

使用**keepalived**+宝塔面板**nginx**实现**Web**的双主多集群负载均衡

1，环境介绍。

主机：LB01，IP: 192.168.137.30，centos7，BT面板（只装nginx），Keepalived.

主机：LB02，IP: 192.168.137.31，centos7，BT面板（只装nginx），Keepalived.

docker主机：IP: 192.168.137.32，centos7，docker，安装4个nginx容器模拟4个web站点，两个业务集群，如下：

192.168.137.32:8341，OA业务集群站点

192.168.137.32:8342，OA业务集群站点

192.168.137.32:8343，shop业务集群站点

192.168.137.32:8344，shop业务集群站点

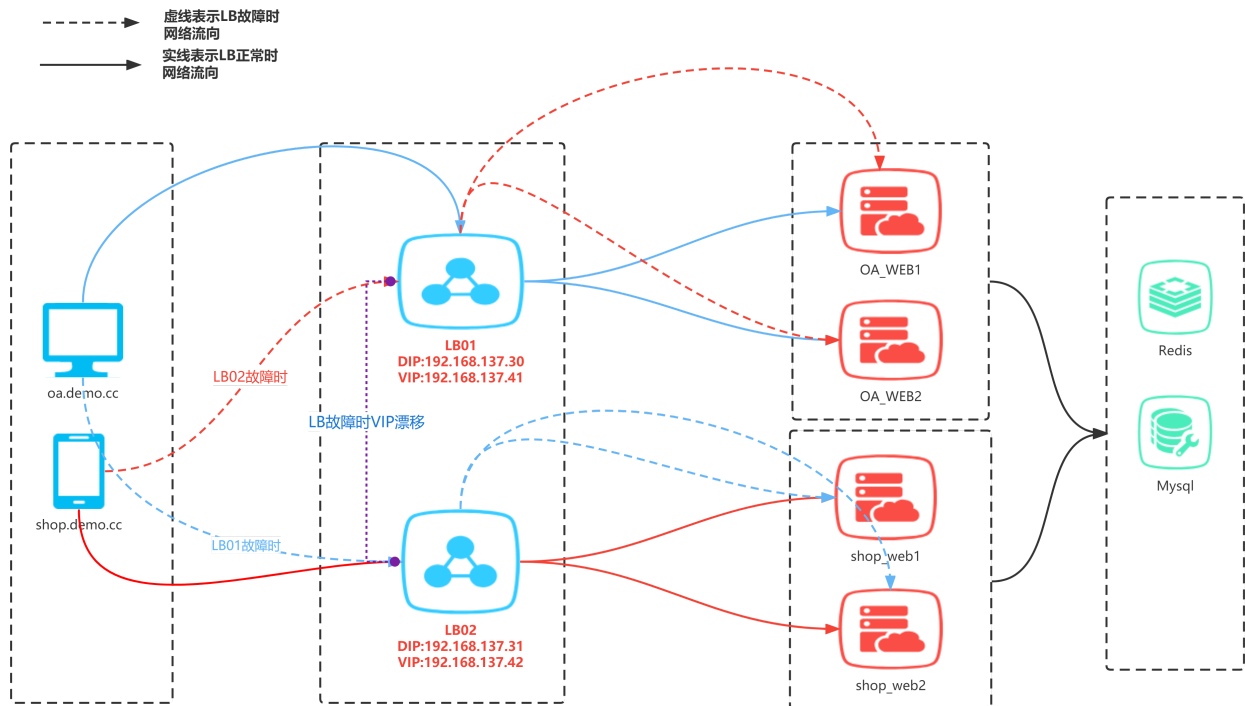
VIP: 192.168.137.41，OA业务的域名 oa.demo.cc

VIP: 192.168.137.42 shop业务的域名 shop.demo.cc

2，集群架构。

采用两个vrrp实例，priority的优先级不一样，实现两台负载均衡主机都会启用一个vip。

keepalived多主业务集群架构



3, keepalived配置

以下是第一台keepalived主机的keepalived.conf配置文件内容:

```
global_defs {
    notification_email {
        root@localhost
    }
    notification_email_from Alexandre.Cassen@firewall1.1oc
    smtp_server 127.0.0.1
    smtp_connect_timeout 30
    router_id web-1                                     ##两台主机的router_id要配置不同的值
    vrrp_skip_check_adv_addr
    # vrrp_strict
    vrrp_garp_interval 0
    vrrp_gna_interval 0
}

vrrp_instance VI_1 {
    state BACKUP
    interface eth0
```

```

    virtual_router_id 55                ## VI_1与VI_2的virtual_router_id
配置不同的值
    priority 150                        ## 主机1比主机2的值大,主机1才可以启动
VI_1实例的vip
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.137.41/24              ## VI_1与VI_2的
virtual_router_id配置不同IP,实现每台主机启用不同的ip
    }
}

vrrp_instance VI_2 {
    state MASTER
    interface eth0
    virtual_router_id 56
    priority 100                        ## 主机1比主机2的值小,主机2才可以启动
VI_2实例的vip
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.137.42/24              ## VI_1与VI_2的
virtual_router_id配置不同IP,实现每台主机启用不同的ip
    }
}

```

以下是第2台keepalived主机的keepalived.conf配置文件内容:

```

global_defs {
    notification_email {
        root@localhost
    }
    notification_email_from Alexandre.Cassen@firewall1.loc
    smtp_server 127.0.0.1
    smtp_connect_timeout 30
    router_id web-2
}

```

```
vrrp_skip_check_adv_addr
