

SYSPRO and SQL Server Encryption Configuration

SYSPRO 8 Technical Article

Last Published: March 2021



Copyright © 2021 SYSPRO Ltd

All rights reserved

No part of this document may be copied, photocopied, or reproduced in any form or by any means without permission in writing from SYSPRO Ltd. SYSPRO is a trademark of SYSPRO Ltd. All other trademarks, service marks, products or services are trademarks or registered trademarks of their respective holders.

SYSPRO Ltd reserves the right to alter the contents of this document without prior notice. While every effort is made to ensure that the contents of this document are correct, no liability whatsoever will be accepted for any errors or omissions.

This document is a copyright work and is protected by local copyright, civil and criminal law and international treaty. This document further contains secret, confidential and proprietary information belonging to SYSPRO Ltd. It is disclosed solely for the purposes of it being used in the context of the licensed use of the SYSPRO Ltd computer software products to which it relates. Such copyright works and information may not be published, disseminated, broadcast, copied or used for any other purpose. This document and all portions thereof included, but without limitation, copyright, trade secret and other intellectual property rights subsisting therein and relating thereto, are and shall at all times remain the sole property of SYSPRO Ltd.

Contents

SYSPRO and SQL Server Encryption Configuration	3
Introduction	3
Audience.....	3
Setting up Data Encryption at Rest using TDE	4
Create a Database Master Key (DMK)	5
Create a certificate.....	6
Create the Database Encryption Key (DEK).....	7
Enable TDE on the SYSPRO database.....	9
Disabling TDE on a SYSPRO Database	10
Backing up Master and Private Keys and Certificates	11
Setting up Data Encryption in Motion using TLS	12
Adding a Certificate using Microsoft Management Console	13
Connecting to SSMS using the certificate	24
Verify you have a driver named: ODBC Driver 17 for SQL Server.....	25
Configuring SYSPRO to connect to SQL Server using TLS	26
Troubleshooting	28
If I have problems can I undo the secure connection?.....	28
How can I verify that a SYSPRO Database is encrypted?.....	29
How can I verify that a SQL Server connection is encrypted?.....	30



SYSPRO and SQL Server Encryption Configuration

INTRODUCTION

This document contains detailed technical information about how to configure SYSPRO, SQL Server and your Windows Server operating system to use **Data Encryption at Rest** and **Data Encryption in Motion** technologies.

A companion document – **SYSPRO and SQL Server Encryption Overview** – introduces data encryption relating to SYSPRO and SQL Server. It summarizes the ability to configure SYSPRO and SQL Server for **Data Encryption at Rest** and **Data Encryption in Motion**. Benchmarks included in the document demonstrate that these technologies have a negligible effect on system performance.

We should also mention that the subject of data encryption and security is moving rapidly in the modern highly regulated world. Technologies that seemed secure a few years ago are now considered vulnerable to eavesdroppers and hackers. This document serves a moment-in-time, but we recommend that you remain vigilant in protecting your data and are aware of newly developing technologies and their implications.

Note: This document includes screenshots to clarify the steps taken. Remember that your version of SYSPRO, SQL Server and/or Windows operating systems may mean that the exact steps or screen images differ. Please consult the relevant resources when setting up your system.

AUDIENCE

This document is aimed at those requiring a detailed explanation about how to configure SYSPRO, SQL Server and Windows to work with **Data Encryption at Rest (TDE)** and/or **Data Encryption in Motion (TLS)**.

For an overview of these technologies and how they relate to SYSPRO (or to see SYSPRO benchmarks relating to these technologies) then see the companion document: **SYSPRO and SQL Server Encryption Overview**.

Setting up Data Encryption at Rest using TDE

Data Encryption at Rest describes the technique of configuring SQL Server so that the physical database files stored on the Windows file system are encrypted.

This ensures that in the event of a network or other security breach, even if someone is able to access the physical database data or log files (or a backup of these files) the information remains secure.

To set up Data Encryption at Rest using TDE (Transparent Data Encryption):

1. Create a Database Master Key (DMK) in the **master** database.
2. Create a certificate in the **master** database to secure the Database Encryption Key (DEK).
3. Create a Database Encryption Key (DEK) in each of your SYSPRO databases to be encrypted.
4. Enable Transparent Data Encryption (TDE) on your SYSPRO databases.

It's critical that you backup the System Master Key, Database Master Key and certificates. If anything goes wrong and you need to restore or move an encrypted database, you will need those keys or certificates. This is covered in the topic below: [Backing up Master and Private Keys and Certificates](#)

When working with Data Encryption at Rest using TDE (or reading additional documentation that relates to this subject) you will encounter many acronyms. This document mostly uses the full phrases, but for your information, some of the common acronyms include:

- TDE – Transparent Data Encryption
- SMK – Service Master Key
- DMK – Database Master Key
- DEK - Database Encryption Key

The examples use the following names:

- SYSPRO company database name: **SysproCompanyLive**
- Certificate name in the master database: **SYSPROCert**
- Certificate issuer: **SYSPRO certificate**

In addition, the placeholder password '**Pass1234!**' is used. It's critical that you replace this with a strong password.

It's assumed that before starting these steps, you already have a successfully running SYSPRO system and want to add Transparent Data Encryption (TDE).

CREATE A DATABASE MASTER KEY (DMK)

This only needs to be performed once on your SQL Server instance.

Using a system administrative SQL login, execute the following SQL statement against your master database.

```
USE master
GO

CREATE MASTER KEY
ENCRYPTION BY PASSWORD = 'Pass1234!'
```

Remember to use a strong password instead of the one shown here.

You can validate that the Database Master Key has been created by querying the `sys.symmetric_keys` view.

For example:

```
SELECT
    name KeyName,
    symmetric_key_id KeyID,
    key_length KeyLength,
    algorithm_desc KeyAlgorithm
FROM sys.symmetric_keys
```

Example output:

KeyName	KeyID	KeyAlgorithm	KeyLength
##MS_DatabaseMasterKey##	101	256	AES_256
##MS_ServiceMasterKey##	102	256	AES_256

Note: The output includes both the Database Master Key (DMK) and Service Master Key (SMK). SQL Server creates the Service Master Key in the master database automatically (i.e. even before you have started the encryption process the row with the KeyName of `##MS_DatabaseMasterKey##` is returned). In this example, the two keys are based on the 256-bit AES encryption algorithm.

CREATE A CERTIFICATE

Next, you need to create a certificate in the master database.

A SQL Server certificate is a digitally-signed, database-level securable binding public and private keys.

For demonstration purposes we are going to use a self-signed certificate – this is protected by the Database Master Key (DMK). However, you should typically use a certificate authority (CA) to issue and sign the certificate. This isn't covered here.

We will create the certificate as follows:

```
CREATE CERTIFICATE SYSPROCert  
WITH SUBJECT = 'SYSPRO certificate'
```

Where **SYSPROCert** is the name of the certificate being created and 'SYSPRO certificate' is designed to be the certificate issuer name. For our purposes, we will use 'SYSPRO certificate'.

Once the CREATE CERTIFICATE statement has finished, we can verify that the certificate has been created using the `sys.certificates` view.

```
SELECT  
    name CertName,  
    certificate_id CertID,  
    pvt_key_encryption_type_desc EncryptType,  
    issuer_name Issuer  
FROM sys.certificates  
WHERE issuer_name = 'SYSPRO certificate'
```

Example output:

CertName	CertID	EncryptType	Issuer
SYSPROCert	258	ENCRYPTED_BY_MASTER_KEY	SYSPRO certificate

This confirms that SQL Server has used the Database Master Key (DMK) to encrypt the certificate.

CREATE THE DATABASE ENCRYPTION KEY (DEK)

Now we are going to work with the SYSPRO databases.

You should repeat this process for each SYSPRO company database (don't forget the SYSPRO system-wide database).

Create the encryption key in your SYSPRO database.

```
USE SysproCompanyLive
GO
```

```
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_256
ENCRYPTION BY SERVER CERTIFICATE SYSPROCert
```

We specified to use the 256-bit AES encryption algorithm and the certificate named **SYSPROCert** used in the prior step.

You will receive the following warning:

Warning: The certificate used for encrypting the database encryption key has not been backed up. You should immediately back up the certificate and the private key associated with the certificate. If the certificate ever becomes unavailable or if you must restore or attach the database on another server, you must have backups of both the certificate and the private key or you will not be able to open the database.

The subject of key and certificate backup is covered later in this document (see: [Backing up Master and Private Keys and Certificates](#)).

We can verify the key by querying the `sys.dm_database_encryption_keys` dynamic management view:

```
SELECT
    DB_NAME(database_id) DbName,
    encryption_state EncryptState,
    key_algorithm KeyAlgorithm,
    key_length KeyLength,
    encryptor_type EncryptType
FROM sys.dm_database_encryption_keys
```

Example output:

DbName	EncryptState	KeyAlgorithm	KeyLength	EncryptType
SysproCompanyLive	1	AES	256	CERTIFICATE

Type CERTIFICATE confirms that a certificate was used to encrypt the Database Encryption Key (DEK).

The EncryptState value '1' means that the database is unencrypted.

Values can include:

Value	Description
0	No database encryption key present, no encryption
1	Unencrypted
2	Encryption in progress
3	Encrypted
4	Key change in progress
5	Decryption in progress
6	The certificate or asymmetric key encrypting the Database Encryption Key (DEK) is being changed

ENABLE TDE ON THE SYSPRO DATABASE

Repeat the following for each SYSPRO database:

We recommend that you read the following Microsoft article about Transparent Data Encryption (TDE) for considerations and restrictions before using TDE.

