Online on-demand Project Support

# SQL Server Sicherheit & Verschlüsselung

## Für Entwickler


Thorsten Kansy (tkansy@dotnetconsulting.eu)

# Thorsten Kansy

Freier Consultant, Software Architekt,
Entwickler, Trainer & Fachautor

C# .NET
.NET Core
Visual Basic .NET
ASP.NET MVC
Microsoft .NET Windows Presentation Foundation
Microsoft SQL Server
TS
Microsoft SQL Server Analysis Services
Microsoft WPF
Microsoft SQL Server Integration Services
Microsoft SQL Server Reporting Services
Windows Azure

DOTNETCONSULTING
by Thorsten Kansy

# Motivation

- Übersicht
- Grundlagen
- Wichtiges

- Spaß

# Agenda

- Verbindung zum Server
- Sicherheit auf dem Server
- Verschlüsselung

- Sicherheit mit SQL Server 2016

# Verbindung zum Server

# Zugriff via Netzwerk

- Tabular Data Stream (TDS)-Protokoll
- Verschlüsselung möglich
  - SSL/ TLS

# TCP/IP

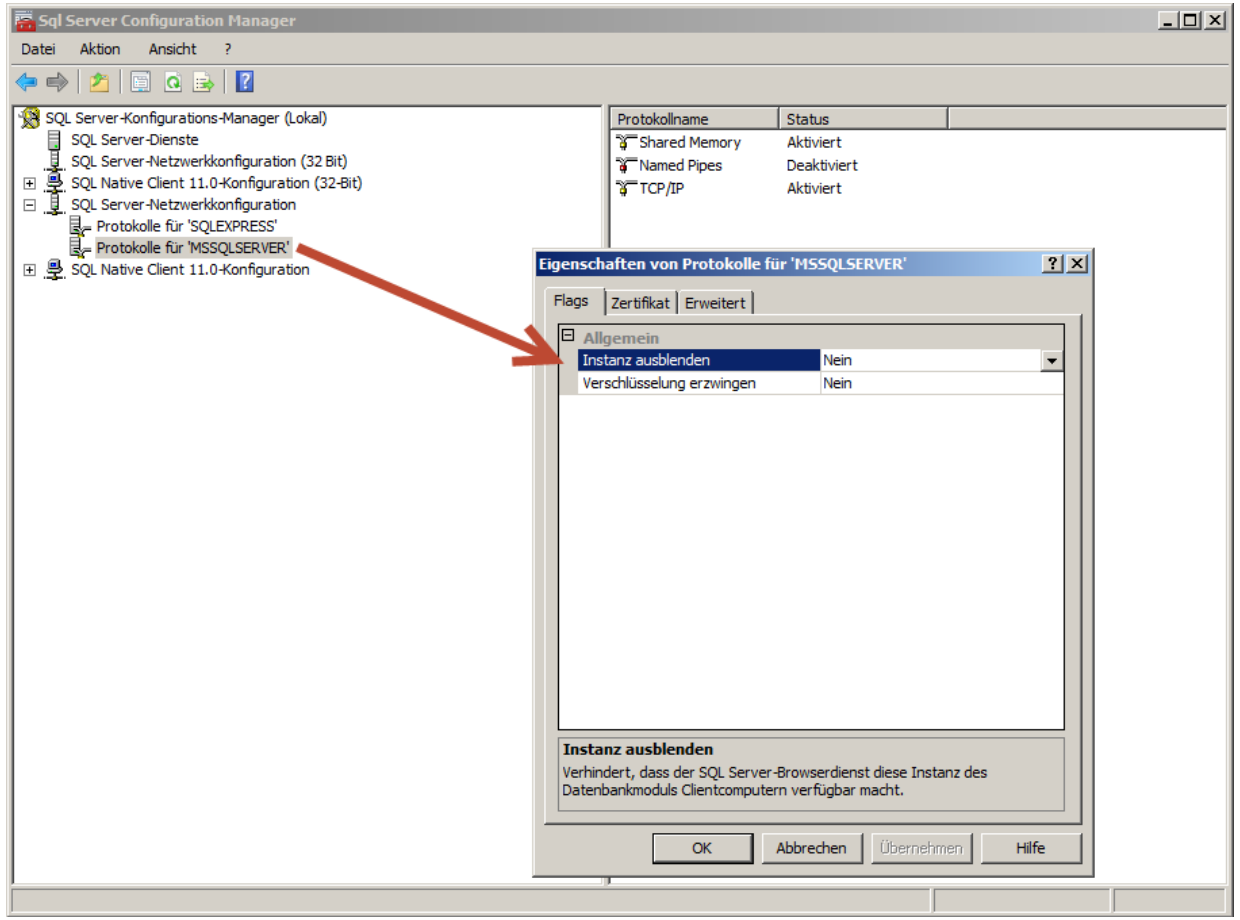
- Firewall
- Connection String „TCP:127.0.0.1,1433“

