MYSQL REPLICATION BACKUP

Sommaire

[1. SSL CONFIGURATION 1](#_Toc423507227)

[a. Generate private key for CA 1](#_Toc423507228)

[b. Generate root certificate 1](#_Toc423507229)

[c. Create request of certificate for server 2](#_Toc423507230)

[d. Extract private key from req 2](#_Toc423507231)

[e. Generate server certificat from root certificate 2](#_Toc423507232)

[f. Configuration on /etc/my.cnf.d/server.cnf 2](#_Toc423507233)

[2. Replication 2](#_Toc423507234)

[a. Setup 2](#_Toc423507235)

[b. Bin file management 9](#_Toc423507236)

[c. Setup in order to be able to switch replication way 12](#_Toc423507237)

[3. Backup 12](#_Toc423507238)

[a. Full online backup 13](#_Toc423507239)

[b. Incremental online backup 13](#_Toc423507240)

[c. Bin log backup 14](#_Toc423507241)

[4. Restore 15](#_Toc423507242)

[a. from a full 15](#_Toc423507243)

[b. From an Incr 15](#_Toc423507244)

[c. BIN LOG FILE 16](#_Toc423507245)

[d. options. 16](#_Toc423507246)

# SSL CONFIGURATION

# Generate private key for CA

openssl genrsa 2048 > ca-key.pem

# Generate root certificate

openssl req -new -x509 -nodes -days 3600 -key ca-key.pem -out ca-cert.pem

# Create request of certificate for server

openssl req -newkey rsa:2048 -days 3600 -nodes -keyout server-key.pem -out server-req.pem

# Extract private key from req

openssl rsa -in server-key.pem -out server-key.pem

# Generate server certificat from root certificate

openssl x509 -req -in server-req.pem -days 3600 -CA ca-cert.pem -CAkey ca-key.pem -set\_serial 01 -out server-cert.pem

# Configuration on /etc/my.cnf.d/server.cnf

In [mysqld] part, you should specify :

* ssl-ca : location of the root certificate
* ssl-cert : location of the server certificate.
* ssl-key : location of private key of the server.

Example :

|  |
| --- |
| ssl-ca=/etc/mysql-ssl/ca-cert.pem  ssl-cert=/etc/mysql-ssl/frmsazr-p12-cert.pem  ssl-key=/etc/mysql-ssl/frmsazr-p12-key.pem |

# Replication

# Setup

<https://mariadb.com/kb/en/mariadb/setting-up-replication/>

<https://www.percona.com/doc/percona-xtrabackup/2.1/howtos/setting_up_replication.html>

La mise en réplication en gros (master : p12, slave : p11) :

/etc/my.cnf.d/server.cnf sur le master (p12), relance du moteur :

[mariadb]

log-basename=frmsazr-p12

log-bin

binlog-format=row

server\_id=12

mysql> show variables like '%bin%';

+-----------------------------------------+--------------------------------------+

| Variable\_name | Value |

+-----------------------------------------+--------------------------------------+

| bind\_address | \* |

| binlog\_cache\_size | 32768 |

| binlog\_checksum | CRC32 |

| binlog\_direct\_non\_transactional\_updates | OFF |

| binlog\_error\_action | IGNORE\_ERROR |

| binlog\_format | ROW |

| binlog\_gtid\_simple\_recovery | OFF |

| binlog\_max\_flush\_queue\_time | 0 |

| binlog\_order\_commits | ON |

| binlog\_row\_image | FULL |

| binlog\_rows\_query\_log\_events | OFF |

| binlog\_stmt\_cache\_size | 32768 |

| binlogging\_impossible\_mode | IGNORE\_ERROR |

| innodb\_api\_enable\_binlog | OFF |

| innodb\_locks\_unsafe\_for\_binlog | OFF |

| log\_bin | ON |

| log\_bin\_basename | /var/lib/mysql/frmsazr-p12-bin |

| log\_bin\_index | /var/lib/mysql/frmsazr-p12-bin.index |

| log\_bin\_trust\_function\_creators | OFF |

| log\_bin\_use\_v1\_row\_events | OFF |

| max\_binlog\_cache\_size | 18446744073709547520 |

| max\_binlog\_files | 0 |

| max\_binlog\_size | 1073741824 |

| max\_binlog\_stmt\_cache\_size | 18446744073709547520 |

| simplified\_binlog\_gtid\_recovery | OFF |

| sql\_log\_bin | ON |

| sync\_binlog | 0 |

+-----------------------------------------+--------------------------------------+

