Keepalived—高可用

一、keepalived 简述

1、VRRP(Virtual Router Redundancy Protocol: 虚拟路由冗余协议)

(1)VRRP(虚拟路由冗余协议)定义：

Virtual Router Redundancy Protocol(虚拟路由冗余协议)，是一种容错协议，通过把几台路由设备虚拟成一台对外服务的网关路由，内部的路由器组之间就是采用 VRRP 协议进行通信，MASTER 节点固定频率地通过 BROADCAST 向 BACKUP 节点 advert(通告)自己的 HEARTBEAT(心跳信息)和 PRIORITY(优先级)等信息。一旦主节点故障，备节点就会从 BACKUP 状态转为 MASTER 状态，显然，如果主节点降低优先级，备节点通过抢占(preempt)即可转为 MASTER 状态，也就是 VIP 的转移。因此，新 MASTER 需发起 ARP 广播以告知自己和其他主机 VMAC(虚拟主机地址)地址发生了变化。 以此，两台冗余的路由器提供了高可用的服务。

(2)VRRP(虚拟路由冗余协议)工作过程

1）虚拟路由器中的路由器根据优先级选举出 Master。Master 路由器通过发送免费 ARP（Address Resolution Protocol：地址解析协议）报文，将自己的虚拟 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 路由器周期性发送 VRRP 报文，以公布其配置信息（优先级等）和工作状况；

3） 如果 Master 路由器出现故障，虚拟路由器中的 Backup 路由器将根据优先级重新选举新的 Master；

4）虚拟路由器状态切换时，Master 路由器由一台设备切换为另外一台设备，新的 Master 路由器只是简单地发送一个携带虚拟路由器的 MAC 地址和虚拟 IP 地址信息的免费 ARP 报文，这样就可以更新与它连接的主机或设备中的 ARP 相关信息。网络中的主机感知不到 Master 路由器已经切换为另外一台设备。

5）Backup 路由器的优先级高于 Master 路由器时，由 Backup 路由器的工作方式（抢占方式和非抢占方式）决定是否重新选举 Master。 由此可见，为了保证 Master 路由器和 Backup 路由器能够协调工作，VRRP 需要实现以下功能：

Master 路由器的选举；

Master 路由器健康状态报告；

为了安全性，VRRP 提供了认证机制。

在 VRRP 协议实现里，虚拟路由器使用 00-00-5E-00-01-XX 作为虚拟 MAC 地址，XX 就是唯一的 VRID （Virtual Router IDentifier），这个地址同一时间只有一个物理路由器占用。在虚拟路由器里面的物理路由器组里面通过多播 IP 地址 224.0.0.18 来定时发送通告消息。每个 Router 都有一个 1-255 之间的优先级别，级别最高的（highest priority）将成为主控（master）路由器。通过降低 master 的优先权可以让处于 backup 状态的路由器抢占（pro-empt）主路由器的状态，两个 backup 优先级相同的 IP 地址较大者为 master，接管虚拟 IP。

2、keepalived 简介

(1)keepalived 引入

keepalived 最初的设计目标是为了实现 lvs(Linux Virtual Server)设备的高可用，keepalived 能够根据配置文件中的定义生成 ipvs 规则，并能够对各个 real server 的健康状态进行检测(LVS 实际上按照高可用的角度来讲，只有两个资源，一个是 VIP，一个是内核上的 ipvs 规则)。Keepalived 是一个基于 VRRP 协议来实现的服务高可用方案，可以利用其来避免 IP 单点故障，类似的工具还有 heartbeat、corosync、pacemaker(corosync+pacemaker 是最佳组合，但与 keepalived 是不同工作机制的高可用方案)。但是 keepalived 一般不会单独出现，而是与其它负载均衡技术（如 lvs、haproxy、nginx）一起工作来达到集群的高可用。

(2)keepalived 介绍

keepalive 启动后，是由主进程(Control Plane) 读取和分析配置文件，并根据配置文件，指挥两个子进程(Checkers & VRRP Stack)完成相关工作。

VRRP Stack : 是整个 keepalive 功能实现的核心子进程 ；

Checkers : 主要用于检测 real server 的健康状态 ，可以基于 TCP Check ，http_get ，https_get , misc_get 等多种方法;

主进程还能利用内核提供的 watchdog 模块，实现对两个子进程的健康性检查，当发现某个子进程故障后，主进程会 kill 掉这个子进程，然后再从新启一个子进程，而且 watchdog 能够让两个子进程每隔一段时间，向主进程的 socket 套接字上发送信息，当某一时刻，主进程无法收到子进程的信息，就判断子进程故障，然后就杀掉，重启子进程

