私塾在线《高级软件架构师实战培训阶段二》

——跟着cc学架构系列精品教程

本部分课程概览

- 根据实际的应用需要，学习Keepalived的相关知识，以快速上手、理解并掌握Keepalived，大致包括：

1：Keepalived介绍和基本实现思路
2：VRRP协议：概念、理解、工作机制、负载分担等
3：Keepalived安装
4：Keepalived体系结构
5：Keepalived配置
6：Keepalived+Nginx的HA
7：Keepalived的HA，比如和：Varnish、Tomcat、Redis、MySql等
8：LVS基础知识
9：Keepalived+LVS实现高可用的负载均衡

简介

- Keepalived介绍

Keepalived是一款用于保障服务高可用性的软件，它能自动侦测服务器状态、移出故障服务器、切换到正常运行的服务器、添加恢复后的服务器到集群中。

- 实现的基本思路

Keepalived是基于VRRP协议的实现，主要用在IP层、TCP层和应用层。

1：IP层：Keepalived会定期向服务器群中的服务器发送一个数据包（既Ping），如果发现IP地址没有激活，Keepalived便报告这台服务器失效，并将它从服务器群中剔除。

2：TCP层：类似IP层，只不过这里是检测服务的端口

3：应用层：Keepalived将根据用户的设定来检查服务程序的运行是否正常

- VRRP协议解决的问题

在现实网络中，两台服务器之间通常是没有直接连接的，那么A机如何选择到达B机的路由呢？通常的解决方案有两种：

1：在A机上使用动态路由协议　（问题：管理维护成本、是否支持等）

2：在A机上配置静态路由（问题：路由器或默认的网关成单点、需重启网络等）

VRRP协议的目的就是解决路由单点故障问题。

VRRP协议-1

- VRRP协议

VRRP (Virtual Router Redundancy Protocol)，虚拟路由冗余协议，是解决局域网中配置静态网关出现单点故障的路由协议。

在具有多播或广播能力的局域网中，借助VRRP能在某台路由器出现故障时仍然提供高可靠的缺省链路，有效避免单一链路发生故障后网络中断的问题，而无需修改动态路由协议、路由发现协议等配置信息。

- 名词解释

1：VRRP路由器：物理的路由器，上面运行了实现VRRP协议的VRRPD程序

2：VRRP虚拟路由器：逻辑上的路由器，通常由多台路由器组成，可以看成是一个VRRP路由器池，对外看起来是一个路由器，就是那个虚拟的路由器，其标识称为VRID(范围是0-255)

3：Master和Backup：一个虚拟路由里面的多个路由器，并不是同时工作的，工作的那台称为Master，其他的就是Backup。

- 对VRRP协议的理解

1：VRRP是一种选择协议，它可以把一个虚拟路由器的职责，动态转交给Master进行处理

2：VRRP是一种路由容错协议，也可以叫做备份路由协议。当Master宕掉后，虚拟路由将启用备份路由器，从而实现网络通信可用

VRRP协议-2

- VRRP协议工作机制

1：路由器开启VRRP功能后，会根据优先级确定出Master

2：Master会通过IP多播包的形式来发送公告报文，Backup会接收到这些报文

3：如果是抢占式：Backup会跟发送报文的Master比较优先级，如果Backup优先级更高，那么Backup会抢占成为Master，而Master会让位成为Backup

4：如果是非抢占式：只要Master没有故障，不会出现新的Master。

5：如果备份路由器在连续三个公告间隔内收不到VRRP公告，或收到优先级为0的公告的话，就会按照竞选协议来选出新的Master，以保证服务的可用

- VRRP负载分担

在实际组网中一般会进行VRRP负载分担方式的设置。负载分担方式是指多台路由器同时承担业务，避免设备闲置。

同一台路由器可以加入多个备份组，在不同组中有不同的优先级，使得该路由器可以在一个组中作为Master，在其他的备份组中作为Backup

Keepalived安装-1

- 下载并安装IPVS

Keepalived需要IPVS，首先要确保安装了IPVS（IP虚拟服务器，IP Virtual Server，是一种提供负载平衡功能的技术）。

1：检查是否安装了IPVS：在任意路经下执行ipvsadm命令

2：去http://www.linuxvirtualserver.org/software/ipvs.html下载相应的包，注意要跟你的linux内核版本匹配。察看linux版本的命令：cat /proc/version

