。下面我们来配置master与slave。

masterg与slave:

1.安装keepalived与ipvsadm

```
[root@master ~]# yum install -y keepalived ipvsadm
[root@slave ~]# yum install -y keepalived ipvsadm
```

2.修改配置文件

```
[root@master ~]# cat /etc/keepalived/keepalived.conf
! Configuration File for keepalived
global_defs {
   notification_email {
15251076067@163.com #配置管理员邮箱
   }
   notification_email_from root #配置发件人
   smtp_server 127.0.0.1 #配置邮件服务器
   smtp_connect_timeout 30
   router_id LVS_DEVEL
}
vrrp_instance VI_1 {
    state MASTER #配置模式
    interface eth0
    virtual_router_id 51
    priority 101 #配置优先级
    advert_int 1
    authentication {
```

```
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.18.200 #配置虚拟IP地址
    }
}
virtual_server 192.168.18.200 80 {
    delay_loop 6
    lb_algo rr
    lb_kind DR
    nat_mask 255.255.255.0
    #persistence_timeout 50
    protocol TCP
    real_server 192.168.18.201 80 { #配置realaserver
        weight 1
        HTTP_GET { #监控配置
            url {
              path /
          status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    real_server 192.168.18.202 80 {
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
```

```
    }
}
```

3.将配置文件同步到slave

```
[root@master ~]# scp /etc/keepalived/keepalived.conf root@192.168.18.207:/etc/keepalived/
```

4.简单修改一下slave配置文件

```
[root@slave keepalived]# cat keepalived.conf
! Configuration File for keepalived
global_defs {
   notification_email {
     15251076067@163.com
   }
   notification_email_from root
   smtp_server 127.0.0.1
   smtp_connect_timeout 30
   router_id LVS_DEVEL
}
vrrp_instance VI_1 {
    state BACKUP #修改为BACKUP
    interface eth0
    virtual_router_id 51
    priority 100 #修改优先级
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.18.200
    }
}
virtual_server 192.168.18.200 80 {
    delay_loop 6
```

```
    lb_algo rr
    lb_kind DR
    nat_mask 255.255.255.0
    #persistence_timeout 50
    protocol TCP
    real_server 192.168.18.201 80 {
        weight 1
        HTTP_GET {
            url {
              path /
          status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    real_server 192.168.18.202 80 {
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
}
```

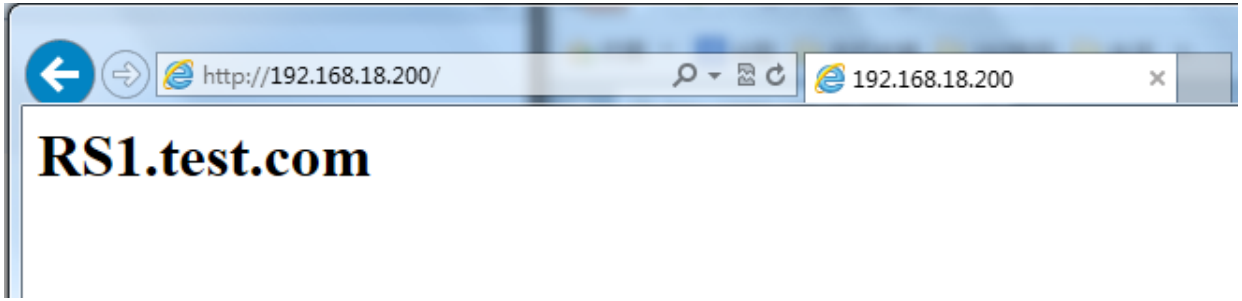
5.启动master与slave的keepalived服务

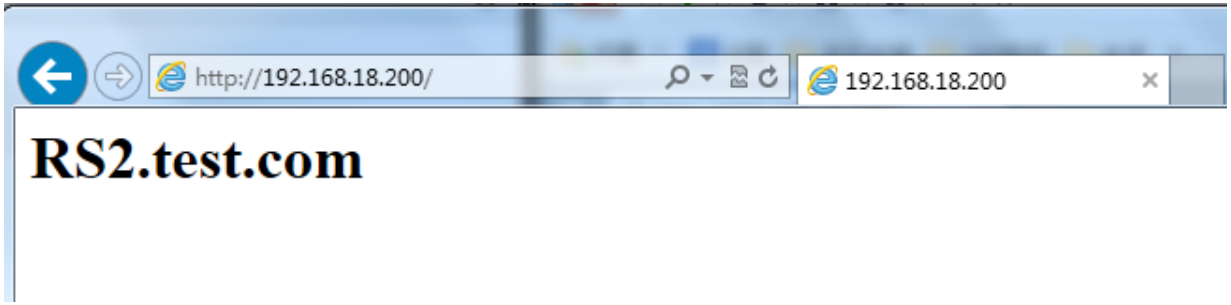
```
[root@master ~]# service keepalived start
正在启动 keepalived:                                       [确定]
[root@slave ~]# service keepalived start
正在启动 keepalived:                                       [确定]
```