(3)keepalived 主备选择

两个节点之间哪个为主，哪个为备用节点，是由优先级来决定的，当初始时，优先级高的为主节点，当其中某一个节点故障，无法启动，则另一个节点会替换上来，作为主的，向外提供服务。另一种情况，当节点本身正常，但是节点上的某服务，也就是某资源不正常时，这时 keepalived 的子进程 checkers 就会将当前节点的优先级降级，从而实现了另一个备用节点的优

先级要高于该节点，当下次节点间相互通告信息时，备用节点就会发现自己的优先级比当前活动节点的优先级要高，然后就提升为主节点，启动资源，对外提供服务

优先级的数字是 0-255，值越大，优先级越高。其中 0 和 255 被系统自身占用，不能作为可用的优先级的数字进行调整。

二、keepalived 的安装配置

1、keepalived 安装前提：

(1) 各节点时间必须同步(ntpdate(chrony)、crond 服务)。

(2) 确保 iptables 及 selinux 不会成为阻碍。

(3) 各节点之间可通过主机名互相通信；建议使用/etc/hosts 文件实现；

(4) 各节点之间的 root 用户可以基于密钥认证的 ssh 服务完成互相通信。

```
[root@web ~]# ssh-keygen -f /root/.ssh/id_rsa -P ""                    ##建立 ssh 主机秘钥
[root@web ~]#ssh-copy-id -I /root/.ssh/id_rsa.pub root@node1           ##传播 ssh 主机公钥至个主机
```

(5) 网卡的 MULTICAST(多播)功能应打开：[root@ha ~]#ip link set DEVICE multicast on

2、keepalived 的安装配置

(1)安装 keepalived 软件：

CentOS 6.4 以后的版本以及收录到 base repository 中，直接 yum 安装即可。也可在其官方网站：http://www.keepalived.org/download.html，下载最新版本进行编译安装

```
[root@web ~]#wget http://www.keepalived.org/software/keepalived-1.2.15.tar.gz
[root@web ~]#tar zxvf keepalived-1.2.15.tar.gz
[root@web ~]#cd keepalived-1.2.15
[root@web ~]#./configure --prefix=/usr/local/keepalived
[root@web ~]#make && make install
[root@web ~]#echo "export PATH=$PATH:/usr/local/keepalived/bin" >/etc/profile.d/ka.sh
[root@web ~]#source /etc/profile.d/ka.sh
```

或者配置本地 yum 源，yum 安装：

```
[root@web ~]#yum install keepalived -y
```

(2)keepalived 相关文件介绍

配置文件：/etc/keepalived/keepalived.conf

Unit File：/etc/rc.d/init.d/keepalived

主程序：/usr/sbin/keepalived

配置文件示例：/usr/share/doc/keepalived-1.2.13/samples/*

配置文件说明

keepalived 的主配置(/etc/keepalived.conf)文件分为3 段，分别为全局的配置段、VRRP 实例段(可包含多个实例)、VS 实例段(可包含多个实例)：

1)全局配置段：设置象征性的通知邮箱、路由标识、vrrp 多播地址(D 类)；

```
1.global_defs {
2.    notification_email {
3.        root@localhost            ##通知邮件
4.    }
5.    notification_email_from keepaliced@localhost
6.    smtp_server 127.0.0.1         ##邮件服务器
7.    smtp_connect_timeout 30       ##设置连接邮件服务器超时时间
8.    router_id LVS_node1           ##设置连接##邮件服务器的路由
9.    vrrp_mcast_group4 224.0.22.22 # optional, default 224.0.0.18 ~ 239，选择虚拟路由
10.}
```

(2)VRRP 实例段：初始状态、物理接口、唯一编号、优先级、vrrp 认证、VIP、检测的网卡、抢占模式、通知脚本等；

```
1.vrrp_instance <STRING> {
2.    state MASTER
      #MASTER|BACKUP，当前节点在此虚拟路由器上的初始状态，只能有一个 MASTER，多个 BACKUP
3.    interface eth0                # 绑定当前虚拟路由器的物理接口
4.    virtual_router_id 12          # 当前虚拟路由器的唯一标识，0 ~ 255
5.    priority 100                  # 当前主机在此虚拟路由器的优先级，1 ~ 254
6.    advert_int 1                  # vrrp 通告的时间间隔
7.    authentication {
8.        auth_type PASS            # PASS||AH，Simple Passwd (suggested)认证方式
9.        auth_pass 1234            # Only the first eight (8) characters 认证密码，明文(8 位)
10.    }
11.    virtual_ipaddress {          ##虚拟 IP 地址的设置
12.        #<IPADDR>/<MASK>  brd  <IPADDR> dev <STRING> scope <SCOPE> label <LABEL>
13.        192.168.200.17/24 dev eth0                          # 不用别名也可有多个 ip 地址
14.        #192.168.200.18/24 dev eth2 label eth2:1            # 以网卡别名形式配置多个 ip 地址
15.    }
16.    track_interface {
17.        eth0
18.        eth1 # 监听的接口，一旦网卡故障则转为 fault 状态，通常用于监控内网接口的网卡
19.        ...
20.    }
```