<https://docs.microsoft.com/en-za/sql/relational-databases/security/encryption/transparent-data-encryption?view=sql-server-2017>

Ensure no one else is using, attempting to use, or backing-up the database.

Issue the following statement to encrypt the database:

```
ALTER DATABASE SysproCompanyLive  
SET ENCRYPTION ON
```

For larger databases, this process can take a considerable time.

When completed we can query the `sys.dm_database_encryption_keys` dynamic management view to see that the database encryption has a state of '3' – encrypted.

Example output:

DbName	EncryptState	KeyAlgorithm	KeyLength	EncryptType
tempdb	3	AES	256	ASYMMETRIC KEY
SysproCompanyLive	3	AES	256	CERTIFICATE

Note that when you implement TDE on any user table, SQL Server also encrypts the **tempdb** database.

This ensures that any temporary information, such as temporary user objects or internal objects (which can expose sensitive data) is also encrypted. This does mean that even if you encrypt a single SYSPRO database, all database access using the temporary database can be affected. However, according to Microsoft (and the SYSPRO benchmarks echo this) the overhead should be minimal.

From then onwards we can query, change data or transact against the TDE-encrypted database and everything will work seamlessly.

The database file itself is encrypted, along with the database log and all backup information.

DISABLING TDE ON A SYSPRO DATABASE

If you decide to disable encryption on one or more SYSPRO databases, you can simply run the following SQL statement:

```
ALTER DATABASE SysproCompanyLive  
SET ENCRYPTION OFF
```

You can verify that the encryption has been disabled by querying the `sys.dm_database_encryption_keys` dynamic management view.

Example output:

DbName	EncryptState	KeyAlgorithm	KeyLength	EncryptType
tempdb	3	AES	256	ASYMMETRIC KEY
SysproCompanyLive	1	AES	256	CERTIFICATE

Note that even though all user databases are now unencrypted, the **tempdb** database is still encrypted. The temporary database **tempdb** will remain encrypted until it is re-created. This occurs when the **SQL Server (MSSQLSERVER)** service is restarted.

If you disable TDE on your SYSPRO database, you can drop the Database Master Key (DMK) and Database Encryption Key (DEK) using the following SQL statements.

The process to remove database encryption must have completed before using these statements:

```
DROP MASTER KEY  
DROP CERTIFICATE  
DROP DATABASE ENCRYPTION KEY
```

Warning: Even if your database is not encrypted, part of the transaction log may remain protected and the certificate may be needed for some operations until the full backup of the database is performed. In addition, the certificate will be needed to restore from the backups created at the time the database was encrypted. Only drop the certificate or encryption key when you are sure that they are no longer required.

BACKING UP MASTER AND PRIVATE KEYS AND CERTIFICATES

It is critical that you backup and safely store your keys and certificates – preferably, immediately after you have created them and **before** you use them.

Failure to perform this adequately can lead to your database being unusable. In addition, you may be unable to restore a database from an encrypted backup.

BACKUP A SERVICE MASTER KEY

To backup your Service Master Key (SMK) use the following SQL statement:

```
USE master
GO

BACKUP SERVICE MASTER KEY
TO FILE = 'C:\BackupLocation\ServiceMasterKey.key'
ENCRYPTION BY PASSWORD = 'Pass1234!'
```

Note: You must use a full path when specifying the target backup file. This includes the file name and extension; otherwise, you may receive an error. If the file already exists, you will also receive an error.

BACKUP UP A CERTIFICATE

You must backup the private key along with the certificate. Use the following SQL statement:

```
BACKUP CERTIFICATE SYSPROCert
TO FILE = 'C:\BackupLocation\SYSPROCert.cer'
WITH PRIVATE KEY(
    FILE = 'C:\BackupLocation\SYSPROCert.key',
    ENCRYPTION BY PASSWORD = 'Pass1234!'
)
```

This generates a file for both the certificate and private key, as well as providing a password for the private key.

Remember to store these files in a remote location separate from your database files – ensuring they are appropriately protected and that you have recorded the password used.

Setting up Data Encryption in Motion using TLS

Data Encryption in Motion describes the technique of configuring SYSPRO and SQL Server so that all communication between SYSPRO and SQL is encrypted. This includes initial connection information, the issuing of SQL statements, and the actual data being passed to-and-from SQL Server.

Data Encryption in Motion ensures that eavesdroppers and hackers can't see what is transmitted. This is particularly useful for private and sensitive information, but also for all information sent between SYSPRO and SQL Server.

When setting up Data Encryption in Motion using TLS (Transport Layer Security) the process is:

1. Set up a certificate on your SQL Server and assign appropriate permissions.
2. Configure a SQL Server instance to use the certificate.
3. Configure SYSPRO to connect to the SQL Server instance using TLS.

The following screenshots were created from a system running Windows 10 and SQL Server 2017. In a real-world example, you would be using a Windows server operating system instead, so details, steps and screen images may differ.

The examples use the following names:

- SQL Server name: **MSSSYSPRO8**
- Domain name: **sysproza.net**
- Hostname: **LEVIOUSD**

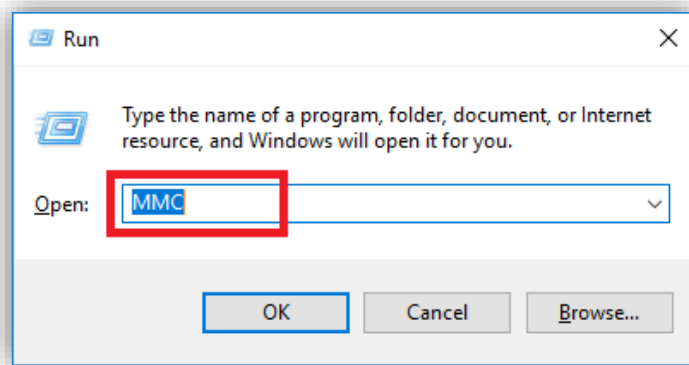
Note: At the time of creating this document (September 2019) the version of SYSPRO that supported TLS (SYSPRO 8 2020 R1) had not been made Generally Available (planned for GA during February 2020). However, the testing and benchmarking was completed with a pre-production version. The remainder of this topic continues as if SYSPRO 8 2020 R1 had already been released. If you require the capability described here, please contact your SYSPRO representative to investigate SYSPRO 8 2020 R1 when it becomes generally available.

ADDING A CERTIFICATE USING MICROSOFT MANAGEMENT CONSOLE

The following steps demonstrate how to create a certificate using the Microsoft Management Console (MMC) and assign appropriate permissions:

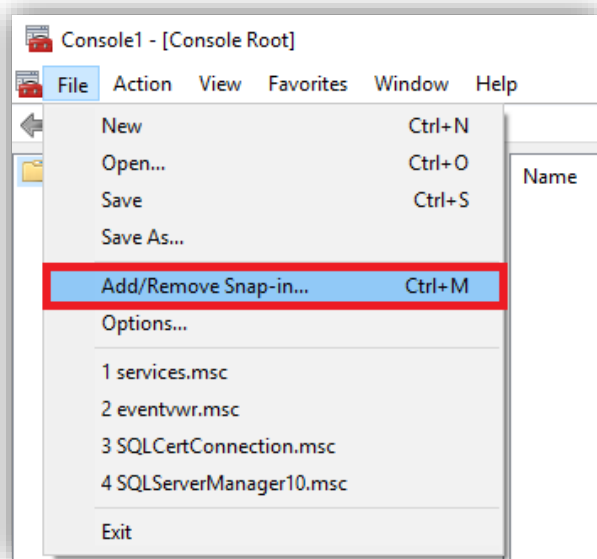
Note: This must be performed on the Windows server on which SQL Server is running.

1. From Windows, open the Run dialog (Windows + R) and run MMC:

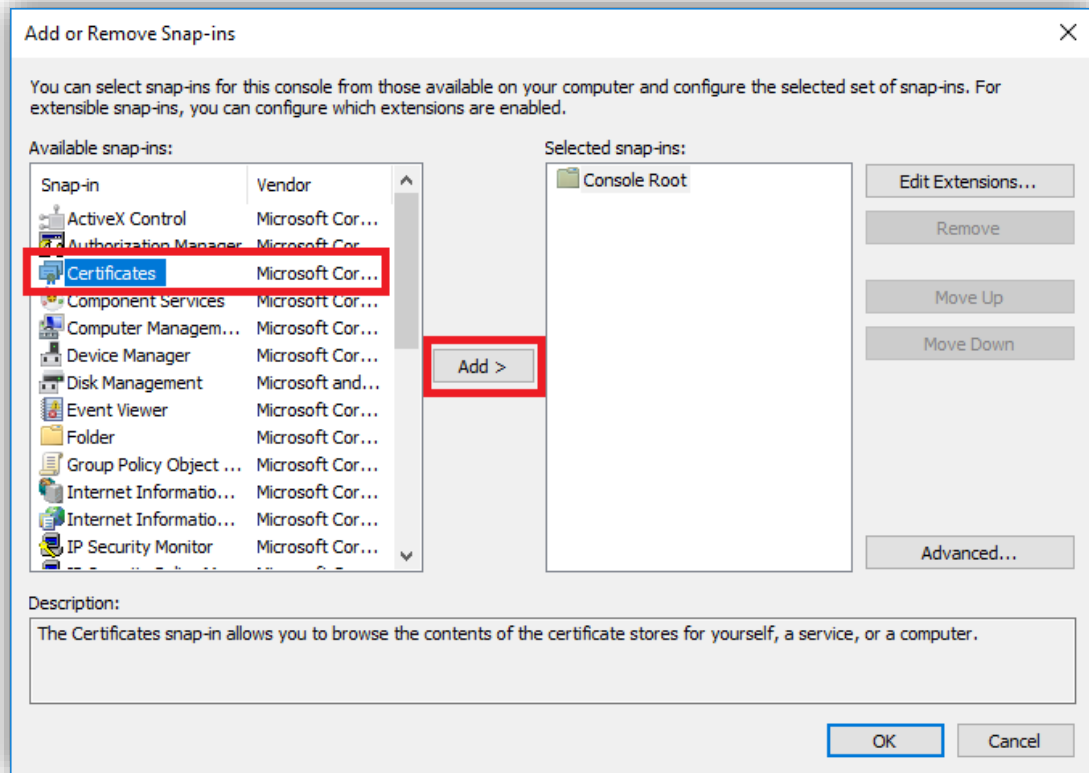


This opens the management console.

