## Transparent Data Encryption (TDE) Implementation Guide

**Purpose:** This document provides a comprehensive guide to implementing Transparent Data Encryption (TDE) in SQL Server. It outlines the general process for setting up TDE on a new database and provides a specific example using the "PIMs" database.

Encryption Keys will be located here as a backup should the instance is compromised or in a state that the server cannot be accessed: [\\futurelocationtobeadded\oncetaskcompleted](file:///\\futurelocationtobeadded\oncetaskcompleted)

### Section 1: Generic TDE Implementation Steps

**Note:** These steps assume that TDE has not been previously configured on the SQL Server instance.

1. **Create a Master Key:**
   * A master key is required to encrypt the certificate used in the TDE process. If one does not exist, create it using a strong password. Store this password securely, preferably in a centralized password management system like CyberArk.

USE [master]  
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'YourStrongPassword';

1. **Backup the Master Key:**
   * It's crucial to back up the master key to a secure location. This backup will be essential for restoring the database in case of a server failure.

BACKUP MASTER KEY TO FILE = 'YourSecureBackupLocation\masterkey\_backup.bak'  
ENCRYPTION BY PASSWORD = 'YourStrongPassword';

1. **Create a Certificate:**
   * Create a certificate that will be used to encrypt the database encryption key.

CREATE CERTIFICATE TDE\_Certificate WITH SUBJECT = 'Certificate for TDE';

1. **Backup the Certificate:**
   * Backup both the certificate and its private key to a secure location. This backup is as crucial as the master key backup.

BACKUP CERTIFICATE TDE\_Certificate  
TO FILE = 'YourSecureBackupLocation\TDE\_Certificate.cer'  
WITH PRIVATE KEY (  
 FILE = 'YourSecureBackupLocation\TDE\_Certificate.pvk',  
 ENCRYPTION BY PASSWORD = 'YourStrongPassword'  
);

1. **Create a Database Encryption Key:**
   * Generate a database encryption key for the specific database you want to encrypt.

USE [YourDatabaseName]  
CREATE DATABASE ENCRYPTION KEY  
WITH ALGORITHM = AES\_256  
ENCRYPTION BY SERVER CERTIFICATE TDE\_Certificate;

1. **Enable TDE:**
   * Turn on encryption for the database. This process might take some time depending on the size of the database.

ALTER DATABASE YourDatabaseName SET ENCRYPTION ON;

### Section 2: TDE Implementation for the PIMs Database

This section explains the TDE setup for the "MassMarketingData" database, illustrating the practical application of the generic steps outlined above. Which would allow you to restore the keys from the network share to the server location.

**Step 1: Master Key and Certificate Creation and Backup**

The provided script checks if a master key already exists. If not, it creates one with the specified password, backs it up, creates a certificate for TDE, and backs up the certificate with its private key. If a master key already exists, it skips the master key creation and backup but proceeds with creating and backing up the certificate.

IF NOT EXISTS (SELECT 1 FROM sys.symmetric\_keys WHERE symmetric\_key\_id = 101)  
BEGIN   
 USE [master]  
 CREATE MASTER KEY ENCRYPTION  
 BY PASSWORD='Password in CyberArk named BACKUP MASTER KEY (Test & Prod PIMS) On exportedmasterkey';

PRINT 'SUCCESS: Master Key and Database Certificate Created and Backed up'  
END

**Step 2: Database Encryption Key Creation and Encryption**

This step checks if the "MassMarketingData" database is already encrypted. If not, it creates a database encryption key using the previously created certificate and enables TDE for the database. It also provides a script to check the encryption status. If the database is already encrypted, it prints a warning message. Prior to start the encryption please ensure that the files are copied from the network share to a folder created “E:\EncryptionKey” on the PIMS instance.