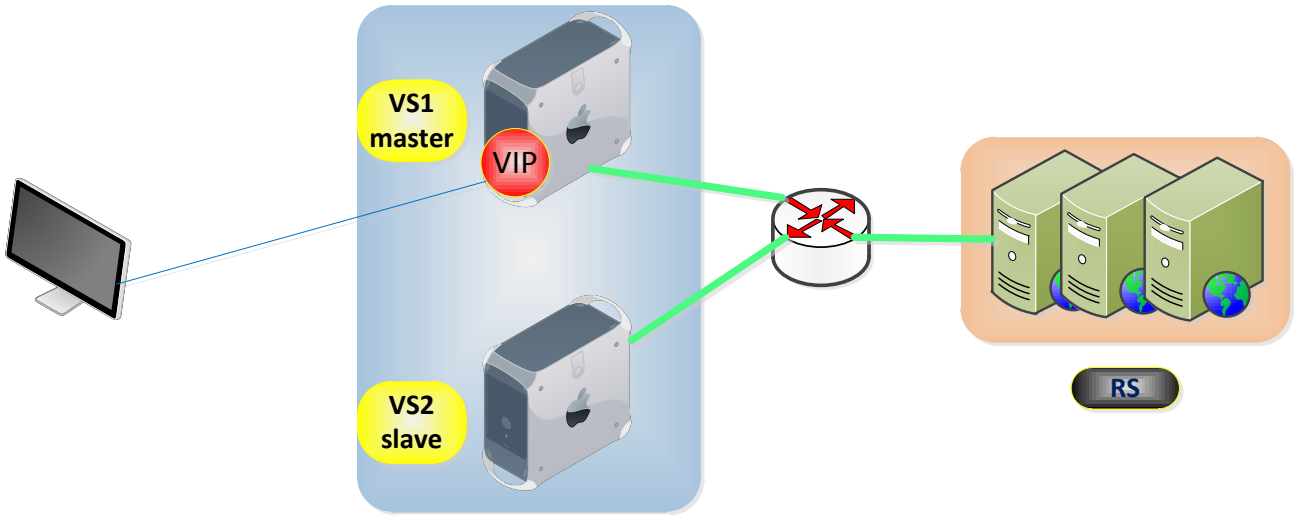
```
21.     #nopreempt # 非抢占模式，默认为 preempt
22.     preempt_delay 300                                  # 抢占模式下，节点上线后触发新选举的延迟时间
23.     notify_master <STRING>|<QUOTED-STRING>             # 节点成为主节点触发的脚本
24.     notify_backup <STRING>|<QUOTED-STRING>             # 节点转为备节点时触发的脚本
25.     notify_fault <STRING>|<QUOTED-STRING>              # 当前节点转为失败状态时触发的脚本
26.     #notify <STRING>|<QUOTED-STRING>                   #一个脚本可完成以上三种状态的转换时的通知；
27.}
```

