Cluster MariaDB

DB, bases de données, MariaDB, cluster, Galera, Haproxy

Infrastructure

L'infrastructure se compose de 5 VMs :

- 2 load balancers (KeepAlived + HA Proxy)
- 3 nodes MariaDB

Installation

Note : l'installation est à effectuer sur les trois noeuds.

L'installation est très simple, Galera étant maintenant installé par défaut.

```
sudo aptitude install mariadb-server
```

Configuration

Note : la configuration est à effectuer sur les trois noeuds.

Afin de sécuriser l'accès à la base de données, nous lançons le script

```
sudo mysql_secure_installation
```

Nous définissons un mot de passe pour root, interdisons l'utilisateur anonyme, interdisons l'accès avec le compte root à distance, interdisons l'accès à la vase de test.

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...
```

```
Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

Nous nous connectons en saisissant le mot de passe défini précédemment

```
sudo mysql -u root -p
```

Nous vérifions que l'accès par socket est configuré

```
USE mysql;
SELECT plugin FROM user WHERE user='root';
```

Vous devriez avoir ceci en retour

```
+-------------+
| plugin      |
+-------------+
| unix_socket |
+-------------+
1 row in set (0.00 sec)
```

Nous désactivons l'accès par socket, mettons à jour les privilèges, et sortons

```
UPDATE user SET plugin='' WHERE User='root';
FLUSH PRIVILEGES;
QUIT
```

Afin d'accéder à l'instance MariaDB depuis l'extérieur, nous éditons le fichier suivant

```
sudo vi /etc/mysql/mariadb.conf.d/50-server.cnf
```

Et mettons en commentaire cette ligne :

```
#bind-address           = 127.0.0.1
```

Configuration noeud 1

Nous stoppons MariaDB

```
sudo systemctl stop mariadb
```

Nous créons le fichier /etc/mysql/mariadb.conf.d/50-galera.cnf

```
sudo vim /etc/mysql/mariadb.conf.d/50-galera.cnf
```

Et nous y intégrons les informations suivantes

```
[galera]
binlog_format=ROW
default_storage_engine=InnoDB
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0
```

```
# Galera Provider Configuration
wsrep_on=ON
wsrep_provider='/usr/lib/galera/libgalera_smm.so'

# Galera Cluster Configuration
wsrep_cluster_name='cluster01'
wsrep_cluster_address='gcomm://'

# Galera synchronisation configuration
wsrep_sst_method=rsync

# Galera node configuration
wsrep_node_address='10.9.214.73'
wsrep_node_name='dbcl01-db01'
```

Nous démarrons MariaDB

```
sudo systemctl start mariadb
```

Nous lançons le cluster (à ne faire que sur le noeud 1, et une seule fois)

```
sudo galera_new_cluster
```

Nous nous connectons à la base pour vérifier

```
mysql -u root -p
```

Et saisissons la requête

```
SHOW GLOBAL STATUS WHERE Variable_name IN ('wsrep_ready',
'wsrep_cluster_size', 'wsrep_cluster_status', 'wsrep_connected');
```

Vous devriez avoir ce résultat

```
+----------------------+---------+
| Variable_name        | Value   |
+----------------------+---------+
| wsrep_cluster_size   | 1       |
| wsrep_cluster_status | Primary |
| wsrep_connected      | ON      |
| wsrep_ready          | ON      |
+----------------------+---------+
4 rows in set (0.01 sec)
```

Configuration noeud 2 et 3

Nous stoppons MariaDB

```
sudo systemctl stop mariadb
```

Nous créons le fichier /etc/mysql/mariadb.conf.d/50-galera.cnf

```
sudo vi /etc/mysql/mariadb.conf.d/50-galera.cnf
```

Et nous y intégrons les informations suivantes

```
[galera]
binlog_format=ROW
default_storage_engine=InnoDB
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0

# Galera Provider Configuration
wsrep_on=ON
wsrep_provider='/usr/lib/galera/libgalera_smm.so'

# Galera Cluster Configuration
wsrep_cluster_name='cluster01'
wsrep_cluster_address='gcomm://10.9.214.73,10.9.214.74'

# Galera synchronisation configuration
wsrep_sst_method=rsync

# Galera node configuration
wsrep_node_address='10.9.214.74'
wsrep_node_name='dbcl01-db02'
```

Nous démarrons MariaDB

```
sudo systemctl start mariadb
```

Nous nous connectons à la base pour vérifier

```
mysql -u root -p
```

Et saisissons la requête

```
SHOW GLOBAL STATUS WHERE Variable_name IN ('wsrep_ready',
'wsrep_cluster_size', 'wsrep_cluster_status', 'wsrep_connected');
```

Vous devriez avoir ce résultat

```
+----------------------+---------+
| Variable_name        | Value   |
+----------------------+---------+
| wsrep_cluster_size   | 2       |
| wsrep_cluster_status | Primary |
| wsrep_connected      | ON      |
```

```
| wsrep_ready          | ON      |
+----------------------+---------+
4 rows in set (0.00 sec)
```

Nous réitérons la même opération sur le noeud 3.