6.查看一下LVS状态

```
[root@master ~]# ipvsadm -L -n
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.18.200:80 rr
  -> 192.168.18.201:80            Route   1      0          0
  -> 192.168.18.202:80            Route   1      0          0
```

7.测试





8.模拟故障

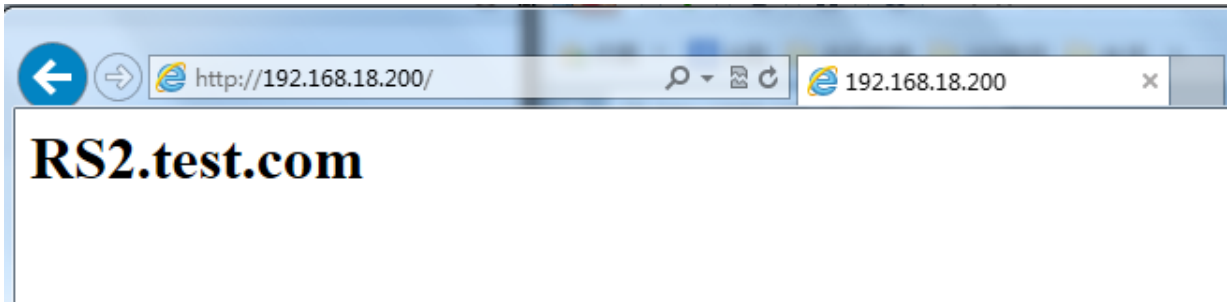
(1).停止一下node1

```
[root@node1 src]# service httpd stop
停止 httpd:                                                [确定]
```

(2).查看一下的lvs

```
[root@master ~]# ipvsadm -L -n
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.18.200:80 rr
  -> 192.168.18.202:80            Route   1      0          0
```

(3).测试一下



(4).查看一下邮件

(5).重新启动一下node1

```
[root@node1 src]# service httpd start
正在启动 httpd:                                            [确定]
```

(6).再查看一下lvs状态

```
[root@master ~]# ipvsadm -L -n
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.18.200:80 rr
```

```
  -> 192.168.18.201:80            Route   1      0          0
  -> 192.168.18.202:80            Route   1      0          0
```

(7).再查看一下邮件

(8).关闭master上keepalived

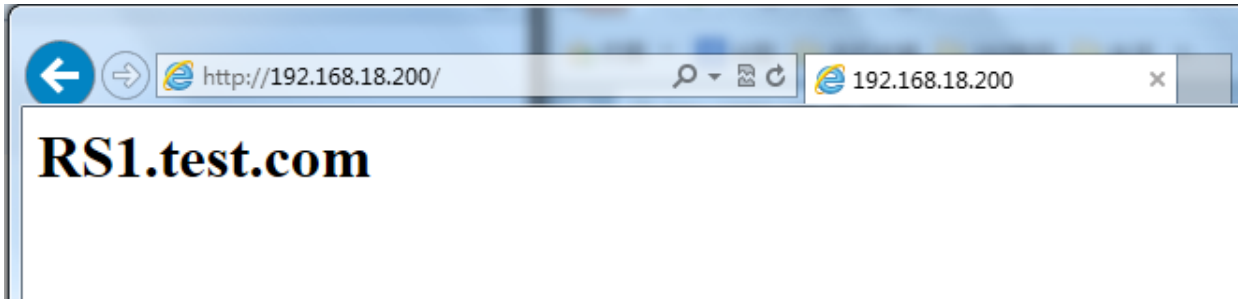
```
[root@master ~]# service keepalived stop
停止 keepalived:                                           [确定]
[root@master ~]# ipvsadm -L -n
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
```

(9).查看一下slave状态

```
[root@slave ~]# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    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 qlen 1000
    link/ether 00:0c:29:f9:e6:26 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.207/24 brd 192.168.18.255 scope global eth0
    inet 192.168.18.200/32 scope global eth0
    inet6 fe80::20c:29ff:fef9:e626/64 scope link
       valid_lft forever preferred_lft forever
[root@slave ~]# ipvsadm -L -n
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.18.200:80 rr
  -> 192.168.18.201:80            Route   1      0          0
```

```
  -> 192.168.18.202:80            Route   1      0          0
```

(10).再次测试一下



注，大家可以看到，经过上面的演示我们现在LVS的高可用即前端负载均衡的高可用，同时实现对后端realserver监控，也实现后端realserver宕机时会给管理员发送邮件。但还有几个问题我们还没有解决，问题如下:

- 所有realserver都down机，怎么处理？是不是用户就没法打开，还是提供一下维护页面。
- 怎么完成维护模式keepalived切换？
- 如何在keepalived故障时，发送警告邮件给指定的管理员？

