



HANSEN Group

Transparent Data Encryption

TDE im Vergleich zu EFS, BitLocker und Co.

Jens Hansen
Principal Consultant, CEO

Hansen Group AG ©2008



- Transparent Data Encryption ergänzt die Verschlüsselungsfunktionen, die seit SQL Server 2005 in die Datenbankengine integriert sind

- Schutz gegen “Offline-Attacken”:
 - Diebstahl von Hardware oder Backups
 - Server außerhalb des Datacenter (z.B. an Standorten)
 - Mobile Szenarien (z.B. Notebooks)
 - Aufbewahrung von Backups (z.B. Securitas)
 - Backup auf anderem Server wiederherstellen
 - Datenbank auf anderem Server attachen
 - Datenbankinhalte “raw” ermitteln

- Verfügbare Verschlüsselungsebenen und deren Scope:
 - Cell-Level = Offline, Zugriffskontrolle
 - Database-Level = Offline
 - Windows-Level = Offline

- Verschlüsselung einzelner Werte bei INSERT/UPDATE und Entschlüsselung bei SELECT durch Nutzung integrierter Funktionen
 - Verfügbar: Passphrase, Symmetric/Asymmetric-Key, Certificate
- Encryption liefert verschlüsselten Wert als VARBINARY, entschlüsselte Werte müssen manuell in den ursprünglichen Typ umgewandelt werden:

```
WITH tabl AS (  
    SELECT EncryptByPassPhrase('SECRET', 'Das ist ein Test') AS coll  
)  
SELECT CAST(DecryptByPassPhrase('SECRET', coll) AS VARCHAR) FROM tabl
```

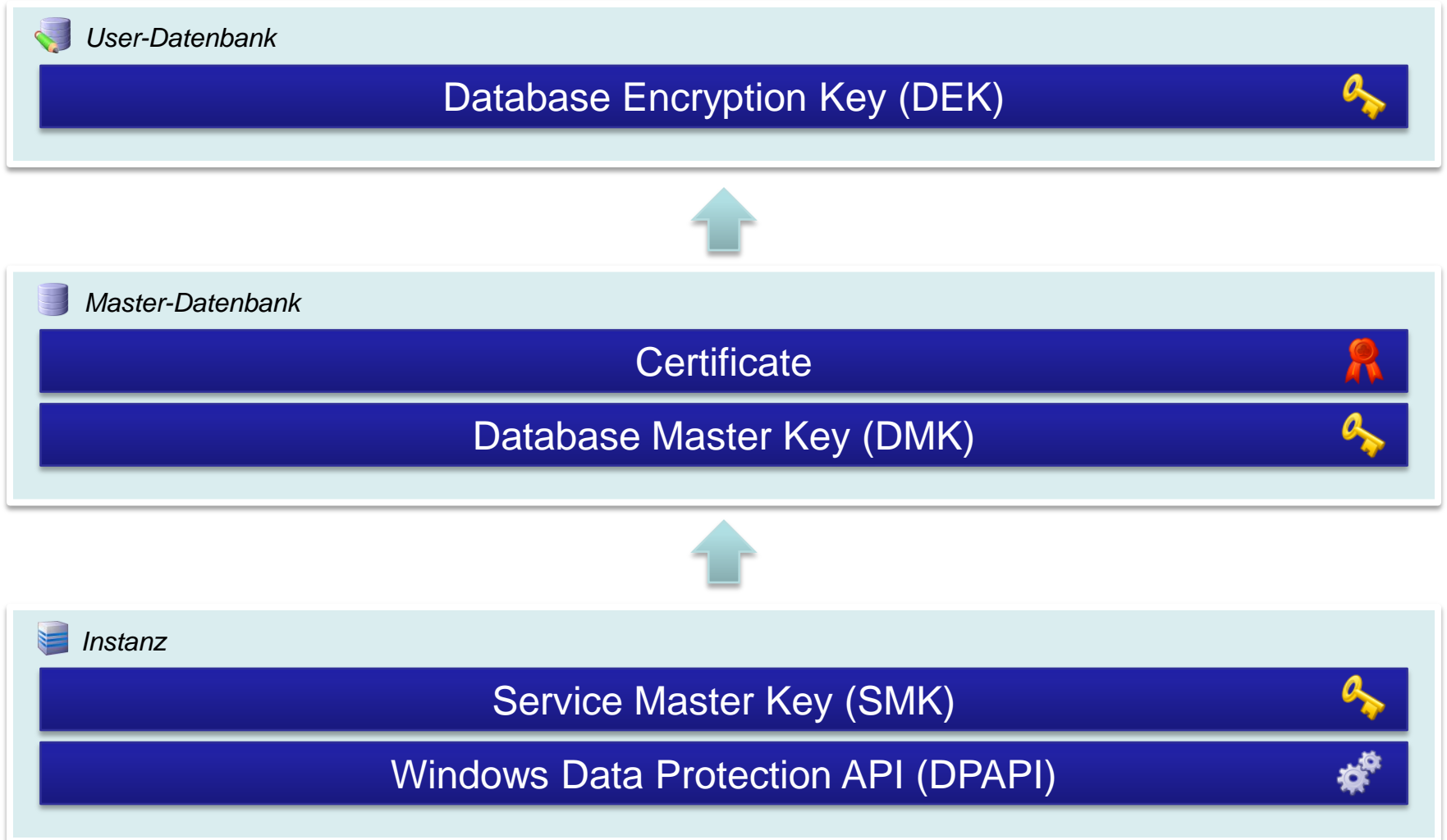
- Verschlüsselte Spalten ungeeignet in WHERE-Bedingungen (schlechte Performance) und ungeeignet als Primär- oder Fremdschlüssel
- Anpassungen am Schema- und Applikationsdesign erforderlich
- Cell-Level-Encryption ist auch in SQL Server 2008 eine valide Option

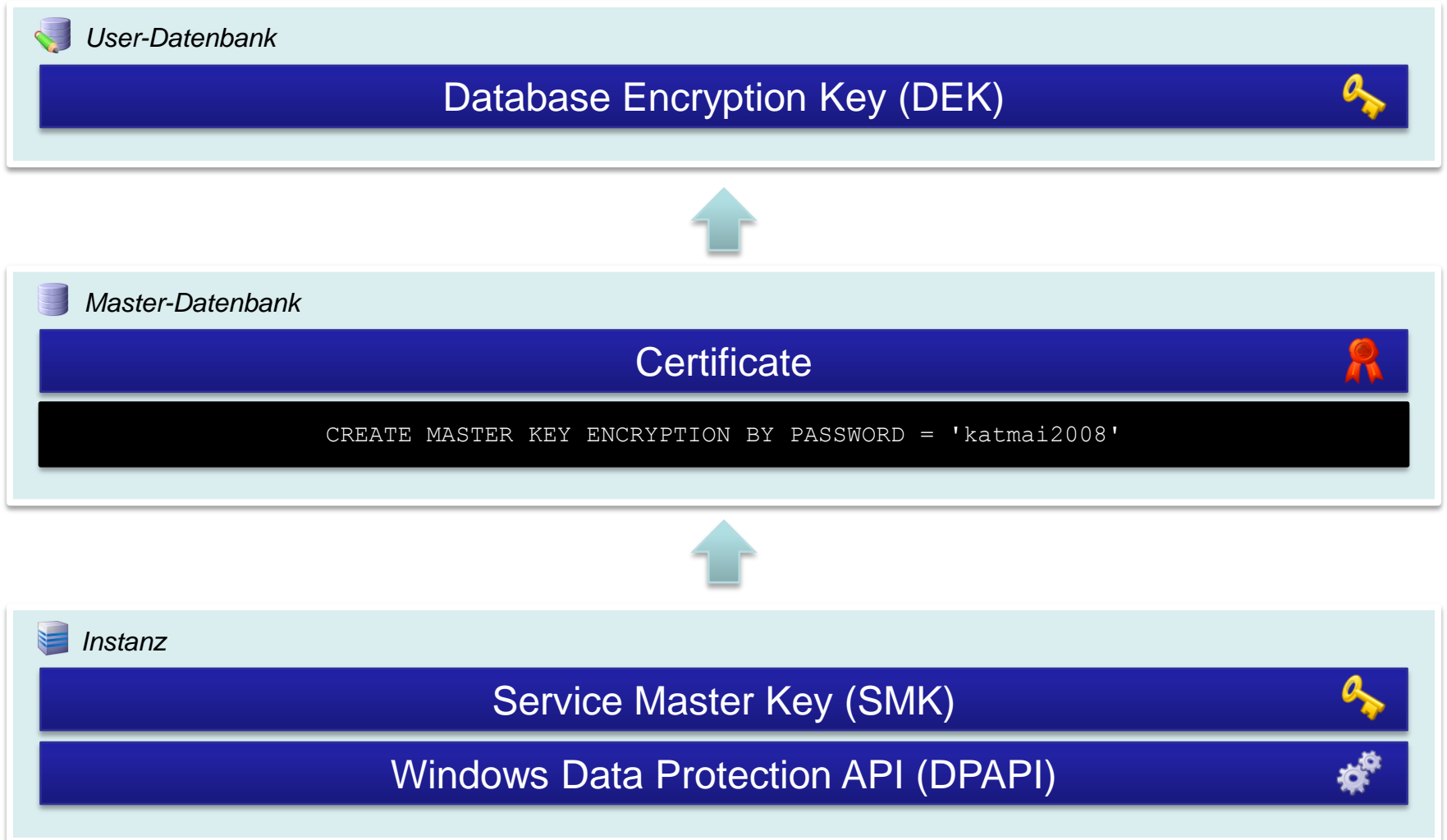
- TDE verschlüsselt pauschal die gesamte Datenbank durch „5 x F5“
- Verschlüsselung ist transparent (keine Design-Änderungen erforderlich)
- Was wird verschlüsselt? Datenfiles, Logfiles, Backups & Snapshots
- User-Datenbanken und Tempdb, aber keine System-Datenbanken
- TDE ist keine Form der Zugriffskontrolle, Zielsetzung von TDE ist Schutz gegen „Offline-Attacks“
- Kommunikation zwischen Server und Client nicht verschlüsselt
- TDE ist ein Feature der Enterprise bzw. Developer Edition


