GNU/Linux Administration - Support #337

Setting Up A MariaDB Galera Cluster on Debian 7

02/14/2014 12:14 PM - Daniel Curtis

Status:	Closed	**Start date:**	02/14/2014
Priority:	Normal	**Due date:**	
Assignee:	Daniel Curtis	**% Done:**	100%
Category:		**Estimated time:**	1.50 hour
Target version:		**Spent time:**	3.50 hours

Description

Here's step-by-step guide to install and configure a MariaDB Galera Clustered Database server software on Debian 7 running across a 3 Node Cluster (3 nodes to increase DB consistency, response time & reduce split-brain occurrences of full DB re-syncs).

Also note that currently (in MariaDB 5.5 at least) the MariaDB Galera Cluster only supports the InnoDB/XtraDB storage engine, so ensure that your Databases to be Clustered are InnoDB based for DB type.

Some Linux or system level requirements or pre-install setup tasks are:

1. Ensure the /etc/hosts files are the same on each Node and that it contains entries for each DB Node's IP addresses and hostname (both the node hostname and its fully qualified hostname).

   ```
   vi /etc/hosts
   ```

   - And add/modify it similar to the following

   ```
   192.168.0.1 db1 db1.example.com
   192.168.0.2 db2 db2.example.com
   192.168.0.3 db3 db3.example.com
   ```

2. Ensure that each Node's Firewall permits each other node host IP addresses access to ports 4444 and 4567 and the standard MySQL port of 3306.

Configure Node 1

- Install prerequisite Debian Packages:

  ```
  apt-get -y install python-software-properties rsync
  ```

- Setup the MariaDB 5.5 Repository:

  ```
  apt-key adv --recv-keys --keyserver keyserver.ubuntu.com 0xcbcb082a1bb943db
  add-apt-repository 'deb http://ftp.osuosl.org/pub/mariadb/repo/5.5/debian wheezy main'
  ```

- Install MariaDB Galera:

  ```
  apt-get install mariadb-galera-server galera
  ```

Once all 3 Nodes have the above software installed and the initial MySQL settings applied, bring them all up and online (mysql running with the default Debian settings on each node is ok at this point).

Install the relevant Database content to the "first" Node, which we'll call "Node1″ or reference as "db1″, this may well involve using and knowing the MySQL 'root' username and its password.

- After that and on Node 1, restart its MySQL software as the initiating node of a Galera Cluster by loading the following

configuration file(s) and restarting Node1's MySQL server software, similar to the following:

```
vi /etc/mysql/conf.d/cluster.cnf
```

- And add the following:

```
[mysqld]
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0

# Galera Provider Configuration
wsrep_provider=/usr/lib/galera/libgalera_smm.so
#wsrep_provider_options="gcache.size=32G"

# Galera Cluster Configuration
wsrep_cluster_name="example_cluster"

# Initialize the cluster; switch after --wsrep-new-cluster has been run and other nodes ha
ve joined.
wsrep_cluster_address="gcomm://

# Set cluster node addresses (hostnames work too)
#wsrep_cluster_address="gcomm://192.168.0.1,192.168.0.2,192.168.0.3"

# Galera Synchronization Congifuration
wsrep_sst_method=rsync
#wsrep_sst_auth=user:pass

# Galera Node Configuration
wsrep_node_address="192.168.0.1"
wsrep_node_name="db1.example.com"
```

- Restart the MySQL Server on Node 1 with the above /etc/mysql/conf.d/cluster.cnf file installed:

```
service mysql stop
service mysql start --wsrep-new-cluster
```

- Check how MySQL Galera Cluster servers is going after restart with:

```
tail -n 20 /var/log/mysql/mysqld_safe.log
```

Configure Node 2

- Install prerequisite Debian Packages:

```
apt-get -y install python-software-properties rsync
```

- Setup the MariaDB 5.5 Repository:

```
apt-key adv --recv-keys --keyserver keyserver.ubuntu.com 0xcbcb082a1bb943db
add-apt-repository 'deb http://ftp.osuosl.org/pub/mariadb/repo/5.5/debian wheezy main'
```

- Install MariaDB Galera:

```
apt-get install mariadb-galera-server galera
```

With Node 1 now running a single-node Galera Cluster, you can configure and start Nodes 2 and 3 with these following commands done and verified on Node 2 and then Node 3:

- Load MariaDB Cluster config file, listing Node 1's IP address as the initial 'gcomm' Cluster server, on Node 2 do:

```
vi /etc/mysql/conf.d/cluster.cnf
```

  - And add the following:

```
[mysqld]
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0

# Galera Provider Configuration
wsrep_provider=/usr/lib/galera/libgalera_smm.so
#wsrep_provider_options="gcache.size=32G"

# Galera Cluster Configuration
wsrep_cluster_name="example_cluster"
wsrep_cluster_address="gcomm://192.168.0.1,192.168.0.2,192.168.0.3"

# Galera Synchronization Congifuration
wsrep_sst_method=rsync
#wsrep_sst_auth=user:pass

# Galera Node Configuration
wsrep_node_address="192.168.0.1"
wsrep_node_name="db2.example.com"
```

- Restart the MySQL Server on Node 2 with the above /etc/mysql/conf.d/cluster.cnf file installed:

```
service mysql restart
```

- Check how MySQL Galera Cluster servers is going after restart with:
  Note: Any UUID numbers are replaced with '…' in the following output

```
cat /var/log/mysql/mysqld_safe.log
```

  - You should see log file entries about each node joining and being syncronised into the Galera MySQL Cluster on Node 1, entries like:

```
[Note] WSREP: declaring ... stable
[Note] WSREP: Node ... state prim
[Note] WSREP: (..., 'tcp://0.0.0.0:4567') turning message relay requesting on, nonlive pee
rs: tcp://192.168.0.3:4567
[Note] WSREP: view(view_id(PRIM,...,11) memb {
...,
...,
} joined {
} left {
} partitioned {
...,
})
```

```
[Note] WSREP: New COMPONENT: primary = yes, bootstrap = no, my_idx = 1, memb_num = 2
[Note] WSREP: STATE EXCHANGE: Waiting for state UUID.
[Note] WSREP: forgetting ... (tcp://192.168.0.3:4567)
[Note] WSREP: (..., 'tcp://0.0.0.0:4567') turning message relay requesting off
[Note] WSREP: STATE EXCHANGE: sent state msg: ...
[Note] WSREP: STATE EXCHANGE: got state msg: ... from 0 (node1.domain.name)
[Note] WSREP: STATE EXCHANGE: got state msg: ... from 1 (node2.domain.name)
[Note] WSREP: Quorum results:
version = 2,
component = PRIMARY,
conf_id = 8,
members = 2/2 (joined/total),
act_id = 162,
last_appl. = 0,
protocols = 0/4/2 (gcs/repl/appl),
group UUID = ...
[Note] WSREP: Flow-control interval: [23, 23]
[Note] WSREP: New cluster view: global state: ...:162, view# 9: Primary, number of nodes:
2, my index: 1, protocol version 2
[Note] WSREP: wsrep_notify_cmd is not defined, skipping notification.
[Note] WSREP: Assign initial position for certification: 162, protocol version: 2
14:24:37 [Note] WSREP: cleaning up ... (tcp://192.168.0.3:4567)
[Note] WSREP: declaring ... stable
[Note] WSREP: declaring ... stable
[Note] WSREP: Node ... state prim
[Note] WSREP: view(view_id(PRIM,...,12) memb {
...,
...,
...,
} joined {
} left {
} partitioned {
})
[Note] WSREP: New COMPONENT: primary = yes, bootstrap = no, my_idx = 1, memb_num = 3
[Note] WSREP: STATE EXCHANGE: Waiting for state UUID.
[Note] WSREP: STATE EXCHANGE: sent state msg: ...
[Note] WSREP: STATE EXCHANGE: got state msg: ... from 0 (node1.domain.name)
[Note] WSREP: STATE EXCHANGE: got state msg: ... from 1 (node2.domain.name)
[Note] WSREP: STATE EXCHANGE: got state msg: ... from 2 (node3.domain.name)
[Note] WSREP: Quorum results:
version = 2,
component = PRIMARY,
conf_id = 9,
members = 3/3 (joined/total),
act_id = 162,
last_appl. = 0,
protocols = 0/4/2 (gcs/repl/appl),
group UUID = ...
[Note] WSREP: Flow-control interval: [28, 28]
[Note] WSREP: New cluster view: global state: ...:162, view# 10: Primary, number of nodes:
 3, my index: 1, protocol version 2
[Note] WSREP: wsrep_notify_cmd is not defined, skipping notification.
[Note] WSREP: Assign initial position for certification: 162, protocol version: 2
[Note] WSREP: Member 2 (node3.domain.name) synced with group.
```

