LISV Laboratoire d'ingénierie des systèmes de Versailles
UVSQ université PARIS-SACLAY

Gestion d'un système de bases de données – TP : Installation et Configuration de MariaDB

Halim Djerroud

révision 1.0

Objectifs du TP

À l'issue de ce TP, vous serez capable de :
— Installer et configurer un serveur MariaDB sur Linux
— Sécuriser l'installation initiale
— Créer et gérer des utilisateurs avec différents niveaux de privilèges
— Configurer l'accès réseau au SGBD
— Mettre en place des règles de pare-feu
— Utiliser phpMyAdmin de manière sécurisée
— Appliquer le principe du moindre privilège
Durée : 2h30
Prérequis : Machine virtuelle Linux (Debian/Ubuntu) avec accès root

1 Partie 1 : Installation et Configuration Initiale (30 min)

1.1 Exercice 1.1 : Préparation du système

1. Mettez à jour votre système :

```
sudo apt update
sudo apt upgrade -y
```

2. Vérifiez l'espace disque disponible :

```
df -h
```

Question 1.1 : Combien d'espace est disponible sur la partition racine ?

3. Configurez le hostname de votre serveur :

```
sudo hostnamectl set-hostname sgbd-tp
```

4. Vérifiez la configuration réseau :

```
ip addr show
hostname
```

Question 1.2 : Quelle est l'adresse IP de votre machine ?

1.2 Exercice 1.2 : Installation de MariaDB

1. Installez MariaDB :

```
sudo apt install mariadb-server mariadb-client -y
```

2. Vérifiez l'installation :

```
mysql --version
```

Question 1.3 : Quelle version de MariaDB avez-vous installée ?

3. Vérifiez que le service est démarré :

```
sudo systemctl status mariadb
```

4. Si le service n'est pas actif, démarrez-le :

   ```
   sudo systemctl start mariadb
   ```

5. Activez le démarrage automatique :

   ```
   sudo systemctl enable mariadb
   ```

6. Vérifiez que MariaDB écoute bien :

   ```
   sudo ss -tlnp | grep mysql
   ```

   Question 1.4 : Sur quelle adresse IP et quel port MariaDB écoute-t-il par défaut ?

1.3 Exercice 1.3 : Sécurisation initiale

1. Lancez le script de sécurisation :

   ```
   sudo mysql_secure_installation
   ```

2. Répondez aux questions comme suit :
   - Enter current password for root : `[Entrée]`
   - Switch to unix_socket authentication : `n`
   - Change the root password : `Y`
   - Nouveau mot de passe : choisissez un mot de passe fort (ex : `Root2024!Secure`)
   - Remove anonymous users : `Y`
   - Disallow root login remotely : `Y`
   - Remove test database : `Y`
   - Reload privilege tables : `Y`

3. Testez la connexion avec le nouveau mot de passe :

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

4. Une fois connecté, listez les bases de données :

   ```
   SHOW DATABASES;
   ```

5. Quittez le client MySQL :

   ```
   EXIT;
   ```

   Question 1.5 : Pourquoi est-il important de supprimer les utilisateurs anonymes et la base de données test ?

2 Partie 2 : Gestion des Utilisateurs et des Droits (45 min)

2.1 Exercice 2.1 : Création d'utilisateurs

Connectez-vous à MariaDB en tant que root :

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

1. Créez un utilisateur pour une application web (accès local uniquement) :

   ```
   CREATE USER 'appweb'@'localhost'
   IDENTIFIED BY 'WebApp2024!';
   ```

2. Créez un utilisateur administrateur pouvant se connecter depuis le réseau local :

   ```
   CREATE USER 'admin_db'@'192.168.%.%'
   IDENTIFIED BY 'AdminDB2024!';
   ```

3. Créez un utilisateur en lecture seule :

   ```
   CREATE USER 'readonly'@'localhost'
   IDENTIFIED BY 'ReadOnly2024!';
   ```

4. Listez tous les utilisateurs créés :

   ```
   SELECT User, Host FROM mysql.user;
   ```

   Question 2.1 : Pourquoi l'utilisateur appweb est-il limité à 'localhost' ?

2.2 Exercice 2.2 : Création de bases de données

1. Créez une base de données pour une application e-commerce :

```
CREATE DATABASE ecommerce
CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;
```

2. Créez une base de données pour un blog :

```
CREATE DATABASE blog
CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;
```

3. Créez une base de données pour les statistiques :

```
CREATE DATABASE stats
CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;
```

4. Listez toutes les bases de données :

```
SHOW DATABASES;
```

Question 2.2 : Pourquoi utilise-t-on utf8mb4 plutôt que utf8 ?

