MYSQL高可用集群架构-MHA架构

**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能在最大程度上保证数据的一致性，以达到真正意义上的高可用。

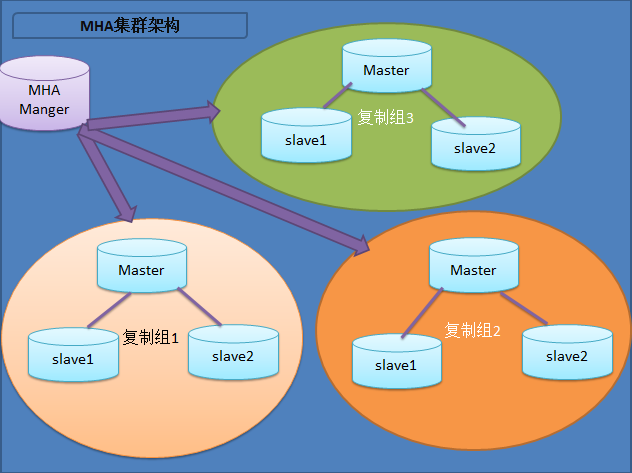
该软件由两部分组成：MHA Manager（管理节点）和MHA Node（数据节点）。MHA Manager可以单独部署在一台独立的机器上管理多个master-slave集群，也可以部署在一台slave节点上。MHA Node运行在每台MySQL服务器上，MHA Manager会定时探测集群中的master节点，当master出现故障时，它可以自动将最新数据的slave提升为新的master，然后将所有其他的slave重新指向新的master。整个故障转移过程对应用程序完全透明。

在MHA自动故障切换过程中，MHA试图从宕机的主服务器上保存二进制日志，最大程度的保证数据的不丢失，但这并不总是可行的。例如，如果主服务器硬件故障或无法通过ssh访问，MHA没法保存二进制日志，只进行故障转移而丢失了最新的数据。使用MySQL 5.5的半同步复制，可以大大降低数据丢失的风险。MHA可以与半同步复制结合起来。如果只有一个slave已经收到了最新的二进制日志，MHA可以将最新的二进制日志应用于其他所有的slave服务器上，因此可以保证所有节点的数据一致性。

目前MHA主要支持一主多从的架构，要搭建MHA,要求一个复制集群中必须最少有三台数据库服务器，一主二从，即一台充当master，一台充当备用master，另外一台充当从库，因为至少需要三台服务器，出于机器成本的考虑，淘宝也在该基础上进行了改造，目前淘宝TMHA已经支持一主一从。MHA 适合任何存储引擎, 只要能主从复制的存储引擎它都支持，不限于支持事物的 innodb 引擎。

官方介绍：https://code.google.com/p/mysql-master-ha/

图01展示了如何通过MHA Manager管理多组主从复制。可以将MHA工作原理总结为如下：



（ 图01 ）

（1）从宕机崩溃的master保存二进制日志事件（binlog events）;

（2）识别含有最新更新的slave；

（3）应用差异的中继日志（relay log）到其他的slave；

（4）应用从master保存的二进制日志事件（binlog events）；

（5）提升一个slave为新的master；

（6）使其他的slave连接新的master进行复制；

MHA软件由两部分组成，Manager工具包和Node工具包，具体的说明如下。

Manager工具包主要包括以下几个工具：

masterha\_check\_ssh 检查MHA的SSH配置状况

masterha\_check\_repl 检查MySQL复制状况

masterha\_manger 启动MHA

masterha\_check\_status 检测当前MHA运行状态

masterha\_master\_monitor 检测master是否宕机

masterha\_master\_switch 控制故障转移（自动或者手动）

masterha\_conf\_host 添加或删除配置的server信息

Node工具包（这些工具通常由MHA Manager的脚本触发，无需人为操作）主要包括以下几个工具：

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将其差异的事件应用于其他的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：

为了尽可能的减少主库硬件损坏宕机造成的数据丢失，因此在配置MHA的同时建议配置成MySQL 5.5的半同步复制。关于半同步复制原理各位自己进行查阅。（不是必须）

1.部署MHA

接下来部署MHA，具体的搭建环境如下（所有操作系统均为CentOS7.4 64bit）：

|  |  |  |  |
| --- | --- | --- | --- |
| IP地址 | 主机名 | 角色 | 软件 |
| 10.10.10.67 | **xuegod67.cn** | **manager** | **mha4mysql-manager、mha4mysql-node** |
| 10.10.10.68 | **xuegod68.cn** | **master** | **mha4mysql-node** |
| 10.10.10.69 | **xuegod69.cn** | **Slave1,Candicate master** | **mha4mysql-node** |
| 10.10.10.70 | **xuegod70.cn** | **Slave2** | **mha4mysql-node** |

