Why MySQL Replication Fails, and How to Get it Back

September, 26, 2017
Sveta Smirnova
Sveta Smirnova

- MySQL Support engineer
- Author of
  - MySQL Troubleshooting
  - JSON UDF functions
  - FILTER clause for MySQL
- Speaker
  - Percona Live, OOW, Fosdem, DevConf, HighLoad...
Thank You Sponsors!
Replication in MySQL

- Exists since very first versions
Replication in MySQL

- Exists since very first versions
- Easy to use
Replication in MySQL

- Exists since very first versions
- Easy to use
- Minimal setup
Turn Replication On

● Master
  ● --log-bin
  ● --server-id
  ● GRANT REPLICATION SLAVE ON *.* ...
Turn Replication On

- **Master**
  - --log-bin
  - --server-id
  - GRANT REPLICATION SLAVE ON *.* ...

- **Slave**
  - --server-id
  - CHANGE MASTER ...
  - START SLAVE
Star
Creative
Typical Replication Errors
Replication Stopped
Slave Lags from the Master
Increased Resource Usage on Master
# Not a Full List!

## Tickets (1600)

<table>
<thead>
<tr>
<th>ID</th>
<th>Subject</th>
<th>Required</th>
<th>Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>#150388</td>
<td>Slave stuck reading event?</td>
<td>Yesterday 00:49</td>
<td>Yesterday 04:58</td>
</tr>
<tr>
<td>#177366</td>
<td>mysql slave replication</td>
<td>Wednesday 22:34</td>
<td>Wednesday 00:11</td>
</tr>
<tr>
<td>#112449</td>
<td>slave failed to initialize</td>
<td>Tuesday 17:19</td>
<td>Tuesday 18:00</td>
</tr>
<tr>
<td>#140827</td>
<td>Slave replication issue</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>RECS slave for Percona</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Slave Hardware Recommendations</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Slave is breaking repeatedly</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>slave server logging behind</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Slave replication issue</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>New slave for?</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Configure wrong slave_threads</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>SLAVE RE-CONFIGURATION</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Master Slave Wiki</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Slave out of sync</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Slave not starting</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Slave replication issue</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Overloading on read slaves</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
<tr>
<td>#137037</td>
<td>Bringing up new slave</td>
<td>Tuesday 17:10</td>
<td>Wednesday 18:01</td>
</tr>
</tbody>
</table>

**Note:** The list above is not complete and may not represent all issues.
MySQL Replication: Must Know
Asynchronous

Master

Slave

← Initiates
Asynchronous

Master

Slave

← Initiates
← Requests a packet
Asynchronous

Master

Sends the packet →

Slave

← Initiates
← Requests a packet
Asynchronous

Master

Sends the packet →

Slave

← Initiates

← Requests a packet

... ?
Did Slave Receive Data?

- Network error
- Authorization
Tool #1: SHOW SLAVE STATUS

Slave_IO_State: Waiting for master to send event
Master_Host: 127.0.0.1
Master_User: root
Master_Port: 13000
Connect_Retry: 60

Read_Master_Log_Pos: 6381061
Relay_Log_Pos: 1156
Relay_Master_Log_File: master-bin.000001

Slave_IO_Running: Yes
Slave_SQL_Running: No
Replicate_Do_DB:
Replicate_Ignore_DB:
Replicate_Do_Table:
Replicate_Ignore_Table:
Replicate_Wild_Do_Table:
Replicate_Wild_Ignore_Table:

Last_Errno: 1032
Last_Error: Could not execute Update_rows event on...

Skip_Counter: 0
Exec_Master_Log_Pos: 989
Relay_Log_Space: 63814652
Until_Condition: None

Master_SSL_Allowed: No | Master_SSL_CA_File:
Master_SSL_Cert:
Master_SSL_Client:
Master_SSL_Cipher:
Master_SSL_Crl:
Master_SSL_CrlPath:

Seconds_Behind_Master: NULL
Master_SSL_Verify_Server_Cert: No
Last_IO_Errno: 0
Last_IO_Error:
Last_SQL_Errno: 1032
Last_SQL_Error: Could not execute Update_rows event on...

Replicate_Ignore_Server_Ids:
Master_Server_Id: 1
Master_UUID: d08c509e-6857-11e6-8872-30b5c2208a0f
Master_Info_File: mysql.slave_master_info
SQL_Delay: 0
SQL_Remaining_Delay: NULL
Slave_SQL_Running_State:
Master_Retry_Count: 10
Master_Bind:
Last_IO_Error_Timestamp: 160823 15:11:21

Master_SSL_Crl:
Master_SSL_CrlPath:
Retrieved_Gtid_Set:
Executed_Gtid_Set:
Auto_Position: 0
Replicate_Rewrite_DB:

Channel_Name: master-1 | Master_TLS_Version:
Network Errors

Slave_IO_Running: Connecting
Slave_SQL_Running: Yes
...
  Last_IO_Errno: 1045
  Last_IO_Error: error connecting to master ‘root@127.0.0.1:13000’ -
  Last_SQL_Errno: 0
  Last_SQL_Error:
  ...
Slave_SQL_Running_State: Slave has read all relay log; waiting for more updates
Master_Retry_Count: 86400
Master_Bind:
Last_IO_Error_Timestamp: 160824 03:18:36
Last_SQL_Error_Timestamp:
#2: connection_status in Performance Schema

```
mysql> select * from performance_schema.replication_connection_status
+--------------------------+--------------------------+--------------------------+--------------------------+--------------------------+
<table>
<thead>
<tr>
<th>CHANNEL_NAME</th>
<th>GROUP_NAME</th>
<th>SOURCE_UUID</th>
<th>THREAD_ID</th>
<th>SERVICE_STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>NULL</td>
<td>CONNECTING</td>
</tr>
<tr>
<td>COUNT_RECEIVED_HEARTBEATS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAST_HEARTBEAT_TIMESTAMP</td>
<td>0000-00-00 00:00:00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECEIVED_TRANSACTION_SET</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAST_ERROR_NUMBER</td>
<td>1045</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAST_ERROR_MESSAGE</td>
<td>error connecting to master 'root@127.0.0.1:13000' - retry-time: 60 retries: 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAST_ERROR_TIMESTAMP</td>
<td>2016-08-24 03:21:36</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
+--------------------------+--------------------------+--------------------------+--------------------------+--------------------------+
1 row in set (0.01 sec)
```
2016-08-24T00:18:36.077384Z 3 [ERROR] Slave I/O for channel ''': error connecting to master 'root@127.0.0.1:13000' - retry-time: 60 retries: 1, Error_code: 1045
2016-08-24T00:19:36.299011Z 3 [ERROR] Slave I/O for channel ''': error connecting to master 'root@127.0.0.1:13000' - retry-time: 60 retries: 2, Error_code: 1045
2016-08-24T00:20:36.485315Z 3 [ERROR] Slave I/O for channel ''': error connecting to master 'root@127.0.0.1:13000' - retry-time: 60 retries: 3, Error_code: 1045
2016-08-24T00:21:36.677915Z 3 [ERROR] Slave I/O for channel ''': error connecting to master 'root@127.0.0.1:13000' - retry-time: 60 retries: 4, Error_code: 1045
2016-08-24T00:22:36.872066Z 3 [ERROR] Slave I/O for channel ''': error connecting to master 'root@127.0.0.1:13000' - retry-time: 60 retries: 5, Error_code: 1045
$ perror 1045
MySQL error code 1045 (ER_ACCESS_DENIED_ERROR): Access denied for user '%-.48s'@'%-.64s'
(using password: %s)
#5: MySQL Command Line Client

- On the slave

```bash
$ mysql -h127.0.0.1 -P13000 -uslave_user -pslave_password
Warning: Using a password on the command line interface can be insecure.
ERROR 1045 (28000): Access denied for user 'slave_user'@'localhost' (using password: YES)
```
#5: MySQL Command Line Client

- On the slave
- On the master

```sql
mysql> SHOW GRANTS;
+-----------------------------------------+
| Grants for slave_user@% |
+-----------------------------------------+
| GRANT SELECT ON *.* TO 'slave_user'@'%' |
+-----------------------------------------+
1 row in set (0.00 sec)
```
#5: MySQL Command Line Client