3：创建一个连接文件，其命令为：ln -sv /usr/src/kernels/2.6.18-194.el5-i686/ /usr/src/linux，注意一定要与当前的运行的内核相一致，因为usr/src/kernels目录下可多个目录。如果不创建这个连接文件，在编译时会出错，从而不能继续进行安装。

4：然后 make ， make install ，ipvsadm命令会被安转到 /sbin下面

5：在任意路经下执行ipvsadm命令，检查是否正确安装

Keepalived安装-2

- 下载并安装Keepalived，去http://www.keepalived.org/下载最新的源码包

1：解压包，tar zvxf，然后进入到解压的文件夹里面

2：第一步：./configure --prefix=/usr/common/keepalived

（1）如果出现No SO_MARK declaration in headers 这样的错误提示，可以在命令上添加--disable-fwmark

（2）如果要使用lvs，还需要指定内核的目录，也就是添加：--with-kernel-dir=具体的路经，以指定使用内核源码里面的文件

第二步：make　第三步：make install

3：验证安装

（1）到sbin下，执行 keepalived 命令

（2）察看进程，ps -ef | grep keepalived ，应该有三种进程，父进程（内存管理，监控子进程），健康检查子进程，VRRP子进程

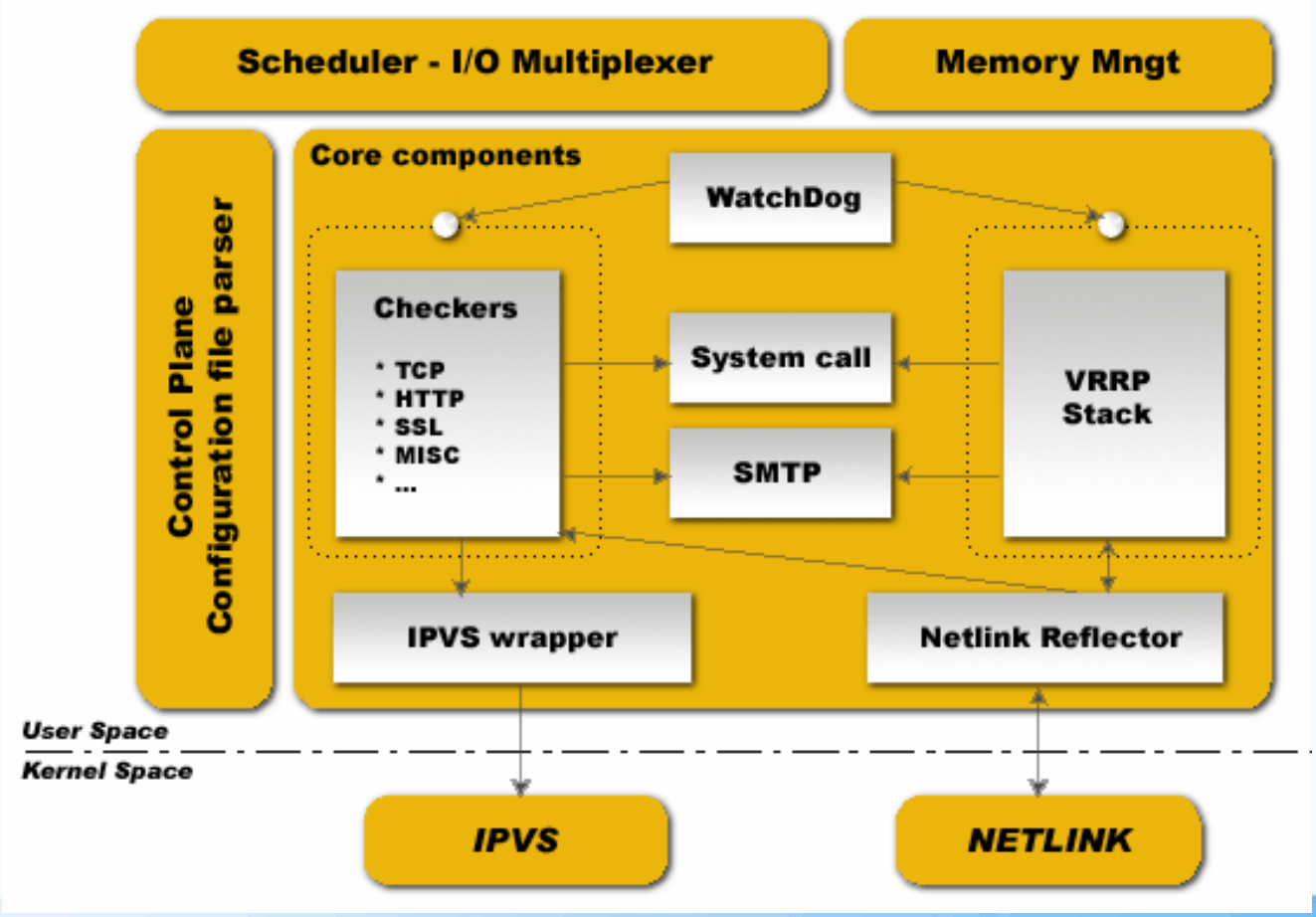
（3）察看内核模块ip_vs是否装载到内核空间，lsmod | grep ip_vs