其中master对外提供写服务，备选Candicate master（实际为slave1）提供读服务，slave2也提供读服务，一旦master宕机，将会把备选master提升为新的master，slave指向新的master

（1）在所有节点安装MHA node所需的perl模块（DBD:mysql），安装脚本如下：

**先要安装epel源，**

**Centos6安装源：rpm -ivh** [**http://dl.fedoraproject.org/pub/epel/6/x86\_64/epel-release-6-8.noarch.rpm**](http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm)

**Centos7安装源：rpm –ivh http://dl.fedoraproject.org/pub/epel/7/x86\_64/e/epel-release-7-10.noarch.rpm**

以下设置为CentOS6操作

[epel]

name=Extra Packages for Enterprise Linux 6 - $basearch

baseurl=http://download.fedoraproject.org/pub/epel/6/$basearch #将注释的#去掉

#mirrorlist=https://mirrors.fedoraproject.org/metalink?repo=epel-6&arch=$basearch#前面加上#

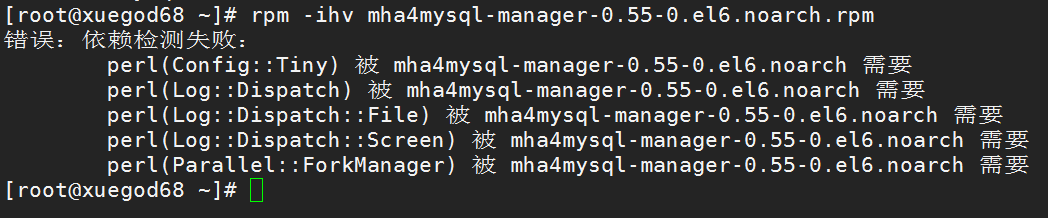
yum clean all

yum list

使用yum安装

全部依赖

yum -y install perl-DBD-MySQL perl-Config-Tiny perl-Log-Dispatch perl-Parallel-ForkManager --skip-broken



（2）上传MHA相关包，在所有的节点安装mha-node：

rpm -ivh mha4mysql-node-0.54-0.el6.noarch.rpm

安装完成后会在/usr/bin/目录下生成以下脚本文件：

[root@xuegod67 bin]# pwd

/usr/bin/

[root@xuegod67 bin]# ll app\* filter\* purge\* save\*

-r-xr-xr-x 1 root root 15498 Apr 20 10:05 apply\_diff\_relay\_logs

-r-xr-xr-x 1 root root 4807 Apr 20 10:05 filter\_mysqlbinlog

-r-xr-xr-x 1 root root 7401 Apr 20 10:05 purge\_relay\_logs

-r-xr-xr-x 1 root root 7263 Apr 20 10:05 save\_binary\_logs

**2.安装MHA Manager**

MHA Manager中主要包括了几个管理员的命令行工具，例如master\_manger，master\_master\_switch等。MHA Manger也依赖于perl模块，具体如下：

（1）安装MHA Node软件包之前需要安装依赖。我这里使用yum完成，首先epel源要安装。注意：刚才我们已经配置epel源。

（2）安装MHA Manager。首先安装MHA Manger依赖的perl模块（我这里使用yum安装）：

yum install perl-DBD-MySQL perl-Config-Tiny perl-Log-Dispatch perl-Parallel-ForkManager perl-Time-HiRes perl-ExtUtils-CBuilder perl-ExtUtils-MakeMaker perl-CPAN -y