IF NOT EXISTS (SELECT 1 FROM sys.dm\_database\_encryption\_keys WHERE DB\_NAME(database\_id) = 'MassMarketingData')  
BEGIN

CREATE CERTIFICATE TDE\_Certificate

FROM FILE = N'E:\EncryptionKey\ALLIED\_PIMSDB\_master\_TDE\_Certificate.cer'

WITH PRIVATE KEY (

FILE = N'E:\EncryptionKey\ALLIED\_PIMSDB\_master\_TDE\_Certificate.pvk',

DECRYPTION BY PASSWORD = 'Password in CyberArk named TDE PIMS MassMarketingData Certificate - Archived On TDE PIMS Certificate’ );

PRINT 'ENCYPTION KEY APPLIED TO DATABASE AND TURNED ON PLEASE CHECK ON STATUS WITH THE FOLLOWING SCRIPT:  
  
 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 '  
END   
 ELSE  
BEGIN  
  
 PRINT 'WARNING: DATABASE IS ALREADY ENCRYPTED!'  
  
END

**Step 3: Validation**

This script will allow validation to see the progress of the encryption.

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

### Section 3: TDE Implementation for the Unitrac Database

This section explains the TDE setup for the "Unitrac" database, illustrating the practical application of the generic steps outlined above. Which would allow you to restore the keys from the network share to the server location.

**Step 1: Master Key and Certificate Creation and Backup**

The provided script checks if a master key already exists. If not, it creates one with the specified password, backs it up, creates a certificate for TDE, and backs up the certificate with its private key. If a master key already exists, it skips the master key creation and backup but proceeds with creating and backing up the certificate.

IF NOT EXISTS (SELECT 1 FROM sys.symmetric\_keys WHERE symmetric\_key\_id = 101)  
BEGIN   
 USE [master]  
 CREATE MASTER KEY ENCRYPTION  
 BY PASSWORD='Password in CyberArk named UniTrac TDE Master Key On UniTrac TDE Master Key';

PRINT 'SUCCESS: Master Key and Database Certificate Created and Backed up'  
END

**Step 2: Database Encryption Key Creation and Encryption**

This step checks if the "Unitrac" database is already encrypted. If not, it creates a database encryption key using the previously created certificate and enables TDE for the database. It also provides a script to check the encryption status. If the database is already encrypted, it prints a warning message. Prior to start the encryption please ensure that the files are copied from the network share to a folder created “E:\EncryptionKey” on the PIMS instance.

IF NOT EXISTS (SELECT 1 FROM sys.dm\_database\_encryption\_keys WHERE DB\_NAME(database\_id) = 'Unitrac')  
BEGIN

CREATE CERTIFICATE Unitrac\_Certificate

FROM FILE = N'Add\_location\_and\_filename.cer'

WITH PRIVATE KEY (

FILE = N'Add\_location\_and\_filename.pvk',

DECRYPTION BY PASSWORD = 'Password in CyberArk named TDE UnitracCert On TDE UniTrac Certificate’ );

PRINT 'ENCYPTION KEY APPLIED TO DATABASE AND TURNED ON PLEASE CHECK ON STATUS WITH THE FOLLOWING SCRIPT:  
  
 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 '  
END   
 ELSE  
BEGIN  
  
 PRINT 'WARNING: DATABASE IS ALREADY ENCRYPTED!'  
  
END

(THIS MAY ONLY EXIST IN TEST FOR UNITRAC)   
IF NOT EXISTS (SELECT 1 FROM sys.dm\_database\_encryption\_keys WHERE DB\_NAME(database\_id) = 'LFP')  
BEGIN

CREATE CERTIFICATE LFP\_Certificate

FROM FILE = N'Add\_location\_and\_filename.cer'

WITH PRIVATE KEY (

FILE = N'Add\_location\_and\_filename.pvk',

DECRYPTION BY PASSWORD = 'Password in CyberArk named

TDE LFPCert On TDE LFPCert’ );

PRINT 'ENCYPTION KEY APPLIED TO DATABASE AND TURNED ON PLEASE CHECK ON STATUS WITH THE FOLLOWING SCRIPT:  
  
 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 '  
END   
 ELSE  
BEGIN  
  
 PRINT 'WARNING: DATABASE IS ALREADY ENCRYPTED!'  
  
END

**Step 3: Validation**

This script will allow validation to see the progress of the encryption.

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