（4）执行tail －f /var/log/messages来查看日志

（5）执行pkill keepalived来关闭keepalived

Keepalived的体系结构-1

- Keepalived的体系结构图



Keepalived的体系结构-2

- 大致分两层结构：用户空间 user space和内核空间 kernel space

1：IPVS：IP虚拟服务器（IP Virtual Server），是一种提供负载平衡功能的技术
2：NetLink：提供高级路由及其他相关的网络功能
3：WatchDog：负责监控checkers和VRRP进程的状况
4：Checkers：负责真实服务器的健康检查，是keepalived最主要的功能。可以没有VRRP Stack,但健康检查healthchecking是一定要有的。
5：VRRP Stack：负责负载均衡器之间的失败切换FailOver。如果只用一个负载均衡器，则VRRP不是必须的。
6：IPVS wrapper：用来发送设定的规则到内核的IPVS部分
7：Netlink Reflector：用来设定VRRP的vip地址等
8：控制面板：对配置文件的编译和解析。Keepalived不是一次性解析所有的配置文件，而是用到一个模块解析一个，因此可以在每个模块看到XXX_parser.c这样的文件

Keepalived的配置-1

- 配置概述

Keepalived各种功能的实现是通过设置其配置文件keepalived.conf来完成的。配置大致分成如下几类：全局配置、VRRPD配置、LVS配置

- 全局配置

全局配置包含全局定义和静态地址路由。

1：全局定义主要设置Keepalived的通知机制和标识

```
global_defs{
 notification_email{
        admin@example1.com
 }
 notification_email_from admin@example.com
 smtp_server 127.0.0.1
 smtp_connect_timeout 30
 router_id my_hostname
 vrrp_mcast_group4 224.0.0.18
 vrrp_mcast_group6 ff02::12
 enable_traps
}
```

Keepalived的配置-2

（1）notification_email：在有事件时，比如切换的时候，发消息到哪些email，可以多个，一行一个

（2）notification_email_from：通知邮件从哪个地址发出

（3）smpt_server：通知邮件的smtp地址

（4）smtp_connect_timeout：连接smtp服务器的超时时间，单位秒

（5）enable_traps：开启SNMP陷阱

（6）router_id：运行Keepalived的机器的标识，一个网络内应该是唯一的，通常为hostname，但不一定非得是hostname。故障发生时，邮件通知会用到。VRRP协议中不传输这个字段，传输的是Vritual route id，在vrrp instance中配置

（7）vrrp_mcast_group4：vrrp心跳的多播组，如果广播域中有多个Keepalived HA组的话，不同的组建议使用不同的多播地址，同时使用不同的虚拟路由器ID，可选，缺省是224.0.0.18

（8）vrrp_mcast_group6：可选，default ff02::12

Keepalived的配置-3

2：静态地址和路由：配置的是是本节点的IP和路由信息，也就是不随VRRP实例变化而变化的地址和路由。如果你的机器上已经配置了IP和路由，那么这两个区域可以不用配置。其实，一般情况下你的机器都会有IP地址和路由信息的，因此没必要再在这两个区域配置可以不配置，配置如下：

```
static_ipaddress{
 192.168.1.1/24 brd + dev eth0 scope global
 ...
}
static_routes{
 192.168.2.0/24 via 192.168.1.100 dev eth0
 ...
}
```

