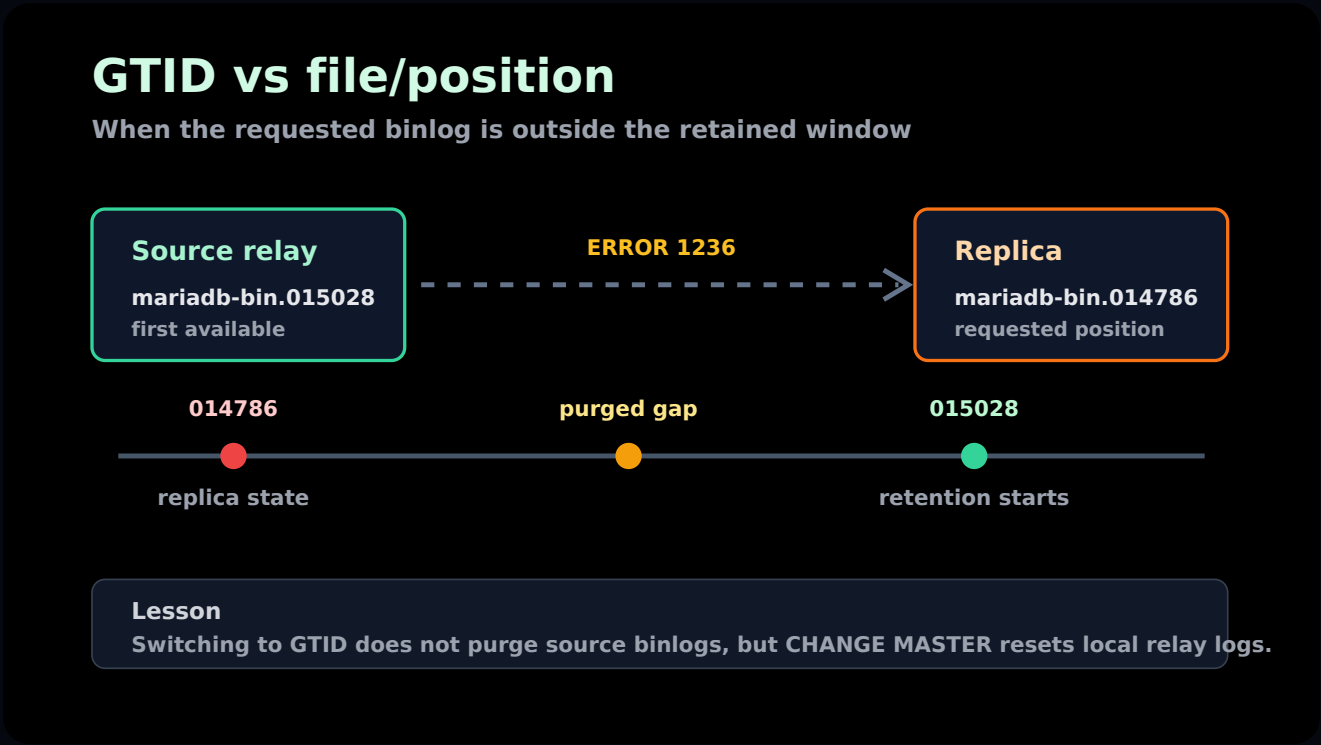
Когда переход на GTID ломает репликацию

Aurélien LEQUOY · May 18, 2026

MARIADB MYSQL REPLICATION GTID BINLOG PMACONTROL DBA INCIDENT



Симптом

Сценарий знакомый: репликация MariaDB работает в режиме file/position, GTID включают для теста или для более надежного подключения, затем конфигурацию возвращают в обычный режим. Через несколько секунд IO thread отказывается запускаться:

```
SHOW SLAVE STATUS\G
```

```
Slave_IO_Running: No
Slave_SQL_Running: Yes
Using_Gtid: No
Last_IO_Errno: 1236
Last_IO_Error: Could not find first log file name in binary log index file
```

Первая мысль понятна: "переход на GTID удалил relay logs или binlogs". В разобранном инциденте это было не так.

Что на самом деле делает кнопка GTID

В PmaControl действие `GTID > Activate` для репликации MariaDB не запускает purge. Оно также не выполняет `RESET SLAVE`.

Оно делает только это:

```
STOP SLAVE;
CHANGE MASTER TO MASTER_USE_GTID = slave_pos;
START SLAVE;
```

Действие отката выполняет:

```
STOP SLAVE;
CHANGE MASTER TO MASTER_USE_GTID = no;
START SLAVE;
```

Эти команды не удаляют binlogs на источнике. Но `CHANGE MASTER` пересоздает локальное состояние репликации и сбрасывает relay logs реплики. Это важная разница: источник ничего не теряет, но реплика выбрасывает то, что уже скачала локально.

Полномасштабный тест

Чтобы проверить поведение, я воспроизвел случай на двух лабораторных серверах MariaDB 11.8:

```
source -> replica
в начале file/position
relay_log_purge=ON
```

Тест:

1. запустить чистую репликацию file/position;
2. остановить только SQL thread на реплике;
3. сгенерировать 20 000 строк на источнике;

4. убедиться, что IO thread скачивает события в relay logs;
5. выполнить ту же последовательность, что и кнопка `GTID > Activate`.