- On the slave
- On the master
- Fix privileges on master

```
GRANT REPLICATION SLAVE
ON *.* TO 'slave_user'@'%
```
#5: MySQL Command Line Client

- On the slave
- On the master
- Fix privileges on master
- Restart replication
Semisynchronous plugin

Master

Slave

← Initiates
Semisynchronous plugin

Master

Slave

← Initiates

← Requests a packet
Semisynchronous plugin

Master
Sends the packet →

Slave
← Initiates
← Requests a packet
Semisynchronous plugin

**Master**
- Sends the packet →
- Waits "Ack"

**Slave**
- Initiates ←
- Requests a packet ←
Semisynchronous plugin

Master

Sends the packet →
Waits "Ack"

Slave

← Initiates
← Requests a packet

← Sends "Ack"
Semisynchronous: Troubleshooting

- Writes on master are slower
Semisynchronous: Troubleshooting

- Writes on master are slower
- How many "Ack"-s master waits?
Semisynchronous: Troubleshooting

- Writes on master are slower
- How many "Ack"-s master waits?
  - Before 5.7: from single slave
Semisynchronous: Troubleshooting

- Writes on master are slower
- How many "Ack"-s master waits?
  - Before 5.7: from single slave
  - Now in MySQL:
    \[ \text{rpl}_\text{semi}_\text{sync}_\text{master}_\text{wait}_\text{for}_\text{slave}_\text{count} \]
Semisynchronous: Troubleshooting

- Writes on master are slower
- How many ”Ack”-s master waits?
  - Before 5.7: from single slave
  - Now in MySQL:
    - `rpl_semi_sync_master_wait_for_slave_count`
  - Won’t wait others
- What happens in case of timeout?
- What does ”Ack” mean?
Semisynchronous: Troubleshooting

- Writes on master are slower
- How many "Ack"-s master waits?
- What happens in case of timeout?
Semisynchronous: Troubleshooting

- Writes on master are slower
- How many "Ack"-s master waits?
- What happens in case of timeout?
  - Replication becomes asynchronous
Semisynchronous: Troubleshooting

- Writes on master are slower
- How many "Ack"-s master waits?
- What happens in case of timeout?
- What does "Ack" mean?
Semisynchronous: Troubleshooting

- Writes on master are slower
- How many "Ack"-s master waits?
- What happens in case of timeout?
- What does "Ack" mean?
  - Event written into relay log
Semisynchronous: Troubleshooting

- Writes on master are slower
- How many "Ack"-s master waits?
- What happens in case of timeout?
- What does "Ack" mean?
  - Event written into relay log
  - It is unknown if event applied
Two Kinds of Slave Threads

IO thread
Reads from the master

SQL thread
Two Kinds of Slave Threads

IO thread
Reads from the master
Stores in the relay log

SQL thread
Two Kinds of Slave Threads

IO thread
Reads from the master
Stores in the relay log

SQL thread
← Reads from relay log
Two Kinds of Slave Threads

IO thread
- Reads from the master
- Stores in the relay log

SQL thread
- Reads from relay log
- Executes

←
Single SQL Thread

- Easier for troubleshooting
Single SQL Thread

- Easier for troubleshooting
- Slower than master
  - High parallel load
Writes on the Master
Writes on the Slave: Single SQL Thread
Multiple SQL Threads: 5.6+
Performance Tuning

- **MySQL:** `--slave_parallel_workers`
- **MySQL:** `--slave_parallel_type=DATABASE | LOGICAL_CLOCK`
- **MySQL 8.0.1+:**
  
  ```
  SET @@GLOBAL.binlog_transaction_dependency_tracking = WRITESET | WRITESET_SESSION | COMMIT_ORDER;
  ```
Performance Tuning

- **MariaDB:** `--slave_parallel_threads`
- **MariaDB:** `--slave_parallel_maxqueued`
- **MariaDB:** `--slave_domain_parallel_threads`
- **MariaDB:** `--slave_parallel_mode=optimistic | conservative | aggressive | minimal | none`
mysql> select WORKER_ID, SERVICE_STATE, LAST_SEEN_TRANSACTION, LAST_ERROR_NUMBER, 
   -> LAST_ERROR_MESSAGE from performance_schema.replication_applier_status_by_worker\G