每行设置一个ip，格式符合linux下ip命令参数的格式，比如上面的：

brd：目的是广播地址(broadcast)

dev：指定设备名称

scope：设置地址的有效范围

via：指定下一跳路由器的地址

Keepalived的配置-4

- VRRPD配置

包含VRRP同步组和VRRP实例两个部分。

1：VRRP同步组

用来保证组里面任何一个实例出问题，都会导致切换。比如某个机器在两个网段，当一个网段出问题的时候，就应该切换。例如：

```
vrrp_sync_group VG_1 {
    group {
        inside_network
        outside_network
        ......
    }
    notify_master /path/to/to_master.sh
    notify_backup /path_to/to_backup.sh
    notify_fault "/path/fault.sh VG_1 “
    notify /path/notify.sh
    smtp_alert
}
```

Keepalived的配置-5

（1）inside_network：具体的VRRP实例名

（2）outside_network：对外的ip

（3）notify_master：指定当切换到Master时执行的脚本，可以传入参数，用“”括起来，notify_backup和notify_fault类似

（4）notify：任何状态的切换变化都会触发并运行的脚本，参数自动添加：$1 = “GROUP”|“INSTANCE”；$2 = 组或实例的名字； $3 = 切换成的目标状态("MASTER"|"BACKUP"|"FAULT")

（5）smtp_alert：使用全局定义的设置，在切换后发送邮件通知

2： VRRP实例

用来定义对外提供服务的VIP区域及其相关属性

```
vrrp_instance inside_network {
  state MASTER
  interface eth0
  use_vmac <VMAC_INTERFACE>
  vmac_xmit_base
  dont_track_primary
```

Keepalived的配置-6

```
track_interface {
  eth0
  eth1      ...
}
mcast_src_ip <IPADDR>
unicast_src_ip <IPADDR>
unicast_peer {
  <IPADDR>     ...
}
lvs_sync_daemon_interface eth1
garp_master_delay 10
virtual_router_id 51
priority 100
advert_int 1
authentication {
    auth_type PASS
    auth_pass 1234
}
virtual_ipaddress {
    <IPADDR>/<MASK> brd <IPADDR> dev <STRING> scope <SCOPE> label <LABEL>
    192.168.200.17/24 dev eth1
    192.168.200.18/24 dev eth2 label eth2:1
}
```

Keepalived的配置-7

```
virtual_ipaddress_excluded {
   <IPADDR>/<MASK> brd <IPADDR> dev <STRING> scope <SCOPE>
   <IPADDR>/<MASK> brd <IPADDR> dev <STRING> scope <SCOPE>
      ...
  }
  virtual_routes {
    src 192.168.100.1 to 192.168.109.0/24 via 192.168.200.254 dev eth1
    192.168.110.0/24 via 192.168.200.254 dev eth1
    192.168.111.0/24 dev eth2
    192.168.112.0/24 via 192.168.100.254
    192.168.113.0/24 via 192.168.200.254 or 192.168.100.254 dev eth1
    blackhole 192.168.114.0/24
  }
  nopreempt
  preempt_delay 300
  debug
  notify_master <STRING>|<QUOTED-STRING>
  notify_backup <STRING>|<QUOTED-STRING>
  notify_fault <STRING>|<QUOTED-STRING>
  notify <STRING>|<QUOTED-STRING>
  smtp_alert
}
```

Keepalived的配置-8

（1）state：指定实例的初始状态，运行起来后，会动态的改变。只有MASTER和BACKUP两种状态，并且需要大写这些单词。

（2）interface：实例绑定的网卡，用来发VRRP包

（3）use_vmac：是否使用VRRP的虚拟MAC地址

（4）vmac_xmit_base：发送和接收VRRP包的虚拟MAC地址

（5）dont_track_primary：忽略VRRP网卡错误。（默认未设置）

（6）track_interface：监控以下网卡，任何一个不通就会切换到FALT（可选）

（7）mcast_src_ip：修改vrrp组播包的源地址，默认源地址为master的IP。（由于是组播，因此即使修改了源地址，该master还是能收到回应的）

（8）unicast_src_ip：不使用组播的源地址

（9）unicast_peer：不使用多播发送vrrp心跳包，而改为使用单播发送。这里要配置的是单播组的IP，即在这个单播组的机器会收到vrrp心跳包，所以主备的Keepalived都需要加进来