/etc/my.cnf.d/server.cnf sur le slave (p11), relance du moteur :

[mariadb]

server\_id=11

read-only=true

Côté master, en root, créer le user de réplication :

GRANT REPLICATION SLAVE ON \*.\* TO 'maria\_slave'@'192.168.2.4' IDENTIFIED BY '%D\*WN28dCNFBl(' ;

FLUSH PRIVILEGES ;

 GRANT REPLICATION SLAVE ON \*.\* TO 'repl\_slave'@'192.168.2.4' IDENTIFIED BY '%D\*WN28dCNFBl(' ;

FLUSH PRIVILEGES ;

In case, you use X509 certificate to connect :

GRANT REPLICATION SLAVE ON \*.\* TO 'maria\_slave'@'192.168.2.4' REQUIRE X509 ;

FLUSH PRIVILEGES ;

GRANT REPLICATION SLAVE ON \*.\* TO 'repl\_slave'@'192.168.2.4' REQUIRE X509 ;

FLUSH PRIVILEGES ;

Côté master toujours, figer la base, noter le nom et la position du binlog, et prendre un backup :

FLUSH TABLES WITH READ LOCK;

show master status;

+------------------------+----------+--------------+------------------+

| File                   | Position | Binlog\_Do\_DB | Binlog\_Ignore\_DB |

+------------------------+----------+--------------+------------------+

| frmsazr-p12-bin.000001 |      892 |              |                  |

+------------------------+----------+--------------+------------------+

<< faire un backup complet sans arrêter cette session ! >>

unlock tables;

C’est tout pour le master. Transférer le dump vers le slave, et restorer sur le slave :

mysql -u root -p < /tmp/dump.sql

Se connecter au slave, lui dire où est son master et à partir d’où appliquer les logs (parametres MASTER\_LOG\_POS et MASTER\_LOG\_FILE obtenu via un SHW MASTER status):

CHANGE MASTER TO

MASTER\_HOST='192.168.1.4',

MASTER\_USER='maria\_slave',

MASTER\_PASSWORD='%D\*WN28dCNFBl(',

MASTER\_PORT=3306,

MASTER\_LOG\_FILE='frmsazr-p12-bin.000001',

MASTER\_LOG\_POS=892,

MASTER\_CONNECT\_RETRY=10;

 CHANGE MASTER TO

MASTER\_HOST='192.168.1.4',

MASTER\_USER='repl\_slave',

MASTER\_PASSWORD='%D\*WN28dCNFBl(',

MASTER\_PORT=3306,

MASTER\_LOG\_FILE='frmsazr-p12-bin.000002',

MASTER\_LOG\_POS=120,

MASTER\_CONNECT\_RETRY=10;

In case of X509 authentification :

CHANGE MASTER TO

MASTER\_HOST='192.168.1.4',

MASTER\_USER='maria\_slave',

MASTER\_SSL=1,

MASTER\_SSL\_CA='/etc/mysql-ssl/ca-cert.pem',

MASTER\_SSL\_CERT='/etc/mysql-ssl/frmsazr-p11-cert.pem',

MASTER\_SSL\_KEY='/etc/mysql-ssl/frmsazr-p11-key.pem',

MASTER\_PORT=3306,

MASTER\_LOG\_FILE='frmsazr-p12-bin.000027',

MASTER\_LOG\_POS=1681,

MASTER\_CONNECT\_RETRY=10;

CHANGE MASTER TO

MASTER\_HOST='192.168.1.4',

MASTER\_USER='repl\_slave',

MASTER\_SSL=1,

MASTER\_SSL\_CA='/etc/mysql-ssl/ca-cert.pem',

MASTER\_SSL\_CERT='/etc/mysql-ssl/frmsazr-p11-cert.pem',

MASTER\_SSL\_KEY='/etc/mysql-ssl/frmsazr-p11-key.pem',

MASTER\_PORT=3306,

MASTER\_LOG\_FILE='frmsazr-p12-bin.000002',

MASTER\_LOG\_POS=120,

MASTER\_CONNECT\_RETRY=10;

Et démarrer le tout:

start slave;

Et c’est parti.

