Complementos de Bases de Dados

– Segurança –

(*MS SS*)

Engenharia Informática

2º Ano / 1º Semestre

Cláudio Miguel Sapateiro
claudio.sapateiro@estsetubal.ips.pt

DSI :: Escola Superior de Tecnologia de Setúbal :: Instituto Politécnico de Setúbal

Sumário

- Introdução
- Modalidades de Autenticação
- Autorização: Permissões e *Roles*
- Hierarquia de *Securables*
- Hierarquia de Permissões
- Discussão/Distinção de Conceitos

- Encriptação: Conceitos e Aplicação

Motivação

Proteger …

i.e. manter operacional e de acesso controlado

- Dados
- Objetos
- Regras/Lógica
- BD(s)
- SGBD

Introdução

Várias Dimensões da Segurança

- Segurança Física (espaço/localizações, acessos,…)
- Segurança da Rede (firewall,…)
- Modalidades de Autenticação (*MSSS mode*, *Windows mode*, ….)
- Esquemas de Autorização (Privilégios e *Roles*)
- Opções na definição e atribuição de (tipo de) contas
- Auditoria/Monitorização regular e pró-ativa
- …

Introdução

Várias Dimensões da Segurança II

- Politica de Backups (*offsite*, regularidade, cobertura, …)
- Validação de dados submetidos pelo(s) “utilizador(es)”

  (e modalidade, e.g. via sp’s)
- Ponderação das alternativas *exequíveis* de encriptação
- Desativação de serviços desnecessários
- Eventual alteração de portos *default (e.g. DBA backdoor 1434)*
- …

Autenticação

Definição

- Processo de verificação de que um utilizador é genuíno

- Modos de Autenticação MSSS (Login):
  - **Windows**
    Modo integrado com o SO, as contas e grupos do SO são confiáveis na autenticação perante o MSSS
    - Adequado quando existe um *domain controller*
    - Ou, quando a aplicação e a BD correm na mesma maquina

  - **Misto (*Mixed*)**
    Modo Windows +
    Credenciais (utilizador + senha) definidos e armazenados no MSSS
    - Utilização típica quando as aplicações se ligam a partir de outros domínios (e e.g. Internet)

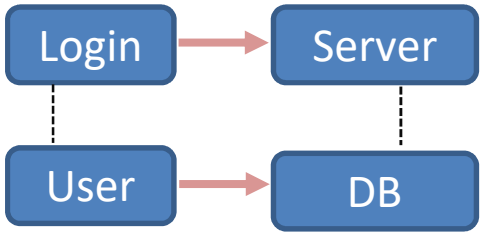
Autorização

Definição

- Processo que especifica os direitos de acesso de determinado utilizador/grupo relativamente aos objetos da BD
  - *Privileges & Roles*
    São especificados os privilégios sobre os objetos (sistema + específicos) da BD

➢ *Logins (acesso ao servidor)*
vs Users (operações na BD)

- Faculta o acesso ao MSSS
  (estabelece a relação entre nível de acesso ao servidor e o nível de utilizador da BD)
- Podem estar associados a contas Windows ou MSSS
- São armazenados na BD *master*