*************************** 1. row ***************************
WORKER_ID: 1
SERVICE_STATE: OFF
LAST_SEEN_TRANSACTION: d318bc17-66dc-11e6-a471-30b5c2208a0f:4988
LAST_ERROR_NUMBER: 0
LAST_ERROR_MESSAGE: 
*************************** 2. row ***************************
WORKER_ID: 3
SERVICE_STATE: OFF
LAST_SEEN_TRANSACTION: d318bc17-66dc-11e6-a471-30b5c2208a0f:4986
LAST_ERROR_NUMBER: 1032
LAST_ERROR_MESSAGE: Worker 2 failed executing transaction...
### #6: Error of One Thread Stops All

```
MariaDB [test]> select id, command, time, state from information_schema.processlist
  -> where user='system user';
```

<table>
<thead>
<tr>
<th>id</th>
<th>command</th>
<th>time</th>
<th>state</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Connect</td>
<td>4738</td>
<td>Waiting for master to send event</td>
</tr>
<tr>
<td>24</td>
<td>Connect</td>
<td>5096</td>
<td>Slave has read all relay log; waiting for the slave I/O thread t</td>
</tr>
<tr>
<td>23</td>
<td>Connect</td>
<td>0</td>
<td>Waiting for work from SQL thread</td>
</tr>
<tr>
<td>22</td>
<td>Connect</td>
<td>0</td>
<td>Unlocking tables</td>
</tr>
<tr>
<td>21</td>
<td>Connect</td>
<td>0</td>
<td>Update_rows_log_event::ha_update_row(-1)</td>
</tr>
<tr>
<td>20</td>
<td>Connect</td>
<td>0</td>
<td>Waiting for prior transaction to start commit before starting ne</td>
</tr>
<tr>
<td>19</td>
<td>Connect</td>
<td>0</td>
<td>Update_rows_log_event::ha_update_row(-1)</td>
</tr>
<tr>
<td>18</td>
<td>Connect</td>
<td>0</td>
<td>Update_rows_log_event::ha_update_row(-1)</td>
</tr>
<tr>
<td>17</td>
<td>Connect</td>
<td>0</td>
<td>Update_rows_log_event::find_row(-1)</td>
</tr>
</tbody>
</table>
```

...
Which Kind of Errors?

- Different data
  - Slave cannot apply event from relay log
Which Kind of Errors?

- Different data
  - Slave cannot apply event from relay log
- Different errors on master and slave
  - Triggers
  - Transactional and non-transactional tables in the same transaction
Different Data on Master and Slave

- Did table change outside of the replication?
  - How?
  - Can it cause conflict with changes on the master?
Different Data on Master and Slave

- Did table change outside of the replication?
- Are table structures identical?
  - Percona Toolkit
    - `pt-table-checksum`, `pt-table-sync`
  - MySQL Utilities
    - `mysqlrplsync`, `mysqldbcompare`, `mysqldiff`
Different Data on Master and Slave

- Did table change outside of the replication?
- Are table structures identical?
- Are changes in the correct order?
  - `mysqlbinlog`
  - Application logic on the master
Master
Recieves a change →

Storage Engine
Writes into binary log →
Synchronizes

Logical
Master
Recieves a change
Sends to SE →

Storage Engine
Logical

Master
Recieves a change
Sends to SE →

Storage Engine
Writes into table
Master
Recieves a change
Sends to SE →

Storage Engine
 Writes into table
Returns control
Master
Recieves a change
Sends to SE →

Storage Engine
 Writes into table
← Returns control

Writes into binary log
Logical

Master
Recieves a change
Sends to SE →

Storage Engine
Writes into table
← Returns control

Writes into binary log
Synchronizes →

← Synchronizes
Master Performance

- More writes
  - RBR: --binlog_row_image

Do not disable! You may set it greater than 1.
More writes

- **RBR:** `--binlog_row_image`
- `--binlog_cache_size`
  - Watch `Binlog.cache_disk_use`

Master Performance
More writes

- RBR: `--binlog_row_image`
- `--binlog_cache_size`
  - Watch `Binlog_cache_disk_use`