До `CHANGE MASTER` на реплике были непримененные relay logs:

```
Slave_IO_Running: Yes
Slave_SQL_Running: No
Read_Master_Log_Pos: 635180
Exec_Master_Log_Pos: 3464
Relay_Log_Space: 632576
relay-bin.000002: 632273 bytes
```

После `STOP SLAVE` ничего не было потеряно:

```
Slave_IO_Running: No
Slave_SQL_Running: No
Relay_Log_Space: 632576
relay-bin.000002: 632273 bytes
```

Сразу после:

```
CHANGE MASTER TO MASTER_USE_GTID=slave_pos;
```

relay logs были сброшены:

```
Using_Gtid: Slave_Pos
Read_Master_Log_Pos: 3464
Exec_Master_Log_Pos: 3464
Relay_Log_Space: 256
relay-bin.000001: 256 bytes
```

Обратный тест дает тот же результат. В режиме GTID, при наличии непримененных relay logs, возврат к file/position через:

```
CHANGE MASTER TO MASTER_USE_GTID=no;
```

тоже сбрасывает локальные relay logs.

Вывод теста: один `STOP SLAVE` сохраняет relay logs. `CHANGE MASTER`, даже если он ограничен `MASTER_USE_GTID`, сбрасывает их.

Что произошло на самом деле

Реплика вернулась в режим file/position со слишком старой сохраненной координатой:

```
mariadb-bin.014786:35578691
```

Relay source уже не имел этого файла в индексе binary logs. Первый доступный файл был уже новее:

```
first available: mariadb-bin.015028
last available:  mariadb-bin.015130
retained files:  103
retained size:   about 10 GiB
```

При этом источник был настроен так:

```
binlog_expire_logs_seconds = 864000
expire_logs_days           = 10
binlog_space_limit         = 0
max_binlog_size            = 104857600
```

То есть текущая конфигурация говорила "10 дней". Но координата, которую запросила реплика, была старше фактически доступного окна.

Почему 10 дней не гарантируют этот файл

Хранение 10 дней означает, что сервер может удалить binlogs старше этого лимита. Это не гарантирует, что реплика сможет вернуться к любой старой физической координате после смены режима.

Есть несколько опасных случаев:

- репликация уже отставала до клика;
- реплика сохранила старую координату file/position во время работы в GTID;
- предыдущий purge уже удалил нужные файлы;
- значение retention могло меняться раньше;
- maintenance или rebuild могли пересоздать более короткое окно;
- upstream master может иметь другую политику, чем relay source.

Главное просто: MariaDB не может отдать файл, которого нет в `SHOW BINARY LOGS`, даже если текущая конфигурация показывает 10 дней.

Почему GTID скрыл проблему

В режиме GTID MariaDB реплика больше не запрашивает пару `MASTER_LOG_FILE` / `MASTER_LOG_POS`. Она запрашивает логический набор транзакций через `gtid_slave_pos`.

Когда реплика возвращается в file/position, старые физические координаты снова становятся важными. Если они указывают на удаленный файл, IO thread сразу падает с ошибкой `1236`.

GTID не очистил binlogs источника. Однако переключение через `CHANGE MASTER` удалило локальные relay logs, уже скачанные репликой. Если транзакции, находившиеся в этих relay logs, больше недоступны в binlogs источника, восстановление невозможно без ресинхронизации.

Почему простого CHANGE MASTER недостаточно

Можно попытаться переставить реплику на текущую позицию источника:

```
STOP SLAVE;
CHANGE MASTER TO
  MASTER_LOG_FILE='mariadb-bin.015130',
  MASTER_LOG_POS=97211185,
  MASTER_USE_GTID=no;
START SLAVE;
```

IO thread может запуститься. Но SQL thread затем может упасть на ограничении, например:

```
Last_SQL_Errno: 1452
Cannot add or update a child row:
foreign key constraint fails
```

Это ожидаемо: прямой переход к свежей позиции пропускает диапазон транзакций. Данные реплики больше не соответствуют следующей последовательности событий.

Это не ремонт. Это прыжок по журналу.

Правильное исправление

Надежное исправление - полная ресинхронизация реплики из согласованного источника:

1. остановить репликацию;
2. сделать согласованный backup с source или relay source;
3. восстановить реплику;
4. получить точную позицию backup;
5. настроить репликацию в GTID или file/position;
6. запустить репликацию;
7. убедиться, что оба thread здоровы.

Ожидаемый результат:

```
Slave_IO_Running: Yes
Slave_SQL_Running: Yes
Seconds_Behind_Master: 0 or decreasing
Last_IO_Error:
Last_SQL_Error:
```

В таком случае не надо использовать `sql_slave_skip_counter`. Пропуск FK ошибок скрывает расхождение и оставляет неконсистентные данные.

Проверка перед кликом

Перед включением или выключением GTID PmaControl должен ответить на три вопроса:

```
SHOW SLAVE STATUS\G
SHOW BINARY LOGS;
SHOW VARIABLES WHERE Variable_name IN (
  'expire_logs_days',
  'binlog_expire_logs_seconds',
  'binlog_space_limit',
  'max_binlog_size'
);
```

Затем:

- существует ли `Relay_Master_Log_File` реплики на источнике?

- реплика уже отстает или находится в ошибке?
- покрывает ли окно binlogs запрошенную позицию?

Если ответ отрицательный, кнопка не должна просто отправлять `CHANGE MASTER`. Она должна показать предупреждение и предложить rebuild.

Вывод

Переход на GTID не удалил binlogs источника. Но `CHANGE MASTER`, использованный для перехода в GTID и последующего возврата в file/position, сбросил локальные relay logs реплики.

В здоровой среде реплика скачивает их заново с источника. В разобранном инциденте запрошенная координата уже была вне окна binlogs источника. После удаления локальных relay logs больше не осталось источника, из которого можно было бы воспроизвести пропущенный диапазон.

Операционный урок прямой: перед переключением между GTID и file/position нужно проверять окно binlogs на источнике. Иначе обратимый тест превращается в обязательную ресинхронизацию.