Roles

Definição

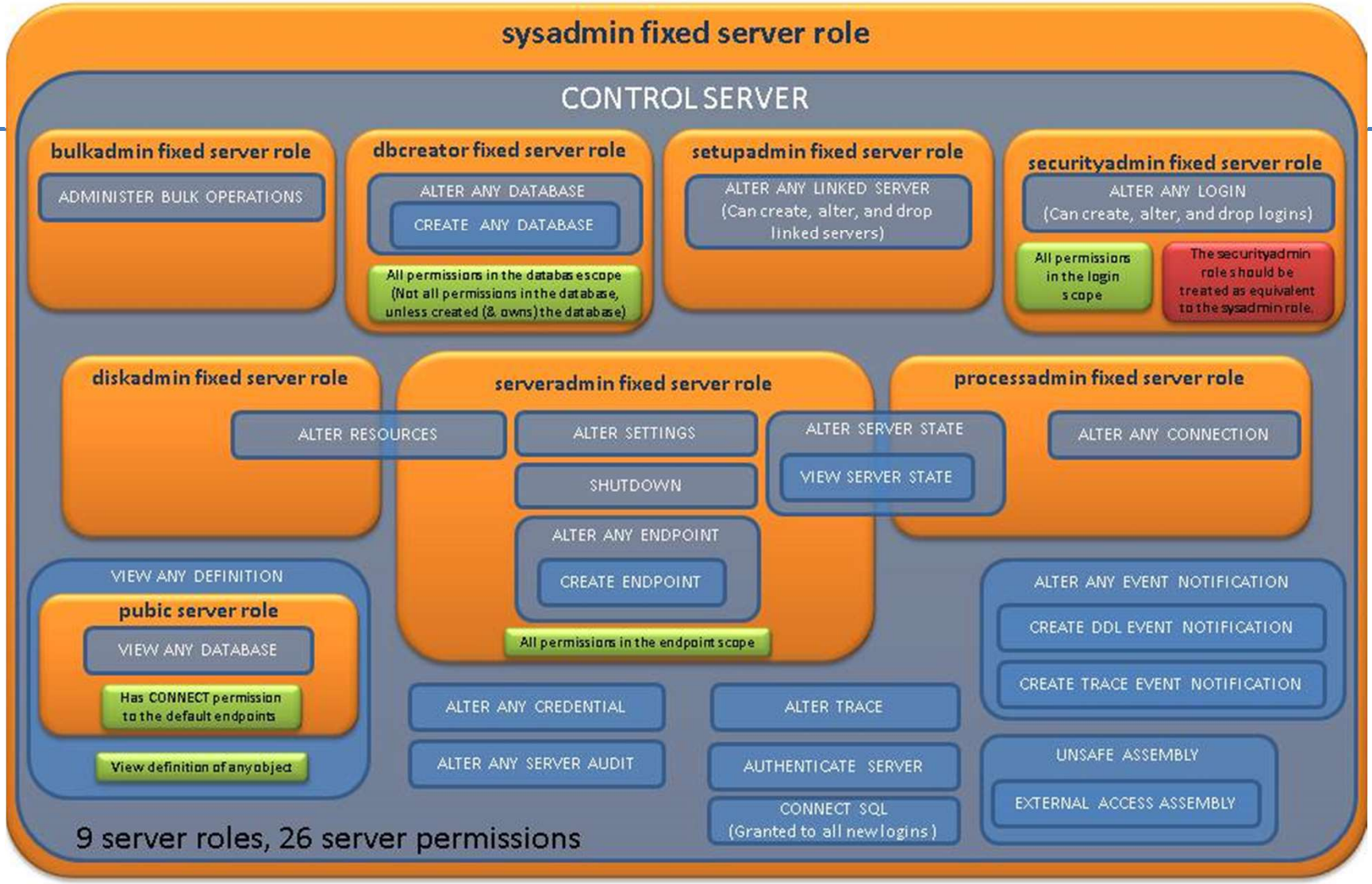
- Modelo flexível de administração de segurança
- Conjunto agregado de privilégios
- Podem ser geridos/alterados "dinamicamente"
- Preferível a atribuição de permissões/privilégios a *Roles* do que a *users*
- Um *user* pode pertencer a múltiplos *roles*

- Disponíveis no MSSS:
  - ***Fixed Server Roles*** *(Dbcreator, Diskadmin, Sysadmin, …)*
  - ***Custom Server Roles*** *(since 2012)*
  - ***Fixed Database Roles*** [Criados com cada BD]
    (Db_owner, Db_denydatareader/writer, Db_datareader/writer,...)
  - ***Custom Database Roles*** » *Mais granular*