# vrrp_strict
vrrp_garp_interval 0
vrrp_gna_interval 0
}

vrrp_instance VI_1 {
    state BACKUP
    interface eth0
    virtual_router_id 55
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.137.41/24
    }
}

vrrp_instance VI_2 {
    state MASTER
    interface eth0
    virtual_router_id 56
    priority 150
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.137.42/24
    }
}
```

4, nginx面板配置

1), 两台主机, 安装宝塔面板后, 登陆面板, 只安装nginx就可以, 然后创建一个demo.cc域名的站点, 这里域名可以随意编写。

然后修改nginx的配置文件, 如下图:



2), 两台主机的nginx配置文件内容一样, 面板自带多的内容也可以删除, 具体如下:

```
upstream oa_server_pools { # OA业务服务器池
    server 192.168.137.32:8341 max_fails=2 fail_timeout=3s
weight=1;
    server 192.168.137.32:8342 max_fails=2 fail_timeout=3s
weight=1;
    # 假设为OA业务的两个集群节点
}

upstream shop_server_pools { # shop业务服务器池
    server 192.168.137.32:8343 max_fails=2 fail_timeout=3s
weight=1;
    server 192.168.137.32:8344 max_fails=2 fail_timeout=3s
weight=1;
    # 假设为shop业务的两个集群节点
}
```

```

server
{
    listen 192.168.137.41:80;
    #需要监听第一个vip的80端口
    server_name oa.demo.cc;
    #需要绑定OA业务的域名
    index index.php index.html index.htm default.php default.htm
default.html;
    root /www/wwwroot/demo.cc;
    #配置代理信息
    location / {
        proxy_pass http://oa_server_pools;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header Host $host;
    }
    #SSL-START SSL相关配置，请勿删除或修改下一行带注释的404规则
    #error_page 404/404.html;
    #SSL-END

    #ERROR-PAGE-START 错误页配置，可以注释、删除或修改
    #error_page 404 /404.html;
    #error_page 502 /502.html;
    #ERROR-PAGE-END

    #PHP-INFO-START PHP引用配置，可以注释或修改
    include enable-php-00.conf;
    #PHP-INFO-END

    #REWRITE-START URL重写规则引用,修改后将导致面板设置的伪静态规则失效
    include /www/server/panel/vhost/rewrite/demo.cc.conf;
    #REWRITE-END

    #禁止访问的文件或目录
    location ~
^/(\.user.ini|\.htaccess|\.git|\.svn|\.project|LICENSE|README.md)
    {
        return 404;
    }

    #一键申请SSL证书验证目录相关设置
    location ~ /\.well-known{
        allow all;
    }
}