(3)VS 示例段：VIP 及端口、检测间隔、调度算法、集群类型、持久连接时长、SorryServer、RealServer{权重、通知脚本、健康检测{HTTP/SSL/TCP/SMTP}、超时、重试、延时}等。以下以 lvs 的高可用为例：

```
1.virtual_server IP port {
3.     delay_loop  <INT>                          # 对 RS 健康检测的间隔
4.     lb_algo rr|wrr|lc|wlc|lblc|sh|dh           # LB 调度算法
5.     lb_kind NAT|DR|TUN                         # LVS 类型
6.     persistence_timeout <INT>                  # 持久时间
7.     protocol TCP                               #协议
8.     sorry_server <IPADDR> <PORT>               #本机作为 RS 不可用的 sorry 主机及端口，备用页面
9.     real_server <IPADDR> <PORT>{               #后端实际服务器设置
10.         weight <INT>                           #权重
11.         notify_up <STRING>|<QUOTED-STRING>
12.         notify_down <STRING>|<QUOTED-STRING>
13.         HTTP_GET|SSL_GET|TCP_CHECK|SMTP_CHECK|MISC_CHECK {     #可使用不同的检测方法定义主机的
健康状态，较常用为 HTTP_GET 和 TCP_CHECK
14.             url {
15.                 path <URL_PATH>                # 要监控的 URL
16.                 status_code <INT>              # 健康状态的响应码（200 正常）
17.                 digest <STRING>                # 健康状态响应内容的摘要
18.             }
19.             nb_get_retry <INT>                 # 重试次数
20.             delay_before_retry <INT>           # 向当前 RS 的哪个 IP 地址发起监控状态请求
21.             connect_port <PORT>                # 向 RS 的哪个 port 发起健康监测
22.             bindto <IP ADDRESS>                # 发出健康状态检测请求的原 IP
23.             connect_timeout <INTEGER>          # 连接请求的超时时长
24.         }
25.     }
26.}
注：使用 man keepalived.conf 查看更多的配置文件的 设置
```

三、keepalived 的模型和配置

1、单主(主从)模型虚拟路由的配置文件：



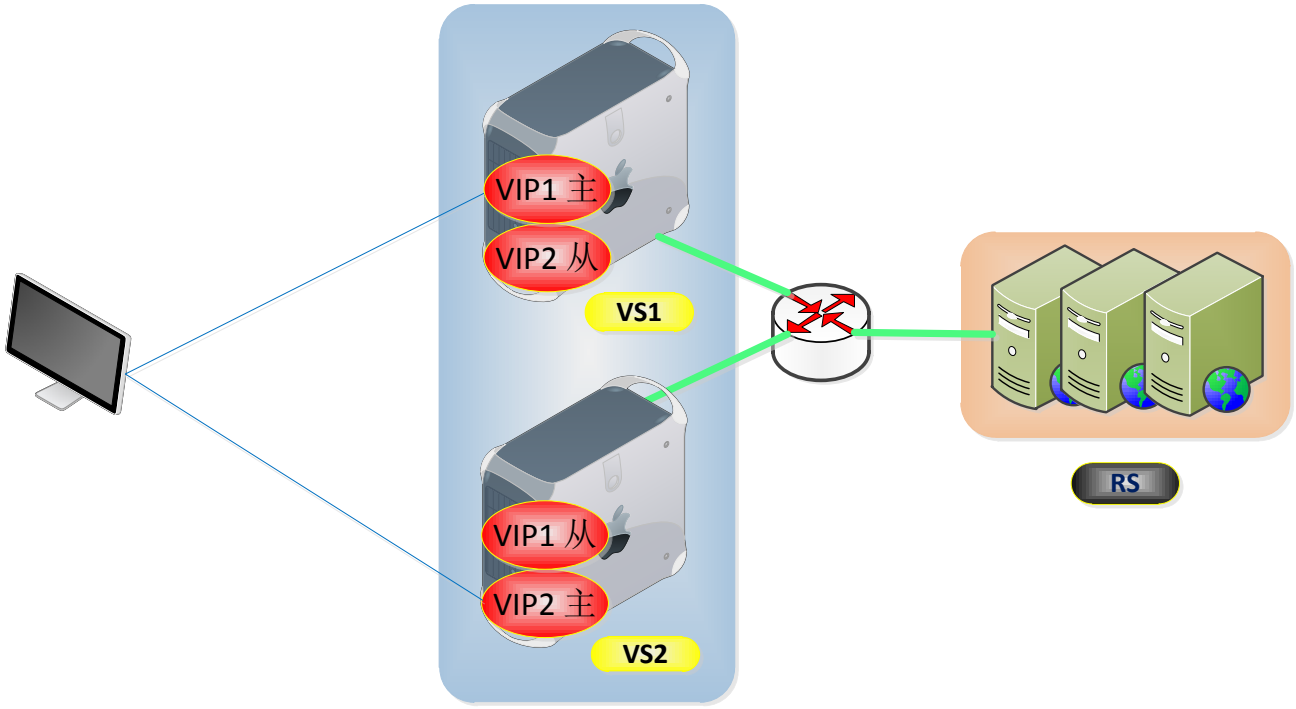
前提：VS1 的初始状态为 MASTER，VS2 的初始状态为 BACKUP；且 BACKUP 节点的优先级应小于 MASTER 节点。*红色标注的为从机上必须修改的地方*。(经验：对于 keepalived 的运用，主要在于脚本的编写和配置文件的设计)

```
[root@web ~]# vim /etc/keepalived.conf
2.    global_defs {
3.       notification_email {
4.         root@localhost
5.       }
6.       notification_email_from keepalived@
jasonmc.com
7.       smtp_server localhost
8.       smtp_connect_timeout 30
9.       router_id node1
10.       vrrp_mcast_group4 224.22.29.1
11.    }
12
13.    vrrp_instance VI_1 {
14.        state MASTER      #在从机上应为 BACKUP
15.        interface eth0
16.        virtual_router_id 10
17.        priority 100     #从机上应低于此值
18.        advert_int 10
19.        authentication {
20.            auth_type PASS
21.            auth_pass d351ac09
22.        }
23.        virtual_ipaddress {
24.            10.1.253.11 dev eth0
25.        }
26     }
```

