1、主机环境

```
teleport-1、mysql-1、nginx、keepalived(backup): 192.168.56.4
teleport-2、mysql-2、nginx、keepalived(backup): 192.168.56.5
keepalived vip: 192.168.56.8
```

2、安装相关包

2.1、安装nginx

```
rpm -Uvh http://nginx.org/packages/centos/7/noarch/RPMS/nginx-release-centos-7-0.el7.ngx.noarch.rpm
yum -y install nginx
```

2.2、安装keepalived相关包

```
yum install -y psmisc ipvsadm  keepalived
```

3、mysql双主配置

3.1、mysql-1 mysql配置文件

> /opt/teleport_docker_compose/data/etc/my.cnf

```
[mysql]

[mysqld]
pid-file    = /var/run/mysqld/mysqld.pid
socket        = /var/run/mysqld/mysqld.sock
datadir        = /var/lib/mysql
#log-error    = /var/log/mysql/error.log
# By default we only accept connections from localhost
#bind-address    = 127.0.0.1
# Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links=0

lower_case_table_names = 1 #不区分大小写
character_set_server = utf8 #字符编码

log-bin=mysql-bin # 开启bin-log 日志，MySQL主从配置，必须开启
log-bin-index=mysql-bin

server_id=1 # 唯一的标识，与slave不同

log-slave-updates = true # 双主互备必须开启，否则只是主从关系
relay-log= relaylog
relay-log-index=relaylog
relay-log-purge=on

binlog-do-db=teleport #开启同步的数据库

#auto-increment-increment = 2
#auto-increment-offset = 1
```

3.2、mysql-2 mysql配置文件

> /opt/teleport_docker_compose/data/etc/my.cnf

```
[mysql]

[mysqld]
pid-file    = /var/run/mysqld/mysqld.pid
socket        = /var/run/mysqld/mysqld.sock
datadir        = /var/lib/mysql
#log-error    = /var/log/mysql/error.log
# By default we only accept connections from localhost
#bind-address    = 127.0.0.1
# Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links=0
```

```
lower_case_table_names = 1 #不区分大小写
character_set_server = utf8 #字符编码

log-bin=mysql-bin # 开启bin-log 日志，MySQL主从配置，必须开启
log-bin-index=mysql-bin

server_id=2 # 唯一的标识，与master不同

log-slave-updates = true # 双主互备必须开启，否则只是主从关系
relay-log= relaylog
relay-log-index=relaylog
relay-log-purge=on

binlog-do-db=teleport #开启同步的数据库

#auto-increment-increment = 2
#auto-increment-offset = 2
```

> 注：二都只有server-id不同和auto-increment-offset不同
> auto-increment-offset是用来设定数据库中自动增长的起点的，为这两能服务器都设定了一次自动增长值2，所以它们的起点必须得不同，这样才能避免两台服务器数据同步时出现主键冲突
>
> replicate-do-db 指定同步的数据库，我们只在两台服务器间同步test数据库
>
> 另：auto-increment-increment的值应设为整个结构中服务器的总数，本案例用到两台mysql服务器，所以值设为2

3.3、配置mysql双主

3.3.1、在mysql-1上查看master状态

```
docker exec -it mysql-1 mysql -uroot -p12wsxCDE#

mysql> show master status;
+------------------+----------+--------------+------------------+-------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------------+----------+--------------+------------------+-------------------+
| mysql-bin.000003 |      154 | teleport     |                  |                   |
+------------------+----------+--------------+------------------+-------------------+
1 row in set (0.01 sec)

mysql>
```

3.3.2、在mysql-2上配置数据库同步

```
docker exec -it mysql-2 mysql -uroot -p12wsxCDE#

mysql> change master to master_host='172.17.0.2',master_user='root',master_password='root',master_port=3306, master_log_file='mysql-bin.
Query OK, 0 rows affected, 2 warnings (0.05 sec)

mysql> start slave;
Query OK, 0 rows affected (0.04 sec)

mysql> show slave status\G;
...
            Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
...

mysql> show master status;
+------------------+----------+--------------+------------------+-------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------------+----------+--------------+------------------+-------------------+
| mysql-bin.000003 |      154 | teleport     |                  |                   |
+------------------+----------+--------------+------------------+-------------------+
1 row in set (0.00 sec)

mysql>
```

3.3.3、在mysql-1上配置数据库同步

```
docker exec -it mysql-1 mysql -uroot -p12wsxCDE#

mysql> change master to master_host='172.17.0.3',master_user='root',master_password='root',master_port=3306, master_log_file='mysql-bin.
Query OK, 0 rows affected, 2 warnings (0.05 sec)
```

```
mysql> start slave;
Query OK, 0 rows affected (0.00 sec)

mysql> show slave status\G;
...
            Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
...
```

4、nginx配置

4.1、nginx主配置文件

> /etc/nginx/nginx.conf

```
# grep "include /etc/nginx/" /etc/nginx/nginx.conf
include /etc/nginx/conf.d/*.conf;
include /etc/nginx/tcp.d/*.conf;
```

4.2、nginx反向代理配置文件

> /etc/nginx/conf.d/teleport.conf

```
upstream teleport {
    server 127.0.0.1:7190; #teleport端口
}
server {
    listen       80;
    server_name  localhost;
    access_log   /var/log/nginx/teleport.log;

    location / {
        proxy_pass  http://teleport;

        #Proxy Settings
        proxy_redirect     off;
        proxy_set_header   Host             $host;
        proxy_set_header   X-Real-IP        $remote_addr;
        proxy_set_header   X-Forwarded-For  $proxy_add_x_forwarded_for;
        proxy_next_upstream error timeout invalid_header http_500 http_502 http_503 http_504;
        proxy_max_temp_file_size 0;
        proxy_connect_timeout      90;
        proxy_send_timeout         90;
        proxy_read_timeout         90;
        proxy_buffer_size          4k;
        proxy_buffers              4 32k;
        proxy_busy_buffers_size    64k;
        proxy_temp_file_write_size 64k;

        # 以下三行是websocket需要的
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
   }

}
```

4.3、tcp反向代理配置文件

> /etc/nginx/tcp.d/mysql.conf

```
stream {
    # 添加socket转发的代理
    upstream mysql {
        hash $remote_addr consistent;
        # 转发的目的地址和端口
        server 192.168.56.4:3306 weight=5 max_fails=3 fail_timeout=30s;
        server 192.168.56.5:3306 weight=5 max_fails=3 fail_timeout=30s;
    }

    # 提供转发的服务，即访问localhost:port，会跳转至代理teleport指定的转发地址
    server {
       listen 3308;
```

```
        proxy_connect_timeout 30s;
        proxy_timeout 30s;
        proxy_pass mysql;
    }

}
```

4.4、重新加载nginx

```
nginx -s reload
```

5、teleport配置

5.1、teleport docker-compose配置文件

> /opt/teleport_docker_compose/docker-compose.yml

```
version: '3.1'
services:
  db:
    image: harbor.mxnet.io/library/mysql:5.7
    container_name: mysql
    volumes:
      - /etc/localtime:/etc/localtime:ro
      - ./data/etc/my.cnf:/etc/mysql/my.cnf:ro
      - ./data/db:/var/lib/mysql
    restart: always
    command: [
      "--character-set-server=utf8mb4",
      "--collation-server=utf8mb4_unicode_ci",
      "--innodb_flush_log_at_trx_commit=1",
      "--sync_binlog=1"
      ]
    environment:
      MYSQL_ROOT_PASSWORD: 12wsxCDE#
      MYSQL_DATABASE: teleport
      MYSQL_USER: teleport
      MYSQL_PASSWORD: 12wsxCDE#
    ports:
      - 3306:3306

  teleport:
    build: .
    image: harbor.mxnet.io/library/tp4a/teleport:v3.2.2
    container_name: teleport
    depends_on:
      - db
    tty: true
    command: bash -c "/usr/local/teleport/start.sh && tail -f /usr/local/teleport/data/log/*.log"
    volumes:
      - /etc/localtime:/etc/localtime:ro
      - ./data/etc:/usr/local/teleport/data/etc
      - ./data/replay:/usr/local/teleport/data/replay
      - ./data/log:/usr/local/teleport/data/log
    ports:
      - 7190:7190
      - 127.0.0.1:52080:52080
      - 52089:52089
      - 52189:52189
      - 52389:52389
```

