MySQL Group Replication

Kenny Gryp (@gryp)

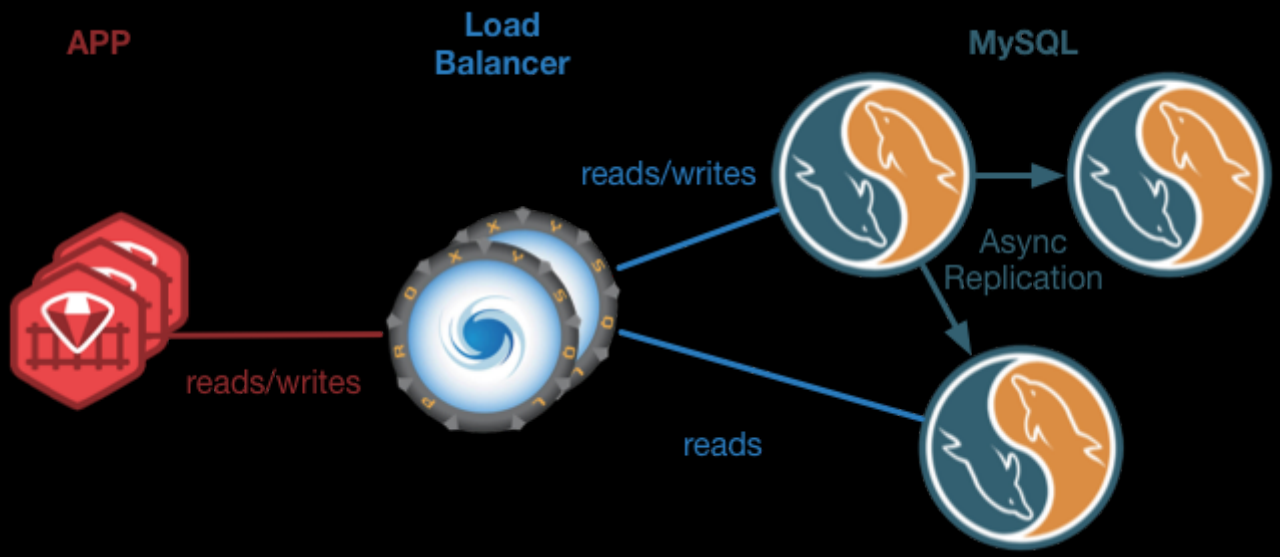
Table of Contents

Apologies upfront. I was able to spend limited time researching Group Replication, my knowledge is a lot more limited compared to many MySQL (Group Replication) developers in this room. I may have made wrong assumptions or discuss problems, missing features which are likely known and on the roadmap to be fixed/developed in reasonable time. Group Replication is a quite new feature and only recently became GA. Hereby I give full responsibility to the MySQL Developers to respond whenever I made a wrong statement :-), but please keep in mind that my talk is only 25 minutes long. Even though MySQL Group Replication is marked GA, it is still a new feature and database software adoption usually takes a long time, bugs mentioned (some of which are not verified yet) in these slides are not here to try to tell you the feature is not good, I believe it is my duty as member of the community to provide feedback and getting the opportunity to talk in this room with a lot of Oracle employees clearly demonstrates this is their desire as well, even though it does scare me quite a lot! This talk describes the status of Group Replication on 31/01/2017.

