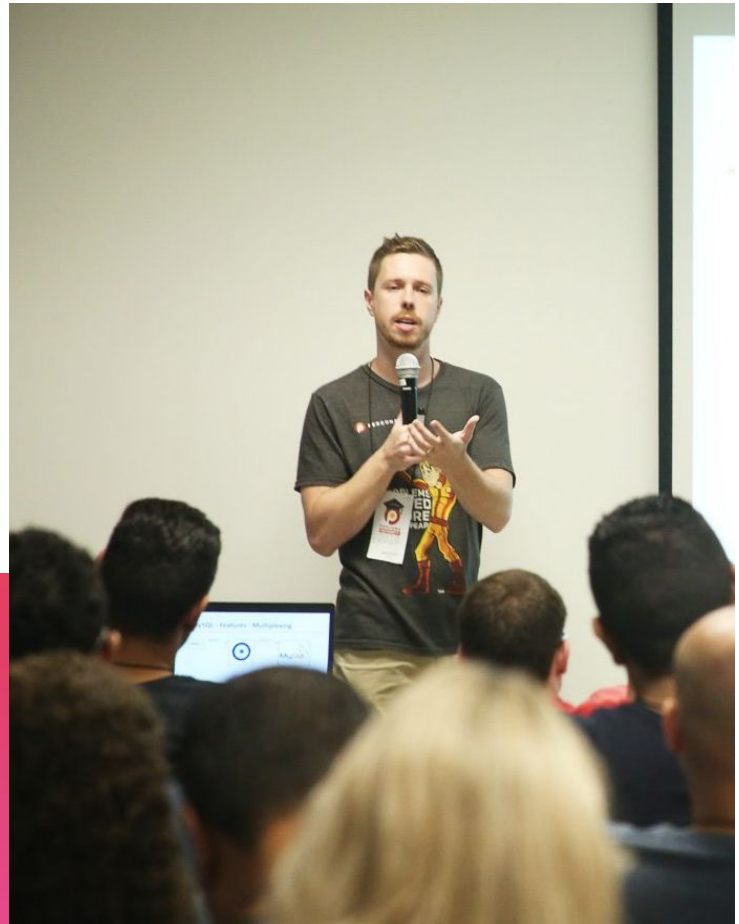
PERCONALIVE

Troubleshooting MySQL from a MySQL Developer Perspective

Marcelo Altmann
Software Developer - Percona
Percona Live Austin / May 2022

Marcelo Altmann

- Software Developer @ Percona
  - Joined in 2016 as Senior Support Engineer
  - PXB / PS / PXC
  - Member Bugs Escalation Committee



Software bug

Software bug

A software bug is an error, flaw or fault in computer software that causes it to produce an incorrect or unexpected result, or to behave in unintended ways - wikipidia

- Wrong query results
- Performance degradations
- Security flaw
- Crash

Reproducible test case

The history of a dev

PERCONALIVE

Reproducible test case

- Been able to reproduce a bug is key
- Projects have their own test framework
- Used to ensure we don't introduce regressions.
- MySQL MTR
  - Start mysql instance:
    ./mtr --start alias &
    mysql –defaults-file=var/my.cnf
  - Interactive GDB session:
    ./mtr --manual-gdb alias
    gdb -cd ./ -x ./var/tmp/gdbinit.mysqld.1 ../bin/mysqld

Reproducible test case

- Recompile with Debug synchronization :

```
open_tables(...)
DEBUG_SYNC(thd, "after_open_tables");
lock_tables(...)

--connection conn1
SET DEBUG_SYNC= 'after_open_tables SIGNAL opened WAIT_FOR flushed';
send INSERT INTO t1 VALUES(1);
--connection conn2
SET DEBUG_SYNC= 'now WAIT_FOR opened';
SET DEBUG_SYNC= 'after_abort_locks SIGNAL flushed';
FLUSH TABLE t1;
```

Reproducible test case

- Recompile with Test Faults Macros :

```
somefile.cc
DBUG_EXECUTE_IF("some_keyword", {
  some_function();
  some_variable++;
};);

mysql> SET debug = '+d,some_keyword';
```

Stack traces

Crash fingerprint

Stack Traces - Signals

- A way of software / kernel / user to communicate with a process.
- Program will interpret and handle it - mysqld.cc my_init_signals()

```
/* ISO C99 signals.  */
#define SIGINT          2       /* Interactive attention signal.  */
#define SIGILL          4       /* Illegal instruction.  */
#define SIGABRT         6       /* Abnormal termination.  */
#define SIGFPE          8       /* Erroneous arithmetic operation.  */
#define SIGSEGV         11      /* Invalid access to storage.  */
#define SIGTERM         15      /* Termination request.  */
```

