3.5 MySQL：从机故障重启后主从同步报错案例分析

作者：付祥

一. 环境说明

1. MySQL 版本

```
root@3306 (none)> SELECT @@VERSION;
+------------+
| @@VERSION |
+------------+
| 5.7.34-log |
+------------+
1 row in set (0.01 sec)
```

2. binlog 相关参数

```
root@3306 (none)> select @@log_bin,@@log_slave_updates;
+-----------+---------------------+
| @@log_bin | @@log_slave_updates |
+-----------+---------------------+
| 1 | 1 |
+-----------+---------------------+
1 row in set (0.00 sec)
root@3306 (none)>
```

3. GTID 相关参数

```
root@3306 (none)> select
@@binlog_gtid_simple_recovery,@@enforce_gtid_consistency,@@gtid_mode;
+--------------------------------+----------------------------+-------------+
| @@binlog_gtid_simple_recovery | @@enforce_gtid_consistency | @@gtid_mode |
+--------------------------------+----------------------------+-------------+
| 1 | ON | ON |
+--------------------------------+----------------------------+-------------+
1 row in set (0.01 sec)
root@3306 (none)>
```

4. 半同步相关参数

```
root@3306 (none)> show global variables like '%semi%';
```

```
+-----------------------------------------+------------+
| Variable_name | Value |
+-----------------------------------------+------------+
| rpl_semi_sync_master_enabled | ON |
| rpl_semi_sync_master_timeout | 1000 |
| rpl_semi_sync_master_trace_level | 32 |
| rpl_semi_sync_master_wait_for_slave_count | 1 |
| rpl_semi_sync_master_wait_no_slave | ON |
| rpl_semi_sync_master_wait_point | AFTER_SYNC |
| rpl_semi_sync_slave_enabled | ON |
| rpl_semi_sync_slave_trace_level | 32 |
+-----------------------------------------+------------+
8 rows in set (0.00 sec)
root@3306 (none)>
```

5. 多线程同步相关参数

```
root@3306 (none)> select
@@binlog_transaction_dependency_tracking,@@slave_parallel_type,@@slave_parallel_
workers;
+-----------------------------------------+----------------------+---------
-----------------+
| @@binlog_transaction_dependency_tracking | @@slave_parallel_type |
@@slave_parallel_workers |
+-----------------------------------------+----------------------+---------
-----------------+
| COMMIT_ORDER | LOGICAL_CLOCK |
2 |
+-----------------------------------------+----------------------+---------
-----------------+
1 row in set (0.00 sec)
root@3306 (none)>
```

二、故障现象

MySQL 从库所在主机故障重启后，sql_thread 线程报错：

```
root@3306 (none)> show slave status\G
-- 摘取有用信息如下:
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: 1062
Last_Error: Coordinator stopped because there were
error(s) in the worker(s). The most recent failure being: Worker 1 failed
executing transaction '471c2974-f9bb-11eb-afb1-52540010fb89:88313207' at master
log MySql-bin.000685, end_log_pos 1004756557. See error log and/or
performance_schema.replication_applier_status_by_worker table for more details
about this failure or others, if any.
......省略..................
Retrieved_Gtid_Set: 471c2974-f9bb-11eb-afb1-
52540010fb89:88313207-88341912
Executed_Gtid_Set: 471c2974-f9bb-11eb-afb1-52540010fb89:1-
88313206,
d4c228df-f9c6-11eb-a2d8-5254006f63b6:1-5
Auto_Position: 1
root@3306 (none)> select * from
performance_schema.replication_applier_status_by_worker\G
*************************** 1. row ***************************
CHANNEL_NAME:
WORKER_ID: 1
THREAD_ID: NULL
SERVICE_STATE: OFF
LAST_SEEN_TRANSACTION: 471c2974-f9bb-11eb-afb1-52540010fb89:88313207
LAST_ERROR_NUMBER: 1062
LAST_ERROR_MESSAGE: Worker 1 failed executing transaction '471c2974-f9bb-
11eb-afb1-52540010fb89:88313207' at master log MySql-bin.000685, end_log_pos
1004756557; Could not execute Write_rows event on table kefumobile.i_sms_proxy;
Duplicate entry '14765130' for key 'PRIMARY', Error_code: 1062; handler error
HA_ERR_FOUND_DUPP_KEY; the event's master log FIRST, end_log_pos 1004756557
LAST_ERROR_TIMESTAMP: 2022-01-24 23:05:02
*************************** 2. row ***************************
CHANNEL_NAME:
WORKER_ID: 2
THREAD_ID: NULL
SERVICE_STATE: OFF
LAST_SEEN_TRANSACTION:
LAST_ERROR_NUMBER: 0
LAST_ERROR_MESSAGE:
LAST_ERROR_TIMESTAMP: 0000-00-00 00:00:00
2 rows in set (0.00 sec)
```

通过报错信息可知，worker 线程在回放事务'471c2974-f9bb-11eb-afb1-52540010fb89:88313207'时，由于要插入的记录主键冲突报错。

三、故障分析

主机重启前，主从同步正常，主机重启后，主从同步由于主键冲突报错，对比了冲突主键所在行记录在主从库是一致的，初步分析事务'471c2974-f9bb-11eb-afb1-52540010fb89:88313207'在主机故障前已经在从库进行了回放，那为何事务会重复回放呢？

在开启 GTID 模式下，如果指定 master_auto_position=1，start slave 时，从库会把 Retrieved_Gtid_Set 和 Executed_Gtid_Set 的并集发送给主库，主库将收到的并集和自己的 gtid_executed 比较，把从库 GTID 集合里缺失的事务全都发送给从库。