- Again, use the mysql client on Node 2 to query Node 2's Galera Server.

```
mysql -uroot -p -e "SHOW STATUS LIKE 'wsrep_%';"
```

With Node 1 and 2 now running a dual-node Galera Cluster, you can configure and start Node 3 with these following commands done and verified on Node 1 and then Node 2.

Configure Node 3

- Install prerequisite Debian Packages:

```
apt-get -y install python-software-properties rsync
```

- Setup the MariaDB 5.5 Repository:

```
apt-key adv --recv-keys --keyserver keyserver.ubuntu.com 0xcbcb082a1bb943db
add-apt-repository 'deb http://ftp.osuosl.org/pub/mariadb/repo/5.5/debian wheezy main'
```

- Install MariaDB Galera:

```
apt-get install mariadb-galera-server galera
```

- Load MariaDB Cluster config file, listing Node 1 and 2′s IP address as the initial ‘gcomm’ Cluster server, on Node 3 do:

```
vi /etc/mysql/conf.d/cluster.cnf
```

  - And add the following:

```
[mysqld]
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0

# Galera Provider Configuration
wsrep_provider=/usr/lib/galera/libgalera_smm.so
#wsrep_provider_options="gcache.size=32G"

# Galera Cluster Configuration
wsrep_cluster_name="example_cluster"
wsrep_cluster_address="gcomm://192.168.0.1,192.168.0.2,192.168.0.3"

# Galera Synchronization Congifuration
wsrep_sst_method=rsync
#wsrep_sst_auth=user:pass

# Galera Node Configuration
wsrep_node_address="192.168.0.1"
wsrep_node_name="db3.example.com"
```

- Restart the MySQL Server on Node 3 with the above /etc/mysql/conf.d/cluster.cnf file installed:

```
service mysql restart
```

Check the mysqld_safe.log logfiles on all nodes to see that Node 3 also joins and syncronises with the Cluster.

At this point you have all 3 x Nodes in the Galera MySQL Cluster and you can now update Node 1 and Node 2′s cluster.cnf file to set the IP Addresses of all Nodes in the Cluster.

- On Node 1, change the line in /etc/mysql/conf.d/cluster.cnf that says:

```
wsrep_cluster_address = 'gcomm://192.168.0.1,192.168.0.2,192.168.0.3'
```

And then restart MySQL on Node 1, checking the relevant log files and mysql client command to query the Cluster membership once Node 1's mysql server software has restarted, with:

```
service mysql restart
```

- Check the log for any problems:

  ```
  tail -f /var/log/mysql/mysqld_safe.log
  ```

- Check the status of the Galera cluster

  ```
  mysql -uroot -p -e "SHOW STATUS LIKE 'wsrep_%';"
  ```

Finalize the Galera Cluster configuration

Once Node 1 has been reconfigured to know of all Cluster Members, do the same step above on Node 2 and Node 3, setting Node 2 and Node 3's **wsrep_cluster_address** in /etc/mysql/conf.d/cluster.cnf to:

```
wsrep_cluster_address = 'gcomm://192.168.0.1,192.168.0.2,192.168.0.3'
```

And restarting their MySQL Servers and checking that they then each rejoin the live Cluster.

Finally by now you should have a running live 3 Node MariaDB Galera MySQL Cluster, the status of which on each Node should list that there's 3 members of the Cluster, via:

```
mysql -uroot -pMYSQLROOTPASSWD -e "SHOW STATUS LIKE 'wsrep_%';" | egrep "wsrep_incoming_addresses|
wsrep_cluster_size"
```

Transferring the Debian Maintenance Configuration

Currently, Ubuntu and Debian's MariaDB servers use a special maintenance user to do routine maintenance. Some tasks that fall outside of the maintenance category also are run as this user, including important functions like stopping MySQL.