Server Level

DB Level

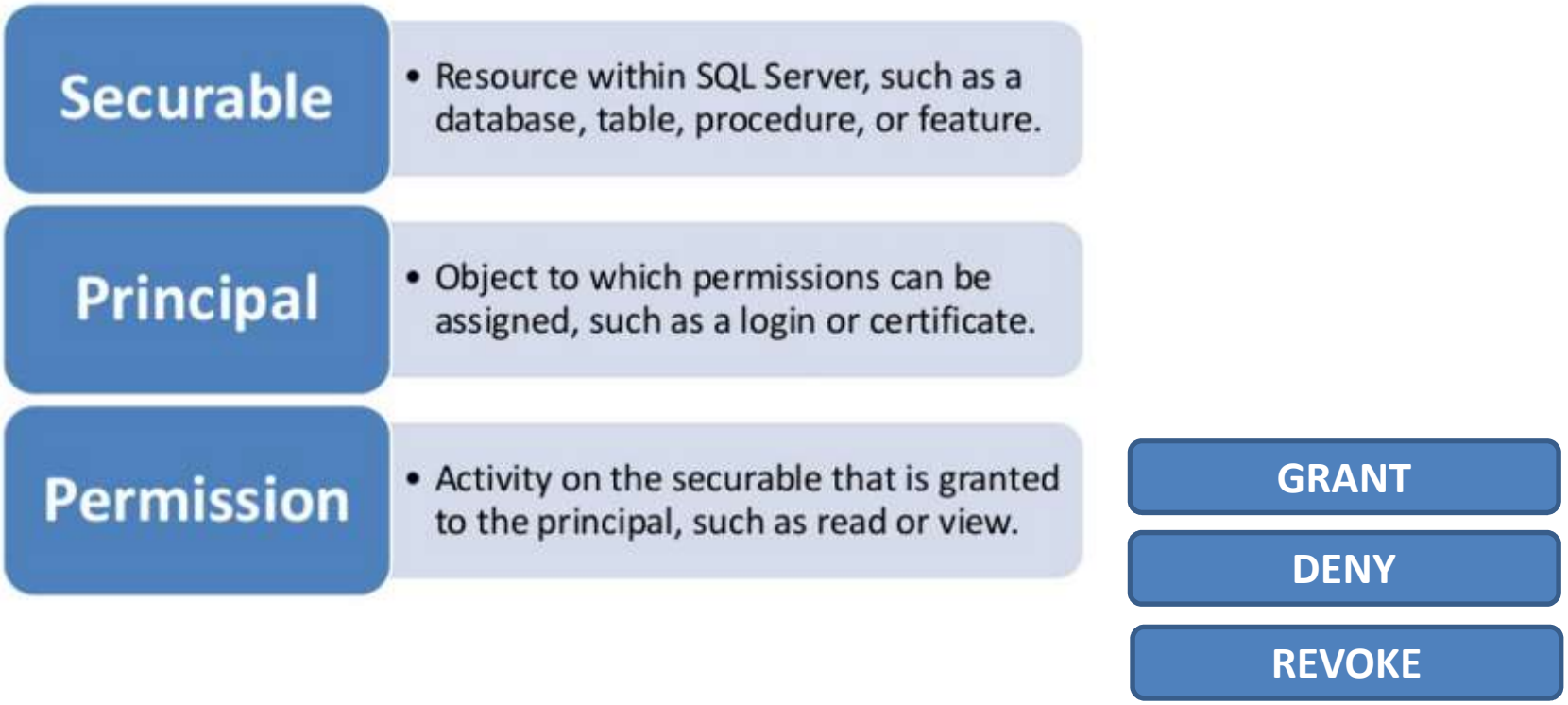
SERVER LEVEL ROLES AND PERMISSIONS



https://msdn.microsoft.com/en-us/library/ms188659.aspx

Permissões no MSSS

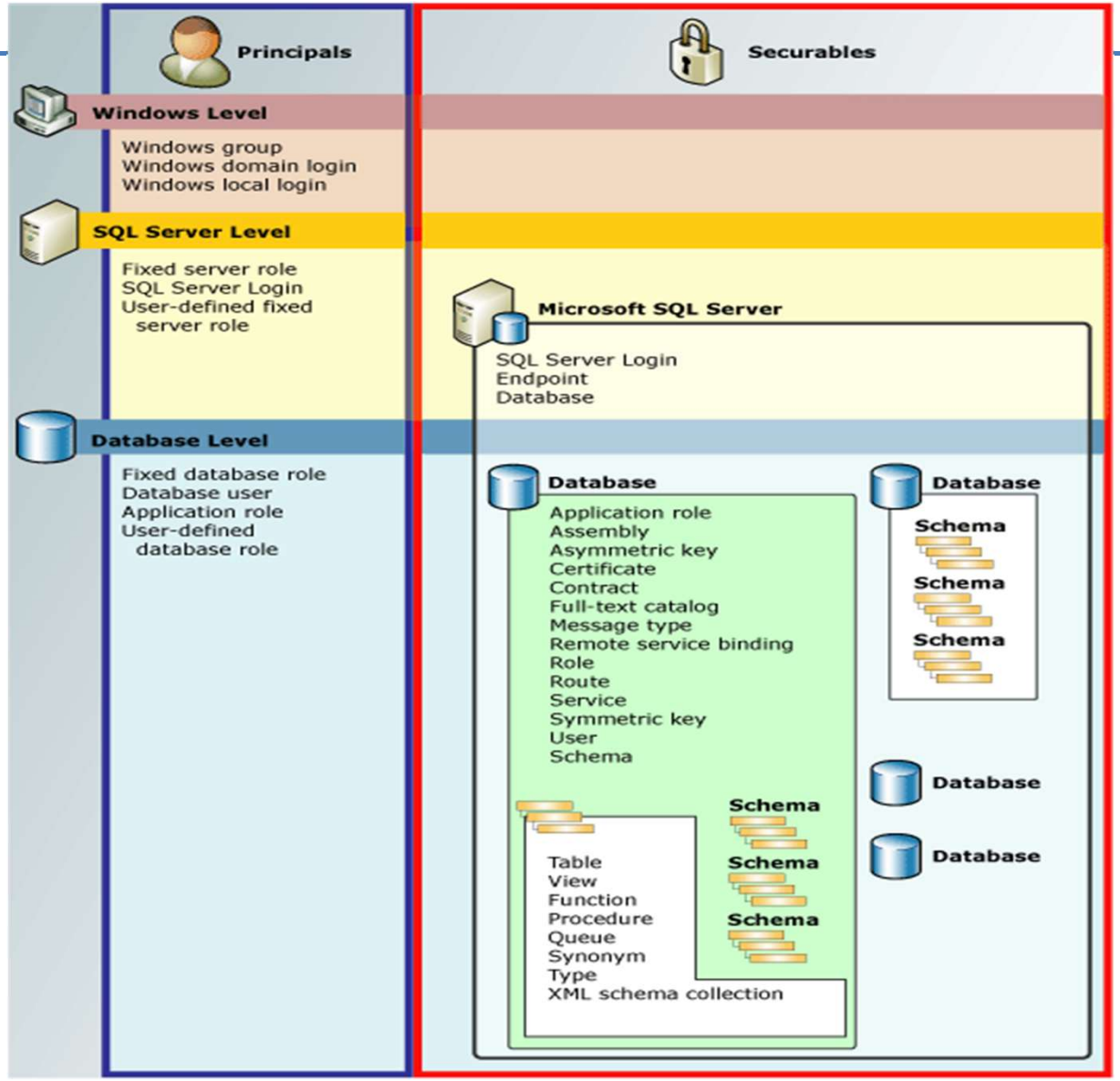
Vocabulário



Permissões no MSSS

- **GRANT**
  - Confere permissão ao *Principal* sobre o *Securable* em determinada acção
- **DENY**
  - Nega determinada acção ao *Principal* sobre o *Securable*
  - Se houver conflito com existência de GRANT e DENY simultaneamente,
    - o DENY sobrepõe-se sempre
- **REVOKE**
  - Repõe no estado anterior à ultima “permissão” concedida
    - Seja esta GRANT ou DENY

Hierarquia de *Securables*



Exemplo

```
USE TestDB;
GO
CREATE ROLE TestRole;
GO
CREATE USER TestUser WITHOUT LOGIN;
GO
EXEC sp_addrolemember @rolename = 'TestRole', @membername = 'TestUser';
GO
```

```
CREATE SCHEMA Test;
GO
CREATE TABLE Test.TestTable (TableID int);
GO
GRANT SELECT ON OBJECT::Test.TestTable TO TestRole;
GO
CREATE TABLE Test.TestTable2 (TableID int);
GO
```

Exemplo

```
EXECUTE AS USER = 'TestUser';
GO
SELECT * FROM Test.TestTable;
GO
REVERT;
GO
-- Test Harness to verify how permissions work for Test.TestTable2.
EXECUTE AS USER = 'TestUser';
GO
-- This should fail initially, as there is no permission for this table
SELECT * FROM Test.TestTable2;
GO
REVERT;
GO
```

```
-- Let's undo the permission using REVOKE;
REVOKE SELECT ON OBJECT::Test.TestTable FROM TestRole;
```

