学习请参考：https://blog.51cto.com/13570193/2161637

搭建请参考：https://www.cnblogs.com/northeastTycoon/p/10292050.html

一、准备工作

（1）需要两台服务器 192.168.1.129 和 192.168.1.131（公司服务器）

（2）在两台服务器安装 nginx

（3）在两台服务器安装 keepalived

二、安装Nginx

为知笔记地址：CentOS7 下安装nginx

GitHub地址：
https://github.com/wangliu1102/StudyNotes/tree/master/%E5%B0%9A%E7%A1%85%E8%B0%B7Java/%E5%9B%9B%E3%80%81JavaEE%E9%AB%98%E7%BA%A7/6%E3%80%81Nginx/%E5%AE%89%E8%A3%85

在192.168.1.129 和 192.168.1.131两台服务器上安装相同的Nginx，Nginx配置文件设置相同。

三、安装 keepalived

方法一：使用yum命令安装，需要联网，安装之后，在 etc 里面生成目录 keepalived ，有文件 keepalived.conf

```
yum install keepalived -y

#centos7 启动keepalived服务
# 开启自启
systemctl enable keepalived
#启动
systemctl start keepalived

#centos6 启动keepalived服务
/etc/init.d/keepalived start

#查看进程
ps -ef|grep keepalived
```

方法二：离线安装，需要下载压缩包

keepalived安装需要某些依赖：

gcc、openssl（https://www.openssl.org/）、libnl和kernel-headers和libnl-devel（https://pkgs.org/）、libnfnetlink-devel（https://pkgs.org/）

依次下载openssl-xxx.tar.gz，libnl-xxx.rpm和kernel-headers-xxx.rpm和libnl-devel-xxx.rpm，libnfnetlink-devel-xxx.rpm

openssl-xxx.tar.gz解压缩使用编译命令安装，参考CentOS 7.4下安装nginx中OpenSSL源码安装。

gcc安装参考CentOS 7.4下安装nginx中gcc安装。

libnl和kernel-headers和libnl-devel-xxx.rpm和libnfnetlink-devel-xxx.rpm使用如下命令安装：

```
rpm -ivh libnl-1.1.4-3.el7.x86_64.rpm
rpm -ivh kernel-headers-3.10.0-1127.el7.x86_64.rpm
rpm -ivh libnl-devel-1.1.4-3.el7.x86_64.rpm
rpm -ivh libnfnetlink-devel-1.0.1-4.el7.x86_64.rpm
```

安装过程出现错误，加上忽略依赖--force --nodeps即可

```
rpm -ivh xxx.rpm --force --nodeps
```

在线安装：

```
yum -y install libnl libnl-devel
yum install -y libnfnetlink-devel
```

安装keepalived：

在https://www.keepalived.org/software/ 中下载keepalived压缩包，放到服务器上，使用如下命令解压缩（keepalived-2.0.20）：

百度云：

> 链接：https://pan.baidu.com/s/1jCdFuabjyKnQmkzX3GDI4Q
>
> 提取码：je2y

```
tar -xzvf keepalived-2.0.20.tar.gz
```

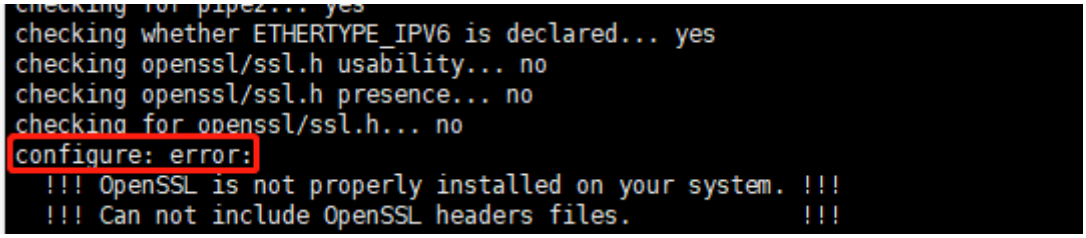
进入解压缩目录下：

```
cd keepalived-2.0.20
```

执行如下命令安装：

```
./configure --prefix=/usr/local/keepalived
```

报错：



```
# 指明openssl安装后的include目录，默认安装到/usr/local/ssl/include
# 这里我们使用openssl源码安装，目录如下
CFLAGS="$CFLAGS -I /home/nginx/openssl-1.0.2n/include" ./configure --
prefix=/usr/local/keepalived
```

使用如下命令编译执行：

```
make && make install

# 查看安装后的keepalived目录信息
whereis keepalived
```

然后，设置快捷方式到keepalived的默认路径下/etc/keepalived：

```
mkdir /etc/keepalived

cp /usr/local/keepalived/etc/keepalived/keepalived.conf
/etc/keepalived/keepalived.conf
cp /usr/local/keepalived/etc/sysconfig/keepalived /etc/sysconfig/keepalived
cp /usr/local/keepalived/sbin/keepalived /usr/sbin/

# 这个从keepalived源码目录复制，安装目录中没有
cp /home/nginx/keepalived-2.0.20/keepalived/etc/init.d/keepalived
/etc/init.d/keepalived
```

