MariaDB Galera as Master to Master replication cluster on Ubuntu 16.04 LTS

March 5, 2017

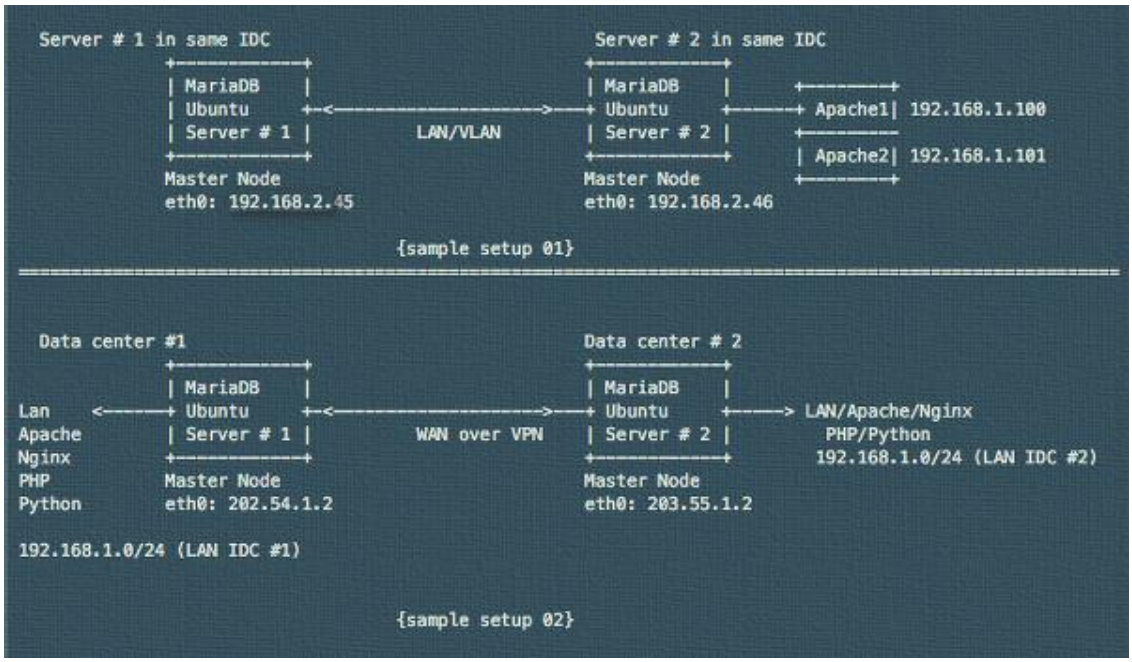
How do I install and configure MariaDB Galera master to master cluster on Ubuntu Linux 16.04 LTS server to get both read and write scalability?

MariaDB Galera Cluster is an open source and free synchronous multi-master cluster for MariaDB database. It is available on Linux only and only supports the XtraDB/InnoDB storage engines. There is experimental support for MyISAM, but it is not well tested. Starting with MariaDB 10.1, the wsrep API for Galera Cluster is included by default.

Benefits of Galera cluster

1. True Multi-master Read and write to any node at any time.
2. Synchronous Replication No slave lag, no data is lost at node crash.
3. Tightly Coupled All nodes hold the same state. No diverged data between nodes allowed.
4. Multi-threaded Slave For better performance. For any workload.
5. No Master-Slave Failover Operations or Use of VIP.
6. Hot Standby No downtime during failover (since there is no failover).
7. Automatic Node Provisioning No need to manually back up the database and copy it to the new node.
8. Supports InnoDB.
9. Transparent to Applications Required no (or minimal) changes) to the application.
10. No Read and Write Splitting Needed.
11. The result is a high-availability solution that is both robust in terms of data integrity and high-performance with instant failovers.

Our sample setup (Fig: 1)



Galera Cluster for MariaDB is an easy-to-use, high-availability solution, which provides high system up-time, no data loss, and scalability for future growth. You can setup it as follows:

1. Between two data centers (wan data center cluster) – For security purpose, you must setup VPN and MariaDB over SSL.
2. Within your LAN/VLAN (single data center cluster) – VPN is not needed but you must setup MariaDB over SSL.

What you need to setup MariaDB Galera high availability cluster?

1. Minimum two servers (can be a cloud or bare metal boxes). For production use four servers (minimum three servers recommended).
2. A private network (LAN/VLAN) between servers
3. A VPN between two data center if setting between two IDCs
4. Ubuntu Linux 16.04 LTS on both servers

Okay enough talk, lets get started with MariaDB Galera installation and configuration on a Ubuntu.

