How-To Library

Informatica

Configuring SSL for MySQL Connector in Cloud Data Integration

Abstract

To establish SSL communication for MySQL Connector, you must install the MySQL JDBC and ODBC drivers, version 8.0.12 on the Secure Agent machine. This article explains how to install the MySQL JDBC and ODBC drivers on Windows or Linux systems and configure SSL for a MySQL connection.

Supported Versions

- Cloud Data Integration

Table of Contents

Overview

You can configure SSL for MySQL Connector to securely communicate between Cloud Data Integration and the MySQL database. To configure the SSL protocol for a MySQL connection, you must install and configure the MySQL JDBC and ODBC drivers version 8.0.12 on your system.

To configure SSL for a MySQL connection, perform the following tasks:

1. Install the MySQL JDBC driver to access metadata securely from the MySQL database.
2. Install the MySQL ODBC driver to run mappings to read from or write data to the MySQL database.

   Note: The MySQL ODBC driver that you need to install varies based on the operating system that you use. Ensure that both the MySQL JDBC and ODBC drivers are of 8.0.12 version.
3. Restart the Secure Agent.
4. Update the DTM flag for the MySQL ODBC driver on your Windows or Linux system.
5. Configure SSL for the MySQL connection.

Installing the MySQL JDBC Driver

Install the MySQL JDBC driver to access the metadata from the MySQL database.

Perform the following steps to install the MySQL JDBC driver on the Windows or Linux system:

1. Download the MySQL JDBC driver from the following website: https://downloads.mysql.com/archives/c-j/
2. Select the **Product Version** as **8.0.12**.

3. Select the **Operating System** as **Platform Independent**.
4. Download the **.zip** file for the Windows system or the **.tar** file for the Linux system.
5. Extract the downloaded file and copy the `mysql-connector-java-8.0.12.jar` file.
6. Paste the `mysql-connector-java-8.0.12.jar` file to the following Secure Agent directory, based on the operating system:

   For Windows system: `<Secure Agent installation directory>\apps\Data_Integration_Server\ext\drivers`

   For Linux system: `<Secure Agent installation directoty>/apps/Data_Integration_Server/ext/drivers`

   Note: Before you paste the file, you must ensure to remove any existing MySQL JDBC driver from the Secure Agent directory. Ensure that there is only one MySQL JDBC driver in the directory from where the Secure Agent reads the MySQL JDBC driver.

Installing the MySQL ODBC Driver

Before you use MySQL Connector, you must install the MySQL ODBC driver on Windows or Linux systems so that you can run mappings to read or write data from the MySQL database.

Install the MySQL ODBC Driver on Windows

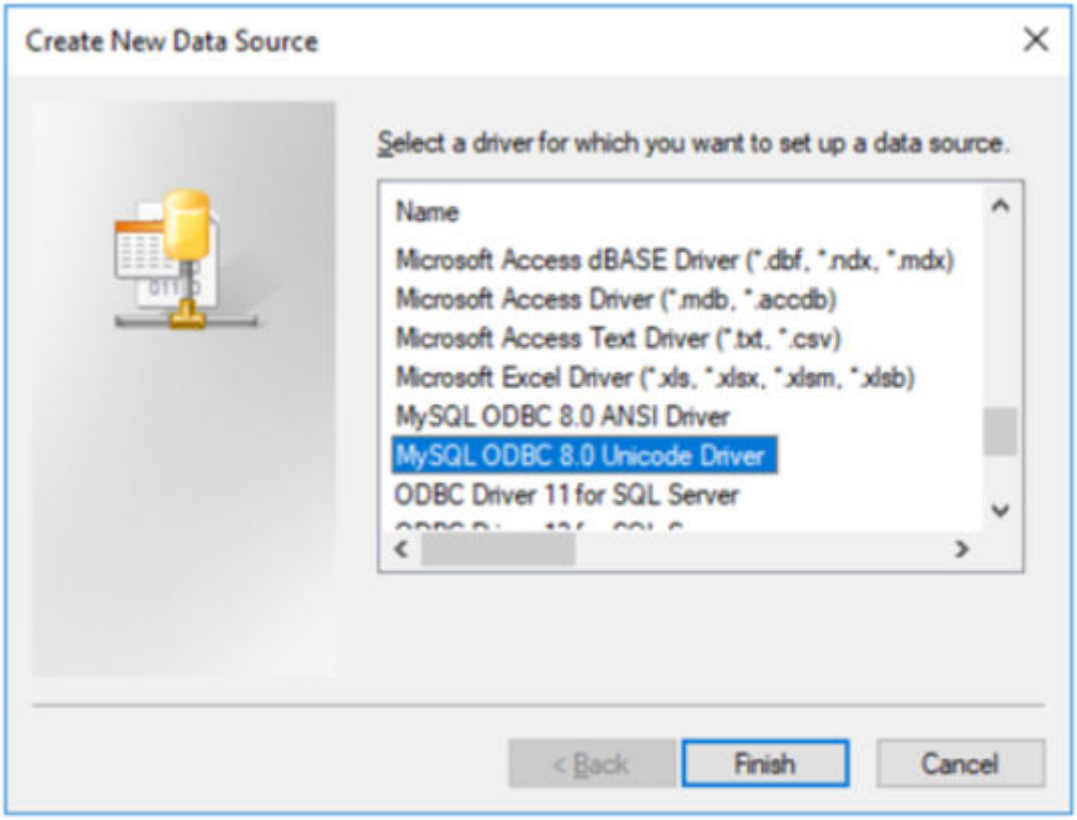
Before you can configure SSL for the MySQL connection, you must install the MySQL ODBC driver version 8.0.12 on the Secure Agent machine.

