Mysql - Cluster Management

La procédure a été testée sur des machines ubuntu server 22.04 fresh install. L'infrastructure est composée des machines suivantes :

- Ubuntu Server (db-01) => 192.168.56.61
- Ubuntu Server (db-02) => 192.168.56.62

Installation du cluster

Les packages suivants doivent être installés sur l'ensemble des noeuds du cluster.

```
apt update
apt install -y mariadb-server mariadb-client galera-4
mysql_secure_installation
```

Options du secure_installation :

- Empty
- n
- Y
- root
- root
- Y
- Y
- Y
- Y

Le fichier de configuration suivant doit être mis en oeuvre : `/etc/mysql/mariadb.conf.d/60-galera.conf`

```
[galera]
wsrep_on = ON
wsrep_provider = /usr/lib/galera/libgalera_smm.so
wsrep_cluster_name = "Toto Cluster"
wsrep_cluster_address = gcomm://192.168.56.61,192.168.56.62
binlog_format = row
default_storage_engine = InnoDB

bind-address = 0.0.0.0

wsrep_slave_threads = 1
wsrep_sst_method = rsync
wsrep_node_address = "192.168.56.61"
wsrep_node_name = "db-01"
```

> Note : La ligne wsrep_provider est extrêmement importante pour le bon fonctionnement du cluster.

La configuration doit être adaptée sur le deuxième noeud au niveau de la node_address pour contenir l'adresse de la machine locale.

> Note : Le champ wsrep_cluster_address doit contenir l'intégralité des machines du cluster incluant la machine courante.

Une fois la configuration préparée, il va falloir éteindre le service mariadb et initier le cluster.

```
systemctl stop mariadb
galera_new_cluster
```

Le script galera_new_cluster est un wrapper de systemd qui va se charger de relancer le service mariadb.

Pour valider que le cluster est bien lancé :

- Verification du nombre de noeuds du cluster

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

```
root@db-01:~# mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
Enter password:
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 2     |
+--------------------+-------+
root@db-01:~#
```

- Vérification de la bonne réplication des données

La création de la base de données devrait s'être opérée sur la vision des deux machines.

```
# Sur la machine 1
mysql -u root -p -e "CREATE DATABASE toto"
mysql -u root -p -e "SHOW DATABASES"

# Sur la machine 2
mysql -u root -p -e "SHOW DATABASES"
```

```
root@db-02:~# mysql -u root -p -e "SHOW DATABASES"
Enter password:
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| toto               |
+--------------------+
root@db-02:~#
```

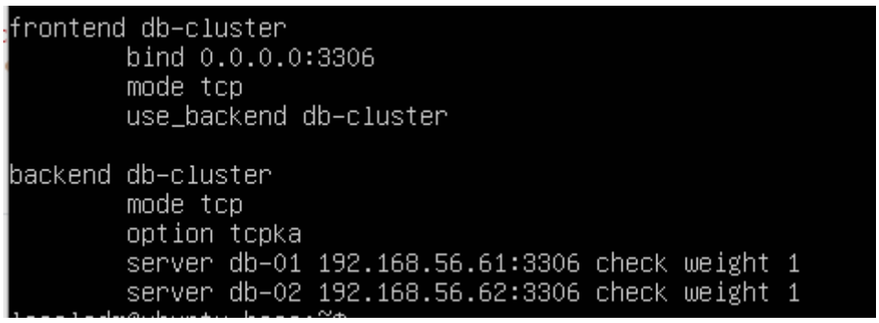
Mise en place du load balancer

Pour que le cluster ait du sens, il est nécessaire d'ajouter un reverse proxy (load balancer) devant.

Le load balancer qui sera utilisé est haproxy.

```
apt install -y haproxy
```

La configuration du frontend et du backend nécessaire est la suivante :



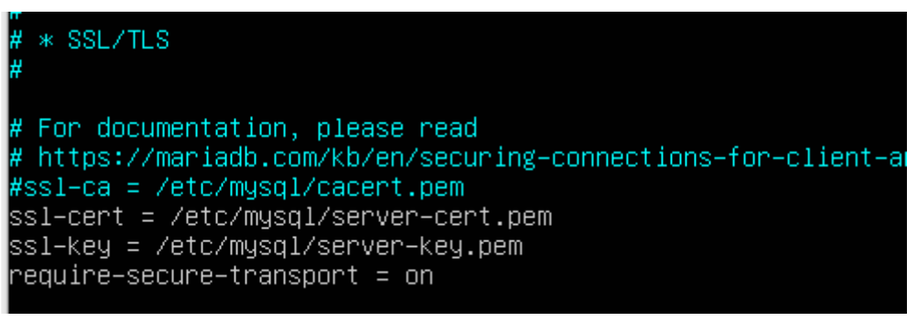
```
frontend db-cluster
        bind 0.0.0.0:3306
        mode tcp
        use_backend db-cluster

backend db-cluster
        mode tcp
        option tcpka
        server db-01 192.168.56.61:3306 check weight 1
        server db-02 192.168.56.62:3306 check weight 1
```

