YubiHSM 2 for Microsoft SQL Server Deployment Guide

Yubico

May 05, 2022
## CONTENTS

1 Introduction 1
   1.1 YubiHSM 2 for Microsoft SQL Server Guide 1
   1.2 Introduction to Always Encrypted 1

2 Prerequisites and Preparations 3
   2.1 Configuration for this Integration 3

3 Basic Setup of YubiHSM 2 and SQL Server 5
   3.1 Installing and Configuring YubiHSM 2 5
   3.2 Creating a Test Database 6

4 Use SSMS to Generate the CMK and CEK 9
   4.1 Generate the CMK 9
   4.2 Validate Generation of the CMK 12
   4.3 Generate the CEK 13

5 Use PowerShell Script to Generate the CMK and CEK 17
   5.1 Create a CMK in the YubiHSM 2 with CNG Provider (KSP) 17
   5.2 Import SQL Server Module 18
   5.3 Connect to your Database 18
   5.4 Create SQL CMK Settings Object for your CMK 18
   5.5 Create CMK Metadata in Database 18
   5.6 Generate CEK, Encrypt with CMK, and Create CEK Metadata in Database 19
   5.7 Customize the Script 21
   5.8 Validate Generation of the CMK and the CEK 21

6 Encrypt Database Columns 23
   6.1 Encrypt Database Column with PowerShell-Generated Keys 23
   6.2 Encrypt Database Column with SSMS-generated Keys 29
   6.3 Verify Encrypted Database Column 30

7 Configure SSMS for Database Encryption 31

8 Back Up Key Material 35

9 Getting Help and Further Reading 37

10 Terminology 39

11 Copyright 41
1.1 YubiHSM 2 for Microsoft SQL Server Guide

In a Microsoft SQL Server environment, the Column Master Key (CMK) must be protected in hardware. The YubiHSM 2 protects the CMK in hardware and guards the Microsoft SQL Server database encryption services.

This guide is intended to help systems administrators deploy YubiHSM 2 for use with Microsoft SQL Server in a Windows server environment. The expected outcome is that the Column Master Key (CMK) is created securely on a YubiHSM 2 and that a hardware-based backup copy of key materials has been produced.

These guidelines for deployment cover basic topics, so the instructions should be modified as required for your specific environment. It is assumed that you are familiar with the concepts and processes for working with Microsoft SQL Server. It is also assumed that the installation is performed on a single Microsoft SQL Server database, but the concept can be extended to more servers and databases.

Important: Before deploying to production, we recommend that you install and test the Microsoft SQL Server installation and setup of the YubiHSM 2 in a test or lab environment. For guidance on enabling the Microsoft SQL Server feature Always Encrypted in a production environment, see the Microsoft SQL Docs for Always Encrypted.

1.2 Introduction to Always Encrypted

Introduced in 2016, Microsoft SQL Server’s Always Encrypted feature enables the encryption of selected columns in a database.


The Always Encrypted encryption mechanisms rely upon two cryptographic keys, described in detail in the Microsoft SQL Docs, Overview of Key Management for Always Encrypted. In brief:

- The Column Encryption Key (CEK) is a symmetric key used for encrypting the contents of a selected database column.
- The Column Master Key (CMK) is an asymmetric key that is used for protecting the encryption key. The CMK for Always Encrypted can be protected in a local key store, which is in the scope of this document, or in a centralized key store, which is not in scope.

A CMK can be stored in a local key store that supports Microsoft’s Cryptography Next Generation (CNG) API. In order to protect the CMK in hardware, a hardware security module (HSM) should be used. In this scenario, Always Encrypted accesses the HSM through the CNG API by using a key storage provider (KSP).
In order to protect the CMK in hardware, the YubiHSM 2 can be deployed as the local key store. Microsoft’s Always Encrypted accesses the YubiHSM 2 through the KSP that is provided with the YubiHSM software tools. With this setup, the Microsoft SQL Server Management Studio (SSMS) can be used to manage the CMK in the YubiHSM 2 device. This deployment guide describes two ways to generate the CMK and CEK in YubiHSM 2:

- By using the SSMS wizard, as described in *Use SSMS to Generate the CMK and CEK*.
- By running a PowerShell script, as described in *Use PowerShell Script to Generate the CMK and CEK*. 


The audience of this document is an experienced system administrator with a good understanding of Microsoft SQL Server management. In addition, it is helpful to be familiar with the terminology, software and tools specific to YubiHSM 2. As a primer for these, refer to the Terminology.