安装MHA Manager软件包：

[root@xuegod67 ~]# rpm -ivh mha4mysql-manager-0.55-0.el6.noarch.rpm

安装完成后会在/usr/bin目录下面生成以下脚本文件，前面已经说过这些脚本的作用，这里不再重复

[root@xuegod67 ~]# ll /usr/bin/mast\*

-rwxr-xr-x 1 root root 1995 12月 13 2012 /usr/bin/masterha\_check\_repl

-rwxr-xr-x 1 root root 1779 12月 13 2012 /usr/bin/masterha\_check\_ssh

-rwxr-xr-x 1 root root 1865 12月 13 2012 /usr/bin/masterha\_check\_status

-rwxr-xr-x 1 root root 3201 12月 13 2012 /usr/bin/masterha\_conf\_host

-rwxr-xr-x 1 root root 2517 12月 13 2012 /usr/bin/masterha\_manager

-rwxr-xr-x 1 root root 2165 12月 13 2012 /usr/bin/masterha\_master\_monitor

-rwxr-xr-x 1 root root 2373 12月 13 2012 /usr/bin/masterha\_master\_switch

-rwxr-xr-x 1 root root 3879 12月 13 2012 /usr/bin/masterha\_secondary\_check

-rwxr-xr-x 1 root root 1739 12月 13 2012 /usr/bin/masterha\_stop

3.配置所有主机相互SSH登录无密码验证（使用key登录，工作中常用）。但是有一点需要注意：不能禁止 password 登陆，否则会出现错误

**ssh免密码登录**

**[root@xuegod67 ~]# ssh-keygen -t rsa**

**Generating public/private rsa key pair.**

**Enter file in which to save the key (/root/.ssh/id\_rsa): 回车**

**Enter passphrase (empty for no passphrase): 回车**

**Enter same passphrase again: 回车**

**Your identification has been saved in /root/.ssh/id\_rsa.**

**Your public key has been saved in /root/.ssh/id\_rsa.pub.**

**The key fingerprint is:**

**e1:9f:7e:15:f1:de:07:d3:33:03:cc:3d:36:0c:96:26 root@xuegod67.cn**

**The key's randomart image is:**

**+--[ RSA 2048]----+**

**| oo= |**

**| E.\*.\* |**

**| . o o+o|**

**| . . ++o|**

**| S ==|**

**| . . . +|**

**| o . .|**

**| . . |**

**| .. |**

**+-----------------+**

**[root@xuegod67 ~]# ssh-copy-id 10.10.10.68**

**[root@xuegod67 ~]# ssh-copy-id 10.10.10.69**

**[root@xuegod67 ~]# ssh-copy-id 10.10.10.70**

**其他主机重复上面操作。**

**4.搭建主从复制环境**

注意：binlog-do-db 和 replicate-ignore-db 设置必须相同。 MHA 在启动时候会检测过滤规则，如果过滤规则不同，MHA 不启动监控和故障转移。

（**1）在xuegod68配置主数据库服务器**

创建需要同步的数据库：

mysql> create database HA;

mysql> use HA;

mysql> create table test(id int,name varchar(20));

service mysqld stop

配置my.cnf:

vim /etc/my.cnf

log-bin=mysql-bin-master #启用二进制日志

server-id=1 #本机数据库ID 标示

binlog-do-db=HA #可以被从服务器复制的库。二进制需要同步的数据库名

binlog-ignore-db=mysql #不可以被从服务器复制的库

validate-password=off

重启mysql:

systemctl restart mysqld

授权：

mysql> grant replication slave on \*.\* to repl@'10.10.10.%' identified by '123456';

mysql> flush privileges;

查看状态信息：

mysql> show master status;

**导出数据库到从服务器(2个从)**

**mysqldump -uroot -p123456 –B HA >HA.sql**

**scp HA.sql 10.10.10.65:/root**

**scp HA.sql 10.10.10.66:/root**

**（2）在xuegod69导入数据库并配置从服务：**

[root@xuegod69 ~]# mysql -uroot -p123456 <HA.sql

配置my.cnf:

vim /etc/my.cnf

log-bin=mysql-slave1 #启用二进制日志

server-id=2 #本机数据库ID 标示

binlog-do-db=HA #可以被从服务器复制的库。二进制需要同步的数据库名

binlog-ignore-db=mysql #不可以被从服务器复制的库

log\_slave\_updates=1 #只有开启log\_slave\_updates，从库binlog才会记录主库同步的操作日志

validate-password=off

[root@xuegod69 ~]# systemctl restart mysqld 重启mysql

mysql> grant replication slave on \*.\* to 'repl'@'10.10.10.%' identified by '123456';

mysql> flush privileges;

建立主从关系

mysql> stop slave;

mysql> change master to master\_host='10.10.10.68',master\_user='repl',master\_password='123456';

**（3）在xuegod70导入数据库并配置从服务：**

[root@xuegod70 ~]# mysql -uroot -p123456 <HA.sql

配置my.cnf:

vim /etc/my.cnf

log-bin=mysql-slave2 #启用二进制日志

server-id=3 #本机数据库ID 标示

binlog-do-db=HA #可以被从服务器复制的库。二进制需要同步的数据库名

binlog-ignore-db=mysql #不可以被从服务器复制的库

log\_slave\_updates=1 #只有开启log\_slave\_updates，从库binlog才会记录主库同步的操作日志

validate-password=off

[root@xuegod70 ~]# systemctl restart mysqld 重启mysql

mysql> grant replication slave on \*.\* to 'repl'@'10.10.10.%' identified by '123456';

mysql> flush privileges;