> Note : L'option tcpka permet d'activer le TCP KeepAlive ce qui permet de conserver la session ouverte au niveau réseau (intéressant pour les échanges avec une base de données)

Sécurisation par le TLS

L'activation du TLS se fait en mode STARTTLS dans le fichier de configuration `/etc/mysql/mariadb.conf.d/50-server.conf` et en décommentant les lignes suivantes :



```
# * SSL/TLS
#

# For documentation, please read
# https://mariadb.com/kb/en/securing-connections-for-client-a
#ssl-ca = /etc/mysql/cacert.pem
ssl-cert = /etc/mysql/server-cert.pem
ssl-key = /etc/mysql/server-key.pem
require-secure-transport = on
```

```
ssl-cert = /etc/mysql/server-cert.pem
ssl-key = /etc/mysql/server-key.pem
require-secure-transport = on
```

Il sera nécessaire de créer la paire de clé en question :

```
openssl req -new -newkey rsa:2048 -days 365 -nodes -x509 -keyout /etc/mysql/server-
key.pem -out /etc/mysql/server-cert.pem
```

Et de donner l'appartenance de ces fichiers à l'utilisateur mysql :

```
chown mysql /etc/mysql/server-*.pem
```

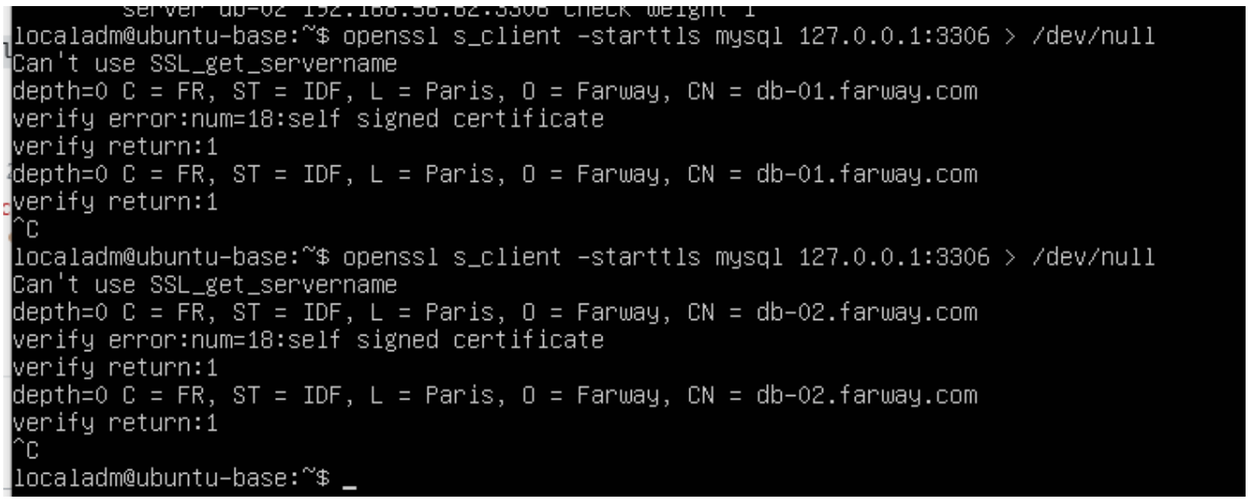
Il ne restera plus qu'à redémarrer le service :

```
systemctl restart mariadb
```

> Note : Le cluster peut fonctionner avec des noeuds en TLS et des noeuds sans. Il faut faire attention à bien reproduire le même schéma de configuration sur l'ensemble des noeuds.

Pour vérifier, on utilisera le client openssl de la manière suivante :

```
openssl s_client -starttls mysql 127.0.0.1:3306 > /dev/null
```



> Note : Sur la capture ci-dessus, on travaille au niveau du load balancer, nous avons donc bien à la fois la partie chiffrement et la partie load balancing d'actifs.