Step 1 – Setup /etc/hosts

First setup /etc/hosts files on both servers:

```
$ sudo vi /etc/hosts
```

Set correct private IP address as per fig.01 or as per your setup:

```
192.168.2.45    mdb01
192.168.2.46     mdb02
```

Close and save the file. Test it:

```
$ ping -c2 mdb01
$ ping -c2 mdb02
```

Sample outputs:

```
PING mdb02 (192.168.2.46) 56(84) bytes of data.
64 bytes from gfs02 (192.168.2.46): icmp_seq=1 ttl=64 time=0.487 ms
64 bytes from gfs02 (192.168.2.46): icmp_seq=2 ttl=64 time=0.497 ms

--- mdb02 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.487/0.492/0.497/0.005 ms
```

Step 2 – Enable the MariaDB repositories for version 10.1

Type the following commands to enable mariadb repositories to install 10.1 version on both servers:

```
$ sudo apt-get install software-properties-common
$ sudo apt-key adv --recv-keys --keyserver hkp://keyserver.ubuntu.com:80
0xF1656F24C74CD1D8
$ sudo add-apt-repository 'deb [arch=amd64,i386,ppc64el]
http://mirror.lstn.net/mariadb/repo/10.1/ubuntu xenial main'
$ apt-get update
```

Sample outputs:

```
root@ubuntu-box3:~# apt-get install software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
software-properties-common is already the newest version (0.96.20.5).
The following packages were automatically installed and are no longer required:
  libacl1-dev libattr1-dev libc-dev-bin libc6-dev linux-libc-dev manpages-dev
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 6 not upgraded.
root@ubuntu-box3:~# apt-key adv --recv-keys --keyserver hkp://keyserver.ubuntu.com:80 0xF1656F24C74CD1D8
Executing: /tmp/tmp.PFRisXXC0v/gpg.1.sh --recv-keys
--keyserver
hkp://keyserver.ubuntu.com:80
0xF1656F24C74CD1D8
gpg: requesting key C74CD1D8 from hkp server keyserver.ubuntu.com
gpg: key C74CD1D8: "MariaDB Signing Key <signing-key@mariadb.org>" not changed
gpg: Total number processed: 1
gpg:              unchanged: 1
root@ubuntu-box3:~# add-apt-repository 'deb [arch=amd64,i386,ppc64el] http://download.nus.edu.sg/mirror/mariadb/repo/10.1/ubuntu xenial main'
root@ubuntu-box3:~#
```

Fig.02: Setting up MariaDB Repositories

Step 3 – Install the MariaDB server v10.1 on Ubuntu Linux

Type the following apt-get command/apt command on both servers:

```
$ sudo apt-get install mariadb-server rsync
```

Sample outputs:

