To use MySQL Shell to verify X Plugin is installed, issue:

```
shell> mysqlsh -u user --sqlc -P 3306 -e "SHOW plugins"
```

To use MySQL Client to verify X Plugin is installed, issue:

```
shell> mysql -u user -p -e "SHOW plugins"
```

An example result if X Plugin is installed is highlighted here:

```
+----------------------------+----------+--------------------+---------+---------+
| Name                       | Status   | Type               | Library | License |
+----------------------------+----------+--------------------+---------+---------+

...

| mysqlx                     | ACTIVE   | DAEMON             | NULL    | GPL     |

...

+----------------------------+----------+--------------------+---------+---------+
```

20.5.2 Disabling X Plugin

The X Plugin can be disabled at startup by either setting `mysqlx=0` in your MySQL configuration file, or by passing in either `--mysqlx=0` or `--skip-mysqlx` when starting the MySQL server.

Alternatively, use the `-DWITH_MYSQLX=OFF` CMake option to compile MySQL Server without X Plugin.

20.5.3 Using Encrypted Connections with X Plugin

This section explains how to configure X Plugin to use encrypted connections. For more background information, see Section 6.3, “Using Encrypted Connections”.

To enable configuring support for encrypted connections, X Plugin has `mysqlx_ssl_xxx` system variables, which can have different values from the `ssl_xxx` system variables used with MySQL Server. For example, X Plugin can have SSL key, certificate, and certificate authority files that differ from those used for MySQL Server. These variables are described at Section 20.5.6.2, “X Plugin Options and System Variables”. Similarly, X Plugin has its own `Mysqlx_ssl_xxx` status variables that correspond to the MySQL Server encrypted-connection `Ssl_xxx` status variables. See Section 20.5.6.3, “X Plugin Status Variables”.

At initialization, X Plugin determines its TLS context for encrypted connections as follows:

- If all `mysqlx_ssl_xxx` system variables have their default values, X Plugin uses the same TLS context as the MySQL Server main connection interface, which is determined by the values of the `ssl_xxx` system variables.

- If any `mysqlx_ssl_xxx` variable has a nondefault value, X Plugin uses the TLS context defined by the values of its own system variables. (This is the case if any `mysqlx_ssl_xxx` system variable is set to a value different from its default.)

This means that, on a server with X Plugin enabled, you can choose to have MySQL Protocol and X Protocol connections share the same encryption configuration by setting only the `ssl_xxx` variables, or have separate encryption configurations for MySQL Protocol and X Protocol connections by configuring the `ssl_xxx` and `mysqlx_ssl_xxx` variables separately.

To have MySQL Protocol and X Protocol connections use the same encryption configuration, set only the `ssl_xxx` system variables in `my.cnf`: