centos7.6搭建Mysql集群（基于MHA）

1.基础环境

软件名称	内容
系统版本	CentOS-7-aarch64-Everything-1810
数据库版本	mysql-5.7.27-aarch64
MHA版本	mha0.58-centos
VIP	192.168.101.250/24

角色	IP	server-id	读写权限
master/mha-node	192.168.101.223/24	1	读写
slave1/mha-node	192.168.101.224/24	2	只读
slave2/mha-manager	192.168.101.225/24	3	只读

2.安装mysql

请查阅文档《centos7.6ARM架构安装MySql5.7.22.pdf》，这里不再赘述。

如果三个mysql是克隆复制，需要更改auto.cnf文件的uuid，保证每个都不同。

3.关闭防火墙

查看防火墙状态

```
firewall-cmd --state
```

关闭防火墙

```
systemctl stop firewalld.service
```

禁止防火墙开机自启

```
systemctl disable firewalld.service
```

关闭selinux，打开/etc/selinux/config文件,SELINUX=enforcing改为SELINUX=disabled

```
[root@master ~]# vim /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are
protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

4.配置ssh免密登录

配置hosts文件，三个服务器均要做此操作。

```
[root@master ~]# vim /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.101.223  master
192.168.101.224  slave1
192.168.101.225  slave2
```

在每一台服务器上做如下操作。

```
[root@master ~]# ssh-keygen -t rsa                    #然后三个回车即可
[root@master ~]# ssh-copy-id master
[root@master ~]# ssh-copy-id slave1
[root@master ~]# ssh-copy-id slave2
```

三台服务器配置完成后，可以使用ssh检验免密登录是否成功。

```
[root@master ~]# ssh slave1
Last login: Fri Sep 27 14:28:40 2019 from 192.168.101.41
[root@slave1 ~]#
```

5.配置mysql主从复制

1.在每个服务器上修改my.cnf文件如下：

```
[root@master ~]# vim /etc/my.cnf
[client]
user=root
password=123456
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
#每个server上不一致，见规划
server-id = 1
```

```
#read-only=1    #不在配置文件中限定只读，但是要记得在slave上限制只读
#mysql5.6已上的特性，开启gtid，必须主从全开
gtid_mode = on
enforce_gtid_consistency = 1
log_slave_updates = 1
#开启半同步复制   否则自动切换主从的时候会报主键错误
plugin_load =
"rpl_semi_sync_master=semisync_master.so;rpl_semi_sync_slave=semisync_slave.so"
loose_rpl_semi_sync_master_enabled = 1
loose_rpl_semi_sync_slave_enabled = 1
loose_rpl_semi_sync_master_timeout = 5000
log-bin=mysql-bin
relay-log = mysql-relay-bin
replicate-wild-ignore-table=mysql.%
replicate-wild-ignore-table=test.%
replicate-wild-ignore-table=information_schema.%
```

2.在三个mysql节点做授权配置（主从复制授权）

```
mysql> grant replication slave on *.* to 'repl_user'@'192.168.101.%' identified
by '123456';
mysql> grant all on *.* to 'root'@'192.168.101.%' identified by '123456';
```

3.在两个slave节点做只读限制，防止意外写入数据导致数据无法同步

```
mysql> set global read_only=1;
```

4.在master节点查看状态。

```
mysql> show master status\G;
*************************** 1. row ***************************
             File: mysql-bin.000007
         Position: 798461
     Binlog_Do_DB:
 Binlog_Ignore_DB:
Executed_Gtid_Set: 81011f97-e007-11e9-b0dd-44004d371d98:1-4166
1 row in set (0.00 sec)
```

5.在两个slave节点执行如下操作。

```
    mysql> change master to
master_host='192.168.101.223',master_user='repl_user',master_password='123456',m
aster_log_file='mysql-bin.000007',master_log_pos=798461;
###########其中master_log_file文件看master状态的File，master_log_pos就是master状态的
Position
    mysql> start slave;
    mysql> show slave status\G;
    *************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.101.223
                  Master_User: repl_user
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000007
          Read_Master_Log_Pos: 798461
```

