任务背景

一、真实案例

公司现在需要做MySQL数据库迁移，之前数据库是一主两从，使用MHA做了高可用，跑在私有云平台，现在需要这三台数据库迁移到物理机上，而且云上的数据库正在跑，希望做到热迁移，即不影响业务的情况下，把现有数据库从私有云平台迁移到物理真机上。

——来自北京2期李同学

二、案例背后核心技术

1、熟悉MHA高可用的原理和部署

2、掌握MySQL数据库的迁移

3、熟悉MySQL主从复制的模式和搭建（基于GTIDs）

三、今日场景

随着业务功能的逐步完善，现有MySQL数据库架构虽然可以保障数据的相对可靠性，但是不能够完全保障==服务的可用性。==当我们的主库挂掉后，mysql服务不能立马切换到从服务器。所以，需要在现有架构的基础上==扩展和升级，==进而在保障数据的可靠性的同时能够保障服务的可用性。

#任务要求

1、使用三台服务器搭建mysql的复制组

2、使用==MHA==管理复制组，当master挂掉后，会立马提升一台slave作为新的master

#任务拆解

- [ ] 搭建MySQL的复制组（M-S1-S1，并联架构）
- [ ] 安装MHA相关软件来管理复制组

#理论储备

##一、MHA简介

MHA（Master High Availability）目前在MySQL高可用方面是一个相对成熟的解决方案，它由日本DeNA公司youshimaton（现就职于Facebook公司）开发，是一套优秀的作为MySQL高可用性环境下==故障切换和主从提升==的高可用软件。在MySQL故障切换过程中，MHA能做到在0~30秒之内自动完成数据库的故障切换操作，并且在进行故障切换的过程中，MHA能在==较大程度==上保证数据的一致性，以达到真正意义上的高可用。

##二、MHA工作原理

1. 当master出现故障时，通过对比slave之间I/O线程读取master上binlog的位置，选取最接近的slave做为最新的slave（latest slave）。
2. 其它slave通过与latest slave对比==生成差异中继日志，并应用==。
3. 在latest slave上==应用从master保存的binlog==，同时将latest slave==提升为master==。
4. 最后在其它slave上应用相应的差异中继日志并开始从新的master开始复制.

##三、MHA组件

###1、MHA相关组件

- **MHA Manager**(管理节点)

  MHA Manager可以单独部署在一台独立的机器上管理多个==master-slave集群==，也可以部署在一台slave节点上。

- **MHA Node**（数据节点）

  MHA Node运行在==每台MySQL服务器==上，MHA Manager会定时探测集群中的master节点，当master出现故障时，它可以自动将数据的slave提升为新的master，然后将所有其他的slave重新指向新的master。整个故障转移过程对应用程序完全透明。

###2、MHA组件介绍

- MHA Manager

运行一些工具，比如masterha_manager工具实现==自动监控MySQL Master==和实现==master故障切换==，其它工具手动实现master故障切换、在线mater转移、连接检查等等。一个Manager可以管理多 个master-slave集群

- MHA Node

部署在所有运行MySQL的服务器上，无论是master还是slave。主要有三个作用：

1）保存二进制日志

```
如果能够访问故障master，会拷贝master的二进制日志
```

2）应用差异中继日志

从拥有最新数据的slave上生成差异中继日志，然后应用差异日志。

3）清除中继日志

```
在不停止SQL线程的情况下删除中继日志
```

###3、相关工具介绍

㈠ Manager工具

工具	说明
==masterha_check_ssh==	检查MHA的SSH配置
==masterha_check_repl==	检查MySQL复制
==masterha_manager==	启动MHA
==masterha_check_status==	检测当前MHA运行状态
masterha_master_monitor	监测master是否宕机
masterha_master_switch	控制故障转移(自动或手动)
masterha_conf_host	添加或删除配置的server信息

㈡ Node工具

工具	说明
save_binary_logs	保存和复制master的二进制日志
apply_diff_relay_logs	识别差异的中继日志事件并应用于其它slave
filter_mysqlbinlog	去除不必要的ROLLBACK事件(MHA已不再使用这个工具)
purge_relay_logs	清除中继日志(不会阻塞SQL线程)