Nous créons le fichier /etc/mysql/mariadb.conf.d/50-galera.cnf

```
sudo vi /etc/mysql/mariadb.conf.d/50-galera.cnf
```

Et nous y intégrons les informations suivantes

```
[galera]
binlog_format=ROW
default_storage_engine=InnoDB
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0

# Galera Provider Configuration
wsrep_on=ON
wsrep_provider='/usr/lib/galera/libgalera_smm.so'

# Galera Cluster Configuration
wsrep_cluster_name='cluster01'
wsrep_cluster_address='gcomm://10.9.214.73,10.9.214.74,10.9.214.75'

# Galera synchronisation configuration
wsrep_sst_method=rsync

# Galera node configuration
wsrep_node_address='10.9.214.75'
wsrep_node_name='dbcl01-db03'
```

Nous démarrons MariaDB

```
sudo systemctl start mariadb
```

Nous nous connectons à la base pour vérifier

```
mysql -u root -p
```

Et saisissons la requête

```
SHOW GLOBAL STATUS WHERE Variable_name IN ('wsrep_ready',
'wsrep_cluster_size', 'wsrep_cluster_status', 'wsrep_connected');
```

Vous devriez avoir ce résultat

```
+----------------------+---------+
| Variable_name        | Value   |
```

```
+----------------------+---------+
| wsrep_cluster_size   | 3       |
| wsrep_cluster_status | Primary |
| wsrep_connected      | ON      |
| wsrep_ready          | ON      |
+----------------------+---------+
4 rows in set (0.00 sec)
```

Il ne nous reste plus qu'à éditer les fichiers 50-galera.cnf sur les noeuds 1 et 2 à ajouter tous les noeuds du cluster à la ligne “wsrep_cluster_address” de la façon suivante

```
wsrep_cluster_address='gcomm://10.9.214.73,10.9.214.74,10.9.214.75'
```

Nous redémarrons MariaDB sur les noeuds 1 et 2

```
sudo systemctl restart mariadb
```

Mise en place du load balancer

Nous installons les paquets haproxy et keepalived :

```
sudo aptitude install haproxy keepalived
```

Nous créons les comptes haproxy sur l'un des noeuds MariaDB, mettons à jour les privilèges, et sortons

```
mysql -u root -p
CREATE USER 'haproxy'@'10.9.214.71';
CREATE USER 'haproxy'@'10.9.214.72';
FLUSH PRIVILEGES;
quit
```

Pour contrôler que nous avons bien nos deux utilisateurs

```
mysql -u root -p -e "select * from mysql.user" | grep haproxy | cut -d$'\t'
-f1,2
```

Vous devriez avoir ceci en retour

```
Enter password:
10.9.214.71 haproxy
10.9.214.72 haproxy
```

Editez le fichier /etc/haproxy/haproxy.cfg et ajoutez ceci à la fin du fichier

```
listen mariadb-galera-writes
    bind 0.0.0.0:3307
    mode tcp
```

```
    option mysql-check user haproxy
    server db1 10.9.214.73:3306 check
    server db2 10.9.214.74:3306 check backup
    server db3 10.9.214.75:3306 check backup

listen mariadb-galera-reads
    bind 0.0.0.0:3306
    mode tcp
    balance leastconn
    option mysql-check user haproxy
    server db1 10.9.214.73:3306 check
    server db2 10.9.214.74:3306 check
    server db3 10.9.214.75:3306 check

# HAProxy web ui
listen stats
    bind 0.0.0.0:80
    mode http
    stats enable
    stats uri /haproxy
    stats realm HAProxy\ Statistics
    stats auth admin:password
    stats admin if TRUE
```

Keepalived

Sur le noeud 1, nous éditons le fichier /etc/keepalived/keepalived.conf et y ajoutons

```
# config file for keepalived on lac-dbcl01-lb01

global_defs {
        notification_email {
                root@neotion.com
        }
        notification_email_from lac-dbcl01-lb01-noreply@neotion.com
        smtp_server smtp.neotion.com
        smtp_connect_timeout 30
        router_id lac-dbcl01-lb01
}

vrrp_script chk_haproxy {
        script "/usr/bin/systemctl is-active --quiet haproxy"
        interval 2
        weight 2
}

vrrp_instance Cluster01 {
        state MASTER
        interface eth0
```

```
        smtp_alert
        virtual_router_id 10
        priority 101
        advert_int 1
        authentication {
                auth_type PASS
                auth_pass password # use 8 chars & something better
        }
        virtual_ipaddress {
                10.9.214.70
        }
        track_script {
                chk_haproxy
        }
}
```

Sur le noeud 2, nous éditons le fichier /etc/keepalived/keepalived.conf et y ajoutons

```
# config file for keepalived on lac-dbcl01-lb02

global_defs {
        notification_email {
                root@example.com
        }
        notification_email_from lac-dbcl01-lb02-noreply@neotion.com
        smtp_server smtp.example.com
        smtp_connect_timeout 30
        router_id lac-dbcl01-lb02
}

vrrp_script chk_haproxy {
        script "/usr/bin/systemctl is-active --quiet haproxy"
        interval 2
        weight 2
}

vrrp_instance Cluster01 {
        state MASTER
        interface eth0
        smtp_alert
        virtual_router_id 10
        priority 100
        advert_int 1
        authentication {
                auth_type PASS
                auth_pass password # use 8 chars & something better
        }
        virtual_ipaddress {
                10.9.214.70
        }
        track_script {
```

```
                chk_haproxy
        }
}
```

Redémarrer haproxy et keeaplived sur chaque noeud

```
sudo service haproxy restart
sudo service keepalived restart
```

Liens

- how to configure a galera cluster with mariadb 10.1 on ubuntu 16.04 servers
- what is galera and how to configure glb with galera cluster
- monter un cluster Galera MariaDB
- howto setup high available haproxy with keepalived