A voir: purge des binlogs, switchover/failover, détails de la proc de backup sur le slave.

MariaDB [(none)]> show slave status \G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Slave\_IO\_State: Waiting for master to send event

Master\_Host: 192.168.1.4

Master\_User: maria\_slave

Master\_Port: 3306

Connect\_Retry: 10

Master\_Log\_File: frmsazr-p12-bin.000001

Read\_Master\_Log\_Pos: 491560

Relay\_Log\_File: frmsazr-p11-relay-bin.000004

Relay\_Log\_Pos: 489644

Relay\_Master\_Log\_File: frmsazr-p12-bin.000001

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: 491560

Relay\_Log\_Space: 489947

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: 0

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: 12

Master\_SSL\_Crl:

Master\_SSL\_Crlpath:

Using\_Gtid: No

Gtid\_IO\_Pos:

1 row in set (0.00 sec)

pt-slave-delay -p 'G2IZOM=nU3#>lh' --delay 1m --interval 15s --run-time 10m

# Bin file management

<https://mariadb.com/kb/en/mariadb/using-and-maintaining-the-binary-log/>

In order not to have big bin files, we should use parameters:

expire\_logs\_days = 2

max\_binlog\_size = 100M

In /etc/my.cnf.d.server.cnf

For Percona, use variable max\_binlog\_files (if not null corresponds to number of file to be kept).

If we need to purge bin file, you should check on slave, that replication is OK (see show slave STATUS \G).

MariaDB [(none)]> SHOW BINARY LOGS ;

+------------------------+------------+

| Log\_name | File\_size |

+------------------------+------------+

| frmsazr-p12-bin.000001 | 367 |

| frmsazr-p12-bin.000002 | 382 |

| frmsazr-p12-bin.000003 | 48885 |

| frmsazr-p12-bin.000004 | 1002038752 |

| frmsazr-p12-bin.000005 | 332 |

+------------------------+------------+

5 rows in set (0.00 sec)

MariaDB [(none)]> PURGE BINARY LOGS BEFORE '2015-03-19 00:00:00' ;

Query OK, 0 rows affected (0.06 sec)

MariaDB [(none)]> SHOW BINARY LOGS ;

+------------------------+------------+

| Log\_name | File\_size |

+------------------------+------------+

| frmsazr-p12-bin.000003 | 48885 |

| frmsazr-p12-bin.000004 | 1002038752 |

| frmsazr-p12-bin.000005 | 332 |

+------------------------+------------+

3 rows in set (0.00 sec)

Example I want to delete frmsazr-p12-bin.000004:

MariaDB [(none)]> SHOW BINARY LOGS ;

+------------------------+------------+

| Log\_name | File\_size |

+------------------------+------------+

| frmsazr-p12-bin.000001 | 367 |

| frmsazr-p12-bin.000002 | 382 |

| frmsazr-p12-bin.000003 | 48885 |

| frmsazr-p12-bin.000004 | 1002038752 |

| frmsazr-p12-bin.000005 | 332 |

+------------------------+------------+

5 rows in set (0.00 sec)

I go on slave p11:

MariaDB [(none)]> show slave status \G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Slave\_IO\_State: Waiting for master to send event

Master\_Host: 192.168.1.4

Master\_User: maria\_slave

Master\_Port: 3306

Connect\_Retry: 10

Master\_Log\_File: frmsazr-p12-bin.000005

Read\_Master\_Log\_Pos: 332

Relay\_Log\_File: frmsazr-p11-relay-bin.000014

Relay\_Log\_Pos: 541

Relay\_Master\_Log\_File: frmsazr-p12-bin.000005

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: 332

Relay\_Log\_Space: 844

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: 0

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: 12

Master\_SSL\_Crl:

Master\_SSL\_Crlpath:

Using\_Gtid: No

Gtid\_IO\_Pos:

1 row in set (0.00 sec)

The master log file is Master\_Log\_File: frmsazr-p12-bin.000005. So I can remove frmsazr-p12-bin.000004 with :

MariaDB [(none)]> PURGE BINARY LOGS TO 'frmsazr-p12-bin.000005' ;

Query OK, 0 rows affected (0.37 sec)

MariaDB [(none)]> SHOW BINARY LOGS ;

+------------------------+-----------+

| Log\_name | File\_size |

+------------------------+-----------+

| frmsazr-p12-bin.000005 | 332 |

+------------------------+-----------+

1 row in set (0.00 sec)

It will keep only bin file from file frmsazr-p12-bin.000005 (so if I have a bin file generated after bin file frmsazr-p12-bin.000005 as frmsazr-p12-bin.000006 for instance, it will keep frmsazr-p12-bin.000005 and frmsazr-p12-bin.000006).

# Setup in order to be able to switch replication way

GRANT REPLICATION SLAVE ON \*.\* TO 'maria\_slave'@'192.168.1.4' IDENTIFIED BY '%D\*WN28dCNFBl(' ;

# Backup

For backup we use innobackupexe instead of xtrabackup because it offers more possibilities.

[innobackupex](http://www.percona.com/doc/percona-xtrabackup/2.2/innobackupex/innobackupex_script.html)

a wrapper script that provides functionality to backup a whole *MySQL* database instance with [MyISAM](http://www.percona.com/doc/percona-xtrabackup/2.2/glossary.html" \l "term-myisam), [InnoDB](http://www.percona.com/doc/percona-xtrabackup/2.2/glossary.html" \l "term-innodb), and [XtraDB](http://www.percona.com/doc/percona-xtrabackup/2.2/glossary.html" \l "term-xtradb)tables.

[xtrabackup](http://www.percona.com/doc/percona-xtrabackup/2.2/xtrabackup_bin/xtrabackup_binary.html)

a compiled *C* binary, which copies only [InnoDB](http://www.percona.com/doc/percona-xtrabackup/2.2/glossary.html" \l "term-innodb) and [XtraDB](http://www.percona.com/doc/percona-xtrabackup/2.2/glossary.html" \l "term-xtradb) data

<http://www.percona.com/doc/percona-xtrabackup/2.2/>

We need to create user in database:

Htpp://dev.mysql.com/doc/refman/5.0/en/stop-slave.html

CREATE USER 'bkpuser'@'localhost' IDENTIFIED BY 'eUiyDsnnmRzLZARAx7XT';

GRANT RELOAD, LOCK TABLES, REPLICATION CLIENT, SUPER ON \*.\* TO 'bkpuser'@'localhost';

FLUSH PRIVILEGES;

FLUSH PRIVILEGES ;

GRANT RELOAD, SUPER ,REPLICATION ON \*.\* TO 'bkpuser'@'192.168.2.4' IDENTIFIED BY 'eUiyDsnnmRzLZARAx7XT' WITH GRANT OPTION ;

For X509 authentification, we need to add in /etc/my.cnf.d/server.cnf:

[client]

ssl-ca=/etc/mysql-ssl/ca-cert.pem

ssl-cert=/etc/mysql-ssl/frmsazr-p11-cert.pem

ssl-key=/etc/mysql-ssl/frmsazr-p11-key.pem

For user creation :

GRANT RELOAD,SUPER ON \*.\* TO 'bkpuser'@'192.168.2.4' REQUIRE X509 ;

GRANT RELOAD,SUPER ON \*.\* TO 'bkpuser'@'192.168.1.4' REQUIRE X509 ;

GRANT RELOAD,SUPER ON \*.\* TO 'bkpuser'@'localhost' REQUIRE X509 ;

FLUSH PRIVILEGES;

# Full online backup

For full backup we do :

backup\_cmd="innobackupex --incremental --user=${BACKUP\_USER} ${BACKUP\_DIR\_NAME} --SLAVE-INFO --SAFE-SLAVE-BACKUP --incremental-basedir=${BACKUP\_DIR}/last\_full\_backup --no-timestamp "

prepare\_cmd="innobackupex --apply-log --redo-only ${BACKUP\_DIR}/last\_full\_backup"

# Incremental online backup

For incremental :

backup\_cmd="innobackupex --incremental --user=${BACKUP\_USER} ${BACKUP\_DIR\_NAME} --SLAVE-INFO --SAFE-SLAVE-BACKUP --incremental-basedir=${BACKUP\_DIR}/last\_full\_backup --no-timestamp "

prepare\_cmd="innobackupex --apply-log --redo-only ${BACKUP\_DIR}/last\_full\_backup"