Grundlegendes zur transparenten Datenverschlüsselung (TDE):

<http://msdn.microsoft.com/de-de/library/bb934049.aspx>








 *User-Datenbank*

Database Encryption Key (DEK) 




 *Master-Datenbank*

```
CREATE CERTIFICATE [TDE_Certificate] WITH SUBJECT = 'TDE Demo Certificate'
```

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'katmai2008'
```



 *Instanz*

Service Master Key (SMK) 

Windows Data Protection API (DPAPI) 

User-Datenbank

```
CREATE DATABASE ENCRYPTION KEY WITH ALGORITHM = AES_256  
    ENCRYPTION BY SERVER CERTIFICATE [TDE_Certificate]
```



Master-Datenbank

```
CREATE CERTIFICATE [TDE_Certificate] WITH SUBJECT = 'TDE Demo Certificate'
```

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'katmai2008'
```



Instanz

Service Master Key (SMK) 

Windows Data Protection API (DPAPI) 

- Aktivieren/Deaktivieren startet einen asynchronen Background-Thread

```
ALTER DATABASE [AdventureWorksTDE] SET ENCRYPTION ON|OFF
```

- Status kann über sys.dm_database_encryption_keys abgefragt werden

```
SELECT
```

```
    DB_NAME(database_id) AS [database]  
    , encryption_state, percent_complete
```

```
FROM
```

```
    sys.dm_database_encryption_keys
```

database	encryption_state	percent_complete
tempdb	3	0
AdventureWorksTDE	3	0

- Sobald die erste User-Datenbank verschlüsselt wurde, wird automatisch auch Tempdb verschlüsselt → Performance-Impact auf gesamte Instanz
- „Percent Complete“ → Status des Background-Thread, sonst immer 0
 - „Encryption State“ = 3: Datenbank ist verschlüsselt
 - State = 2|5: Background-Thread ver- bzw. entschlüsselt Datenbank

1. Ersten Datensatz (physisch) der Tabelle Person.Address ermitteln:

```
SELECT TOP 1 * FROM Person.Address
```

AddressID	AddressLine1	AddressLine2	City	StateProvinceID	PostalCode
1	1970 Napa Ct.	NULL	Bothell	79	98011