设置开机自启：

```
chkconfig keepalived on
chmod +x /etc/init.d/keepalived
```

然后使用如下命令即可启动、停止、重启keepalived：

```
service keepalived start   # 启动服务

service keepalived stop    # 停止服务

service keepalived restart  # 重启服务

service keepalived status  # 查看服务
```

启动过程发生的问题：error while loading shared libraries: libssl.so.1.x: cannot open shared object file: No such file or directory

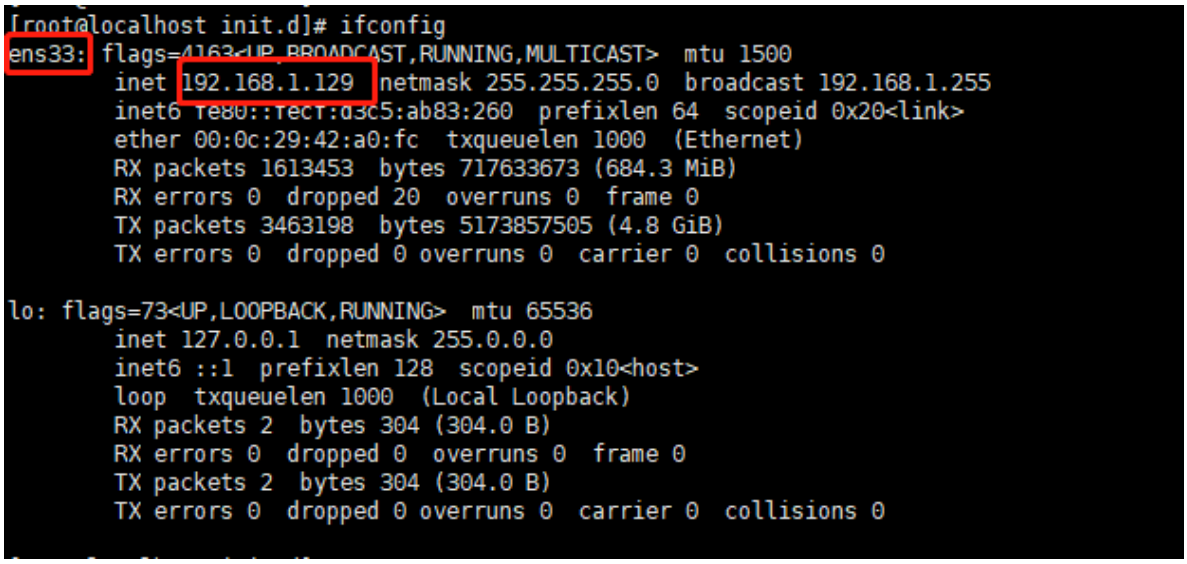
默认libssl.so.1.x(openssl组件）会安装在/usr/local/lib64下面；需要通过软连接放置到/usr/lib64下面：

```
ln -s /usr/local/lib64/libssl.so.1.X /usr/lib64/libssl.so.1.X
ln -s /usr/local/lib64/libcrypto.so.1.X /usr/lib64/libcrypto.so.1.X

# 例如libssl.so.1.0.2k libcrypto.so.1.0.2k
```

启动过程发生的问题：keepalived启动不成功,状态一直是inactive(dead) 的解决办法

```
# 查看ip和网卡
ifconfig
```




```
#打开配置文件 ,在 /etc/keepalived/keepalived.conf 里面修改网卡为ens33
# 如下图，保存退出后就可重新启动
service keepalived start
# 查看
service keepalived status
ps -ef |grep keepalived

# 若启动时报错service keepalived status查看：Active: failed (Result: timeout)
Failed to start LVS and VRRP High Availability Monitor
# ps -ef |grep keepalived查看进程，kill -9 进程号后再重新启动即可
```

```
! Configuration File for keepalived

global_defs {
   notification_email {
     acassen@firewall.loc
     failover@firewall.loc
     sysadmin@firewall.loc
   }
   notification_email_from Alexandre.Cassen@firewall.loc
   smtp_server 192.168.200.1
   smtp_connect_timeout 30
   router_id LVS_DEVEL
   vrrp_skip_check_adv_addr
   vrrp_strict
   vrrp_garp_interval 0
   vrrp_gna_interval 0
}

vrrp_instance VI_1 {
    state MASTER
    interface ens33
    virtual_router_id 51
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.200.16
        192.168.200.17
        192.168.200.18
    }
}
```

```
[root@localhost init.d]# service keepalived status
● keepalived.service - LVS and VRRP High Availability Monitor
   Loaded: loaded (/usr/lib/systemd/system/keepalived.service; enabled; vendor preset: disabled)
   Active: active (running) since 三 2020-05-27 18:40:18 CST; 3s ago
  Process: 68440 ExecStart=/usr/local/keepalived/sbin/keepalived $KEEPALIVED_OPTIONS (code=exited, status=0/SUCCESS)
 Main PID: 68441 (keepalived)
   CGroup: /system.slice/keepalived.service
           ├─68441 /usr/local/keepalived/sbin/keepalived -D
           ├─68442 /usr/local/keepalived/sbin/keepalived -D
           └─68443 /usr/local/keepalived/sbin/keepalived -D

5月 27 18:40:18 localhost.localdomain Keepalived_healthcheckers[68442]: Activating healthchecker for service [192.168.201.100]:tcp:443 for VS [192.168.200.100]:tcp:4
5月 27 18:40:18 localhost.localdomain Keepalived_healthcheckers[68442]: Activating healthchecker for service [192.168.200.2]:tcp:1358 for VS [10.10.10.2]:tcp:1358
5月 27 18:40:18 localhost.localdomain Keepalived_healthcheckers[68442]: Activating healthchecker for service [192.168.200.3]:tcp:1358 for VS [10.10.10.2]:tcp:1358
5月 27 18:40:18 localhost.localdomain Keepalived_healthcheckers[68442]: Activating healthchecker for service [192.168.200.4]:tcp:1358 for VS [10.10.10.3]:tcp:1358
5月 27 18:40:18 localhost.localdomain Keepalived_healthcheckers[68442]: Activating healthchecker for service [192.168.200.5]:tcp:1358 for VS [10.10.10.3]:tcp:1358
5月 27 18:40:18 localhost.localdomain Keepalived_vrrp[68443]: Registering gratuitous ARP shared channel
5月 27 18:40:18 localhost.localdomain Keepalived_vrrp[68443]: (VI_1) removing VIPs.
5月 27 18:40:18 localhost.localdomain Keepalived_vrrp[68443]: (VI_1) removing firewall drop rule
5月 27 18:40:18 localhost.localdomain Keepalived_vrrp[68443]: (VI_1) Entering BACKUP STATE (init)
5月 27 18:40:18 localhost.localdomain Keepalived_vrrp[68443]: VRRP sockpool: [ifindex(2), family(IPv4), proto(112), unicast(0), fd(11,12)]
[root@localhost init.d]#
[root@localhost init.d]#
[root@localhost init.d]#
[root@localhost init.d]#
[root@localhost init.d]#
[root@localhost init.d]#
[root@localhost init.d]#
[root@localhost init.d]# ps -ef |grep keepalived
root      68441      1  0 18:40 ?        00:00:00 /usr/local/keepalived/sbin/keepalived -D
root      68442  68441  0 18:40 ?        00:00:00 /usr/local/keepalived/sbin/keepalived -D
root      68443  68441  3 18:40 ?        00:00:00 /usr/local/keepalived/sbin/keepalived -D
root      68491  66845 19 18:40 pts/0    00:00:00 grep --color=auto keepalived
[root@localhost init.d]#
```

分别在192.168.1.129 和 192.168.1.131两台服务器上安装好keepalived。

四、搭建高可用（主从模式）

1、主节点192.168.1.129

（1）修改/etc/keepalived/keepalived.conf

使用如下命令进入编辑：

```
vim /etc/keepalived/keepalived.conf
```

配置文件部分如下（放到配置文件中注释删掉）：

```
#全局配置
global_defs {
    notification_email {
        acassen@firewall.loc
        failover@firewall.loc
        sysadmin@firewall.loc
    }
    notification_email_from Alexandre.Cassen@firewall.loc c
    smtp_server 192.168.1.129
    smtp_connect_timeout 30
    router_id 192.168.1.129 # 修改为主机ip
}

#脚本配置,当Nginx服务down掉，需要执行脚本内的机制重启nginx或停止keepalived（只有keepalived被关闭时，从节点才能切换成主节点提供服务）
vrrp_script chk_http_port {
    script "/usr/local/src/nginx_check.sh"
    interval 2 #（检测脚本执行的间隔）
    weight -20 #httpd进程出现异常后，该节点keepalived的权值减少20，原本权值为100。为了保证weight的值能够保证脚本在成功与失败后触发主备切换，通常设置weight的绝对值大于主备节点priority之差
}

#虚拟IP配置
vrrp_instance VI_1 {
    state MASTER #  备份服务器上将 MASTER  改为 BACKUP
    interface ens33 # 网卡
    virtual_router_id 233 #  主、备机的 virtual_router_id  必须相同
    priority 100 #  主、备机取不同的优先级，主机值较大，备份机值较小
    advert_int 1
    authentication {
        auth_type PASS # 验证类型为密码认证
        auth_pass 1111 # 验证密码，需要注意密码最大长度为8位
    }

    track_script {
        chk_http_port
    }

    virtual _ipaddress {
        192.168.1.233 #虚拟地址,主、备机的虚拟地址必须相同
    }
}
```

（2）在/usr/local/src 添加检测脚本nginx_check.sh，并授权

```
#!/bin/bash
A=`ps -C nginx --no-header |wc -l`
if [ $A -eq 0 ];then
    /usr/local/nginx/sbin/nginx
    sleep 2
    if [ `ps -C nginx --no-header |wc -l` -eq 0 ];then
    service keepalived stop
    fi
fi
```

```
chmod +x /usr/local/src/nginx_check.sh
```

2、从节点192.168.1.131

（1）修改/etc/keepalived/keepalived.conf

使用如下命令进入编辑：

```
vim /etc/keepalived/keepalived.conf
```

配置文件部分如下（放到配置文件中注释删掉）：

```
#全局配置
global_defs {
    notification_email {
        acassen@firewall.loc
        failover@firewall.loc
        sysadmin@firewall.loc
    }
    notification_email_from Alexandre.Cassen@firewall.loc c
    smtp_server 192.168.1.131
    smtp_connect_timeout 30
    router_id 192.168.1.131  # 修改为主机ip
}

#脚本配置,当Nginx服务down掉，需要执行脚本内的机制重启nginx或停止keepalived（只有
keepalived被关闭时，从节点才能切换成主节点提供服务）
vrrp_script chk_http_port {
    script "/usr/local/src/nginx_check.sh"
    interval 2 #（检测脚本执行的间隔）
    weight -20 #httpd进程出现异常后，该节点keepalived的权值减少20，原本权值为100。为了保
证weight的值能够保证脚本在成功与失败后触发主备切换，通常设置weight的绝对值大于主备节点
priority之差
}

#虚拟IP配置
vrrp_instance VI_1 {
    state BACKUP #  主服务器上将 BACKUP  改为 MASTER
    interface ens33 # 网卡
    virtual_router_id 233 #  主、备机的 virtual_router_id  必须相同
    priority 90 #  主、备机取不同的优先级，主机值较大，备份机值较小
```

```
    advert_int 1
    authentication {
        auth_type PASS # 验证类型为密码认证
        auth_pass 1111 # 验证密码，需要注意密码最大长度为8位
    }

    track_script {
        chk_http_port
    }

    virtual _ipaddress {
        192.168.1.233 #虚拟地址,主、备机的虚拟地址必须相同
    }
}
```

（2）在/usr/local/src 添加检测脚本nginx_check.sh，并授权

```
#!/bin/bash
A=`ps -C nginx --no-header |wc -l`
if [ $A -eq 0 ];then
    /usr/local/nginx/sbin/nginx
    sleep 2
    if [ `ps -C nginx --no-header |wc -l` -eq 0 ];then
    service keepalived stop
    fi
fi
```

```
chmod +x /usr/local/src/nginx_check.sh
```

3、启动两台服务器上的nginx和keepalived

（1）启动nginx，使用如下命令（需要配置环境变量）

修改全局的环境变量，使得nginx命令在任意目录下都能执行,例如：停止nginx -s stop ,重新加载：nginx -s reload ,启动：nginx

```
[root@VM_2_13_centos ~]# vim /etc/profile

export PATH=$PATH:/usr/local/nginx/sbin


# 生效
source /etc/profile
```

```
nginx -s stop #停止
nginx # 启动
nginx -s reload # 重新加载
```

（2）启动keepalived

```
service keepalived start   # 启动服务

service keepalived stop    # 停止服务

service keepalived restart  # 重启服务
```

（3）使用如下命令查看主节点192.168.1.129的虚拟IP：192.168.1.233是否配置成功，从节点没有

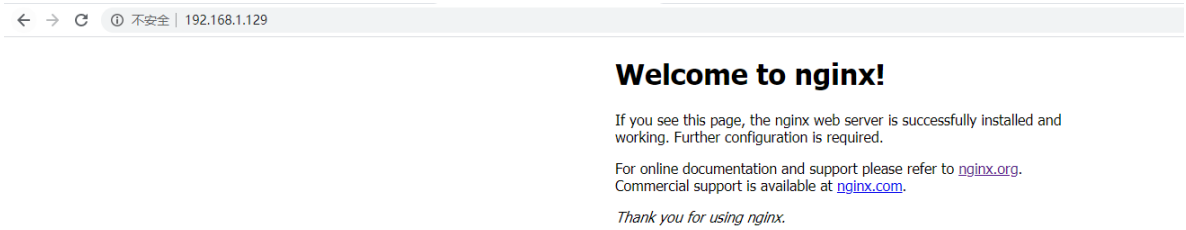
```
ip a
```

```
[root@localhost keepalived]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:42:a0:fc brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.129/24 brd 192.168.1.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet 192.168.1.233/32 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::fecf:d3c5:ab83:260/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

```
[root@server1 keepalived]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:49:ea:4f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.131/24 brd 192.168.1.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::dfb8:99bb:e3bd:c06c/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@server1 keepalived]#
```

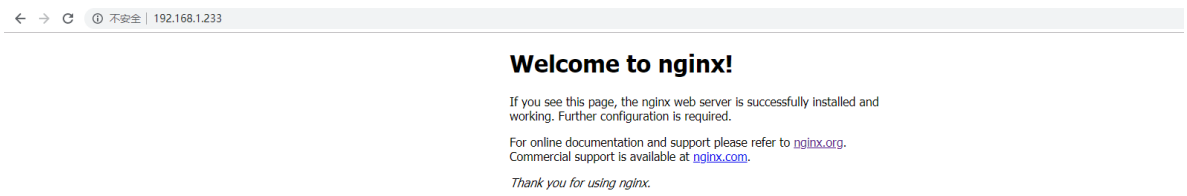
4、测试

浏览器访问192.168.1.129或192.168.1.131，都可访问到Nginx页面



此时，通过虚拟IP：192.168.1.233也可访问。

注意：nginx配置过的端口，这里都可以通过虚拟ip:端口访问。比如nginx配置了8080端口的服务，这里通过192.168.1.233:8080就能访问服务。



主节点192.168.1.129关闭keepalived：service keepalived stop ，从节点192.168.1.131抢占了虚拟ip， 通过虚拟IP：192.168.1.233也可访问

```
[root@localhost keepalived]# service keepalived stop
Stopping keepalived (via systemctl):                       [  确定  ]
[root@localhost keepalived]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:42:a0:fc brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.129/24 brd 192.168.1.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::fecf:d3c5:ab83:260/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@localhost keepalived]#
```

```
[root@server1 keepalived]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:49:ea:4f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.131/24 brd 192.168.1.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet 192.168.1.233/32 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::dfb8:99bb:e3bd:c06c/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@server1 keepalived]#
```

主节点192.168.1.129再次开启keepalived：service keepalived start，主节点192.168.1.129又抢占了虚拟ip，从节点192.168.1.131虚拟ip消失， 通过虚拟IP：192.168.1.233也可访问

```
[root@localhost keepalived]# service keepalived start
Starting keepalived (via systemctl):                       [  确定  ]
[root@localhost keepalived]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:42:a0:fc brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.129/24 brd 192.168.1.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet 192.168.1.233/32 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::fecf:d3c5:ab83:260/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@localhost keepalived]#
```

```
[root@server1 keepalived]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 10
    link/ether 00:0c:29:49:ea:4f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.131/24 brd 192.168.1.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::dfb8:99bb:e3bd:c06c/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@server1 keepalived]#
```

5、防火墙配置

Keepalived是一个轻量级的HA集群解决方案，但开启防火墙后各节点无法感知其它节点的状态，各自都绑定了虚拟IP。网上很多文章讲要配置防火墙放过tcp/112，在CentOS7下是无效的，正确的做法是配置放过vrrp协议，方法如下：

```
# centos7防火墙配置
firewall-cmd --direct --permanent --add-rule ipv4 filter INPUT 0 --destination
224.0.0.18 --protocol vrrp -j ACCEPT
firewall-cmd --direct --permanent --add-rule ipv4 filter OUTPUT 0 --destination
224.0.0.18 --protocol vrrp -j ACCEPT
firewall-cmd --reload
```

Keepalived使用vrrp组播，默认地址是224.0.0.18，因此要配置防火墙放过。

完成后再用ip a查看，集群已经正常了，只有主节点绑定虚拟IP，备份节点不会绑定了。

五、搭建高可用（主主模式）

还是按照上面的环境继续做实验，只是修改 LB 节点上面的 keepalived 服务的配置文件即可。此时192.168.1.129节点即为 Keepalived 的主节点也为备节点，192.168.1.131 节点同样即为 Keepalived 的主节点也为备节点。

192.168.1.129节点默认的主节点 VIP（192.168.1.233），192.168.1.131 节点默认的主节点 VIP（192.168.1.244）

1、即主也从节点192.168.1.129

（1）修改/etc/keepalived/keepalived.conf

使用如下命令进入编辑：

```
vim /etc/keepalived/keepalived.conf
```

配置文件部分如下（放到配置文件中注释删掉）：

```
#全局配置
global_defs {
    notification_email {
        acassen@firewall.loc
        failover@firewall.loc
        sysadmin@firewall.loc
    }
    notification_email_from Alexandre.Cassen@firewall.loc c
    smtp_server 192.168.1.129
    smtp_connect_timeout 30
    router_id 192.168.1.129 # 修改为主机ip
}

#脚本配置,当Nginx服务down掉，需要执行脚本内的机制重启nginx或停止keepalived（只有
keepalived被关闭时，从节点才能切换成主节点提供服务）
vrrp_script chk_http_port {
    script "/usr/local/src/nginx_check.sh"
    interval 2 #（检测脚本执行的间隔）
    weight -20 #httpd进程出现异常后，该节点keepalived的权值减少20，原本权值为100。为了保
证weight的值能够保证脚本在成功与失败后触发主备切换，通常设置weight的绝对值大于主备节点
priority之差
}

#虚拟IP配置
vrrp_instance VI_1 {
```

```
    state MASTER #  备份服务器上将 MASTER  改为 BACKUP
    interface ens33 # 网卡
    virtual_router_id 233 #  主、备机的 virtual_router_id  必须相同
    priority 100 #  主、备机取不同的优先级，主机值较大，备份机值较小
    advert_int 1
    authentication {
        auth_type PASS # 验证类型为密码认证
        auth_pass 1111 # 验证密码，需要注意密码最大长度为8位
    }

    track_script {
        chk_http_port
    }

    virtual _ipaddress {
        192.168.1.233  #虚拟地址,主、备机的虚拟地址必须相同
        # 虚拟ip网段要和real server 真实ip的网络地址一致,比如 192.168.171.128,那么虚拟
ip必须是 192.168.171.* ,否则虚拟ip无法访问
    }
}

vrrp_instance VI_2 {
    state BACKUP
    interface ens33 # 网卡
    virtual_router_id 244 #  主、备机的 virtual_router_id  必须相同
    priority 90 #  主、备机取不同的优先级，主机值较大，备份机值较小
    advert_int 1
    authentication {
        auth_type PASS # 验证类型为密码认证
        auth_pass 1111 # 验证密码，需要注意密码最大长度为8位
    }

    track_script {
        chk_http_port
    }

    virtual _ipaddress {
        192.168.1.244  #虚拟地址,主、备机的虚拟地址必须相同
    }
}
```

（2）在/usr/local/src 添加检测脚本nginx_check.sh，并授权

```
#!/bin/bash
A=`ps -C nginx --no-header |wc -l`
if [ $A -eq 0 ];then
    /usr/local/nginx/sbin/nginx
    sleep 2
    if [ `ps -C nginx --no-header |wc -l` -eq 0 ];then
    service keepalived stop
    fi
fi
```

```
chmod +x /usr/local/src/nginx_check.sh
```

2、即主也从节点192.168.1.131

（1）修改/etc/keepalived/keepalived.conf

使用如下命令进入编辑：

```
vim /etc/keepalived/keepalived.conf
```

配置文件部分如下（放到配置文件中注释删掉）：

```
#全局配置
global_defs {
    notification_email {
        acassen@firewall.loc
        failover@firewall.loc
        sysadmin@firewall.loc
    }
    notification_email_from Alexandre.Cassen@firewall.loc c
    smtp_server 192.168.1.131
    smtp_connect_timeout 30
    router_id 192.168.1.131  # 修改为主机ip
}

#脚本配置,当Nginx服务down掉，需要执行脚本内的机制重启nginx或停止keepalived（只有
keepalived被关闭时，从节点才能切换成主节点提供服务）
vrrp_script chk_http_port {
    script "/usr/local/src/nginx_check.sh"
    interval 2 #（检测脚本执行的间隔）
    weight -20 #httpd进程出现异常后，该节点keepalived的权值减少20，原本权值为100。为了保
证weight的值能够保证脚本在成功与失败后触发主备切换，通常设置weight的绝对值大于主备节点
priority之差
}

#虚拟IP配置
vrrp_instance VI_1 {
    state BACKUP #  主服务器上将 BACKUP  改为 MASTER
    interface ens33 # 网卡
    virtual_router_id 233 #  主、备机的 virtual_router_id  必须相同
    priority 90 #  主、备机取不同的优先级，主机值较大，备份机值较小
    advert_int 1
    authentication {
        auth_type PASS # 验证类型为密码认证
        auth_pass 1111 # 验证密码，需要注意密码最大长度为8位
    }

    track_script {
        chk_http_port
    }

    virtual _ipaddress {
        192.168.1.233  #虚拟地址,主、备机的虚拟地址必须相同
```

```
        # 虚拟ip网段要和real server 真实ip的网络地址一致,比如 192.168.171.128,那么虚拟ip必须是 192.168.171.* ,否则虚拟ip无法访问
    }
}

vrrp_instance VI_2 {
    state MASTER
    interface ens33 # 网卡
    virtual_router_id 244 #  主、备机的 virtual_router_id  必须相同
    priority 100 #  主、备机取不同的优先级，主机值较大，备份机值较小
    advert_int 1
    authentication {
        auth_type PASS # 验证类型为密码认证
        auth_pass 1111 # 验证密码，需要注意密码最大长度为8位
    }

    track_script {
        chk_http_port
    }

    virtual _ipaddress {
        192.168.1.244  #虚拟地址,主、备机的虚拟地址必须相同
    }
}
```

（2 ）在/usr/local/src 添加检测脚本nginx_check.sh，并授权

```
#!/bin/bash
A=`ps -C nginx --no-header |wc -l`
if [ $A -eq 0 ];then
    /usr/local/nginx/sbin/nginx
    sleep 2
    if [ `ps -C nginx --no-header |wc -l` -eq 0 ];then
    service keepalived stop
    fi
fi
```

```
chmod +x /usr/local/src/nginx_check.sh
```

3、启动两台服务器上的nginx和keepalived

（1）启动nginx，使用如下命令（需要配置环境变量）

修改全局的环境变量，使得nginx命令在任意目录下都能执行,例如：停止nginx -s stop ,重新加载：nginx -s reload ,启动：nginx

```
[root@VM_2_13_centos ~]# vim /etc/profile

export PATH=$PATH:/usr/local/nginx/sbin
```

```
nginx -s stop #停止
nginx # 启动
nginx -s reload # 重新加载
```

（2）启动keepalived

```
service keepalived start   # 启动服务

service keepalived stop    # 停止服务

service keepalived restart  # 重启服务
```

（3）使用如下命令查看主节点虚拟IP：192.168.1.233和192.168.1.244是否配置成功

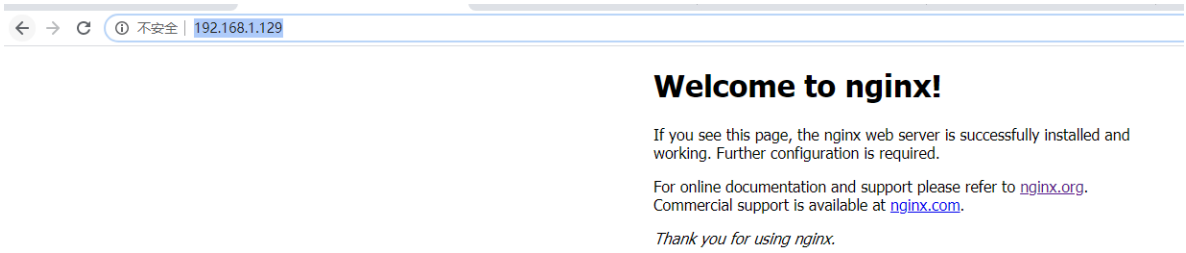
```
ip a
```

```
[root@localhost keepalived]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:42:a0:fc brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.129/24 brd 192.168.1.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet 192.168.1.233/32 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::fecf:d3c5:ab83:260/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

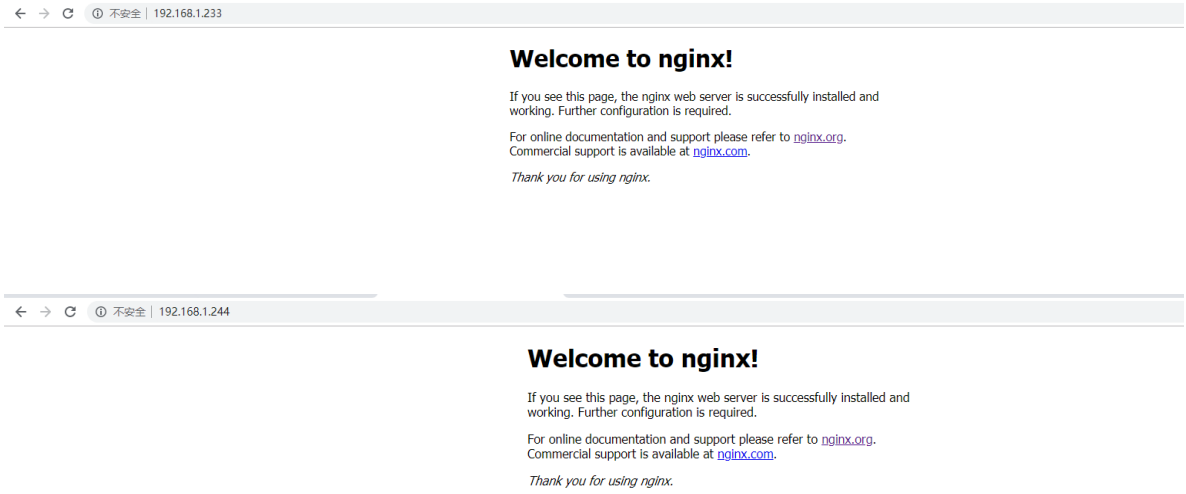
```
[root@server1 keepalived]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:49:ea:4f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.131/24 brd 192.168.1.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet 192.168.1.244/32 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::dfb8:99bb:e3bd:c06c/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@server1 keepalived]#
```

4、测试

浏览器访问192.168.1.129或192.168.1.131，都可访问到Nginx页面



此时，通过虚拟IP：192.168.1.233或192.168.1.244也可访问



5、防火墙配置

Keepalived是一个轻量级的HA集群解决方案，但开启防火墙后各节点无法感知其它节点的状态，各自都绑定了虚拟IP。网上很多文章讲要配置防火墙放过tcp/112，在CentOS7下是无效的，正确的做法是配置放过vrrp协议，方法如下：

```
# centos7防火墙配置
firewall-cmd --direct --permanent --add-rule ipv4 filter INPUT 0 --destination
224.0.0.18 --protocol vrrp -j ACCEPT
firewall-cmd --direct --permanent --add-rule ipv4 filter OUTPUT 0 --destination
224.0.0.18 --protocol vrrp -j ACCEPT
firewall-cmd --reload
```

Keepalived使用vrrp组播，默认地址是224.0.0.18，因此要配置防火墙放过。

完成后再用ip a查看，集群已经正常了，只有主节点绑定虚拟IP，备份节点不会绑定了。

六、keepalived实现高可用ipvs(未能搞通)

keepalived的另一个重要的配置段是关于LVS的配置，LVS配置段是实现LVS高可用功能。该配置段以virtual_server为开始标识。

LVS配置段参数	具体含义
virtual_server	LVS配置段开始标识，格式为virtual_server vip port {...}
delay_loop	对后端服务器集群进行健康状态监测的时间间隔，单位是秒
lb_algo	定义负载均衡的调度算法，有rr，wrr，lc，wlc，lblc，sh，dh等
lb_kind	定义LVS的工作模式，有NAT、DR和TUN三种模式
persistence_timeout	定义ipvs的持久连接时长
protocol	ipvs服务的协议类型，目前keepalived仅支持ipvs的TCP协议
sorry_server	指定备用后端服务器的IP地址，仅在所有real server失效后，备用节点才会生效，格式为*sorryserver ip port*
real_server	后端真实服务器的配置段的开始标识，格式为*realserver ip port {...}*

算法	说明
rr	轮询算法，它将请求依次分配给不同的rs节点，也就是RS节点中均摊分配。这种算法简单，但只适合于RS节点处理性能差不多的情况
wrr	加权轮训调度，它将依据不同RS的权值分配任务。权值较高的RS将优先获得任务，并且分配到的连接数将比权值低的RS更多。相同权值的RS得到相同数目的连接数。
Wlc	加权最小连接数调度，假设各台RS的全职依次为Wi，当前tcp连接数依次为Ti，依次去Ti/Wi为最小的RS作为下一个分配的RS
Dh	目的地址哈希调度（destination hashing）以目的地址为关键字查找一个静态hash表来获得需要的RS
SH	源地址哈希调度（source hashing）以源地址为关键字查找一个静态hash表来获得需要的RS
Lc	最小连接数调度（least-connection）,IPVS表存储了所有活动的连接。LB会比较将连接请求发送到当前连接最少的RS.
Lblc	基于地址的最小连接数调度（locality-based least-connection）：将来自同一个目的地址的请求分配给同一台RS，此时这台服务器是尚未满负荷的。否则就将这个请求分配给连接数最小的RS，并以它作为下一次分配的首先考虑。