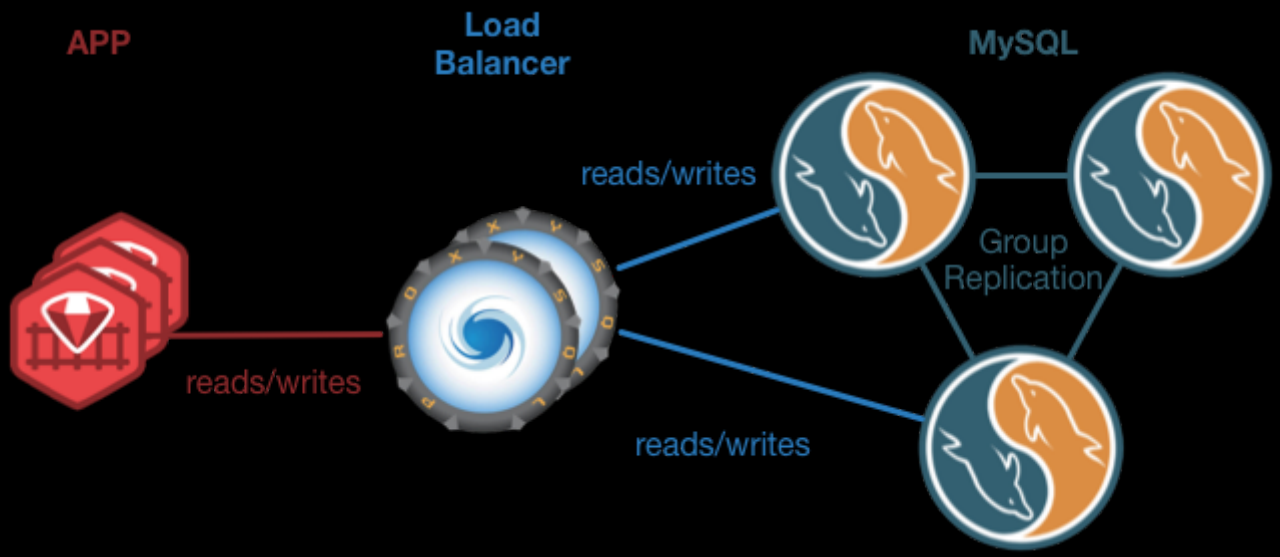
MySQL Group Replication

Overview

MySQL Asynchronous Replication



MySQL Group Replication



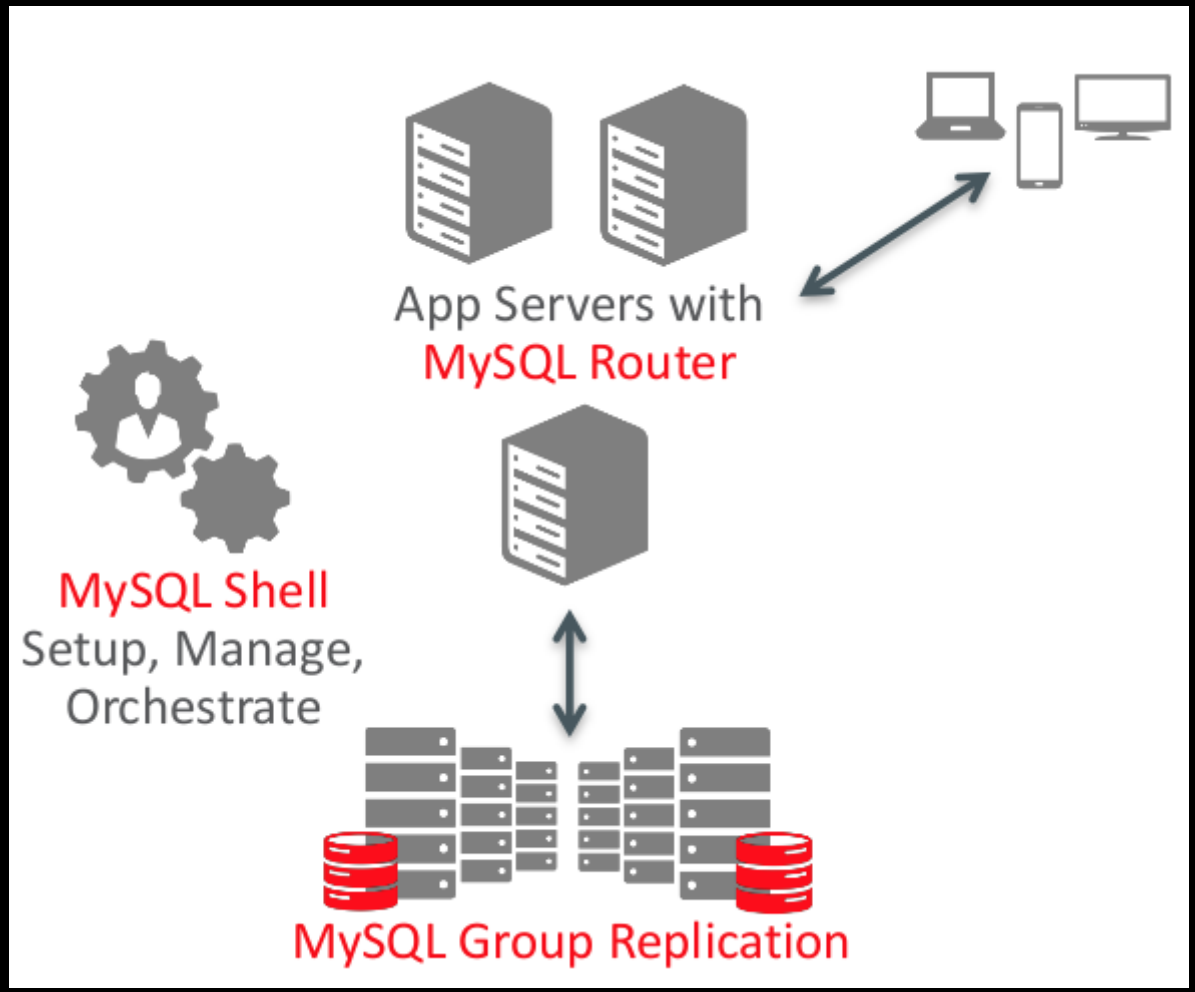
Quick Overview

- 'Synchronous replication'
- Writes in entire Group Replication executed in 'Global Total Order'
- Majority consensus (Paxos Mencius)
- Optimistic Locking: Conflict Detection after replicating trx: 'Certification'
  - First Committer Wins
- Every node has all data, cluster is 'as fast as slowest node'.
- Nodes can join/leave cluster

