By Falko Timme

Published: 2007-10-23 18:03

Setting Up Master-Master Replication With MySQL 5 On Debian Etch

Version 1.0
Author: Falko Timme <ft [at] falkotimme [dot] com>
Last edited 10/15/2007

Since version 5, MySQL comes with built-in support for master-master replication, solving the problem that can happen with self-generated keys. In former MySQL versions, the problem with master-master replication was that conflicts arose immediately if node A and node B both inserted an auto-incrementing key on the same table. The advantages of master-master replication over the traditional master-slave replication are that you don't have to modify your applications to make write accesses only to the master, and that it is easier to provide high-availability because if the master fails, you still have the other master.

I do not issue any guarantee that this will work for you!

1 Preliminary Note

In this tutorial I will show how to replicate the database `exampledb` from the server `server1.example.com` with the IP address `192.168.0.100` to the server `server2.example.com` with the IP address `192.168.0.101` and vice versa. Each system is the slave of the other master and the master of the other slave at the same time. Both systems are running Debian Etch; however, the configuration should apply to almost all distributions with little or no modifications.

2 Installing MySQL 5.0

If MySQL 5.0 isn't already installed on `server1` and `server2`, install it now:

server1/server2:

```
apt-get install mysql-server-5.0 mysql-client-5.0
```

To make sure that the replication can work, we must make MySQL listen on all interfaces, therefore we comment out the line `bind-address = 127.0.0.1` in `/etc/mysql/my.cnf`:

server1/server2:

```
vi /etc/mysql/my.cnf
```

```
[...]
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
#bind-address           = 127.0.0.1
[...]
```

Restart MySQL afterwards:

server1/server2:

```
/etc/init.d/mysql restart
```

Then check with

server1/server2:

```
netstat -tap | grep mysql
```

that MySQL is really listening on all interfaces:

```
server1:~# netstat -tap | grep mysql
```

```
tcp        0      0 *:mysql                 *:*                     LISTEN     2671/mysqld
server1:~#
```

Afterwards, set a MySQL password for the user *root@localhost*:

server1/server2:

```
mysqladmin -u root password yourrootsqlpassword
```

Next we create a MySQL password for *root@server1.example.com*:

server1:

```
mysqladmin -h server1.example.com -u root password yourrootsqlpassword
```

Now we set up a replication user *slave2_user* that can be used by *server2* to access the MySQL database on *server1*:

server1:

```
mysql -u root -p
```

On the MySQL shell, run the following commands:

server1:

```
GRANT REPLICATION SLAVE ON *.* TO 'slave2_user'@'%' IDENTIFIED BY 'slave2_password';

FLUSH PRIVILEGES;

quit;
```

Now we do the last two steps again on `server2`:

server2:

```
mysqladmin -h server2.example.com -u root password yourrootsqlpassword
```

```
mysql -u root -p
```

```
GRANT REPLICATION SLAVE ON *.* TO 'slave1_user'@'%' IDENTIFIED BY 'slave1_password';

FLUSH PRIVILEGES;

quit;
```

3 Some Notes

In the following I will assume that the database `exampledb` is **already existing** on `server1`, and that there are tables with records in it. We will set up replication of `exampledb` to `server2`, and afterwards we set up replication of `exampledb` from `server2` to `server1`.

Before we start setting up the replication, we create an **empty** database `exampledb` on `server2`:

server2:

```
mysql -u root -p
```

```
CREATE DATABASE exampledb;

quit;
```

4 Setting Up Replication

Now we set up master-master replication in `/etc/mysql/my.cnf`. The crucial configuration options for master-master replication are `auto_increment_increment` and `auto_increment_offset`:

- `auto_increment_increment` controls the increment between successive AUTO_INCREMENT values.
- `auto_increment_offset` determines the starting point for AUTO_INCREMENT column values.

Let's assume we have N MySQL nodes (N=2 in this example), then `auto_increment_increment` has the value N on all nodes, and each node must have a different value for `auto_increment_offset` (1, 2, ..., N).

Now let's configure our two MySQL nodes:

server1:

```
vi /etc/mysql/my.cnf
```

Search for the section that starts with `[mysqld]`, and put the following options into it (commenting out all existing **conflicting** options):

```
[...]
[mysqld]
server-id = 1
replicate-same-server-id = 0
auto-increment-increment = 2
auto-increment-offset = 1

master-host = 192.168.0.101
master-user = slave1_user
master-password = slave1_password
master-connect-retry = 60
replicate-do-db = exampledb
```

```
log-bin = /var/log/mysql/mysql-bin.log
binlog-do-db = exampledb

relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index

expire_logs_days        = 10
max_binlog_size         = 500M
[...]
```