```
               Relay_Log_File: mysql-relay-bin.000002
                Relay_Log_Pos: 320
        Relay_Master_Log_File: mysql-bin.000007
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
#############看到IO_Running和SQL_Running为Yes说明是正常的。IO负责与主机通信，SQL线程执行主从复制。
```

6.安装mha-node

三个mysql节点都需要安装。在每台服务器上操作如下.

先安装依赖

```
[root@master ~]# wget http://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
[root@master ~]# rpm -ivh epel-release-latest-7.noarch.rpm
[root@master ~]# yum install perl-DBD-MySQL perl-Config-Tiny perl-Log-Dispatch perl-Parallel-ForkManager -y
```

安装mha-node,两种下载方式选择其一即可。

```
[root@master ~]# wget https://qiniu.wsfnk.com/mha4mysql-node-0.58-0.el7.centos.noarch.rpm
/**
[root@master ~]# wget https://github.com/yoshinorim/mha4mysql-node/releases/download/v0.58/mha4mysql-node-0.58-0.el7.centos.noarch.rpm
**/
[root@master ~]# rpm -ivh mha4mysql-node-0.58-0.el7.centos.noarch.rpm
```

7.安装mha-manager

manager只需要在一台服务器上安装，当前是选择slave2安装manager。在slave2上做如下操作

两种下载方式，任选其一。

```
/**
[root@slave2 ~]# wget https://qiniu.wsfnk.com/mha4mysql-manager-0.58-0.el7.centos.noarch.rpm
**/
[root@slave2 ~]# wget https://github.com/yoshinorim/mha4mysql-manager/releases/download/v0.58/mha4mysql-manager-0.58-0.el7.centos.noarch.rpm
[root@slave2 ~]# rpm -ivh mha4mysql-manager-0.58-0.el7.centos.noarch.rpm
[root@slave2 ~]# yum install mailx -y //该软件是用来发送邮件的
```

8.配置mha-manager节点

创建目录，存放脚本，配置全局文件

```
[root@slave2 ~]# mkdir -p /etc/mha/scripts
[root@slave2 ~]# vi /etc/masterha_default.cnf
[server default]
user=root
password=123456
ssh_user=root
```

```
repl_user=repl_user
repl_password=123456
ping_interval=1
#master_binlog_dir= /var/lib/mysql,/var/log/mysql
secondary_check_script=masterha_secondary_check -s 192.168.101.223 -s
192.168.101.224 -s 192.168.101.225
master_ip_failover_script="/etc/mha/scripts/master_ip_failover"
master_ip_online_change_script="/etc/mha/scripts/master_ip_online_change"
report_script="/etc/mha/scripts/send_report
```

配置主配置文件

```
[root@slave2 ~]# vim /etc/mha/app1.cnf
[server default]
manager_log=/var/log/mha/app1/manager.log
manager_workdir=/var/log/mha/app1

[server1]
hostname=192.168.101.223
candidate_master=1
master_binlog_dir="/usr/local/mysql/data"

[server2]
candidate_master=1
hostname=192.168.101.224
master_binlog_dir="/usr/local/mysql/data"

[server3]
hostname=192.168.101.225
master_binlog_dir="/usr/local/mysql/data"
no_master=1
```

配置VIP

为了防止脑裂发生,推荐生产环境采用脚本的方式来管理虚拟 ip,而不是使用 keepalived来完成。

```
[root@slave2 ~]# vim /etc/mha/scripts/master_ip_failover

#!/usr/bin/env perl
use strict;
use warnings FATAL => 'all';
use Getopt::Long;
my (
    $command,   $ssh_user,      $orig_master_host,
    $orig_master_ip,$orig_master_port, $new_master_host,
$new_master_ip,$new_master_port
);
#定义VIP变量
#注意这个ens33是你自己网卡信息的名称
my $vip = '192.168.101.250/24';
my $key = '1';
my $ssh_start_vip = "/sbin/ifconfig enp125s0f0:$key $vip";
my $ssh_stop_vip = "/sbin/ifconfig enp125s0f0:$key down";
GetOptions(
    'command=s'         => \$command,
    'ssh_user=s'                => \$ssh_user,
    'orig_master_host=s'        => \$orig_master_host,
```