| Port | |
|---|---|
| 1433 TCP | Standard Instanz |
| 1434 UDP | SQL Server Browser |
| 2383 TCP | Analysis Service |
| 80 TCP | Reporting Service (HTTP) |
| 443 TCP | Reporting Service (HTTPS) |
| 4022 TCP | Service Broker |

# SQL Server Browser

- Windows Dienst
- Stellt Information zu bestehenden Instanzen bereit
  - Über das Netzwerk (UDP)
  - Für Verbindungsaufbau

# Instanz verstecken



# Demo

# SQL Server Sicherheit

# SQL Server Sicherheit

Authentifizierung

Authorisierung

Logins/ User

Rollen

Securables

Rechte

# Authentifizierung

- Login auf Serverebene
  - Windows Authentifizierung
  - SQL Server Authentifizierung
  - Active Directory
    - Universal mit MFA Support
    - Password
    - Integrated
- Erfolgreicher Anmeldung => Prozess/ Session

# Demo

# Logins und Users

- Logins auf Server Ebene
- User auf Datenbank Ebene
- Contained Databases
  - User OHNE Login

- Mapping von Login => User pro Datenbank
- Reiner Zugriff auf DB => CONNECT-Recht

# Authorisierung

- Authorisierung
  - Rechte & Verbote (negative Rechte)

- GRANT => Recht vergeben
- REVOKE => Recht entziehen
- DENY => Verbot

# Rollen

- „Gruppen“ für den SQL Server mit entsprechenden Rechten
- Server Rollen
- Datenbank Rollen
- Vordefinierte Rollen
  - Unveränderlich
- Benutzerdefinierte Rollen können erstellt werden
- Jeder ist Mitglied von PUBLIC

# Verschachtelte Rollen

- Rollen können andere Rolle als Mitglied beinhalten
- GUI zeigt dies ziemlich “unglücklich” an
- Mitgliedschaft kann abgefragt werden
  - `IS_MEMBER()`
  - `IS_SRVROLEMEMBER()`

# Vordefinierte Rollen

- Serverrollen
  - Administrative Aufgaben auf Serverebene
- Datenbankrollen
  - Rechte innerhalb einer Datenbank

# Vordefinierte Serverrollen

| Rolle | Berechtigungen |
|---|---|
| Bulkadmin | BULK INSERT-Anweisungen ausführen |
| DBCreator | Datenbanken erstellen, löschen, wiederherstellen |
| DiskAdmin | Datenbankmedien verwalten |
| ProcessAdmin | Prozesse verwalten |
| SecurityAdmin | Anmeldungen, Kennwörter und Berechtigungen verwalten |
| ServerAdmin | Server weit konfigurieren + Server stoppen |
| SetupAdmin | Verbindungsserver bearbeiten |
| SysAdmin | Alles auf dem SQL Server/ Instanz |