Stack Traces - Signals

- SIGABRT / 6 - Code Assertions
  - ut_a(condition) / ut_ad(condition)
  - Critical points of code - continuing might cause damage
- SIGSEGV / 11 - Segmentation Fault
  - Memory access violation
  - Tried to access restricted memory area
- SIGTERM / 15 - Terminate
  - shutdown

Stack Traces

```
19:03:30 UTC - mysqld got signal 6 ;
Most likely, you have hit a bug, but this error can also be caused by malfunctioning hardware.
```
1

```
Build ID: ae15f57aa1c932e10b043d5f3e353ef09fe0e724
Server Version: 8.0.27-18-debug Percona Server (GPL), Revision 63a5835a303-Debug
```
2

```
Thread pointer: 0x7fe1f8001040
Attempting backtrace. You can use the following information to find out
where mysqld died. If you see no messages after this, something went
terribly wrong...
stack_bottom = 7fe2b02ebbd8 thread_stack 0x100000
mysqld(my_print_stacktrace(unsigned char const*, unsigned long)+0x55) [0x5579240abc6c]
mysqld(handle_fatal_signal+0x2e0) [0x557922e6e3ec]
...
mysqld(ut_dbg_assertion_failed(char const*, char const*, unsigned long)+0x1e3) [0x55792457bce8]
mysqld(lock_rec_convert_active_impl_to_expl( . . . ) [0x5579242ffa15]
mysqld(row_convert_impl_to_expl_if_needed(btr_cur_t*, undo_node_t*)+0x143) [0x55792449ede2]
mysqld(+0x5586364) [0x5579248b0364]
mysqld(+0x55866ba) [0x5579248b06ba]
mysqld(+0x5588323) [0x5579248b2323]
 . . .
```
3

```
Trying to get some variables.
Some pointers may be invalid and cause the dump to abort.
Query (7fe1f800a888): UPDATE t1 SET u=oo,v=ef WHERE id=10
Connection ID (thread ID): 11
Status: NOT_KILLED
```
4

Divide and conquer

Page Cleaner bug

Divide and conquer

- Goal is to establish when this started to happen
- Remove as much as unknown as you can
  - Issue is happening on latest version?
  - Issue is happening on latest version minus 1 / 2 / 3 / … versions?
  - Issue is happening on previous major version (8.0 -> 5.7) ?
- (PXB / PXC) - Is the issue specific to the product?

Divide and conquer

- PXB-2742 as example:

Percona XtraBackup / PXB-2742

PXB crashes during incremental backup prepare of partitioned tables: Assertion failure: buf0flu.cc:3567:UT_LIST_GET_LEN(buf_pool->flush_list) == 0

- Start Point - partitions -> full backup -> inc backup (one or many) -> prepare full -> prepare inc (one or many) -> crash (sometimes)
- partitions ? = full backup -> inc backup (one or many) -> prepare full -> prepare inc (one or many) -> crash (sometimes)

Divide and conquer

- Incrementals ? = full backup -> prepare full -> crash (sometimes)
- Investigate (prepare full):

  a. Innodb Master Thread Doing idle task
  b. Merging Insert Buffer at full io capacity
     - Ask the pages to be read in to BP (async IO) (IO_BUF_READ)
  c. IO Read Thread read the page
     - Merge Ibuf changes - Add page to Flush list
  d. Shutdown
  e. IO Read Thread complete the read/ibuf merge of page (IO_BUF_NONE)

Divide and conquer

- Can I reproduce the same on server?
- DBUG_EXECUTE_IF - shutdown - full ibuf merge
- Yes !!! 8.0 & 5.7 PS and upstream affected.
- PS-8174 / #107069 Crash -> private =(
- From a complex set of multiple variables (partition, xtrabackup, multiple incremental) to "simple" server bug.