```

```

}

location ~ .*\. (gif|jpg|jpeg|png|bmp|swf)$
{
    expires      30d;
    error_log    /dev/null;
    access_log   /dev/null;
}

location ~ .*\. (js|css)?$
{
    expires      12h;
    error_log    /dev/null;
    access_log   /dev/null;
}

access_log    /www/wwwlogs/oa.demo.cc.log;
error_log     /www/wwwlogs/oa.demo.cc.error.log;

}

server {
    listen 192.168.137.42:80;
    #需要监控第二个vip的80端口
    server_name shop.demo.cc;
    #绑定shop业务的访问域名,并配置代理
    location / {
        proxy_pass http://shop_server_pools;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header Host $host;
    }
    access_log    /www/wwwlogs/shop.demo.cc.log;
    error_log     /www/wwwlogs/shop.demo.cc.error.log;
}

```

5，测试集群工作情况。

- 1)，当一台主机出故障时，两个vip都会集中到同一个主机上面，这样就可以保证服务不中断。
- 2)，当两台主机正常工作时，两个vip会分布到不同的主机上面，实现资源更合理利用。

6，扩展方案。

域名采用DNS轮询，两台LB负载均衡器启用多个VIP，每个VIP对应不同的业务，多个业务在每个LB上都会流量。
这样就能实现LB高效资源利用。

7，根据扩展方案,调整keepalived配置。

1) OA与SHOP两个业务的域名采用dns轮询解析到不同的vip。

VIP: 192.168.137.41, 192.168.137.43, OA业务的域名 oa.demo.cc

VIP: 192.168.137.42, 192.168.137.44, shop业务的域名 shop.demo.cc

2) LB正常工作时，两个业务的vip，分布到不同的LB主机上，当其中一个LB故障时，所有IP集中到同一台LB主机。

3) LB的配置配置文件中，添加 vrrp_script检测，当检测到nginx进程不在时，关闭keepalived服务。

```
vrrp_script check_nginx {  
    script "/etc/keepalived/bin/check_nginx.sh"          # 检测脚本位置  
    weight -10      # 脚本异常退出后，使keepalived的优先级-10  
    interval 2      # 脚本每2秒执行一次  
}
```

check_nginx.sh示例脚本，当nginx进程不存在时，关闭keepalived服务。内容如下：

```
#!/bin/bash  
A=`ps -C nginx --no-header | grep -v grep | wc -l`  
if [ $A -eq 0 ];then  
    #尝试重新启动nginx  
    /etc/init.d/nginx restart  
    #睡眠3秒  
    sleep 3  
    if [ `ps -C nginx --no-header | grep -v grep | wc -l` -eq 0  
];then  
        #启动失败，将keepalived服务杀死。将vip漂移到其它备份节点  
        killall keepalived  
    fi  
fi
```

当nginx进程不存在时，重启nginx,再检测不到nginx进程，返回状态码，keepalived会进行降级处理。脚本内容如下：


```
count=$(ps -ef |grep nginx |egrep -cv "grep|$$")

if [ "$count" -eq 0 ];then
    /etc/init.d/nginx restart
    sleep 2
    if [ `ps -C nginx --no-header | grep -v grep | wc -l` -eq 0
];then
    exit 1      # 当nginx挂掉后，返回退出状态码1，以使其异常退出
    fi
fi
```

- 4) 实例都采用BACKUP角色，优先级的数值差要小于10，因为之前配置的优先级降低-10。
实例角色一样，优先级高的，会使用vip。
实例角色与优先级一样，先启动的实例，会使用vip。

当这个LB Master节点故障恢复后，由于业务有一定的延时，所以还不想VIP立马漂移回来，就需要添加以下参数。

```
preempt_delay 300    # master，不会立马夺权，而是5分钟后再夺权
```

除了抢占式策略，还有一种是非抢占式策略

这种策略下，主从的配置文件的“state”属性不再区分“MASTER和BACKUP”。都使用“BACKUP”。

不同的keepalived服务器之间通过priority属性值的大小区分主从，最后一个不同就是在state BACKUP下一行添加 no_preempt即可，表示“非抢占”。