（10）lvs_sync_daemon_interface：lvs同步服务绑定的网卡

Keepalived的配置-9

（11）garp_master_delay：角色转换成master后，延迟多长时间发送免费ARP包．宣告IP和MAC对应关系，主要用于告诉广播域的其他主机或路由器当前虚拟IP对应的MAC地址是什么

（12）virtual_router_id：虚拟路由标识，是一个数字，整个VRRP内唯一

（13）priority：优先级，是一个数字，数值愈大，优先级越高

（14）advert_int：MASTER与BACKUP之间同步检查的时间间隔，单位为秒

（15）authentication：验证，包含验证类型和验证密码。类型主要有PASS、AH两种，常用PASS，密码是明文，同一VRRP实例MASTER与BACKUP使用相同的密码才能正常通信。AH为ipsec 认证头认证，不需要配置密码 。

（16）virtual_ipaddress：虚拟ip地址，可多个，每个地址占一行，不需要指定子网掩码。注意：如果用LVS的话，这个ip必须与LVS客户端设定的vip一致

（17）virtual_ipaddress_excluded：排除心跳包发送的IP地址

（18）virtual_routes：路由配置，当角色为主时，自动添加这些路由，当角色为备时，自动删除这些路由

（19）nopreempt：非抢占式

（20）preempt_delay：抢占延迟，默认300秒

（21）debug：debug级别

Keepalived的配置-10

- LVS配置

跟LVS相关的配置，如果不使用LVS的话，不需要配置。包含虚拟服务器组和虚拟服务器两个部分。

1：虚拟服务器组

用来实现一台真实服务器上的某个服务，可以属于多个虚拟服务器，并且只做一次健康检查，是可选的。形如：

```
virtual_server_group <STRING> {
  <IPADDR> <PORT>    #VIP和端口
  ...
  <IPADDR RANGE> <PORT> #例如：192.168.200.1-10
  ...
  fwmark <INT> #经过iptables 标记过的服务类型，这样利于按标记进行处理，
               #比如：ip rule add fwmark 3  table 3 （fwmark 3是标记，table 3 是
               #路由表3。意思就是凡是标记了 3 的数据使用table3 路由表）
  ...
}
```

Keepalived的配置-11

2：虚拟服务器

可以有下面三种定义方式：

（1）virtual_server IP port

（2）virtual_server fwmark int

（3）virtual_server group string

形如：

```
virtual_server IP port |
virtual_server fwmark int |
virtual_server group string{
    delay_loop <INT>
    lb_algo rr|wrr|lc|wlc|lblc|sh|dh
    ops
    lb_kind NAT|DR|TUN
    persistence_timeout <INT>
    persistence_granularity <NETMASK>
    protocol TCP
    ha_suspend
    virtualhost <STRING>
    alpha
```

Keepalived的配置-12

```
omega
quorum <INT>
hysteresis <INT>
quorum_up <STRING>|<QUOTED-STRING>
quorum_down <STRING>|<QUOTED-STRING>
sorry_server <IPADDR> <PORT>
sorry_server_inhibit
real_server <IPADDR> <PORT>
   {
       weight <INT>
       inhibit_on_failure
       notify_up <STRING>|<QUOTED-STRING>
       notify_down <STRING>|<QUOTED-STRING>
       HTTP_GET|SSL_GET
       {
           url {
             path <STRING>
             digest <STRING>
             status_code <INT>
           }
```

Keepalived的配置-13

```
    nb_get_retry <INT>
    delay_before_retry <INT>
    connect_ip <IP ADDRESS>
    connect_port <PORT>
    bindto <IP ADDRESS>
    bind_port <PORT>
    connect_timeout <INTEGER>
    fwmark <INTEGER>
    warmup <INT>
} #HTTP_GET|SSL_GET
TCP_CHECK
{
    connect_ip <IP ADDRESS>
    connect_port <PORT>
    bindto <IP ADDRESS>
    bind_port <PORT>
    connect_timeout <INTEGER>
    fwmark <INTEGER>
    warmup <INT>
}
```

Keepalived的配置-14

```
    SMTP_CHECK {
        host {
          connect_ip <IP ADDRESS>
          connect_port <PORT>
          bindto <IP ADDRESS>
          bind_port <PORT>
          connect_timeout <INTEGER>
          fwmark <INTEGER>
        }
        connect_timeout <INTEGER>
        retry <INTEGER>
        delay_before_retry <INTEGER>
        helo_name <STRING>|<QUOTED-STRING>
        warmup <INT>
    }
    MISC_CHECK  {
        misc_path <STRING>|<QUOTED-STRING>
        misc_timeout <INT>
        warmup <INT>
        misc_dynamic
    }
} }
```