Regression = git bisect

Wrong query result bug

Regression = git bisect

- PS-7019 / #99398
- Works ok on 8.0.19
- Does NOT work on 8.0.20

Regression = git bisect

```
mysql> SELECT * FROM t1;
+-------+-------+
| t1_id | t2_id |
+-------+-------+
|     1 |  1000 |
|     2 |     5 |
+-------+-------+
2 rows in set (0,00 sec)

mysql> SELECT * FROM t2;
+-------+-----------+
| t2_id | is_active |
+-------+-----------+
|     2 |         1 |
|     3 |         0 |
|  1000 |         1 |
+-------+-----------+
3 rows in set (0,00 sec)
```

PERCONALIVE

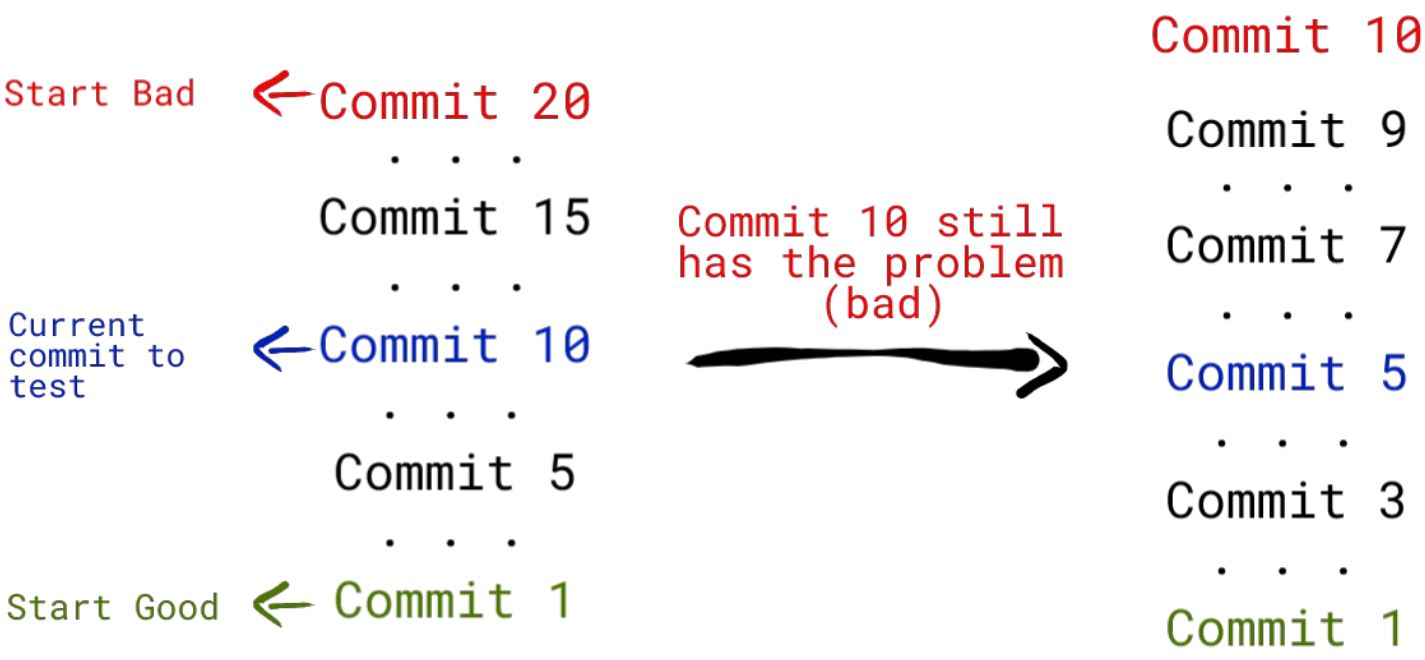
```
8.0.19> SELECT t1.*, t2.t2_id FROM t1 LEFT JOIN
t2 ON (t1.t2_id = t2.t2_id) GROUP BY t1_id;
+-------+-------+-------+
| t1_id | t2_id | t2_id |
+-------+-------+-------+
|     1 |  1000 |  1000 |
|     2 |     5 |  NULL |
+-------+-------+-------+
2 rows in set (0,00 sec)
```

```
8.0.20> SELECT t1.*, t2.t2_id FROM t1 LEFT JOIN
t2 ON (t1.t2_id = t2.t2_id) GROUP BY t1_id;
+-------+-------+-------+
| t1_id | t2_id | t2_id |
+-------+-------+-------+
|     1 |  1000 |  NULL |
|     2 |     5 |  NULL |
+-------+-------+-------+
2 rows in set (0,00 sec)
```

Regression = git bisect

- Finding a Regression in MySQL Source Code: A Case Study
- Lines (737+K):
```
git diff mysql-8.0.19..mysql-8.0.20 | wc -l
737454
```
- Files (~4.5K):
```
git diff mysql-8.0.19..mysql-8.0.20 --name-only | wc -l
4495
```
- Commits (~2K):
```
git log mysql-8.0.19..mysql-8.0.20 --pretty=oneline | wc -l
1966
```