1. Download the MySQL ODBC driver from the following website: https://downloads.mysql.com/archives/c-odbc/
2. Select the **Product Version** as **8.0.12**.

   Informatica recommends that you use the MySQL ODBC driver version 8.0.12.
3. Select the **Operating System** as **Microsoft Windows**.
4. Select the **OS Version** as **Windows (x86, 64-bit)**.
5. Download the **MSI Installer**.
6. Run the **MSI Installer** and follow the installation wizard to complete the MySQL ODBC driver installation.

After you install the MySQL ODBC driver, check if the **MySQL ODBC 8.0 Unicode Driver** name appears as one of the available ODBC drivers under the **ODBC Data Source Administrator (64-bit)** dialog box.

The following image shows the **MySQL ODBC 8.0 Unicode Driver** name in the **ODBC Data Source Administrator (64-bit)** dialog box when you create a new data source:



Install the MySQL ODBC Driver on Linux

Before you can configure SSL for the MySQL connection, you must install the MySQL ODBC driver version 8.0.12 on the Secure Agent machine.

1. Click on the following link to download the MySQL ODBC driver: https://downloads.mysql.com/archives/c-odbc/
2. Select the **Product Version** as **8.0.12**.

   Informatica recommends that you use the MySQL ODBC driver version 8.0.12.
3. Select the **Operating System** as **Linux- Generic**.
4. Select the **OS Version** as **All**.
5. Download the **Linux - Generic (glibc 2.12) (x86, 64-bit)** file.
6. Extract the downloaded file to a local directory in your system.

   Note: When you extract the file, ensure that you copy all files from the downloaded driver package.
7. In the Secure Agent installation directory, edit the `odbcinst.ini` file, and add the following values:

   ```
   [MySQL ODBC 8.0 Unicode Driver]
   Description = ODBC for MySQL
   Driver = <DRIVER_INSTALL_FILEPATH>/lib/libmyodbc8w.so
   Setup = <DRIVER_INSTALL_FILEPATH>/lib/libmyodbc8w.so
   FileUsage = 1
   ```

   In the Driver and Setup fields, you must specify the file path of the `libmyodbc8w.so` files that you extracted to the local directory in your system.

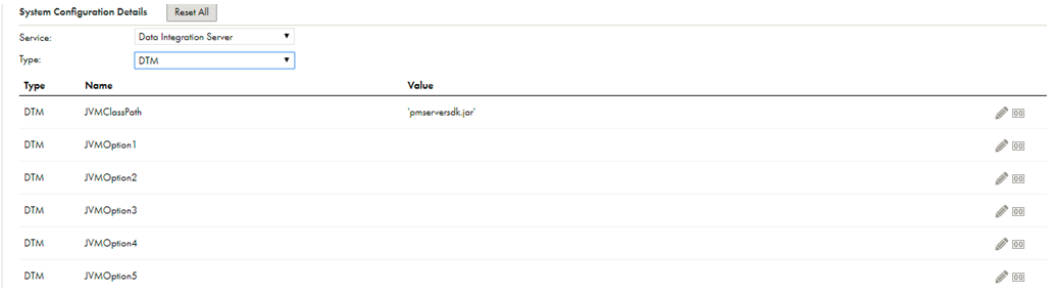
Updating the DTM Flag for the MySQL ODBC Driver

If you had previously installed the MySQL ODBC driver on your system and you now install version 8.0.12 of the MySQL ODBC driver on your Windows or Linux system, you must update the DTM flag.