```
    'orig_master_ip=s'  => \$orig_master_ip,
    'orig_master_port=i'        => \$orig_master_port,
    'new_master_host=s' => \$new_master_host,
    'new_master_ip=s'   => \$new_master_ip,
    'new_master_port=i' => \$new_master_port,
);
exit &main();
sub main {
    print "\n\nIN SCRIPT TEST====$ssh_stop_vip==$ssh_start_vip===\n\n";
    if ( $command eq "stop" || $command eq "stopssh" ) {
        my $exit_code = 1;
        eval {
            print "Disabling the VIP on old master: $orig_master_host \n";
            &stop_vip();
            $exit_code = 0;
        };
        if ($@) {
            warn "Got Error: $@\n";
            exit $exit_code;
        }

    elsif ( $command eq "start" ) {
    my $exit_code = 10;
    eval {
        print "Enabling the VIP - $vip on the new master - $new_master_host \n";
        &start_vip();
        $exit_code = 0;
    };

    if ($@) {
        warn $@;
        exit $exit_code;
        }
    exit $exit_code;
    }

    elsif ( $command eq "status" ) {
        print "Checking the Status of the script.. OK \n";
        exit 0;
    }
    else {
        &usage();
        exit 1;
    }
}

sub start_vip() {
    `ssh $ssh_user\@$new_master_host \" $ssh_start_vip \"`;
}
sub stop_vip() {
    return 0 unless ($ssh_user);
    `ssh $ssh_user\@$orig_master_host \" $ssh_stop_vip \"`;
}
sub usage {
    print
    "Usage: master_ip_failover --command=start|stop|stopssh|status --
orig_master_host=host --orig_master_ip=ip --orig_master_port=port --
new_master_host=host --new_master_ip=ip --new_master_port=port\n";
```

```
}
```

配置邮件告警脚本

mail邮件发送程序，需要先配置好发送这信息

```
[root@slave2 ~]# vim /etc/mail.rc
set from=822102408@qq.com
set smtp=smtp.qq.com
set smtp-auth-user=822102408
#拿163邮箱来说这个不是密码，而是授权码
set smtp-auth-password=wqwqwq123
set smtp-auth=login
```

具体邮件发送脚本

```
[root@slave2 ~]# vim /etc/mha/scripts/send_report
#!/bin/bash
source /root/.bash_profile
# 解析变量
orig_master_host=`echo "$1" | awk -F = '{print $2}'`
new_master_host=`echo "$2" | awk -F = '{print $2}'`
new_slave_hosts=`echo "$3" | awk -F = '{print $2}'`
subject=`echo "$4" | awk -F = '{print $2}'`
body=`echo "$5" | awk -F = '{print $2}'`
#定义收件人地址
email="822102408@qq.com"
tac /var/log/mha/app1/manager.log | sed -n 2p | grep 'successfully' > /dev/null
if [ $? -eq 0 ]
    then
    messages=`echo -e "MHA $subject 主从切换成功\n master:$orig_master_host -->
$new_master_host \n $body \n 当前从库:$new_slave_hosts"`
    echo "$messages" | mail -s "Mysql 实例宕掉，MHA $subject 切换成功" $email
>>/tmp/mailx.log 2>&1
    else
    messages=`echo -e "MHA $subject 主从切换失败\n master:$orig_master_host -->
$new_master_host \n $body" `
    echo "$messages" | mail -s ""Mysql 实例宕掉，MHA $subject 切换失败"" $email
>>/tmp/mailx.log 2>&1
fi
```

配置VIP脚本

```
[root@slave2 ~]# vim  /etc/mha/scripts/master_ip_online_change
#!/bin/bash
source /root/.bash_profile
vip=`echo '192.168.101.250/24'`  #设置VIP
key=`echo '1'`
command=`echo "$1" | awk -F = '{print $2}'`
orig_master_host=`echo "$2" | awk -F = '{print $2}'`
new_master_host=`echo "$7" | awk -F = '{print $2}'`
orig_master_ssh_user=`echo "${12}" | awk -F = '{print $2}'`
new_master_ssh_user=`echo "${13}" | awk -F = '{print $2}'`
#要求服务的网卡识别名一样，都为ens33(这里是)
stop_vip=`echo "ssh root@$orig_master_host /usr/sbin/ifconfig enp125s0f0:$key
down"`
```

