SQL Server Encryption Overview

September 2, 2015

ABOUT ME

- Edmund Poillion
- Data Platform Systems Engineer
- Skyline Associate since 1999
- Started in App Dev, changed focus to SQL Server in 2012
- Email: epoillion@skylinetechnologies.com
- Twitter: @epoillion

TOPICS

- SQL Server Data
- Encryption concepts supported by SQL Server
- SQL Server encryption hierarchy
- Transparent Data Encryption (TDE)
- Column-Level Encryption

SQL SERVER DATA

- DATA AT REST
  - Data files
  - Transaction Log files
  - Backup files
- DATA IN USE
  - Buffer pool
  - In transit

SQL SERVER DATA

- DATA FILES
  - Table and index data stored in data files (*.mdf, *.ndf)
  - Can have multiple data files, grouped into Filegroups
  - Data in files organized into 8 kb data pages
  - Data pages organized into 8 page extents
    - Uniform extent – all pages belong to same object
    - Mixed extent – pages from different objects

SQL SERVER DATA

- LOG FILES
  - Divided into multiple Virtual Log Files (VLF)
  - Records all transactions and the database modifications made by each transaction
  - Must be truncated on a regular basis so it doesn’t fill up
- BACKUP FILES
  - Types: Full, Differential, Log

SQL SERVER DATA

- BUFFER POOL
  - Holds cached data pages in memory
  - Pages stored exactly as on disk
  - Pages required for query are loaded into buffer pool, then query is executed against buffer pool
  - Pages updated in buffer pool, written to disk later
  - Transaction log allows recovery of changes in case of outage

SQL SERVER DATA

- IN TRANSIT DATA
  - Data after it leaves the database engine on the way to data consumer
    - Query results in SSMS
    - Across network to corporate applications
    - Across world via web application

ENCRYPTION CONCEPTS

- HASHES
- ASYMMETRIC KEYS
- SYMMETRIC KEYS
- CERTIFICATES
- ENCRYPTION ALGORITHMS

ENCRYPTION CONCEPTS

- HASHES
  - Very fast.
  - One-way encryption.
  - Common for passwords and comparisons.
  - Commands
    - HASHBYTES - MD2, MD4, MD5, SHA, SHA1, SHA2_256, SHA2_512
    - PWDCOMPARE

ENCRYPTION CONCEPTS

- SYMMETRIC KEYS
  - Good performance
  - Same key used for encryption and decryption
  - Should be secured by another key
  - Can be secured by
    - Certificate
    - Password
    - Symmetric Key
    - Asymmetric Key
    - EKM Module

ENCRYPTION CONCEPTS

- ASYMMETRIC KEYS
  - Uses combination of a public and a private key
  - Public key is shared openly, private key guarded
  - Public key is used to encrypt, private key used to decrypt
  - Not best choice for securing data because of performance
  - Best used to encrypt other keys

ENCRYPTION CONCEPTS

- CERTIFICATES
  - A certificate is a digitally signed security object that contains a public (and optionally a private) key for SQL Server
  - You can use externally generated certificates or SQL Server can generate certificates.
  - SQL Server certificates comply with the IETF X.509v3 certificate standard

ENCRYPTION CONCEPTS

- CERTIFICATES
  - Useful because of the option of both exporting and importing keys to X.509 certificate files
  - Can offer expiration management (SQL Server does not enforce when used for encryption)

