Microsoft SQL Server 2019
Always Encrypted
nShield® HSM Integration Guide

01 Apr 2022

© 2022 Entrust Corporation. All rights reserved.
# Contents

1. Introduction ......................................................................................................................... 3
   1.1. Product configurations .................................................................................................... 3
   1.2. Supported nShield features ............................................................................................ 3
   1.3. Supported nShield hardware and software versions ....................................................... 4
   1.4. Role separation ................................................................................................................ 4
   1.5. Using multiple on-premises client servers ....................................................................... 5
   1.6. Always Encrypted and TDE ............................................................................................ 5

2. Install and configure ............................................................................................................... 6
   2.1. Install the Security World software and create a Security World .................................... 6
   2.2. Install and register the CNG provider ............................................................................. 7
   2.3. Install and configure SqlServer PowerShell module ...................................................... 15

3. Generate the encryption keys ............................................................................................... 17
   3.1. Generate the Always Encrypted Column Master Key (CMK) protected by the nShield HMS .................................................................................................................. 17
   3.2. Generate My Column Master Key (MyCMK) and My Column Encryption Key (MyCEK) with SSMS ............................................................................................................ 21
   3.3. Generate My Column Master Key (MyCMK) and My Column Encryption Key (MyCEK) with PowerShell ................................................................................................. 33

4. Encrypt or decrypt a column with SSMS ............................................................................. 36
   4.1. Encrypt a column ............................................................................................................. 36
   4.2. View an encrypted column .............................................................................................. 46
   4.3. Remove column encryption ............................................................................................ 53

5. Encrypt or decrypt a column with PowerShell .................................................................... 64
   5.1. Encrypt a column ............................................................................................................. 64
   5.2. View an encrypted column .............................................................................................. 64
   5.3. Remove column encryption ............................................................................................ 64

6. Supported PowerShell SqlServer cmdlets .......................................................................... 66
1. Introduction

Always Encrypted is a feature in Windows SQL Server 2019 designed to protect sensitive data both at rest and in flight between an on-premises client application server and Azure or SQL Server database(s).

Data protected by Always Encrypted remains in an encrypted state until it has reached the on-premises client application server. This effectively mitigates man-in-the-middle attacks and provides assurances against unauthorized activity from rogue DBAs or admins with access to Azure or SQL server Databases.

The nShield HSM secures the key used to protect the Column Master Key, stored in an encrypted state on the on-premises client application server.

1.1. Product configurations

Entrust successfully tested nShield HSM integration with Windows SQL Server 2019 and the Always Encrypted feature in the following configurations:

1.1.1. Remote server

<table>
<thead>
<tr>
<th>Product</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL Server</td>
<td>Microsoft SQL Server 2019</td>
</tr>
<tr>
<td>Base OS</td>
<td>Windows Server 2019 Datacenter</td>
</tr>
</tbody>
</table>

1.1.2. On-premises client

<table>
<thead>
<tr>
<th>Product</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL Server GUI</td>
<td>Microsoft SQL Server Management Studio V18.8</td>
</tr>
<tr>
<td>Base OS</td>
<td>Windows 10 Enterprise</td>
</tr>
</tbody>
</table>

1.2. Supported nShield features

Entrust successfully tested nShield HSM integration with the following features:
1.3. Supported nShield hardware and software versions

Entrust successfully tested with the following nShield hardware and software versions:

### 1.3.1. Connect XC

<table>
<thead>
<tr>
<th>Security World Software</th>
<th>Firmware</th>
<th>Image</th>
<th>OCS</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.80.4</td>
<td>FIPS 12.50.11</td>
<td>12.60.10</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>12.80.4</td>
<td>CC 12.50.7</td>
<td>12.50.7</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

### 1.3.2. Connect +

<table>
<thead>
<tr>
<th>Security World Software</th>
<th>Firmware</th>
<th>Image</th>
<th>OCS</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.80.4</td>
<td>FIPS 12.50.8</td>
<td>12.60.10</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>12.40 Compatibility Package</td>
<td>CC 2.55.4</td>
<td>12.45.1</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

### 1.4. Role separation

The generation of keys, and the application of these keys for encryption or decryption are separate processes. The processes can be assigned to users with various access permissions, or Duty Roles. The table below shows the processes and duty roles with reference to the Security Administrator and the Database Administrator.

<table>
<thead>
<tr>
<th>Process</th>
<th>Duty Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generating the Column Master Key (CMK) and Column Encryption Key (CEK)</td>
<td>Security Administrator</td>
</tr>
</tbody>
</table>
Applying the CMK and CEK in the database

Duty Role: Database Administrator

Four database permissions are required for Always Encrypted.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTER ANY COLUMN MASTER KEY</td>
<td>Required to generate and delete a column master key</td>
</tr>
<tr>
<td>ALTER ANY COLUMN ENCRYPTION KEY</td>
<td>Required to generate and delete a column encryption key</td>
</tr>
<tr>
<td>VIEW ANY COLUMN MASTER KEY</td>
<td>Required to access and read the metadata of the column master keys to manage keys or query encrypted columns</td>
</tr>
<tr>
<td>VIEW ANY COLUMN ENCRYPTION KEY</td>
<td>Required to access and read the metadata of the column encryption key to manage keys or query encrypted columns</td>
</tr>
</tbody>
</table>

1.5. Using multiple on-premises client servers

Each client server wanting access to the contents of data encrypted with a given CEK must have access to an HSM in the same Security World and have a copy of the CMK key token stored on its local drive.

1.6. Always Encrypted and TDE

The same Security World can be used for Always Encrypted and TDE.
2. Install and configure

This installation must be performed on the on-premises client computer.

The nShield Security World software will be installed on this computer. This computer will also be made a client of the HSM.

2.1. Install the Security World software and create a Security World

1. Install the Security World software. For instructions, see the *Installation Guide* and the *User Guide* for the HSM.

2. Add the Security World utilities path `C:\Program Files\nCipher\nfast\bin` to the Windows system path.

3. Open the firewall port 9004 for the HSM connections.

4. Install the nShield Connect HSM locally, remotely, or remotely via the serial console. See the following nShield Support articles, and the *Installation Guide* for the HSM:
   - [https://nshieldsupport.entrust.com/hc/en-us/articles/360021378272-How-To-Locally-Set-up-a-new-or-replacement-nShield-Connect](https://nshieldsupport.entrust.com/hc/en-us/articles/360021378272-How-To-Locally-Set-up-a-new-or-replacement-nShield-Connect)

5. Open a command window and run the following to confirm that the HSM is operational.

   ```
   C:\Users\dbuser>enquiry
   Server:
   enquiry reply flags: none
   enquiry reply level: Six
   serial number: 530E-02E0-D947 7724-8509-81E3 09AF-0BE9-53AA 9E10-03E0-D947
   mode: operational
   ...
   Module #1:
   enquiry reply flags: none
   enquiry reply level: Six
   serial number: 530E-02E0-D947
   mode: operational
   ...
   ```

6. Create your Security World if one does not already exist, or copy an existing one. Follow your organization’s security policy for this.

7. Confirm that the Security World is usable.
2.2. Install and register the CNG provider

1. Open a command window as administrator and type the following to put the HSM in **pre-initialization** mode. This operation takes about a minute to complete.

```
C:\Windows\system32>enquiry -m 1
Module #1:
enquiry reply flags  none
enquiry reply level  Six
serial number      530E-02E0-0947
mode               operational
...
```

2. Select the **Start** button to access all applications. Look for the recently installed nShield utilities.

3. Double-click the CNG configuration wizard and run it as Administrator.

```
C:\Windows\system32>nopclearfail -I -m 1
Module 1, command ClearUnitEx: OK
```

```
C:\Windows\system32>enquiry -m 1
Module #1:
enquiry reply flags  none
enquiry reply level  Six
serial number      530E-02E0-0947
mode               pre-initialization
...
```

4. Select **Next** on the **CNG Install** welcome screen.
5. Select **Next** on the **Enable HSM Pool Mode** screen. Leave the **Enable HSM Pool Mode for CNG Providers** check box un-checked.

6. At the **Security World** screen, select:
   - **Use the existing security world** if you already have a Security World that you intend to use for Always Encrypted. The corresponding `world` and `module_xxxx-`
xxxx-xxxx files must be present in the %NFAST_KMDATA%\local folder. Be prepared to present the quorum of Administrator cards.

- **Create a new Security World** if you do not currently have a Security World or would like to create a new Security World.

  For the purposes of this integration guide we have chosen to use an existing Security World. For instructions on how to create and configure a new Security World, see the *Installation Guide* and *User Guide* for your HSM.

Select Next.

7. The **Set Module States** pop-up shows the available HSM(s). Select the desired HSM. The state of the selected HSM should be *(pre-)*initialisation. Select Next.
8. At the Module Programming Options screen, clear **Enable this module as a remote target** and select **Next**. It will take about a minute before the screen changes.

Please be aware that this is not to be confused with the nShield Remote Administration utility.
9. Insert the first Administrator Card in the HSM, enter the passphrase and select **Next**. Repeat this step for the other Administrator Cards as required.

Loading or creating the Security World takes about a minute.

10. Return the HSM to **Operational** mode. This operation takes about a minute to complete.

The module state will change to **Usable**.

Select Next.

12. Enter the OCS name, K of N values, and check Persistent and Usable remotely as show.
13. Insert a blank Operator Card in the HSM. On the **Insert Next Card** screen enter a name to for the OCS card and passphrase.

Select **Next**.
14. Select **Next** and **Finish**. The nShield CNG providers will now be installed and the key Storage Provider will be registered.

15. Open a command window as administrator and type the following to confirm the KSP has been successfully registered. Look for **nCipher Security World Key Storage Provider**.
C:\Windows\system32>cnglist.exe --list-providers
Microsoft Key Protection Provider
Microsoft Passport Key Storage Provider
Microsoft Platform Crypto Provider
Microsoft Primitive Provider
Microsoft Smart Card Key Storage Provider
Microsoft Software Key Storage Provider
Microsoft SSL Protocol Provider
Windows Client Key Protection Provider
nCipher Primitive Provider
nCipher Security World Key Storage Provider

16. Check the registry in CNGRegistry:

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Cryptography\Providers\nCipherSecurityWorldKeyStorageProvider

![Registry Editor](image)

2.3. Install and configure.SqlServer PowerShell module

1. Open a PowerShell session as Administrator and run:

   Install-PackageProvider Nuget -force -verbose

2. Update PowerShellGet:

   Install-Module -Name PowerShellGet -force -verbose

3. Download and install the SqlServer module to configure Always Encrypted using Power Shell:

   Install-Module -Name SqlServer -force -verbose -AllowClobber
The `-AllowClobber` parameter allows you to import the specified commands if it exists in the current session.

4. Once installed (if you are using PowerShell ISE refresh the Commands pane if you are using PowerShell open a new session), confirm the install by running:

```powershell
Get-Module -list -Name SqlServer
```

5. You should see something similar to the output below:

```plaintext
Directory: C:\Program Files\WindowsPowerShell\Modules

ModuleType Version    Name          ExportedCommands
---------- -------    ----          ----------------
Manifest   21.0.17152 SqlServer     {Add-SqlColumnEncryptionKeyValue, Complete-SqlColumnMasterKeyRotatio...
```
3. Generate the encryption keys

3.1. Generate the Always Encrypted Column Master Key (CMK) protected by the nShield HMS

1. Launch PowerShell on the on-premises client computer as Administrator, and run the Generate_CMK.ps1 script.

```powershell
$cngProviderName = "nCipher Security World Key Storage Provider"
$cngAlgorithmName = "RSA"
$cngKeySize = 2048
$cngKeyName = "AECMK"
$cngKeyParams.provider = $cngProvider
$cngKeyParams.KeyCreationOptions = {[System.Security.Cryptography.CngKeyCreationOptions]::OverwriteExistingKey}
$cngKeyParams.Parameters.Add($keySizeProperty)
$cngKey = [System.Security.Cryptography.CngKey]::Create($cngAlgorithm, $cngKeyName, $cngKeyParams)

The command line is

```powershell
> PowerShell -ExecutionPolicy Bypass -File Generate_CMK.ps1
```

a. The following pop-window should appear. Select Next.
b. Select the **Operator Card Set Protection**. Insert the OCS card in the HSM and select **Next**.

c. Select **Next**.
d. Select the HSM and select **Finish**.

```
Select token to protect the key with.
```

```
Choose modules you wish to load the key onto.
```

d. Select the HSM and select **Finish**.

e. Enter the OCS passphrase and select **Next**.
A 2048-bit RSA key pair, called AECMK, has been generated. The key is encrypted in the HSM, and then pushed to the requesting On-Premise Client server, where it is stored as an Application Key Token in the %NFAST_KMDATA%\local folder (:\ProgramData\nCipher\Key Management Data\local).

2. Verify the new key as follows on a command window.
3. Display the information about the key by copy-pasting the key name above as follows.

```plaintext
C:\Users\dbuser>nfkminfo -k caping s-1-5-21-2556418611-2173580918-1658130183-1001--7b7eb65c095c556e5da059480e6ca2ed512dacc1

Key AppName caping Ident s-1-5-21-2556418611-2173580918-1658130183-1001--7b7eb65c095c556e5da059480e6ca2ed512dacc1
BlobKA length         1128
BlobPubKA length      484
BlobRecoveryKA length 1496
name                  "AECMK"
hash                  76071834044810539e7354f4f68cc2caae61a448da
recovery              Enabled
protection            CardSet
other flags           PublicKey !SEEAppKey !NVMemBlob +0x0
cardset               0e8d19801b25d774c3b2bab5a64ae7c20a5255d
gentime               2021-03-30 19:18:14
SEE integrity key     NONE

BlobKA
format                6 Token
other flags           0x0
hkm                   2a2e0b22ada72673473511d91304ed2f76e197
hkt                   0e8d19801b25d774c3b2bab5a64ae7c20a5255d
hkr                   none

BlobRecoveryKA
format                9 UserKey
other flags           0x0
hkm                   none
hkt                   none
hkr                   fc4c8d1a6e88c08dd35912d0aeacbf47ff1e0c2a

BlobPubKA
format                5 Module
other flags           0x0
hkm                   c2be99fe1c77f175d40e2f2df8fff0c6e960cb
hkt                   none
hkr                   none

Extra entry #1
typecode              0x10000 65536
length                60
Not a blob
```

3.2. Generate My Column Master Key (MyCMK) and My Column Encryption Key (MyCEK) with SSMS

This key will encrypt all subsequent Column Encryption keys (CEKs) in your database.

1. Launch **Microsoft SQL Server Management Studio** on the on-premises client computer.

2. As the **dbuser** user, connect to the database on the SQL server on the hosting site.
3. Using **Object Explorer**, select the **Security** directory under the desired Database. Select **Always Encrypted Keys** to expand it, then select **New Column Master Key**.
4. Enter the following information on the **Column Master Keys** pop-up window, then select **Next**
   a. Enter a name, for example **MyCMK**.
   b. Select **Key Storage Provider (CNG)** from the **Key store** drop-down list. This will then present the option to **Select a provider**.
   c. Select **nCipher Security World Key Storage Provider** from the drop-down list.
   d. The **AECMK** key created in an earlier step appears in **Name**. Select **OK** to create a new key using the nShield HSM and CNG KSP.
5. Notice the newly created **MyCMK** in the database **Security\Always Encrypted Keys\Column Master Keys**.
6. Using **Object Explorer**, select the **Security** directory under the desired Database. Select **Always Encrypted Keys** to expand it, then select **New Column MEncryption Key**.
7. Enter **Name** and select **OK**.
8. Present the OCS and select **Next**.
9. Select the HSM and select **Finish**.
10. Enter the passphrase and select Next.
11. Select **Finish** after the OCS card reading completes.
12. Notice the newly created **MyCEK** in the database *Security\Always Encrypted Keys\Column Encryption Keys*. 
3.3. Generate My Column Master Key (MyCMK) and My Column Encryption Key (MyCEK) with PowerShell

1. Delete MyCEK and MyCMK created above by right-clicking each key and selecting Delete.

2. Launch PowerShell on the on-premises client computer and run the Generate_MyCMK_and_MyCEK.ps1 script.
# Import the SqlServer module.
Import-Module SqlServer

# Connect to database.
$ConnectionString = "Data Source=<DB_Server_IP>,49170;Initial Catalog=TestDatabase;User
ID=dbuser;Password=<dbuser_Password>;MultipleActiveResultSets=False;Connect
Timeout=30;Encrypt=True;TrustServerCertificate=True;Packet Size=4096;Application Name="Microsoft SQL Server Management Studio"
$Database = Get-SqlDatabase -ConnectionString $ConnectionString

