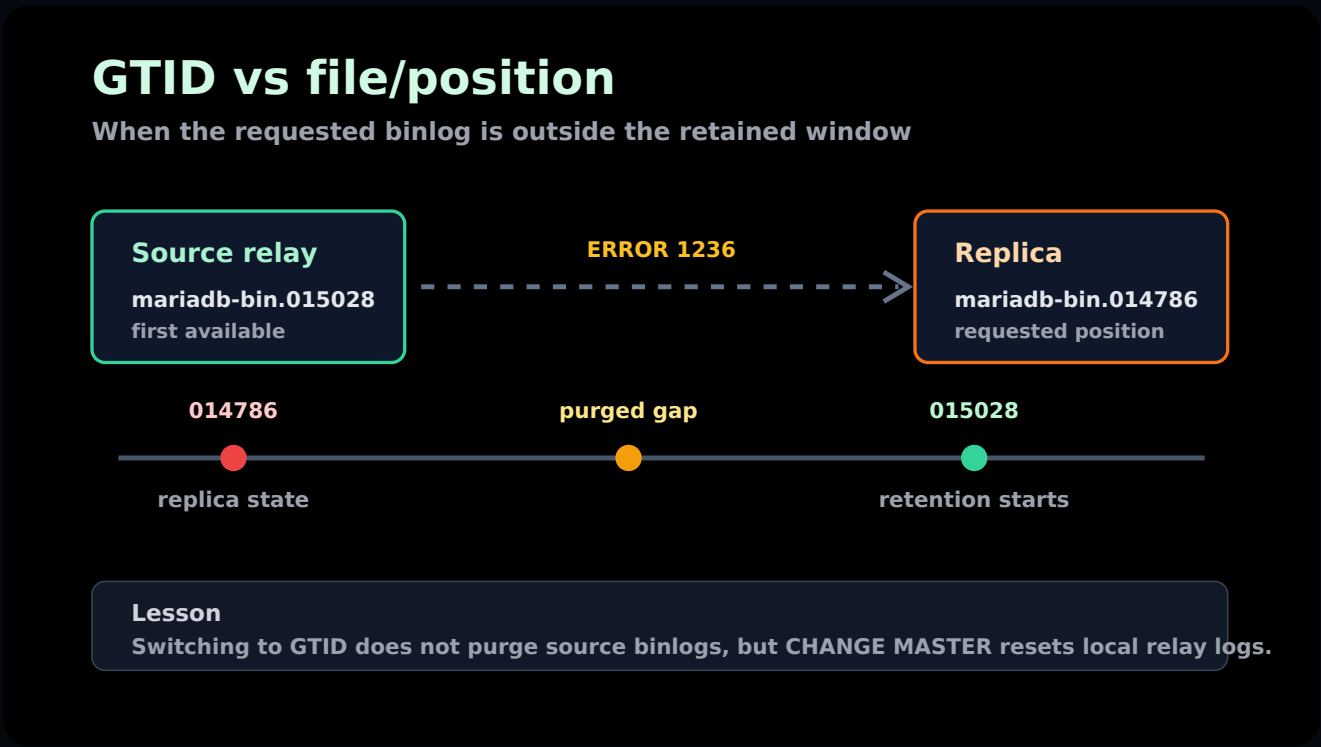
Gdy przejście na GTID psuje replikację

Aurélien LEQUOY · May 18, 2026

MARIADB MYSQL REPLICATION GTID BINLOG PMACONTROL DBA INCIDENT



Objaw

Scenariusz jest znany: replikacja MariaDB działa w trybie file/position, GTID zostaje włączone, aby przetestować lub utwardzić sposób podłączenia repliki, a następnie konfiguracja wraca do trybu standardowego. Po chwili thread IO odmawia startu:

```
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
```

Pierwsza hipoteza jest kusząca: "przejście na GTID wyczyściło relay logi albo binlogi". W analizowanym incydencie tak nie było.

Co naprawdę robi przycisk GTID

W PmaControl akcja `GTID > Activate` dla replikacji MariaDB nie uruchamia purge. Nie wykonuje też `RESET SLAVE`.

Wykonuje tylko:

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

Akcja powrotu wykonuje:

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

Te polecenia nie usuwają binlogów ze źródła. Jednak `CHANGE MASTER` odtwarza lokalny stan replikacji i resetuje relay logi repliki. To ważna różnica: źródło niczego nie traci, ale replika wyrzuca to, co już pobrała lokalnie.

Test w pełnej skali

Aby potwierdzić zachowanie, odtworzyłem przypadek na dwóch laboratoryjnych serwerach MariaDB 11.8:

```
source -> replica
na początku file/position
relay_log_purge=ON
```

Test:

1. uruchomić czystą replikację file/position;
2. zatrzymać tylko SQL thread repliki;
3. wygenerować 20 000 wierszy na źródle;

4. sprawdzić, że IO thread pobiera zdarzenia do relay logów;
5. wykonać sekwencję identyczną jak przycisk `GTID > Activate`.