==*注意：Node这些工具通常由MHA Manager的脚本触发,无需人手操作*==。

#任务解决方案

##一、MHA部署

###1、部署规划

角色	IP	主机名	server-id	功能	备注
MHA-Manager	10.1.1.40	mgr.heima.cc	—	管理节点	
MHA-Node（Master）	10.1.1.10	master.heima.cc	10	数据节点	写
MHA-Node（Slave1）	10.1.1.20	slave1.heima.cc	20	数据节点	读
MHA-Node（Slave2）	10.1.1.30	slave2.heima.cc	30	数据节点	读

###2、系统和软件版本

系统版本	MySQL版本	MHA版本
CentOS 7.6	MySQL-5.7.25	mha4mysql-manager-0.57 mha4mysql-node-0.57

###3、系统环境初始化

####㈠ 修改主机名和hosts

```
# hostnamectl set-hostname master.heima.cc
# hostnamectl set-hostname slave1.heima.cc
# hostnamectl set-hostname slave2.heima.cc
# hostnamectl set-hostname mgr.heima.cc
# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.1.1.10    master.heima.cc  master
10.1.1.20    slave1.heima.cc  slave1
10.1.1.30    slave2.heima.cc  slave2
10.1.1.40    mgr.heima.cc     mgr
```

####㈡ 关闭防火墙和selinux

```
# systemctl stop firewalld
# systemctl disable firewalld
# setenforce 0
# sed -i '/SELINUX=enforcing/cSELINUX=disabled' /etc/selinux/config
```

####㈢ 关闭NetworkManager服务

```
# systemctl stop NetworkManager
# systemctl disable NetworkManager
```

㈣ 配置yum源

说明：每台服务器都需要配置！

方案一：分别配置 aliyun、epel 和本地源;

```
# rpm -ivh /soft/mha/epel-release-latest-7.noarch.rpm
# cat server.repo
[local]
name=local yum
baseurl=file:///mnt
enabled=1
gpgcheck=0
[aliyun]
name=this is aliyun yum
baseurl=http://mirrors.aliyun.com/centos/7/os/x86_64/
enabled=1
gpgcheck=0
```

方案二：配置自建仓库，提前准备好软件包

注意：如果没有网络可以使用本地仓库，提前下载好包

```
# cat server.repo
[local]
name=local yum
baseurl=file:///mnt
enabled=1
gpgcheck=0
[mha]
name=mha soft
baseurl=file:///soft/mha/mha-yum
enabled=1
gpgcheck=0
说明:
1）每台服务器都需要配置该文件
2）每台服务器都需要/soft/mha/mha-yum目录来保存相应的软件包
```

####㈤ 安装依赖包

注意：所有服务器均需要安装

```
yum -y install perl-DBD-MySQL \
perl-Config-Tiny \
perl-Time-HiRes \
perl-Mail-Sender \
perl-Mail-Sendmail \
perl-MIME-Base32 \
perl-MIME-Charset \
perl-MIME-EncWords \
perl-Params-Classify \
perl-Params-Validate.x86_64 \
perl-Log-Dispatch \
perl-Parallel-ForkManager \
net-tools
```

###4、部署MySQL复制环境

####㈠ MySQL部署规划

安装目录	数据库目录	配置文件	套接字文件	端口
/usr/local/mysql	/usr/local/mysql/data	/etc/my.cnf	/usr/local/mysql/mysql.sock	3307

㈡ 搭建主从复制

① 创建配置文件

- master

```
[root@master ~]# cat /etc/my.cnf
[mysqld]
basedir=/usr/local/mysql
datadir=/usr/local/mysql/data
socket=/usr/local/mysql/mysql.sock
port=3307
log-error=/usr/local/mysql/master.err
log-bin=/usr/local/mysql/data/binlog
server-id=10
character_set_server=utf8mb4
gtid-mode=on
log-slave-updates=1
enforce-gtid-consistency
[client]
socket=/usr/local/mysql/mysql.sock
```