# Create a SqlColumnMasterKeySettings object for your column master key.
$cmkSettings = New-SqlCngColumnMasterKeySettings -CngProviderName "nCipher Security World Key Storage Provider" -KeyName "AECMK"

# Create column master key metadata in the database.
New-SqlColumnMasterKey -Name "MyCMK" -InputObject $Database -ColumnMasterKeySettings $cmkSettings

# Generate a column encryption key, encrypt it with the column master key and create column encryption key metadata in the database.
New-SqlColumnEncryptionKey -Name "MyCEK" -InputObject $Database -ColumnMasterKey "MyCMK"

The command line is

> PowerShell -ExecutionPolicy Bypass -File Generate_MyCMK_and_MyCEK.ps1

Name
-----
MyCMK
MyCEK

3. Present the OCS, select the HSM, and enter the passphrase.
4. Notice the newly created **MyCMK** in the database **Security\Always Encrypted Keys\Column Master Keys**.
4. Encrypt or decrypt a column with SSMS

4.1. Encrypt a column

1. Launch **Microsoft SQL Server Management Studio** on the on-premises client. Connect with the dbuser account to the database on the SQL server.

2. Right-click the database, **TestDatabase**, and select **Tasks > Encrypt Columns**.

4. Select the column and encryption type on the **Column Selection** screen and select **Next**.
5. Select **MyCMK** on the **Master key Configuration** window. Select **Next**.
6. Select **Proceed to finish now** radio button and select **Next**.
7. Verify the configuration choices on the **Summary** screen. Select **Next**.
8. Present the OCS that is protecting the CMK and select **Finish**.
9. Select the HSM and select **Next**.
10. Enter the passphrase, and select **Next**.
11. Select **Finish**.
12. Select Close.
The column has been encrypted in the SQL server, but it shows as clear text on the Microsoft SQL Server Management Studio GUI on the on-premises client. This is because Always Encrypted is performing the decryption at the on-premises client site.

4.2. View an encrypted column

Reconnect to the SQL server with Always Encrypted disabled to view the encrypted data stored in the SQL server.

1. Connect to the SQL server from the on-premises client, but with the Enable Always Encrypted unchecked.
2. Right-click **dbo.Table**, and select **Select Top 1000 Rows**. The column that was chosen for encryption now appears as ciphertext, that is, as an encrypted value.
3. Reconnect to the SQL server from the on-premises client, but with the Enable Always Encrypted checked. Be prepared to provide the OCS. Select Next.
4. Select the HSM. Select Finish.
5. Enter the passphrase. Select **Next**.
6. Select **Finish**.
7. Right-click `dbo.Table`, and select **Select Top 1000 Rows**. The column that was chosen for encryption is now being decrypted by **Always Encrypted** with the key protected by the nCipher HSM.

8. Select **Finish**.
4.3. Remove column encryption

1. Right-click the required database and in the Tasks menu and select Encrypt Columns.
2. Select **Next** on the Introduction screen.
3. Select **Plaintext** from the drop down list in the **Encryption Type** and select **Next**.
4. Select **Next** on the **Master Key Configuration** window.
5. Select the **Proceed to finish now** radio button and select **Next**.
6. Select **Finish** on the **Summary** window.
7. Present the OCS and select **Next**.
8. Select the HSM and select **Finish**.
9. Enter the passphrase and select **Next**.
10. Select **Finish** upon **Card reading complete**.
The column has been decrypted in the SQL server. To view the plain text data stored SQL server, reconnect to the server with Always Encrypted disabled, see View an encrypted column.
5. Encrypt or decrypt a column with PowerShell

5.1. Encrypt a column

1. Launch PowerShell on the on-premises client computer and run the following script named Encrypt_Column_Named_Password.ps1.