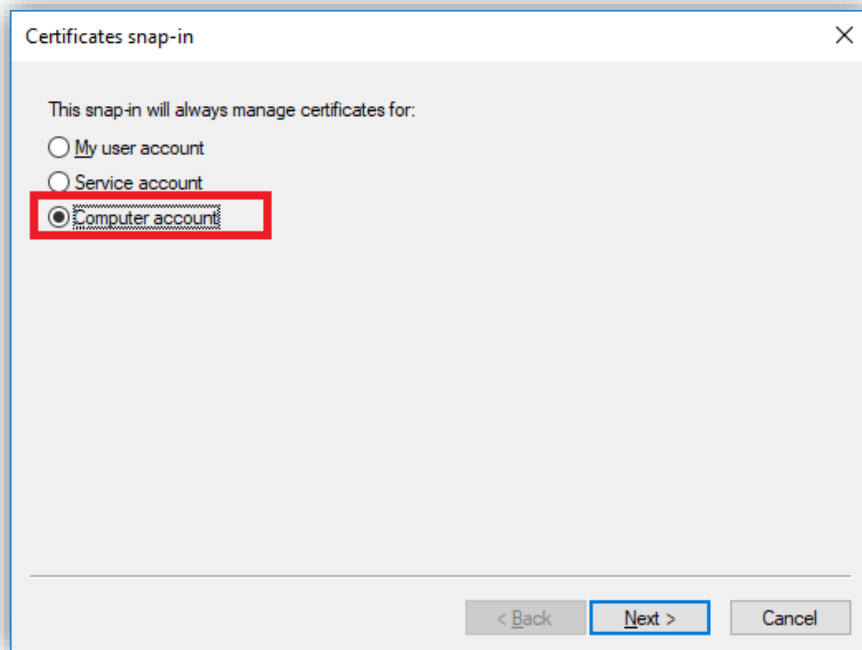
2. Select File > Add/Remove Snap-In... (Ctrl+M):



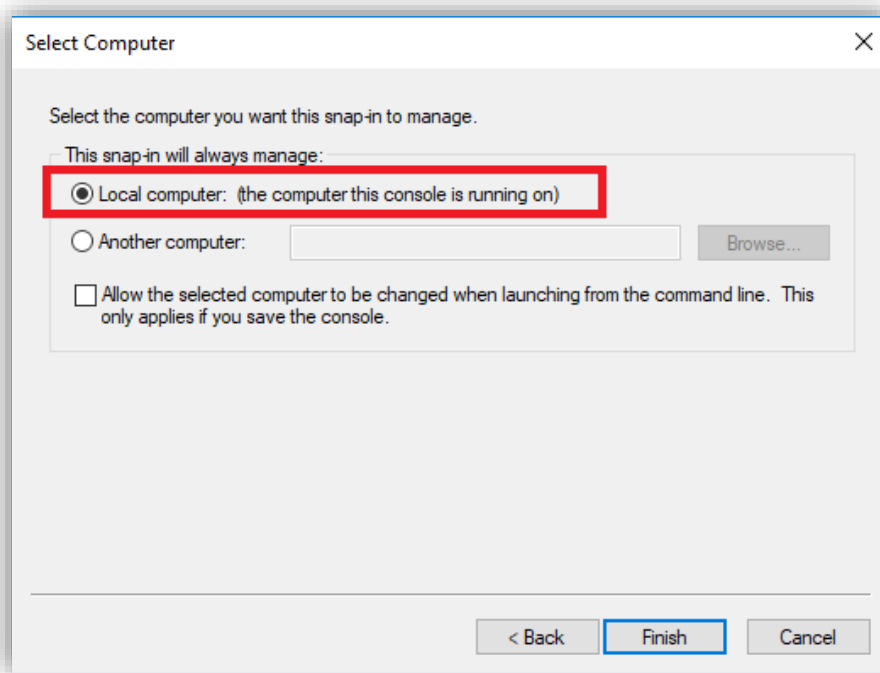
- From the list of **Available snap-ins**, select **Certificates** and add to the list of **Selected snap-ins**.



- Select **Computer account** and **Next**.

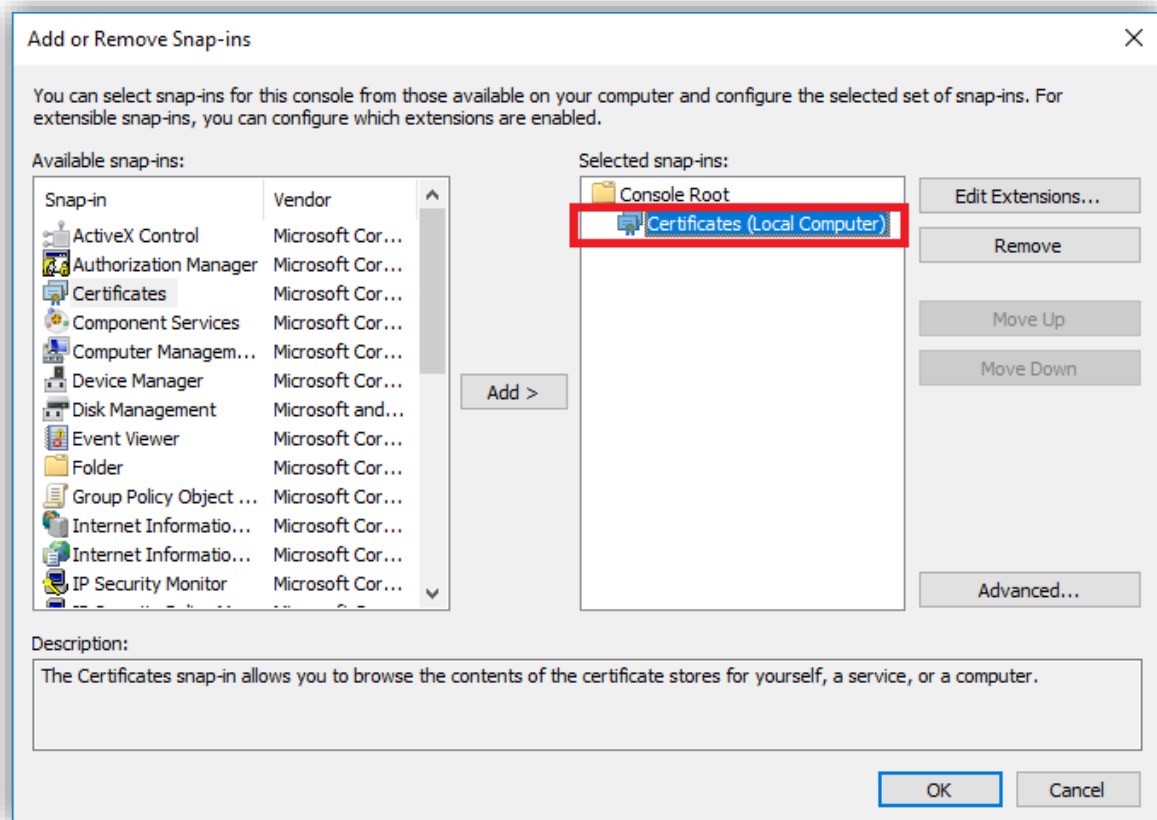


5. Select **Local computer** and **Finish**.

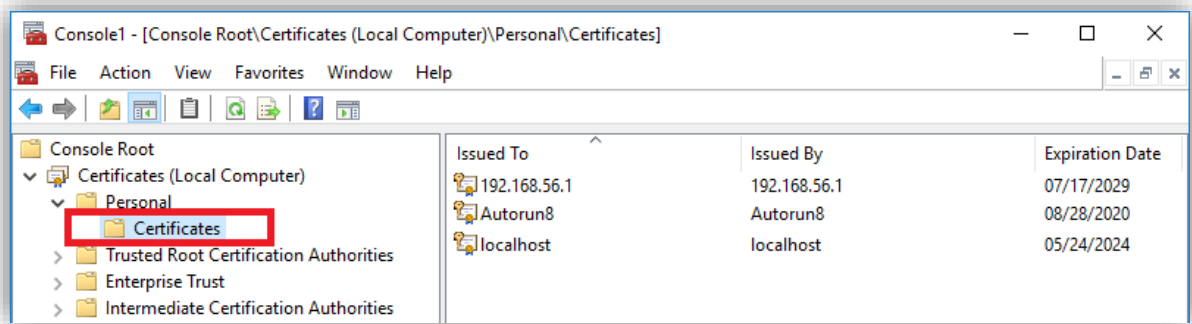


This adds the certificate on the Console Root.