- `--binlog_stmt_cache_size`
  - Watch `Binlog_stmt_cache_disk_use`
Master Performance

- More writes
- Synchronization
  - `--binlog_sync`
  - Do not disable!
  - You may set it greater than 1
Master Behavior

• Binary log lifetime
  • --expire_log_days
Master Behavior

- Binary log lifetime
- Synchronization
  - SBR is not safe with **READ COMMITTED** and **READ UNCOMMITTED**
Master Behavior

- Binary log lifetime
- Synchronization
- Order of records in the binary log
  - Non-deterministic events and SBR
Statement-Based Binary Log Format

Client

Binary log
Statement-Based Binary Log Format

Client

INSERT INTO ... ➔

Binary log
Statement-Based Binary Log Format

Client
INSERT INTO ... →

Binary log
SET TIMESTAMP...

Statement-Based Binary Log Format
Statement-Based Binary Log Format

Client

`INSERT INTO ...` →

Binary log

`SET TIMESTAMP ...`

`SET sql_mode ...`
Statement-Based Binary Log Format

Client
INSERT INTO ... →

Binary log
SET TIMESTAM...  
SET sql_mode...  
INSERT INTO ...
SBR: Strong Sides

- Exists since very first versions
SBR: Strong Sides

- Exists since very first versions
- Table definitions on master and slave can significantly vary
SBR: Strong Sides

- Exists since very first versions
- Table definitions on master and slave can significantly vary
- Usually less writes
  - There are exceptions!
Exists since very first versions
Table definitions on master and slave can significantly vary
Usually less writes
Human readable
SBR: Strong Sides

- Exists since very first versions
- Table definitions on master and slave can significantly vary
- Usually less writes
- Human readable
- Easy to troubleshoot
### #7: SHOW BINLOG EVENTS

**Query:**
```sql
mysql> SHOW BINLOG EVENTS IN 'mysql-bin.000316' FROM 422;
```

<table>
<thead>
<tr>
<th>Log_name</th>
<th>Pos</th>
<th>Event_type</th>
<th>Server_id</th>
<th>End_log_pos</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>mysql-bin.000316</td>
<td>422</td>
<td>Query</td>
<td>1456667904</td>
<td>509</td>
<td>BEGIN</td>
</tr>
<tr>
<td>mysql-bin.000316</td>
<td>509</td>
<td>Query</td>
<td>1456667904</td>
<td>609</td>
<td>use 'PgDay'; update ai set f1=1</td>
</tr>
<tr>
<td>mysql-bin.000316</td>
<td>609</td>
<td>Xid</td>
<td>1456667904</td>
<td>640</td>
<td>COMMIT /* xid=60328 */</td>
</tr>
</tbody>
</table>

3 rows in set (0,12 sec)
SBR: Weak Sides

- Not all queries are safe
  - Non-deterministic functions
  - MySQL extentions
  - Triggers
  - Mix with non-transactional tables
  - Temporary tables
SBR: Weak Sides

- Not all queries are safe
- Order of events matter!
  - Row-based locks
SBR: Weak Sides

- Not all queries are safe
- Order of events matter!
  - Row-based locks
  - Triggers
    - SET GLOBAL slave_skip_counter – No GTIDs!
    - Skip transaction – GTIDs
    - Synchronize tables!
Row-Based Binary Log Format

Client → Binary log

Row before changes
Row with changes
Row-Based Binary Log Format

Client
UPDATE ... →

Binary log
Row-Based Binary Log Format

Client
UPDATE ... →

Binary log
SET TIMESTAMP...

Row-Based Binary Log Format
Row-Based Binary Log Format

Client UPDATE →

Binary log

SET TIMESTAMP...
SET sql_mode...
Row-Based Binary Log Format

Client
UPDATE ... →

Binary log
SET TIMESTAMP...
SET sql_mode...
Row before changes
Row-Based Binary Log Format

Client
UPDATE ...

Binary log
SET TIMESTAMP ...
SET sql_mode ...
Row before changes
Row with changes
RBR: Strong Sides

- Safe
  - You do not need to care about
    - Order of events