In order to follow the steps provided in this guide, the following prerequisites must be met:

- Microsoft Windows Server 2012 SP2 or higher, with Microsoft .NET Framework 4.7.2 or higher. The operating system should be installed in a secure computer network. The system administrator must also have elevated system privileges.
- Access to Microsoft SQL Server 2017 with SQL Server Management Studio (SSMS) 2017 or higher.
- YubiHSM 2 software and tools for Windows downloaded from the Yubico YubiHSM 2 Release page and available on the system to be used.

Note: The 32-bit version of the YubiHSM KSP DLL is needed for use with SSMS.

- Two (2) YubiHSM 2 devices, one for deployment and one for backup in hardware.
- Your organization policies may require key custodians to be available for the YubiHSM 2 deployment. For more information about key custodians and the associated $M$ of $N$ key shares, see “Understanding Key Splitting and Key Custodians” in the YubiHSM 2 for Windows Server Deployment Guide.

### 2.1 Configuration for this Integration

For the integration described in this guide, the following hardware and software configuration was used:

- Microsoft Windows Server 2012 SP2.
- Microsoft .NET Framework 4.7.2.
- Microsoft SQL Server 2017.
- Yubico YubiHSM v 2.1.2.
- Yubico YubiHSM software tools v TBD. In particular, the 32-bit YubiHSM KSP DLL is needed for use with SSMS (which is a 32-bit application).
3.1 Installing and Configuring YubiHSM 2

Install and configure the YubiHSM 2 device and software using the instructions in the following sections in the YubiHSM 2 Windows Deployment Guide—Configure YubiHSM 2 Key Storage Provider for Microsoft Windows Server.

- Install the YubiHSM 2 Tools and Software
- Configure the Primary YubiHSM 2 Device
- Configure the YubiHSM 2 Software

When these instructions have been completed, the YubiHSM 2 should be configured with — for example — one domain with a wrap key (id 0x0002), an application authentication key (id 0x0003), and an audit key (id 0x0004). The configuration of the YubiHSM 2 can be inspected by using the YubiHSM-Shell in a command prompt as shown in the screenshot below.

![YubiHSM configuration example](image)

Figure - Example of the YubiHSM 2 basic configuration
3.2 Creating a Test Database

Create a test database that will be used for the Always Encrypted deployment with YubiHSM 2. If you already have a Microsoft SQL Server database installed, you can skip ahead to Configure SSMS for Database Encryption.

Note: At least one row with values needs to be inserted into the database table before the columns are encrypted (see Figure 4 below).

Step 1 Create a test database:

Launch Microsoft SQL Server Management Studio (SSMS) 2018, right-click on the Databases icon, select New Database..., and enter an appropriate name for the database. In this guide, a test database named “Sales3” will be used for the tests with Always Encrypted in conjunction with YubiHSM 2.

![New Database]

Figure - Example of test database

Step 2 Create table:

Expand Databases > Sales3 > Tables, right-click on Tables and select Create new table. Add some columns, for example “Name”, “Address”, “ZipCode”, “City”, “Country”. Save the table and give it the name “Table_Customers” for example.
Figure - Example of test table

**Step 3** Add one or more rows to the table:

Expand **Databases > Sales3**, right-clicking on **Table** and selecting **New > Query**… Use the SQL Query window to insert rows into the database table, for example, with the SQL query shown below.

Figure - Example of SQL query to insert values into the table
CHAPTER
FOUR

USE SSMS TO GENERATE THE CMK AND CEK

The Microsoft SQL Server Management Studio (SSMS) provides tools for manual creation of the CMK and CEK. These can also be generated using a PowerShell script (see Use PowerShell Script to Generate the CMK and CEK). Note that all the examples and screenshots in this document are based on different key names being used for the SSMS wizard and the PowerShell script.

4.1 Generate the CMK

Step 1 To generate the CMK for a database:

Launch SSMS and expand Databases > Database-Name > Security > Column Master Key. We use the example shown below, expanding the tree Databases > Sales3 > Security > Column Master Key.
Step 2  Edit Master Key Values.

Right-click on Column Master Keys, and select New Column Master Key… In the New Column Master Key window, enter the following values:

- In the Name text field, enter an appropriate name for the CMK, for example, CMK-YubiHSM-SSMS.
- In the Key Store drop-down list, select Key Storage Provider (CNG).
- In the Select a provider drop-down list, select YubiHSM Key Storage Provider.
Step 3 Generate an asymmetric key-pair on the YubiHsm 2, click **Generate Key**.

Figure - Create new column master key with SSMS
4.2 Validate Generation of the CMK

The presence of the asymmetric CMK in the YubiHSM 2 can also be validated by running the following sequence of YubiHSM-Shell commands in a command prompt:

```bash
$yubihs> connect
$yubihs> session open <slot-ID> <password>
$yubihs> list objects <session-ID>
$yubihs> get objectinfo <session-ID> <key-ID> asymmetric-key
```

Example output from the YubiHSM-Shell commands is shown in the screenshot below.
4.3 Generate the CEK

The next step is to generate the CEK for a database.

**Step 1** Generate the CEK.

Launch SSMS and expand **Databases > Database-Name > Security > Column Encryption Key**. In our example, expand the tree **Databases > Sales3 > Security > Column Encryption Key**, which is illustrated in the screenshot below.

**Figure - Column Encryption Keys in SSMS**

**Step 2** Right-click **Column Encryption Keys**, and select **New Column Encryption Key**.

**Step 3** In the **New Encryption Master Key** window, enter the following values:

a. In the **Name** text field, enter an appropriate name for the CEK, for example **CEK-YubiHSM-SSMS**.

**Figure - New Column Master Key listed in YubiHSM**
b. In the Column master key drop-down list, select the CMK that was generated on the YubiHSM, for example CMK-YubiHSM-SSMS.

![Create new Column Encryption Key with SSMS](image)

**Figure - Create new Column Encryption Key with SSMS**

**Step 4** Generate and verify the CEK.

Press OK. To verify the success of the operation, check to see whether the CEK is listed under Always Encrypted Keys in SSMS.
4.3. Generate the CEK
Instead of using SSMS to generate the CMK and CEK (as described in the foregoing section, *Use SSMS to Generate the CMK and CEK*), another option is to use PowerShell to generate the CMK and CEK. Microsoft has published a PowerShell script that can be used to generate the CMK in an HSM. The following instructions list the activities in the script, then describe how to modify that PowerShell script to generate the CMK in the YubiHSM 2 by calling its KSP.

### 5.1 Create a CMK in the YubiHSM 2 with CNG Provider (KSP)

```powershell
$cngProviderName = "YubiHSM Key Storage Provider"
$cngAlgorithmName = "RSA"
$cngKeySize = 2048 # Recommended key size for column master keys
$cngKeyName = "AlwaysEncryptedKey-PS" # Name identifying your key in the KSP
$cngProvider = New-Object
    System.Security.Cryptography.CngProvider($cngProviderName)
$cngKeyParams = New-Object
    $cngKeyParams.provider = $cngProvider
    $cngKeyParams.KeyCreationOptions =
            OverwriteExistingKey
$keySizeProperty = New-Object
        [System.BitConverter]::GetBytes($cngKeySize),
        [System.Security.Cryptography.CngPropertyOptions]::None);
$cngKeyParams.Parameters.Add($keySizeProperty)
$cngAlgorithm = New-Object
$cngKey = [System.Security.Cryptography.CngKey].Create($cngAlgorithm,
    $cngKeyName, $cngKeyParams)
```
5.2 Import SQL Server Module

```
Import-Module "SqlServer"
```

5.3 Connect to your Database

```
$servername = "<server name>"
$databaseName = "<database name>"
$connStr = "Server = " + $servername + "; Database = " + $databaseName + "; Integrated Security = True"
$connection.ConnectionString = $connStr
$connection.Connect()
$database = $server.Databases[$databaseName]
```

5.4 Create SQL CMK Settings Object for your CMK

```
$cmkSettings = New-SqlCngColumnMasterKeySettings -CngProviderName $cngProviderName -KeyName $cngKeyName
```

5.5 Create CMK Metadata in Database

```
$cmkName = "CMK-YubiHSM-PS"
New-SqlColumnMasterKey -Name $cmkName -InputObject $database -ColumnMasterKeySettings $cmkSettings -Verbose
```

5.6 Generate CEK, Encrypt with CMK, and Create CEK Metadata in Database

```
$cekName = "CEK-YubiHSM-PS"
New-SqlColumnEncryptionKey -Name $cekName -InputObject $database -ColumnMasterKeyName $cmkName -Verbose
```
5.7 Customize the Script

Step 1 To customize this script, change the placeholders server name and database name to the actual values of the Microsoft SQL Server name and the database used. For the test database used in this example, the database name is set to Sales3, while the server name should be set to the name of your Windows server.

Step 2 Save the PowerShell script file in a folder with an appropriate name, for example CreateColumnMasterAndEncryptionKeys-YubiHSM.ps1.

