MariaDB Galera Cluster Best Practices

Nirbhay Choubey
work for MariaDB Corporation
Twitter: @nirbhay_c
Blog: http://nirbhay.in

Table Of Contents

A quick introduction to MariaDB Galera Cluster

- Synchronous replication
- Active-active multi-master topology
- Read/write to any cluster node
- Automatic membership control
- True parallel replication
- Direct client connection, native mysql look & feel
- Incredibly easy to setup
- Versions
- Packages available for all major linux distributions
  - apt-get install mariadb-galera-server
  - yum install MariaDB-Galera-server
  - zypper install MariaDB-Galera-server

Setting up the cluster

Mandatory settings

- `wsrep_provider`
  - libgalera_smm.so
  - none == vanilla MariaDB server
- `wsrep_cluster_address`
  - more on this later…
- `binlog_format` = ROW
- `default_storage_engine` = InnoDB

Number of nodes?

- **Odd isn't really ODD**
- galera arbitrator (stateless)

Bootstrapping the cluster

- service mysql bootstrap
- service mysql start – wsrep-new-cluster
- wsrep_cluster_address=gcomm://

State transfer

the donor-joiner thing

Kinds

- Snapshot state transfer (SST)
- Incremental state transfer (IST)

Snapshot state transfer (SST)

- SSL methods
  - `wsrep_sst_rsync`
  - `wsrep_sst_xtrabackup`
  - `wsrep_sst_xtrabackup-v2`
  - `wsrep_sst_mysqldump`
- SST API (implement/propose one!)
- `wsrep_sst_donor`=<donor-list>

Incremental state transfer (IST)

- gcahe buffer
- Always preferred

Schema upgrades

Applications evolve over time, do does their schema

Methods (wsrep_OSU_method)

- Total order isolation (TOI)
- Rolling schema upgrade (RSU)

Total order isolation

- Default method
- Master node detects and replicates DDL during parsing
- Processed at the same ‘slot’ on all the nodes (thus, total order)
- Uses STATEMENT binlog format

Rolling schema upgrade

- Node is desynced
- Incoming writesets are buffered
- Nothing gets replicated out of the node
- Post-DDL, the node joins back
- Manually execute DDL on each node
- Changes should be backward compatible

Securing Galera traffic

Encrypted replication traffic using SSL

- Enable SSL using `wsrep_provider_options`
  - socket.ssl_cert
  - socket.ssl_key

- Same cert/key on all the nodes
- IST is encrypted too
- **`SST, by default, isn't`**

SST scripts can enable encrypt

- `wsrep_sst_xtrabackup(-v2)` support encryption
- `wsrep_sst_rsync` and `wsrep_sst_mysqldump` do not

Parallel replication

multiple applier threads

What is applier thread?

- `wsrep_slave_threads` = N

How many applier threads?

- `wsrep_cert_deps_distance`
- Maximum of ~4 x #CPUCores

What about MyISAM table updates?

yes, replication of MyISAM updates work, but:

- `wsrep_replicate_myisam` = ON
- Its **experimental**
- Why?
  - cuz MyISAM is *non-transactional**

Load balancing

Multiple options available

- XAProxy
  - clustercheck script (returns “200 OK” or “503 Service unavailable”)
  - MaxScale
  - GLB

Load balancing policies

- read/write splitting
- round robin
- least-connected

Bracing for disaster

Using galera cluster as master

- `log-bin`
- `log-slave-updates`
- `server-id` (same across all nodes)
- `gtid-domain-id`
- `wsrep-gtid-mode` (introduced in 10.1)
- `wsrep_gtid_domain-id` (introduced in 10.1)

Understanding limitations

- Tables should have a primary key
  - – `innodb-force-primary-key` (introduced in 10.1.0)
- Only InnoDB storage engine is supported
- Transaction size
  - `wsrep_max_ws_size` = 1G
  - ~~`wsrep_max_ws_rows`~~ = 128K

Troubleshooting

Network partitioning/Split-brain

- Even number od nodes
- garbd
- … as was mentioned earlier

Multi-master conflicts

- Optimistic concurrency control
- Victim trx is abortem with deadlock error
  - Application should have retry logic
  - `wsrep_retry_autocommit` = N (works only with autocommit transactions)

- Diagnosis
  - `wsrep_log_conflicts`, `wsrep_local_bf_aborts`, `wsrep_local_cert_failures`

Applier failures

- GRA_X_X.log file
  - Headless binlog

- GRA_X_X_v2.log
  - Automatinally includes binlog header
  - Introduced in 10.1.4

> $ mysqlbinlog GRA_X_X.log

Detecting slow nodes

- `wsrep_flow_control_sent`
- `wsrep_local_recv_queue`

> Upper limit : gcs.fc_limit
> Lower limit : gcs.fc_limit * gcs.fc_factor

Pit falls

Non-sequential auto-increment keys

- Application should be aware of this
- `wsrep_auto_increment_control` = 0 | 1

Principle of least variation

- SST method
- SSL setting
- etc…

MariaDB project

- Source: https://github.com/MariaDB/
- Documentation: https://mariadb.com/kb/
- Report bugs/FRs: mariadb.org/jira
- Discussion mailing list: **maria-discuss@lists.launchpad.net**
- IRC: #maria (freenode)

Note Time 2015.11.10 18:33 Tuesday