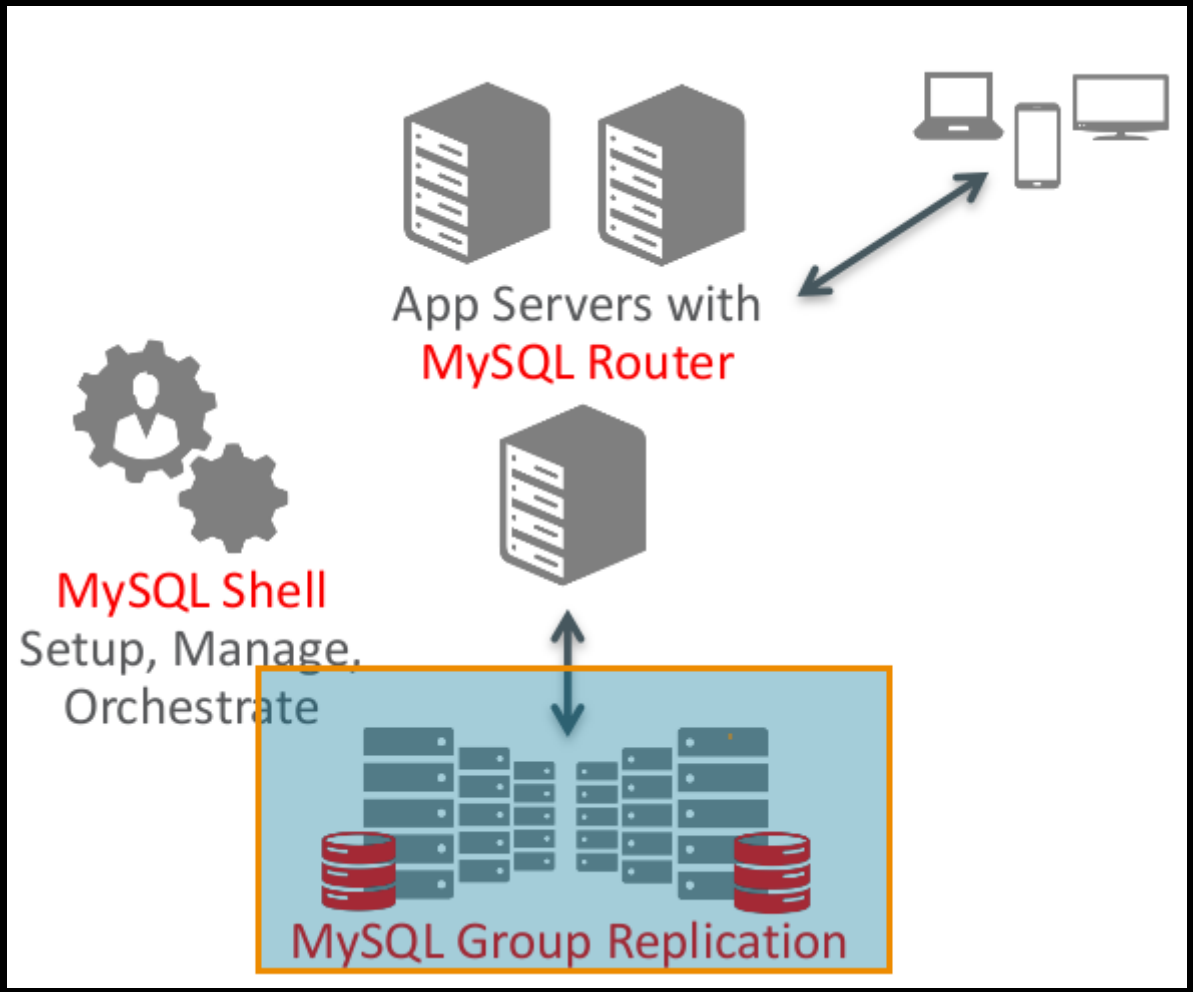
Properties

- No concept of master/slave, only 'members'
- Durability:
  No data loss, when failure of nodes happen. Does not accept writes if there is no Quorum.
- Active:Active Master:
  All nodes can be configured to accept writes at same time *
- No (time consuming) failover is necessary, every member can become a writer member at any time
- Added latency to every transaction `COMMIT`.

MySQL InnoDB Cluster



MySQL InnoDB Cluster



MySQL Group Replication

- Main focus: Design & Usability
- Performance & Stability was not yet analyzed

Use Cases for Group Replication

- Environments with strict durability requirements
  (no data loss if master member is lost)
- Write to multiple nodes
  ('scalability' by splitting write/read workloads)
- Improve failover time
- ...

MySQL Group Replication

Provisioning Nodes

GTID

Please note that Group Replication uses GTID

GTID

Please note that Group Replication uses GTID

Keep into Account:

- Creating a cluster and provisioning nodes requires 'compatible' GTID-sets
- Errant Transactions!

Errant Transactions

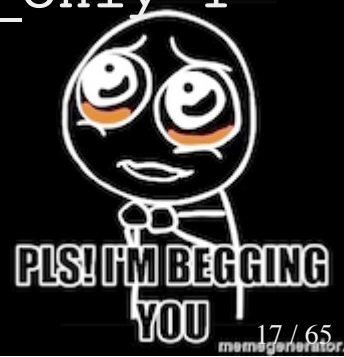
Ensure there are no errant transactions before starting group replication:

```
[ERROR] Plugin group_replication reported:
'This member has more executed transactions than those present
 in the group.
    Local transactions: 74dc6ab2-e1cc-11e6-92aa-08002789cd2e:1
    >
    Group transactions: 72149827-e1cc-11e6-9daf-08002789cd2e:1,
                        da7aba5e-dead-da7a-ba55-da7aba5e57ab:1-5'
[ERROR] Plugin group_replication reported:
     'The member contains transactions not present in the group.
      The member will now exit the group.'
[Note] Plugin group_replication reported:
    'To force this member into the group you can use
     the group_replication_allow_local_disjoint_gtids_join option'
```

Best Practice

Please never use `group_replication_allow_local_disjoint_gtids_joi`

- once you use it, you always have to keep it on.
- they might have been writes to the individual node (GR not active)
  - data consistency/split brain/data loss/...
  - #84728: GR failure at start still starts MySQL
  - #84733: not possible to start with `super_read_only=1`



Starting Cluster

First Choose the right node to bootstrap:

```
mysql> select @@global.gtid_executed\G
 *************************** 1. row ***************************
@@global.gtid_executed: 72149827-e1cc-11e6-9daf-08002789cd2e:1,
                        740e1fd2-e1cc-11e6-a8ec-08002789cd2e:1-2,
                        74dc6ab2-e1cc-11e6-92aa-08002789cd2e:1-2,
                        da7aba5e-dead-da7a-ba55-da7aba5e57ab:1-399
1 row in set (0.00 sec)
```

- ensure compatible GTID sets or forget about it!
- choose node with all GTIDs

Cluster Membership Operations

- Start a new cluster:

```
SET GLOBAL group_replication_bootstrap_group=on;
START GROUP_REPLICATION;
SET GLOBAL group_replication_bootstrap_group=off;
```

Cluster Membership Operations

- Start a new cluster:

```
SET GLOBAL group_replication_bootstrap_group=on;
START GROUP_REPLICATION;
SET GLOBAL group_replication_bootstrap_group=off;
```

- Restore (GTID-enabled) Backup

```
SET GLOBAL group_replication_group_seeds='node1,node2,node3';
START GROUP_REPLICATION;
```

- #84674: unresolved hostnames block GR from starting

MySQL Group Replication

Configuration

Configuration Requirements

```
[mysqld]
log-bin
binlog-format=row
binlog-checksum=NONE
gtid-mode=ON
log-slave-updates
master-info-repository=TABLE
relay-log-info-repository=TABLE
transaction-write-set-extraction=XXHASH64
```

Group Replication Configuration:

```
group_replication_group_name="da7aba5e-dead-da7a-ba55-da7aba5e57ab"
group_replication_local_address= "gr-2:24901"
group_replication_group_seeds= "gr-1:24901,gr-2:24901,gr-3:24901"
```

MySQL InnoDB Cluster

- No Security/Authentication is described in these slides
- Possible to create a cluster in the MySQL Shell w. AdminAPI
  - Also performs configuration checks

Other Requirements/Limitations

Required:

- InnoDB Required
- PK on every table

Other Requirements/Limitations

Required:

- InnoDB Required
- PK on every table

Not supported:

- Transaction Savepoints
  - #84799: mysqldump --single-transaction uses savepoints, does not work with GR
- In multi-writer/active:active
  - Concurrent DDL vs DML/DDL operations

Oracle's (Valid) Recommendations

- Only use `group_replication_single_primary_mode=ON`
  - write to a single node only
- Not recommended for WAN
  (Probably because of Majority Consensus in Paxos Mencius)
- Requires uneven amount of nodes for proper Quorum

My Recommendations - `my.cnf`

My Recommendations - `my.cnf`

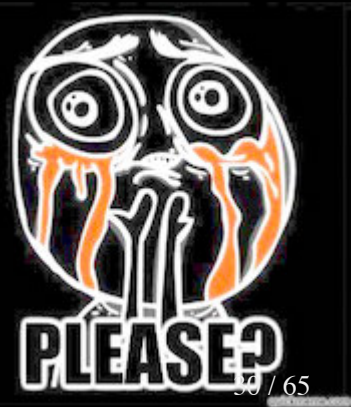
- Do not use `loose-` even though it is mentioned in the manual
  - #84631: installation documentation issues

My Recommendations - `my.cnf`

- Do not use `loose-` even though it is mentioned in the manual
  - #84631: installation documentation issues
- `._allow_local_disjoint_gtids_join=OFF`

My Recommendations - `my.cnf`

- Do not use `loose-` even though it is mentioned in the manual
  - #84631: installation documentation issues
- `._allow_local_disjoint_gtids_join=OFF`
- Single writer mode?
  `group_replication_auto_increment_increment=7`
  default is too high. Set to 1.