6. Click **OK**.

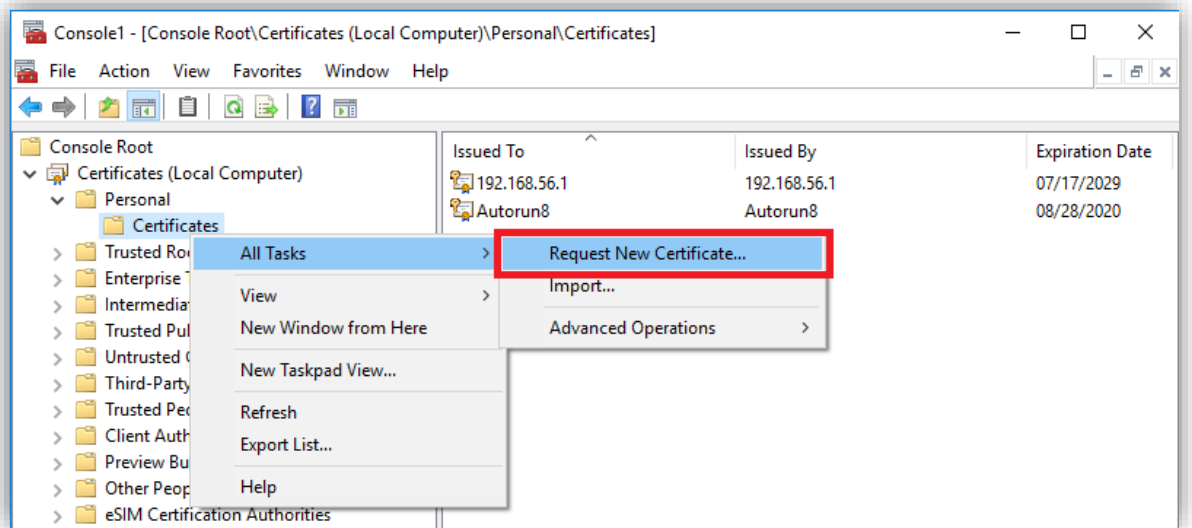


7. From the **Console Root**, expand the **Personal** folder to view details:



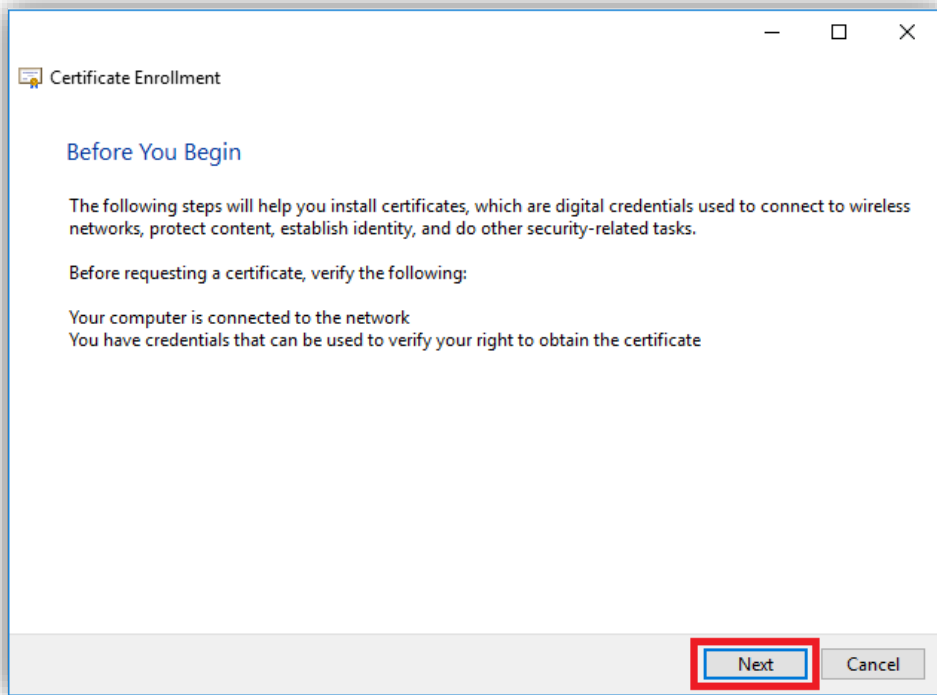
You will then see a **Certificates** folder.

8. Right-click on **Certificates** and select All Tasks > Request New Certificate...

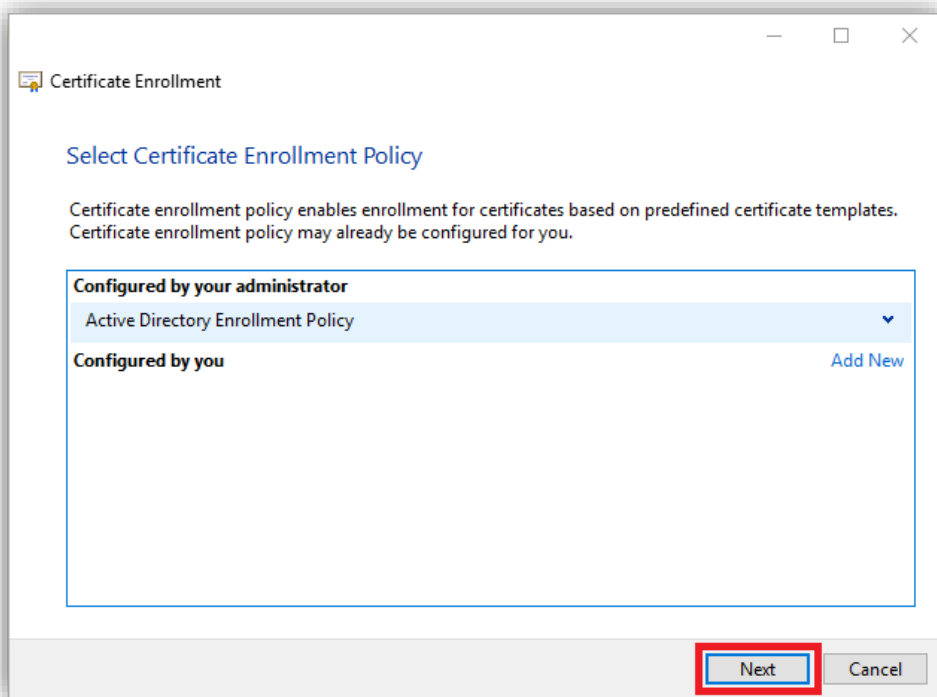


The **Certificate Enrollment** wizard is displayed.

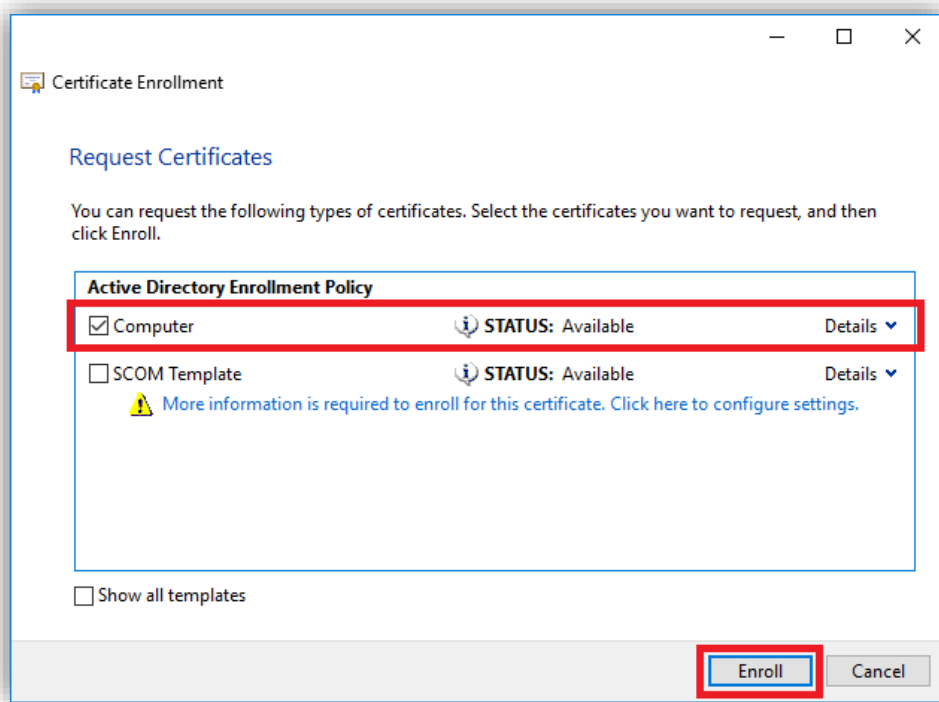
9. Click **Next** at the **Before You Begin** dialog.