We can compress backup but you will have to do preparation before restoring and you can do incremental after…

Example :

innobackupex --stream=tar --user=bkpuser --password=eUiyDsnnmRzLZARAx7XT --SLAVE-INFO --SAFE-SLAVE-BACKUP /var/lib/backups | gzip - > /var/lib/backups/full\_20150319\_161509.tgz

|  |  |  |
| --- | --- | --- |
|  | Size of backup | Duration |
| Non compress full | 2.5GB | 2 min 25 |
| Compress full (tar gunzip) | 761 MB | 8min 11s |

# Bin log backup

<http://www.percona.com/blog/2014/03/28/innodb-redo-log-archiving/>

<http://www.percona.com/blog/2012/01/18/backing-up-binary-log-files-with-mysqlbinlog/>

mysqlbinlog -R --stop-position=4260 --host 192.168.1.4 --port 3306 --password='eUiyDsnnmRzLZARAx7XT' --user='bkpuser' frmsazr-p12-bin.000005 frmsazr-p12-bin.000006 frmsazr-p12-bin.000007

mysqlbinlog -R --start-position=4260 --stop-position=4798 --host 192.168.1.4 --port 3306 --password='eUiyDsnnmRzLZARAx7XT' --user='bkpuser' frmsazr-p12-bin.000007 > /var/lib/backups/frmsazr-p12-bin.backup3

[root@frmsazr-p11 ~]# echo "show master status " | mysql --host 192.168.1.4 --port 3306 --password='eUiyDsnnmRzLZARAx7XT' --user='bkpuser' | grep -v File\_size | grep -v Position

frmsazr-p12-bin.000007 4798

cat /var/lib/backups/inc\_20150319\_163227\_bin.backup | grep frmsazr-p12-bin

# Restore

# from a full

restore\_full()

{

echo "restore of FULL $1"

log\_date=`date +%Y%m%d\_%H%M%S`

tar\_cmd="tar -czvf ${BACKUP\_DIR}/old\_mysql\_content${log\_date}.tgz ${datadir}"

clean\_cmd="rm -rf ${datadir}/\*"

restore\_cmd="innobackupex --copy-back ${BACKUP\_DIR}/${1}"

chown\_cmd="chown -R mysql:mysql ${datadir}"

launch\_cmd "${tar\_cmd}" || exit $?

launch\_cmd "${clean\_cmd}"|| exit $?

launch\_cmd "${restore\_cmd}"|| exit $?

launch\_cmd "${chown\_cmd}"|| exit $?

}

# From an Incr

restore\_inc()

{

echo "restore of INC $1"

log\_date=`date +%Y%m%d\_%H%M%S`

tar\_cmd="tar -czvf ${BACKUP\_DIR}/old\_mysql\_content${log\_date}.tgz ${datadir}"

clean\_cmd="rm -rf ${datadir}/\*"

chown\_cmd="chown -R mysql:mysql ${datadir}"

from\_lsn=$(grep from\_lsn ${BACKUP\_DIR}/${1}/xtrabackup\_checkpoints | cut -d= -f2 )

full\_backup\_name=$(dirname $(grep "to\_lsn =${from\_lsn}" ${BACKUP\_DIR}/full\*/xtrabackup\_checkpoints | cut -d: -f1))

if [[ -d ${full\_backup\_name} ]]; then

cp\_cmd="cp -rp ${full\_backup\_name} ${BACKUP\_DIR}/restore${log\_date} "

prepare\_cmd2="innobackupex --apply-log --redo-only ${BACKUP\_DIR}/restore${log\_date} --incremental-dir=${BACKUP\_DIR}/${1}"

restore\_cmd="innobackupex --copy-back ${BACKUP\_DIR}/restore${log\_date}"

else

log\_failure\_msg "UNABLE TO FIND FULL BACKUP FOR INC BACKUP ${1}: ${full\_backup\_name} DOES NOT EXIST"

fi

launch\_cmd "${tar\_cmd}"|| exit $?

launch\_cmd "${clean\_cmd}"|| exit $?

launch\_cmd "${cp\_cmd}"|| exit $?

launch\_cmd "${prepare\_cmd2}"|| exit $?

launch\_cmd "${restore\_cmd}"|| exit $?

launch\_cmd "${chown\_cmd}"|| exit $?

rm -rf ${BACKUP\_DIR}/restore${log\_date}

}

# BIN LOG FILE

# options.

When you restore a slave replicate database, you should setup the replication see 1Set up.