Regression = git bisect



Regression = git bisect

Regression = git bisect

Regression = git bisect

Regression = git bisect

- Manual:
  a. `git bisect start mysql-8.0.20 mysql-8.0.19`
  b. `test`
  c. `git bisect [good | bad]`

Regression = git bisect

- Automated:

```
git bisect run sh -c '
compile_mysql.sh
if [ "$?" -ne "0" ]; then
  exit 125
fi
./mysql-test/mtr bisect
if [ "$?" -eq "0" ]; then
  exit 0
else
  exit 1
fi'
```

Regression = git bisect

- Bug#30460528: RENAME FIELD::REAL_MAYBE_NULL() TO FIELD::IS_NULLABLE()
- https://github.com/mysql/mysql-server/commit/3039fac3969f7c1521863bfe1513631986d2b6bd

Regression = git bisect

GDB / Coredump / PMP

Internal thread Deadlock bug

GDB

- GNU Debugger
- Works in various languages (C / C++ / Go / others)
- Can be used:
  a. Remote
  b. Live process - gdb -p PID
  c. Starting a process - gdb –args mysqld –datadir=.....
  d. Offline (coredump) - gdb bin/mysqld core.xxxxx

GDB

- Break point - stop the execution when a function is called.
- Condition break point - same as above but with condition (var1 == value)
- Watchpoints - same as break point but stop the execution when a variable is read / written / or both
- Next - execute the code until the next line.
- Continue - execute the code until the next break point.
- Step - enter the function.
- Bt - Mostrar Backtrace / Stack trace.
- Frame - pular para frame especídico dentro da backtrace.

Coredump

- Snapshot / Dump of process memory
- Used alongside with GDB + binary of process
- Allows to check variables when the snapshot was collected
- Normally collected when process crash
- Can be collected on demand by gcore or gdb generate-core-file (eg: mysqld is frozen and I can't get in)

Coredump

- On crash:
  a. Requires mysqld to be configured with –core-file and linux config:

```
echo 2 > /proc/sys/fs/suid_dumpable
mkdir /tmp/corefiles
chmod 777 /tmp/corefiles
echo "/tmp/corefiles/core" >
/proc/sys/kernel/core_pattern
echo "1" > /proc/sys/kernel/core_uses_pid
Mysqld --core-file
```

  b. PS - `mysqld --coredumper=/PATH`

Coredump

```
5024 Thread 0x7f37ed6ef700 (LWP 82840) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5025 Thread 0x7f37ed6a7700 (LWP 82841) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5026 Thread 0x7f37ed65f700 (LWP 82846) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5027 Thread 0x7f37ed617700 (LWP 82847) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5028 Thread 0x7f37ed5cf700 (LWP 82848) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5029 Thread 0x7f37ed587700 (LWP 82849) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5030 Thread 0x7f37ed53f700 (LWP 82850) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5031 Thread 0x7f37ed4f7700 (LWP 82851) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5032 Thread 0x7f37ed4af700 (LWP 82852) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5033 Thread 0x7f37ed467700 (LWP 82853) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5034 Thread 0x7f37ed41f700 (LWP 82854) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5035 Thread 0x7f37ed3d7700 (LWP 82855) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5036 Thread 0x7f37ed38f700 (LWP 82856) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5037 Thread 0x7f37ed347700 (LWP 82857) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
5038 Thread 0x7f37ed2ff700 (LWP 82858) __lll_lock_wait (futex=futex@entry=0x55d6b662d1d8, private=0) at lowlevellock.c:52
(gdb)
```

Poor Man's Profiler - PMP

- Aggregates identical stack traces
- Very useful for coredumps with high number of threads
- https://poormansprofiler.org/
- Available via percona toolkit