9.所有realserver都down机，怎么处理？

问题：在集群中如果所有real server全部宕机了，客户端访问时就会出现错误页面，这样是很不友好的，我们得提供一个维护页面来提醒用户，服务器正在维护，什么时间可以访问等，下面我们就来解决一下这个问题。解决方案有两种，一种是提供一台备用的real server当所有的服务器宕机时，提供维护页面，但这样做有点浪费服务器。另一种就是在负载均衡器上提供维护页面，这样是比较靠谱的，也比较常用。下面我们就来具体操作一下。

(1).master与slave安装上httpd

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

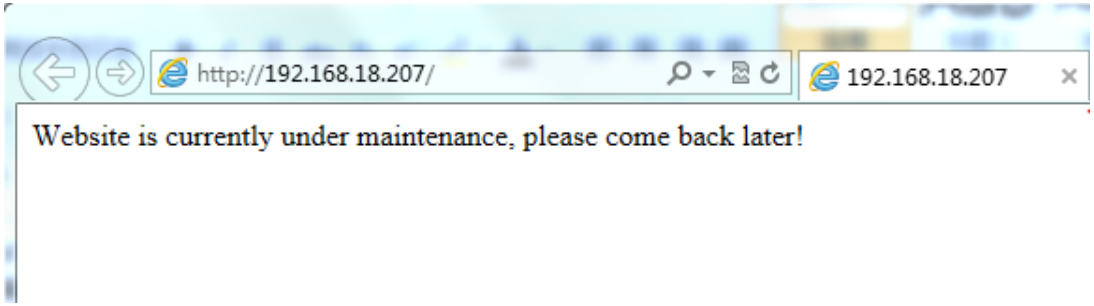
(2).配置维护页面

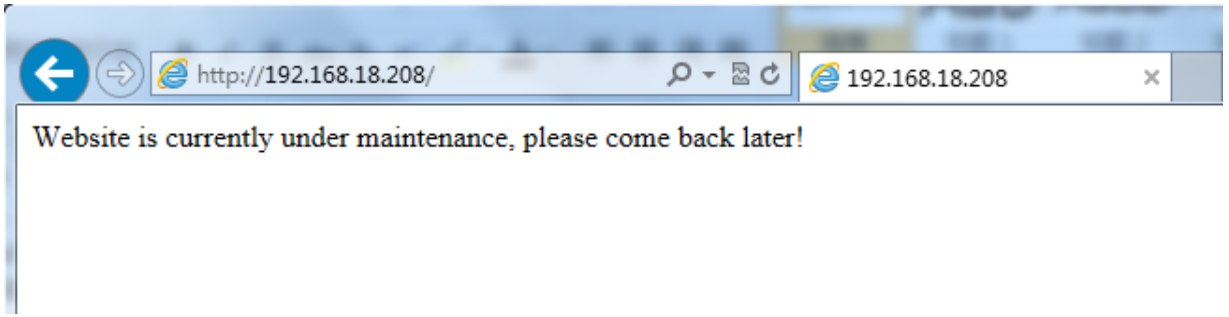
```
[root@master ~]# vim /var/www/html/index.html
Website is currently under maintenance, please come back later!
 [root@slave ~]# vim /var/www/html/index.html
```

```
Website is currently under maintenance, please come back later!
```

(3).启动httpd服务并测试

```
[root@master ~]# service httpd start
正在启动 httpd:                                            [确定]
[root@slave ~]# service httpd start
正在启动 httpd:                                            [确定]
```





(4).修改配置文件

master:

```
[root@master ~]# cat /etc/keepalived/keepalived.conf
! Configuration File for keepalived
global_defs {
   notification_email {
     15251076067@163.com
   }
```

```
   notification_email_from root
   smtp_server 127.0.0.1
   smtp_connect_timeout 30
   router_id LVS_DEVEL
}
vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 51
    priority 101
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.18.200
    }
}
virtual_server 192.168.18.200 80 {
    delay_loop 6
    lb_algo rr
    lb_kind DR
    nat_mask 255.255.255.0
    protocol TCP
    real_server 192.168.18.201 80 {
        weight 1
        HTTP_GET {
            url {
              path /
          status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    real_server 192.168.18.202 80 {
```

```
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    sorry_server 127.0.0.1 80    #增加一行sorry_server
}
```

slave:

```
[root@slave ~]# cat /etc/keepalived/keepalived.conf
! Configuration File for keepalived
global_defs {
   notification_email {
     15251076067@163.com
   }
   notification_email_from root
   smtp_server 127.0.0.1
   smtp_connect_timeout 30
   router_id LVS_DEVEL
}
vrrp_instance VI_1 {
    state BACKUP
    interface eth0
    virtual_router_id 51
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
```

```
    virtual_ipaddress {
        192.168.18.200
    }
}
virtual_server 192.168.18.200 80 {
    delay_loop 6
    lb_algo rr
    lb_kind DR
    nat_mask 255.255.255.0
    protocol TCP
    real_server 192.168.18.201 80 {
        weight 1
        HTTP_GET {
            url {
              path /
          status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    real_server 192.168.18.202 80 {
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    sorry_server 127.0.0.1 80 #增加一行sorry_server
}
```