Perform the following tasks to update the DTM flag:

1. In Administrator, select **Runtime Environments**.
2. On the **Runtime Environments** page, select the Secure Agent from the list of available Secure Agents.
3. In the upper-right corner, click **Edit**.
4. In the **System Configuration Details** page, select the **Type** field as **DTM** for the Secure Agent.

   The following image shows the **System Configuration Details** section:



5. Select the **MySQL_ODBC_DRIVER** driver name under the **Name** column and click on the pencil icon to edit the value.
6. Update the name of the driver that you installed in the **Value** field.
7. Click **Save**.

Configuring SSL for a MySQL Connection

After you install the drivers, you can enable SSL in the MySQL connection and specify the TLS protocols that you want to use for the secure communication.

When you enable SSL for the MySQL connection, you must configure the SSL properties for both the MySQL JDBC and ODBC drivers. Configure the required SSL properties for the JDBC driver so that the Secure Agent can access metadata securely from MySQL. Also, configure the required SSL properties for the ODBC driver so that the Secure Agent runs mappings to securely read from or write data to MySQL.

Note: SSL is not applicable when you use the Hosted Agent.

MySQL Connection Properties

Provide the MySQL properties and configure SSL for the MySQL connection. Specify the SSL protocol for the secure communication.

The following table describes the MySQL connection SSL properties:

Connection property	Description
Use SSL	Determines whether the Secure Agent establishes a secure connection to the MySQL database. When you select this option and the database server supports SSL, the Secure Agent establishes an encrypted connection. If the MySQL database server does not support SSL, the connection either fails or the Secure Agent establishes an unencrypted connection depending on whether you enable or disable the **Require SSL** check box. If you do not select the **Use SSL** check box, the Secure Agent attempts to establish an unencrypted connection.
Verify Server Certificate	If you select **Use SSL** and select this option, the client validates the server certificate that is sent by the database server.
Require SSL	Applicable only if you select **Use SSL**. If you select the **Require SSL** check box, and the MySQL database supports SSL, the Secure Agent establishes an SSL connection. If you select the **Require SSL** check box, and the MySQL database does not support SSL, the Secure Agent attempts to establish an SSL connection but fails. If you clear the **Require SSL** check box, and the MySQL database does not support SSL, the Secure Agent establishes an unencrypted connection.
TLS Protocols	TLS protocols used for the secure communication when you select **Use SSL**. You can select from the following protocols: - TLSv1 - TLSv1.1 - TLSv1.2 Default is TLSv1.2. The TLSv1 and TLSv1.1 protocols are not applicable.

SSL Properties for the JDBC Driver

When you enable SSL, you must specify the required SSL properties for the MySQL JDBC driver.

The following table describes the MySQL connection properties for the JDBC driver version 8.0.12:

Connection property	Description
Trust Certificate Key Store	The path and file name of the truststore file. You must prefix the file path with file colon (`file:`). For example, `file:C:\SSL\mysql_new\truststore`
Trust Certificate Key Store Password	The password for the truststore file.
Client Certificate Key Store	The path and file name of the keystore file. You must prefix the file path with file colon (`file:`). For example, `file:C:\SSL\mysql_new\keystore`

Connection property	Description
Client Certificate Key Store Password	The password to access the keystore file.

JDBC Cipher Suites	Colon-separated cipher suite values in RFC format.
	For example:
	`TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256:`
	`TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256`

SSL Properties for the ODBC Driver

If you configure SSL for the ODBC driver, you must also configure the SSL properties for MySQL ODBC driver.

The following table describes the MySQL connection properties for the ODBC driver version 8.0.12:

Connection property	Description
SSL Certificate Authority	The path and name of the CA certificate. For example, `C:\SSL\mysql_new\ca.pem`
SSL Certificate	The path and name of the client certificate. For example, `C:\SSL\mysql_new\client-cert.pem`
SSL Key	The path and the name of the private key of the client. For example, `C:\SSL\mysql_new\client-key.pem`

SSL Cipher	Colon-separated cipher-suite values in OpenSSL format.
	For example:
	`ECDHE-ECDSA-AES128-GCM-SHA256:`
	`ECDHE-ECDSA-AES256-GCM-SHA384:`
	`ECDHE-RSA-AES128-GCM-SHA256:`

Verify Server's Identity	Verifies the host name in the certificate while verifying the server CA certificate. This property is applicable only when you enable Verify Server Certificate in the SSL properties.

Authors

Gurumoorthy N

Dimple Rai

Acknowledgements

The authors would like to acknowledge Kriti Suwalka, for her technical assistance with this article.