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libacl1-dev libattr1-dev libc-dev-bin libc6-dev linux-libc-dev manpages-dev
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  galera-3 iproute libcgi-fast-perl libcgi-pm-perl libdbd-mysql-perl libdbi-perl libencode-locale-perl
  libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl libjemalloc1 liblwp-med
  libtimedate-perl liburi-perl mariadb-client-10.1 mariadb-client-core-10.1 mariadb-common mariadb-ser
Suggested packages:
  libclone-perl libmldbm-perl libnet-daemon-perl libsql-statement-perl libdata-dump-perl libipc-shared
The following NEW packages will be installed:
  galera-3 iproute libcgi-fast-perl libcgi-pm-perl libdbd-mysql-perl libdbi-perl libencode-locale-perl
  libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl libjemalloc1 liblwp-med
  libtimedate-perl liburi-perl mariadb-client-10.1 mariadb-client-core-10.1 mariadb-common mariadb-ser
0 upgraded, 29 newly installed, 0 to remove and 6 not upgraded.
Need to get 24.0 MB of archives.
After this operation, 194 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://mirror.lstn.net/mariadb/repo/10.1/ubuntu xenial/main amd64 mysql-common all 10.1.21+maria
Get:2 http://mirror.lstn.net/mariadb/repo/10.1/ubuntu xenial/main amd64 mariadb-common all 10.1.21+mar
Get:3 http://in.archive.ubuntu.com/ubuntu xenial/main amd64 iproute all 1:4.3.0-1ubuntu3 [2,428 B]
Get:4 http://mirror.lstn.net/mariadb/repo/10.1/ubuntu xenial/main amd64 galera-3 amd64 25.3.19-xenial
....
...
..
Setting up mariadb-client-core-10.1 (10.1.21+maria-1~xenial) ...
Setting up mariadb-client-10.1 (10.1.21+maria-1~xenial) ...
Setting up mariadb-server-core-10.1 (10.1.21+maria-1~xenial) ...
Setting up mariadb-server-10.1 (10.1.21+maria-1~xenial) ...
2017-03-06  1:01:29 140562209519872 [Note] /usr/sbin/mysqld (mysqld 10.1.21-MariaDB-1~xenial) starting
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: Using mutexes to ref count buffer pool pages
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: The InnoDB memory heap is disabled
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: Mutexes and rw_locks use GCC atomic builtins
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: GCC builtin __atomic_thread_fence() is used for mem
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: Compressed tables use zlib 1.2.8
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: Using Linux native AIO
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: Using SSE crc32 instructions
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: Initializing buffer pool, size = 256.0M
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: Completed initialization of buffer pool
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: Highest supported file format is Barracuda.
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: 128 rollback segment(s) are active.
2017-03-06  1:01:29 140562209519872 [Note] InnoDB: Waiting for purge to start
2017-03-06  1:01:29 140562209519872 [Note] InnoDB:  Percona XtraDB (http://www.percona.com) 5.6.34-79.
2017-03-06  1:01:29 140562209519872 [Note] Plugin 'FEEDBACK' is disabled.
2017-03-06  1:01:29 140561294001920 [Note] InnoDB: Dumping buffer pool(s) not yet started
Setting up mariadb-server (10.1.21+maria-1~xenial) ...
Processing triggers for libc-bin (2.23-0ubuntu5) ...
Processing triggers for systemd (229-4ubuntu16) ...
Processing triggers for ureadahead (0.100.0-19) ...
```

Fig.03: Install MariaDB with apt/apt-get command

You must setup MariaDB root password:

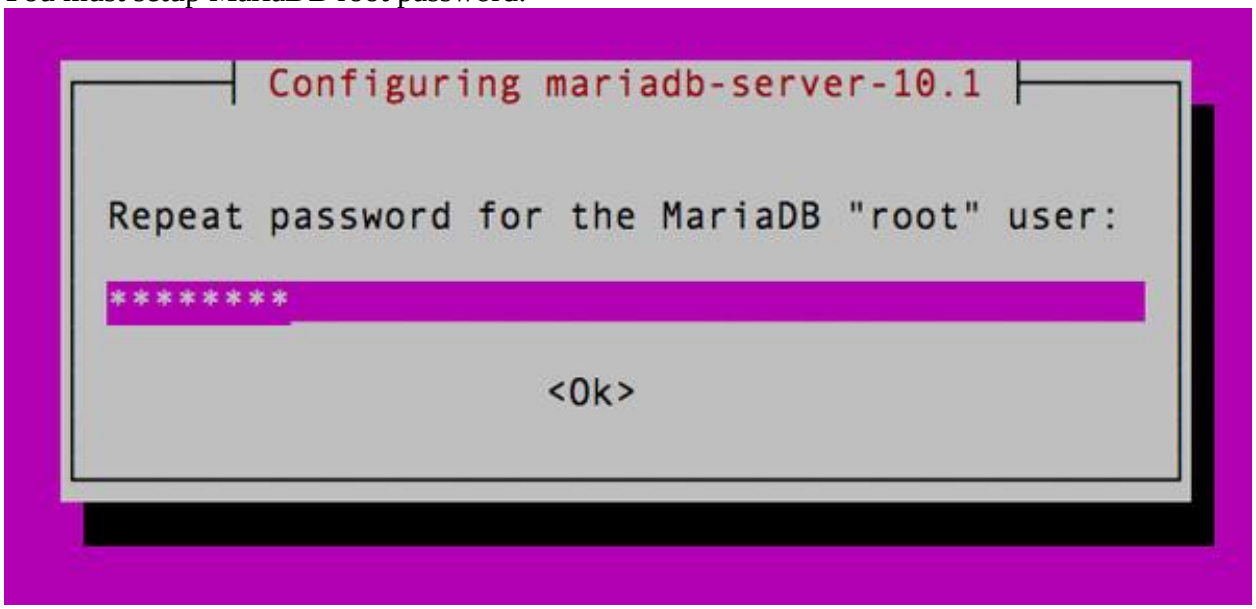


Fig.04: Confirm the MariaDB “root” user password

Step 4 – Configure the MariaDB Galera cluster server

Create the following file on both servers:

```
$ sudo vi /etc/mysql/conf.d/galera.cnf
```

Append the following text on **mdb01** server:

```
[mysqld]
#mysql settings
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
query_cache_size=0
query_cache_type=0
innodb_flush_log_at_trx_commit=0
innodb_buffer_pool_size=256M
bind-address=192.168.2.45

#Galera settings
wsrep_provider="/usr/lib/galera/libgalera_smm.so"
wsrep_cluster_name="cbz_cluster"
wsrep_cluster_address="gcomm://192.168.2.45,192.168.2.46"
wsrep_sst_method=rsync
wsrep_on=ON
wsrep_node_address="192.168.2.45"
wsrep_node_name="mdb01"
```

Append the following text on **mdb02** server:

```
[mysqld]
#mysql settings
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
query_cache_size=0
query_cache_type=0
innodb_flush_log_at_trx_commit=0
```

```
innodb_buffer_pool_size=256M
bind-address=192.168.2.46

#Galera settings
wsrep_provider="/usr/lib/galera/libgalera_smm.so"
wsrep_cluster_name="cbz_cluster"
wsrep_cluster_address="gcomm://192.168.2.45,192.168.2.46"
wsrep_sst_method=rsync
wsrep_on=ON
wsrep_node_address="192.168.2.46"
wsrep_node_name="mdb02"
```

Save and close the file.

Step 5 – Start the MariaDB cluster on mdb01 server

Type the following command:

```
$ sudo systemctl stop mysql
```

Bootstrapping the cluster

Bootstrapping is nothing but staring the initial cluster. Type the following command on **mdb01** server:

```
$ sudo /usr/bin/galera_new_cluster
```

Verify that MySQL started:

```
$ ps aux | grep mysql
mysql 10587 14.0 15.5 1266120 155268 ? Ssl 01:50 0:00 /usr/sbin/mysqld --wsrep-new-cluster --
wsrep_start_position=00000000-0000-0000-0000-000000000000:-1
root 20822 0.0 0.0 12948 980 pts/0 S+ 01:22 0:00 grep --color=auto mysql
```

Type the following command to verify that cluster is up and first node is running:

```
$ mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size';"
```

Sample outputs:

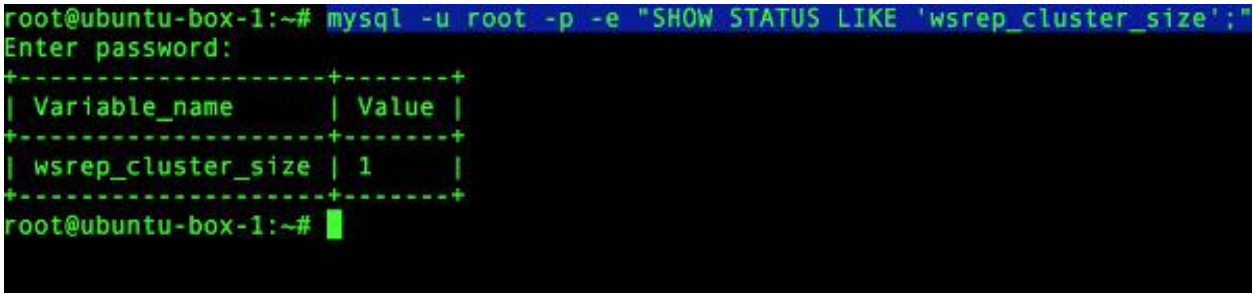


Fig.05: Verify that cluster is up and running

Step 6 – Join the MariaDB cluster on mdb02 server

Type the following command on **mdb02** server to join it to the *mdb01 cluster*:

```
$ sudo systemctl mysql stop
$ sudo systemctl mysql start
```

Verify it:

```
$ mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size';"
```

Sample size:

```
Enter password:
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 2     |
+--------------------+-------+
```

Please note that value 2 indicate that our cluster has two master to master node. If you join 3rd node, it should be as follows:

```
Enter password:
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 3     |
+--------------------+-------+
```

The following shows more info about your cluster:

```
$ mysql -u root -p -e "show status like 'wsrep%';"
```

```
Enter password:
+------------------------------+--------------------------------------+
| Variable_name                | Value                                |
+------------------------------+--------------------------------------+
| wsrep_apply_oooe             | 0.000000                             |
| wsrep_apply_oool             | 0.000000                             |
| wsrep_apply_window           | 0.000000                             |
| wsrep_causal_reads           | 0                                    |
| wsrep_cert_deps_distance     | 0.000000                             |
| wsrep_cert_index_size        | 0                                    |
| wsrep_cert_interval          | 0.000000                             |
| wsrep_cluster_conf_id        | 2                                    |
| wsrep_cluster_size           | 2                                    |
| wsrep_cluster_state_uuid     | 1ff614b2-01e1-11e7-8e6b-6613e80d4934 |
| wsrep_cluster_status         | Primary                              |
| wsrep_commit_oooe            | 0.000000                             |
| wsrep_commit_oool            | 0.000000                             |
| wsrep_commit_window          | 0.000000                             |
| wsrep_connected              | ON                                   |
| wsrep_desync_count           | 0                                    |
| wsrep_evs_delayed            |                                      |
| wsrep_evs_evict_list         |                                      |
| wsrep_evs_repl_latency       | 0/0/0/0/0                            |
| wsrep_evs_state              | OPERATIONAL                          |
| wsrep_flow_control_paused    | 0.000000                             |
| wsrep_flow_control_paused_ns | 0                                    |
| wsrep_flow_control_recv      | 0                                    |
| wsrep_flow_control_sent      | 0                                    |
| wsrep_gcomm_uuid             | 1ff51705-01e1-11e7-96a2-ca7fb2d337c2 |
| wsrep_incoming_addresses     | 192.168.2.46:3306,192.168.2.45:3306  |
| wsrep_last_committed         | 0                                    |
| wsrep_local_bf_aborts        | 0                                    |
| wsrep_local_cached_downto    | 18446744073709551615                 |
| wsrep_local_cert_failures    | 0                                    |
| wsrep_local_commits          | 0                                    |
| wsrep_local_index            | 1                                    |
| wsrep_local_recv_queue       | 0                                    |
| wsrep_local_recv_queue_avg   | 0.000000                             |
| wsrep_local_recv_queue_max   | 1                                    |
```

```
| wsrep_local_recv_queue_min   | 0                                    |
| wsrep_local_replays          | 0                                    |
| wsrep_local_send_queue       | 0                                    |
| wsrep_local_send_queue_avg   | 0.000000                             |
| wsrep_local_send_queue_max   | 1                                    |
| wsrep_local_send_queue_min   | 0                                    |
| wsrep_local_state            | 4                                    |
| wsrep_local_state_comment    | Synced                               |
| wsrep_local_state_uuid       | 1ff614b2-01e1-11e7-8e6b-6613e80d4934 |
| wsrep_protocol_version       | 7                                    |
| wsrep_provider_name          | Galera                               |
| wsrep_provider_vendor        | Codership Oy     |
| wsrep_provider_version       | 25.3.19(r3667)                       |
| wsrep_ready                  | ON                                   |
| wsrep_received               | 6                                    |
| wsrep_received_bytes         | 407                                  |
| wsrep_repl_data_bytes        | 0                                    |
| wsrep_repl_keys              | 0                                    |
| wsrep_repl_keys_bytes        | 0                                    |
| wsrep_repl_other_bytes       | 0                                    |
| wsrep_replicated             | 0                                    |
| wsrep_replicated_bytes       | 0                                    |
| wsrep_thread_count           | 2                                    |
+------------------------------+--------------------------------------+
```

You can now create database and table on any node and it will get replicated on both nodes:

```
{vivek@mdb01:~ }$ mysql -u root -p -e 'create database foobar;'
```

From second node verify it:

```
{vivek@mdb02:~ }$ mysql -u root -p -e 'show databases;'
```

Sample outputs:

```
Enter password:
+--------------------+
| Database           |
+--------------------+
| demo               |
| foobar             |
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
```

You can migrate from stand alone MariaDB to clustered setup as follows:

```
## On old MariaDB server ##
## Dump database named foobar without engine ##
$ mysqldump -u root -p --skip-create-options foobar > foobar.sql
## Copy foobar.sql to any one of the clustered node named mdb02 ##
$ scp foobar.sql 192.168.2.46:/root/
## Restore database, from mdb02 ##
$ mysql -u root -p foobar < foobar.sql
```

For more information see Galera Cluster, MaraiDB documentation and MariaDB Galera Cluster known Limitations.