建立主从关系

mysql> stop slave;

mysql> change master to master\_host='10.10.10.68',master\_user='repl',master\_password='123456';

**（4）两台slave服务器设置read\_only（从库对外提供读服务，只所以没有写进配置文件，是因为slave随时会提升为master）**

[root@xuegod69~]# mysql -uroot -p123456 -e 'set global read\_only=1'

[root@xuegod70 ~]# mysql -uroot -p123456 -e 'set global read\_only=1'

**（5）创建监控用户（在主从上都执行）：**

mysql> grant all privileges on \*.\* to 'root'@'10.10.10.%' identified by '123456';

mysql> flush privileges;

到这里整个集群环境已经搭建完毕，剩下的就是配置MHA软件了。

**5.配置MHA**

（**1）创建MHA的工作目录，并且创建相关配置文件（在软件包解压后的目录里面有样例配置文件）。**

[root@ xuegod67 ~]# mkdir -p /etc/masterha

[root@ xuegod67 ~]# mkdir -p /var/log/masterha/app1

[root@ xuegod67 ~]# vim /etc/masterha/app1.cnf

修改app1.cnf配置文件，修改后的文件内容如下（注意，配置文件中的注释需要去掉，我这里是为了解释清楚）：

[server default]

manager\_workdir=/var/log/masterha/app1 //设置manager的工作目录

manager\_log=/var/log/masterha/app1/manager.log //设置manager的日志

master\_binlog\_dir=/data/mysql //设置master 保存binlog的位置，以便MHA可以找到master的日志，我这里的也就是mysql的数据目录

master\_ip\_failover\_script= /usr/local/bin/master\_ip\_failover //设置自动failover时候的切换脚本

master\_ip\_online\_change\_script= /usr/local/bin/master\_ip\_online\_change //设置手动切换时候的切换脚本

password=123456 //设置mysql中root用户的密码，这个密码是前文中创建监控用户的那个密码

user=root 设置监控用户root

ping\_interval=1 //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railover

remote\_workdir=/tmp //设置远端mysql在发生切换时binlog的保存位置

repl\_password=123456 //设置复制用户的密码

repl\_user=repl //设置复制环境中的复制用户名

report\_script=/usr/local/send\_report //设置发生切换后发送的报警的脚本

shutdown\_script="" //设置故障发生后关闭故障主机脚本（该脚本的主要作用是关闭主机放在发生脑裂,这里没有使用）

ssh\_user=root //设置ssh的登录用户名

[server1]

hostname=10.10.10.68

port=3306

[server2]

hostname=10.10.10.69

port=3306

candidate\_master=1 #设置为候选master，如果设置该参数以后，发生主从切换以后将会将此从库提升为主库，即使这个主库不是集群中事件最新的slave

check\_repl\_delay=0 #默认情况下如果一个slave落后master 100M的relay logs的话，MHA将不会选择该slave作为一个新的master，因为对于这个slave的恢复需要花费很长时间，通过设置check\_repl\_delay=0,MHA触发切换在选择一个新的master的时候将会忽略复制延时，这个参数对于设置了candidate\_master=1的主机非常有用，因为这个候选主在切换的过程中一定是新的master

[server3]

hostname=10.10.10.70

port=3306

**（2）设置relay log的清除方式（在每个slave节点上）：**

[root@xuegod69 ~]# mysql -uroot -p123456 -e 'set global relay\_log\_purge=0'

[root@xuegod70 ~]# mysql -uroot -p123456 -e 'set global relay\_log\_purge=0'