10. Click **Next** at the **Certificate Enrollment Policy** dialog.

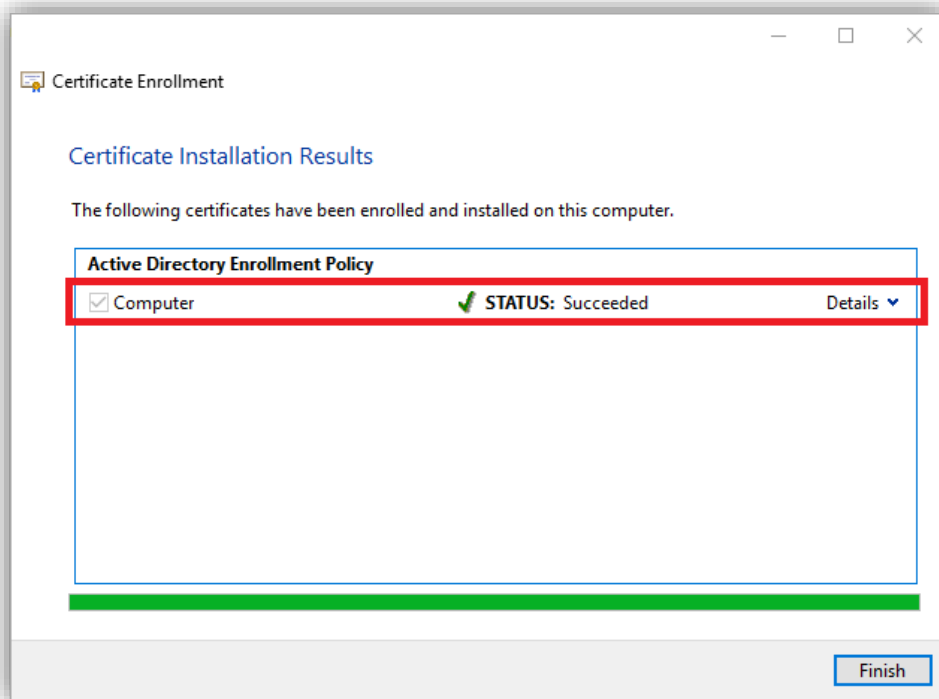


11. Select the **Computer** checkbox and click **Enroll**.

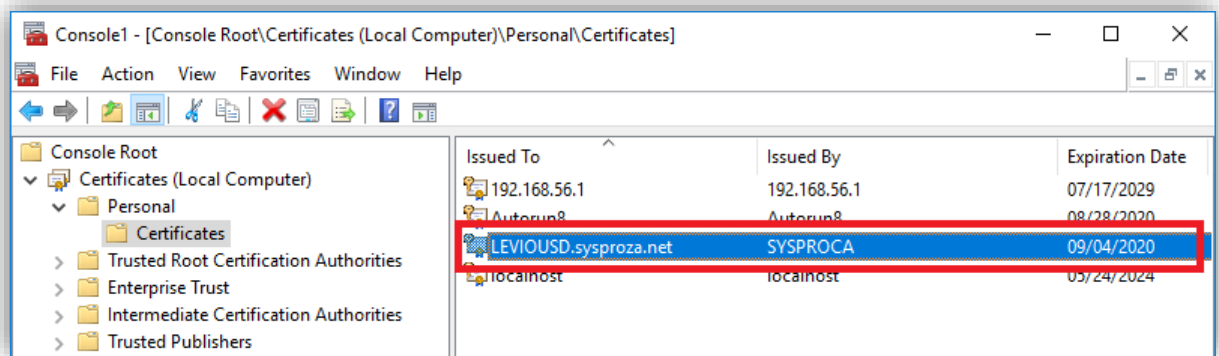


This will take a few seconds.

12. Click **Finish** to complete the installation of the certificate.

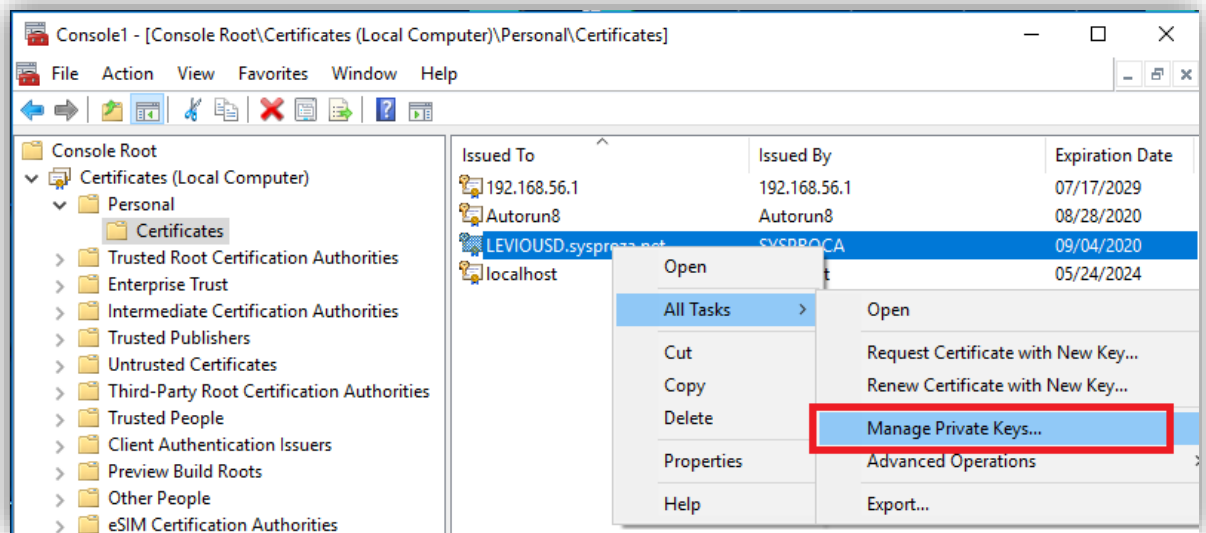


You will see the certificate with the **Issued To**, **Issued By** and **Expiration Date** information:



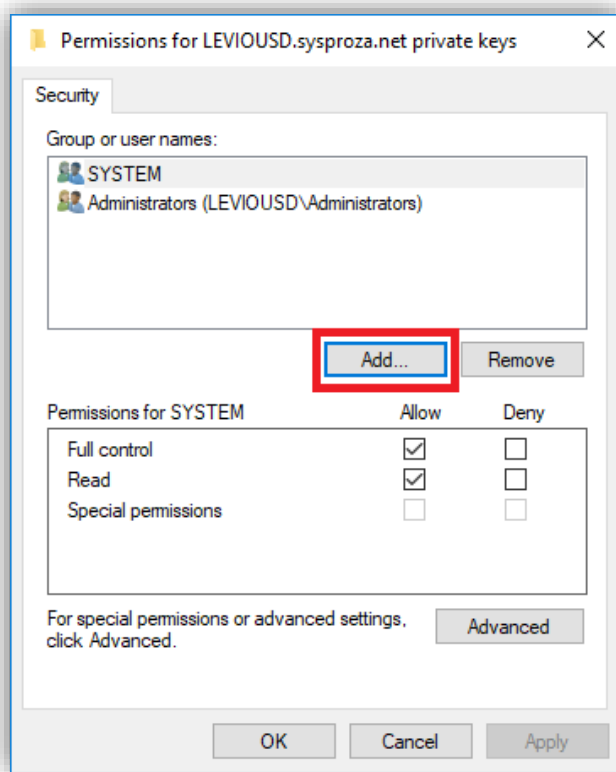
The **Issued To** value is the fully qualified domain name (FQDN) and consists of the hostname-dot-domain name (in our example: **LEVIOUSD.sysproza.net**).

13. Right-click the newly issued certificate and select All Tasks > Manage Private Keys...



You will be prompted to assign permissions.

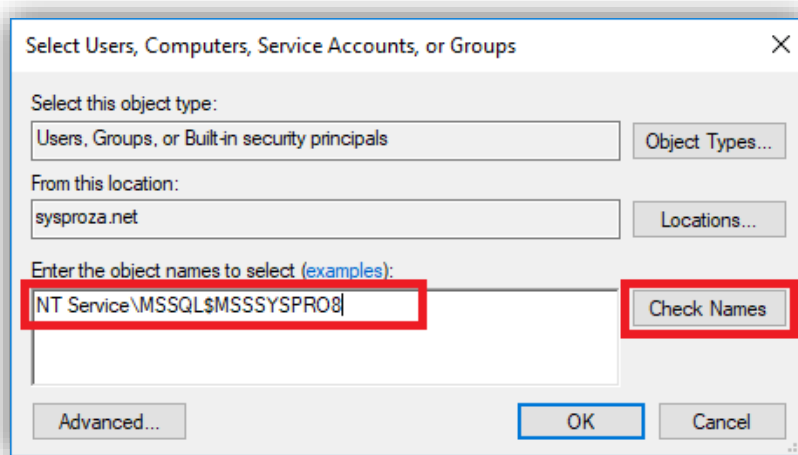
14. Select **Add** and then apply the required Users or Groups.



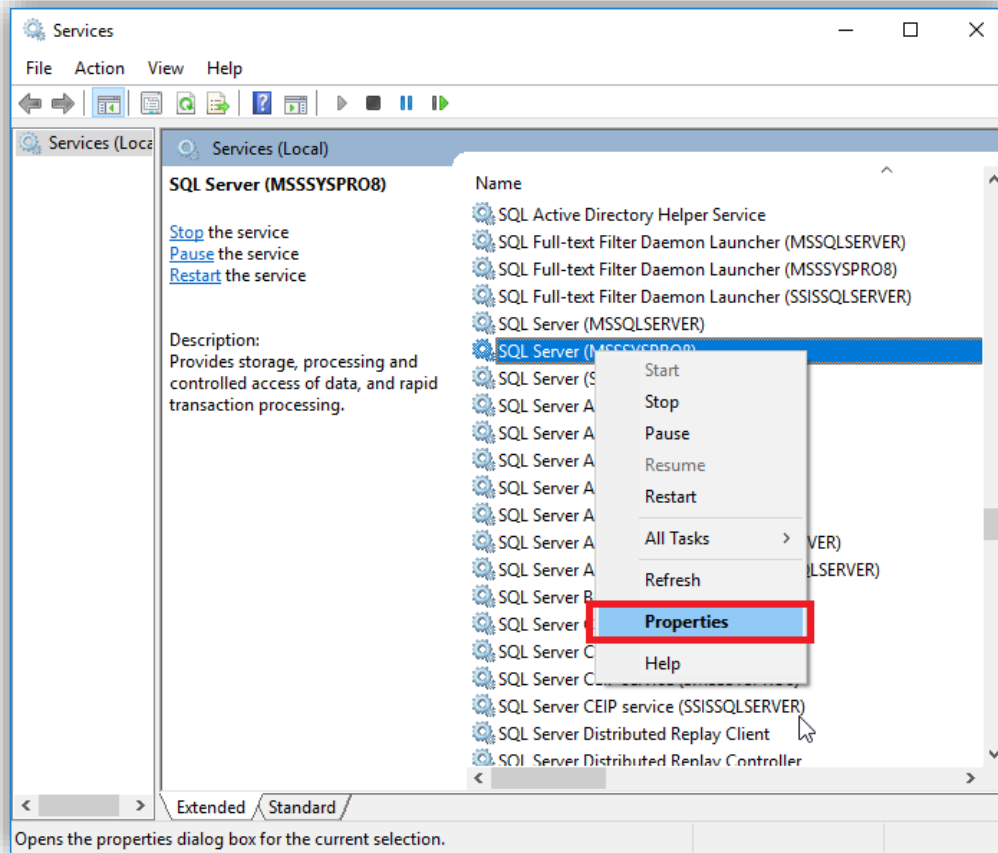
In the example shown, we are giving a specific SQL Instance named **MSSSYSPRO8** access to the newly created certificate.

First, we enter the object name (this requires you to type the full service name) then validate using the **Check Names** function. In our example the full service name is:

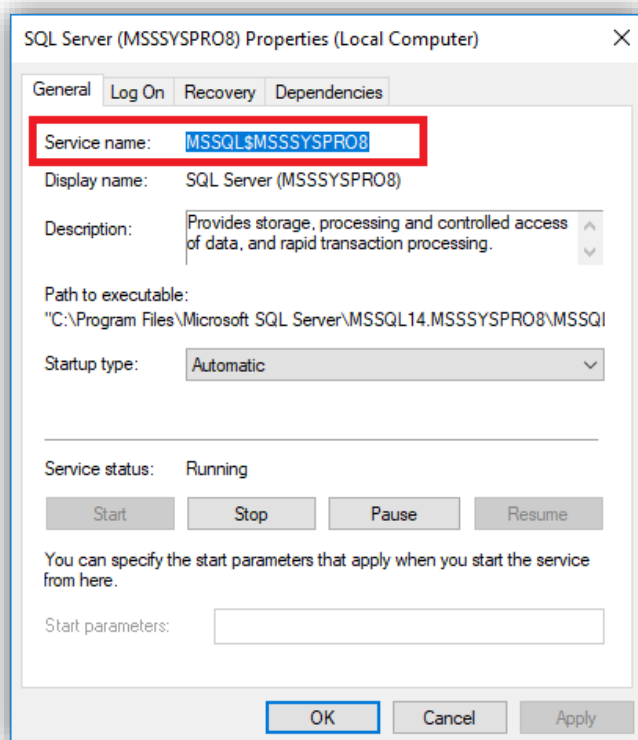
NT Service\MSSQL\MSSSYSPRO8



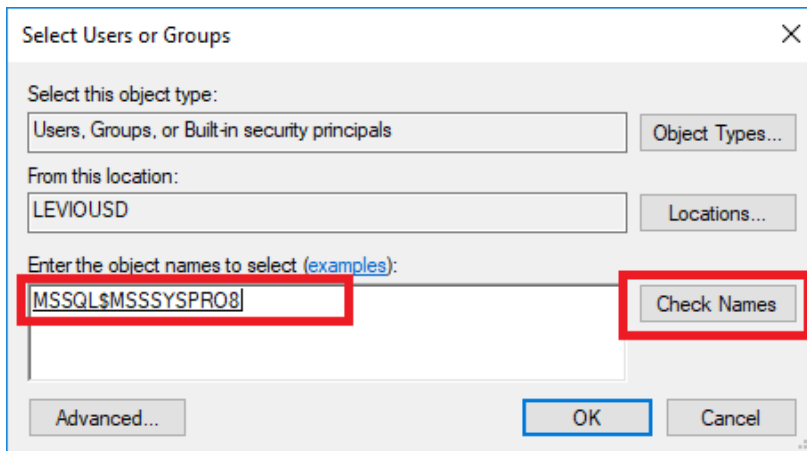
If you don't know the service name, locate your Windows Services and select the SQL Server instance name, double-click or right-click and select **Properties**.



Under the **General** tab you can see the Service name. In this case: **MSSQL\$MSSSYSPRO8**

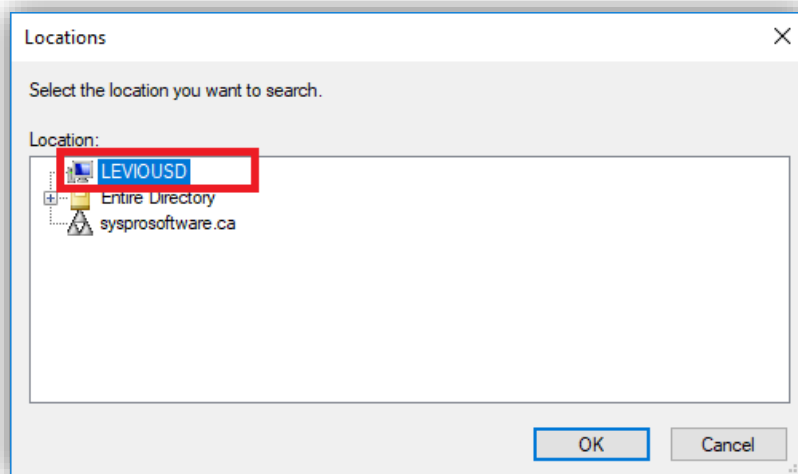


15. Back to the **Select User and Groups** dialog.
Enter the object name (in this case the SQL Service name); validate it using the **Check Names** function; then click **OK**. When validated, it is shown underlined.

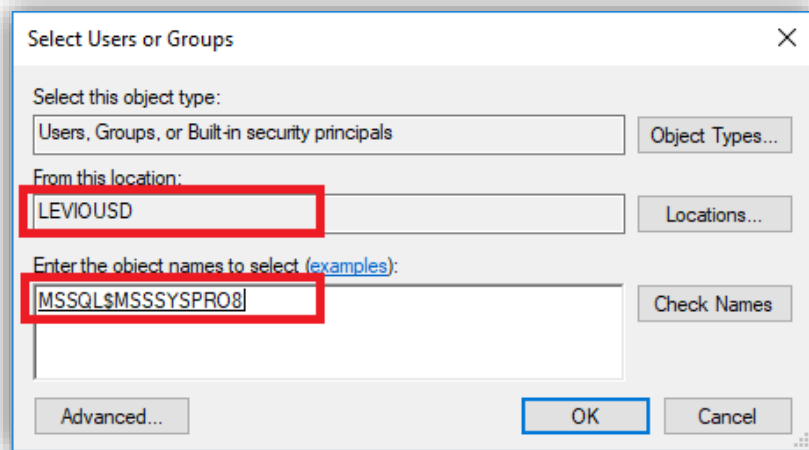


Next, we need to define the location.

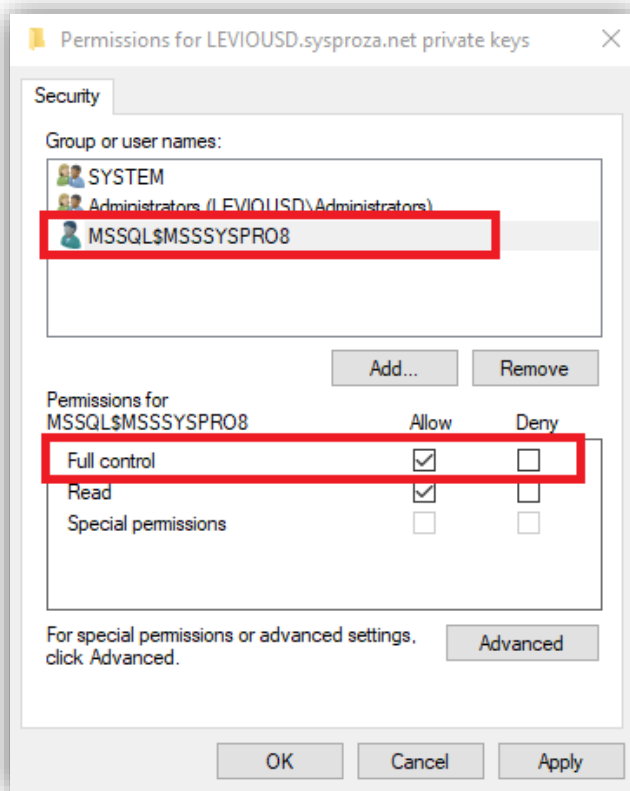
16. Click on **Location** and select the computer hostname **LEVIUOSD** (as this is a local certificate not a domain certificate) and click **OK**.



At this stage, we have defined valid Location and Object names, so click **OK**.



17. Ensure that the permissions allow **Full control** and click **OK**.

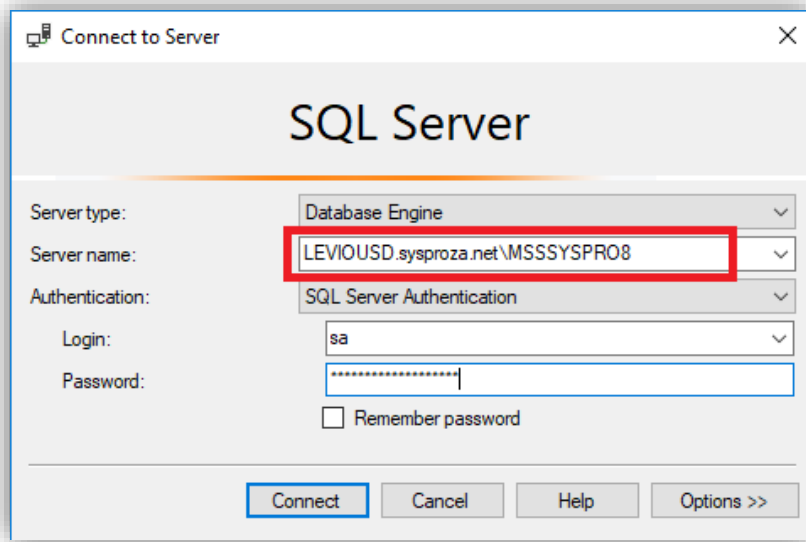


You have now configured the certificate ready for use.