Exemplo

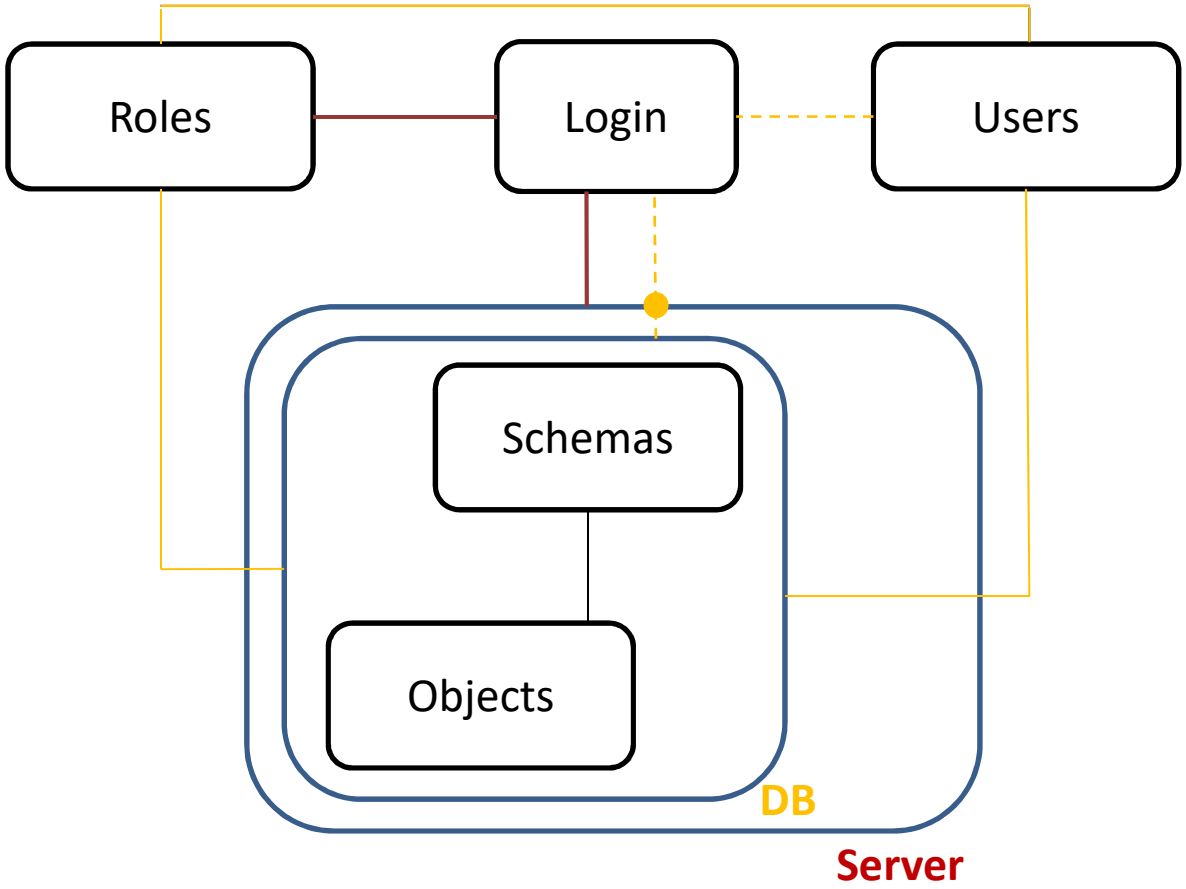
```
-- Permission at the schema level
GRANT SELECT ON SCHEMA::Test TO TestRole;
GO
```

```
-- Specific DENY will block the GRANT
DENY SELECT ON OBJECT::Test.TestTable TO TestUser;
```

» Agora será possível ao TestUser fazer o SELECT à TestTtable2 mas não à TestTable

Discussão

Como se articulam então os vários conceitos?