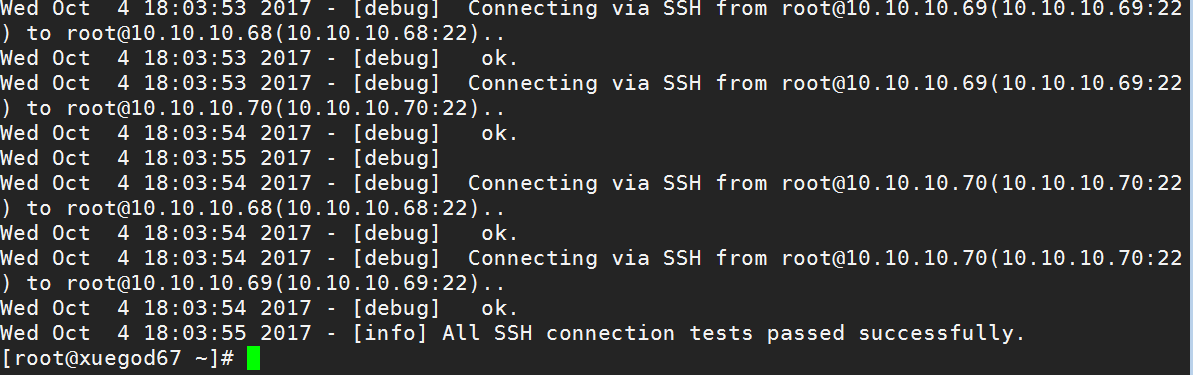
注意：

MHA在发生切换的过程中，从库的恢复过程中依赖于relay log的相关信息，所以这里要将relay log的自动清除设置为OFF，采用手动清除relay log的方式。在默认情况下，从服务器上的中继日志会在SQL线程执行完毕后被自动删除。但是在MHA环境中，这些中继日志在恢复其他从服务器时可能会被用到，因此需要禁用中继日志的自动删除功能。定期清除中继日志需要考虑到复制延时的问题。在ext3的文件系统下，删除大的文件需要一定的时间，会导致严重的复制延时。为了避免复制延时，需要暂时为中继日志创建硬链接，因为在Linux系统中通过硬链接删除大文件速度会很快。（在mysql数据库中，删除大表时，通常也采用建立硬链接的方式）

**（3）.检查SSH配置**

检查MHA Manger到所有MHA Node的SSH连接状态：

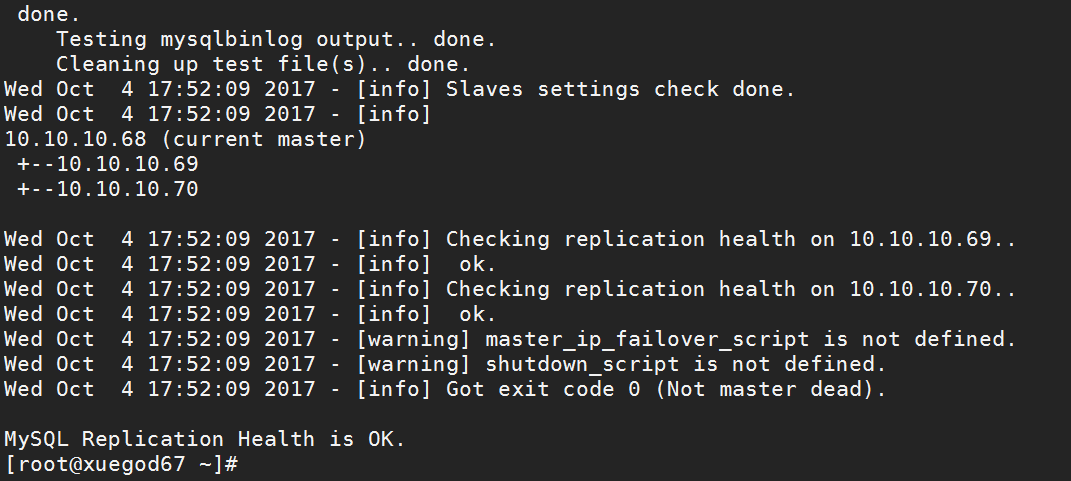
[root@xuegod67~]# masterha\_check\_ssh --conf=/etc/masterha/app1.cnf

可以看见各个节点ssh验证都是ok的。

**（4）.检查整个复制环境状况。**

通过masterha\_check\_repl脚本查看整个集群的状态

[root@ xuegod67 ~]# masterha\_check\_repl --conf=/etc/masterha/app1.cnf



**MySQL Replication Health is NOT OK! 如果提示这个不ok，说明有问题**

**MySQL Replication Health is OK. 显示Ok ，正常！**

**（5）.检查MHA Manager的状态：**

通过master\_check\_status脚本查看Manager的状态：

[root@xuegod67 ~]# masterha\_check\_status --conf=/etc/masterha/app1.cnf

app1 is stopped(2:NOT\_RUNNING).