# Vordefinierte Datenbankrollen

| Rolle | Berechtigungen |
|---|---|
| db_Owner | Alles in der DB |
| db_SecurityAdmin | Rollenmitgliedschaften und Berechtigungen verändern |
| db_AccessAdmin | Zugriff auf Datenbank für Anmeldungen bearbeiten |
| db_BackupOperator | Backups der DB durchführen |
| db_DDLAdmin | Alle DDL Anweisungen ausführen |
| db_DataReader/ db_DataWriter | Aus allen (Benutzer-)Tabellen & Sicht lesen/ schreiben |
| db_DenyDataReader/ db_DenyDataWriter | Aus keiner (Benutzer-)Tabellen & Sicht lesen/ schreiben |

# Demo

Roles.sql

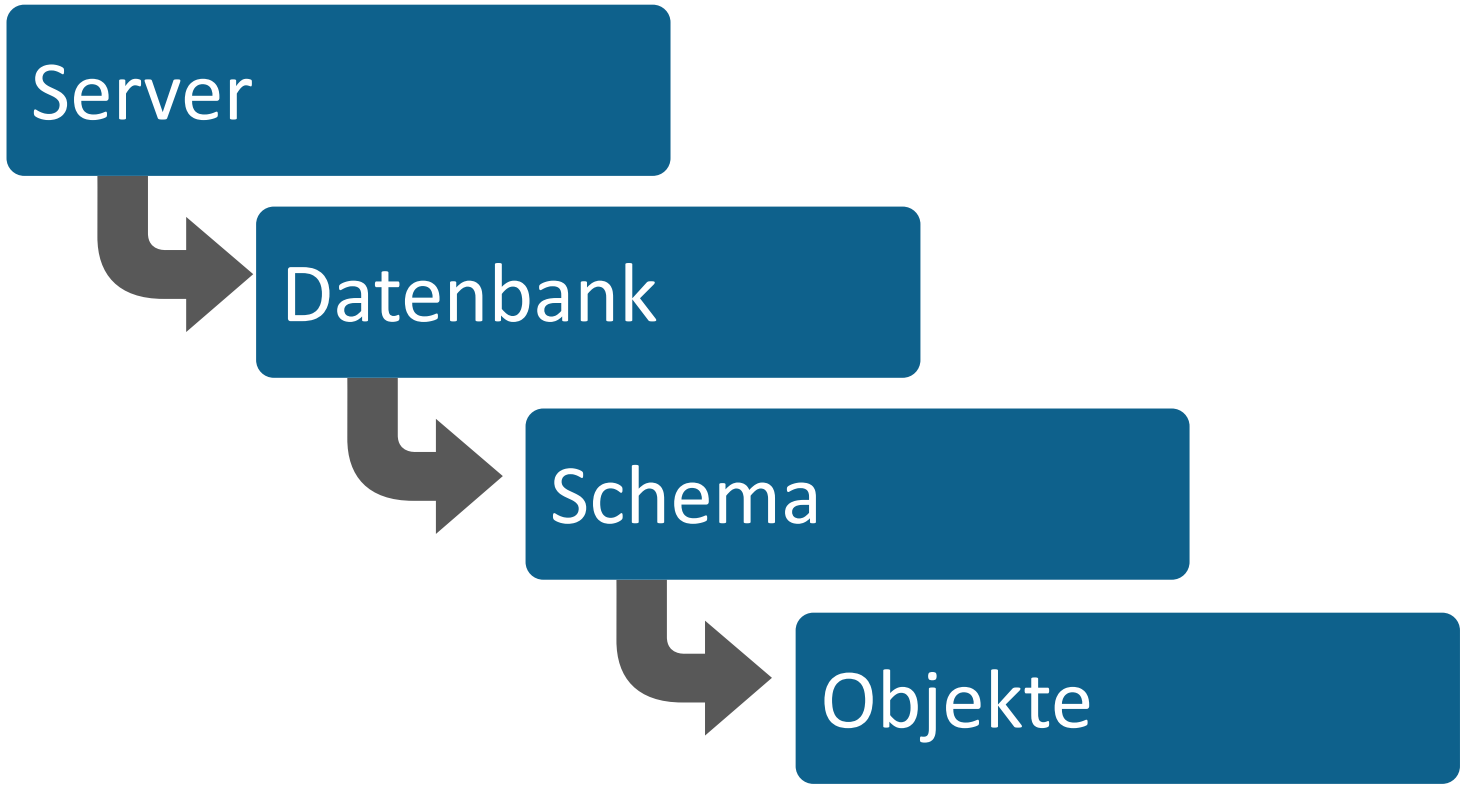
# Spezielle Benutzer

- dbo (db_owner)
- Guest

- Sys
- INFORMATION_SCHEMA

# Securables (Objekte, die gesichert werden können)

Scope



# Securables: Server

- Endpoint
- Login
- Datenbank

# Securables: Datenbank

- Anwendungsrollen
- Assembly
- Message Type
- Route
- Services
- Remote Service Binding
- Fulltext Catalog
- Zertifikate
- Asymmetric Key
- Symmetric Key
- Contract
- User
- Rollen
- Schemata

# Securables: Schemata

- Type
- XML Schema Collection
- Objekte

# Securables: Objekte

- Tabellen & Sichte
- Funktionen & Prozeudren
- Aggregate
- Queue
- Synonym
- …

# Demo

# Vergabe von Rechten

- Positive Rechte (`GRANT`)
- Negative Rechte (`DENY`)
  - Verbote haben immer höchste Priorität
- DML-Rechte (`SELECT`, `INSERT`, `DELETE`, `UPDATE`)
- DDL-Rechte (`CREATE`, `DROP`, `ALTER`, …)

# Demo

GrantRevokeDeny.sql

# Besitzer

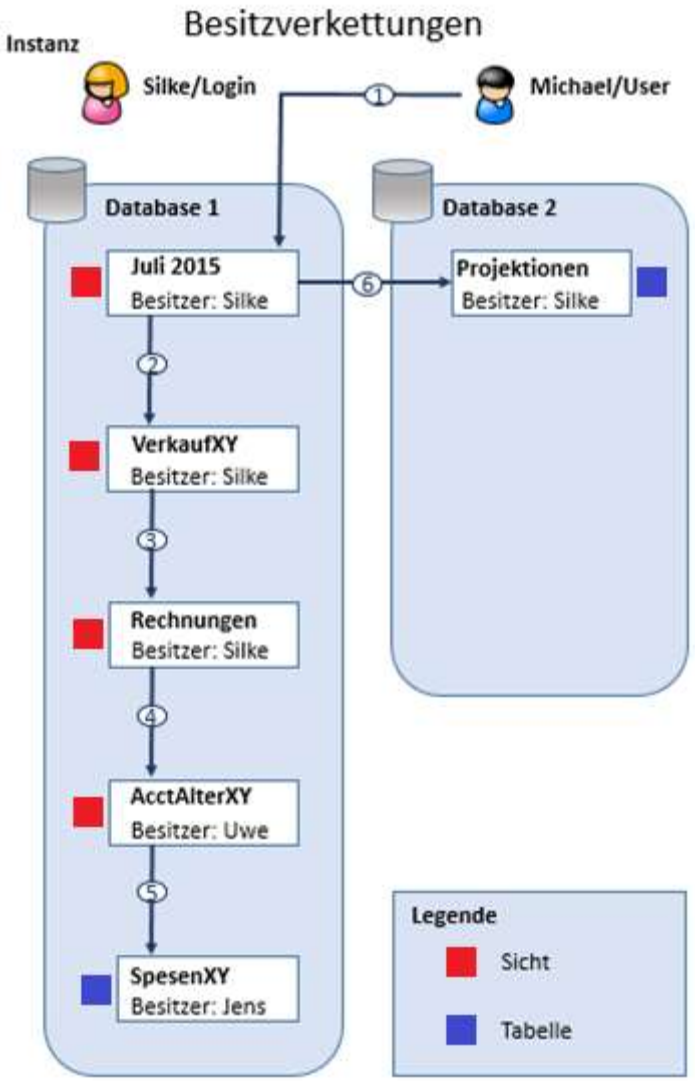
- Der Besitzer eines Objektes darf alles
- Ersteller ist Standard mässig Besitzer
- Kann, aber sollte **NICHT** verändert werden

# Demo

Query Owner.sql

# Besitzerverkettung

Zugriff auf Objekte **ohne** Rechte auf Basisobjekte solange der Besitzer der **gleiche** ist



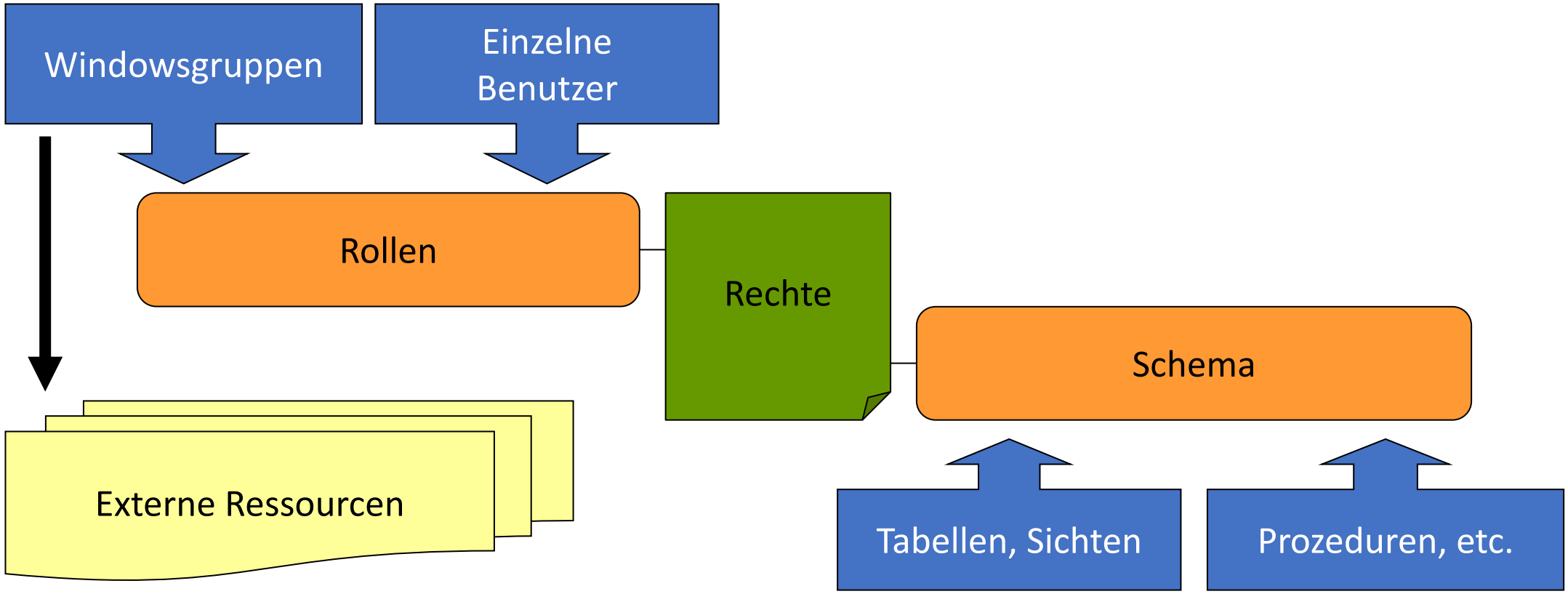
# Demo

ALTER AUTHORIZATION.sql

# Datenbank übergreifende Besitzerverkettung

- Standardmäßig deaktivert
- Sollte es auch besser bleiben!

# Best Pratices



# (Ungebetene) Gäste

- Für User ohne eigene Logins
- GUEST deaktivert lassen !
  - Wirklich!

# EXECUTE AS

- User für bestimmen
  - Stored Procedure, Funktionen
  - Trigger
- `CALLER` | `SELF` | `OWNER` | ‚login_name/ user_name'
- Standard => Caller

| | |
|---|---|
| CALLER | Der Aufrufer |
| SELF | Besitzer des Objektes |
| OWNER | Ersteller des Objektes |

# Demo

ExecuteAs.sql

# Anwendungsrollen

- Eine Anwendung kann andere Rechte als einem Benutzer haben
- Besteht aus Name und Kennwort
- Kann Berechtigungen wie „gewöhnliche“ Rollen haben
- Nach Aktivierung kein DB Wechsel möglich

# Anwendungsrollen

```
sp_setapprole
    @rolename = 'role',
    @password = 'password',
    @fCreateCookie = true,
    @cookie OUTPUT;


sp_unsetapprole @cookie;
```

# Demo

ApplicationRole.sql, ApplicationRole.cs

# Verschlüsslung und Hashing

# Verschlüsselung

- Symmetrische Schlüssel
- Asymmetrische Schlüssel
- Zertifkate
  - Selbst erstellen
  - Zertifizierungsstelle (CA)
- Hash erstellen

# Wer überhaupt hat Zugriff?

- Anwender muss Berechtigung haben auf
  - Schlüssel
  - Zertifikat
- Ggf auch das Password kennen

Infrastruktur: PKI (Public Key Infrastructure) 