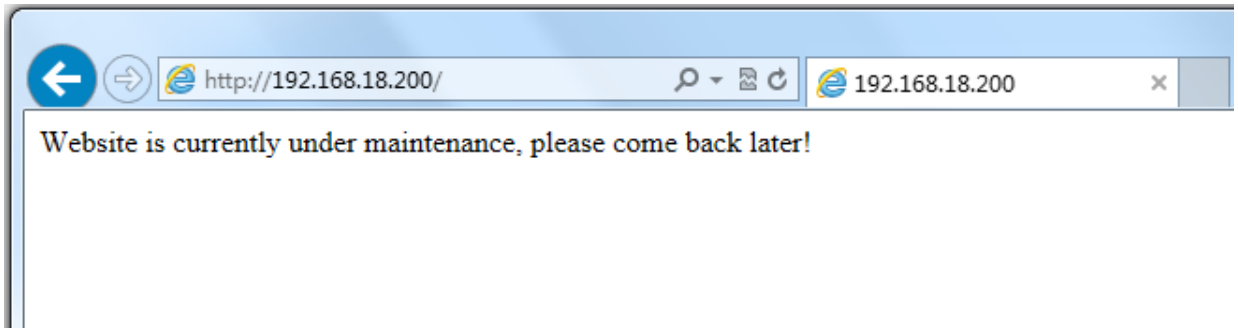
(5).关闭所有的real server并重新启动一下master与slave的keepalived

```
[root@node1 ~]# service httpd stop
停止 httpd:                                                [确定]
[root@node2 ~]# service httpd stop
停止 httpd:                                                [确定]
[root@master ~]# service keepalived restart
停止 keepalived:                                           [确定]
正在启动 keepalived:                                       [确定]
[root@slave ~]# service keepalived restart
停止 keepalived:                                           [确定]
正在启动 keepalived:                                       [确定]
```

(6).查看一下lvs

```
[root@master ~]# ipvsadm -L -n
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.168.18.200:80 rr
  -> 127.0.0.1:80                 Local   1      0          0
```

(7).测试



注，sorry_server测试成功，下面我们继续。

10.怎么完成维护模式keepalived切换？

问题：我们一般进行主从切换测试时都是关闭keepalived或关闭网卡接口，有没有一种方法能实现在不关闭keepalived下或网卡接口来实现维护呢？方法肯定是有的，在keepalived新版本中，支持脚本vrrp_srcipt，具体如何使用大家可以man keepalived.conf查看。下面我们来演示一下具体怎么实现。

(1).定义脚本

```
vrrp_srcipt chk_schedown {
    script "[ -e /etc/keepalived/down ] && exit 1 || exit 0"
    interval 1 #监控间隔
    weight -5 #减小优先级
    fall 2 #监控失败次数
    rise 1 #监控成功次数
}
```

(2).执行脚本

```
track_script {
    chk_schedown #执行chk_schedown脚本
}
```

(3).修改配置文件

master:

```
[root@master ~]# cat /etc/keepalived/keepalived.conf
! Configuration File for keepalived
global_defs {
    notification_email {
      15251076067@163.com
    }
    notification_email_from root
    smtp_server 127.0.0.1
    smtp_connect_timeout 30
    router_id LVS_DEVEL
}
vrrp_script chk_schedown { #定义vrrp执行脚本
```

```
    script "[ -e /etc/keepalived/down ] && exit 1 || exit 0" #查看是否有down文件，有就进入维护模式
    interval 1 #监控间隔时间
    weight -5 #降低优先级
    fall 2 #失败次数
    rise 1 #成功数次
}
vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 51
    priority 101
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.18.200
    }
    track_script { #执行脚本
        chk_schedown
    }
}
virtual_server 192.168.18.200 80 {
    delay_loop 6
    lb_algo rr
    lb_kind DR
    nat_mask 255.255.255.0
    protocol TCP
    real_server 192.168.18.201 80 {
        weight 1
        HTTP_GET {
            url {
              path /
          status_code 200
            }
            connect_timeout 2
```

```
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    real_server 192.168.18.202 80 {
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
   sorry_server 127.0.0.1 80
}
```

slave:

```
[root@slave ~]# cat /etc/keepalived/keepalived.conf
! Configuration File for keepalived
global_defs {
   notification_email {
     15251076067@163.com
   }
   notification_email_from root
   smtp_server 127.0.0.1
   smtp_connect_timeout 30
   router_id LVS_DEVEL
}
vrrp_script chk_schedown {
   script "[ -e /etc/keepalived/down ] && exit 1 || exit 0"
   interval 1
   weight -5
   fall 2
```

```
   rise 1
}
vrrp_instance VI_1 {
    state BACKUP
    interface eth0
    virtual_router_id 51
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.18.200
    }
    track_script {
    chk_schedown
    }
}
virtual_server 192.168.18.200 80 {
    delay_loop 6
    lb_algo rr
    lb_kind DR
    nat_mask 255.255.255.0
    protocol TCP
    real_server 192.168.18.201 80 {
        weight 1
        HTTP_GET {
            url {
              path /
          status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
```

```
    real_server 192.168.18.202 80 {
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    sorry_server 127.0.0.1 80
```