Look at the content of the file xtrabackup\_binlog\_info, it will be something like:

TheSlave$ cat /var/lib/mysql/xtrabackup\_binlog\_info

TheMaster-bin.000001 481

But you should have some error :

MariaDB [(none)]> CHANGE MASTER TO

    -> MASTER\_HOST='192.168.1.4',

    -> MASTER\_USER='maria\_slave',

    -> MASTER\_PASSWORD='%D\*WN28dCNFBl(',

    -> MASTER\_PORT=3306,

    -> MASTER\_LOG\_FILE='frmsazr-p12-bin.000001',

    -> MASTER\_LOG\_POS=892,

    -> MASTER\_CONNECT\_RETRY=10;

Query OK, 0 rows affected (0.14 sec)

MariaDB [(none)]> start slave;

Query OK, 0 rows affected (0.01 sec)

MariaDB [(none)]> show slave status \G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

               Slave\_IO\_State:

                  Master\_Host: 192.168.1.4

                  Master\_User: maria\_slave

                  Master\_Port: 3306

                Connect\_Retry: 10

              Master\_Log\_File: frmsazr-p12-bin.000001

          Read\_Master\_Log\_Pos: 892

               Relay\_Log\_File: frmsazr-p11-relay-bin.000001

                Relay\_Log\_Pos: 4

        Relay\_Master\_Log\_File: frmsazr-p12-bin.000001

             Slave\_IO\_Running: No

            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: 892

              Relay\_Log\_Space: 248

              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: NULL

Master\_SSL\_Verify\_Server\_Cert: No

                Last\_IO\_Errno: 1236

                Last\_IO\_Error: Got fatal error 1236 from master when reading data from binary log: 'Client requested master to start replication from impossible position; the first event 'frmsazr-p12-bin.000001' at 892, the last event read from 'frmsazr-p12-bin.000001' at 4, the last byte read from 'frmsazr-p12-bin.000001' at 4.'

               Last\_SQL\_Errno: 0

               Last\_SQL\_Error:

  Replicate\_Ignore\_Server\_Ids:

             Master\_Server\_Id: 12

               Master\_SSL\_Crl:

           Master\_SSL\_Crlpath:

                   Using\_Gtid: No

                  Gtid\_IO\_Pos:

1 row in set (0.00 sec)

Which you can solve like this :

MariaDB [(none)]> stop slave ;

Query OK, 0 rows affected (0.02 sec)

MariaDB [(none)]> CHANGE MASTER TO MASTER\_LOG\_FILE='frmsazr-p12-bin.000001' , MASTER\_LOG\_POS=4;

Query OK, 0 rows affected (0.43 sec)

MariaDB [(none)]> start slave;

Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> show slave status \G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

               Slave\_IO\_State: Waiting for master to send event

                  Master\_Host: 192.168.1.4

                  Master\_User: maria\_slave

                  Master\_Port: 3306

                Connect\_Retry: 10

              Master\_Log\_File: frmsazr-p12-bin.000001

          Read\_Master\_Log\_Pos: 318

               Relay\_Log\_File: frmsazr-p11-relay-bin.000002

                Relay\_Log\_Pos: 611

        Relay\_Master\_Log\_File: frmsazr-p12-bin.000001

             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: 318

              Relay\_Log\_Space: 914

              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: 0

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: 12

               Master\_SSL\_Crl:

           Master\_SSL\_Crlpath:

                   Using\_Gtid: No

                  Gtid\_IO\_Pos:

1 row in set (0.00 sec)

My.cnf on p12:

[server]

server-id=12

log-bin

binlog-format=row

[client]

ssl-ca=/etc/mysql-ssl/ca-cert.pem

ssl-cert=/etc/mysql-ssl/frmsazr-p12-cert.pem

ssl-key=/etc/mysql-ssl/frmsazr-p12-key.pem

[mysqld]

datadir=/var/lib/mysql

ssl-ca=/etc/mysql-ssl/ca-cert.pem

ssl-cert=/etc/mysql-ssl/frmsazr-p12-cert.pem

ssl-key=/etc/mysql-ssl/frmsazr-p12-key.pem

On p11:

