Mysql incluye soporte para conexiones seguras entre cliente y servidor. Habitualmente no se tiene activado en SGBD sin accesos remotos ya que carga de trabajo al procesador y ralentiza las operaciones. Sin embargo con conexiones externas SIEMPRE se activa alguno de estos dos mecanismos.

Vamos a ver la forma de utilizar **conexiones TLS**(Transport Layer Security) entre clientes y servidores. Utilizando certificados. Vamos a seguir estos pasos por este orden:

- Crear certificados de cliente
- Crear certificados de servidor
- Crear un usuario con acceso obligatorio con ssl(seguridad)
- Reinicializar el servidor para que trabaje con acceso seguro.
- Comprobar mediante las variables del sistema si está activo ssl.
- Comprobamos los ficheros log para ver el error, si lo hay.
- Lanzamos un cliente en modo seguro.
- Revisamos si la sesión del cliente se ha realizado en modo seguro.

En caso de que todo el proceso vaya correctamente se podría hacer una captura de paquetes utilizando Wireshark para ver que realmente la información enviada del servidor al cliente está encriptada y no es legible a simple vista.

Otro método es configurar **SSH** en Windows para **hacer un túnel** (este proceso viene explicado en los apuntes de teoría, vamos a hacer el más sencillo y si tenemos tiempo otro día probamos éste).

1. Generamos certificados de cliente para trabajar con SSH (ojo, los creamos como superusuario del sistema operativo). Todos los certificados, de cliente y de servidor se almacenarán en **/etc/mysql/ (en mi caso en /var/lib/mysql/)**:

   Primero se crea el de la autoridad certificadora:

```
shell> cd /etc/mysql

shell> mkdir newcerts && cd newcerts


# Create CA certificate
shell> openssl genrsa 2048 > ca-key.pem
shell> openssl req -new -x509 -nodes -days 3600 -key ca-key.pem -out ca-cert.pem
```

   A continuación creamos los certificados para el cliente:

```
# Create client certificate, remove passphrase, and sign it

# client-cert.pem = public key, client-key.pem = private key
shell> openssl req -newkey rsa:2048 -days 3600 -nodes -keyout client-key.pem -out
client-req.pem
shell> openssl rsa -in client-key.pem -out client-key.pem
shell> openssl x509 -req -in client-req.pem -days 3600 -CA ca-cert.pem -CAkey ca-key.pem
-set_serial 01 -out client-cert.pem
```

2. Generamos los certificados para el servidor:

```
# Create server certificate, remove passphrase, and sign it
# server-cert.pem = public key, server-key.pem = private key
shell> openssl req -newkey rsa:2048 -days 3600 -nodes -keyout server-key.pem -out
server-req.pem
shell> openssl rsa -in server-key.pem -out server-key.pem
shell> openssl x509 -req -in server-req.pem -days 3600 -CA ca-cert.pem -CAkey ca-key.pem
-set_serial 01 -out server-cert.pem
```

y comprobamos que todos los certificados están bien:

```
shell> openssl verify -CAfile ca-cert.pem server-cert.pem client-cert.pem
server-cert.pem: OK
client-cert.pem: OK
```

3. Añadimos las rutas a los ficheros de claves en el fichero de configuración del servidor **/etc/my.cnf (se modifican las rutas a las correctas en nuestro caso)**:

```
[client]
ssl-ca=/etc/mysql/ca-cert.pem
ssl-cert=/etc/mysql/client-cert.pem
ssl-key=/etc/mysql/client-key.pem

[mysqld] (Venían previamente comentados en el fichero)
ssl-ca=/etc/mysql/ca-cert.pem
ssl-cert=/etc/mysql/server-cert.pem
ssl-key=/etc/mysql/server-key.pem
```

4. Creamos un usuario con acceso obligatorio con SSL:

```
mysql> GRANT ALL ON  *.* to 'superlopez'@'localhost' IDENTIFIED BY 'rompetechos' REQUIRE
SSL;
```

5. Paramos el servicio mysqld:

```
shell> sudo service mysql stop
```

***NOTA**: Si no se para otra forma de pararlo es:*

```
shell> mysqladmin -u root -p shutdown
```

6. Lanzamos el servicio mysql con soporte ssl:

   --ssl-ca , certificado de la Autoridad certificadora (CA)

   --ssl-cert , certificado con la clave pública del servidor

   --ssl-key , clave privada del servidor

   Indicando para cada una de las opciones la ruta absoluta al certificado en caso de que no esté en /etc/mysql/

```
shell> sudo mysqld --ssl-ca=ca-cert.pem --ssl-cert=server-cert.pem -ssl-key=server-
key.pem
```

7. Comprobamos si nuestro servidor tiene soporte para OpenSSL:

```
mysql> show variables like '%ssl%';
```

8. Comprobamos los ficheros log para ver el error, si lo hay:

```
shell> cd /var/log
shell> cat mysqld.log|grep ERROR
```

9. Entramos con el usuario seguro:

```
shell> mysql --ssl-ca=ca-cert.pem -u superlopez -p
```

Si se quiere especificar en la conexión un certificado propio del cliente es necesario al crear el usuario escribir **require x509** en lugar de require ssl. En este caso el cliente deberá lanzarse con el siguiente comando:

```
shell> mysql --ssl-ca=ca-cert.pem --ssl-cert=client-cert.pem --ssl-key=client-key.pem
```

En ambos casos indicando la ruta correcta para los ficheros de certificados.

10. Comprobamos si la sesión del usuario es segura:

```
mysql> \s
```

Otra forma de comprobarlo es buscar el valor de la variable **Ssl_cipher**. Si esta variable tiene un valor se está utilizando ssl, en caso contrario no:

```
mysql> show status like 'Ssl_cipher';
```