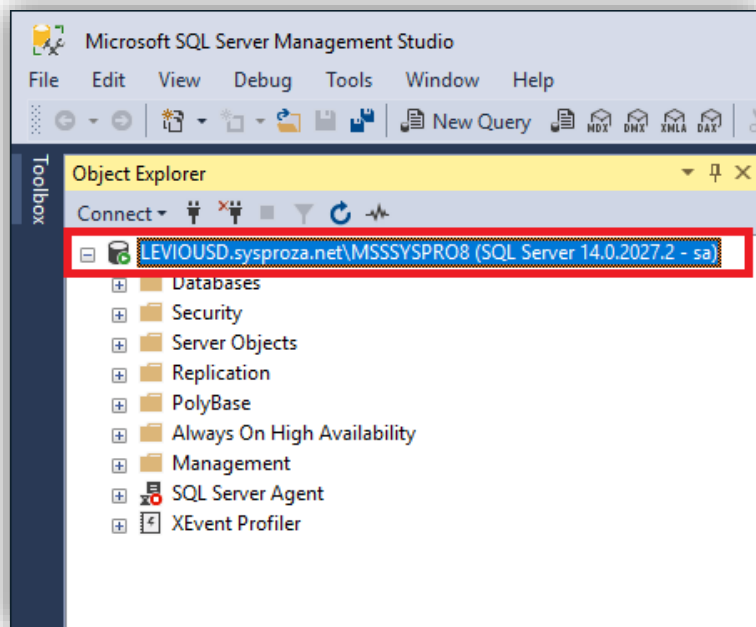
CONNECTING TO SSMS USING THE CERTIFICATE

Once you have created a certificate and set up access control, you can connect to SQL Server Management Studio (SSMS) using the fully qualified domain name and SQL Server name.

1. Run SQL Server Management Studio and select your SQL Server Instance.
You must use the fully qualified SQL Server instance name.
For example:



You are then connected to your SQL Server instance:



2. To confirm that a SQL Server connection is encrypted, see the topic later in this document: [How can I verify that a SQL Server connection is encrypted?](#)

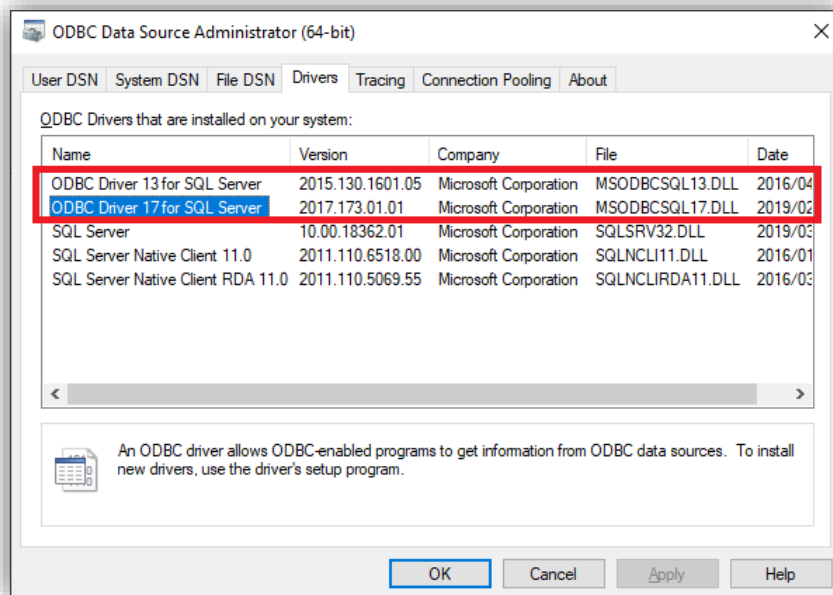
VERIFY YOU HAVE A DRIVER NAMED: ODBC DRIVER 17 FOR SQL SERVER

Check that you have installed the appropriate ODBC Driver. In our example, we are using **ODBC Driver 17 for SQL Server**.

In a 3-tier environment (where SQL Server and the SYSPRO Application server are different servers) you may have to install the ODBC Driver 17 for SQL Server manually on the SYSPRO application server.

To check the available ODBC drivers:

1. From your SYSPRO Application server, use Windows to find and run the application: **ODBC Data Sources (64-bit)**.
2. Select the **Drivers** tab.
In the following example, the driver named **ODBC Driver 17 for SQL Server** has been installed:



3. If the driver isn't displayed, search the web to locate and download it.
See the following link: <https://www.microsoft.com/en-us/download/details.aspx?id=56567>

CONFIGURING SYSPRO TO CONNECT TO SQL SERVER USING TLS

The last step is to set up SYSPRO to connect to the SQL Server instance that was configured and enable a secure connection.

1. Login to SYSPRO as a system administrator:



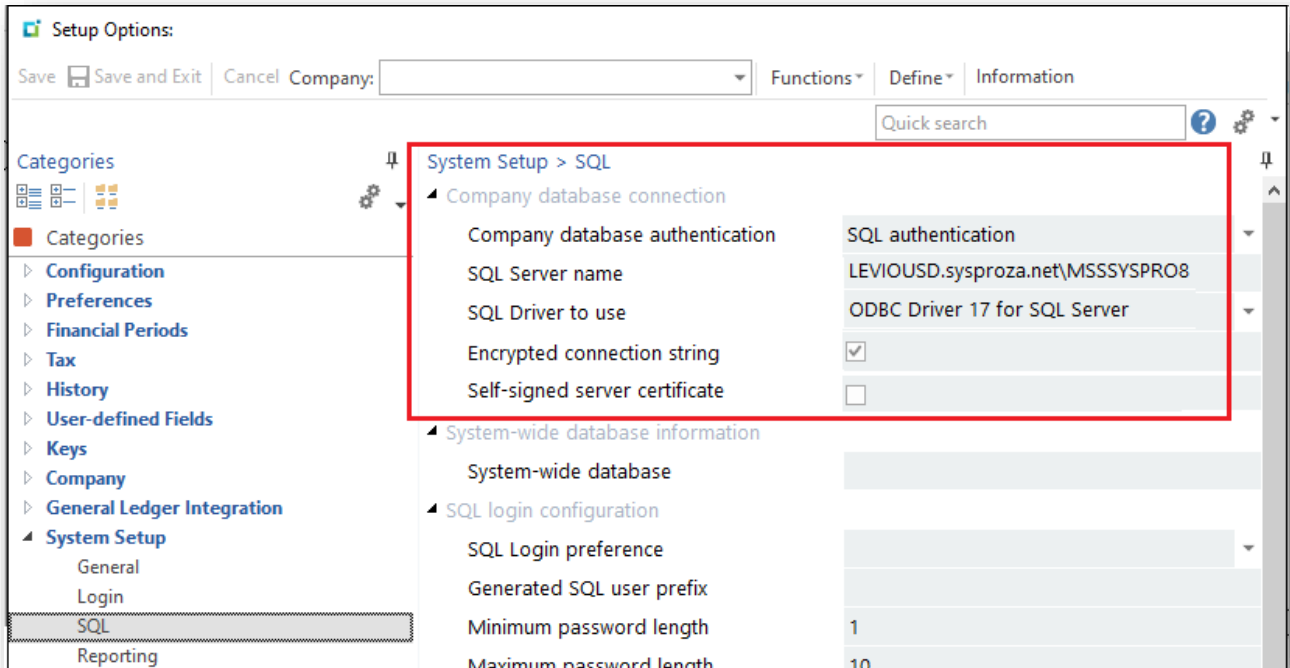
The screenshot shows the SYSPRO 8 login interface. The window title is "SYSPRO". The header area contains the SYSPRO 8 logo and a search icon. The main form has the following fields and controls:

- User name:** A dropdown menu with "ADMIN" selected. Below it, the text "ADMIN - SYSPRO Administrator" is displayed.
- Password:** A text input field with masked characters (asterisks).
- Company:** A dropdown menu with "LIVE" selected and a search icon to its right.
- Company password:** A text input field with masked characters (asterisks).

Below the form is a large blue "Login" button. Underneath the button are two links: "Forgot password?" and "Exit SYSPRO". At the bottom of the window, a message reads: "Program protected as described in Help About SYSPRO".

2. From the SYSPRO menu, run the **Setup Options** program (*Ribbon bar > Setup > Setup Options*), expand the **System Setup** menu and select the **SQL** form.
3. Under **Database connection**, enter the fully qualified SQL Server instance name in the **SQL Server name** field.
4. Select the **ODBC Driver 17 for SQL Server** driver against the **SQL Driver to use** field.

5. Enable the **Encrypted connection string** checkbox and ensure that the **Self-signed server certificate** checkbox is unchecked.



6. Save and Exit SYSPRO.

When next you login to SYSPRO, it will be connecting using the Encryption in Motion (TLS) technology.

Troubleshooting

The following answers common questions relating to SYSPRO and SQL Server encryption.