Step 3 Launch a command prompt with administrator privileges and enter the PowerShell mode by typing PowerShell. Navigate to the directory where the PowerShell script is located, and execute the script:

```
PS> .\CreateColumnMasterAndEncryptionKeys-YubiHSM.ps1
```

The PowerShell script generates the CMK and the CEK, and displays the output from these operations. An example of such a script is shown in the screenshot below.
Figure - PowerShell script to create Column Master Key and Column Encryption Key
5.8 Validate Generation of the CMK and the CEK

Step 1  After executing the PowerShell script, switch back to SSMS, and expand the objects Databases > Database-Name > Security > Column Master Key and Databases > Database-Name > Security > Column Encryption Key. Right-click each object and select the Refresh option.

The CMK and CEK that were generated by the PowerShell script will appear in SSMS as CMK-YubiHSM-PS and CEK-YubiHSM-PS respectively.

![Figure - Column Master Keys and Column Encryption Keys in SSMS](image)

Step 2  Verify the presence of the asymmetric CMK in the YubiHSM 2 by running the following sequence of YubiHSM-Shell commands in a command prompt.

```
$yubihs> connect
$yubihs> session open <slot-ID> <password>
$yubihs> list objects <session-ID>
$yubihs> get objectinfo <session-ID> <key-ID> asymmetric-key
```

Example output for the YubiHSM-Shell commands is shown in the screenshot below.
Database columns can be encrypted with PowerShell- or SSMS-generated keys.

6.1 Encrypt Database Column with PowerShell-Generated Keys

**Step 1** To encrypt a database column, expand the database’s columns: Databases > Database-Name > Tables > Table-Name > Columns. Our example expands the tree Databases > Sales3 > Tables > dbo.Table_Customers > Columns, as shown in the screenshot below.

![Figure - Expanded columns to be encrypted](image)

**Step 2** Right-click the column to be encrypted and select Encrypt Column…. In our example, right-click the table Name and select Encrypt Column…. The Introduction window in the SSMS Always Encrypted wizard appears:
Step 3  Click Next. The Column Selection window of the Always Encrypted wizard appears:
Step 4 In this example, the CEK that was generated with the PowerShell script is used for encrypting the database column.

a. In the Column Selection window, select the Encryption Key named CEK-YubiHSM-PS.

   The Encryption Type can be set to either Deterministic or Randomized; in this example Deterministic is selected.

b. Click Next, and the Master Key Configuration window in the Always Encrypted wizard appears.
Step 5 In the Master Key Configuration window, click Next, since the master column key in the YubiHSM 2 will be used. The Run Settings window in the Always Encrypted wizard appears.
Step 6 In the Run Settings window, select Proceed to finish now (unless you want to generate a PowerShell script to run later) and click Next. The Summary window in the Always Encrypted wizard appears.
Figure - Always encrypted wizard: Summary

**Step 7** Review the settings in the **Summary** window and click **Finish**. The **Results** window appears:
When the column encryption operation succeeds, the word “Passed” is displayed in the Details column of the relevant row in the Results window.

6.2 Encrypt Database Column with SSMS-generated Keys

To use the CMK and CEK that were generated in Use SSMS to Generate the CMK and CEK follow the instructions above for encrypting a database column with PowerShell-generated keys (Encrypt Database Column with PowerShell-generated Keys), but select a different column (for example, Address) and use the column encryption key CEK-YubiHSM-SSMS and the related column master key CMK-YubiHSM-SSMS.
6.3 Verify Encrypted Database Column

To check that the columns have been encrypted, expand the object **Database > Database-Name** (in our example the database name is **Sales3**). Then select **New Query** in the top menu. Type the SQL query in the example below and click **Execute**.

```
SELECT * FROM Table_Customers;
```

![Figure – Checking the encrypted columns](image-url)
To configure Microsoft SQL Server and SSMS with the basic database settings needed for testing Always Encrypted in conjunction with YubiHSM 2, set SSMS to display the encrypted columns in clear text.

**Step 1** Click the **Connect Object Explorer** icon. The **Connect to Server** window appears. Click **Options**, select the **Always Encrypted** tab, and select **Enable Always Encrypted** (column encryption). To make the changes take effect, click the **Disconnect** icon and then the **Connect** icon.

![Figure – Enable Always Encrypted in SSMS](image-url)
**Step 2** In the main menu, click **Query** and from the drop-down list, select **Query options...** The **Query Options** window appears. Select **Execution > Advanced** and the checkbox for **Enable Parameterization for Always Encrypted**.

![Query Options Window](image)

**Figure – Enable Parameterization for Always Encrypted queries in SSMS**

These are the basic database settings in Microsoft SQL Server and SSMS for testing Always Encrypted in conjunction with YubiHSM.