pt-pmp

```
   4996 __lll_lock_wait,__GI___pthread_mutex_lock,native_mutex_lock(thr_mutex.h:93),safe_mutex
_lock(thr_mutex.cc:117),my_mutex_lock(thr_mutex.h:180),inline_mysql_mutex_lock(mysql_mutex.h:2
50),plugin_thdvar_init(sql_plugin.cc:2832),THD::init(sql_class.cc:740)
     10 syscall,libaio::??(libaio.so.1),LinuxAIOHandler::collect(os0file.cc:2326),LinuxAIOHand
ler::poll(os0file.cc:2457),os_aio_linux_handler(os0file.cc:2506),os_aio_handler(os0file.cc:582
0),fil_aio_wait(fil0fil.cc:7394),io_handler_thread(srv0start.cc:267)
      3 futex_wait_cancelable,__pthread_cond_wait_common,__pthread_cond_wait,os_event::wait(os
0event.cc:179),os_event::wait_low(os0event.cc:337),os_event_wait_low(os0event.cc:513),srv_work
er_thread(srv0srv.cc:2668),void(srv0srv.cc:2668)
      2 futex_abstimed_wait_cancelable,__pthread_cond_wait_common,__pthread_cond_timedwait,nat
ive_cond_timedwait(thr_cond.h:100),safe_cond_timedwait(thr_cond.cc:113),my_cond_timedwait(thr_
cond.h:147),inline_mysql_cond_timedwait(mysql_cond.h:217),xpl::Cond::timed_wait(cond.cc:44)
      1 syscall,epoll_wait(epoll_sub.c:64),epoll_dispatch(epoll.c:462),event_base_loop(event.c
:1947),ngs::Socket_events::loop(socket_events.cc:173),ngs::Socket_acceptors_task::loop(socket_
acceptors_task.cc:206),ngs::Server::run_task(server.cc:99),ngs::Server::start(server.cc:133)
      1 futex_wait_cancelable,__pthread_cond_wait_common,__pthread_cond_wait,os_event::wait(os
0event.cc:179),os_event::wait_low(os0event.cc:337),os_event_wait_low(os0event.cc:513),srv_purg
e_coordinator_suspend(srv0srv.cc:2798),srv_purge_coordinator_thread(srv0srv.cc:2912)
```

GDB / Coredump / pt-pmp

```
(gdb) t 5038
[Switching to thread 5038 (Thread 0x7fae2c147700 (LWP 899658))]
#0  __lll_lock_wait (futex=futex@entry=0x55f9c9d363a8, private=0) at lowlevellock.c:52
52      in lowlevellock.c
(gdb) bt 9
#0  __lll_lock_wait (futex=futex@entry=0x55f9c9d363a8, private=0) at lowlevellock.c:52
#1  0x00007fb2f1840235 in __GI___pthread_mutex_lock (mutex=0x55f9c9d363a8) at ../nptl/pthread_
mutex_lock.c:135
#2  0x000055f9c6ebb60a in native_mutex_lock (mutex=0x55f9c9d363a8) at /work/mysql/src/include/
thr_mutex.h:93
#3  0x000055f9c6ebb875 in safe_mutex_lock (mp=0x55f9c9d36380, try_lock=false, file=0x55f9c7f38
d48 "/work/mysql/src/sql/sql_plugin.cc", line=2832) at /work/mysql/src/mysys/thr_mutex.cc:117
#4  0x000055f9c59d9e1b in my_mutex_lock (mp=0x55f9c9606140 <LOCK_global_system_variables>, fil
e=0x55f9c7f38d48 "/work/mysql/src/sql/sql_plugin.cc", line=2832) at /work/mysql/src/include/th
r_mutex.h:180
#5  0x000055f9c59d9fd8 in inline_mysql_mutex_lock (that=0x55f9c9606140 <LOCK_global_system_var
iables>, src_file=0x55f9c7f38d48 "/work/mysql/src/sql/sql_plugin.cc", src_line=2832) at /work/
mysql/src/include/mysql/psi/mysql_mutex.h:250
#6  0x000055f9c59e3271 in plugin_thdvar_init (thd=0x7fb088058630, enable_plugins=true) at /wor
k/mysql/src/sql/sql_plugin.cc:2832
#7  0x000055f9c591ddd9 in THD::init (this=0x7fb088058630) at /work/mysql/src/sql/sql_class.cc:
740
#8  0x000055f9c591cbc6 in THD::THD (this=0x7fb088058630, enable_plugins=true) at /work/mysql/s
rc/sql/sql_class.cc:507
(More stack frames follow...)
(gdb)
```

GDB / Coredump / pt-pmp

```
(gdb) f 2
#2  0x000055f9c6ebb60a in native_mutex_lock (mutex=0x55f9c9d363a8) at /work/mysql/src/include/
thr_mutex.h:93
93          return pthread_mutex_lock(mutex);
(gdb) p mutex->__data->__owner
$1 = 889019
(gdb) thread find 889019
Thread 38 has target id 'Thread 0x7fb2dc094700 (LWP 889019)'
```