Keepalived的配置-15

（1）delay_loop：延迟轮询时间（单位秒）
（2）lb_algo：负载均衡调度算法，互联网应用常使用wlc或rr
（3）ops：为UDP开启One-Packet-Scheduling
（4）lb_kind：负载均衡转发规则。一般包括DR,NAT,TUN3种
（5）persistence_timeout：LVS会话保持的超时时间
（6）persistence_granularity：LVS会话保持粒度，也就是ipvsadm中的-M参数，默认是0xffffffff，即为每个客户端保持会话
（7）protocol：转发使用的协议，TCP或UDP
（8）ha_suspend：暂停健康检查活动
（9）virtualhost：健康检查时，检查的web服务器的头信息
（10）alpha：开启后，当健康检查程序启动失败时，可以预防误报，缺省关闭
（11）omega：当守护进程关闭时，做出合适的处理，缺省关闭
（12）quorum：所有运行的服务器的总权重数的最小值，缺省是1
（13）hysteresis：缺省是0
（14）quorum_up：quorum达到多少就启动脚本
（15）quorum_down： quorum丢失多少就启动脚本

Keepalived的配置-16

（16）sorry_server：当所有real server宕掉时，sorry server顶替
（17）sorry_server_inhibit：在sorry_server直接应用inhibit_on_failure的行为
（18）real_server：真正提供服务的服务器
（19）weight：权重，默认为1，0表示失效
（20）inhibit_on_failure：健康检查失败时，将weight设置为0，而不是从ipvs里面删除
（21）notify_up/down：当real server宕掉或启动时执行的脚本
（22）HTTP_GET：健康的检查方式，HTTP_GET|SSL_GET|TCP_CHECK|SMTP_CHECK|MISC_CHECK
（23）url：HTTP/SSL检查的URL，可以指定多个
（24）path：请求real serserver上的路径
（25）digest/status_code：检查后的摘要信息，和检查后返回的http状态码
（26）nb_get_retry：重试次数
（27）delay_before_retry：下次重试的时间延迟
（28）connect_ip：健康检查的ip
（29）connect_port：健康检查，如果端口通则认为服务器正常
（30）bindto：以此地址发送请求，来对服务器进行健康检查
（31）bind_port：绑定的端口

Keepalived的配置-17

（32）connect_timeout：表示超时时长
（33）fwmark：经过iptables 标记过的服务类型
（34）warmup：开始健康检查前，随机延迟的最大时间数，单位秒
（35）TCP_CHECK：TCP健康检查
（36）SMTP_CHECK：SMTP健康检查
（37）retry：重试次数
（38）delay_before_retry：重连接的间隔时间，单位秒
（39）helo_name：”smtp helo “请求命令的参数
（40）MISC_CHECK：MISC健康检查
（41）misc_path：外部程序或脚本路径
（42）misc_timeout：脚本执行的超时时间
（43）misc_dynamic：如果设置了这个参数，健康检查程序的退出状态码会用来动态调整服务器的权重，如下：

a：返回0：健康检查通过，不修改权重
b：返回1：健康检查失败，权重设为0
c：返回2-255：健康检查通过，权重设置为 返回的状态码 减去 2

Keepalived+Nginx的HA-1

- 准备工作

1：准备2台机器，分别安装上Java、Tomcat、Nginx、Ipvs和Keepalived

2：规划：

(1)虚拟ip用 192.168.1.77

(2)Master机器的ip：192.168.1.106 ，Nginx端口80，Tomcat端口8080

(3)Backup机器的ip：192.168.1.201 ，Nginx端口80，Tomcat端口8080

- 配置Nginx，Nginx对两台机器的Tomcat做负载均衡
- 配置Keepalived

1：master配置，示例如下：

```
global_defs {
   notification_email {       cc@cc.com   }
   notification_email_from cc@cc.com
   smtp_server smtp.cc.com
   smtp_connect_timeout 30
   router_id nginx_master
}
```

Keepalived+Nginx的HA-2

```
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 {
      192.168.1.77
  }
}
```