2.3 Exercice 2.3 : Attribution de privilèges

1. Donnez tous les droits à appweb sur la base ecommerce :

```
GRANT SELECT, INSERT, UPDATE, DELETE
ON ecommerce.*
TO 'appweb'@'localhost';
```

2. Donnez tous les privilèges à admin_db sur toutes les bases :

```
GRANT ALL PRIVILEGES
ON *.*
TO 'admin_db'@'192.168.%.%'
WITH GRANT OPTION;
```

3. Donnez uniquement le droit de lecture à readonly sur la base stats :

```
GRANT SELECT
ON stats.*
TO 'readonly'@'localhost';
```

4. Rechargez les privilèges :

```
FLUSH PRIVILEGES;
```

5. Vérifiez les privilèges de chaque utilisateur :

```
SHOW GRANTS FOR 'appweb'@'localhost';
SHOW GRANTS FOR 'admin_db'@'192.168.%.%';
SHOW GRANTS FOR 'readonly'@'localhost';
```

Question 2.3 : Pourquoi n'a-t-on pas donné les privilèges DROP, CREATE, ALTER à appweb ?

2.4 Exercice 2.4 : Tests de connexion

1. Ouvrez un nouveau terminal et testez la connexion avec appweb :

```
mysql -u appweb -p
```

2. Une fois connecté, essayez :

```
USE ecommerce;
SHOW TABLES;
-- Essayez de créer une table (devrait échouer)
CREATE TABLE test (id INT);
-- Essayez d'accéder à une autre base (devrait échouer)
USE blog;
EXIT;
```

3. Testez la connexion avec readonly :

```
mysql -u readonly -p
```

```
USE stats;
-- Essayez d'insérer des données (devrait échouer)
CREATE TABLE test (id INT);
EXIT;
```

Question 2.4 : Que se passe-t-il quand readonly essaie de créer une table ? Pourquoi ?

3 Partie 3 : Configuration Réseau et Sécurité (45 min)

3.1 Exercice 3.1 : Configuration de l'écoute réseau

1. Affichez la configuration actuelle :

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

2. Cherchez la ligne `bind-address` et notez sa valeur actuelle.

Question 3.1 : Quelle est la valeur actuelle de bind-address ? Que signifie-t-elle ?

3. Modifiez bind-address pour écouter sur toutes les interfaces :

```
bind-address = 0.0.0.0
```

ATTENTION : En production, on préférerait une IP spécifique ou le maintien en localhost avec un tunnel SSH.

4. Sauvegardez et quittez (Ctrl+X, puis Y, puis Entrée).
5. Redémarrez MariaDB :

```
sudo systemctl restart mariadb
```

6. Vérifiez que MariaDB écoute maintenant sur toutes les interfaces :

```
sudo ss -tlnp | grep 3306
```

Question 3.2 : Quelle est la différence entre 127.0.0.1 :3306 et 0.0.0.0 :3306 ?

3.2 Exercice 3.2 : Configuration du pare-feu

1. Vérifiez l'état du pare-feu :

```
sudo ufw status
```

2. Si le pare-feu n'est pas actif, activez-le :

```
sudo ufw enable
```

3. Autorisez SSH (important pour ne pas vous bloquer !) :

```
sudo ufw allow 22/tcp
```

4. Autorisez MariaDB uniquement depuis le réseau local :

```
sudo ufw allow from 192.168.0.0/16 to any port 3306
```

Note : Adaptez le réseau (192.168.0.0/16) à votre configuration.

5. Bloquez tout le reste par défaut :

```
sudo ufw default deny incoming
sudo ufw default allow outgoing
```

6. Vérifiez les règles :

```
sudo ufw status verbose
```

Question 3.3 : Pourquoi limite-t-on l'accès au port 3306 à certaines adresses IP ?