5.2、teleport核心服务配置文件

> /opt/teleport_docker_compose/data/etc/core.ini

```
; codec: utf-8

[common]
; 'log-file' define the log file location. if not set, default locate
; to $INSTDIR%/data/log/tpcore.log
;log-file=/var/log/teleport/tpcore.log

; log-level can be 0 ~ 4, default value is 2.
; LOG_LEVEL_DEBUG     0   log every-thing.
; LOG_LEVEL_VERBOSE   1   log every-thing but without debug message.
; LOG_LEVEL_INFO      2   log infomation/warning/error message.
; LOG_LEVEL_WARN      3   log warning and error message.
; LOG_LEVEL_ERROR     4   log error message only.
log-level=2

; 0/1. default to 0.
; in debug mode, `log-level` force to 0 and display more message for debug purpose.
debug-mode=0

; 'replay-path' define the replay file location. if not set, default locate
; to `$INSTDIR%/data/replay`
;replay-path=/var/lib/teleport/replay

; `web-server-rpc` is the rpc interface of web server.
; default to `http://127.0.0.1:7190/rpc`.
; DO NOT FORGET update this setting if you modified common::port in web.ini.
web-server-rpc=http://127.0.0.1:7190/rpc

[rpc]
; Request by web server. `bind-ip` should be the ip of core server. If web server and
; core server running at the same machine, it should be `127.0.0.1`.
; DO NOT FORGET update `common::core-server-rpc` in web.ini if you modified this setting.
bind-ip=127.0.0.1
bind-port=52080

[protocol-ssh]
enabled=true
lib=tpssh
bind-ip=0.0.0.0
bind-port=52189

[protocol-rdp]
enabled=true
lib=tprdp
bind-ip=0.0.0.0
bind-port=52089

[protocol-telnet]
enabled=true
lib=tptelnet
bind-ip=0.0.0.0
bind-port=52389
```

5.3、teleport web服务配置文件

> /opt/teleport_docker_compose/data/etc/web.ini

```
; codec: utf-8

[common]

; ip=0.0.0.0

; port listen by web server, default to 7190.
; DO NOT FORGET update `common::web-server-rpc` in core.ini if you modified this setting.
port=7190

; log file of web server, default to /var/log/teleport/tpweb.log
log-file=/usr/local/teleport/data/log/tpweb.log

; `log-level` can be 0 ~ 4, default to 2.
; LOG_LEVEL_DEBUG     0   log every-thing.
; LOG_LEVEL_VERBOSE   1   log every-thing but without debug message.
; LOG_LEVEL_INFO      2   log information/warning/error message.
; LOG_LEVEL_WARN      3   log warning and error message.
; LOG_LEVEL_ERROR     4   log error message only.
log-level=0

; 0/1. default to 0.
; in debug mode, `log-level` force to 0 and display more message for debug purpose.
debug-mode=0

; `core-server-rpc` is the rpc interface of core server.
; default to `http://127.0.0.1:52080/rpc`.
; DO NOT FORGET update this setting if you modified rpc::bind-port in core.ini.
core-server-rpc=http://127.0.0.1:52080/rpc

[database]
```

```
; database in use, should be sqlite/mysql, default to sqlite.
type=mysql

; sqlite-file=/usr/local/teleport/data/db/teleport.db

mysql-host=db

mysql-port=3306

mysql-db=teleport

mysql-prefix=tp_

mysql-user=teleport

mysql-password=12wsxCDE#
```

5.4、启动teleport

```
cd /opt/teleport_docker_compose && docker-compose up -d
```

5.5、停止并移除teleport

```
cd /opt/teleport_docker_compose && docker-compose down -v
```