```
start_vip=`echo "ssh root@$new_master_host /usr/sbin/ifconfig enp125s0f0:$key
$vip"`

if [ $command = 'stop' ]
  then
    echo -e "\n\n\n*****************************\n"
    echo -e "Disabled thi VIP - $vip on old master: $orig_master_host \n"
    $stop_vip
    if [ $? -eq 0 ]
      then
    echo "Disabled the VIP successfully"
      else
    echo "Disabled the VIP failed"
    fi
    echo -e "*****************************\n\n\n"
  fi

if [ $command = 'start' -o $command = 'status' ]
  then
    echo -e "\n\n\n*************************\n"
    echo -e "Enabling the VIP - $vip on new master: $new_master_host \n"
    $start_vip
    if [ $? -eq 0 ]
      then
    echo "Enabled the VIP successfully"
      else
    echo "Enabled the VIP failed"
    fi
    echo -e "*************************\n\n\n"
fi
```

给脚本赋权可执行

```
[root@slave2 ~]# chmod +x /etc/mha/scripts/master_ip_failover
[root@slave2 ~]# chmod +x /etc/mha/scripts/master_ip_online_change
[root@slave2 ~]# chmod +x /etc/mha/scripts/send_report
```

验证ssh是否成功

```
[root@slave2 ~]# masterha_check_ssh --conf=/etc/mha/app1.cnf
Sun Sep 29 15:28:26 2019 - [info] Reading default configuration from
/etc/masterha_default.cnf..
Sun Sep 29 15:28:26 2019 - [info] Reading application default configuration from
/etc/mha/app1.cnf..
Sun Sep 29 15:28:26 2019 - [info] Reading server configuration from
/etc/mha/app1.cnf..
Sun Sep 29 15:28:26 2019 - [info] Starting SSH connection tests..
Sun Sep 29 15:28:27 2019 - [debug]
Sun Sep 29 15:28:26 2019 - [debug]  Connecting via SSH from
root@192.168.101.223(192.168.101.223:22) to
root@192.168.101.224(192.168.101.224:22)..
Sun Sep 29 15:28:26 2019 - [debug]   ok.
Sun Sep 29 15:28:26 2019 - [debug]  Connecting via SSH from
root@192.168.101.223(192.168.101.223:22) to
root@192.168.101.225(192.168.101.225:22)..
Sun Sep 29 15:28:27 2019 - [debug]   ok.
```

```
Sun Sep 29 15:28:28 2019 - [debug]
Sun Sep 29 15:28:27 2019 - [debug]  Connecting via SSH from
root@192.168.101.224(192.168.101.224:22) to
root@192.168.101.223(192.168.101.223:22)..
Sun Sep 29 15:28:27 2019 - [debug]   ok.
Sun Sep 29 15:28:27 2019 - [debug]  Connecting via SSH from
root@192.168.101.224(192.168.101.224:22) to
root@192.168.101.225(192.168.101.225:22)..
Sun Sep 29 15:28:27 2019 - [debug]   ok.
Sun Sep 29 15:28:29 2019 - [debug]
Sun Sep 29 15:28:27 2019 - [debug]  Connecting via SSH from
root@192.168.101.225(192.168.101.225:22) to
root@192.168.101.223(192.168.101.223:22)..
Sun Sep 29 15:28:27 2019 - [debug]   ok.
Sun Sep 29 15:28:27 2019 - [debug]  Connecting via SSH from
root@192.168.101.225(192.168.101.225:22) to
root@192.168.101.224(192.168.101.224:22)..
Sun Sep 29 15:28:28 2019 - [debug]   ok.
Sun Sep 29 15:28:29 2019 - [info] All SSH connection tests passed successfully.
```

验证mysql主从复制是否正常