    - Triggers
    - Functions
    - Which queries you send to master
RBR: Weak Sidex

- Sensitive for table structures
- More writes
  - `--binlog_row_image=FULL | MINIMAL | NOBLOB`
- Harder to read
$ mysqlbinlog ./mysqld.1/data/master-bin.000001 --start-position=989 --stop-position=1213
...
# at 1167
#160822 14:15:11 server id 1  end_log_pos 1213 CRC32 0x1f346c6b
Update_rows: table id 109 flags: STMT_END_F

BINLOG 'v966VxMBAAAAKwAAAI8EAAAAAG0AAAAAAAEEAAm0yAAJ0MQABAwABY2H0oQ==v966Vx8BAAAALgAAAL0EAAAAAG0AAAAAAAEEAAgAB///+BQAAAP4GAAAAa2w0Hw==') /* added by mysqlbinlog */ !*/;
ROLLBACK /* added by mysqlbinlog */ /*!*/;
SET @@SESSION.GTID_NEXT= 'AUTOMATIC' /* added by mysqlbinlog */ /*!*/;
...
$ mysqlbinlog -v ./mysqld.1/data/master-bin.000001 --start-position=989 --stop-position=1213
...
# at 1167
#160822 14:15:11 server id 1 end_log_pos 1213 CRC32 0x1f346c6b
Update_rows: table id 109 flags: STMT_END_F

BINLOG 'v966VxMBAAAAKwAAAI8EAAAAAG0AAAAAAAEAAm0yAAJ0MQABAwABY2HOoQ==
v966Vx8BAAAALgAAL0EAAAAAG0AAAAAAAEAAgAB///+BQAAAP4GAAAAa2w0Hw==
'/*!*/;
### UPDATE 'm2'.'t1'
### WHERE
### @1=5
### SET
### @1=6
ROLLBACK /* added by mysqlbinlog */ /*!*/;
SET @@SESSION.GTID_NEXT= 'AUTOMATIC' /* added by mysqlbinlog */ /*!*/;
Position-Based

- You must specify
  - Name of the master’s binary log file
  - Position
Position-Based

- You must specify
  - Name of the master’s binary log file
  - Position
- From the troubleshooting point of view
  - Event executes if on the current position
Position-Based

- You must specify
  - Name of the master’s binary log file
  - Position
- From the troubleshooting point of view
  - Event executes if on the current position
  - Easy to skip
Position-Based

- You must specify
  - Name of the master’s binary log file
  - Position

- From the troubleshooting point of view
  - Event executes if on the current position
  - Easy to skip
  - Easy to move position backward
Position-Based

- You must specify
  - Name of the master’s binary log file
  - Position

- From the troubleshooting point of view
  - Event executes if on the current position
  - Easy to skip
  - Easy to move position backward
  - No conflict resolution
Global Transaction Identifiers (GTID)

- Each transaction has unique number: GTID
Global Transaction Identifiers (GTID)

- Each transaction has unique number: GTID
- MySQL: AUTO_POSITION=1
Global Transaction Identifiers (GTID)

- Each transaction has unique number: GTID
- MySQL: `AUTO_POSITION=1`
- MariaDB: `master_use_gtid = { slave_pos | current_pos }`
Global Transaction Identifiers (GTID)

- Each transaction has unique number: GTID
- MySQL: AUTO_POSITION=1
- MariaDB: `master_use_gtid = { slave_pos | current_pos }`
- No need to specify binary log and position
Global Transaction Identifiers (GTID)

- Each transaction has unique number: GTID
- **MySQL**: AUTO_POSITION=1
- **MariaDB**: master_use_gtid = { slave_pos | current_pos }
- No need to specify binary log and position
- Hard to skip erroneous event
sveta@thinkie> mysqlslavetrx --gtid-set=fb776095-8474-11e5-ad41-30b5c2208a0f:3 \ --slaves=root:@127.0.0.1:13001

WARNING: Using a password on the command line interface can be insecure.
#
# GTID set to be skipped for each server:
# - 127.0.0.1@13001: fb776095-8474-11e5-ad41-30b5c2208a0f:3
#
# Injecting empty transactions for ’127.0.0.1:13001’...
#
#...done.
#
Complicated Setups

• Same specifics and methods
Complicated Setups

- Same specifics and methods