3.3 Exercice 3.3 : Installation et sécurisation de phpMyAdmin

1. Installez Apache et PHP :

```
sudo apt install apache2 php libapache2-mod-php php-mysql -y
```

2. Installez phpMyAdmin :

```
sudo apt install phpmyadmin -y
```

3. Lors de l'installation :
   — Sélectionnez apache2
   — Configurez la base de données : Yes
   — Entrez un mot de passe pour phpmyadmin
4. Créez un lien symbolique :

```
sudo ln -s /usr/share/phpmyadmin /var/www/html/phpmyadmin
```

5. Redémarrez Apache :

```
sudo systemctl restart apache2
```

6. Accédez à phpMyAdmin via votre navigateur :

```
http://VOTRE_IP/phpmyadmin
```

7. Connectez-vous avec l'utilisateur root.
   Question 3.4 : Quels sont les risques de laisser phpMyAdmin accessible avec l'URL par défaut /phpmyadmin ?

3.4 Exercice 3.4 : Sécurisation de phpMyAdmin

1. Changez l'URL d'accès :

```
sudo mv /usr/share/phpmyadmin /usr/share/pma_secure_2024
sudo rm /var/www/html/phpmyadmin
sudo ln -s /usr/share/pma_secure_2024 /var/www/html/pma_secure_2024
```

2. Configurez phpMyAdmin pour refuser l'accès root :

```
sudo nano /etc/phpmyadmin/config.inc.php
```

3. Ajoutez à la fin du fichier :

```
$cfg['Servers'][$i]['AllowRoot'] = false;
```

4. Créez un utilisateur dédié pour phpMyAdmin :

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

```
CREATE USER 'phpmyadmin'@'localhost'
IDENTIFIED BY 'PMA2024!Secure';

GRANT SELECT, INSERT, UPDATE, DELETE
ON ecommerce.*
TO 'phpmyadmin'@'localhost';

GRANT SELECT, INSERT, UPDATE, DELETE
ON blog.*
TO 'phpmyadmin'@'localhost';

FLUSH PRIVILEGES;
EXIT;
```

5. Redémarrez Apache :

   ```
   sudo systemctl restart apache2
   ```

6. Testez l'accès avec la nouvelle URL :

   ```
   http://VOTRE_IP/pma_secure_2024
   ```

7. Essayez de vous connecter avec root (devrait échouer).
8. Connectez-vous avec phpmyadmin.

   Question 3.5 : Listez 3 autres mesures de sécurité pour phpMyAdmin.

4 Partie 4 : Manipulation de Données et Audit (30 min)

4.1 Exercice 4.1 : Création de tables et insertion de données

Connectez-vous avec l'utilisateur appweb :

```
mysql -u appweb -p ecommerce
```

1. Créez une table clients :

   ```
   CREATE TABLE clients (
       id INT PRIMARY KEY AUTO_INCREMENT,
       nom VARCHAR(100) NOT NULL,
       email VARCHAR(100) UNIQUE NOT NULL,
       ville VARCHAR(50),
       date_inscription DATETIME DEFAULT CURRENT_TIMESTAMP
   );
   ```

   Question 4.1 : Pourquoi cette commande échoue-t-elle ? Comment résoudre le problème ?

2. Reconnectez-vous en tant que root et accordez les privilèges nécessaires :

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

   ```
   GRANT CREATE, ALTER, INDEX
   ON ecommerce.*
   TO 'appweb'@'localhost';
   FLUSH PRIVILEGES;
   EXIT;
   ```

3. Reconnectez-vous avec appweb et créez la table :

   ```
   mysql -u appweb -p ecommerce
   ```

   ```
   CREATE TABLE clients (
       id INT PRIMARY KEY AUTO_INCREMENT,
       nom VARCHAR(100) NOT NULL,
       email VARCHAR(100) UNIQUE NOT NULL,
       ville VARCHAR(50),
       date_inscription DATETIME DEFAULT CURRENT_TIMESTAMP
   );

   CREATE TABLE produits (
       id INT PRIMARY KEY AUTO_INCREMENT,
       nom VARCHAR(100) NOT NULL,
       prix DECIMAL(10, 2) NOT NULL,
       stock INT DEFAULT 0
   );
   ```

4. Insérez des données :

   ```
   INSERT INTO clients (nom, email, ville) VALUES
   ('Alice Martin', 'alice@example.com', 'Paris'),
   ('Bob Durand', 'bob@example.com', 'Lyon'),
   ('Charlie Petit', 'charlie@example.com', 'Marseille');
   ```

```
INSERT INTO produits (nom, prix, stock) VALUES
('Ordinateur portable', 899.99, 15),
('Souris sans fil', 29.99, 50),
('Clavier mécanique', 149.99, 30);
```

5. Vérifiez les données :

```
SELECT * FROM clients;
SELECT * FROM produits;
```

4.2 Exercice 4.2 : Tests de sécurité

1. Avec l'utilisateur appweb, essayez de supprimer la table :

```
DROP TABLE clients;
```

Question 4.2 : Que se passe-t-il ? Pourquoi ?