官方三种负载均衡技术比较总结表：

工作模式	VS/NAT	VS/TUN	VS/DR
Real server（节点服务器）**	Config dr gw	Tunneling	Non-arp device/tie vip
Server Network	Private	LAN/WAN	LAN
Server number（节点数量）**	Low 10-20	High 100	High 100
Real server gateway	Load balance	Own router	Own router
优点	地址和端口转换	Wan环境加密数据	性能最高
缺点	效率低	需要隧道支持	不能跨域LAN

real_server段配置参数	具体含义
weight	设置后端服务器节点的权值，数字越大权值越大
notify_up	当后端节点切换为UP状态触发的脚本，格式为*notifyup scriptlocation arg1 arg2 …*，功能类似于notify_master参数
notify_down	当后端节点切换为DOWN状态触发的脚本
健康监测段配置	HTTP_GET、SSL_GET、TCP_CHECK、SMTP_CHECK、MISC_CHECK

健康监测端配置参数	具体含义
HTTP_GET、SSL_GET	这两个参数是基于应用层的检测方式，格式为HTTP_GET {…}
TCP_CHECK	基于四层的监测方式，格式为TCP_CHECK {…}

应用层检测配置段的参数： HTTP_GET|SSL_GET { url { path /index.html status_code 200 digest xxxxxxxx

}

nb_get_retry 3 delay_before_retry 2 connect_ip connect_port bindto bind_port connect_timeout } url：指定HTTP/SSL监控的URL信息. path：定义要监控的详细的URL status_code：指定返回http检测正常的状态码类型，就是当返回指定的状态码时即可认定节点正常，一般为200。 digest：status_code定义的状态码并不准确，即使返回200的状态码，还是有网页内容被篡改的可能，这样就无法发现错误信息，因此加入了digest参数，对网页内容的摘要信息进行比对，如果一致则认为页面没有发生改变。该摘要信息可以使用命令genhash生成。

```
genhash -s 192.168.239.129 -p 80 -u /index.html
```

nb_get_retry：重试的次数 delay_before_retry：重试之前等待的时间延迟,即两次重试之间的间隔，单位是秒 上边的两个参数是在检测到错误信息之后才会生效。 connect_ip：向当前RS的哪个IP地址发送健康状态监测信息 connect_port：向当前RS的哪个端口发送健康状态监测信息 如果connect_ip和connect_port都没有指定，则默认使用real_server参数指定的IP和port。 bindto：指定负载均衡器对

RS发送健康状态监测的源IP地址 bind_port：指定负载均衡器对RS发送健康状态监测的源端口 connect_timeout：定义健康状态监测的连接超时时间 **基于四层监测配置段参数**： TCP_CHECK { connect_ip connect_port bindto bind_port connect_timeout }

根据需求，在两台服务器的/etc/keepalived/keepalivec.conf 配置文件中添加如下的内容（**主从模式**）：

```
virtual_server 192.168.1.233 80 {
    delay_loop 6
    lb_algo wrr
    lb_kind DR
    persistence_timeout 50
    protocol TCP

    real_server 192.168.1.129 80 {
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 3
            nb_get_retry 3
            delay_before_retry 3
        }
    }
    real_server 192.168.1.131 80 {
        weight 1
        HTTP_GET {
            url {
              path /
              status_code 200
            }
            connect_timeout 3
            nb_get_retry 3
            delay_before_retry 3
        }
    }

}
```

```
yum -y install ipvsadm
ipvsadm -Ln
#想知道当前测试机的访问请求被转发到那个服务器去了，可以在 ipvsadm 命令后带一个选项，其完整形式为:
ipvsadm -lcn | grep 192.168.1.233
```

编写脚本 vim lvs_dr.sh

```
#!/bin/bash
#虚拟ip
vip=192.168.1.233
mask='255.255.255.255'
dev=lo:1
case $1 in
start)
    echo 1 > /proc/sys/net/ipv4/conf/all/arp_ignore
    echo 1 > /proc/sys/net/ipv4/conf/lo/arp_ignore
    echo 2 > /proc/sys/net/ipv4/conf/all/arp_announce
    echo 2 > /proc/sys/net/ipv4/conf/lo/arp_announce
    ifconfig $dev $vip netmask $mask #broadcast $vip up
    #route add -host $vip dev $dev
    echo "The RS Server is Ready!"
    ;;
stop)
    ifconfig $dev down
    echo 0 > /proc/sys/net/ipv4/conf/all/arp_ignore
    echo 0 > /proc/sys/net/ipv4/conf/lo/arp_ignore
    echo 0 > /proc/sys/net/ipv4/conf/all/arp_announce
    echo 0 > /proc/sys/net/ipv4/conf/lo/arp_announce
    echo "The RS Server is Canceled!"
    ;;
*)
    echo "Usage: $(basename $0) start|stop"
    exit 1
    ;;
esac
```

```
#授权
chmod u+x  lvs_dr.sh
#执行
bash -x  lvs_dr.sh start

# 重启后生效
service keepalived restart
```