**Step 3** To verify the settings, expand the object **Database > Database-Name** (in our example the database name is Sales3), and select **New Query** in the top menu again. Re-enter the SQL query in the example below and click **Execute**.

```
SELECT * FROM Table_Customers;
```

When the SSMS settings take effect, the encrypted database columns are decrypted, and the values displayed in clear text as shown in the screenshot below.
Figure – Decrypted values in the database columns
CHAPTER
EIGHT

BACK UP KEY MATERIAL

We strongly recommend making a backup copy of all production objects residing on your production devices, particularly once the CMK has been generated on the YubiHSM 2. If there is a hardware failure of the production device, having a backup ensures that you can resume operations quickly. The backup process will result in two identical YubiHSM 2 devices with the same number of objects, keys, labels, etc.

**Note:** Making specific recommendations for governance of your critical key material is out of scope for this guide. Make sure to design these security procedures to meet the requirements of your organization - and then document them carefully, not least in order to be prepared for audits.

Back up and restore the YubiHSM 2 according to the instructions in Back Up and Restore Key Material in the *YubiHSM 2 for Windows Server Deployment Guide*. 
GETTING HELP AND FURTHER READING

Should you require assistance when using this guide to deploy YubiHSM 2 with Microsoft SQL Server Always Encrypted, start by referencing the product documentation and currently known issues:

- Yubico Developers website
- Yubico Support
- YubiHSM 2 Product Overview
- Known Issues and Limitations

If you need additional help, contact Yubico directly by filling in a ticket at Yubico’s support site.

In addition to the Yubico web sites listed above, Microsoft has published the following articles on SQL Server Always Encrypted:

- Always Encrypted
- Overview of Key Management for Always Encrypted
- Always Encrypted Cryptography
- Configure Always Encrypted using SQL Server Management Studio
- Configure Always Encrypted Keys using PowerShell
The following software terminology as it relates to YubiHSM 2 is used throughout this guide.

**Application** AES key used to authenticate to the device.

**authentication key** Performs operations according to its defined capabilities.

**Audit key** AES authentication key with rights to access audit log.

**Capability** A description of what operations are allowed on or with an object such as a key.

**Column Encryption Key (CEK)** CEKs are content-encryption keys used to encrypt data in a Microsoft SQL Server Always Encrypted database.

**Column Master Key (CMK)** CMKs are key-protecting keys used to encrypt CEKs for a Microsoft SQL Server Always Encrypted database.

**Cryptographic API Next Generation (CNG)** A CNG is Microsoft’s cryptographic architecture, which allows developers to implement applications with features for encryption, electronic signatures, certificate management, etc.

**Default authentication key** Factory-installed AES key used when initializing the device. Possesses all capabilities.

**Delegated capability** An operation that an object is allowed to perform by virtue of receiving those permissions from the authentication key or wrap key that was used to create it. **Domain** A logical “container” for objects that can be used to control access to objects on the device.

**Key custodian** Holder of a wrap key share.

**Key Storage Provider (KSP)** A KSP is a DLL that is loaded by Microsoft CNG. KSPs can be used to create, delete, export, import, open and store keys.

**Object ID** Object IDs are unique identifiers for any kind of object stored on YubiHSM2. An ID can range between 1 and 65535; however, the device can hold a maximum of 256 unique objects.

**M of n** Scheme where wrap key is split into a total number of shares (n) held by key custodians, where at least a minimum number of shares (m) (sometimes this is also called ‘quorum’) is needed to use the key.

**SQL Server Management Studio (SSMS)** SQL Server Management Studio (SSMS) is a software application that is used for configuring, managing, and administering all components within Microsoft SQL Server.

**Wrap key** AES key used to protect key material when exporting to file from device and when importing from file to device. Key material exported under wrap will be encrypted and can only be decrypted using the wrap key.
CHAPTER

ELEVEN

COPYRIGHT

© 2022 Yubico AB. All rights reserved.

Trademarks

Yubico and YubiKey are registered trademarks of Yubico AB. All other trademarks are the property of their respective owners.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design, and manufacturing. Yubico shall have no liability for any error or damages of any kind resulting from the use of this document.

The Yubico Software referenced in this document is licensed to you under the terms and conditions accompanying the software or as otherwise agreed between you or the company that you are representing.

Contact Information

Yubico Inc.
530 Lytton Street
Suite 301
Palo Alto, CA 94301
USA

Click the links to:
    • Submit a support request
    • Send a Contact Me request
    • See additional contact options for getting touch with us

Document Updated

2022-05-05 00:39:08 UTC