注意：如果正常，会显示"PING\_OK"，否则会显示"NOT\_RUNNING"，这代表MHA监控没有开启。

**（6）.开启MHA Manager监控**

[root@ xuegod67 ~]# nohup masterha\_manager --conf=/etc/masterha/app1.cnf --remove\_dead\_master\_conf --ignore\_last\_failover < /dev/null > /var/log/masterha/app1/manager.log 2>&1 &

[1] 30867

启动参数介绍：

--remove\_dead\_master\_conf 该参数代表当发生主从切换后，老的主库的ip将会从配置文件中移除。

--manger\_log 日志存放位置

--ignore\_last\_failover 在缺省情况下，如果MHA检测到连续发生宕机，且两次宕机间隔不足8小时的话，则不会进行Failover，之所以这样限制是为了避免ping-pong效应。该参数代表忽略上次MHA触发切换产生的文件，默认情况下，MHA发生切换后会在日志目录，也就是上面我设置的/data产生app1.failover.complete文件，下次再次切换的时候如果发现该目录下存在该文件将不允许触发切换，除非在第一次切换后收到删除该文件，为了方便，这里设置为--ignore\_last\_failover。

查看MHA Manager监控是否正常：

[root@ xuegod67 ~]# masterha\_check\_status --conf=/etc/masterha/app1.cnf

app1 (pid:20386) is running(0:PING\_OK), master:10.10.10.64

可以看见已经在监控了，而且master的主机为10.10.10.64

**（7）**.查看启动日志

复制代码

[root@xuegod67 ~]# tail -n20 /var/log/masterha/app1/manager.log

………………..

…………………

Sun Apr 20 19:12:01 2014 - [info] Ping(SELECT) succeeded, waiting until MySQL doesn't respond..

其中"Ping(SELECT) succeeded, waiting until MySQL doesn't respond.."说明整个系统已经开始监控了。

**（8）.关闭MHA Manage监控**

关闭很简单，使用masterha\_stop命令完成。

[root@xuegod67~]# masterha\_stop --conf=/etc/masterha/app1.conf

Stopped app1 successfully.

[1]+ Exit 1 nohup masterha\_manager --conf=/etc/masterha/app1.cnf --remove\_dead\_master\_conf --ignore\_last\_failover --manager\_log=/data/mamanager.log

**（9）模拟故障**

[root@xuegod67 ~]# tail -f /var/log/masterha/app1/manager.log 打开新窗口观察日志

[root@xuegod68~]# service mysqld stop 模拟主库挂掉

看日志是否切换master成功

From:

10.10.10.68 (current master)

+--10.10.10.69

+--10.10.10.70

To:

10.10.10.69 (new master)

+--10.10.10.70

Mon Oct 9 22:02:36 2017 - [info]

Mon Oct 9 22:02:36 2017 - [info] \* Phase 3.3: New Master Diff Log Generation Phase..

Mon Oct 9 22:02:36 2017 - [info]

Mon Oct 9 22:02:36 2017 - [info] This server has all relay logs. No need to generate diff files from the latest slave.

Mon Oct 9 22:02:36 2017 - [info]

Mon Oct 9 22:02:36 2017 - [info] \* Phase 3.4: Master Log Apply Phase..

Mon Oct 9 22:02:36 2017 - [info]

Mon Oct 9 22:02:36 2017 - [info] \*NOTICE: If any error happens from this phase, manual recovery is needed.

Mon Oct 9 22:02:36 2017 - [info] Starting recovery on 10.10.10.69(10.10.10.69:3306)..

Mon Oct 9 22:02:36 2017 - [info] This server has all relay logs. Waiting all logs to be applied..

Mon Oct 9 22:02:36 2017 - [info] done.

Mon Oct 9 22:02:36 2017 - [info] All relay logs were successfully applied.

Mon Oct 9 22:02:36 2017 - [info] Getting new master's binlog name and position..

Mon Oct 9 22:02:36 2017 - [info] mysql-slave1.000001:598

Mon Oct 9 22:02:36 2017 - [info] All other slaves should start replication from here. Statement should be: CHANGE MASTER TO MASTER\_HOST='10.10.10.69', MASTER\_PORT=3306, MASTER\_LOG\_FILE='mysql-slave1.000001', MASTER\_LOG\_POS=598, MASTER\_USER='repl', MASTER\_PASSWORD='xxx';

Mon Oct 9 22:02:36 2017 - [warning] master\_ip\_failover\_script is not set. Skipping taking over new master ip address.

Mon Oct 9 22:02:36 2017 - [info] Setting read\_only=0 on 10.10.10.69(10.10.10.69:3306)..