- slave1

```
[root@slave1 ~]# cat /etc/my.cnf
[mysqld]
basedir=/usr/local/mysql
datadir=/usr/local/mysql/data
socket=/usr/local/mysql/mysql.sock
port=3307
log-error=/usr/local/mysql/slave1.err
relay-log=/usr/local/mysql/data/relaylog
log-bin=/usr/local/mysql/data/binlog
server-id=20
```

```
character_set_server=utf8mb4
log-slave-updates=1
gtid-mode=on
enforce-gtid-consistency
skip-slave-start
[client]
socket=/usr/local/mysql/mysql.sock
```

- slave2

```
[root@slave2 ~]# cat /etc/my.cnf
[mysqld]
basedir=/usr/local/mysql
datadir=/usr/local/mysql/data
socket=/usr/local/mysql/mysql.sock
port=3307
log-error=/usr/local/mysql/slave2.err
relay-log=/usr/local/mysql/data/relaylog
log-bin=/usr/local/mysql/data/binlog
server-id=30
character_set_server=utf8mb4
log-slave-updates=1
gtid-mode=on
enforce-gtid-consistency
skip-slave-start
[client]
socket=/usr/local/mysql/mysql.sock
```

② 同步数据到从服务器

```
[root@master ~]# rsync -av /usr/local/mysql 10.1.1.20:/usr/local/
[root@master ~]# rsync -av /usr/local/mysql 10.1.1.30:/usr/local/
注意：
保证两台slave服务器上有mysql用户
```

③ 删除auto.cnf文件

```
[root@master data]# rm -f auto.cnf
[root@slave1 data]# rm -f auto.cnf
[root@slave2 data]# rm -f auto.cnf
注意：三台服务器都删除
```

④ 启动数据库

```
[root@master ~]# service mysql start
[root@slave1 ~]# service mysql start
[root@slave2 ~]# service mysql start
```

⑤ master创建复制用户

```
mysql> create user 'slave'@'10.1.1.%' identified by '123';
Query OK, 0 rows affected (0.00 sec)
mysql> grant replication slave on *.* to 'slave'@'10.1.1.%';
Query OK, 0 rows affected (0.00 sec)
mysql> flush privileges;
```

```
Query OK, 0 rows affected (0.00 sec)

说明：创建MHA的监控用户和密码，为后续做准备
mysql> create user 'mha'@'10.1.1.40' identified by '123';
Query OK, 0 rows affected (0.01 sec)

mysql> grant all privileges on *.* to 'mha'@'10.1.1.40';
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

⑥ slave上配置同步信息

注意：两台slave都需要配置

```
mysql> change master to
master_host='10.1.1.10',master_port=3307,master_user='slave',master_password='12
3',master_auto_position=1;

mysql> start slave;
Query OK, 0 rows affected (0.02 sec)
```

⑦ 测试验证

略

###5、MHA软件安装

####㈠ 不同节点安装软件

说明：在所有节点安装 ==mha-node== 软件包，在 ==mha 管理==端再安装 mha-manager 软件包

```
[root@mgr ~]# yum -y install mha4mysql-node-0.57-0.el7.noarch.rpm
[root@master ~]# yum -y install mha4mysql-node-0.57-0.el7.noarch.rpm
[root@slave1 ~]# yum -y install mha4mysql-node-0.57-0.el7.noarch.rpm
[root@slave2 ~]# yum -y install mha4mysql-node-0.57-0.el7.noarch.rpm
[root@mgr ~]# yum -y install mha4mysql-manager-0.57-0.el7.noarch.rpm
```

####㈡ 配置ssh互信

说明：

1. 在生产环境中几乎都是禁止root远程登陆服务器的，所以ssh免密码登陆要在admin用户下进行配置，这是处于安全角度考虑出发。
2. admin用户可以是任意普通用户
3. 该普通用户用于mha的管理节点远程访问mysql复制组中的所有主机，完成一些其他工作

① 所有机器创建admin用户

```
# useradd admin
# echo 123|passwd --stdin admin
```

#####② 配置mgr主机到其他主机的admin用户互信