2. Erste Page des Clustered Index mit DBCC ermitteln:

```
DBCC IND(N'AdventureWorksTDE', N'Person.Address', -1)
```

3. Page-Dump mit DBCC ansehen:

```
DBCC PAGE(N'AdventureWorksTDE', 1, 9312, 3) WITH TABLERESULTS
```

Slot 0 Column 1 Offset 0x4 L...	AddressID	1
Slot 0 Column 2 Offset 0x31 ...	AddressLine1	1970 Napa Ct.
Slot 0 Column 3 Offset 0x0 L...	AddressLine2	[NULL]
Slot 0 Column 4 Offset 0x4b ...	City	Bothell
Slot 0 Column 5 Offset 0x8 L...	StateProvinceID	79
Slot 0 Column 6 Offset 0x59 ...	PostalCode	98011

1. Offset in MDF-Datendatei ermitteln

```
SELECT CONVERT (VARBINARY (8), (9312*8192) + 100)
```

→ Offset: 0x048C0064

AdventureWorks.mdf

048C0080	D5 8B 00 00 08 00 04 04	00 4B 00 4B 00 59 00 63	01.....K.K.Y.c
048C0090	00 31 00 39 00 37 00 30	00 20 00 4E 00 61 00 70	.1.9.7.0. .N.a.p
048C00A0	00 61 00 20 00 43 00 74	00 2E 00 42 00 6F 00 74	.a. .C.t...B.o.t
048C00B0	00 68 00 65 00 6C 00 6C	00 39 00 38 00 30 00 31	.h.e.l.l.9.8.0.1
048C00C0	00 31 00 30 00 24 00 02	00 00 00 4F 00 00 00 9E	.1.0.\$.....O...!