My Recommendations - `my.cnf`

- Do not use `loose-` even though it is mentioned in the manual
  - #84631: installation documentation issues
- `._allow_local_disjoint_gtids_join=OFF`
- Single writer mode?
  `group_replication_auto_increment_increment=7`
  default is too high. Set to 1.
- `group_replication_bootstrap_group=OFF`

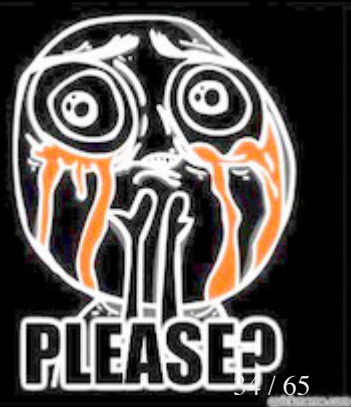
My Recommendations - `my.cnf`

- Do not use `loose-` even though it is mentioned in the manual
  - #84631: installation documentation issues
- `._allow_local_disjoint_gtids_join=OFF`
- Single writer mode?
  `group_replication_auto_increment_increment=7`
  default is too high. Set to 1.
- `group_replication_bootstrap_group=OFF`
- `group_replication_start_on_boot=ON`
  - #84728: GR failure at start still starts MySQL

My Recommendations

My Recommendations

- Ensure all **FQDN** hostnames are resolvable.
  - #84674: unresolved hostnames block GR from starting
  - `@@global.hostname` is used by other members



My Recommendations

- Ensure all **FQDN** hostnames are resolvable.
  - #84674: unresolved hostnames block GR from starting
  - `@@global.hostname` is used by other members
- Dangerous to issue:

```
SET GLOBAL read_only=OFF;
SET GLOBAL super_read_only=OFF;
STOP GROUP_REPLICATION;
```

  - #84795: `STOP GROUP_REPLICATION` sets `super_read_only=off`

My Failed Recommendation

- In an attempt to prevent split brain and because of:
  - #84728: GR failure at start still starts MySQL
- I tried to enforce `super_read_only=1` at boot, but that failed too:
  - #84733: not possible to start with `super_read_only=1`

I did not find a way to prevent a MySQL node from starting as a individual r/w MySQL server when Group Replication failed to start.

MySQL Group Replication

Monitoring

(Profiling, Trending, Alerting, Status, Troubleshooting)

Performance Schema

```
SELECT TABLE_NAME FROM information_schema.TABLES
WHERE TABLE_SCHEMA='performance_schema'
  AND TABLE_NAME LIKE '%replication%';
+-------------------------------------------+
| TABLE_NAME                                |
+-------------------------------------------+
| replication_applier_configuration         |
| replication_applier_status                |
| replication_applier_status_by_coordinator |
| replication_applier_status_by_worker      |
| replication_connection_configuration      |
| replication_connection_status             |
| replication_group_member_stats            |
| replication_group_members                 |
+-------------------------------------------+
```

2 replication appliers:

- `group_replication_applier` <- group replication

Trending - SHOW GLOBAL STATUS

Limited Status Information (which are usually easy to gather):

```
mysql> show global status like '%group%';
+---------------------------------+--------------------------------------
| Variable_name                   | Value
+---------------------------------+--------------------------------------
| Com_group_replication_start     | 0
| Com_group_replication_stop      | 0
| group_replication_primary_member | 72149827-e1cc-11e6-9daf-08002789cd2e
+---------------------------------+--------------------------------------
```

Trending - PFS

```
mysql> select * from replication_group_member_stats\G
*************************** 1. row ***************************
                      CHANNEL_NAME: group_replication_applier
                           VIEW_ID: 14860449946972589:2
                         MEMBER_ID: 74dc6ab2-e1cc-11e6-92aa-08002789cd2e
       COUNT_TRANSACTIONS_IN_QUEUE: 0       # Certification queue
        COUNT_TRANSACTIONS_CHECKED: 4
          COUNT_CONFLICTS_DETECTED: 0
COUNT_TRANSACTIONS_ROWS_VALIDATING: 0
TRANSACTIONS_COMMITTED_ALL_MEMBERS:
      72149827-e1cc-11e6-9daf-08002789cd2e:1,
      740e1fd2-e1cc-11e6-a8ec-08002789cd2e:1-2,
      74dc6ab2-e1cc-11e6-92aa-08002789cd2e:1-2,
      da7aba5e-dead-da7a-ba55-da7aba5e57ab:1-444:1000041-1000503:2000041-
    LAST_CONFLICT_FREE_TRANSACTION:
      da7aba5e-dead-da7a-ba55-da7aba5e57ab:444
1 row in set (0.00 sec)
```

Group Replication Status

```
mysql> select * from replication_connection_status\G
 *************************** 1. row ***************************
             CHANNEL_NAME: group_replication_applier
               GROUP_NAME: da7aba5e-dead-da7a-ba55-da7aba5e57ab
              SOURCE_UUID: da7aba5e-dead-da7a-ba55-da7aba5e57ab
                THREAD_ID: NULL
            SERVICE_STATE: ON
COUNT_RECEIVED_HEARTBEATS: 0
 LAST_HEARTBEAT_TIMESTAMP: 0000-00-00 00:00:00
  RECEIVED_TRANSACTION_SET:
    72149827-e1cc-11e6-9daf-08002789cd2e:1,
    740e1fd2-e1cc-11e6-a8ec-08002789cd2e:1-2,
    74dc6ab2-e1cc-11e6-92aa-08002789cd2e:1-2,
    da7aba5e-dead-da7a-ba55-da7aba5e57ab:1-444:1000041-1000503:2000041-20
       LAST_ERROR_NUMBER: 0
      LAST_ERROR_MESSAGE:
    LAST_ERROR_TIMESTAMP: 0000-00-00 00:00:00
```

Group Replication Members

```
mysql> select * from replication_group_members;
+--------------------------------------+------+------+--------+
| ID                                   | HOST | PORT | STATE  |
+--------------------------------------+------+------+--------+
| 72149827-e1cc-11e6-9daf-08002789cd2e | gr-1 | 3306 | ONLINE |
| 74dc6ab2-e1cc-11e6-92aa-08002789cd2e | gr-3 | 3306 | ONLINE |
+--------------------------------------+------+------+--------+
2 rows in set (0.00 sec)
# slightly modified output
```

- #84796: GR Member status is wrong

Group Replication Read or Write.

```
mysql> select @@global.super_read_only;
+-------------------------+
| @@global.super_read_only |
+-------------------------+
|                       1 |
+-------------------------+
1 row in set (0.05 sec)
```

Group Replication Lag

```
SELECT sys.gtid_count(
        GTID_SUBTRACT(
          (
          SELECT
            Received_transaction_set
          FROM performance_schema.replication_connection_status
          WHERE
            Channel_name = 'group_replication_applier'
          ),
          (SELECT
             @@global.GTID_EXECUTED)
          )
        )
      )
```

Thanks to @lefred:
https://github.com/lefred/mysql_gr_routing_check/

Commands

```
mysql> SHOW SLAVE STATUS FOR CHANNEL 'group_replication_recovery'\G

mysql> SHOW SLAVE STATUS FOR CHANNEL 'group_replication_applier'\G
ERROR 3139 (HY000): SHOW SLAVE STATUS cannot be performed on channel 'gr
```

Member State Changes

Can use improvements:

- #84796: GR Member status is wrong
- #84798: Group Replication can use some verbosity in the error log

Multi Node Conflicts

- Optimistic Locking
- First committer wins
- #84730: ability to troubleshoot transaction rollbacks

```
ERROR 3101 (HY000) at line 1: Plugin instructed the server
                              to rollback the current transaction.
```

MySQL Group Replication

Backups

Backups

It's just InnoDB, use your favorite GTID supported backup tool:

- Percona XtraBackup
- MySQL Enterprise Backup
- `mysqldump`
- `mysqlpump`
- `mydumper`

but...

Backups

It's just InnoDB, use your favorite GTID supported backup tool:

- Percona XtraBackup
- MySQL Enterprise Backup
- `mysqldump`
- `mysqlpump`
- `mydumper`

but...