Mon Oct 9 22:02:36 2017 - [info] ok.

Mon Oct 9 22:02:36 2017 - [info] \*\* Finished master recovery successfully.

Mon Oct 9 22:02:36 2017 - [info] \* Phase 3: Master Recovery Phase completed.

Mon Oct 9 22:02:36 2017 - [info]

Mon Oct 9 22:02:36 2017 - [info] \* Phase 4: Slaves Recovery Phase..

Mon Oct 9 22:02:36 2017 - [info]

Mon Oct 9 22:02:36 2017 - [info] \* Phase 4.1: Starting Parallel Slave Diff Log Generation Phase..

Mon Oct 9 22:02:36 2017 - [info]

Mon Oct 9 22:02:36 2017 - [info] -- Slave diff file generation on host 10.10.10.70(10.10.10.70:3306) started, pid: 2923. Check tmp log /etc/masterha/app1/10.10.10.70\_3306\_20171009220233.log if it takes time..

Mon Oct 9 22:02:37 2017 - [info]

Mon Oct 9 22:02:37 2017 - [info] Log messages from 10.10.10.70 ...

Mon Oct 9 22:02:37 2017 - [info]

Mon Oct 9 22:02:36 2017 - [info] This server has all relay logs. No need to generate diff files from the latest slave.

Mon Oct 9 22:02:37 2017 - [info] End of log messages from 10.10.10.70.

Mon Oct 9 22:02:37 2017 - [info] -- 10.10.10.70(10.10.10.70:3306) has the latest relay log events.

Mon Oct 9 22:02:37 2017 - [info] Generating relay diff files from the latest slave succeeded.

Mon Oct 9 22:02:37 2017 - [info]

Mon Oct 9 22:02:37 2017 - [info] \* Phase 4.2: Starting Parallel Slave Log Apply Phase..

Mon Oct 9 22:02:37 2017 - [info]

Mon Oct 9 22:02:37 2017 - [info] -- Slave recovery on host 10.10.10.70(10.10.10.70:3306) started, pid: 2925. Check tmp log /etc/masterha/app1/10.10.10.70\_3306\_20171009220233.log if it takes time..

Mon Oct 9 22:02:38 2017 - [info]

Mon Oct 9 22:02:38 2017 - [info] Log messages from 10.10.10.70 ...

Mon Oct 9 22:02:38 2017 - [info]

Mon Oct 9 22:02:37 2017 - [info] Starting recovery on 10.10.10.70(10.10.10.70:3306)..

Mon Oct 9 22:02:37 2017 - [info] This server has all relay logs. Waiting all logs to be applied..

Mon Oct 9 22:02:37 2017 - [info] done.

Mon Oct 9 22:02:37 2017 - [info] All relay logs were successfully applied.

Mon Oct 9 22:02:37 2017 - [info] Resetting slave 10.10.10.70(10.10.10.70:3306) and starting replication from the new master 10.10.10.69(10.10.10.69:3306)..

Mon Oct 9 22:02:38 2017 - [info] Executed CHANGE MASTER.

Mon Oct 9 22:02:38 2017 - [info] Slave started.

Mon Oct 9 22:02:38 2017 - [info] End of log messages from 10.10.10.70.

Mon Oct 9 22:02:38 2017 - [info] -- Slave recovery on host 10.10.10.70(10.10.10.70:3306) succeeded.

Mon Oct 9 22:02:38 2017 - [info] All new slave servers recovered successfully.

Mon Oct 9 22:02:38 2017 - [info]

Mon Oct 9 22:02:38 2017 - [info] \* Phase 5: New master cleanup phase..

Mon Oct 9 22:02:38 2017 - [info]

Mon Oct 9 22:02:38 2017 - [info] Resetting slave info on the new master..

Mon Oct 9 22:02:38 2017 - [info] 10.10.10.69: Resetting slave info succeeded.

Mon Oct 9 22:02:38 2017 - [info] Master failover to 10.10.10.69(10.10.10.69:3306) completed successfully.

Mon Oct 9 22:02:38 2017 - [info] Deleted server1 entry from /etc/masterha/app1.conf .

Mon Oct 9 22:02:38 2017 - [info]

----- Failover Report -----

app1: MySQL Master failover 10.10.10.68 to 10.10.10.69 succeeded

Master 10.10.10.68 is down!

Check MHA Manager logs at xuegod67.cn:/var/log/masterha/app1/manager.log for details.