GDB / Coredump / pt-pmp

- All 4996 threads wait on mutex from Thread 38

GDB / Coredump / pt-pmp

```
(gdb) t 38
[Switching to thread 38 (Thread 0x7fb2dc094700 (LWP 889019))]
#0  __lll_lock_wait (futex=futex@entry=0x55f9c9d36cc8, private=0) at lowlevellock.c:52
52      lowlevellock.c: No such file or directory.
(gdb) f 26
#26 0x000055f9c59a93d3 in do_command (thd=0x7fb22c00c880) at /work/mysql/src/sql/sql_parse.cc:
1260
1260      return_value = dispatch_command(thd, &com_data, command);
(gdb) p thd->m_query_string->str
$5 = 0x7fb22c0e9f88 "SELECT * FROM performance_schema.session_variables WHERE VARIABLE_NAME LI
KE 'binlog_transaction_dependency_tracking'"
(gdb) f 2
#2  0x000055f9c6ebb60a in native_mutex_lock (mutex=0x55f9c9d36cc8) at /work/mysql/src/include/
thr_mutex.h:93
93        return pthread_mutex_lock(mutex);
(gdb) p mutex->__data->__owner
$6 = 889099
(gdb) thread find 889099
Thread 44 has target id 'Thread 0x7fb2bc7b2700 (LWP 889099)'
(gdb)
```

GDB / Coredump / pt-pmp

- All 4996 threads are trying to connect and waiting on mutex from Thread 38
- Thread 38
  - SELECT * FROM performance_schema.session_variables WHERE VARIABLE_NAME LIKE 'binlog_transaction_dependency_tracking'
  - Wait on mutex from Thread 44

GDB / Coredump / pt-pmp

```
(gdb) t 44
[Switching to thread 44 (Thread 0x7fb2bc7b2700 (LWP 889099))]
#0  __lll_lock_wait (futex=futex@entry=0x55f9c9d37a88, private=0) at lowlevellock.c:52
52      in lowlevellock.c
(gdb) f 11
#11 0x000055f9c59a93d3 in do_command (thd=0x7fb214000e80) at /work/mysql/src/sql/sql_parse.cc:
1260
1260      return_value = dispatch_command(thd, &com_data, command);
(gdb) p thd->m_query_string->str
$7 = 0x7fb214009bf8 "SHOW BINARY LOGS"
(gdb) f 2
#2  0x000055f9c6ebb60a in native_mutex_lock (mutex=0x55f9c9d37a88) at /work/mysql/src/include/
thr_mutex.h:93
93        return pthread_mutex_lock(mutex);
(gdb) p mutex->__data->__owner
$8 = 889097
(gdb) thread find 889097
Thread 42 has target id 'Thread 0x7fb2c4052700 (LWP 889097)'
(gdb)
```

GDB / Coredump / pt-pmp

- All 4996 threads are trying to connect and waiting on mutex from Thread 38
- Thread 38
  - SELECT * FROM performance_schema.session_variables WHERE VARIABLE_NAME LIKE 'binlog_transaction_dependency_tracking'
  - Wait on mutex from Thread 44
- Thread 44
  - SHOW BINARY LOGS
  - Wait on mutex from Thread 42

GDB / Coredump / pt-pmp

```
(gdb) t 42
[Switching to thread 42 (Thread 0x7fb2c4052700 (LWP 889097))]
#0  __lll_lock_wait (futex=futex@entry=0x7fb22c0275b8, private=0) at lowlevellock.c:52
52      in lowlevellock.c
(gdb) f 18
#18 0x000055f9c59a93d3 in do_command (thd=0x7fb21c000e80) at /work/mysql/src/sql/sql_parse.cc:
1260
1260      return_value = dispatch_command(thd, &com_data, command);
(gdb) p thd->m_query_string->str
$9 = 0x7fb21c009bf8 "PURGE BINARY LOGS BEFORE NOW()"
(gdb) f 2
#2  0x000055f9c6ebb60a in native_mutex_lock (mutex=0x7fb22c0275b8) at /work/mysql/src/include/
thr_mutex.h:93
93        return pthread_mutex_lock(mutex);
(gdb) p mutex->__data->__owner
$10 = 889019
(gdb) thread find 889019
Thread 38 has target id 'Thread 0x7fb2dc094700 (LWP 889019)'
```

GDB / Coredump / pt-pmp

- All 4996 threads are trying to connect and waiting on mutex from Thread 38
- Thread 38
  - SELECT * FROM performance_schema.session_variables WHERE VARIABLE_NAME LIKE 'binlog_transaction_dependency_tracking'
  - Wait on mutex from Thread 44
- Thread 44
  - SHOW BINARY LOGS
  - Wait on mutex from Thread 42
- Thread 42
  - PURGE BINARY LOGS BEFORE NOW()
  - Wait on mutex from Thread 38

