Microsoft SQL Server TDE and Entrust KeyControl

Integration Guide

2024-05-24

Member of
Microsoft Intelligent
Security Association

Microsoft Security

Table of Contents

1. Introduction . . . 1
    1.1. Product configurations . . . 1
2. Procedures . . . 2
    2.1. Prerequisites . . . 2
    2.2. Workflow overview . . . 2
    2.3. Configure KeyControl Vault . . . 2
    2.4. Install the Policy Agent client on the SQL Server machine . . . 3
    2.5. Register with the KeyControl server . . . 3
    2.6. Create a TDE database keyset to hold keys . . . 4
    2.7. Create the database connector . . . 5
    2.8. Create a master key on KeyControl . . . 6
    2.9. Create the TDE key within Microsoft SQL Server . . . 6
    2.10. Test the database encryption . . . 7
    2.11. Rotate the key manually in KeyControl . . . 8
    2.12. Shut down encryption on the database and remove credentials . . . 9
3. Additional resources and related products . . . 11
    3.1. KeyControl . . . 11
    3.2. Entrust products . . . 11
    3.3. nShield product documentation . . . 11

Chapter 1. Introduction

This document describes the procedure to integrate Entrust KeyControl and Microsoft SQL Server TDE for establishing KeyControl as an EKM provider for SQL Server.

1.1. Product configurations

Entrust has successfully tested Microsoft SQL Server TDE integration with KeyControl in the following configurations:

Product	Version
KeyControl	10.2
Microsoft SQL Server	2022
SQL Server Management Studio	20.1.10.0
Operating System	Windows Server 2022

Chapter 2. Procedures

2.1. Prerequisites

- Entrust KeyControl nodes, the Vault Management node with a **Database Vault**, have been deployed and configured. For details, see the *Entrust KeyControl 10.2 Install & Upgrade Guide* in the HyTrust Online documentation: https://docs.hytrust.com/KeyControlVault/10.2/Online/Content/OLH-Files/Help-content-map-all-books.html.
- Microsoft SQL Server 2022 (Developer Edition) and SQL Server Management Tools have been installed.

  Microsoft SQL Server Express Edition does not support EKM so it is not compatible with this integration.

2.2. Workflow overview

Throughout this guide, queries will be run from the command line. Examples queries will show how to accomplish the integration of KeyControl and SQL Server. Edit these queries to meet your requirements.

The workflow to accomplish the integration of KeyControl and SQL Server is:

1. Configure KeyControl Vault.
2. Install the Policy Agent client on the SQL Server machine
3. Register with the KeyControl server
4. Create a TDE database keyset to hold keys
5. Create the database connector
6. Create a master key on KeyControl
7. Create the TDE key within Microsoft SQL Server
8. Test the database encryption

Operations for later:

- Rotate the key manually in KeyControl
- Shut down encryption on the database and remove credentials

2.3. Configure KeyControl Vault

1. Sign in to your Key Control Vault Management node.
2. Select **Create Vault**.
3. Enter the Vault Type as **Database**.
4. Enter the Vault **Name**.
5. Optional, enter a description.
6. Enter **Admin Name**.
7. Enter **Admin Email**.
8. Select **Create Vault**.
9. Sign in to your vault first with your Admin email and temporary password.
10. Change the Admin password.
11. Sign in to your Database Vault with your new password.

2.4. Install the Policy Agent client on the SQL Server machine

1. Sign in to the KeyControl Vault Management WebGUI using an account with Cloud Admin privileges.
2. Select the **Workloads** tab.
3. Select **Actions > Download Policy Agent**.
4. Select **Download** next to `hcs-client-agent-xyz`.
5. Copy the download to the machine containing Microsoft SQL Server and run it as `administrator`.
6. Navigate through the install with the default options.
7. Select **Reboot now**.

2.5. Register with the KeyControl server

1. Sign in to the KeyControl Vault Management WebGUI.
2. Select the **Workloads** tab.
3. Select **Actions > Create Cloud VM Set**.
4. Enter a **Name** for the Cloud VM Set.
5. Select **Create**.
6. Sign in to the machine containing SQL Server.
7. Open **Entrust DataControl** through the Windows Start menu.

8. Select **Register**.
9. Enter the **IP address** of the KeyControl Server.
10. Enter the **Username** of the KeyControl Cloud Admin User.
11. Enter the **Password** for the KeyControl Cloud Admin User.
12. Enter the name of the previously created **Cloud VM Set**.
13. Enter the **Vault ID** (you can find it from the user profile).
14. Select **Register**.
15. Open an Admin command prompt window.
16. Enable TDE on the server:

```
C:\Users\Administrator.INTEROP>hcl tde status
TDE is  not enabled on this VM

C:\Users\Administrator.INTEROP>hcl tde enable
Enabling tde will change permissions of some Files.
Do you want to proceed? (y/n) y

C:\Users\Administrator.INTEROP>hcl tde status
TDE is enabled on this VM

C:\Users\Administrator.INTEROP>hcl status

Summary
--------------------------------------------------------------------------------
KeyControl: xx.xxx.xxx.xxx:443
KeyControl list: xx.xxx.xxx.xxx:443
Vault ID: 5bd41334-xxxx-xxxx-xxxx-93b369278012
Status: Connected
Last heartbeat: Tue May 21 11:22:25 2024 (successful)
AES_NI: enabled
Certificate Expiration: May 21 14:26:46 2025 GMT

Device details
--------------------------------------------------------------------------------
Drive      Disk Part Cipher      Status    GUID
--------------------------------------------------------------------------------
C:         0    3    none        Avail-Sys N/A
```

The VM will now appear under **WORKLOADS > VMs** in KeyControl.

2.6. Create a TDE database keyset to hold keys

1. Sign in to the KeyControl WebGUI using an account with Cloud Admin privileges.
2. Select the **CLOUDKEYS** tab.
3. Select **Actions > Create Key Set > TDE**.
4. Enter a name for the keyset.
5. For **Admin Group**, select **Cloud Admin Group**.

6. For **Database Type**, select **Microsoft SQL Server**.
7. Select **Continue**.
8. Leave **Enable HSM** unchecked.
9. Select **Continue**.
10. Select an option for the **Rotation Schedule**.
11. Select **Apply**. The keyset is created.
12. Select **Close**.

2.7. Create the database connector

1. Select the previously created keyset from the list.
2. Then select the **Database Connectors** tab.
3. Select **Create Connector**.
4. Select your machine from the list.
5. Enter a **Connector Name**.
6. Select an option for the **Expiration**.
7. Select **Create**.

To generate and apply an Access Token:

1. Check the box next to the Database Connector in the list and then select **Actions > Generate Access Token**.
2. Select **Generate Token**. You will need to copy the **Identity** and **Secret** and then paste them into the following queries that will be run.
3. Sign in to the machine containing Microsoft SQL Server and open Microsoft SQL Server Management Studio.
4. Connect to Database Server.
5. Enable and load the EKM provider:

```
USE master
go

-- Enable EKM provider
sp_configure 'show advanced options', 1 ;
GO
reconfigure;
go

sp_configure 'EKM provider enabled', 1 ;
GO
RECONFIGURE ;
GO
```

6. Set up the login credentials. Edit the query and paste in the previously copied `Identity and Secret` before executing it.

```
-- Load Cryptographic provider
CREATE CRYPTOGRAPHIC PROVIDER EKM_Prov
FROM FILE = 'C:\Program Files\hcs\bin\htsqlekm_provider.dll';
GO

-- Create credential for System Administrator
CREATE CREDENTIAL sa_ekm_tde_cred
WITH IDENTITY = '',
SECRET = ''
FOR CRYPTOGRAPHIC PROVIDER EKM_Prov ;
GO

-- Add credential to admin login
ALTER LOGIN [MS-SQL-TDE-KC10\Administrator]
ADD CREDENTIAL "sa_ekm_tde_cred";
GO
```

2.8. Create a master key on KeyControl

1. Sign in to the KeyControl WebGUI using an account with Cloud Admin privileges.
2. Select the **CLOUDKEYS** tab.
3. Select **CloudKeys**.
4. In the **Key Set** dropdown, select your previously created TDE keyset.
5. Select **Actions > Create CloudKey**.
6. Enter a **Name** for the key.
7. Select a **Cipher**. See KeyControl product documentation for selecting a cipher best suited for your implementation.
8. Select **Continue**.
9. Edit the remaining options as needed.
10. Select **Apply**.

    The CloudKey is now created.

11. Select **Close**.

2.9. Create the TDE key within Microsoft SQL Server

1. Create a TDE key:

```
USE master;
CREATE ASYMMETRIC KEY TDE_KEY
```

```
FROM PROVIDER EKM_Prov WITH
PROVIDER_KEY_NAME = 'tdersa2048key',
CREATION_DISPOSITION = OPEN_EXISTING;
GO
```

2. Create a login for the TDE user:

```
CREATE LOGIN TDE_Login
FROM ASYMMETRIC KEY TDE_KEY ;
GO
```

3. Create another credential for the TDE login, remembering to edit the SECRET and IDENTITY parameters:

```
CREATE CREDENTIAL tde_ekm_cred
WITH IDENTITY = '',
SECRET = ''
FOR CRYPTOGRAPHIC PROVIDER EKM_Prov ;
GO
```

4. Add this credential to TDE login:

```
ALTER LOGIN TDE_Login
ADD CREDENTIAL tde_ekm_cred
GO
```

2.10. Test the database encryption

The following queries test encryption with a database encryption key which is wrapped by the TDE key that was created in Create the TDE key within Microsoft SQL Server.

1. Prepare to use the test database:

```
USE testdb
GO
```

2. Create a symmetric encryption key for the database which will be wrapped by the TDE key:

```
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM  = AES_256
ENCRYPTION BY SERVER ASYMMETRIC KEY TDE_KEY;
GO
```

3. Enable encryption on the test database:

```
ALTER DATABASE testdb
SET ENCRYPTION ON ;
GO
```

4. Check the state of keys and encryption:

```
USE master
GO

Select * from sys.dm_database_encryption_keys
Select * from sys.asymmetric_keys
GO

SELECT DB_NAME(database_id) AS DatabaseName, encryption_state,
encryption_state_desc =
CASE encryption_state
   WHEN '0' THEN 'No database encryption key present, no encryption'
   WHEN '1' THEN 'Unencrypted'
   WHEN '2' THEN 'Encryption in progress'
   WHEN '3' THEN 'Encrypted'
   WHEN '4' THEN 'Key change in progress'
   WHEN '5' THEN 'Decryption in progress'
   WHEN '6' THEN 'Protection change in progress (The certificate or asymmetric key that is encrypting the
database encryption key is being changed.)'
   ELSE 'No Status'
   END,
percent_complete,encryptor_thumbprint, encryptor_type FROM sys.dm_database_encryption_keys

SELECT DB_NAME(database_id) AS DatabaseName, encryption_state,percent_complete,encryptor_thumbprint,
encryptor_type FROM sys.dm_database_encryption_keys
```

5. If required, you can trace events to debug EKM API usage:

```
create event session testsession on server
add event sqlserver.sec_ekm_provider_called (action
(package0.callstack,sqlserver.tsql_stack,sqlserver.client_app_name, sqlserver.client_hostname,
sqlserver.context_info, sqlserver.database_name,sqlserver.nt_username, sqlserver.session_id,
sqlserver.sql_text) where counter <= 100)
add target package0.ring_buffer ( set max_memory=1, occurrence_number=1)
with (MAX_DISPATCH_LATENCY=1 seconds)
go

alter event session testsession on server state=start
```

This concludes the basic workflow for TDE.

2.11. Rotate the key manually in KeyControl

Auto-rotation is not supported.

1. Sign in to the KeyControl WebGUI using an account with Cloud Admin privileges.
2. Select the **CLOUDKEYS** tab.

3. Select **CloudKeys > Key Set**, then select a **CloudKey** from the list.
4. Select **Rotate Now**. A new version of the key will be created. You can see this in the **Versions** tab. The new version has a star next to it.
5. Create the new key on Microsoft SQL Server. Edit the Secret and Identity as required.

```
USE master;
CREATE ASYMMETRIC KEY TDE_KEY_v2
FROM PROVIDER EKM_Prov WITH
PROVIDER_KEY_NAME = 'tdersa2048key',
CREATION_DISPOSITION = OPEN_EXISTING;
GO

CREATE CREDENTIAL tde_ekm_cred_v2
WITH IDENTITY = 'htdc-tde-cred',
SECRET = ''
FOR CRYPTOGRAPHIC PROVIDER EKM_Prov ;
GO
```

6. Create new logins for the new key:

```
CREATE LOGIN TDE_Login_v2
FROM ASYMMETRIC KEY TDE_KEY_v2 ;
GO

ALTER LOGIN TDE_Login_v2
ADD CREDENTIAL tde_ekm_cred_v2
GO
```

7. Set the second key as the new key to use:

```
use testdb
GO
ALTER DATABASE ENCRYPTION KEY
ENCRYPTION BY SERVER ASYMMETRIC KEY TDE_KEY_v2;
GO
```

The database encryption key has been rewrapped. Now the test database is encrypted using the database key version 2.

2.12. Shut down encryption on the database and remove credentials

1. Shut down encryption:

```
ALTER DATABASE testdb
SET ENCRYPTION OFF ;
GO

USE testdb
```

```
DROP DATABASE ENCRYPTION KEY
GO

USE master
GO

DROP ASYMMETRIC KEY TDE_KEY
DROP ASYMMETRIC KEY TDE_KEY_v2

ALTER LOGIN TDE_Login
DROP CREDENTIAL tde_ekm_cred
GO
DROP LOGIN TDE_Login
DROP CREDENTIAL tde_ekm_cred
GO

ALTER LOGIN TDE_Login_v2
DROP CREDENTIAL tde_ekm_cred_v2
GO
DROP LOGIN TDE_Login_v2
DROP credential tde_ekm_cred_v2
GO
```

2. Remove the credential from the admin login:

```
ALTER LOGIN [MS-SQL-TDE-KC10\Administrator]
DROP CREDENTIAL sa_ekm_tde_cred;
GO
DROP credential sa_ekm_tde_cred
GO
```

Chapter 3. Additional resources and related products

3.1. KeyControl

3.2. Entrust products

3.3. nShield product documentation