Comment active SSL Sur MySQL

Lorsque les utilisateurs veulent avoir une connexion sécurisés à leur serveur MySQL, ils comptent souvent sur les tunnels VPN ou SSH. Pourtant, une autre option pour la sécurisation des connexions MySQL est d'activer SSL wrapper sur un serveur MySQL.
Chacune de ces approches a ses propres avantages et inconvénients.

Par exemple, dans des environnements très dynamiques où beaucoup de connexions MySQL de courte durée, le VPN ou les tunnels SSH peuvent être un meilleur choix que le protocole SSL qui implique un coût de calcul sur la liaison SSL par connexion.

D'autre part, pour les applications avec relativement peu de connexions MySQL de longue durée, le chiffrement basé sur SSL peut être raisonnable.
MySQL possède déjà un support intégré SSL, vous ne devez pas mettre en œuvre une couche de sécurité distinct comme sur un VPN ou un tunnel SSH, qui a son propre temps de maintenance.

La mise en œuvre de SSL dans un serveur MySQL, chiffre toutes les données entre le serveur et le client, empêchant ainsi les écoutes ou les données reniflé dans les réseaux étendus.
En outre, SSL fournit également une possibilité d'identifier, de vérifier le serveur au moyen du certificat SSL, ce qui peut protéger les utilisateurs contre les attaques possibles de phishing.

Dans cet article, nous allons voir ensemble, comment activer le protocole SSL sur un serveur MySQL.
A noter que la même procédure est également applicable à MariaDB.

Création du certificat SSL et la clé privée

Nous devons créer un certificat SSL et la clé privée d'un serveur MySQL, qui sera utilisé lors de la connexion au serveur via SSL.
Tout d'abord, créer un répertoire de travail temporaire où nous allons garder les clés et certificats fichiers.

```
mkdir ~/cert
cd ~/cert
```

Assurez-vous que OpenSSL est installé sur votre système et un serveur MySQL est en cours d'exécution.
Pour vérifier si OpenSSL est installé, utilisez la commande suivante.

```
openssl version
```

```
OpenSSL 1.0.1k 8 Jan 2015
```

Maintenant, nous allons créer votre certificat pour votre autorité de certification “CA” pour que nous puissions après auto-signé nos certificats.
Les commandes suivantes vont créer ca-key.pem et ca-cert.pem :

```
openssl genrsa 4096 > ca-key.pem
```

```
openssl req -new -x509 -nodes -days 365000 -key ca-key.pem -out ca-cert.pem
```

La seconde commande va vous posez quelques questions.
ce que vous mettez dans ces champs n'a pas d'importance, il suffit de remplir ces champs.

La prochaine étape est de créer une requête de certificat et une paire de clés.

```
openssl req -newkey rsa:4096 -days 365000 -nodes -keyout server-key.pem -out
server-req.pem
```

Cette commande va poser plusieurs questions à nouveau et vous pouvez mettre les mêmes réponses que vous avez fournies à l'étape précédente.

Ensuite, exporter la clé privée du serveur de type RSA.

```
openssl rsa -in server-key.pem -out server-key.pem
```

Enfin, générer un certificat de serveur à l'aide du certificat de CA.

```
openssl x509 -req -in server-req.pem -days 365000 -CA ca-cert.pem -CAkey ca-
key.pem -set_serial 01 -out server-cert.pem
```

Configuration de SSL sur le serveur MySQL

Après les procédures ci-dessus, nous devrions avoir un certificat de CA, la clé privée d'un serveur et son certificat.

nom du fichier	**details**
ca-cert.pem	certificat de l'autorité de certification
server-key.pem	fichier contenant la clé RSA privé du serveur
server-cert.pem	certificat x509 pour une durée de validité de 10 ans et auto-signé

L'étape suivante consiste à configurer notre serveur MySQL d'utiliser la touche et les certificats.

Avant de configurer le serveur MySQL, vérifiez si les options SSL sont activés ou désactivés.
Pour cela, connectez-vous au serveur MySQL, et tapez la requête ci-dessous

```
mysql> SHOW GLOBAL VARIABLES LIKE 'have_%ssl';
```

Le résultat de cette requête va ressembler à :

```
+---------------+----------+
| Variable_name | Value    |
+---------------+----------+
| have_openssl  | DISABLED |
| have_ssl      | DISABLED |
+---------------+----------+
2 rows in set (0.08 sec)
```

Notez que la valeur par défaut de 'have_openssl' et variables »de HAVE_SSL» est «DISABLED» comme indiqué ci-dessus.
Pour activer SSL sur le serveur MySQL suivez les étapes ci-dessous.

