Master-Slave-Replikation einrichten mit innobackupex

Mit innobackupex läßt sich auf einfache Weise im Live-Betrieb eine Master-Slave-Replikation einrichten. Die Datenbank muss ggf. nur für Änderungen an der Konfiguration neu gestartet werden (sofern nicht schon vorher entsprechend konfiguriert).

Voraussetzungen:

- ein Masterserver mit MySQL und vollen Benutzerrechten (root)
- auf dem Master müssen binlogs eingeschaltet und die server-id auf 1 gesetzt sein
- ein Slaveserver mit MySQL und vollen Benutzerrechten (root)
- auf dem Slaveserver müssen binlogs eingeschaltet (ist ggf. wichtig, wenn man einen Slave aus dem Slave erstellen will) und die server-id auf 2 gesetzt sein
- rsync auf beiden Servern

Zuerst installieren wir das Tool innobackupex, bzw. das Paket percona-xtrabackup. Das Toolkit läßt sich entweder direkt als .deb Paket herunterladen oder über das Percona-Repository installieren. Ich verwende das Repository.

Am einfachsten läßt sich das Percona Repo über das zur Verfügung gestellt .deb installieren. Dies fügt den Percona-GPG Key hinzu und legt ein entsprechendes sources.list-File an:

Percona Repository anlegen

```
wget https://repo.percona.com/apt/percona-release_0.1-4.$(lsb_release -sc)_all.deb
dpkg -i percona-release_0.1-4.$(lsb_release -sc)_all.deb
apt-get update
apt-get -y install percona-xtrabackup qpress
# oder für die neuere Version 2.4
apt-get -y install percona-xtrabackup-24 qpress
```

`qpress` wird für die komprimierten Backups benötigt. Dazu unten mehr.

Um in den nächsten Schritten mit weniger Passwortabfragen auszukommen, lege ich eine Datei mit den MySQL-Zugangsdaten an, die von allen Tools ausgelesen wird.

/root/.my.cnf

```
[client]
user=root
pass=secret
```

Jetzt ziehen wir eine Binärkopie/Snapshot des aktuellen MySQL-Datadirs (/var/lib/mysql) in unser Sicherungsverzeichnis. Darauf wird dann das aktuelle Binlog angewendet, um einen definitiven Zustand zu erhalten. Der erste Befehl legt das Backup in ein Unterverzeichnis von /backup mit dem aktuellen <TIMESTAMP>, das muss im zweiten Befehl berücksichtigt werden:

```
mkdir /backup

innobackupex --compress --compress-threads=6 --rsync --slave-info /backup
```

Da das Backup komprimiert wird (kann auch weggelassen werden), werden wir das apply-log erst auf dem Ziel durchführen. Siehe unten. Wenn man nicht komprimiert, kann man –apply-log direkt ausführen.

Jetzt muss noch der Slaveserver vorbereitet werden. Dazu wird der evtl. schon laufende MySQL-Server gestoppt und das Datendir gelöscht:

```
systemctl stop mysql.service
rm -rf /var/lib/mysql
```

Nun wird das Backup vom Master auf den Slave kopiert:

```
rsync -av -e ssh /backup/<TIMESTAMP> slave:/var/lib/mysql
```

Auf dem Slave muss das Backup jetzt entpackt und vorbereitet werden:

```
innobackupex --decompress /var/lib/mysql/
innobackupex --apply-log /var/lib/mysql/
```

Auf dem Slave setzen wir nun noch passende Berechtigungen:

```
chown -R mysql.mysql /var/lib/mysql
```

Der „debian-sys-maint“-User auf dem Slave hat nun das Kennwort vom Master. Das müssen wir in der Konfigurationsdatei anpassen (/etc/mysql/debian.cnf), damit u.a. das Init-Script mit dem MySQL-Server kommunizieren kann.

Danach kann der MySQL auf dem Slave mit den Daten vom Master gestartet werden:

```
systemctl start mysql.service
```

Für die Replikation brauchen wir noch einen User mit entsprechenden Rechten, root sollte dafür natürlich nicht verwendet werden. Auf dem Master legen wir den User wie folgt an (IP ist dabei die IP-Adresse des Slave-Servers, ein sicheres Passwort sollte gewählt werden):

```
mysql> GRANT REPLICATION SLAVE ON *.*  TO 'repl'@'IP' IDENTIFIED BY '$PASS';
```

Vom Slave aus testen, ob der Zugang funktioniert:

```
mysql -h master-server -u repl -p$PASS
```

Im letzten Schritt sagen wir dem Slave wer sein Herr ist und an welcher Position im Binlog die Replikation beginnen soll. Diese Infos stehen auf dem durch innobackupex angelegten File (xtrabackup_binlog_info, in diesem Beispiel lief der Master schon länger, daher die hohen Werte).

/var/lib/mysql/xtrabackup_binlog_info

```
mysql-bin.002284 40083139
```

In der MySQL-Shell konfigurieren wir den Slave:

```
mysql> change master to master_host='$MASTER-IP', master_user='repl', master_password='$PASS',
master_log_file='mysql-bin.002284', master_log_pos=40083139;

mysql> start slave;
```

Den Status der Replikation kann man mit diesem Befehl überprüfen:

```
mysql> show slave status\G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: $MASTER-IP
                  Master_User: repl
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.002289
          Read_Master_Log_Pos: 16815558
               Relay_Log_File: mysql-relay-bin.000016
                Relay_Log_Pos: 15880380
        Relay_Master_Log_File: mysql-bin.002289
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 15880234
              Relay_Log_Space: 16815903
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
```

```
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 49
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 1
1 row in set (0.00 sec)
```

Dabei sollte Slave_IO_Running und Slave_SQL_Running auf YES stehen. Seconds_Behind_Master zeigt ungefähr wieviele Sekunden der Slave dem Master-Server hinterherhängt.

Revision #1
Created 2021-07-27 13:01:00 UTC by magenbrot
Updated 2021-07-27 13:01:11 UTC by magenbrot