2、双主模型虚拟路由(两个 VIP)的配置文件：

前提：两台主机都需要安装 keepalived，在 VS1 中 VIP1 为 MASTER，VIP2 为 BACKUP；在 VS2 中 VIP1 为 BACKUP，VIP2 为 MASTER。且要保证对应 VIP 的优先级(priority)不同，即从 VIP 的优先级值小于主 VIP 的优先级，两台主机应保持使用相同名称的网卡，相同的 VIP

使用相同的虚拟路由 ID(virtual_router_id)，配置文件中红线部分为 VS2 上必须修改项。




```
[root@node1 ~]# vim /etc/keepalived.conf
2.global_defs {
3.   notification_email {
4.     root@localhost
5.   }
6.   notification_email_from keepalived@jasonmc.com
7.   smtp_server localhost
8.   smtp_connect_timeout 30
9.   router_id node1
10.  vrrp_mcast_group4 224.22.29.1 #224~239
11.}
13.vrrp_instance VI_1 {
14.    state MASTER          #VS2 应修改为 BACKUO
15.    interface eth0
16.    virtual_router_id 1
17.    priority 100          #VS2 应小于该值
18.    advert_int 1          #vrrp 通告时长
19.    authentication {
20.        auth_type PASS
21.        auth_pass cc5042a6
                   #openssl rand -hex 4
22.    }
```

```
23.    virtual_ipaddress {
24.        10.1.253.11 dev eth0
25.    }
29.}
31.vrrp_instance VI_2 {              ##备用 VIP
32.    state BACKUP                  #VS2 应为 MASTER
33.    interface eth0
34.    virtual_router_id 2           #虚拟路由器的唯一标识，0 ~ 255
35.    priority 98                   #VS2 应大于该值
36.    advert_int 1
37.    authentication {
38.        auth_type PASS
39.        auth_pass ac5342a5
40.    }
41.    virtual_ipaddress {
42.        10.1.253.12 dev eth0
43.    }
44.      notify_master "/etc/keepalived/script/notify.sh master"      #状态转化 master 触发脚本
45.      notify_backup "/etc/keepalived/script/notify.sh backup"      #状态转化 slave 触发脚本
46.      notify_fault "/etc/keepalived/script/notify.sh fault"        #状态转化 fault 触发脚本
47.}
```

3、通知脚本示例：

当以上各节点状态发生转化时会自动触发以下脚本：

```
[root@web ~]#vim /etc/keepalived/notify.sh
2.#!/bin/bash
4.receiver='root@localhost'
5.notify() {
6.    mailsubject="$(hostname) to $1,vip floating."
7.    content="$(date + '%F %T') vrrp state transion, $(hostname) changed to be $1"
8.    echo "$content" | mail -s "$mailsubject" $receiver
9.}
10.case $1 in
11.master)
12.    notify master
13.    ;;
14.backup)
15.    notify backup
16.    ;;
```

```
17.fault)
18.    notify fault
19.    ;;
20.*)
21.    echo "Usage $(basename $0) {master|backup|fault}"
22.    exit 1
23.    ;;
24.esac
```

4、使用检测脚本的过程：

```
定义一个脚本
vrrp_script  <NAME> {
         script “string or /path”
         interval INT      ##间隔时间
         weight -INT       ##权重分配
}
##可添加多了检测脚本
##定义不同的脚本名称
```

```
在 vrrp 实例中调用 vrrp_script
vrrp_instance VI_1 {
......
track_script {
         NAME
         }
......
}
```

注：keepalived 使服务拥有高可用性的特点，可与 redis、memcache、mysql、nginx、httpd、lvs 等各种以 tcp 等协议连接为基础的服务相结合，实现各个对应服务的高可用性，在实际业务中有广泛的运用