(4).测试

master:

```
[root@master keepalived]# touch down #新建一下down文件
[root@master keepalived]# ll
总用量 4
-rw-r--r-- 1 root root    0 8月  22 13:39 down
-rw-r--r-- 1 root root 1317 8月  22 13:35 keepalived.conf
[root@master keepalived]# tail -f /var/log/messages #查看一下日志
Aug 22 13:43:52 master Keepalived_vrrp[12003]: VRRP_Instance(VI_1) Entering MASTER STATE
Aug 22 13:43:52 master Keepalived_vrrp[12003]: VRRP_Instance(VI_1) setting protocol VIPs.
Aug 22 13:43:52 master Keepalived_vrrp[12003]: VRRP_Instance(VI_1) Sending gratuitous ARPs on et
Aug 22 13:43:52 master Keepalived_vrrp[12003]: VRRP_Instance(VI_1) Received higher prio advert
Aug 22 13:43:52 master Keepalived_vrrp[12003]: VRRP_Instance(VI_1) Entering BACKUP STATE
Aug 22 13:43:52 master Keepalived_vrrp[12003]: VRRP_Instance(VI_1) removing protocol VIPs.
Aug 22 13:43:52 master Keepalived_healthcheckers[12002]: Netlink reflector reports IP 192.168.1
Aug 22 13:43:52 master Keepalived_healthcheckers[12002]: Netlink reflector reports IP 192.168.1
Aug 22 13:43:52 master Keepalived_healthcheckers[12002]: SMTP alert successfully sent.
Aug 22 13:43:52 master Keepalived_healthcheckers[12002]: SMTP alert successfully sent.
^C
[root@master keepalived]# ip add show #查看VIP
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
```

```
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    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 qlen 1000
    link/ether 00:0c:29:4b:a1:85 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.208/24 brd 192.168.18.255 scope global eth0
    inet6 fe80::20c:29ff:fe4b:a185/64 scope link
       valid_lft forever preferred_lft forever
```

slave:

```
[root@slave ~]# ip addr show #查看一下VIP已转移到slave上
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    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 qlen 1000
    link/ether 00:0c:29:f9:e6:26 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.207/24 brd 192.168.18.255 scope global eth0
    inet 192.168.18.200/32 scope global eth0
    inet6 fe80::20c:29ff:fef9:e626/64 scope link
       valid_lft forever preferred_lft forever
```

好了，自写监测脚本，完成维护模式切换，到这里就演示成功，下面我们来解决最后一个问题，就是keepalived主从切换的邮件通告。

11.如何在keepalived故障时(或主备切换时)，发送警告邮件给指定的管理员？

(1).keepalived通知脚本进阶示例

下面的脚本可以接受选项，其中

- -s, --service SERVICE,...：指定服务脚本名称，当状态切换时可自动启动、重启或关闭此服务；
- -a, --address VIP: 指定相关虚拟路由器的VIP地址；
- -m, --mode {mm|mb}：指定虚拟路由的模型，mm表示主主，mb表示主备；它们表示相对于同一种服务而方，其VIP的工