2. Essayez de créer une nouvelle base de données :

```
CREATE DATABASE test_hack;
```

Question 4.3 : Cette commande fonctionne-t-elle ? Expliquez.

3. Connectez-vous avec readonly :

```
mysql -u readonly -p stats
```

4. Créez une table de test dans stats :

```
CREATE TABLE logs (
    id INT PRIMARY KEY AUTO_INCREMENT,
    message TEXT,
    date_log DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

Note : Utilisez root pour cette création.

5. Reconnectez-vous avec readonly et testez :

```
SELECT * FROM logs;
INSERT INTO logs (message) VALUES ('Test');
```

Question 4.4 : Quelle commande fonctionne et laquelle échoue ? Pourquoi ?

4.3 Exercice 4.3 : Surveillance et audit

1. Connectez-vous en tant que root :

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

2. Affichez les connexions actives :

```
SHOW PROCESSLIST;
```

3. Vérifiez les dernières connexions (logs système) :

```
sudo tail -n 50 /var/log/mysql/error.log
```

4. Activez le log général (pour audit) :

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

5. Ajoutez dans la section [mysqld] :

```
general_log_file = /var/log/mysql/mysql.log
general_log = 1
```

6. Redémarrez MariaDB :

```
sudo systemctl restart mariadb
```

7. Effectuez quelques requêtes, puis consultez le log :

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

Question 4.5 : Pourquoi ne faut-il pas laisser le general_log activé en production ?

5 Synthèse et Évaluation

5.1 Questions de réflexion

1. **Principe du moindre privilège :** Expliquez en 5 lignes pourquoi ce principe est essentiel dans la gestion d'un SGBD.
2. **Architecture réseau :** Vous devez déployer un SGBD pour une application web. Décrivez l'architecture réseau sécurisée que vous mettriez en place (VLAN, pare-feu, accès).
3. **Comparaison :** Quels sont les avantages et inconvénients de laisser MariaDB écouter sur 0.0.0.0 vs 127.0.0.1 ?
4. **Sécurité :** Listez 5 bonnes pratiques de sécurité à appliquer lors de l'installation d'un SGBD en production.
5. **Cas d'usage :** Une entreprise a 3 types d'utilisateurs :
   — Développeurs : besoin de créer/modifier tables en dev
   — Application production : CRUD uniquement
   — Analystes : lecture seule sur données production
   Proposez une stratégie de gestion des utilisateurs et privilèges.

5.2 Checklist de validation du TP

Vérifiez que vous avez réalisé toutes les étapes :
☐ MariaDB installé et sécurisé
☐ Service activé au démarrage
☐ Utilisateurs créés avec différents niveaux de privilèges
☐ Bases de données créées
☐ Privilèges attribués correctement
☐ Configuration réseau modifiée
☐ Pare-feu configuré avec règles strictes
☐ phpMyAdmin installé et sécurisé
☐ Tests de connexion et privilèges effectués
☐ Logs consultés et compris

5.3 Pour aller plus loin

Si vous avez terminé en avance, explorez ces sujets :

1. **SSL/TLS :** Configurez une connexion chiffrée avec des certificats auto-signés (voir chapitre 4 du cours).
2. **Fail2ban :** Installez et configurez fail2ban pour protéger MariaDB contre les attaques par force brute.
3. **Tunnel SSH :** Pratiquez la connexion à distance via un tunnel SSH au lieu d'ouvrir le port 3306.
4. **Réplication :** Documentez-vous sur la réplication Master-Slave dans MariaDB.
5. **Backup :** Réalisez une sauvegarde complète avec mysqldump et testez la restauration.

Conclusion

Ce TP vous a permis de mettre en pratique les concepts vus en cours :
— Installation et configuration d'un SGBD
— Application du principe du moindre privilège
— Sécurisation multicouche (authentification, réseau, pare-feu)
— Gestion des utilisateurs et des privilèges
— Bonnes pratiques d'administration
Ces compétences sont essentielles pour tout professionnel RT amené à gérer des services de bases de données.

Ressources complémentaires :
— Documentation officielle MariaDB : https://mariadb.com/kb/
— Guide de sécurité MySQL : https://dev.mysql.com/doc/refman/8.0/en/security.html
— CIS Benchmark pour MySQL : https://www.cisecurity.org/