抢占与非抢占模式的区别在于：

抢占模式在主服务器挂了之后，由从服务器接手“主服务器”角色；当主服务器恢复则主服务器会抢回其“主服务器”的角色。

非抢占式则是当主服务器恢复后，担当从的角色。二者的区别主要在于此。

LB01与LB02的keepalived优先级配置数值要唯一。

内容如下：

```
global_defs {
    notification_email {
        root@localhost
    }
}
```

```

notification_email_from Alexandre.Cassen@firewall1.loc
smtp_server 127.0.0.1
smtp_connect_timeout 30
router_id mysql-1
# vrrp_skip_check_adv_addr
# vrrp_strict
# vrrp_garp_interval 0
# vrrp_gna_interval 0
}

vrrp_script check_nginx {
    script "/etc/keepalived/bin/check_nginx.sh" # 检测脚本位置
    weight -10 # 脚本异常退出后,使keepalived的优先级-10
    interval 2 # 脚本每2秒执行一次
}

vrrp_instance VI_1 {
    state BACKUP
    # nopreempt #优先级高的LB主机故障恢复后,设置非抢占模式,不会抢回vip
    # preempt_delay 300 #优先级高的LB主机故障恢复后,延时300秒后抢回vip
    interface eth0
    virtual_router_id 55
    priority 150 # LB01是150,LB02使用145
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.137.41/24
    }
    track_script {
        check_nginx # 检测名称,类似函数调用方法
    }
}

vrrp_instance VI_2 {
    state BACKUP

```

```
interface eth0
virtual_router_id 56
priority 145          # LB01是145, LB02使用150
advert_int 1
authentication {
    auth_type PASS
    auth_pass 1111
}
virtual_ipaddress {
    192.168.137.42/24
}
track_script {
    check_nginx      # 检测名称,类似函数调用方法
}
}
```

```
vrrp_instance VI_3 {
    state BACKUP
    interface eth0
    virtual_router_id 57
    priority 150        # LB01是150, LB02使用145
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.137.43/24
    }
    track_script {
        check_nginx
    }
}
```

```
vrrp_instance VI_4 {
    state BACKUP
    interface eth0
    virtual_router_id 58
    priority 145        # LB01是145, LB02使用150
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
}
```

```

    }
    virtual_ipaddress {
        192.168.137.44/24
    }
    track_script {
        check_nginx
    }
}

```

5) nginx负载均衡的配置类似，具体内容如下。

```

upstream oa_server_pools {          # OA业务服务器池
    server 192.168.137.32:8341 max_fails=2 fail_timeout=3s
    weight=1;
    server 192.168.137.32:8342 max_fails=2 fail_timeout=3s
    weight=1;
    # 假设为OA业务的两个集群节点
}

upstream shop_server_pools {        # shop业务服务器池
    server 192.168.137.32:8343 max_fails=2 fail_timeout=3s
    weight=1;
    server 192.168.137.32:8344 max_fails=2 fail_timeout=3s
    weight=1;
    # 假设为shop业务的两个集群节点
}

server
{
    listen 192.168.137.41:80;
    server_name oa.demo.cc;
    index index.php index.html index.htm default.php default.htm
    default.html;
    location / {
        proxy_pass http://oa_server_pools;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header Host $host;
    }
    access_log /www/wwwlogs/oa.demo.cc.log;
    error_log /www/wwwlogs/oa.demo.cc.error.log;
}

```

```
}

server {
    listen 192.168.137.42:80;
    server_name shop.demo.cc;
    location / {
        proxy_pass http://shop_server_pools;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header Host $host;
    }
    access_log /www/wwwlogs/shop.demo.cc.log;
    error_log /www/wwwlogs/shop.demo.cc.error.log;
}

server {
    listen 192.168.137.43:80;
    server_name oa.demo.cc;
    location / {
        proxy_pass http://oa_server_pools;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header Host $host;
    }
    access_log /www/wwwlogs/oa43.demo.cc.log;
    error_log /www/wwwlogs/oa43.demo.cc.error.log;
}

server {
    listen 192.168.137.44:80;
    server_name shop.demo.cc;
    location / {
        proxy_pass http://shop_server_pools;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header Host $host;
    }
    access_log /www/wwwlogs/shop44.demo.cc.log;
    error_log /www/wwwlogs/shop44.demo.cc.error.log;
}
```