作类型；

- -n, --notify {master|backup|fault}：指定通知的类型，即vrrp角色切换的目标角色；
- -h, --help：获取脚本的使用帮助；

```
#!/bin/bash
# Author: freeloda
# description: An example of notify script
# Usage: notify.sh -m|--mode {mm|mb} -s|--service SERVICE1,... -a|--address VIP  -n|--notify {master|backup|fa
contact='1521076067@163.com'
helpflag=0
serviceflag=0
modeflag=0
addressflag=0
notifyflag=0
Usage() {
   echo  "Usage: notify.sh [-m|--mode {mm|mb}] [-s|--service SERVICE1,...] <-a|--address VIP>  <-n|--notify {ma
   echo  "Usage: notify.sh -h|--help"
}
ParseOptions() {
   local I=1;
   if [ $# -gt 0 ]; then
      while [ $I -le $# ]; do
         case  $1  in
         -s|--service)
            [ $# -lt 2 ] && return 3
             serviceflag=1
             services=(`echo $2|awk -F"," '{for(i=1;i<=NF;i++) print $i}'`)
            shift 2 ;;
         -h|--help)
             helpflag=1
            return 0
            shift
            ;;
         -a|--address)
```

```
            [ $# -lt 2 ] && return 3
            addressflag=1
            vip=$2
            shift 2
            ;;
        -m|--mode)
            [ $# -lt 2 ] && return 3
            mode=$2
            shift 2
            ;;
        -n|--notify)
            [ $# -lt 2 ] && return 3
            notifyflag=1
            notify=$2
            shift 2
            ;;
        *)
            echo "Wrong options..."
            Usage
            return 7
            ;;
         esac
      done
      return 0
   fi
}
#workspace=$(dirname $0)
RestartService() {
   if [ ${#@} -gt 0 ]; then
      for I in $@; do
         if [ -x /etc/rc.d/init.d/$I ]; then
            /etc/rc.d/init.d/$I restart
         else
            echo "$I is not a valid service..."
         fi
```

```
    done
  fi
}
StopService() {
  if [ ${#@} -gt 0 ]; then
    for I in $@; do
      if [ -x /etc/rc.d/init.d/$I ]; then
        /etc/rc.d/init.d/$I stop
      else
        echo "$I is not a valid service..."
      fi
    done
  fi
}
Notify() {
    mailsubject="`hostname` to be $1: $vip floating"
    mailbody="`date '+%F %H:%M:%S'`, vrrp transition, `hostname` changed to be $1."
    echo $mailbody | mail -s "$mailsubject" $contact
}
# Main Function
ParseOptions $@
[ $? -ne 0 ] && Usage && exit 5
[ $helpflag -eq 1 ] && Usage && exit 0
if [ $addressflag -ne 1 -o $notifyflag -ne 1 ]; then
  Usage
  exit 2
fi
mode=${mode:-mb}
case $notify in
'master')
  if [ $serviceflag -eq 1 ]; then
      RestartService ${services[*]}
  fi
  Notify master
  ;;
```

```
'backup')
    if [ $serviceflag -eq 1 ]; then
        if [ "$mode" == 'mb' ]; then
            StopService ${services[*]}
        else
            RestartService ${services[*]}
        fi
    fi
    Notify backup
    ;;
'fault')
    Notify fault
    ;;
*)
    Usage
    exit 4
    ;;
esac
```

(2).在keepalived.conf配置文件中，其调用方法如下所示：

- notify_master "/etc/keepalived/notify.sh -n master -a 192.168.18.200"
- notify_backup "/etc/keepalived/notify.sh -n backup -a 192.168.18.200"
- notify_fault "/etc/keepalived/notify.sh -n fault -a 192.168.18.200"

(3).修改配置文件

master:

```
[root@master keepalived]# cat keepalived.conf
! Configuration File for keepalived
global_defs {
   notification_email {
     15251076067@163.com
   }
```

```
   notification_email_from root
   smtp_server 127.0.0.1
   smtp_connect_timeout 30
   router_id LVS_DEVEL
}
vrrp_script chk_schedown {
   script "[ -e /etc/keepalived/down ] && exit 1 || exit 0"
   interval 1
   weight -5
   fall 2
   rise 1
}
vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 51
    priority 101
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.18.200
    }
    track_script {
        chk_schedown
    }
    #增加以下三行
    notify_master "/etc/keepalived/notify.sh -n master -a 192.168.18.200"
    notify_backup "/etc/keepalived/notify.sh -n backup -a 192.168.18.200"
    notify_fault "/etc/keepalived/notify.sh -n fault -a 192.168.18.200"
}
virtual_server 192.168.18.200 80 {
    delay_loop 6
    lb_algo rr
    lb_kind DR
```

```
    nat_mask 255.255.255.0
    protocol TCP
    real_server 192.168.18.201 80 {
        weight 1
        HTTP_GET {
            url {
              path /
            status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    real_server 192.168.18.202 80 {
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
   sorry_server 127.0.0.1 80
}
```

slave:

```
[root@slave keepalived]# cat keepalived.conf
! Configuration File for keepalived
global_defs {
   notification_email {
     15251076067@163.com
   }
```

```
   notification_email_from root
   smtp_server 127.0.0.1
   smtp_connect_timeout 30
   router_id LVS_DEVEL
}
vrrp_script chk_schedown {
   script "[ -e /etc/keepalived/down ] && exit 1 || exit 0"
   interval 1
   weight -5
   fall 2
   rise 1
}
vrrp_instance VI_1 {
    state BACKUP
    interface eth0
    virtual_router_id 51
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.18.200
    }
    track_script {
    chk_schedown
    }
    #增加以下三行
    notify_master "/etc/keepalived/notify.sh -n master -a 192.168.18.200"
    notify_backup "/etc/keepalived/notify.sh -n backup -a 192.168.18.200"
    notify_fault "/etc/keepalived/notify.sh -n fault -a 192.168.18.200"
}
virtual_server 192.168.18.200 80 {
    delay_loop 6
    lb_algo rr
    lb_kind DR
```

```
    nat_mask 255.255.255.0
    protocol TCP
    real_server 192.168.18.201 80 {
        weight 1
        HTTP_GET {
            url {
              path /
          status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    real_server 192.168.18.202 80 {
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 2
            nb_get_retry 3
            delay_before_retry 1
        }
    }
    sorry_server 127.0.0.1 80
}
```

