# MyRocks in MariaDB

Sergei Petrunia <sergey@mariadb.com>

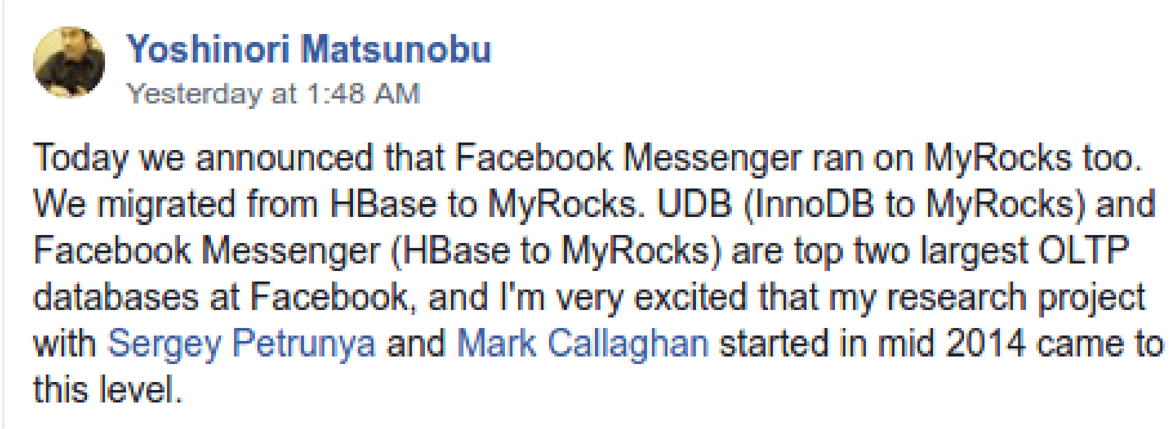
MariaDB Tampere Meetup
June 2018

# What is MyRocks

- Hopefully everybody knows by now
- A storage engine based on RocksDB
- LSM-architecture
  - Uses less space and does less writes than InnoDB for the same workload
- MyRocks “lives” in Facebook’s MySQL branch
  - https://github.com/facebook/mysql-5.6

# What is MyRocks (2)

- MyRocks is used in production at Facebook
- Now in Messenger, too:



Yoshinori Matsunobu
Yesterday at 1:48 AM

Today we announced that Facebook Messenger ran on MyRocks too. We migrated from HBase to MyRocks. UDB (InnoDB to MyRocks) and Facebook Messenger (HBase to MyRocks) are top two largest OLTP databases at Facebook, and I'm very excited that my research project with Sergey Petrunya and Mark Callaghan started in mid 2014 came to this level.

# FB/MySQL is not very user-friendly

- No binaries or packages
  - Source only
- No stable releases
- Contains extra features

# MyRocks “distributions”

- MariaDB 10.2+
- Percona Server 5.7+
  - Also present in percona-5.6 but AFAIK it is not [fully] functional there.

# MyRocks distribution releases

- March 2017
  - Alpha-maturity plugin in MariaDB 10.2
  - Percona server: “test builds”
- Jan, 2018
  - Stable in Percona-5.7
  - Beta in MariaDB 10.2/10.3
- February 2018: RC in MariaDB 10.2/10.3
- **May 2018: GA in MariaDB 10.2/10.3**

# MyRocks life in distributions

- FB continues to develop MyRocks
  - RocksDB is developed too
  - Quite a few RocksDB features are for MyRocks
- Distros periodically (manually) merge from facebook/mysql-5.6/storage/rocksdb
  - Interesting definition of “Stable”
- MyRocks is a loadable module in distros
  - Compression libraries

# MyRocks in MariaDB

# Tests

# MyRocks tests

- MTR tests are in storage/rocksdb/mysql-test/*
- Facebook runs tests
  - On machines with lots of RAM/CPU
  - …
- Testing includes some amount of stress-testing
  - MTR tests invoke RQG, mysqlslap

# Testing in MariaDB

- Lots of plaforms
- Debug builds
- Memory/CPU/Disk constrained environments
  - Timeout errors
  - Race conditions [in tests]
    - Have exposed some test/code bugs in the upstream

# Unstable tests

- There are unstable tests
- But look who’s talking^W^W^W^W^W

```
mariadb-10.2$ grep rocksdb mysql-test/unstable-tests | wc -l
23
mariadb-10.2$ grep innodb mysql-test/unstable-tests | wc -l
123
mariadb-10.2$ grep main mysql-test/unstable-tests | wc -l
118
```

# Valgrind tests

**MDEV-12439: MariaRocks produces numerous (spurious?) valgrind failures**

- Needs a recent valgrind
- Fixed in upstream
  - Pending merge from the upstream