[server]

server-id=11

read-only=true

[client]

ssl-ca=/etc/mysql-ssl/ca-cert.pem

ssl-cert=/etc/mysql-ssl/frmsazr-p11-cert.pem

ssl-key=/etc/mysql-ssl/frmsazr-p11-key.pem

[mysqld]

datadir=/var/lib/mysql

ssl-ca=/etc/mysql-ssl/ca-cert.pem

ssl-cert=/etc/mysql-ssl/frmsazr-p11-cert.pem

ssl-key=/etc/mysql-ssl/frmsazr-p11-key.pem

Script pour backup bin :

#!/bin/sh

BACKUP\_USER=bkpuser

#PASSWORD=eUiyDsnnmRzLZARAx7XT

BACKUP\_DIR=/var/lib/backups

LOG\_DIR=/var/log/mariadb

log\_date=`date +%Y%m%d\_%H%M%S`

BIN\_BACKUP\_NAME=$(readlink -f ${BACKUP\_DIR}/last\_backup)"\_bin.backup"

master\_info=$(echo "show slave status \G" | mysql --user=${BACKUP\_USER} | egrep -e 'Master\_Host|Master\_Port')

master\_host=$( echo "${master\_info}" | grep Host | cut -d: -f2)

master\_port=$( echo "${master\_info}" | grep Port | cut -d: -f2)

launch\_cmd ()

{

desc=$(expr substr "${1}" 1 42)

$1

if [[ $? -ne 0 ]]; then

echo "${log}"

echo "ERROR `date +%Y-%m-%d\_%H:%M:%S` ${desc}"

exit 5

else

echo "SUCCESS `date +%Y-%m-%d\_%H:%M:%S` ${desc}"

return 0

fi

}

set -x

master\_status=$( echo "show master status " | mysql --host 192.168.1.4 --port 3306 --user='bkpuser' | grep -v File\_size | grep -v Position )

last=$(echo "$master\_status" | awk '{print $2 }')

CURRENT\_BIN=$(echo "$master\_status" | awk '{print $1 }')

ALL\_BIN=$(echo $( echo "show BINARY LOGS " | mysql --host 192.168.1.4 --port 3306 --user='bkpuser' | grep -v File\_size| awk '{ print $1 }' ))

if [[ -f ${BIN\_BACKUP\_NAME} ]]; then # FIRST BACKUP S # RESTART SINCE THE LAST BACKUP

first=$( cat ${BIN\_BACKUP\_NAME} | grep end\_log\_pos | tail -1 | awk '{ print $7}')

FIRST\_BIN=$( cat ${BIN\_BACKUP\_NAME} | grep $(echo ${CURRENT\_BIN}| cut -d'.' -f1) | tail -1 | awk '{print $10}' )

if [[ ${FIRST\_BIN} = ${CURRENT\_BIN} ]]; then #WE DON4T CHANGE BIN FILE SINCE LAST BIN BACKUP

mysqlbinlog -R --start-position=${first} --stop-position=${last} --host ${master\_host} --port ${master\_port} --user=${BACKUP\_USER} ${FIRST\_BIN} >> ${BIN\_BACKUP\_NAME} || echo "FAILURE $? "

else

if [[ ${first} -ge ${last} ]]; then # RESTART SINCE THE LAST BACKUP

mysqlbinlog -R --stop-position=${last} --host ${master\_host} --port ${master\_port} --user=${BACKUP\_USER} ${ALL\_BIN} >> ${BIN\_BACKUP\_NAME} || echo "FAILURE $? "

else

mysqlbinlog -R --start-position=${first} --stop-position=${last} --host ${master\_host} --port ${master\_port} --user=${BACKUP\_USER} ${ALL\_BIN} >> ${BIN\_BACKUP\_NAME} || echo "FAILURE $? "

fi

fi

else

mysqlbinlog -R --stop-position=${last} --host ${master\_host} --port ${master\_port} --user=${BACKUP\_USER} ${ALL\_BIN} >> ${BIN\_BACKUP\_NAME} || echo "FAILURE $? "

fi