主机重启后，事务重复回放，表明 Retrieved_Gtid_Set 和 Executed_Gtid_Set 的并集中有 GTID 事务丢失，导致重复获取事务执行引发主键冲突错误。Retrieved_Gtid_Set 和 Executed_Gtid_Set 均为内存变量，MySQL 重启后，Retrieved_Gtid_Set 初始化为空值，从而推断出 Executed_Gtid_Set 有 GTID 事务丢失。

Executed_Gtid_Set 来源于 gtid_executed 变量，gtid_executed 变量持久化介质有 mysql.gtid_executed 表和 binlog 日志，其中 mysql.gtid_executed 表是 MySQL 5.7 后引入的，在 MySQL 5.6 中，从库要使用 GTID ，必须要先设置 log_bin=on,log_slave_updates=on ，因为从库执行过的 GTID 只保留在 binlog 中。

1. 当 log_bin=on ，log_slave_updates=off 时，gtid_executed 变量的更新实时持久化到 mysql.gtid_executed 表中，MySQL 重启后 gtid_executed 变量值从 mysql.gtid_executed 表读取。
2. 当 log_bin=on ，log_slave_updates=on 时，只有在 binlog 切换时侯才会更新 mysql.gtid_executed 表，保存直到上一个 binlog 执行过的 GTID 集合。MySQL 重启后，在默认参数 binlog_gtid_simple_recovery=1 时，gtid_executed 变量值从最后一个 binlog 文件计算获得。

gtid_executed 变量值陈旧，推断出 binlog 未实时持久化，我们看一下参数 sync_binlog :

```
root@3306 (none)> show variables like 'sync_binlog';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| sync_binlog | 600 |
+---------------+-------+
1 row in set (0.00 sec)
```

通过以上分析，此次故障来龙去脉就清楚了：Worker 线程报 1062 主键冲突错误--> gtid_executed 信息陈旧--> binlog 未实时持久化

四、测试验证

搭建一主一从测试环境，通过 sysbench 模拟主库并发插入，从库主机暴力关机后，故障复现：

```
root@mysql.sock][(none)]> select * from
performance_schema.replication_applier_status_by_worker\G
```

```
*************************** 1. row ***************************
CHANNEL_NAME:
WORKER_ID: 1
THREAD_ID: NULL
SERVICE_STATE: OFF
LAST_SEEN_TRANSACTION: 4a0ad3da-b89e-11eb-9d0b-000c299b4d6c:452362
LAST_ERROR_NUMBER: 1062
LAST_ERROR_MESSAGE: Worker 1 failed executing transaction '4a0ad3da-b89e-
11eb-9d0b-000c299b4d6c:452362' at master log MySql-bin.000012, end_log_pos
1011339749; Could not execute Write_rows event on table sbtest.sbtest5;
Duplicate entry '111305' for key 'PRIMARY', Error_code: 1062; handler error
HA_ERR_FOUND_DUPP_KEY; the event's master log FIRST, end_log_pos 1011339749
LAST_ERROR_TIMESTAMP: 2022-01-26 09:56:38
*************************** 2. row ***************************
CHANNEL_NAME:
WORKER_ID: 2
THREAD_ID: NULL
SERVICE_STATE: OFF
LAST_SEEN_TRANSACTION:
LAST_ERROR_NUMBER: 0
LAST_ERROR_MESSAGE:
LAST_ERROR_TIMESTAMP: 0000-00-00 00:00:00
2 rows in set (0.00 sec)
[root@mysql.sock][(none)]>
```

五、故障处理

既然错误原因是事务重复执行，那跳过错误就好了，有如下两种方式，根据需要选取其中一种方式执行：

1. 通过空事务替代报错事务执行

```
set gtid_next='xxxxxx';
begin;
commit;
set gtid_next=AUTOMATIC;
start slave sql_thread
```

如果最新 binglog 丢失的 GTID 较多，手工执行比较繁琐，需要不断试错。可写一个存储过程批量执行：

```
set sql_log_bin=0;
DELIMITER $$
create procedure p_fx_gtid_next(i_master_Executed_Gtid_max varchar(100))
begin
declare v_gtid_next varchar(100);
```

```
declare master_Executed_Gtid_max varchar(100);
declare slave_Executed_Gtid_max varchar(100);
-- 主库当前执行了的gtid最大值，做为退出循环条件 show master status
set master_Executed_Gtid_max=i_master_Executed_Gtid_max;
loop_name:loop
SELECT ifnull(min(LAST_SEEN_TRANSACTION),'empty') FROM
performance_schema.replication_applier_status_by_worker WHERE
LAST_ERROR_NUMBER=1062
into v_gtid_next;
if v_gtid_next <> 'empty' then
set gtid_next = v_gtid_next;
start transaction;
commit;
set gtid_next =AUTOMATIC;
start slave sql_thread;
end if;
select max(LAST_SEEN_TRANSACTION) from
performance_schema.replication_applier_status_by_worker into
slave_Executed_Gtid_max;
if slave_Executed_Gtid_max = master_Executed_Gtid_max then
leave loop_name;
end if;
select SLEEP(1);
end loop;
end $$
DELIMITER ;
set sql_log_bin=1;
call p_fx_gtid_next('XXXXX:XXX');
```

2. 带参数 slave_skip_errors=1062 重启 MySQL

待主从同步正常后，再取消参数 slave_skip_errors 设置重启 MySQL 。