Copier ou déplacer ca-cert.pem, serveur cert.pem, et le serveur-key.pem sous le répertoire / etc.

```
mkdir /etc/mysql-ssl
cp ca-cert.pem server-cert.pem server-key.pem /etc/mysql-ssl
```

Ouvrir `my.cnf` du serveur en utilisant un éditeur de texte.
Ajouter ou dé-commenter les lignes ci-dessous dans la en section `[mysqld]`.
Ceux-ci devraient pointer vers la clé et certificats que vous avez placé dans `/etc/mysql-ssl`.

```
[mysqld]
require_secure_transport = on
ssl-ca=/etc/mysql-ssl/ca-cert.pem
ssl-cert=/etc/mysql-ssl/server-cert.pem
ssl-key=/etc/mysql-ssl/server-key.pem
ssl-cipher = ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-
ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-
SHA256:DHE-RSA-AES256-GCM-SHA384
tls-version = TLSv1.2,TLSv1.3
```

Dans `my.cnf`, trouver aussi “bind-address = 127.0.0.1 et changer par:

```
bind-address = *
```

ou

```
bind-address = <IP DE VOTRE SERVEUR>
```

De cette façon, vous pouvez vous connecter au serveur MySQL depuis un autre hôte.

Redémarrez le service MySQL.

```
service mysql restart
```

ou

```
systemctl restart mysql
```

ou

```
/etc/init.d/mysql restart
```

Vous pouvez vérifier si la configuration SSL fonctionne ou non en examinant le fichier journal des erreurs MySQL (par exemple, `/var/log/mysql/mysql.log`).
Si aucun avertissement ou une erreur est signalée dans le journal d'erreur, cela signifie que la configuration SSL fonctionne bien.

Une autre façon de vérifier la configuration de SSL est de ré-exécuter la requête `have_% ssl` intérieur du serveur MySQL.

```
mysql> SHOW GLOBAL VARIABLES LIKE 'have_%ssl';
```

Le résultat de cette requête va ressembler à :

```
+---------------+----------+
| Variable_name | Value    |
+---------------+----------+
| have_openssl  | YES      |
| have_ssl      | YES      |
+---------------+----------+
2 rows in set (0.08 sec)
```

Création d'un utilisateur avec SSL Privilège

Après la configuration SSL côté serveur est terminée, la prochaine étape est de créer un utilisateur qui a un privilège pour accéder au serveur MySQL via SSL.
Pour cela, connectez-vous au serveur MySQL :

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'ssluser'@'%' IDENTIFIED BY
'motedepasse' REQUIRE SSL;
mysql> FLUSH PRIVILEGES;
```

Remplacez `ssluser` par le nom d'utilisateur et `motdepasse` par le mot de passe.

Si vous voulez donner une adresse IP spécifique (par exemple, 192.168.2.8) à partir de laquelle l'utilisateur peut accéder au serveur, utilisez la requête suivante à la place.

```
 mysql> GRANT ALL PRIVILEGES ON *.* TO ‘ssluser’@’192.168.2.8’ IDENTIFIED BY
'motdepasse' REQUIRE SSL;
 mysql> FLUSH PRIVILEGES;
```

Configurez SSL sur MySQL client

Maintenant que la configuration côté serveur MySQL est faite, passons côté client.
Pour le client MySQL, nous devons créer une nouvelle clé et certificat basé sur la clé du serveur.

openssl req -newkey rsa:2048 -days 365000 -nodes -keyout client-key.pem -out client-req.pem

Similaire à la configuration côté serveur, la commande ci-dessus va poser plusieurs questions.
Il suffit de remplir les champs comme nous le faisions auparavant.

Nous avons également besoin de convertir la clé de client généré en type RSA comme suit.

```
openssl rsa -in client-key.pem -out client-key.pem
```

Enfin, nous devons créer un certificat client à l'aide de la clé et le certificat CA du serveur.

```
openssl x509 -req -in client-req.pem -days 365000 -CA ca-cert.pem -CAkey ca-key.pem -set_serial 01 -out client-cert.pem
```

Maintenant transférer le ca-cert.pem, client-cert.pem, et les fichiers clients-key.pem à tout hôte où vous voulez exécuter client MySQL.

Sur l'hôte client, utilisez la commande suivante pour se connecter au serveur MySQL avec SSL.

```
mysql --ssl-ca=ca-cert.pem --ssl-cert=client-cert.pem --ssl-key=client-key.pem -h <mysql-server-ip-address> -u ssluser -p
```

Après avoir tapé le mot de passe `ssluser`, vous verrez l'invite MySQL comme d'habitude.

Pour vérifier si vous êtes sur SSL :

```
status;
```

Si vous êtes connecté sur SSL, il va vous montrer les informations de chiffrement dans le domaine de SSL. Si vous ne souhaitez pas spécifier certificat client et de l'information clé dans la ligne de commande, vous pouvez créer fichier `~/.my.cnf` et de mettre les informations suivantes dans la section `[client]`.

```
[client]
ssl-ca=/path/to/ca-cert.pem
ssl-cert=/path/to/client-cert.pem
ssl-key=/path/to/client-key.pem
```

Avec cela, vous pouvez simplement utiliser la ligne de commande suivante pour se connecter au serveur via SSL.

```
mysql -h <mysql-server-ip-address> -u ssluser -p
```

Source: xmodulo.com