# Demo

SymetricEncryptData.sql

# Demo

AsymetricEncryptData.sql

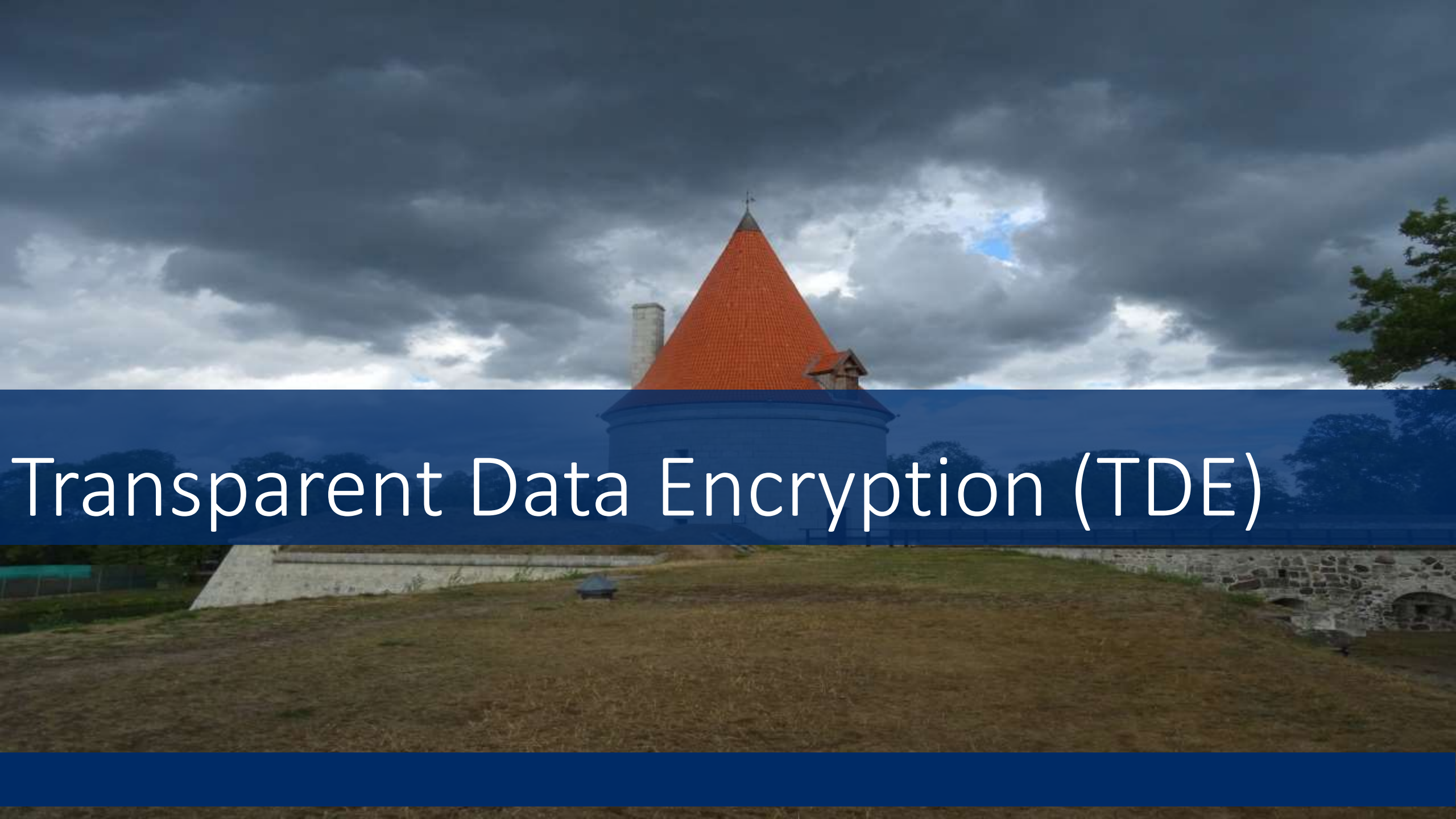
# Hash erzeugen

- MD2, MD4, MD5, SHA, SHA1
- MD (Message-Digest)
- SHA (Secure Hash Algorithm)