```powershell
# Import the SqlServer module.
Import-Module SqlServer

# Connect to database.
$ConnectionString = "Data Source=<DB_Server_IP>,49170;Initial Catalog=TestDatabase;User ID=dbuser;Password=<dbuser_Password>;MultipleActiveResultSets=False;Connect Timeout=30;Encrypt=True;TrustServerCertificate=True;Packet Size=4096;Application Name="Microsoft SQL Server Management Studio"
$Database = Get-SqlDatabase -ConnectionString $ConnectionString

# Change encryption schema.
$encryptionChanges = @()

# Add changes for table [dbo].[TestTable]
$encryptionChanges += New-SqlColumnEncryptionSettings -ColumnName dbo.TestTable.Password -EncryptionType Randomized -EncryptionKey "MyCEK"
Set-SqlColumnEncryption -ColumnEncryptionSettings $encryptionChanges -InputObject $Database
```

The command line is

```
> PowerShell -ExecutionPolicy Bypass -File Encrypt_Column_Named_Password.ps1
```

2. Present the OCS, select the HSM, and enter the passphrase.

The column has been encrypted in the SQL server, but it shows as clear text on the Microsoft SQL Server Management Studio screen on the on-premises client. This is because Always Encrypted is performing the decryption at the on-premises client site.

5.2. View an encrypted column

Reconnect to the SQL server with Always Encrypted disabled to view the encrypted data stored in the SQL server. See View an encrypted column.

5.3. Remove column encryption

1. Launch PowerShell on the on-premises client computer and run the following script named Decrypt_Column_Named_Password.ps1.
## Import the SqlServer module.
```powershell
Import-Module SqlServer
```

## Connect to database.
```powershell
$ConnectionString = "Data Source=<DB_Server_IP>,49170;Initial Catalog=TestDatabase;User ID=dbuser;Password=<dbuser_Password>;MultipleActiveResultSets=False;Connect Timeout=30;Encrypt=True;TrustServerCertificate=True;Packet Size=4096;Application Name="Microsoft SQL Server Management Studio"
$Database = Get-SqlDatabase -ConnectionString $ConnectionString
```

## Change encryption schema
```powershell
$encryptionChanges = @()
$encryptionChanges += New-SqlColumnEncryptionSettings -ColumnName dbo.TestTable.Password -EncryptionType Plaintext
Set-SqlColumnEncryption -ColumnEncryptionSettings $encryptionChanges -InputObject $Database
```

The command line is
```powershell
> PowerShell -ExecutionPolicy Bypass -File Decrypt_Column_Named_Password.ps1
```

2. Present the OCS, select the HSM, and enter the passphrase.

The column has been decrypted in the SQL server. To view the plain text data stored SQL server, reconnect to the server with Always Encrypted disabled, see View an encrypted column.
### 6. Supported PowerShell SqlServer cmdlets

<table>
<thead>
<tr>
<th>PowerShell cmdlet</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-SqlColumnEncryptionKeyValue</td>
<td>Adds a new encrypted value for an existing column encryption key object in the database.</td>
</tr>
<tr>
<td>Complete-SqlColumnMasterKeyRotation</td>
<td>Completes the rotation of a column master key.</td>
</tr>
<tr>
<td>Get-SqlColumnEncryptionKey</td>
<td>Returns all column encryption key objects defined in the database, or returns one column encryption key object with the specified name.</td>
</tr>
<tr>
<td>Get-SqlColumnMasterKey</td>
<td>Returns the column master key objects defined in the database, or returns one column master key object with the specified name.</td>
</tr>
<tr>
<td>Invoke-SqlColumnMasterKeyRotation</td>
<td>Initiates the rotation of a column master key.</td>
</tr>
<tr>
<td>New-SqlAzureKeyVaultColumnMasterKeySettings</td>
<td>Creates a SqlColumnMasterKeySettings object describing an asymmetric key stored in Azure Key Vault.</td>
</tr>
<tr>
<td>New-SqlCngColumnMasterKeySettings</td>
<td>Creates a SqlColumnMasterKeySettings object describing an asymmetric key stored in a key store supporting the Cryptography Next Generation (CNG) API.</td>
</tr>
<tr>
<td>New-SqlColumnEncryptionKey</td>
<td>Creates a new column encryption key object in the database.</td>
</tr>
<tr>
<td>New-SqlColumnEncryptionKeyEncryptedValue</td>
<td>Produces an encrypted value of a column encryption key.</td>
</tr>
<tr>
<td>New-SqlColumnEncryptionSettings</td>
<td>Creates a new SqlColumnEncryptionSettings object that encapsulates information about a single column's encryption, including CEK and encryption type.</td>
</tr>
<tr>
<td>PowerShell cmdlet</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>New-SqlColumnMasterKey</td>
<td>Creates a new column master key object in the database.</td>
</tr>
<tr>
<td>New-SqlCspColumnMasterKeySettings</td>
<td>Creates a SqlColumnMasterKeySettings object describing an asymmetric key stored in a key store with a Cryptography Service Provider (CSP) supporting Cryptography API (CAPI).</td>
</tr>
<tr>
<td>Remove-SqlColumnEncryptionKey</td>
<td>Removes the column encryption key object from the database.</td>
</tr>
<tr>
<td>Remove-SqlColumnEncryptionKeyValue</td>
<td>Removes an encrypted value from an existing column encryption key object in the database.</td>
</tr>
<tr>
<td>Remove-SqlColumnMasterKey</td>
<td>Removes the column master key object from the database.</td>
</tr>
<tr>
<td>Set-SqlColumnEncryption</td>
<td>Encrypts, decrypts or re-encrypts specified columns in the database.</td>
</tr>
</tbody>
</table>

The full list of cmdlets and additions to the SqlServer module can be found at https://docs.microsoft.com/en-us/powershell/module/sqlserver/?view=sqlserver-ps. 