ENCRYPTION CONCEPTS

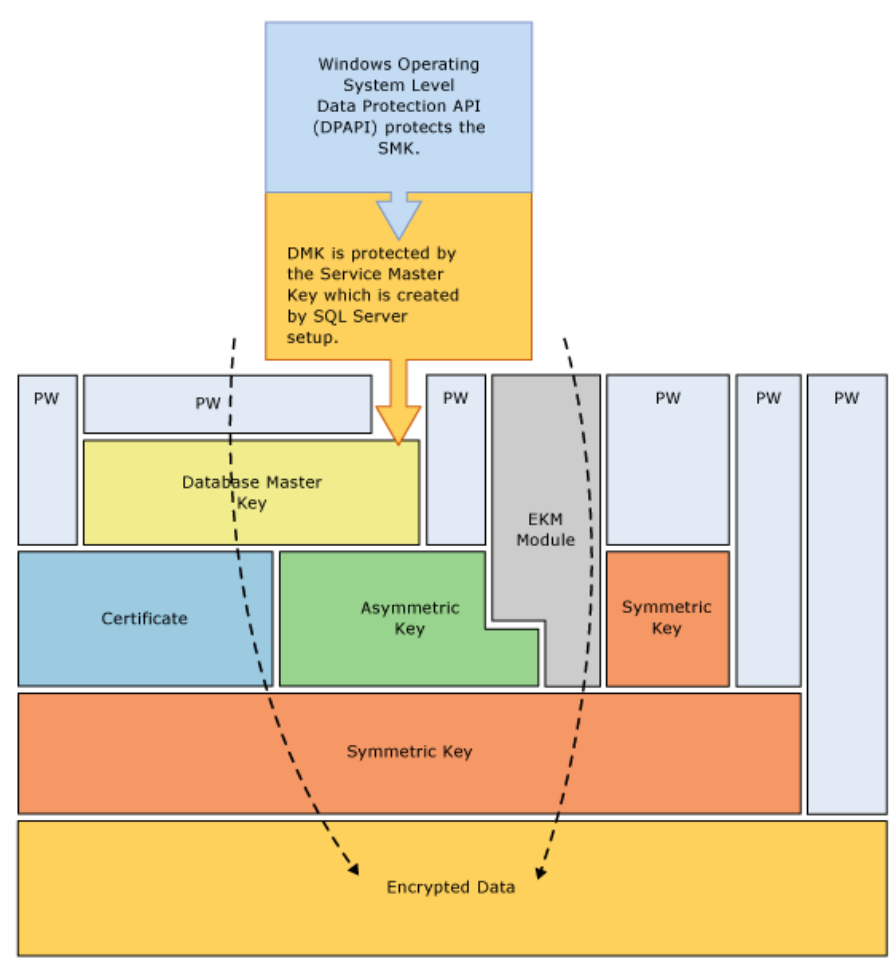
- ENCRYPTION ALGORITHMS
  - SQL Server asymmetric keys support RSA_512, RSA_1024, RSA_2048
  - SQL Server symmetric keys support DES, TRIPLE_DES, TRIPLE_DES_3KEY, RC2, ~~RC4~~, ~~RC4_128~~, DESX, AES_128, AES_192, AES_256
  - RC4 and RC4_128 have been deprecated and should not be used

DEMO

- ENCRYPTION DMVs

SQL SERVER ENCRYPTION HIERARCHY

- Service Master Key (SMK)
- Database Master Key (DMK)
- Database Encryption Key (DEK)
- Key Backups



https://msdn.microsoft.com/en-us/library/ms189586(v=sql.110).aspx

SQL SERVER ENCRYPTION HIERARCHY

- SERVICE MASTER KEY
  - Root of SQL Server Encryption Hierarchy
  - Instance level symmetric key
  - SQL Server 2012+ uses AES encryption. Older versions use 3DES
  - Generated automatically first time it is needed, normally during installation
  - Best Practice: Back up the Service Master Key and store the backed up copy in a secure, off-site location

SQL SERVER ENCRYPTION HIERARCHY

- SERVICE MASTER KEY
  - Encrypted using Windows Data Protection API (DPAPI) and the local machine key using a key derived from the Windows credentials of the SQL Server service account
  - Can only be opened by the Windows service account under which it was created or by a principal with access to both the service account name and its password

SQL SERVER ENCRYPTION HIERARCHY

- SERVICE MASTER KEY
  - To change the SQL Server service account, use SQL Server Configuration Manager
  - Regenerating or restoring the Service Master Key involves decrypting and re-encrypting the complete encryption hierarchy

SQL SERVER ENCRYPTION HIERARCHY

- DATABASE MASTER KEY
  - Database level symmetric key
  - Used to protect the private keys of certificates and asymmetric keys that are present in the database
  - Encrypted by using the Triple DES algorithm and a user-supplied password

SQL SERVER ENCRYPTION HIERARCHY

- DATABASE MASTER KEY
  - To enable the automatic decryption of the master key, a copy of the key is also encrypted by using the SMK. It is stored in both the database where it is used and in the master database.
  - The copy of the DMK stored in the master database is silently updated whenever the DMK is changed. However, this default can be changed by using the DROP ENCRYPTION BY SERVICE MASTER KEY option of the ALTER MASTER KEY statement.
  - A DMK that is not encrypted by the service master key must be opened by using the OPEN MASTER KEY statement and a password.

DEMO

- DATABASE MASTER KEY

SQL SERVER ENCRYPTION HIERARCHY

- DATABASE ENCRYPTION KEY
  - Database level symmetric key
  - Used for transparently encrypting a database
  - Protected by either a certificate protected by the database master key of the master database, or by an asymmetric key stored in an EKM
  - Stored in the database boot record for availability during recovery
  - Supports AES_128, AES_192, AES_256, TRIPLE_DES_3KEY

SQL SERVER ENCRYPTION HIERARCHY

- KEY BACKUPS
  - Always backup your keys and certificates
    - Service Master Key
    - Database Master Key
    - Certificates used to protect database encryption key
  - No “Back Door” if you lose these

TRANSPARENT DATA ENCRYPTION

- Overview
- Availability
- What is encrypted
- What is not encrypted
- Other considerations
- Implementation steps
- Key storage
- Disaster recovery
- SQL Server 2014 In-Memory OLTP objects

TRANSPARENT DATA ENCRYPTION

- OVERVIEW
  - Performs real-time I/O encryption and decryption of the data and log files
  - TDE protects data "at rest", meaning while on disk
  - Enables software developers to encrypt data without changing existing applications.

TRANSPARENT DATA ENCRYPTION

- OVERVIEW
  - Data is encrypted using the database encryption key (DEK)
  - Encryption of the database file is performed at the page level
  - The pages in an encrypted database are encrypted before they are written to disk and decrypted when read into memory

TRANSPARENT DATA ENCRYPTION

- OVERVIEW
  - TDE does not increase the size of the encrypted database
  - All files and filegroups will be encrypted. If there are any read-only filegroups, encryption will fail
  - Cannot encrypt master, msdb, or model

TRANSPARENT DATA ENCRYPTION

- AVAILABILITY
  - SQL Server 2008 and up
  - Enterprise Edition only
    - Cannot restore TDE database to Standard Edition

TRANSPARENT DATA ENCRYPTION

- WHAT IS ENCRYPTED
  - Data files
  - Transaction log files
  - Database backups
  - tempdb database
    - Encrypted if any database on instance is encrypted
    - Encrypted using AES 256, outside of normal TDE encryption hierarchy

TRANSPARENT DATA ENCRYPTION

- WHAT IS NOT ENCRYPTED
  - Buffer pool
  - Data “on the wire”
  - FILESTREAM data

TRANSPARENT DATA ENCRYPTION

- OTHER CONSIDERATIONS
  - Performance
    - Will use extra CPU (5% - 10% more is common)
    - Must test in your environment
  - Database Compression
    - Works with TDE, page data is compressed before it is encrypted
    - More CPU, must decrypt and decompress each page

TRANSPARENT DATA ENCRYPTION

- OTHER CONSIDERATIONS
  - Backup Compression
    - Not recommended with TDE
    - Encrypted data does not compress well