Started automated(non-interactive) failover.

The latest slave 10.10.10.69(10.10.10.69:3306) has all relay logs for recovery.

Selected 10.10.10.69 as a new master.

10.10.10.69: OK: Applying all logs succeeded.

10.10.10.70: This host has the latest relay log events.

Generating relay diff files from the latest slave succeeded.

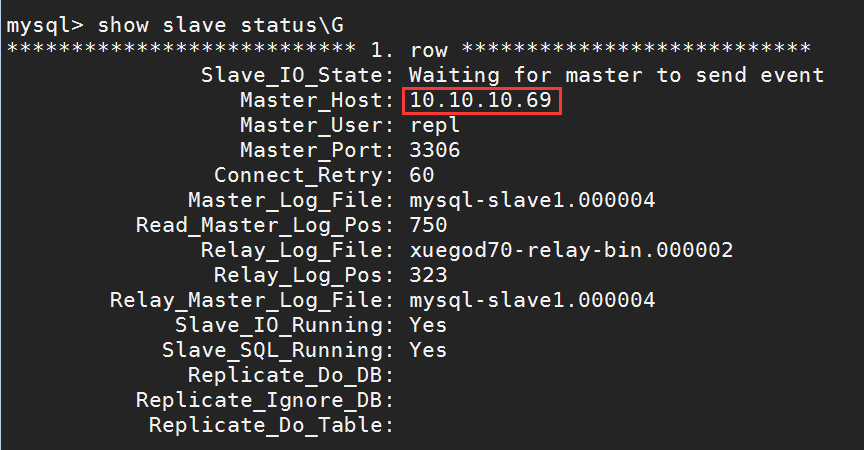
10.10.10.70: OK: Applying all logs succeeded. Slave started, replicating from 10.10.10.69.

10.10.10.69: Resetting slave info succeeded.

Master failover to 10.10.10.69(10.10.10.69:3306) completed successfully.

我们从日志上可以看到故障切换成功，恭喜你了.新的master是xuegod69.

登陆从服务器xuegod70查看show slave status\G是否成功切换



**总结：**

**目前高可用方案可以一定程度上实现数据库的高可用，还有其他方案heartbeat+drbd，Cluster、MGR等。这些高可用软件各有优劣。在进行高可用方案选择时，主要是看业务还有对数据一致性方面的要求。**

………………………………………………….我是MHA分割线……………………………………………………………………

/etc/masterha/app1.cnf 配置文件备份

[server default]

manager\_workdir=/var/log/masterha/app1

manager\_log=/var/log/masterha/app1/manager.log

master\_binlog\_dir=/var/lib/mysql

user=root

password=123456

ping\_interval=1

remote\_workdir=/tmp

repl\_user=repl

repl\_password=123456

ssh\_user=root

[server1]

hostname=10.10.10.68

port=3306

[server2]

hostname=10.10.10.69

port=3306

candidate\_master=1

check\_repl\_delay=0

[server3]

hostname=10.10.10.70

port=3306

**扩展：**

关于配置VIP配合MHA使用

vip配置可以采用两种方式，一种通过keepalived的方式管理虚拟ip的浮动；另外一种通过脚本方式启动虚拟ip的方式（即不需要keepalived或者heartbeat类似的软件）。

关于keepalived方式我这里不讲了，因为为了防止脑裂发生，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。自行脑补百度或者参考http://www.cnblogs.com/gomysql/p/3675429.html

下面是通过脚本的方式管理VIP。这里是修改/usr/local/bin/master\_ip\_failover，修改完成后内容如下，而且如果使用脚本管理vip的话，需要手动在master服务器上绑定一个vip

编写脚本/usr/ /bin/master\_ip\_failover，要会perl脚本语言（主库上操作，xuegod68）。

在MHA Manager修改脚本修改后的内容如下（参考资料比较少）：

#!/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

);

my $vip = '192.168.0.88';

my $ssh\_start\_vip = "/etc/init.d/keepalived start";

my $ssh\_stop\_vip = "/etc/init.d/keepalived stop";

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;

}

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";

#`ssh $ssh\_user\@cluster1 \" $ssh\_start\_vip \"`;

exit 0;

}

else {

&usage();

exit 1;

}

}

# A simple system call that enable the VIP on the new master

sub start\_vip() {

`ssh $ssh\_user\@$new\_master\_host \" $ssh\_start\_vip \"`;

}

# A simple system call that disable the VIP on the old\_master

sub stop\_vip() {

`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";

}