Then restart MySQL:

server1:

```
/etc/init.d/mysql restart
```

Now do the same on `server2`:

server2:

```
vi /etc/mysql/my.cnf
```

```
[...]
server-id = 2
replicate-same-server-id = 0
auto-increment-increment = 2
auto-increment-offset = 2
```

```
master-host = 192.168.0.100
master-user = slave2_user
master-password = slave2_password
master-connect-retry = 60
replicate-do-db = exampledb

log-bin= /var/log/mysql/mysql-bin.log
binlog-do-db = exampledb

relay-log = /var/lib/mysql/slave-relay.log
relay-log-index = /var/lib/mysql/slave-relay-log.index

expire_logs_days        = 10
max_binlog_size         = 500M
[...]
```

server2:

```
/etc/init.d/mysql restart
```

Next we lock the *exampledb* database on *server1*, find out about the master status of *server1*, create an SQL dump of *exampledb* (that we will import into *exampledb* on *server2* so that both databases contain the same data), and unlock the database so that it can be used again:

server1:

```
mysql -u root -p
```

On the MySQL shell, run the following commands:

server1:

```
USE exampledb;

FLUSH TABLES WITH READ LOCK;

SHOW MASTER STATUS;
```

The last command should show something like this (please write it down, we'll need it later on):

```
mysql> SHOW MASTER STATUS;
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000009 |       98 | exampledb    |                  |
+------------------+----------+--------------+------------------+
1 row in set (0.00 sec)

mysql>
```

Now don't leave the MySQL shell, because if you leave it, the database lock will be removed, and this is not what we want right now because we must create a database dump now. While the MySQL shell is still open, we open a **second** command line window where we create the SQL dump `snapshot.sql` and transfer it to `server2` (using scp):

server1:

```
cd /tmp

mysqldump -u root -pyourrootsqlpassword --opt exampledb > snapshot.sql

scp snapshot.sql root@192.168.0.101:/tmp
```

Afterwards, you can close the second command line window. On the first command line window, we can now unlock the database and leave the MySQL shell:

server1:

```
UNLOCK TABLES;

quit;
```

On `server2`, we can now import the SQL dump `snapshot.sql` like this:

server2:

```
/usr/bin/mysqladmin --user=root --password=yourrootsqlpassword stop-slave

cd /tmp

mysql -u root -pyourrootsqlpassword exampledb < snapshot.sql
```

Afterwards, we must find out about the master status of `server2` as well and write it down:

server2:

```
mysql -u root -p
```

```
USE exampledb;

FLUSH TABLES WITH READ LOCK;
```

```
SHOW MASTER STATUS;
```

```
mysql> SHOW MASTER STATUS;
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000009 |      783 | exampledb    |                  |
+------------------+----------+--------------+------------------+
1 row in set (0.00 sec)

mysql>
```

Then unlock the tables:

server2:

```
UNLOCK TABLES;
```

and run the following command to make `server2` a slave of `server1` (it is important that you replace the values in the following command with the values you got from the SHOW MASTER STATUS; command that we ran on server1!):

```
CHANGE MASTER TO MASTER_HOST='192.168.0.100', MASTER_USER='slave2_user', MASTER_PASSWORD='slave2_password', MASTER_LOG_FILE='mysql-bin.000009', MASTER_LOG_POS=98;
```

Finally start the slave:

server2:

```
START SLAVE;
```

Then check the slave status:

server2:

```
SHOW SLAVE STATUS;
```

It is important that both `Slave_IO_Running` and `Slave_SQL_Running` have the value `Yes` in the output (otherwise something went wrong, and you should check your setup again and take a look at `/var/log/syslog` to find out about any errors):

```
mysql> SHOW SLAVE STATUS;
+----------------------------------+---------------+-------------+-------------+---------------+------------------+---------------------+--------------------+---------------+-----------------------+------------------+-------------------+-------------
---+---------------------+--------------------+------------------------+-------------------------+----------------------------+------------+------------+--------------+--------------------+-----------------+-----------------+----------------+-------
--------+--------------------+--------------------+--------------------+-----------------+-------------------+----------------+-----------------------+
| Slave_IO_State                   | Master_Host   | Master_User | Master_Port | Connect_Retry | Master_Log_File  | Read_Master_Log_Pos | Relay_Log_File     | Relay_Log_Pos | Relay_Master_Log_File |
Slave_IO_Running | Slave_SQL_Running | Replicate_Do_DB | Replicate_Ignore_DB | Replicate_Do_Table | Replicate_Ignore_Table | Replicate_Wild_Do_Table | Replicate_Wild_Ignore_Table | Last_Errno |
Last_Error | Skip_Counter | Exec_Master_Log_Pos | Relay_Log_Space | Until_Condition | Until_Log_File | Until_Log_Pos | Master_SSL_Allowed | Master_SSL_CA_File | Master_SSL_CA_Path | Master_SSL_Cert
| Master_SSL_Cipher | Master_SSL_Key | Seconds_Behind_Master |
+----------------------------------+---------------+-------------+-------------+---------------+------------------+---------------------+--------------------+---------------+-----------------------+------------------+-------------------+-------------
---+---------------------+--------------------+------------------------+-------------------------+----------------------------+------------+------------+--------------+--------------------+-----------------+-----------------+----------------+-------
--------+--------------------+--------------------+--------------------+-----------------+-------------------+----------------+-----------------------+
| Waiting for master to send event | 192.168.0.100 | slave2_user |        3306 |            60 | mysql-bin.000009 |                  98 | slave-relay.000002 |           235 | mysql-bin.000009      | Yes              | Yes               |
exampledb     |                 |                 |                    |                    |                        |          0 |            |            0 |                 98 |             235 | None            |                |             0 | No                 |                    |
        |              |                |              |                     0 |
+----------------------------------+---------------+-------------+-------------+---------------+------------------+---------------------+--------------------+---------------+-----------------------+------------------+-------------------+-------------
---+---------------------+--------------------+------------------------+-------------------------+----------------------------+------------+------------+--------------+--------------------+-----------------+-----------------+----------------+-------
--------+--------------------+--------------------+--------------------+-----------------+-------------------+----------------+-----------------------+
1 row in set (0.00 sec)

mysql>
```