GDB / Coredump / pt-pmp

- Loop in wait-for graph
- Bug: Deadlock during purge_logs_before_date - PS-1049 #91941

Record & Replay

Xtrabackup incremental backup bug

PERCONALIVE

Record & Replay

- Created by Mozilla - https://rr-project.org/
- Open Source - https://github.com/rr-debugger/rr
- Record once, replay deterministically as many times you want
- **Let you replay the code backwards**
- Run on a GDB session
- Hard to reproduce bugs
- Complex bugs

Record & Replay

- Replay the Execution of MySQL With RR (Record and Replay)
- PXB-2180
- PXB crash randomly after applying N incremental backups
- Not always the same stack trace - Crash on different parts of InnoDB
- Always trying to apply a record on same Space=>Page

```
#12 0x00000000015ad05f in recv_parse_or_apply_log_rec_body (type=MLOG_COMP_REC_INSERT,
ptr=0x7f2849150556 "\003K4G", '\377' <repeats 13 times>, end_ptr=0x7f2849150573 "",
space_id=<optimized out>, page_no=<optimized out>, block=0x7f2847d7da00, mtr=0x7f286857b4f0,
parsed_bytes=18446744073709551615) at /home/marcelo.altmann/percona-xtrabackup/storage/innobase/
log/log0recv.cc:2002
2002          ptr = page_cur_parse_insert_rec(FALSE, ptr, end_ptr, block, index, mtr);
(gdb) p block->page->id
+p block->page->id
$3 = {
  m_space = 4294967294,
  m_page_no = 5
}
```

Record & Replay

- Page layout diverted somehow between PXB and Server
- Crash on N th inc is a consequence of something wrong on N-M th inc.
- Run MySQL under RR
- Make a copy of all backups so we can re-run –prepare when the issue happens
- Read the LSNs for this same page before/after each backup prepare (od).
- Identify all changes to m_space = 4294967294 & m_page_no = 5 at mysqld.
- Got a reproducible backup - 6th incremental was crashing

Record & Replay

- m_space = 4294967294 correspond to the MySQL data dictionary (mysql.ibd) – dict0dict.h:1146
- On disk page, LSN is stored at the 16th byte of the page and has a size of 8 bytes – fil0types.h:66
- Pages are written sequentially to disk, as an example, for the default 16k page size, from bytes 1 to 16384 will have the data for page 0, from byte 16385 to 32768 data from page 1, and so on.
- Frame is raw data of a page – buf0buf.h:1358

Record & Replay

```
$ od -j $((16384 * 5 + 16)) -N 8 -t x1 full/mysql.ibd
0240020 00 00 00 00 01 10 21 85
0240030
```

```
Breakpoint 1, buf_flush_note_modification (block=0x7fd2df4ad750,
69      ut_ad(!srv_read_only_mode ||
++rr-set-suppress-run-hook 1
(rr) p/x block->frame[16]@8
+p/x block->frame[16]@8
$1 =    {[0x0] = 0x0,
  [0x1] = 0x0,
  [0x2] = 0x0,
  [0x3] = 0x0,
  [0x4] = 0x1,
  [0x5] = 0x10,
  [0x6] = 0x21,
  [0x7] = 0x85}
(rr)
```

BF: 0x1102185

Record & Replay

```
xtrabackup --prepare --apply-log-only --target-dir=full/
. . .
Shutdown completed; log sequence number 17897577
Number of pools: 1
210402 17:46:29 completed OK!

$ od -j $((16384 * 5 + 16)) -N 8 -t x1 full/mysql.ibd
0240020 00 00 00 00 01 11 07 06
0240030

(rr) c
Breakpoint 1, buf_flush_note_modification (block=0x7fd2df4ad750,
69     ut_ad(!srv_read_only_mode ||
++rr-set-suppress-run-hook 1
(rr) p/x block->frame[16]@8
+p/x block->frame[16]@8
$16 =   {[0x0] = 0x0,
  [0x1] = 0x0,
  [0x2] = 0x0,
  [0x3] = 0x0,
  [0x4] = 0x1,
  [0x5] = 0x11,
  [0x6] = 0x7,
  [0x7] = 0x6}
(rr)
```