With our cluster environment being shared between the individual nodes, the maintenance user, who has randomly generated login credentials on each node, will be unable to execute commands correctly. Only the initial server will have the correct maintenance credentials, since the others will attempt to use their local settings to access the shared cluster environment.This can fix this by simply copying the contents of the maintenance file to each individual node:

- On server that bootstrapped the cluster, open the Debian maintenance configuration file:

  ```
  sudo nano /etc/mysql/debian.cnf
  ```

  - The file will look similar to the following:

    ```
    [client]
    host     = localhost
    user     = debian-sys-maint
    password = VRP84dIknkX31uOf
    socket   = /var/run/mysqld/mysqld.sock
    [mysql_upgrade]
    host     = localhost
    user     = debian-sys-maint
    password = VRP84dIknkX31uOf
    socket   = /var/run/mysqld/mysqld.sock
    basedir  = /usr
    ```

- We simply need to copy this information and paste it into the same file on each node.

Automatic Cluster Initialization at Boot

Newer versions of Galera have options that can enable the cluster to start at boot. If you shut down all nodes, you effectively terminated the cluster (not the data of course, but the running cluster), hence the right way is to start the all the nodes with gcomm://<node1 address>,<node2 address>,...?pc.wait_prim=no again.

- On ever node in the Galera cluster edit the mysql cluster configuration file

```
vi /etc/mysql/conf.d/cluster.cnf
```

  - And modify the wsrep_cluster_address paramter:

```
wsrep_cluster_address="gcomm://192.168.0.1,192.168.0.2,192.168.0.3?pc.wait_prim=no"
```

- On one of the nodes set global wsrep_provider_options="pc.bootstrap=true".

```
vi /etc/mysql/conf.d/cluster.cnf
```

  - And add the wsrep_provider_options parameter:

```
wsrep_provider_options="pc.bootstrap=true"
```

BONUS - Setup Xtrabackup

Installing xtrabackup is a decent way to increase the performance and add a minor level of security to the Galera cluster. Xtrabackup is maintained by Percona and as such we need to add their repo and install it the xtrabackup program, then finally modify the cluster.cnf file to use xtrabackup.

- Add the percona repo keys

```
apt-key adv --keyserver keys.gnupg.net --recv-keys 1C4CBDCDCD2EFD2A
```

- Create the percona repo list file:

```
vi /etc/apt/sources.list.d/percona.list
```

  - And add the following:

```
deb http://repo.percona.com/apt wheezy main
deb-src http://repo.percona.com/apt wheezy main
```

- Refresh the local cache:

```
apt-get update
```

- Install xtrabackup

```
apt-get install xtrabackup
```

- On ever node in the Galera cluster edit the mysql cluster configuration file

```
vi /etc/mysql/conf.d/cluster.cnf
```

- And modify the wsrep_sst_method parameter:

```
wsrep_sst_method=xtrabackup
```

- NOTE: If using authentication make sure add the following parameter to the cluster.cnf file:

```
wsrep_sst_auth=username:SuperSecretPassword
```

Resources

- https://www.digitalocean.com/community/tutorials/how-to-configure-a-galera-cluster-with-mariadb-on-ubuntu-12-04-servers
- https://mariadb.com/kb/en/mariadb/documentation/replication/galera/getting-started-with-mariadb-galera-cluster/
- http://www.percona.com/doc/percona-xtrabackup/2.1/installation/apt_repo.html

History

#1 - 02/24/2014 03:57 PM - Daniel Curtis

- Tracker changed from Bug to Support

#2 - 12/08/2014 03:40 PM - Daniel Curtis

- Description updated

#3 - 12/10/2014 12:41 PM - Daniel Curtis

- Description updated

#4 - 12/10/2014 02:36 PM - Daniel Curtis

- Project changed from 80 to GNU/Linux Administration

#5 - 12/15/2014 02:13 PM - Daniel Curtis

- Description updated

- % Done changed from 80 to 90

#6 - 12/15/2014 02:37 PM - Daniel Curtis

- Description updated

- % Done changed from 90 to 100

#7 - 12/15/2014 02:42 PM - Daniel Curtis

- Description updated

#8 - 12/18/2014 04:10 PM - Daniel Curtis

- Description updated

- Status changed from In Progress to Resolved

#9 - 12/26/2014 07:25 PM - Daniel Curtis

- Description updated

- Status changed from Resolved to Closed