Przed `CHANGE MASTER` replika miała niezaaplikowane relay logi:

```
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
```

Po `STOP SLAVE` nic nie zostało utracone:

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

Bezpośrednio po:

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

relay logi zostały zresetowane:

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

Test odwrotny daje taki sam wynik. W trybie GTID, gdy istnieją niezaaplikowane relay logi, powrót do file/position przez:

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

również resetuje lokalne relay logi.

Wniosek z testu: samo `STOP SLAVE` zachowuje relay logi. `CHANGE MASTER`, nawet ograniczony do `MASTER_USE_GTID`, resetuje je.

Co naprawdę się stało

Replika wróciła do trybu file/position ze zbyt starą zapisaną pozycją:

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

Relay source nie miał już tego pliku w indeksie binlogów. Pierwszy dostępny wpis był już nowszy:

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

Źródło nadal miało konfigurację:

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

Konfiguracja mówiła więc "10 dni". Jednak pozycja, której żądała replika, była starsza niż faktycznie dostępne okno.

Dlaczego 10 dni nie gwarantuje tego pliku

Retencja 10 dni oznacza, że serwer może usuwać binlogi starsze niż ten limit. Nie gwarantuje, że replika może wrócić do dowolnej starej fizycznej pozycji po zmianie trybu.

Kilka sytuacji sprawia, że jest to niebezpieczne:

- replikacja miała opóźnienie już przed kliknięciem;
- replika zachowała starą koordynatę file/position podczas pracy w GTID;
- wcześniejsze purge usunęło już żądane pliki;
- wartość retencji mogła zostać zmieniona w przeszłości;
- maintenance albo rebuild mógł odtworzyć krótsze okno niż oczekiwano;
- upstream master może mieć inną politykę niż relay source.

Najważniejszy punkt jest prosty: MariaDB nie może podać pliku, którego nie ma w `SHOW BINARY LOGS`, nawet jeśli obecna konfiguracja pokazuje 10 dni.

Dlaczego GTID ukryło problem

W trybie GTID MariaDB replika nie prosi już o parę `MASTER_LOG_FILE` / `MASTER_LOG_POS`. Prosi o logiczny zestaw transakcji przez `gtid_slave_pos`.

Po powrocie do file/position stare fizyczne koordynaty znów stają się istotne. Jeżeli wskazują na usunięty plik, thread IO natychmiast kończy się błędem `1236`.

GTID nie wyczyściło binlogów źródła. Jednak przełączenie przez `CHANGE MASTER` usunęło lokalne relay logi już pobrane przez replikę. Jeżeli transakcje obecne w tych relay logach nie są już dostępne w binlogach źródła, odzyskanie stanu jest niemożliwe bez resynchronizacji.

Dlaczego prosty CHANGE MASTER nie wystarczy

Można spróbować przepiąć replikę na aktualną pozycję źródła:

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

Thread IO może ruszyć. Ale thread SQL może potem zatrzymać się na ograniczeniu, na przykład:

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

To jest logiczne: skok bezpośrednio do nowszej pozycji pomija zakres transakcji. Dane repliki nie odpowiadają już sekwencji zdarzeń, które replika otrzymuje później.

To nie jest naprawa. To jest skok w dzienniku.

Poprawna naprawa

Bezpieczna naprawa to pełna resynchronizacja repliki ze spójnego źródła:

1. zatrzymać replikację;
2. wykonać spójny backup ze źródła albo relay source;
3. odtworzyć replikę;
4. pobrać dokładną pozycję backupu;
5. skonfigurować replikację w trybie GTID albo file/position;
6. uruchomić replikację;
7. sprawdzić, że oba thready są zdrowe.

Oczekiwany wynik:

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

W tym przypadku należy unikać `sql_slave_skip_counter`. Pomijanie błędów FK ukrywa rozjazd danych i zostawia niespójną bazę.

Kontrola przed kliknięciem

Przed włączeniem lub wyłączeniem GTID PmaControl powinien umieć odpowiedzieć na trzy pytania:

```
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'
);
```

Następnie:

- czy `Relay_Master_Log_File` repliki nadal istnieje na źródle?

- czy replika ma już opóźnienie albo błąd?
- czy okno binlogów pokrywa żądaną pozycję?

Jeżeli odpowiedź brzmi nie, przycisk nie powinien po prostu wysyłać `CHANGE MASTER`. Powinien pokazać ostrzeżenie i zaproponować rebuild.

Wniosek

Przejście na GTID nie wyczyściło binlogów źródła. Ale `CHANGE MASTER` użyty do wejścia w GTID i późniejszego powrotu do file/position zresetował lokalne relay logi repliki.

W zdrowym środowisku replika pobiera je ponownie ze źródła. W analizowanym incydencie żądana koordynata była już poza oknem binlogów źródła. Po wyrzuceniu lokalnych relay logów nie zostało już żadne miejsce, z którego można było odtworzyć brakujący zakres.

Lekcja operacyjna jest jasna: przed przełączaniem między GTID i file/position trzeba sprawdzić okno binlogów źródła. Inaczej odwracalny test zamienia się w obowiązkową resynchronizację.