BF: 0x1102185

AF: 0x1100706

Record & Replay

```
xtrabackup --prepare --apply-log-only --target-dir=full/ --incremental-dir=inc1/
. . .
Shutdown completed; log sequence number 19082430
. . .
210402 18:12:20 completed OK!

$ od -j $((16384 * 5 + 16)) -N 8 -t x1 full/mysql.ibd
0240020 00 00 00 00 01 23 19 06
0240030

(rr) c
Breakpoint 1, buf_flush_note_modification (block=0x7fd2df4ad750, start_lsn=20262
69     ut_ad(!srv_read_only_mode ||
++rr-set-suppress-run-hook 1
(rr) p/x block->frame[16]@8
+p/x block->frame[16]@8
$17 =   {[0x0] = 0x0,
  [0x1] = 0x0,
  [0x2] = 0x0,
  [0x3] = 0x0,
  [0x4] = 0x1,
  [0x5] = 0x23,
  [0x6] = 0x19,
  [0x7] = 0x6}
(rr)
```

BF: 0x1102185

AF: 0x1100706

I1: 0x1231906

Record & Replay

```
xtrabackup --prepare --apply-log-only --target-dir=full/ --incremental-dir=inc3/
. . .
Shutdown completed; log sequence number 21455916
. . .
210402 18:18:25 completed OK!

$ od -j $((16384 * 5 + 16)) -N 8 -t x1 full/mysql.ibd
0240020 00 00 00 00 01 47 4d 3f
0240030

Breakpoint 1, buf_flush_note_modification (block=0x7fd2df4ad750, start_lsn=255294
69     ut_ad(!srv_read_only_mode ||
++rr-set-suppress-run-hook 1
(rr) p/x block->frame[16]@8
+p/x block->frame[16]@8
$19 =   {[0x0] = 0x0,
  [0x1] = 0x0,
  [0x2] = 0x0,
  [0x3] = 0x0,
  [0x4] = 0x1,
  [0x5] = 0x47,
  [0x6] = 0x4d,
  [0x7] = 0x3f}
(rr)
```

```
BF: 0x1102185

AF: 0x1100706

I1: 0x1231906

.  .  .

I3: 0x1474d3f
```

Record & Replay

```
xtrabackup --prepare --apply-log-only --target-dir=full/ --incremental-dir=inc4/
. . .
Shutdown completed; log sequence number 23044902
. . .
210402 18:24:00 completed OK!

$ od -j $((16384 * 5 + 16)) -N 8 -t x1 full/mysql.ibd
0240020 00 00 00 00 01 5f a3 26
0240030


Breakpoint 1, buf_flush_note_modification (block=0x7fd2df4ad750, start_lsn=27218
69     ut_ad(!srv_read_only_mode ||
++rr-set-suppress-run-hook 1
(rr) p/x block->frame[16]@8
+p/x block->frame[16]@8
$242 =   {[0x0] = 0x0,
  [0x1] = 0x0,
  [0x2] = 0x0,
  [0x3] = 0x0,
  [0x4] = 0x1,
  [0x5] = 0x9f,
  [0x6] = 0x3f,
  [0x7] = 0xc9}
(rr)
```

BF: 0x1102185

AF: 0x1100706

I1: 0x1231906

. . .

I3: 0x1474d3f

B: 0x15fa326

S: 0x19f3fc9

Record & Replay

```
(rr) p block->frame
+p block->frame
$243 = (unsigned char *) 0x7fd2e0758000 "\327\064X["
(rr) watch *(unsigned char *) 0x7fd2e0758000
(rr) reverse-cont
Hardware watchpoint 2: *(unsigned char *) 0x7fd2e0758000
Old value = 80 'P'
New value = 43 '+'
0x0000000004c13903 in mach_write_to_4 (b=0x7fd2e0758000 "+k*\304", n=1353655015)
135   b[0] = static_cast<byte>(n >> 24);
++rr-set-suppress-run-hook 1
++rr-set-suppress-run-hook 1
(rr) p/x buf_flush_init_for_writing::block->frame[16]@8
+p/x buf_flush_init_for_writing::block->frame[16]@8
$12 =   {[0x0] = 0x0,
  [0x1] = 0x0,
  [0x2] = 0x0,
  [0x3] = 0x0,
  [0x4] = 0x1,
  [0x5] = 0x47,
  [0x6] = 0x4d,
  [0x7] = 0x3f}
(rr)
```

BF: 0x1102185

AF: 0x1100706

I1: 0x1231906

. . .

I3: 0x1474d3f

B: 0x15fa326

S: 0x19f3fc9


Are you passionate about Open Source?!
We're looking for you!
Join us!
#RemoteWork
APPLY NOW: percona.com/careers