```
mgr端：
[root@mgr ~]# su - admin
[admin@mgr ~]$ ssh-keygen -P "" -f ~/.ssh/id_rsa
[admin@mgr ~]$ cd .ssh/
[admin@mgr .ssh]$ ls
id_rsa  id_rsa.pub
[admin@mgr .ssh]$ mv id_rsa.pub authorized_keys
[admin@mgr .ssh]$ for i in 10 20 30;do scp -r ../.ssh/ 10.1.1.$i:~/;done

测试免密登录：
[admin@mgr .ssh]$ ssh 10.1.1.10
[admin@mgr .ssh]$ ssh 10.1.1.20
[admin@mgr .ssh]$ ssh 10.1.1.30
```

####㈢ 配置admin用户的sudo权限

- 配置admin用户执行sudo命令权限

```
[root@master ~]# vim /etc/sudoers.d/admin
#User_Alias  表示具有sudo权限的用 户 列表; Host_Alias表示主机的列表
User_Alias MYSQL_USERS = admin
#Runas_Alias  表示用户以什么身份登录
Runas_Alias MYSQL_RUNAS = root
#Cmnd_Alias  表示允许执行命令的列表
Cmnd_Alias MYSQL_CMNDS = /sbin/ifconfig,/sbin/arping
MYSQL_USERS ALL = (MYSQL_RUNAS) NOPASSWD: MYSQL_CMNDS

[root@master ~]# for i in 20 30;do scp /etc/sudoers.d/admin
10.1.1.$i:/etc/sudoers.d/;done
```

- 测试admin用户是否可以挂载VIP

```
[admin@master ~]$ sudo /sbin/ifconfig ens33:1 10.1.1.100 broadcast 10.1.1.255
netmask 255.255.255.0
[admin@master ~]$ sudo /sbin/arping -fqc 5 -w 5 -I ens33 -s 10.1.1.100 -U
10.1.1.10
[admin@master ~]$ ifconfig

补充：
arping：用来向局域网内的其它主机发送ARP请求的指令，可以用来测试局域网内的某个IP是否已被使用。

-f：收到第一个响应包后退出。
-q：quite模式，不显示输出。
-c：发送指定的count个ARP REQUEST包后停止。如果指定了-w参数，则会等待相同数量的ARP REPLY包，直到超时为止。
-w：指定一个超时时间，单位为秒，arping在到达指定时间后退出，无论期间发送或接收了多少包。在这种情况下，arping在发送完指定的count（-c）个包后并不会停止，而是等待到超时或发送的count个包都进行了回应后才会退出。
-I：指定设备名，用来发送ARP REQUEST包的网络设备的名称。
-D：重复地址探测模式，用来检测有没有IP地址冲突，如果没有IP冲突则返回0。
-s：设置发送ARP包的IP资源地址
-U：无理由的（强制的）ARP模式去更新别的主机上的ARP CACHE列表中的本机的信息，不需要响应。
-h：显示帮助页。
```

####㈣ 创建mha相关配置文件

- 创建 mha 相关的工作目录

```
[root@mgr ~]# mkdir /etc/mha/
[root@mgr ~]# mkdir -p /data/mha/masterha/app1
[root@mgr ~]# chown -R admin. /data/mha
```

- 创建mha局部配置文件