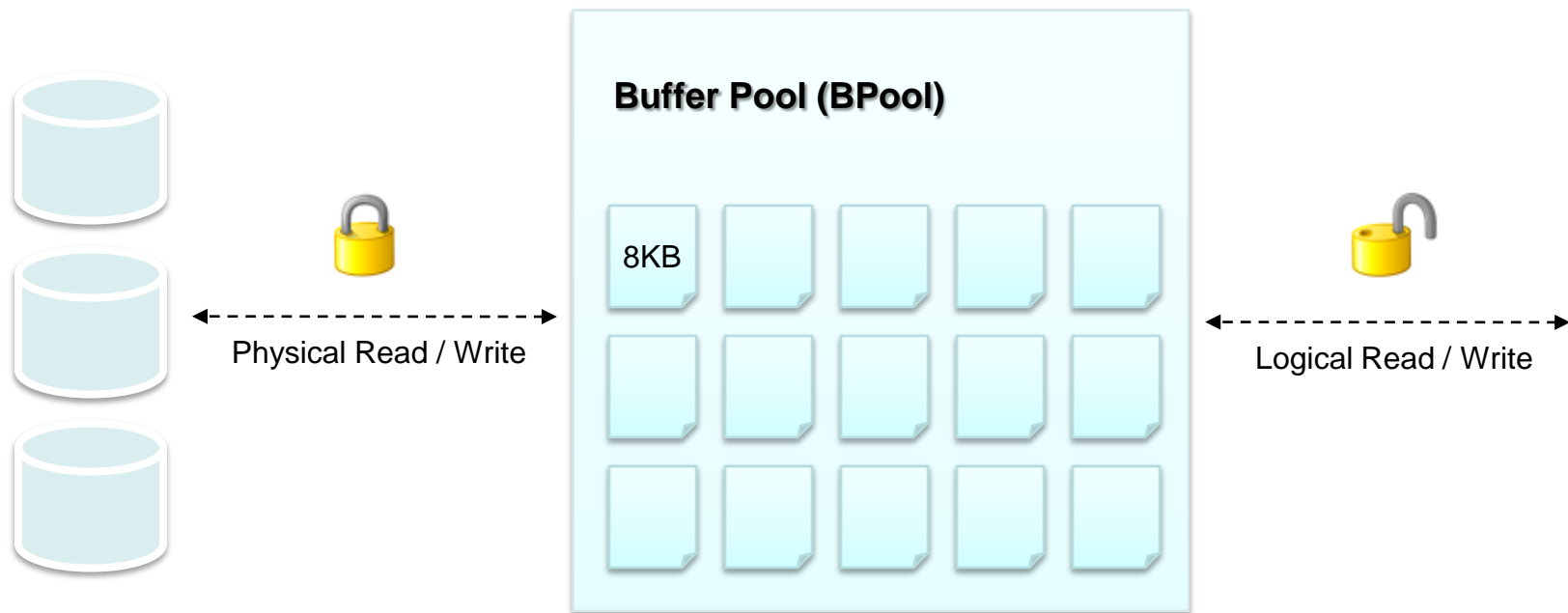


AdventureWorksTDE.mdf

048C0080	CE 27 71 63 40 B2 CE CA	10 39 8E 51 B1 B6 AA FC	I'qc@²îÊ.9 Q±¶³ü
048C0090	DF 23 D3 22 FE CA 13 CC	A7 D6 FF EE 09 67 05 D3	B#Ó"þÊ.ìSöÿi.g.Ó
048C00A0	B5 8A 70 17 DD A1 84 17	F7 67 12 56 21 D6 7F 60	µ p.Ýi .÷g.V!Ö.`
048C00B0	DD 04 C4 95 8B 04 98 85	52 5B 03 50 DA 7D 84 63	Ý.Ä .. R[.PÚ}lc
048C00C0	19 E4 3F 02 93 2E 95 29	F5 05 69 E6 EC 82 4B 9C	.ä?.)ö.iæi K



- Ver- und Entschlüsselung erfolgt “vor” dem Buffer Pool, also beim Ein- und Auslagern von Pages → transparent für höhere SQL-Ebenen
- Vollständige Featureunterstützung durch TDE (z.B. Range/Equal. Scans)



- Performance-Overhead liegt bei ca. 3-5% (vorwiegend CPU-Ressourcen)

```
DBCC DROPCLEANBUFFERS
SET STATISTICS TIME ON
SELECT SUM(OrderQty) FROM AdventureWorks.Sales.SalesOrderDetail
SET STATISTICS TIME OFF
```

SQL Server-Ausführungszeit: CPU-Zeit = 31 ms 

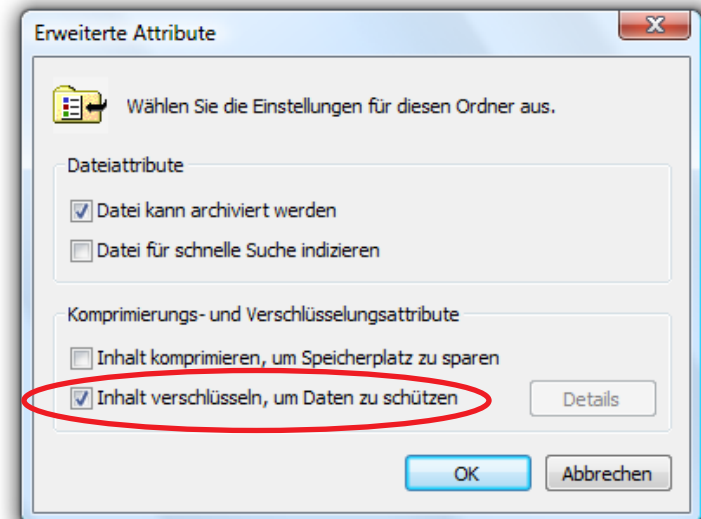
```
DBCC DROPCLEANBUFFERS
SET STATISTICS TIME ON
SELECT SUM(OrderQty) FROM AdventureWorksTDE.Sales.SalesOrderDetail
SET STATISTICS TIME OFF
```

SQL Server-Ausführungszeit: CPU-Zeit = 218 ms 

- EFS (verfügbar ab Windows 2000) verschlüsselt Dateien bzw. Ordner
- Aus Performancegründen keine wirkliche Alternative zu TDE:
Kein asynchrones I/O möglich, kein Read-Ahead (Prefetch)
- Verschlüsselung „außerhalb“ SQL Server, entschlüsselte Daten stehen jedem User mit Dateizugriffsrechten zur Verfügung
- Encrypting File System ggf. sinnvoll in mobilen und Desktop-Szenarien
- Selektive Ordnerverschlüsselung zum Beispiel “MSSQL\LOG” unter Windows 2003 ist ggf. eine Option

Encrypting File System - How It Works:

<http://technet.microsoft.com/magazine/cc160993.aspx>



- BitLocker Filtertreiber (fvevol.sys) ver- und entschlüsselt Daten transparent für Windows I/O-Management. Beim Start von Windows werden verschlüsselte Volumes initialisiert. Mögliche Szenarien:
 - Trusted Platform Module (TPM)
 - TPM und PIN
 - TPM und USB-Startup-Key
 - USB-Startup-Key (erfordert Änderung in Gruppenrichtlinie)
- WMI-Provider „Win32_EncryptableVolume“ und Skript „manage-bde.wsf“ steht für Befehlszeilen- und Remote-Administration zur Verfügung
- BitLocker kann temporär deaktiviert werden (über Systemsteuerung)
- BitLocker unterstützt keine Cluster-Konfigurationen

BitLocker-Laufwerkverschlüsselung (technische Übersicht):

<http://technet.microsoft.com/de-de/library/cc732774.aspx>



- BitLocker nutzt erweiterten AES-Algorithmus (AES-CBC + Elephant)
 - Filtertreiber benötigt CPU-Ressourcen für das Ciphering
- Beispielrechnung für die BitLocker-Performance mit einem Pentium 4:
 - Prozessor (CPU): 3GHz = 3.000.000.000 Cycles
 - Lesegeschwindigkeit: 50 MB/s = 52.428.800 Bytes
 - Verfügbare Cycles pro gelesenem Byte: 57,22
- Bitlocker-Overhead pro Byte bei ca. 25-30 Cycles
 - Overhead ist noch akzeptabel bei dieser CPU-Leistung
 - Impact auf I/O-Performance in Tests bei ca. 5%

White Paper von Niels Ferguson mit Details zur Implementierung:

<http://download.microsoft.com/download/0/2/3/0238acaf-d3bf-4a6d-b3d6-0a0be4bbb36e/BitLockerCipher200608.pdf>



Methoden	Vorteile	Nachteile	Szenarien
<u>Cell-Level</u>	<ul style="list-style-type: none"> ▪ Verschlüsselt bis in BPool ▪ Granulare Einrichtung ▪ Form der Zugriffskontrolle ▪ Verfügbar in allen Editionen 	<ul style="list-style-type: none"> ▪ Schema-Anpassungen ▪ Applikations-Anpassungen ▪ keine Indexes, Foreign-Keys ▪ Administrationsaufwand 	<ul style="list-style-type: none"> ▪ Spezifische Columns sind nur zu verschlüsseln ▪ Neuentwicklung Applikation ▪ Ggf. in Kombination mit TDE
<u>TDE</u>	<ul style="list-style-type: none"> ▪ Einfache Administration ▪ Hohe Performance ▪ Transparent (kein Redesign) ▪ Backups verschlüsselt 	<ul style="list-style-type: none"> ▪ Verfügbar nur in EE ▪ keine Systemdatenbanken ▪ TDE <-> "Zugriffskontrolle" ▪ Impact auf gesamte Instanz 	<ul style="list-style-type: none"> ▪ Schutz gegen "Offline-Attacken" und Backup-Sicherheit in wenigen Minuten
<u>EFS</u>	<ul style="list-style-type: none"> ▪ Alle Dateien möglich ▪ Granulare Einrichtung 	<ul style="list-style-type: none"> ▪ Performance unbrauchbar ▪ DBA <-> Windows-Admin 	<ul style="list-style-type: none"> ▪ Desktops, Notebooks ▪ SQL Server Express
<u>BitLocker</u>	<ul style="list-style-type: none"> ▪ Hohe Performance ▪ Einfache Administration 	<ul style="list-style-type: none"> ▪ Nur gesamtes Laufwerk ▪ "Unlock" beim Systemstart ▪ Windows Server 2008, Vista 	<ul style="list-style-type: none"> ▪ in Verbindung mit TDE, zusätzliche Absicherung von Dumps, Hibernation etc

Vielen Dank!

Jens Hansen
Principal Consultant, CEO

Hansen Group AG
Hansaallee 201, 40549 Düsseldorf

Phone: +49 211-5228973-0

Fax: +49 211-5228973-40

Cell: +49 163-510-610-5

E-Mail: j.hansen@hansen-group.de