# A race condition on Windows

## MDEV-16565: rocksdb.read_only_tx failed in buildbot with server crash

```
ha_rocksdb.dll!rocksdb::log::Writer::AddRecord() [log_writer.cc:44]
ha_rocksdb.dll!rocksdb::DBImpl::WriteToWAL() [db_impl_write.cc:793]
ha_rocksdb.dll!rocksdb::DBImpl::ConcurrentWriteToWAL() [db_impl_write.cc:903]
ha_rocksdb.dll!rocksdb::DBImpl::WriteImplWALOnly() [db_impl_write.cc:581]
ha_rocksdb.dll!rocksdb::DBImpl::WriteImpl() [db_impl_write.cc:95]
ha_rocksdb.dll!rocksdb::WriteCommittedTxn::PrepareInternal()
[pessimistic_transaction.cc:227]
```

- Pending a merge from the upstream
  - (see MDEV text for details)

# Packaging

# Packaging into debs/rpms

- .debs, .rpms: MyRocks is a separate package
  - Like TokuDB, Cassandra, etc
- MyRocks package depends on compression libraries
  - ZStandard, Snappy, Zlib
  - Lz4, … (whatever are available on this distro)
- MyRocks depends on RocksDB library
  - Only support MyRocks@revX - RocksDB@revY combinations
  - Because of that, RocksDB library is “included”
    - We compile-in their source files.

# Packaging into tarballs

- Tarballs include ha_rocksdb.so
  - Compression libraries are linked in to be self-contained
- Same on Windows
- MDEV-15084: Some tarballs are missing ha_rocksdb.so
  - Need to use old distro to depend on old libc/systemd etc
  - But old distro has old compiler and cmake
    - Cannot build
    - Install new toolchain on the old builder?

# Essential Tuning

# Essential MyRocks tuning

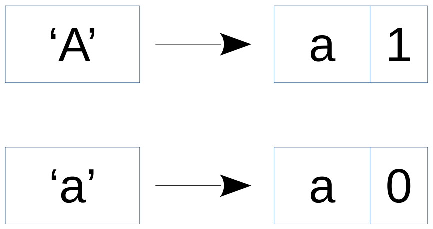
- **rocksdb_flush_log_at_trx_commit=1**
  - Same as **innodb_flush_log_at_trx_commit=1**
  - Together with **sync_binlog=1** makes the master crash-safe
- **rocksdb_block_cache_size=...**
  - Similar to innodb_buffer_pool_size
  - 500Mb by default (will use up to that amount * 1.5?)
  - Set higher if have more RAM
- Safety: **rocksdb_max_row_locks=...**

# Indexes

- Primary Key is the clustered index (like in InnoDB)
- Secondary indexes include PK columns (like in InnoDB)
- Non-unique secondary indexes are cheap
  - Index maintenance is read-free
- Unique secondary indexes are more expensive
  - Maintenance is not read-free
  - Reads can suffer from read amplification
  - Can use **`@@unique_check=false`**, at your own risk

# Charsets and collations

- Index entries are compared with memcmp(), “mem-comparable”
- “**Reversible collations**”: binary, latin1_bin, utf8_bin
  - Can convert values to/from their mem-comparable form
- “**Restorable collations**”
  - 1-byte characters, one weght per character
  - e.g. latin1_general_ci, latin1_swedish_ci, ...
  - Index stores mem-comparable form + restore data



- **Other collations** (e.g. utf8_general_ci)
  - Index-only scans are not supported
  - Table row stores the original value

# Charsets and collations (2)

- Using indexes on “Other” (collation) produces a warning

```
MariaDB> create table t3 (...
         a varchar(100) collate utf8_general_ci, key(a)) engine=rocksdb;
Query OK, 0 rows affected, 1 warning (0.20 sec)

MariaDB> show warnings\G
*************************** 1. row ***************************
  Level: Warning
   Code: 1815
Message: Internal error: Indexed column test.t3.a uses a collation that
does not allow index-only access in secondary key and has reduced disk
space efficiency in primary key.
```

# Data loading

# Data loading – good news

- It’s a write-optimized storage engine
- The same data in MyRocks takes less space on disk than on InnoDB
  - 1.5x, 3x, or more
- Can load the data faster than InnoDB
- But…

# Data loading – limitations

- Limitation: Transaction must fit in memory

```
mysql> ALTER TABLE big_table ENGINE=RocksDB;
ERROR 2013 (HY000): Lost connection to MySQL server during query
```

- Uses a lot of memory, then killed by OOM killer.
- Need to use special settings for loading data
  - https://mariadb.com/kb/en/library/loading-data-into-myrocks/
  - https://github.com/facebook/mysql-5.6/wiki/data-loading

# Sorted bulk loading

```
MariaDB> set rocksdb_bulk_load=1;
... # something that inserts the data in PK order
MariaDB> set rocksdb_bulk_load=0;
```

Bypasses

- Transactional locking
- Seeing the data you are inserting
- Compactions
- ...