```
[root@mgr ~]# cat /etc/mha/app1.conf
[server default]
# 设置监控用户和密码
user=mha
password=123
# 设置复制环境中的复制用户和密码
repl_user=slave
repl_password=123
# 设置ssh的登录用户名
ssh_user=admin
# 设置监控主库,发送ping包的时间间隔,默认是3秒,尝试三次没有回应的时候自动进行failover
ping_interval=3
# 设置mgr的工作目录
manager_workdir=/data/mha/masterha/app1
# 设置mysql master保存binlog的目录,以便MHA可以找到master的二进制日志
master_binlog_dir=/usr/local/mysql/data
# 设置master的pid文件
master_pid_file=/usr/local/mysql/data/master.heima.cc.pid
# 设置mysql master在发生切换时保存binlog的目录（在mysql master上创建这个目录）
remote_workdir=/data/mysql/mha
# 设置mgr日志文件
manager_log=/data/mha/masterha/app1/app1-3307.log
# MHA到master的监控之间出现问题,MHA Manager将会尝试从slave1和slave2登录到master上
secondary_check_script=/usr/bin/masterha_secondary_check -s 10.1.1.20 -s
10.1.1.30 --user=admin --port=22 --master_host=10.1.1.10 --master_port=3307
# 设置自动failover时候的切换脚本
master_ip_failover_script="/etc/mha/master_ip_failover.sh 10.1.1.100 1"
# 设置手动切换时候的切换脚本
#master_ip_online_change_script="/etc/mha/master_ip_online_change.sh 10.1.1.100
1"
# 设置故障发生后关闭故障主机脚本
# shutdown_script="/etc/mha/power_manager"
[server1]
hostname=10.1.1.10
port= 3307
candidate_master=1
[server2]
hostname=10.1.1.20
port= 3307
candidate_master=1
[server3]
hostname=10.1.1.30
port= 3307
candidate_master=1
```

####㈤ 上传相应脚本

```
[root@mgr ~]# ls /etc/mha/
app1.conf  master_ip_failover.sh
注意：脚本内容中要修改网卡名字和连接用户为admin
my $vip  = shift;
my $interface = 'ens33';          网卡名
my $key = shift;
...
sub stop_vip() {
    my $ssh_user = "admin";      用户名
    print "=======$ssh_stop_vip==================\n";
    `ssh $ssh_user\@$orig_master_host \" $ssh_stop_vip \"`;
}

[root@mgr ~]# chmod +x /etc/mha/master_ip_*
```

###6、检查ssh互信和集群状态

- 检查ssh互信

```
[admin@mgr ~]$ masterha_check_ssh --conf=/etc/mha/app1.conf
Sat Apr 11 22:23:59 2020 - [warning] Global configuration file
/etc/masterha_default.cnf not found. Skipping.
Sat Apr 11 22:23:59 2020 - [info] Reading application default configuration from
/etc/mha/app1.conf..
Sat Apr 11 22:23:59 2020 - [info] Reading server configuration from
/etc/mha/app1.conf..
Sat Apr 11 22:23:59 2020 - [info] Starting SSH connection tests..
Sat Apr 11 22:24:00 2020 - [debug]
Sat Apr 11 22:23:59 2020 - [debug]  Connecting via SSH from
admin@10.1.1.10(10.1.1.10:22) to admin@10.1.1.20(10.1.1.20:22)..
Warning: Permanently added '10.1.1.20' (ECDSA) to the list of known hosts.
Sat Apr 11 22:23:59 2020 - [debug]   ok.
Sat Apr 11 22:23:59 2020 - [debug]  Connecting via SSH from
admin@10.1.1.10(10.1.1.10:22) to admin@10.1.1.30(10.1.1.30:22)..
Warning: Permanently added '10.1.1.30' (ECDSA) to the list of known hosts.
Sat Apr 11 22:23:59 2020 - [debug]   ok.
Sat Apr 11 22:24:00 2020 - [debug]
Sat Apr 11 22:23:59 2020 - [debug]  Connecting via SSH from
admin@10.1.1.20(10.1.1.20:22) to admin@10.1.1.10(10.1.1.10:22)..
Sat Apr 11 22:24:00 2020 - [debug]   ok.
Sat Apr 11 22:24:00 2020 - [debug]  Connecting via SSH from
admin@10.1.1.20(10.1.1.20:22) to admin@10.1.1.30(10.1.1.30:22)..
Warning: Permanently added '10.1.1.30' (ECDSA) to the list of known hosts.
Sat Apr 11 22:24:00 2020 - [debug]   ok.
Sat Apr 11 22:24:01 2020 - [debug]
Sat Apr 11 22:24:00 2020 - [debug]  Connecting via SSH from
admin@10.1.1.30(10.1.1.30:22) to admin@10.1.1.10(10.1.1.10:22)..
Sat Apr 11 22:24:00 2020 - [debug]   ok.
Sat Apr 11 22:24:00 2020 - [debug]  Connecting via SSH from
admin@10.1.1.30(10.1.1.30:22) to admin@10.1.1.20(10.1.1.20:22)..
Sat Apr 11 22:24:00 2020 - [debug]   ok.
Sat Apr 11 22:24:01 2020 - [info] All SSH connection tests passed successfully.
[admin@mgr ~]$