(4).增加脚本

```
[root@slave keepalived]# pwd
/etc/keepalived
[root@slave keepalived]# vim notify.sh
[root@slave keepalived]# cat notify.sh
#!/bin/bash
# Author: freeloda
```

```
# description: An example of notify script
# Usage: notify.sh -m|--mode {mm|mb} -s|--service SERVICE1,... -a|--address VIP  -n|--notify {master|backup|fa
contact='15251076067@163.com'
helpflag=0
serviceflag=0
modeflag=0
addressflag=0
notifyflag=0
Usage() {
   echo  "Usage: notify.sh [-m|--mode {mm|mb}] [-s|--service SERVICE1,...] <-a|--address VIP>  <-n|--notify {ma
   echo  "Usage: notify.sh -h|--help"
}
ParseOptions() {
   local I=1;
   if [ $# -gt 0 ]; then
      while [ $I -le $# ]; do
        case $1 in
        -s|--service)
          [ $# -lt 2 ] && return 3
           serviceflag=1
           services=(`echo $2|awk -F"," '{for(i=1;i<=NF;i++) print $i}'`)
          shift 2 ;;
        -h|--help)
           helpflag=1
          return 0
          shift
          ;;
        -a|--address)
          [ $# -lt 2 ] && return 3
          addressflag=1
          vip=$2
          shift 2
          ;;
        -m|--mode)
          [ $# -lt 2 ] && return 3
```

```
            mode=$2
            shift 2
            ;;
          -n|--notify)
            [ $# -lt 2 ] && return 3
            notifyflag=1
            notify=$2
            shift 2
            ;;
          *)
            echo "Wrong options..."
            Usage
            return 7
            ;;
          esac
      done
      return 0
    fi
}
#workspace=$(dirname $0)
RestartService() {
    if [ ${#@} -gt 0 ]; then
      for I in $@; do
        if [ -x /etc/rc.d/init.d/$I ]; then
          /etc/rc.d/init.d/$I restart
        else
          echo "$I is not a valid service..."
        fi
      done
    fi
}
StopService() {
    if [ ${#@} -gt 0 ]; then
      for I in $@; do
        if [ -x /etc/rc.d/init.d/$I ]; then
```

```
            /etc/rc.d/init.d/$I stop
        else
            echo "$I is not a valid service..."
        fi
    done
  fi
}
Notify() {
    mailsubject="`hostname` to be $1: $vip floating"
    mailbody="`date '+%F %H:%M:%S'`, vrrp transition, `hostname` changed to be $1."
    echo $mailbody | mail -s "$mailsubject" $contact
}
# Main Function
ParseOptions $@
[ $? -ne 0 ] && Usage && exit 5
[ $helpflag -eq 1 ] && Usage && exit 0
if [ $addressflag -ne 1 -o $notifyflag -ne 1 ]; then
  Usage
  exit 2
fi
mode=${mode:-mb}
case $notify in
'master')
  if [ $serviceflag -eq 1 ]; then
      RestartService ${services[*]}
  fi
  Notify master
  ;;
'backup')
  if [ $serviceflag -eq 1 ]; then
    if [ "$mode" == 'mb' ]; then
      StopService ${services[*]}
    else
      RestartService ${services[*]}
    fi
```

```
    fi
    Notify backup
    ;;
'fault')
    Notify fault
    ;;
*)
    Usage
    exit 4
    ;;
esac
```

(5).给脚本增加执行权限

```
[root@slave keepalived]# chmod +x notify.sh
```

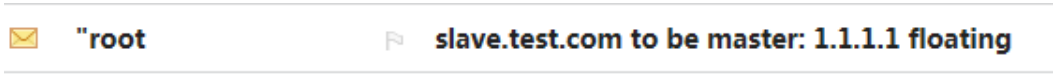
(6).将master上脚本复制到slave上

```
[root@slave keepalived]# scp -p notify.sh root@192.168.18.207:/etc/keepalived/
```

(7).测试一下脚本

```
[root@slave keepalived]# ./notify.sh -h
Usage: notify.sh [-m|--mode {mm|mb}] [-s|--service SERVICE1,...] <-a|--address VIP>  <-n|--notify {ma
Usage: notify.sh -h|--help
[root@slave keepalived]# ./notify.sh --help
Usage: notify.sh [-m|--mode {mm|mb}] [-s|--service SERVICE1,...] <-a|--address VIP>  <-n|--notify {ma
Usage: notify.sh -h|--help
[root@slave keepalived]# ./notify.sh -m mb -a 1.1.1.1 -n master
```