2：Backup的配置，示例如下：

```
global_defs {
   notification_email {    cc@cc.com    }
   notification_email_from cc@cc.com
   smtp_server 192.168.200.1
   smtp_connect_timeout 30
   router_id cc_backeup
}
```

Keepalived+Nginx的HA-3

```
vrrp_instance VI_1 {
    state BACKUP
    interface eth0
    virtual_router_id 51
    priority 99
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.1.77
    }
}
```

3：运行后可以通过如下命令查看

watch ipvsadm -lc：查看IPVS的连接情况

也可用如下命令查看：ipvsadm -Lc

Keepalived+Nginx的HA-4

- 让Keepalived监控Nginx的状态

  当NginX停止服务的时候能够自动切换，从而实现服务的高可用性。

1：这个就需要shell脚本来处理，示例如下：

```
NGINX=/usr/common/nginx/sbin/nginx
PORT=80
nmap localhost -p $PORT | grep "$PORT/tcp open"

#echo $?
if [ $? -ne 0 ];then
    $NGINX -s stop
    $NGINX
    sleep 3
    nmap localhost -p $PORT | grep "$PORT/tcp open"
    [ $? -ne 0 ] && cd /usr/common/keepalived/sbin && pkill keepalived

    echo "over"
fi
```

注意：要对这个脚本文件设置权限，如：chmod 777 checkNginx.sh

Keepalived+Nginx的HA-5

2：然后在keepalived.conf中添加对对脚本的调用

（1）在VRRP服务器配置前，增加：

```
vrrp_script chk_http_port {
    script "/usr/common/keepalived/etc/keepalived/checkNginx.sh"
    interval 2
    weight 2
}
```

（2）在vrrp_instance配置里面的最后部分，添加：

```
    track_script {
      chk_http_port
    }
```

Keepalived用做HA

- Keepalived用做HA

  Keepalived用做HA的时候，和具体的应用或服务是没有关系的，类似于前面学习的Keepalived+Nginx，同样可以实现：

1：Keepalived + Varnish
2：Keepalived + Tomcat
3：Keepalived + Redis
4：Keepalived + MySql
5：Keepalived + ……任意的其他服务或应用

LVS-1

- 简介

LVS（Linux Virtual Server），可用来实现LINUX下的简单负载均衡。

LVS工作在TCP/IP协议的四层，其转发是依赖于四层协议的特征进行转发的，由于其转发要依赖于协议的特征进行转发，因此需要在内核的TCP/IP协议栈进行过滤筛选，而这样的过滤转发规则可由管理员对内核进行定义。

LVS在内核空间中工作的是“ipvs”，而在用户空间中工作的，用来定义集群服务规则的是“ipvsadm”。

- 三种负载均衡转发的机制

1：NAT（Network Address Translation）网络地址翻译技术。

当用户请求到达调度器时，调度器将改写请求的地址为真实Server地址。在服务器端处理后，需要再次经过负载调度器将报文的源地址和源端口改成虚拟IP地址和相应端口，然后把数据发送给用户，完成整个负载调度过程。

2：TUN（IP Tunneling）IP隧道技术

调度器采用IP隧道技术将用户请求转发到某个Real Server，而这个Real Server将直接响应用户的请求，不再经过调度器。

LVS-2

3：DR（Direct Routing）直接路由技术

DR通过改写请求报文的MAC地址，将请求发送到真实Server，而真实Server将响应直接返回给客户，比TUN少了IP隧道开销。这种方式是三种负载调度机制中性能最高最好的，但是要求调度器与真实服务器在同一物理网段上

- 负载调度的算法

1：rr（Round Robin ）轮循，这种算法平等地对待每一台真实服务器，而不管服务器上实际的负载状况和连接状态

2：wrr（Weighted Round Robin ）加权轮循，根据真实服务器的不同处理能力来调度请求

3：lc（Least Connections ）最少连接，动态地将请求调度到已建立的链接数最少的服务器

4：wlc（Weighted Least Connections ）加权最少连接，每个服务节点可以用相应的权值表示其处理能力，较高权值的服务器将承受较大比例的活动连接负载

5：dh（Destination hashing ）目标地址Hash，根据请求的目标IP，作为散列键从静态分配的散列表找出对应的服务器，若该服务器是可用的且未超载，将请求发送到该服务器，否则返回空

