利用mysqlbinlog远程备份binlog及利用备份的binlog恢复数据库

参考 ： http://www.cnblogs.com/ivictor/p/5502240.html

以前用脚本对远程服务器进行备份的方式，有个缺点：无法对MySQL服务器当前正在写的二进制日志文件进行备份。所以，只能等到MySQL服务器全部写完才能进行备份。而写完一个binlog的时间并不固定，这就导致备份周期的不确定。
从MySQL5.6开始，mysqlbinlog支持将远程服务器上的binlog实时复制到本地服务器上。
mysqlbinlog的实时二进制复制功能并非简单的将远程服务器的日志复制过来，它是通过MySQL 5.6公布的Replication API实时获取二进制事件。本质上，就相当于MySQL的从服务器。与普通服务器类似，主服务器发生事件后，一般都会在0.5~1秒内进行备份。

```
mysqlbinlog --read-from-remote-server --raw --host=192.168.244.145 --port=3306 --user=repl --password=repl --stop-never  mysql-bin.000001
```

--read-from-remote-server ： 用于备份远程服务器的binlog，如果不指定该选项，则会查找本地的binlog
--raw ： binlog会以二进制格式存储在磁盘中，如果不指定该选项，则会以文本形式保存
--host ： 远程服务器的IP
--user ： 复制的MySQL用户，只需要授予Replication Slave权限
--srop-never ： mysqlbinlog可以只从远程服务器获取指定的几个binlog, 也可将不断生成的binlog保存到本地
mysql-bin.000001 ： 代表从哪个binlog开始复制

除了以上选项外，还有以下几个选项需要注意：
--stop-never-slave-server-id：在备份远程服务器的binlog时，mysqlbinlog本质上就相当于一个从服务器，该选项就是用来指定从服务器的server-id的。默认为-1。
--to-last-log：代表mysqlbinlog不仅能够获取指定的binlog，还能获取其后生成的binlog，获取完了，才终止。如果指定了--stop-never选项则会隐式打开--to-last-log选项。
--result-file：用于设置远程服务器的binlog，保存到本地的前缀。譬如对于mysql-bin.000001，如果指定--result-file=/test/backup-，则保存到本地后的文件名为/test/backup-mysql-bin.000001。注意：如果将--result-file设置为目录，则一定要带上目录分隔符“/”。譬如--result-file=/test/，而不是--result-file=/test，不然保存到本地的文件名为/testmysql-bin.000001。

不足：
这个方式有个问题，对于常规的主从复制来说，如果主从直接的连接断开了，则从会自动再次连接，而对于mysqlbinlog，如果断开了，并不会自动连接。

解决方案：
可通过脚本来弥补上述不足。

```
#!/bin/sh
BACKUP_BIN=/usr/bin/mysqlbinlog
LOCAL_BACKUP_DIR=/backup/binlog/
BACKUP_LOG=/backup/binlog/backuplog
REMOTE_HOST=192.168.244.145
REMOTE_PORT=3306
REMOTE_USER=repl
REMOTE_PASS=repl
FIRST_BINLOG=mysql-bin.000001
#time to wait before reconnecting after failure
SLEEP_SECONDS=10
##create local_backup_dir if necessary
mkdir -p ${LOCAL_BACKUP_DIR}
cd ${LOCAL_BACKUP_DIR}
## 运行while循环，连接断开后等待指定时间，重新连接
while :
do
  if [ `ls -A "${LOCAL_BACKUP_DIR}" |wc -l` -eq 0 ];then
     LAST_FILE=${FIRST_BINLOG}
  else
     LAST_FILE=`ls -l ${LOCAL_BACKUP_DIR} | grep -v backuplog |tail -n 1 |awk '{print $9}'`
  fi
```

```
  ${BACKUP_BIN} --raw --read-from-remote-server --stop-never --host=${REMOTE_HOST} --port=${REMOTE_PORT} --user=${REMOTE_USER} --password=${REMOTE_PASS} ${LAST_FILE}
  echo "`date +"%Y/%m/%d %H:%M:%S"` mysqlbinlog停止，返回代码：$?" | tee -a ${BACKUP_LOG}
  echo "${SLEEP_SECONDS}秒后再次连接并继续备份" | tee -a ${BACKUP_LOG}
  sleep ${SLEEP_SECONDS}
done
```

脚本解读：

1. 实际上定义了一个死循环，如果备份失败，则10s后重新连接。
2. 第一次运行时需指定FIRST_BINLOG的值，指从哪个binlog开始复制，一般为mysql-bin.000001。后续执行的时候就直接获取备份目录下最新的binlog，从最新的binlog开始复制。

总结：

1. 如果指定了--raw，mysqlbinlog获取事件后，并不会实时落盘，而是先保存在本地服务器的内存中，每4K刷盘一次。这也就减少了频繁的日志写操作。如果此时mysqlbinlog和主服务器之间的连接断开了，则内存中的binlog会马上刷新到磁盘中。
2. 尽管mysqlbinlog类似于从服务器，但从服务器上的relaylog却是实时存盘的，即从服务器获取主服务器产生的事件后，会实时写入到relaylog中。
3. 如果不指定--raw，这个时候会以文本格式存盘，此时，--result-file=/test/不能指定为目录，必须明确写上文件名，譬如--result-file=/test/1.sql，此时，mysqlbinlog获取事件后，是实时落盘的，不会每4K刷盘一次。

eg :

```
[docker@lv067 binlog]$
[docker@lv067 binlog]$ mysqlbinlog --read-from-remote-server --raw --host=10.1.5.6 --port=3306 --user=repl --password=Repl@1415 --stop-never  mysql-bin.000001
mysqlbinlog: [Warning] Using a password on the command line interface can be insecure.
```