- Max 8.000 Bytes Input
  - MD => 16 Bytes Output
  - SHA => 20 Bytes Output

# Demo

HashBytes.sql



# Transparent Data Encryption (TDE)

# Transparent Data Encryption (TDE)

Verschlüsselung von

- Datenbank-Medien
- Backups

Zertifikate und Private Keys müssen sicher aufgehoben werden 

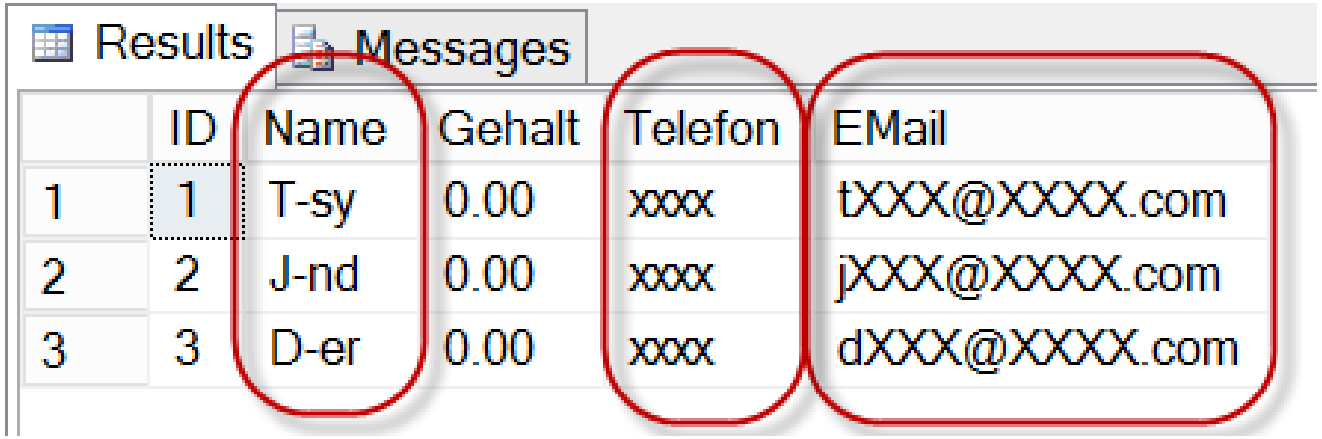
# Demo

TDE.sql

# SQL Server 2016+

# Dynamic Data Masking

- Maskierung sensibler Daten
  - Sortierung/ Filtern trotzdem korrekt
- Demaskierung via Recht pro Datenbank !



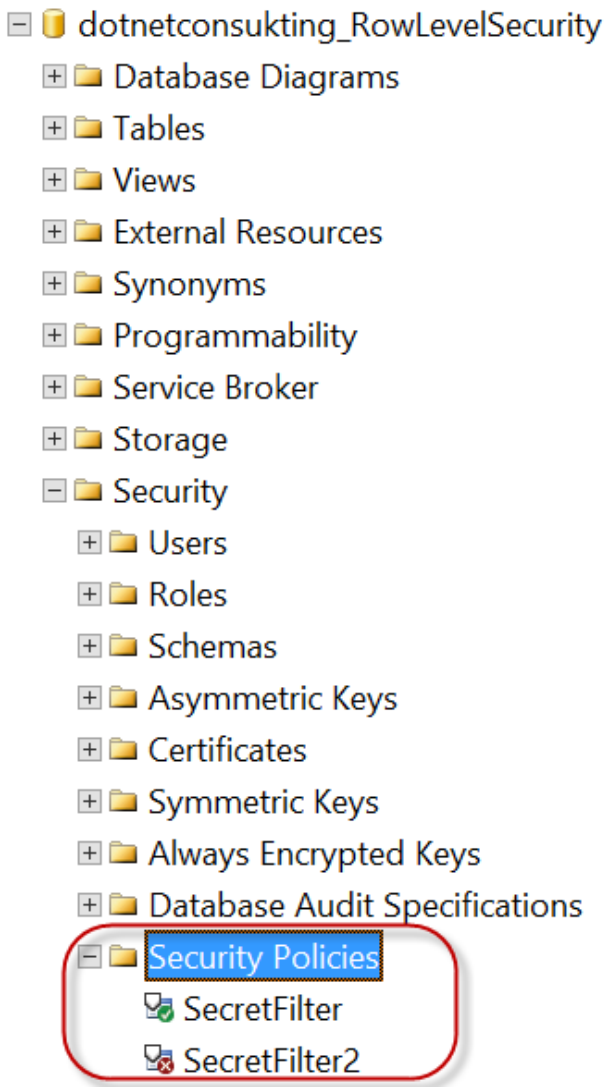
# Demo

Dynamic Data Masking.sql

# Row Level Security

Zeilenbasierte Berechtigung durch eigene Filterfunktion

Performance !



# Demo

Row Level Security.sql

# Always Encrypted

Daten werden in der Datenbank und im Netzwerk verschlüsselt

- Ab .NET 4.6
- Nur Änderungen im ConnectionString notwendig
  - `…;Initial Catalog=dotnetconsulting_AlwaysEncrypted;…`
- PKI wird benötigt !

# Demo

Alyways Encrypted.sql, dotnetconsulting.AlwaysEncrypted.sln

# Fragen?

# Links

http://dotnetconsulting.eu/blog/

@Tkansy

tkansy@dotnetconsulting.eu

www.dotnetconsulting.eu

DOTNETCONSULTING
by Thorsten Kansy