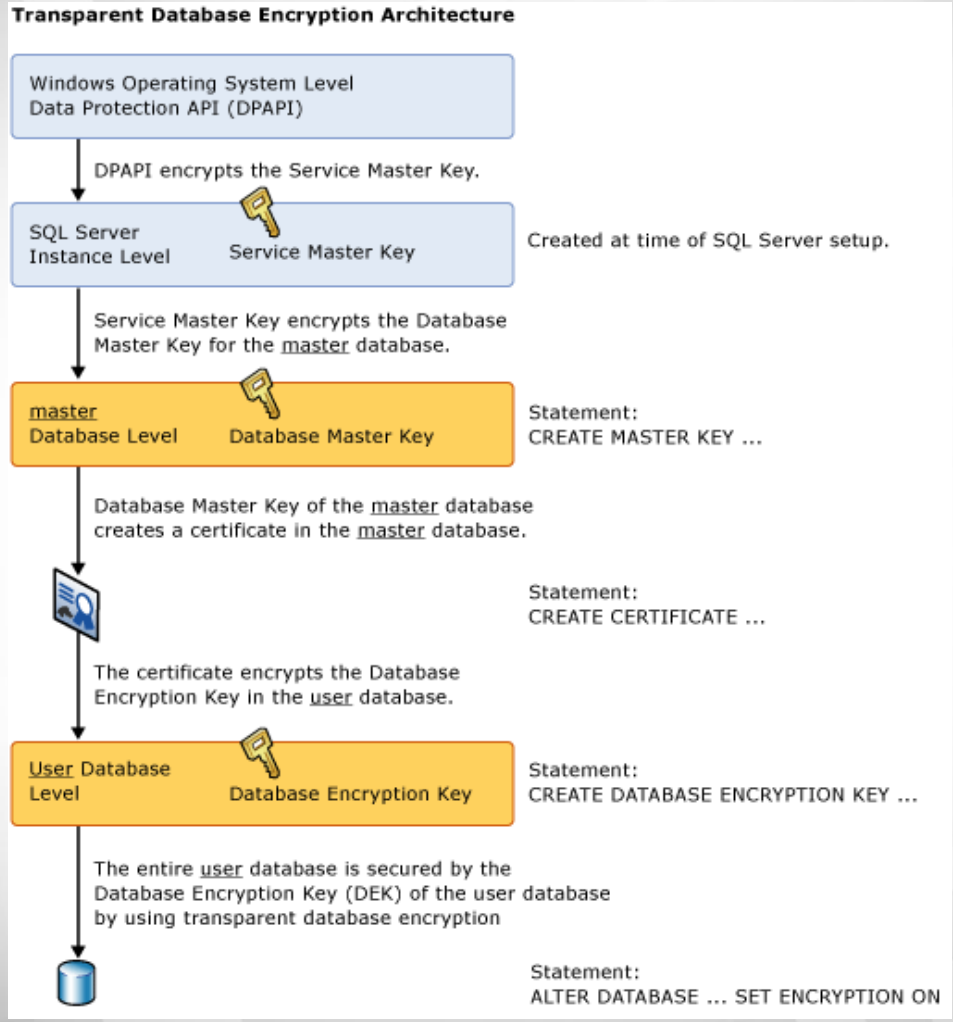
TRANSPARENT DATA ENCRYPTION

- OTHER CONSIDERATIONS
  - Backup Compression

AdventureWorks2014 Data File - 205 MB (189 Used) Log File - 18 MB	
Backup Type	**Backup Size (KB)**
Control	194,676
Compression	45,672
Encryption	194,672
Both	194,368

AdventureWorks2014Enlarged Data File - 1,741 MB (1,516 Used) Log File - 130 MB	
Backup Type	**Backup Size (KB)**
Control	1,551,480
Compression	535,940
Encryption	1,551,480
Both	1,550,212

Transparent Database Encryption Architecture



TRANSPARENT DATA ENCRYPTION

- IMPLEMENTATION STEPS
  - Create database master key for master database, which is protected by password and service master key
  - Create certificate in master database, which is protected by database master key
  - Create database encryption key in database to be encrypted, protected by certificate in master database

TRANSPARENT DATA ENCRYPTION

- IMPLEMENTATION STEPS
  - Set encryption on for database to be encrypted
  - Monitor encryption progress with sys.dm_database_encryption_keys DMV
    - Will show your database and tempdb are now encrypted
  - Verify your keys and certificates are backed up

TRANSPARENT DATA ENCRYPTION

- IMPLEMENTATION STEPS
  - Verify your database master key is in the master database with sys.symmetric_keys DMV
  - Verify your certificate is in the master database with sys.certificates DMV

DEMO

- IMPLEMENT TDE
- BACKUP KEYS
- RESTORE FROM DIFFERENT SERVER

TRANSPARENT DATA ENCRYPTION

- KEY STORAGE
  - Backups
  - Off-Site Storage
  - Cloud Service
  - Extensible Key Management

TRANSPARENT DATA ENCRYPTION

- DISASTER RECOVERY
  - When moving a TDE protected database, you must also move the certificate or asymmetric key that is used to open the DEK
  - The certificate or asymmetric key must be installed in the master database of the destination server, so that SQL Server can access the database files

TRANSPARENT DATA ENCRYPTION

- DISASTER RECOVERY
  - You must retain copies of both the certificate file and the private key file in order to recover the certificate

TRANSPARENT DATA ENCRYPTION

- SQL SERVER 2014 IN-MEMORY OLTP OBJECTS
  - TDE can be enabled on a database that has In-Memory OLTP objects
  - In-Memory OLTP log records are encrypted if TDE is enabled
  - Data in a MEMORY_OPTIMIZED_DATA filegroup is not encrypted if TDE is enabled

COLUMN-LEVEL ENCRYPTION

- Can be more selective on what you encrypt
- Requires application changes
- Can’t index on encrypted columns
- Use SQL Server Cryptographic Functions

DEMO

- COLUMN-LEVEL ENCRYPTION

Cryptographic Functions

Symmetric Encryption and Decryption	
EncryptByKey	DecryptByKey
EncryptByPassPhrase	DecryptByPassPhrase
Key_ID	Key_GUID

Asymmetric Encryption and Decryption	
EncryptByAsmKey	DecryptByAsmKey
EncryptByCert	DecryptByCert
Cert_ID	AsymKey_ID
CertProperty	

Signing and Signature Verification	
SignByAsymKey	VerifySignedByAsmKey
SignByCert	VerifySignedByCert

Symmetric Decryption with Automatic Key Handling
DecryptByKeyAutoCert

Encryption Hashing
HASHBYTES

QUESTIONS?