6：SH（Source hashing）源地址hash，根据请求的源IP，作为散列键从静态分配的散列表找出对应的服务器，若该服务器是可用的且未超载，将请求发送到该服务器，否则返回空

LVS-3

7：SED（Shortest Expected Delay）最短期望的延迟，基于wlc算法，计算每个真实服务器的请求延迟，把请求转发给最短延迟的服务器

8：NQ（Never Queue）最少排队，某台真实服务器没有连接时，就直接转发过去

9：LBLC（Locality-Based Least-Connection）基于局部的最少连接，根据请求的目标IP，找出该地址最近使用的服务器，若该服务器是可用的且没有超载，将请求发送到该服务器；若服务器不存在，或者该服务器超载且有服务器处于一半的工作负载，则用“最少链接”的原则选出一个可用的服务器，将请求发送到该服务器

10：LBLCR（Locality-Based Least-Connection with replication Scheduling）带复制的基于局部最少连接，它与LBLC算法的不同之处是它要维护从一个目标 IP到一组服务器的映射，而LBLC是维护从一个目标IP到一台服务器的映射。根据请求的目标IP，找出该目标IP对应的服务器组，按“最小连接”原则从服务器组中选出一台服务器，若服务器没有超载，将请求发送到该服务器；若服务器超载，则按“最小连接”原则从这个集群中另外选出一台服务器，将该服务器加入到服务器组中，将请求发送到该服务器

LVS+Keepalived-1

- 简介

LVS+Keepalived是一个常见的组合，使用LVS来实现负载均衡，使用Keepalived来实现HA。

由于Keepalived是基于LVS的，因此这两部分功能，都可以通过Keepalived来实现，配置也基本上在Keepalived上配置。

- 配置Keepalived.conf

（1）全局配置和VRRPD的配置跟以前是一样的，只需要添加LVS的配置就可以了。

（2）主服务器和备用服务器的配置是一样的。

（3）LVS+DR模式中，只支持IP的转发，不支持端口转发，也就是说virtual_server和real_server的配置节点中端口必须一样

```
virtual_server  192.168.1.77  8080   {
    delay_loop 6
    lb_algo wrr
    lb_kind DR
    persistence_timeout 60
    protocol TCP
```

LVS+Keepalived-2

```
    real_server  192.168.1.201  8080   {
        weight 3
        #TCP_CHECK {
        #    connect_timeout 10
        #    nb_get_retry 3
        #    delay_before_retry 3
        #    connect_port 80
        #}
        HTTP_GET {
          url {
                path /
                status_code 200
          }
          connect_timeout 10
          nb_get_retry 3
          delay_before_retry 3
        }
     }
}
```

LVS+Keepalived-3

- 真实服务器上的配置

真实服务器上需要对VIP进行绑定，并进行路由设置等一系列操作，这里整理为一个脚本供参考：lvs_real.sh：

```
#!/bin/sh
VIP=192.168.1.77
/etc/rc.d/init.d/functions
case "$1" in
start)
    echo " start tunl port"
    ifconfig lo:0 $VIP netmask 255.255.255.255 broadcast $VIP up
    echo "2">/proc/sys/net/ipv4/conf/all/arp_announce
    echo "1">/proc/sys/net/ipv4/conf/all/arp_ignore
    echo "2">/proc/sys/net/ipv4/conf/lo/arp_announce
    echo "1">/proc/sys/net/ipv4/conf/lo/arp_ignore
    ;;
stop)
    echo " stop tunl port"
    ifconfig lo:0 down
```

LVS+Keepalived-4

```
        echo "0">/proc/sys/net/ipv4/conf/all/arp_announce
        echo "0">/proc/sys/net/ipv4/conf/all/arp_ignore
        echo "0">/proc/sys/net/ipv4/conf/lo/arp_announce
        echo "0">/proc/sys/net/ipv4/conf/lo/arp_ignore
        ;;

*)
        echo "Usage: $0 {start|stop}"
        exit 1
esac
```

- 说明

1：基本的使用方法：sh lvs_real.sh start，如果没有权限的话，需要授权

2：VIP被绑定在环回接口lo:0上，其广播地址是其本身，子网掩码是255.255.255.255。采用这种可变长掩码方式可以避免IP地址冲突

3：echo “1”,echo “2” 这段的作用是抑制ARP（地址广播协议）广播