- #84799: mysqldump --single-transaction uses savepoints, does not work with GR
- `mydumper --use-savepoints` is also affected

MySQL Group Replication

Load Balancers

MySQL Router (Beta)

MySQL Router is part of MySQL InnoDB Cluster

MySQL Router (Beta)

MySQL Router is part of MySQL InnoDB Cluster

I briefly evaluated, but quickly ran into serious problems:

- missing a lot of features
- pretty unknown in the community
- ...:
  - #83237: mysqlrouter connects to wrong metadata server (a partitioned node)
- Visibility:
  - #83236: How to see mysqlrouter membership status?

MySQL Router (Beta)

MySQL Router is part of MySQL InnoDB Cluster

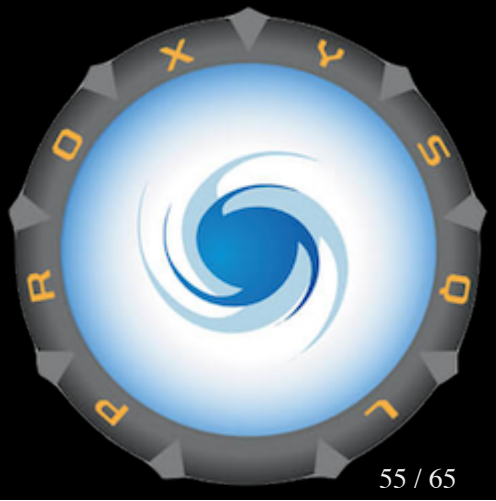
I briefly evaluated, but quickly ran into serious problems:

- missing a lot of features
- pretty unknown in the community
- ...:
  - #83237: mysqlrouter connects to wrong metadata server (a partitioned node)
- Visibility:
  - #83236: How to see mysqlrouter membership status?

I might consider to re-evaluate, but first ^^

ProxySQL

- Really Open Source!
- Becoming very popular
- Example Implementation http://lefred.be/content/ha-with-mysql-group-replication-and-proxysql/
  - Careful: #2: using multiple hostgroups/schedulers with `proxysql_groupreplication_checker.sh` can cause unwanted state changes



MySQL Group Replication

Improvements

#84784 - Nodes Do Not Reconnect

Nodes **do not reconnect** to the group replication **once they got disconnected**, **causing nodes to drop** from the cluster and can lead to **losing** the whole cluster **availability**

Improvements

Improvements

- Reduce impact on applications:
  - #84731: mysql client connections get stuck during GR start

Improvements

- Reduce impact on applications:
  - #84731: mysql client connections get stuck during GR start
- Partition Tolerance issues, split brain cannot be prevented:
  - #84727: partitioned nodes still accept writes: queries hang
  - #84728: GR failure at start still starts MySQL
  - #84729: block reads on partitioned nodes *
  - #84733: not possible to start with `super_read_only=1`
  - #84784: Nodes Do Not Reconnect
  - #84795: `STOP GROUP_REPLICATION` sets `super_read_only=off`

Improvements

- Stability:
  - #84785: Prevent Large Transactions in Group Replication
  - #84792: Member using 100% CPU in idle cluster
  - #84796: GR Member status is wrong

Improvements

- Stability:
  - #84785: Prevent Large Transactions in Group Replication
  - #84792: Member using 100% CPU in idle cluster
  - #84796: GR Member status is wrong
- Usability:
  - #84674: unresolved hostnames block GR from starting
  - #84794: cannot kill query that is stuck inside GR
  - #84799: mysqldump --single-transaction uses savepoints, does not work with GR
  - #84798: Group Replication can use some verbosity in the error log

For Every Bug Fixed: 1 Beer

MySQL Group Replication

Summary

Summary

Use Cases for Group Replication

- Environments with strict durability requirements
  - **Ensure split-brain can be completely avoided**
- Write to multiple nodes
  ('scalability' by splitting write/read workloads)
  - Not recommended (yet)
- Improve failover time
  - **Reducing impact on applications can be improved**
- ...