IF I HAVE PROBLEMS CAN I UNDO THE SECURE CONNECTION?

If you have configured your SYSPRO **System Setup** to use a secure connection, but then find that you are no longer able to login to SYSPRO because the SQL connection has failed, you can manually override the secure connection. This allows you to fix the problem and re-enable the secure connection later.

Warning: Take care when editing your `IMPACT.INI` file manually. Preferably, backup or copy the file before changing it. Some combinations of settings may be invalid and cause further connection issues. You should only edit the `IMPACT.INI` when necessary.

1. Locate your `IMPACT.INI` file in the SYSPRO application server `WORK` folder.

The entry that starts `SQLCSO=` contains information relating to the SQL Server driver, encryption and failover settings. In the examples used earlier, the `IMPACT.INI` file was as follows:

```
[Database Settings]
SQLLGN=SQLSERVER
SQLSSN=LEVIOUSD.sysproza.net\MSSSYSR08
SQLDBN=Syspro80db
SQLADM=U4sJt5orj8qIxs0zFCMU8RL+ogjzQjYIHT0r8KcILjXLsLUIA30vwX+BDec/
SQLSTD=htbZS++x3aLRVdMOPsd8HqK6/yQQMy1Fhh51tZgzyMwWKZXogtrd958p2g6f
SQLBLK=C:\Temp
SQLCPG=01252
SQLCSO=2,1,0,0
; End of IMPACT.INI
```

The first digit of the `SQLCSO=` entry represents the **SQL Driver to use** setting:

- 0 – SQL Server
- 1 – ODBC Driver 13 for SQL Server
- 2 – ODBC Driver 17 for SQL Server

The second digit stores the **Encrypted connection string** option:

- 0 – Unchecked (no encryption)
- 1 – Checked (encrypted)

The third digit stores the **Self-signed server certificate** option:

- 0 – Unchecked (requires provided certificate)
- 1 – checked (no certificate provided – less secure)

The fourth digit is the **MultiSubNetFailover property** (and is not discussed in this document):

0 – Unchecked

1 – Checked

If you are having problems connecting to SQL Server once you have enabled the **Encrypted connection string** option in the **System Setup** program, you can edit your `IMPACT.INI` file and change the `SQLCSO` entry to:

```
SQLCSO=2,0,0,0
```

2. Save the `IMPACT.INI` file and immediately attempt to login to SYSPRO as normal. Once you can login successfully you should use the **System Setup** program to make any subsequent changes to your setup options rather than edit this file directly.

HOW CAN I VERIFY THAT A SYSPRO DATABASE IS ENCRYPTED?

SYSPRO contains a SQL Health Dashboard that lets you view various aspects of SYSPRO and SQL Server.

From SYSPRO 8 2020 R1, the SQL Health Dashboard (*Ribbon bar: Administration > SQL Health Dashboard*) shows whether the selected SYSPRO database is encrypted using Transparent Data Encryption (TDE).

Select the required database, view the Database Details and look for the TDE encryption status as shown below (in this example the database is not encrypted).

The screenshot shows the SQL Health Dashboard interface. On the left, there is a table of SYSPRO Databases. The table has columns for Company, Name, and Database. The selected database is 'Syspro80db' under the 'System' company. On the right, the 'Database Details' pane is open, showing various properties for the selected database. The 'TDE encryption status' property is highlighted with a red box and shows the value 'No database encryption key present, no encryption'.

Company	Name	Database
System	System database	Syspro80db
EDU1	The OUTDOORS Company	Syspro80Edu1

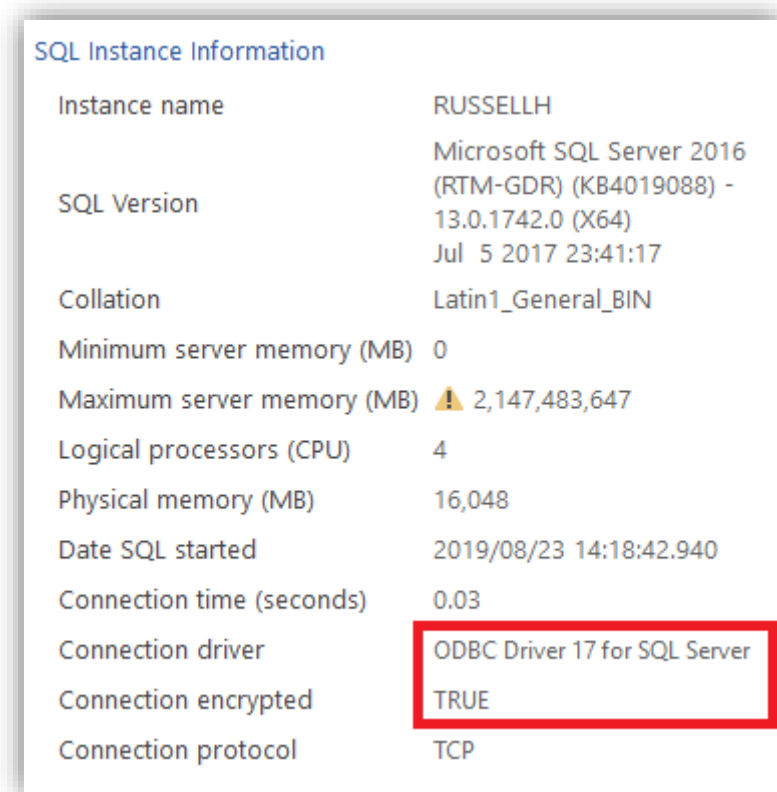
Property	Value
Auto create statistics	<input checked="" type="checkbox"/>
Auto update statistics	<input checked="" type="checkbox"/>
Status	ONLINE
Read only	<input type="checkbox"/>
Last full backup	2019/09/27 13:15:39.000
Last log backup	2019/09/27 13:15:39.000
TDE encryption status	No database encryption key present, no encryption

HOW CAN I VERIFY THAT A SQL SERVER CONNECTION IS ENCRYPTED?

SYSPRO contains a SQL Health Dashboard that lets you view various aspects of SYSPRO and SQL Server.

From SYSPRO 8 2020 R1, the SQL Health Dashboard (*Ribbon bar: Administration > SQL Health Dashboard*) shows whether the current SQL connection is encrypted.

View the **SQL Instance Information** tab and look for the connection information, as shown below:



SQL Instance Information	
Instance name	RUSSELLH
SQL Version	Microsoft SQL Server 2016 (RTM-GDR) (KB4019088) - 13.0.1742.0 (X64) Jul 5 2017 23:41:17
Collation	Latin1_General_BIN
Minimum server memory (MB)	0
Maximum server memory (MB)	⚠ 2,147,483,647
Logical processors (CPU)	4
Physical memory (MB)	16,048
Date SQL started	2019/08/23 14:18:42.940
Connection time (seconds)	0.03
Connection driver	ODBC Driver 17 for SQL Server
Connection encrypted	TRUE
Connection protocol	TCP

In addition, you can use SQL Server Management Studio (SMSS) to query the current connection status of any process.

For example, you can run the following SQL statement using SMSS to identify each SYSPRO process connected to SQL Server and return whether it's using an encrypted connection and which ODBC driver is being used.

```

SELECT
    a.connect_time,
    a.session_id,
    b.login_name,
    b.program_name,
    a.connect_time,
    a.net_transport as Connect_Protocol,
    a.encrypt_option as Encrypted_Connection,
    a.auth_scheme,
    a.protocol_type,
    CASE SUBSTRING(CAST(a.protocol_version as binary(4)),1,1)
        WHEN 0x71 then 'SQL Server Driver'
        WHEN 0x74 then 'ODBC Driver nn for SQL Server'
        ELSE 'Unknown'
    END as Driver_Used
FROM sys.dm_exec_connections a
left join sys.dm_exec_sessions b ON (a.session_id = b.session_id)
WHERE b.program_name IN ('SYSPRO')

```

Example results:

	connect_time	session_id	login_name	program_name	connect_time	Connect_Protocol	Encrypted_Connection	auth_scheme	protocol_type	Driver_Used
1	2019-09-05 12:00:47.413	53	syspro	SYSPRO	2019-09-05 12:00:47.413	TCP	TRUE	SQL	TSQL	ODBC Driver nn for SQL Server
2	2019-09-05 14:31:57.233	101	sa	SYSPRO	2019-09-05 14:31:57.233	TCP	FALSE	SQL	TSQL	SQL Server Driver
3	2019-09-05 13:45:31.690	103	syspro	SYSPRO	2019-09-05 13:45:31.690	TCP	TRUE	SQL	TSQL	ODBC Driver nn for SQL Server
4	2019-09-05 14:59:53.730	107	syspro	SYSPRO	2019-09-05 14:59:53.730	TCP	TRUE	SQL	TSQL	ODBC Driver nn for SQL Server
5	2019-09-05 15:24:44.130	108	syspro	SYSPRO	2019-09-05 15:24:44.130	TCP	TRUE	SQL	TSQL	ODBC Driver nn for SQL Server
6	2019-09-05 15:24:44.283	109	syspro	SYSPRO	2019-09-05 15:24:44.283	TCP	TRUE	SQL	TSQL	ODBC Driver nn for SQL Server

Note: The **Encrypted_Connection** column shows whether the SYSPRO connection to SQL Server is encrypted. In this example (used to demonstrate SQL Encryption) we have a combination of SYSPRO 7 and SYSPRO 8 users connected to the same SQL Server instance. The SYSPRO 7 connection is the one shown as **FALSE** under the **Encrypted_Connection** column.



www.syspro.com

Copyright © SYSPRO. All rights reserved.
All brand and product names are trademarks or
registered trademarks of their respective holders.