- Multiply complexity on number of channels
Complicated Setups

• Same specifics and methods
• Multiply complexity on number of channels
• Control writes
  • What
  • On which server
Master and Slave

Master
- Binary log
- Binlog Dump

Slave
- Relay log
- IO thread
- SQL thread
Master and Two Slaves

Slave 1
- Relay log
- IO thread
- SQL thread

Slave 2
- Relay log
- IO thread
- SQL thread
Multiple Masters (Multi-channel): 5.7+

- **Master 1**
  - Binary log
  - Binlog Dump
  - IO thread
  - SQL thread

- **Slave**
  - Relay log
  - IO thread
  - SQL thread

- **Master 2**
  - Binary log
  - Binlog Dump
Multi-Master: Troubleshooting

- Multiple sets of relay logs
- Multiple IO threads
- Multiple SQL threads
- **MySQL**: `--slave_parallel_workers for each`
Multi-Master: Troubleshooting

- Multiple sets of relay logs
- Multiple IO threads
- Multiple SQL threads
- **MySQL**: `--slave.parallel_workers` for each
- Independent channels
Multi-Master: Troubleshooting

- Multiple sets of relay logs
- Multiple IO threads
- Multiple SQL threads

MySQL: `--slave_parallel_workers` for each

- Independent channels
- Error in one stops only one
Multi-Master: Troubleshooting

- Multiple sets of relay logs
- Multiple IO threads
- Multiple SQL threads
- **MySQL:** `--slave.parallel_workers` for each
- Independent channels
- Error in one stops only one
- No automatic conflict resolution
Basic Tools

- Error log file
Basic Tools

- Error log file
- On the slave
  - SHOW SLAVE STATUS
  - MySQL: Tables in Performance Schema
  - System database mysql
Basic Tools

- Error log file
- On the slave
- On the master
  - `SHOW MASTER STATUS`
  - `SHOW BINLOG EVENTS`
  - `mysqlbinlog`
Basic Tools

- Error log file
- On the slave
- On the master
- Percona Toolkit
Basic Tools

- Error log file
- On the slave
- On the master
- Percona Toolkit
- MySQL Utilities
Replication Must Know

- Always available, requires setup
- Asynchronous
- Master
  - Keeps all changes in the binary log
    - Two formats: ROW and STATEMENT
- Slave
  - IO thread reads from the master into relay log
  - SQL thread executes updates
    - Multiple SQL threads in 5.6+
    - Multiple channels/sources (masters) in 5.7+
- GTID in 5.6+
Typical Issues

- **Master**
  - Same as for standalone server
  - More writes and consistency checks
Typical Issues

- Master
- Slave IO thread
  - Common network issues
  - mysql command line client for tests
Typical Issues

- Master
- Slave IO thread
- Slave SQL thread
  - Regular query-related issues
  - Regular storage engine issues
  - Less execution threads than on master
More Information

- Basic Techniques – troubleshooting webinar
- Troubleshooting hardware resource usage
- Introduction into storage engine troubleshooting...
- Percona Toolkit
- MySQL Utilities
- Book MySQL High Availability
- MySQL Replication Team blog
Contact Information

http://www.slideshare.net/SvetaSmirnova
https://twitter.com/svetsmirnova
https://github.com/svetasmirnova
Support Team at Percona Live

- ... Case Study: .IE Continuous Restore ...
  Marcelo Altmann - Percona, Mick Begley - IE Domain Registry
  Tuesday 2:20PM-3:10PM @ Goldsmith 3

- A ... Walkthrough on pt-stalk
  Marcelo Altmann - Percona, Marcos Albe - Percona
  Wednesday 3:20PM-4:10PM @ Field Suite 2

- Percona XtraBackup Best Practices
  Marcelo Altmann - Percona
  Wednesday 4:30PM-4:55PM @ Sky Suite 2

- MySQL-MongoDB-Redis-Cassandra-HBase
  Marcos Albe - Percona
  Wednesday 4.55PM-5.20PM @ Sky Suite 2
Thank you!