以上信息说明ok
```

- 检查集群状态

```
[admin@mgr ~]$ masterha_check_repl --conf=/etc/mha/app1.conf
Sat Apr 11 22:35:44 2020 - [warning] Global configuration file
/etc/masterha_default.cnf not found. Skipping.
Sat Apr 11 22:35:44 2020 - [info] Reading application default configuration from
/etc/mha/app1.conf..
Sat Apr 11 22:35:44 2020 - [info] Reading server configuration from
/etc/mha/app1.conf..
Sat Apr 11 22:35:44 2020 - [info] MHA::MasterMonitor version 0.57.
Sat Apr 11 22:35:45 2020 - [info] GTID failover mode = 1
Sat Apr 11 22:35:45 2020 - [info] Dead Servers:
Sat Apr 11 22:35:45 2020 - [info] Alive Servers:
Sat Apr 11 22:35:45 2020 - [info]   10.1.1.10(10.1.1.10:3307)
Sat Apr 11 22:35:45 2020 - [info]   10.1.1.20(10.1.1.20:3307)
Sat Apr 11 22:35:45 2020 - [info]   10.1.1.30(10.1.1.30:3307)
Sat Apr 11 22:35:45 2020 - [info] Alive Slaves:
Sat Apr 11 22:35:45 2020 - [info]   10.1.1.20(10.1.1.20:3307)  Version=5.7.25-
log (oldest major version between slaves) log-bin:enabled
Sat Apr 11 22:35:45 2020 - [info]     GTID ON
Sat Apr 11 22:35:45 2020 - [info]     Replicating from 10.1.1.10(10.1.1.10:3307)
Sat Apr 11 22:35:45 2020 - [info]     Primary candidate for the new Master
(candidate_master is set)
Sat Apr 11 22:35:45 2020 - [info]   10.1.1.30(10.1.1.30:3307)  Version=5.7.25-
log (oldest major version between slaves) log-bin:enabled
Sat Apr 11 22:35:45 2020 - [info]     GTID ON
Sat Apr 11 22:35:45 2020 - [info]     Replicating from 10.1.1.10(10.1.1.10:3307)
Sat Apr 11 22:35:45 2020 - [info]     Primary candidate for the new Master
(candidate_master is set)
Sat Apr 11 22:35:45 2020 - [info] Current Alive Master:
10.1.1.10(10.1.1.10:3307)
Sat Apr 11 22:35:45 2020 - [info] Checking slave configurations..
Sat Apr 11 22:35:45 2020 - [info] Checking replication filtering settings..
Sat Apr 11 22:35:45 2020 - [info]  binlog_do_db= , binlog_ignore_db=
Sat Apr 11 22:35:45 2020 - [info]  Replication filtering check ok.
Sat Apr 11 22:35:45 2020 - [info] GTID (with auto-pos) is supported. Skipping
all SSH and Node package checking.
Sat Apr 11 22:35:45 2020 - [info] Checking SSH publickey authentication settings
on the current master..
Sat Apr 11 22:35:45 2020 - [info] HealthCheck: SSH to 10.1.1.10 is reachable.
Sat Apr 11 22:35:45 2020 - [info]
10.1.1.10(10.1.1.10:3307) (current master)
 +--10.1.1.20(10.1.1.20:3307)
 +--10.1.1.30(10.1.1.30:3307)

Sat Apr 11 22:35:45 2020 - [info] Checking replication health on 10.1.1.20..
Sat Apr 11 22:35:45 2020 - [info]  ok.
Sat Apr 11 22:35:45 2020 - [info] Checking replication health on 10.1.1.30..
Sat Apr 11 22:35:45 2020 - [info]  ok.
Sat Apr 11 22:35:45 2020 - [info] Checking master_ip_failover_script status:
Sat Apr 11 22:35:45 2020 - [info]   /etc/mha/master_ip_failover.sh 10.1.1.100 1
--command=status --ssh_user=admin --orig_master_host=10.1.1.10 --
orig_master_ip=10.1.1.10 --orig_master_port=3307
Checking the Status of the script.. OK
Sat Apr 11 22:35:45 2020 - [info]  OK.
Sat Apr 11 22:35:45 2020 - [warning] shutdown_script is not defined.
Sat Apr 11 22:35:45 2020 - [info] Got exit code 0 (Not master dead).

