# MySQL SSL Configure Howto - How to Make MySQL communication secured

**Author :** admin

Recently I've been asked **How to make communication to MySQL database encrypted**. The question was raised by a fellow developer who works on developing a Desktop standalone application in *Delphi Programming Language* with DevArt an (*SQL Connection Component* capable to **connect Delphi applications to *multiple databases like MySQL, Oracle, PostgreSQL, Interbase, Firebird*** etc.

Communicating in Secured form to MySQL database is not common task to do, as MySQL usually communicates to applications hosted on same server or *applications to communicate to MySQL are in secured DMZ* or administrated via *phpMyAdmin* web interface.

MySQL supports encrypted connections to itself using Secure Socket Layer (SSL) encryption. Setting up MySQL db to be communicated encrypted is a must for standalone Desktop applications which has to extract / insert data via remote SQL.
Configuring SQL to support communicated queries encrpytion is supported by default and easily configured on most standard Linux version distributions (*Debian, RHEL, Fedora*) with no need to recompile it.
**1. Generate SSL Certificates**

```
$ mkdir /etc/mysql-ssl && cd mysql-ssl

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

Create server certificate, remove passphrase, and sign it
*server-cert.pem* is public key, *server-key.pem* is private key

```
$ openssl req -newkey rsa:2048 -days 3600 \
        -nodes -keyout server-key.pem -out server-req.pem
```

```
$ openssl rsa -in server-key.pem -out server-key.pem
$ openssl x509 -req -in server-req.pem -days 3600 \
        -CA ca-cert.pem -CAkey ca-key.pem -set_serial 01 -out server-cert.pem
```

Create client certificate, remove passphrase, and sign it
client-cert.pem is public key and client-key.pem is private key

```
$ openssl req -newkey rsa:2048 -days 3600 \
        -nodes -keyout client-key.pem -out client-req.pem
$ openssl rsa -in client-key.pem -out client-key.pem
$ openssl x509 -req -in client-req.pem -days 3600 \
        -CA ca-cert.pem -CAkey ca-key.pem -set_serial 01 -out client-cert.pem
```

After generating the certificates, verify them:

```
$ openssl verify -CAfile ca-cert.pem server-cert.pem client-cert.pem
```

**2. Add SSL support variables to my.cnf**

Once SSL key pair files are generated in order to active SSL encryption support in MySQL server, add to **(/etc/my.cnf,  /etc/mysql/my.cnf, /usr/local/etc/my.cnf ... )** or wherever config is depending on distro **...**

```
# SSL
ssl-ca=/etc/mysql-ssl/ca-cert.pem
ssl-cert=/etc/mysql-ssl/server-cert.pem
ssl-key=/etc/mysql-ssl/server-key.pem
```

**3. Restart MySQL server**

```
 /etc/init.d/mysqld restart
...
```

4. Create SQL user to require SSL login

Create new user with access to database;

```
 GRANT ALL ON Sql_User_DB.* TO Sql_User@localhost;
FLUSH PRIVILEGES;
```

**To create administrator privileges user:**

```
 GRANT ALL PRIVILEGES ON *.* TO ‘ssluser’@'%’ IDENTIFIED BY ‘pass’ REQUIRE
SSL;
FLUSH PRIVILEGES;
```

**5. Test SSL Connection with MySQL CLI client or with few lines of PHP**

To use mysql cli for testing whether SSL connection works:

```
$ mysql -u ssluser -p'pass' --ssl-ca /etc/mysql-ssl/client-cert.pem --ssl-cert /etc/mysql-ssl/client-key.pem
```

**Once connected to MySQL to verify SSL connection works fine**:

```
 mysql> SHOW STATUS LIKE 'Ssl_Cipher';
 +---------------+--------------------+
| Variable_name | Value              |
 +---------------+--------------------+
| Ssl_cipher    | DHE-RSA-AES256-SHA |
+---------------+--------------------+
```

If you get this output this means MySQL SSL Connection is working as should.

Alternative way is to use test-mysqli-ssl.php script to test availability to mysql over SSL.

```
$conn=mysqli_init();
mysqli_ssl_set($conn, '/etc/mysql-ssl/client-key.pem', '/etc/mysql-ssl/client-cert.pem', NULL,
NULL, NULL);
if (!mysqli_real_connect($conn, '127.0.0.1', 'ssluser', 'pass')) { die(); }
$res = mysqli_query($conn, 'SHOW STATUS like "Ssl_cipher"');
print_r(mysqli_fetch_row($res));
mysqli_close($conn);
```

Note: Change username password according to your user / pass before using the script

That's all now you have ***mysql communicating queries data over SSL***