6、keepalived配置

6.1、keepalived master配置文件

> /etc/keepalived/keepalived.conf

```
! Configuration File for keepalived     #全局定义

global_defs {
notification_email {     #指定keepalived在发生事件时(比如切换)发送通知邮件的邮箱
admin@cpms.com.cn   #设置报警邮件地址，可以设置多个，每行一个。 需开启本机的sendmail服务
zabbix@cpms.com.cn
}

notification_email_from admin@cpms.com.cn   #keepalived在发生诸如切换操作时需要发送email通知地址
smtp_server 10.75.13.2      #指定发送email的smtp服务器
smtp_connect_timeout 30    #设置连接smtp server的超时时间
router_id master-node     #运行keepalived的机器的一个标识，通常可设为hostname。故障发生时，发邮件时显示在邮件主题中的信息。
}

vrrp_script chk_nginx {      #检测nginx服务是否在运行。有很多方式，比如进程，用脚本检测等等
    script "/etc/keepalived/nginx_check.sh"   #这里通过脚本监测
    interval 2                   #脚本执行间隔，每2s检测一次
    weight -5                    #脚本结果导致的优先级变更，检测失败（脚本返回非0）则优先级 -5
    fall 2                    #检测连续2次失败才算确定是真失败。会用weight减少优先级（1-255之间）
    rise 1                    #检测1次成功就算成功。但不修改优先级
}

vrrp_instance VI_1 {    #keepalived在同一virtual_router_id中priority（0-255）最大的会成为master，也就是接管VIP，当priority最大的主机发生故障后次
    state BACKUP    #指定keepalived的角色，MASTER表示此主机是主服务器，BACKUP表示此主机是备用服务器。注意这里的state指定instance(Initial)的初始状态
    interface eth1          #指定HA监测网络的接口。实例绑定的网卡，因为在配置虚拟IP的时候必须是在已有的网卡上添加的
    mcast_src_ip 192.168.56.4  # 发送多播数据包时的源IP地址，这里注意了，这里实际上就是在哪个地址上发送VRRP通告，这个非常重要，一定要选择稳定的网卡端口
    virtual_router_id 51         #虚拟路由标识，这个标识是一个数字，同一个vrrp实例使用唯一的标识。即同一vrrp_instance下，MASTER和BACKUP必须是一致的
    priority 98                 #定义优先级，数字越大，优先级越高，在同一个vrrp_instance下，MASTER的优先级必须大于BACKUP的优先级
    advert_int 1                 #设定MASTER与BACKUP负载均衡器之间同步检查的时间间隔，单位是秒
    authentication {             #设置验证类型和密码。主从必须一样
        auth_type PASS           #设置vrrp验证类型，主要有PASS和AH两种
        auth_pass 1111           #设置vrrp验证密码，在同一个vrrp_instance下，MASTER与BACKUP必须使用相同的密码才能正常通信
    }
    virtual_ipaddress {          #VRRP HA 虚拟地址 如果有多个VIP，继续换行填写
        192.168.56.8
    }

track_script {                      #执行监控的服务。注意这个设置不能紧挨着写在vrrp_script配置块的后面（实验中碰过的坑），否则nginx监控失效！！
   chk_nginx                    #引用VRRP脚本，即在 vrrp_script 部分指定的名字。定期运行它们来改变优先级，并最终引发主备切换。
}
}
```

6.2、keepalived slave配置文件

> /etc/keepalived/keepalived.conf

```
! Configuration File for keepalived

global_defs {
notification_email {
admin@cpms.com.cn
zabbix@cpms.com.cn
}

notification_email_from admin@cpms.com.cn
smtp_server 10.75.13.2
smtp_connect_timeout 30
router_id slave-node
}

vrrp_script chk_nginx {
    script "/etc/keepalived/nginx_check.sh"
    interval 2
    weight -5
    fall 2
    rise 1
}

vrrp_instance VI_1 {
    state BACKUP
    interface eth1
    mcast_src_ip 192.168.56.5
    virtual_router_id 51
    priority 98
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        192.168.56.8
    }

track_script {
   chk_nginx
}
}
```

6.3、keepalived nginx_check.sh

> /etc/keepalived/nginx_check.sh

```
#!/bin/bash
counter=$(ps -C nginx --no-heading|wc -l)
if [ "${counter}" = "0" ]; then
    /usr/sbin/nginx
    sleep 2
    counter=$(ps -C nginx --no-heading|wc -l)
    if [ "${counter}" = "0" ]; then
        systemctl stop keepalived
    fi
fi
```

6.4、iptables设置

```
iptables -I INPUT -s 192.168.56.0/24 -d 224.0.0.18 -j ACCEPT #允许组播地址通信
iptables -I INPUT -s 0.0.0.0/0 -p vrrp -j ACCEPT #允许 VRRP（虚拟路由器冗余协）通信
iptables -I INPUT -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT #开通80端口访问
```

6.5、启动keepalived

```
systemctl start keepalived
```

6.6、查看keepalived状态

```
systemctl status keepalived
```

6.7、查看keepalived日志

```
tail -f /var/log/messages
```

6.8、检测vip

```
ip a |grep 192.168.56.8
```

6.9、tcpdump查看VRRP包

```
tcpdump -i eth1|grep VRRP
```

6.10、更新arp信息

```
#arping -I 网卡名 -c 5 -s vip 网关
arping -I eth1 -c 5 -s 192.168.56.8 192.168.56.1
```