MySQL Replication Health is OK.
```

```
以上信息说明ok
```

###7、检查MHA-Mgr状态

```
[admin@mgr ~]$ masterha_check_status --conf=/etc/mha/app1.conf
app1 is stopped(2:NOT_RUNNING).
开启MHA Manager监控：
[admin@mgr ~]$ nohup masterha_manager --conf=/etc/mha/app1.conf --
remove_dead_master_conf --ignore_last_failover &
再次查看监控状态：
[admin@mgr ~]$ masterha_check_status --conf=/etc/mha/app1.conf
app1 (pid:8913) is running(0:PING_OK), master:10.1.1.10


注意：
1. 如果正常，会显示"PING_OK "，否则会显示"NOT_RUNNING "，说明 MHA监控没有开启
2. 使用admin用户启动监控，否则会报权限拒绝
3. 手动停止监控命令:masterha_stop --conf=/etc/mha/app1.conf
```

二、自动Failover测试

###1、安装测试工具

```
[root@master ~]# yum -y install sysbench
```

###2、 创建测试数据

```
master服务器上创建测试库test
mysql> create database test charset utf8mb4;
Query OK, 1 row affected (0.17 sec)

mysql> grant all on *.* to 'mha'@'localhost' identified by '123';
Query OK, 0 rows affected (0.14 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.11 sec)

mysql> exit
Bye


[root@master ~]# sysbench /usr/share/sysbench/oltp_read_only.lua \
 --mysql-host=10.1.1.10  --mysql-port=3307  --mysql-user=mha \
 --mysql-password=123  --mysql-socket=/usr/local/mysql/mysql.sock \
 --mysql-db=test  --db-driver=mysql  --tables=1 \
 --table-size=100000 --report-interval=10 --threads=128 --time=120 prepare

mysql> select count(*) from sbtest1;
+----------+
| count(*) |
+----------+
|   100000 |
+----------+
1 row in set (0.01 sec)
```

###3、模拟故障

```
[root@master ~]# service mysql stop
Shutting down MySQL.............. [ OK ]
```

###4、查看切换过程

```
[root@mgr ~]# tail -f /data/mha/masterha/app1/app1-3307.log
```

##三、经验值分享

###1、管理节点配置文件错误

```
[root@mgr ~]# cat /etc/mha/app1.conf
[server default]
# 设置监控用户和密码,该用户是master上创建的数据库管理账号，拥有所有权限
user=mha
password=123
# 设置复制环境中的复制用户和密码，注意需要有以下权限：
#REPLICATION SLAVE和REPLICATION CLIENT
repl_user=slave
repl_password=123
# 设置ssh的登录用户名
ssh_user=admin
....
[server1]
hostname=10.1.1.10
port= 3307
candidate_master=1
[server2]
hostname=10.1.1.20
port= 3306
candidate_master=1
[server3]
hostname=10.1.1.30
port= 3306
candidate_master=1
注意：一定要配置正确的IP和端口号
```

2、配置MHA时数据只读设置

解决办法：设置从服务器为只读

```
mysql> set @@global.read_only=1;
Query OK, 0 rows affected (0.00 sec)

mysql> show variables like 'read_only';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| read_only     | ON    |
+---------------+-------+
1 row in set (0.00 sec)
```

3、复制用户权限密码错误

原因：

1. 复制用户slave没有相关权限，REPLICATION SLAVE和==REPLICATION CLIENT==
2. 从服务器没有创建复制用户

###4、其他错误

MHA集群至少需要2个slave，所以如果只有一台slave的话，检查也是通不过的！