# Unsorted bulk loading

```
MariaDB> set rocksdb_bulk_load_allow_unsorted=0;
MariaDB> set rocksdb_bulk_load=1;
... # something that inserts data in any order
MariaDB> set rocksdb_bulk_load=0;
```

- Same as previous but doesn’t require PK order

# Other speedups for lading

- **rocksdb_commit_in_the_middle**
  - Auto commit every **rocksdb_bulk_load_size** rows

- **@@unique_checks**
  - Setting to FALSE will bypass unique checks

# Creating indexes

- Index creation doesn’t need any special settings

```
mysql> ALTER TABLE myrocks_table ADD INDEX(...);
```

- Compression is enabled by default
  - Lots of settings to tune it further

# **rocksdb_max_row_locks** as a safety setting

- Avoid run-away memory usage and OOM killer:

```
MariaDB> set rocksdb_max_row_locks=10000;

MariaDB> alter table t1 engine=rocksdb;

ERROR 4067 (HY000): Status error 10 received from RocksDB:
Operation aborted: Failed to acquire lock due to
max_num_locks limit
```

- Now, **rocksdb_max_row_locks=1M** by default

# Upcoming changes

- There is an effort underway to support big transactions:
- https://github.com/facebook/rocksdb/wiki/WriteUnprepared-Transactions

# Replication

# MyRocks only supports RBR

- MyRocks’ highest isolation level is “snapshot isolation”, that is REPEATABLE-READ
- Statement-Based Replication
  - Slave runs the statements sequentially (~ serializable)
  - InnoDB uses “Gap Locks” to provide isolation req’d by SBR
  - MyRocks doesn’t support Gap Locks
- Row-Based Replication must be used
  - **`binlog_format=row`**

# mysql.gtid_slave_pos

- **`mysql.gtid_slave_pos`** stores slave’s position
- Use a transactional storage engine for it
  - Database contents will match slave position after crash recovery
  - Makes the slave crash-safe.
- **`mysql.gtid_slave_pos`** uses a different engine?
  - Cross-engine transaction (slow).

# Per-engine mysql.gtid_slave_pos

- The idea:
  - Have **`mysql.gtid_slave_pos_${engine}`** for each storage engine
  - Slave position is the biggest position in all tables.
  - Transaction affecting only MyRocks will only update **`mysql.gtid_slave_pos_rocksdb`**
- Configuration:

```
--gtid-pos-auto-engines=engine1,engine2,...
```

# Per-engine mysql.gtid_slave_pos

- Available in MariaDB 10.3
- Thanks for the patch to
  - Kristian Nielsen (implementation)
  - Booking.com (request and funding)

- Don’t let your slave be 2-3x slower:
  - 10.3:
    ```
    my.cnf: gtid-pos-auto-engines=INNODB,ROCKSDB
    ```
  - 10.2:
    ```
    MariaDB> ALTER TABLE mysql.gtid_slave_pos
                  ENGINE=RocksDB;
    ```

# Parallel replication

- Facebook/MySQL-5.6 is based on MySQL 5.6
  - Parallel replication for data in different databases
- MariaDB has more advanced parallel slave (controlled by `@@slave_parallel_mode`)
  - **Conservative** (group-commit based)
  - **Optimistic** (apply transactions in parallel, on conflict roll back the transaction with greater seq_no)
  - **Aggressive**

# Conservative Parallel Replication

- Conservative replication works with MyRocks
- For tables without PK, RBR does table/index scans to locate the row to update
  - MDEV-16242: ER_KEY_NOT_FOUND
  - Have a patch for it

# Optimistic Parallel Replication

- Requires the feature:
  - **“if transaction X is waiting for lock held by transaction Y, inform the SQL layer about it so it can kill Y if it is after X in the binlog order”**
- MDEV-16428: this doesn’t work
- Have a candidate patch
  - Needs testing
  - Needs changes in RocksDB (discussing it with FB)

# Backup

# Hot Backup for MyRocks

- LSM files are immutable, conceptually backups are easy
  `set rocksdb_create_snapshot='/path/to/snapshot';`
- FB provides **myrocks_hotbackup** tool
  - Not very user-friendly
- **mariabackup** now supports MyRocks
  - MDEV-13122, closed on June, 6th.
  - No incremental backup.

# Takeaways

- MyRocks is Stable in MariaDB 10.2+
- Packaging is OK
- Some special considerations needed when loading data, etc.
- mariabackup support added recently
- Parallel replication has 2 bugs
  - Have fixes for these.

# Thanks!