

MSC23

Protecting Your SharePoint 2010 Content with SQL Server 2008 Transparent Data Encryption

Michael Noel

Convergent Computing

Twitter: @MichaelTNoel

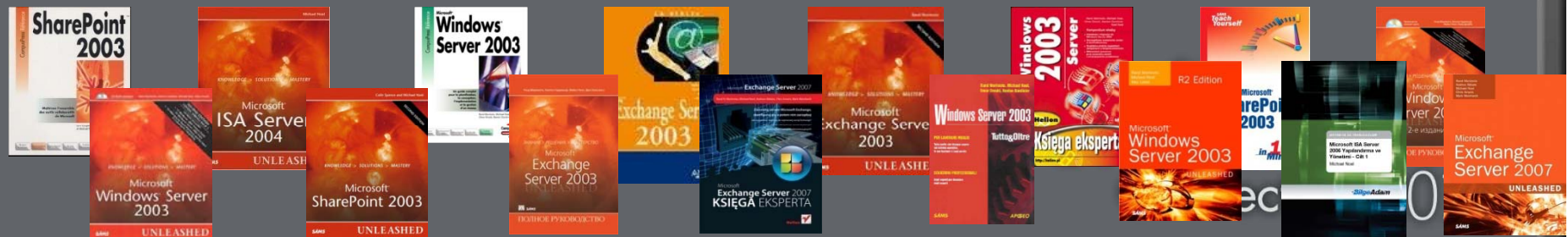
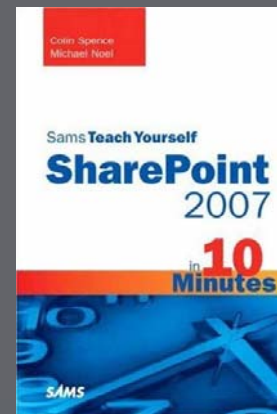
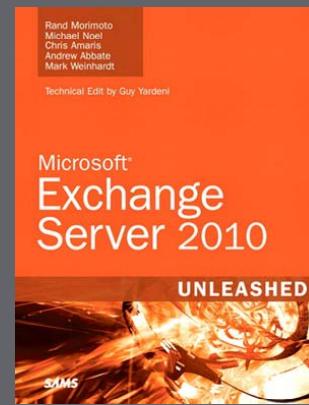
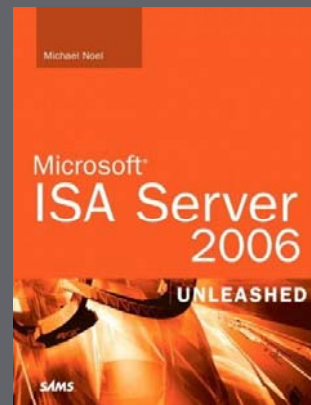
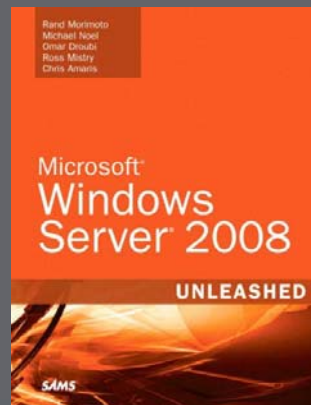
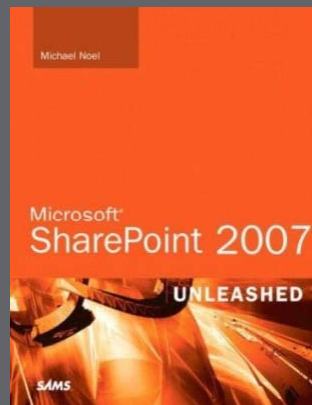


Microsoft®
SharePoint®
Connections2010

Michael Noel



- Technology book author; Over 15 titles translated into 20 languages worldwide
- Partner at Convergent Computing (www.cco.com) – San Francisco, U.S.A. based Consultants
- Specialties in SharePoint, Exchange, Security, and more...



Session Overview

- Discussion of various Encryption Options
 - Cell-level Encryption
 - File-Level Encryption (Bitlocker, EFS)
 - Transparent Data Encryption
 - Active Directory Rights Management Services (AD RMS)
- TDE Overview
- TDE for SharePoint Content Databases

The Problem: Unencrypted Data

- Data Stored Unencrypted on a SQL Server
- Stolen Backups or Administrators of a Server can have access to all SharePoint Content
- Governmental and Industry Regulation Restricts Storage of Content Unencrypted

The Solution: Data Encryption

- Many Options, same concept
- Files are stored in unreadable format, using PKI based encryption
- Some Options require Application Support (i.e. Cell-level Encryption), which SharePoint doesn't support

Cell-level Encryption

- Available with either SQL 2005 or SQL 2008
- Encrypts individual cells in a database
- Requires a password to access the cell
- Requires that columns be changed from their original data type to varbinary
- Advantage is that only specific info is encrypted
- Disadvantage is that you cannot use this for SharePoint Databases

File-level Encryption

- Two forms, older Encrypting File System (EFS) and Bitlocker
- EFS encrypts data at the File Level
- Bitlocker encrypts data at the Volume Level
- Bitlocker Encrypts every file on the disk, not just database files
- Could be used together with TDE

File-level Encryption

- Biggest drawback: Heavy Performance Hit
- No support for prefetch or asynchronous I/O
- I/O operations can become bottlenecked and serialized
- Doesn't protect the volume when accessed across the network
- Only really feasible in very small workgroup scenarios, rarely applies to SharePoint

Active Directory Rights Management Services (AD RMS)