Afterwards, you can leave the MySQL shell on `server2`:

server2:

```
quit
```

Now the replication from `server1` to `server2` is set up. Next we must configure replication from `server2` to `server1`.

To do this, we stop the slave on `server1` and make it a slave of `server2`:

server1:

```
mysql -u root -p
```

```
STOP SLAVE;
```

Make sure that you use the values of the `SHOW MASTER STATUS;` command that you ran on `server2` in the following command:

server1:

```
CHANGE MASTER TO MASTER_HOST='192.168.0.101', MASTER_USER='slave1_user', MASTER_PASSWORD='slave1_password', MASTER_LOG_FILE='mysql-bin.000009',
MASTER_LOG_POS=783;
```

Then start the slave on `server1`:

server1:

```
START SLAVE;
```

Then check the slave status:

server1:

```
SHOW SLAVE STATUS;
```

It is important that both `Slave_IO_Running` and `Slave_SQL_Running` have the value `Yes` in the output (otherwise something went wrong, and you should check your setup again and take a look at `/var/log/syslog` to find out about any errors):

```
mysql> SHOW SLAVE STATUS;
+----------------------------------+---------------+-------------+-------------+--------------+-----------------+--------------------+-------------------+--------------+----------------------+-----------------+------------------+--------------
---+--------------------+-------------------+-----------------------+------------------------+---------------------------+-----------+-----------+-------------+--------------------+----------------+-----------------+---------------+-------
--------+--------------------+--------------------+--------------------+----------------+------------------+---------------+----------------------+
| Slave_IO_State                   | Master_Host   | Master_User  | Master_Port  | Connect_Retry  | Master_Log_File   | Read_Master_Log_Pos  | Relay_Log_File      | Relay_Log_Pos  | Relay_Master_Log_File  |
Slave_IO_Running  | Slave_SQL_Running  | Replicate_Do_DB  | Replicate_Ignore_DB  | Replicate_Do_Table  | Replicate_Ignore_Table  | Replicate_Wild_Do_Table  | Replicate_Wild_Ignore_Table  | Last_Errno  |
Last_Error | Skip_Counter | Exec_Master_Log_Pos | Relay_Log_Space | Until_Condition | Until_Log_File | Until_Log_Pos | Master_SSL_Allowed | Master_SSL_CA_File | Master_SSL_CA_Path | Master_SSL_Cert
| Master_SSL_Cipher | Master_SSL_Key | Seconds_Behind_Master |
+----------------------------------+---------------+-------------+-------------+--------------+-----------------+--------------------+-------------------+--------------+----------------------+-----------------+------------------+--------------
---+--------------------+-------------------+-----------------------+------------------------+---------------------------+-----------+-----------+-------------+--------------------+----------------+-----------------+---------------+-------
--------+--------------------+--------------------+--------------------+----------------+------------------+---------------+----------------------+
| Waiting for master to send event | 192.168.0.101 | slave1_user |        3306 |           60 | mysql-bin.000009 |                783 | slave-relay.000002 |           235 | mysql-bin.000009      | Yes              | Yes               |
exampledb      |                  |                |                    |                    |                        |         0 |           |          0 |             783 |           235 | None            |               |           0 | No               |                |
        |               |               |             |                 0 |
+----------------------------------+---------------+-------------+-------------+--------------+-----------------+--------------------+-------------------+--------------+----------------------+-----------------+------------------+--------------
---+--------------------+-------------------+-----------------------+------------------------+---------------------------+-----------+-----------+-------------+--------------------+----------------+-----------------+---------------+-------
--------+--------------------+--------------------+--------------------+----------------+------------------+---------------+----------------------+
1 row in set (0.00 sec)

mysql>
```

Afterwards you can leave the MySQL shell:

```
quit
```

If nothing went wrong, MySQL master-master replication should now be working. If it isn't, please check */var/log/syslog* for MySQL errors on *server1* and *server2*.

5 Links

- MySQL: **http://www.mysql.com**
- Debian: **http://www.debian.org**