```
mkdir /var/log/mha/app1 -p[root@slave2 ~]# masterha_check_repl --
conf=/etc/mha/app1.cnf
Checking the Status of the script.. OK
Wed May 16 23:59:42 2018 - [info]  OK.
Wed May 16 23:59:42 2018 - [warning] shutdown_script is not defined.
Wed May 16 23:59:42 2018 - [info] Got exit code 0 (Not master dead).
MySQL Replication Health is OK.
```

9.启动MHA（注意：MHA监控脚本切换一次就会退出，需要再次启动）

首先在master上面绑定vip。

```
[root@master ~]# /usr/sbin/ifconfig enp125s0f0:1 192.168.101.250/24
```

启动mha-manager

```
[root@slave2 ~]# mkdir /var/log/mha/app1 -p
[root@slave2 ~]# touch /var/log/mha/app1/manager.log
[root@slave2 ~]# nohup masterha_manager --conf=/etc/mha/app1.cnf --
remove_dead_master_conf --ignore_last_failover < /dev/null >
/var/log/mha/app1/manager.log 2>&1 &              ###启动监控进程
[root@slave2 ~]# tail -10 /var/log/mha/app1/manager.log ###查看mha启动状况
[root@slave2 ~]# masterha_check_status --conf=/etc/mha/app1.cnf  #检查集群情况
app1 (pid:16142) is running(0:PING_OK), master:192.168.101.223
```

10.切换测试

A、杀掉主库 mysql 进程,模拟主库发生故障,进行自动 failover 操作。
B、看 MHA 切换日志,了解整个切换过程

```
[root@slave2 ~]# tailf /var/log/mha/app1.log
```

从上面的日志可以看出：

1).配置文件检查阶段,这个阶段会检查整个集群配置文件配置
2).宕机的 master 处理,这个阶段包括虚拟 ip 摘除操作,主机关机操作（由于没有定义power_manager脚本，不会关机）
3).复制 dead maste 和最新 slave 相差的 relay log,并保存到 MHA Manger 具体的目录下
4).识别含有最新更新的 slave
5).应用从 master 保存的二进制日志事件(binlog events)这点信息对于将故障master修复后加入集群很重要
6).提升一个 slave 为新的 master 进行复制
7).使其他的 slave 连接新的 master 进行复制

切换完成后，关注如下现象

1、vip 自动从原来的 master 切换到新的 master,同时,manager 节点的监控进程自动退出。
2、在日志目录(/var/log/mha/app1)产生一个 app1.failover.complete 文件
3、/etc/mha/app1.cnf 配置文件中原来老的 master 配置被删除

11.将故障节点重新加入集群

通常情况下自动切换以后,原 master 可能已经废弃掉;
待原 master 主机修复后,如果数据完整的情况下,可能想把原来 master 重新作为新主库的 slave;
这时我们可以借助当时自动切换时刻的 MHA 日志来完成对原 master 的修复。
(若是下面第二步：出现的是第二种，可以不用借助日志，定位binlog日志点，可以自动定位)

修改manager配置文件，把删除的节点信息重新加入。

```
[root@slave2 ~]# vi /etc/mha/app1.cnf
[server1]
hostname=192.168.101.223
candidate_master=1
master_binlog_dir="/usr/local/mysql/data"
```

修复老的master，设置角色为slave

```
[root@slave2 ~]# cat /var/log/mha/app1/manager.log
如果含有“MASTER_LOG_FILE='mysql-bin.000009', MASTER_LOG_POS=120,类似的用以下命令”
mysql> change master to
master_host='192.168.101.224,master_user='repl_user',master_password='123456',ma
ster_log_file='mysql-bin.000009',master_log_pos=120

如果含有“MASTER_AUTO_POSITION=1”则用以下命令
mysql> change master to
master_host='192.168.101.221',master_user='repl_user',master_password='123456',M
ASTER_AUTO_POSITION=1;
#开启slave:
mysql> start slave;
mysql> show slave status\G;
```

在manager节点重启监控服务

```
[root@slave2 ~]# nohup masterha_manager --conf=/etc/mha/app1.cnf --
remove_dead_master_conf --ignore_last_failover < /dev/null >
/var/log/mha/app1/manager.log 2>&1 &
```