- Encrypts content upon access and removal, not in storage
- Provides Rights Protection, which can expire a document or limit the ability to:
 - Print
 - Cut/Paste
 - Programmatically access
 - Save As a different file
- Can be used with TDE

Transparent Data Encryption (TDE)

- New in SQL Server 2008
- Only Available with the Enterprise Edition
- Seamless Encryption of Individual Databases
- Transparent to Applications, including SharePoint

Transparent Data Encryption (TDE)

- When enabled, encrypts Database, log file, any info written to TempDB, snapshots, backups, and Mirrored DB instance, if applicable
- Operates at the I/O level through the buffer pool, so any data written into the MDF is encrypted
- Can be selectively enabled on specific databases
- Backups cannot be restored to other servers without a copy of the private key, stolen MDF files are worthless to the thief
- Easier Administration, Minimal server resources required (3%-5% performance hit)

Potential TDE Limitations

- Does not encrypt the Communication Channel (IPSec can be added)
- Does not protect data in memory (DBAs could access)
- Cannot take advantage of SQL 2008 Backup Compression
- TempDB is encrypted for the entire instance, even if only one DB is enabled for TDE, which can have a performance effect for other DBs
- Replication or FILESTREAM data is not encrypted when TDE is enabled

How TDE Works

- Windows Data Protection API (DPAPI) at root of encryption key hierarchy
- DPAPI creates and protects Service Master Key (SMK) during SQL Setup
- SMK used to protect Database Master Key (DMK)
- DMK used to protect Certificate and Asymmetric Key
- Certificate and Asymmetric Key used to create Database Encryption Key (DEK)

Windows OS Level

Data Protection API (DPAPI)

DPAPI Encrypts SMK

SQL Instance Level

Service Master Key

SMK encrypts the DMK for master DB

master DB Level

Database Master Key

DMK creates Cert in master DB

master DB Level

Certificate

Certificate Encrypts DEK in Content DB

Content DB Level

Database Encryption Key

DEK used to encrypt Content DB



High Level Steps to enable TDE

1. Create the DMK
2. Create the TDE Cert
3. Backup the TDE Cert
4. Create the DEK
5. Encrypt the DB
6. Monitor Progress

Creating the Database Master Key (DMK)

- Symmetric key used to protect private keys and asymmetric keys
- Protected itself by Service Master Key (SMK), which is created by SQL Server setup
- Use syntax as follows:
 - USE master;
 - GO
 - CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'CrypticTDEpw4CompanyABC';
 - GO

Create Certificate Protected by DMK

- Protected by the DMK
- Used to protect the database encryption key
- Use syntax as follows:

```
USE master;
```

```
GO
```

```
CREATE CERTIFICATE CompanyABCtdeCert WITH  
SUBJECT = 'CompanyABC TDE Certificate' ;
```

```
GO
```

Backup Master Key and Cert

- Without a backup, data can be lost
- Backup creates two files, the Cert backup and the Private Key File
- Use following syntax:

```
USE master;
```

```
GO
```

```
BACKUP CERTIFICATE CompanyABCtdeCert TO FILE =  
'c:\Backup\CompanyABCtdeCERT.cer'
```

```
WITH PRIVATE KEY (
```

```
FILE = 'c:\Backup\CompanyABCtdeDECert.pvk',
```

```
ENCRYPTION BY PASSWORD = 'CrypticTDEpw4CompanyABC!' );
```

```
GO
```

Create a Database Encryption Key (DEK)

- DEK is used to encrypt specific database
- One created for each database
- Encryption method can be chosen for each DEK
- Use following syntax:

```
USE SharePointContentDB;
```

```
GO
```

```
CREATE DATABASE ENCRYPTION KEY
```

```
WITH ALGORITHM = AES_256
```

```
ENCRYPTION BY SERVER CERTIFICATE CompanyABCtdeCert
```

```
GO
```

Enable TDE

- Data encryption will begin after running command
- Size of DB will determine time it will take, can be lengthy and could cause user blocking
- Use following syntax:

```
USE SharePointContentDB
```

```
GO
```

```
ALTER DATABASE SharePointContentDB
```

```
SET ENCRYPTION ON
```

```
GO
```

Monitor TDE Progress

- State is Returned
- State of 2 = Encryption Begun
- State of 3 = Encryption Complete
- Use following syntax:

```
USE SharePointContentDB
```

```
GO
```

```
SELECT *
```

```
FROM sys.dm_database_encryption_keys
```

```
WHERE encryption_state = 3;
```

```
GO
```

Restoring TDE Encrypted DB to Other Server

- Step 1: Create new Master Key on Target Server (Does not need to match source master key)
- Step 2: Backup Cert and Private Key from Source
- Step 3: Restore Cert and Private Key onto Target (No need to export the DEK as it is part of the backup)

```
USE master;
```

```
GO
```

```
CREATE CERTIFICATE CompanyABCtdeCert
```

```
FROM FILE = 'C:\Restore\CompanyABCtdeCert.cer'
```

```
WITH PRIVATE KEY (
```

```
FILE = 'C:\Restore\CompanyABCtdeCert.pvk'
```

```
, DECRYPTION BY PASSWORD = 'CrypticTDEpw4CompanyABC!'
```

```
)
```

- Step 4: Restore DB

Demo

Encrypting SharePoint Content
DBs using Transparent Data
Encryption

Microsoft®
SharePoint®
Connections2010

Your Feedback is Important

Please fill out a session evaluation form and either put them in the basket near the exit or drop them off at the conference registration desk.

Thank you!

Session Code: MSC23

Microsoft®
SharePoint®
Connections2010

Thanks for attending!

Questions?

Michael Noel

Twitter: @MichaelTNoel

www.cco.com

Session Code: MSC23