(8).查看一下邮件



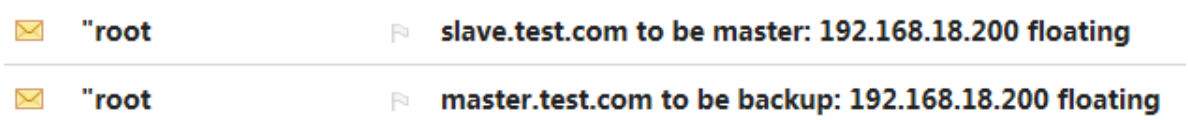
注，大家可以看到成功收到邮件，测试成功。在模拟故障时先重启一下keepalived服务。

(9).模拟故障

```
[root@master keepalived]# ip addr show #查看一下VIP
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    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 qlen 1000
    link/ether 00:0c:29:4b:a1:85 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.208/24 brd 192.168.18.255 scope global eth0
    inet 192.168.18.200/32 scope global eth0
    inet6 fe80::20c:29ff:fe4b:a185/64 scope link
       valid_lft forever preferred_lft forever
[root@master keepalived]# touch down #进入维护模式
[root@master keepalived]# ll
总用量 8
-rw-r--r-- 1 root root    0 8月  22 14:39 down
-rw-r--r-- 1 root root 1543 8月  22 14:04 keepalived.conf
-rwxr-xr-x 1 root root 2516 8月  22 14:15 notify.sh
[root@master keepalived]# ip addr show #再次查看VIP
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    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 qlen 1000
    link/ether 00:0c:29:4b:a1:85 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.208/24 brd 192.168.18.255 scope global eth0
    inet6 fe80::20c:29ff:fe4b:a185/64 scope link
       valid_lft forever preferred_lft forever
[root@slave keepalived]# ip addr show #大家可以看到VIP成功移动到slave上
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    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 qlen 1000
    link/ether 00:0c:29:f9:e6:26 brd ff:ff:ff:ff:ff:ff
    inet 192.168.18.207/24 brd 192.168.18.255 scope global eth0
    inet 192.168.18.200/32 scope global eth0
    inet6 fe80::20c:29ff:fef9:e626/64 scope link
       valid_lft forever preferred_lft forever
```

(10).查看一下邮件



注，大家可以看到，主备切换时，会发送邮件报警，好了到这里所有演示全部完成。希望大家有所收获^_^.....

Linux HA 高可用集群 Linux集群与存储专题

上一篇：CentOS6.4+cman+r... 下一篇：MySQL 5.5 主从复制异步...

陈明乾
163篇文章，440W+人气，60粉丝

提问和评论都可以，用心的回复会被更多人看到和认可

Ctrl+Enter 发布

取消 发布

7条评论

按时间正序 | 按时间倒序

yeyeyeyuyu

1楼 2014-01-16 10:45:59

4.简单修改一下slave配置文件|@|博主，示例中的slave配置文件还少改了个router_id，那个要保持唯一。slave配置与master配置区别一共三处|@|

flymeteor

2楼 2016-12-28 13:22:56

感谢，，通过你的教程，配置成功，并解决了我之前配置不成功的问题。

flymeteor

3楼 2016-12-28 13:32:29

1

vim realserver.sh，这个脚本有什么用？？？我没有使用，，平衡也支持运行。。

> williefly:@flymeteor
> DR模式需要在realsever配置|@|nat模式不需要设置
>
> 2017-01-12 09:48:14 回复

lvnian2009

4楼 2017-02-11 11:31:43

写得灰常详细！很不错

qx517971976

5楼 2017-04-27 11:25:57

不错 相当的详细

paopaomm

6楼 2017-05-16 16:51:33

你好，我想咨询下，您写得是一个virtual_server对应了所有的real_server。有的文章写得是，每个virtual_server 下分开在master和slave上分别对应一个real_server，比如这篇文章：http://www.zhitbar.com/2017/05/15/mysql-mm-keepalived-test/|@|有什么区别吗

推荐专栏

网工2.0晋级攻略 ——零基础入门Python/An…

网络工程师2.0进阶指南

共30章 | 姜汁啤酒

订 阅

¥51.00 1162人订阅

基于Kubernetes企业级容器云平台落地与实践

容器私有云平台实践之路

共15章 | 李振良OK

订 阅

¥51.00 431人订阅

负载均衡高手炼成记

高并发架构之路

共15章 | serv

订 阅

¥51.00 396人订阅

VMware vSAN中小企业应用案例

掌握VMware超融合技术

共41章 | 干春海

订 阅

￥51.00 **252人订阅**

带你玩转高可用

前百度高级工程师的架构高可用实战

共15章 | 曹林华

订 阅

￥51.00 **388人订阅**

猜你喜欢

我的友情链接

开学季出大事：某教育局丢失3台虚拟机

服务器数据恢复通用方法+服务器分区丢失恢复案例

EMC 5400服务器raid阵列瘫痪数据恢复成功案例

性能调优概述

EVA4400存储虚拟机+数据库数据恢复成功案例

在CentOS7上部署squid缓存服务器及代理功能

服务器数据恢复案例 / raid5阵列多块硬盘离线处理方法