文档名称	使用 mysql-proxy 实现 mysql 读写分离
作者/日期	陆文举　2010-11-14
博客	http://blog.luwenju.com
微博	http://weibo.com/luwenju

由于公司数据库负载较大，所以便打算使用读写分离来减轻 mysql 的负载。目前较为常见的 mysql 读写分离分为两种：

1、 **基于程序代码内部实现**：在代码中根据 select、insert 进行路由分类；这类方法也是目前生产环境应用最广泛的。优点是性能较好，因为在程序代码中实现，不需要增加额外的设备作为硬件开支。缺点是需要开发人员来实现，运维人员无从下手。

2、 **基于中间代理层实现**：我们都知道代理一般是位于客户端和服务器之间，代理服务器接到客户端请求后通过判断然后转发到后端数据库。在这有两个代表性程序

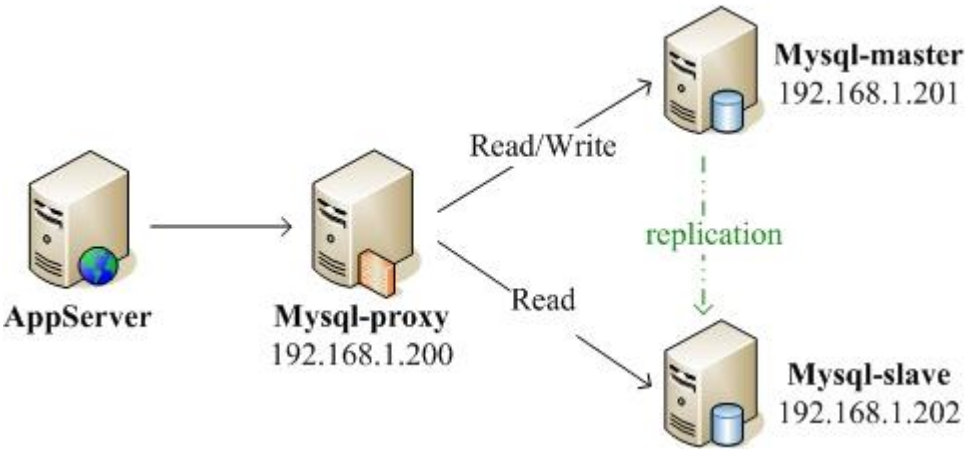
mysql-proxy：mysql-proxy 为 mysql 开源项目，通过其自带的 lua 脚本进行 sql 判断，虽然是 mysql 官方产品，但是 mysql 官方并不建议将 mysql-proxy 用到生产环境。

amoeba：由陈思儒开发，作者曾就职于阿里巴巴，现就职于盛大。该程序由 java 语言进行开发，目前只听说阿里巴巴将其用于生产环境。另外，此项目严重缺少维护和推广（作者有个官方博客，很多用户反馈的问题发现作者不理睬）

经过上述简单的比较，通过程序代码实现 mysql 读写分离自然是一个不错的选择。但是并不是所有的应用都适合在程序代码中实现读写分离，像大型 SNS、B2C 这类应用可以在代码中实现，因为这样对程序代码本身改动较小；像一些大型复杂的 java 应用，这种类型的应用在代码中实现对代码改动就较大了。所以，像这种应用一般就会考虑使用代理层来实现。

下面我们看一下如何搭建 mysql-proxy 来实现 mysql 读写分离

环境拓扑如下：



关于 mysql、mysql 主从的搭建，在此不再演示，如下的操作均在 mysql-proxy（192.168.1.200）服务器进行

一、安装 mysql-proxy

1、安装 lua (mysql-proxy 需要使用 lua 脚本进行数据转发)

#tar zxvf lua-5.1.4.tar.gz
#cd lua-5.1.4
#vi Makefile，修改 INSTALL_TOP= /usr/local/lua
#make posix
#make install

2、安装 libevent

#tar zxvf libevent-2.0.8-rc.tar.gz
#cd libevent-2.0.8-rc
#./configure --prefix=/usr/local/libevent
#make && make install

3、安装 check

#tar zxvf check-0.9.8.tar.gz
#cd check-0.9.8
#./configure && make && make install

4、安装 mysql 客户端

#tar zxvf mysql-5.0.92.tar.gz
#cd mysql-5.0.92
#./configure --without-server && make && make install

5、设置环境变量 （安装 mysql-proxy 所需变量）

#vi /etc/profile
export LUA_CFLAGS="-I/usr/local/lua/include" LUA_LIBS="-L/usr/local/lua/lib -llua -ldl" LDFLAGS="-L/usr/local/libevent/lib -lm"
export CPPFLAGS="-I/usr/local/libevent/include"
export CFLAGS="-I/usr/local/libevent/include"
source /etc/profile

6、安装 mysql-proxy

#tar zxvf mysql-proxy-0.6.0.tar.gz
#cd mysql-proxy-0.6.0
./configure --prefix=/usr/local/mysql-proxy --with-mysql --with-lua
#make && make install

7、启动 mysql-proxy

本次对两台数据库实现了读写分离；mysql-master 为可读可写，mysql-slave 为只读

#/usr/local/mysql-proxy/sbin/mysql-proxy
--proxy-backend-addresses=192.168.1.201:3306
--proxy-read-only-backend-addresses=192.168.1.202:3306
--proxy-lua-script=/usr/local/mysql-proxy/share/mysql-proxy/rw-splitting.lua &

注：如果正常情况下启动后终端不会有任何提示信息，mysql-proxy 启动后会启动两个端口 4040 和 4041，4040 用于 SQL 转发，4041 用于管理 mysql-proxy。如有多个 mysql-slave 可以依次在后面添加

二、测试

1、连接测试

因为默认情况下 mysql 数据库不允许用户在远程连接

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

客户端连接

```
#mysql -uroot -p123456 -h192.168.1.200 -P4040
```

2、读写分离测试

为了测试出 mysql 读写分离的真实性，在测试之前，需要开启两台 mysql 的 log 功能，然后在 mysql-slave 服务器停止复制

①、在两台 mysql 配置文件 my.cnf 中加入 log=query.log，然后重启

②、在 mysql-slave 上执行 SQL 语句 stop slave

③、在两台 mysql 上执行#tail -f /usr/local/mysql/var/query.log

④、在客户端上连接 mysql（三个连接以上），然后执行 create、select 等 SQL 语句，观察两台 mysql 的日志有何变化

注：生产环境中除了进行程序调试外，其它不要开启 mysql 查询日志，因为查询日志记录了客户端的所有语句，频繁的 IO 操作将会导致 mysql 整体性能下降

总结：在上述环境中，mysql-proxy 和 mysql-master、mysql-slave 三台服务器均存在单点故障。如果在可用性要求较高的场合，单点隐患是绝对不允许的。为了避免 mysql-proxy 单点隐患有两种方法，一种方法是 mysql-proxy 配合 keepalived 做双机，另一种方法是将 mysql-proxy 和应用服务安装到同一台服务器上；为了避免 mysql-master 单点故障可以使用 DRBD+heartbear 做双机；避免 mysql-slave 单点故障增加多台 mysql-slave 即可，因为 mysql-proxy 会自动屏蔽后端发生故障的 mysql-slave。

在搭建 mysql-proxy 时遇到不少麻烦，在此再次感谢刘兄和师父的帮助！