Related Catalog Views

Database-Level Views

sys.database_permissions	sys.database_scoped_credentials
sys.database_principals	sys.master_key_passwords
sys.database_role_members	sys.user_token

Server-Level Views

sys.credentials	sys.server_principals
sys.login_token	sys.server_role_members
sys.securable_classes	sys.sql_logins
sys.server_permissions	sys.system_components_surface_area_configuration

Exemplo

```
select distinct
       spr.principal_id,
       spr.name as principal_name,
       spr.type as principal_type,
       spr.type_desc as principal_type_desc,
       spm.permission_name,
       spm.state_desc
  from sys.server_principals spr
  inner join sys.server_permissions spm
  on spr.principal_id = spm.grantee_principal_id
  where spr.type in ('s', 'u','r')
```

principal_id	principal_name	principal_type	principal_type_desc	permission_name	state_desc
1	sa	S	SQL_LOGIN	CONNECT SQL	GRANT
2	public	R	SERVER_ROLE	CONNECT	GRANT
2	public	R	SERVER_ROLE	VIEW ANY DATABASE	GRANT
257	##MS_PolicyTsqlExecutionLogin##	S	SQL_LOGIN	CONNECT SQL	GRANT
257	##MS_PolicyTsqlExecutionLogin##	S	SQL_LOGIN	VIEW ANY DEFINITION	GRANT
257	##MS_PolicyTsqlExecutionLogin##	S	SQL_LOGIN	VIEW SERVER STATE	GRANT
259	Claudio-PC\Claudio	U	WINDOWS_LOGIN	CONNECT SQL	GRANT
260	NT SERVICE\SQLWriter	U	WINDOWS_LOGIN	CONNECT SQL	GRANT
261	NT SERVICE\Winmgmt	U	WINDOWS_LOGIN	CONNECT SQL	GRANT
262	NT Service\MSSQLSERVER	U	WINDOWS_LOGIN	CONNECT SQL	GRANT
263	NT AUTHORITY\SYSTEM	U	WINDOWS_LOGIN	ALTER ANY AVAILABILITY GROUP	GRANT
263	NT AUTHORITY\SYSTEM	U	WINDOWS_LOGIN	CONNECT SQL	GRANT
263	NT AUTHORITY\SYSTEM	U	WINDOWS_LOGIN	VIEW SERVER STATE	GRANT
264	NT SERVICE\SQLSERVERAGENT	U	WINDOWS_LOGIN	CONNECT SQL	GRANT
265	NT SERVICE\ReportServer	U	WINDOWS_LOGIN	CONNECT SQL	GRANT
266	##MS_PolicyEventProcessingLogin##	S	SQL_LOGIN	CONNECT SQL	GRANT
268	distributor_admin	S	SQL_LOGIN	CONNECT SQL	GRANT

mini Sumário

1. Autenticação vs autorização
2. Principals, Securables & Permissions
3. Users and Logins

Exercícios

Responda aos exercícios que serão apresentados no final da Demo

DEMO

ENCRIPTAÇÃO

Encriptação

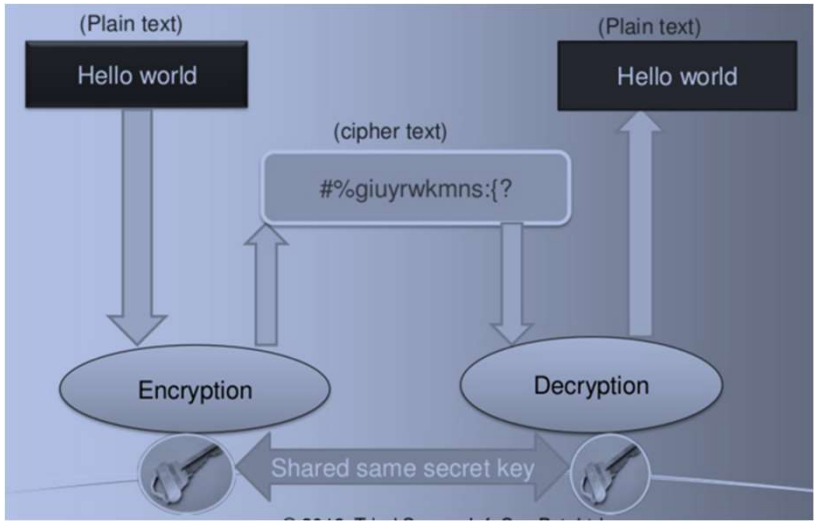
Introdução

- Uma forma adicional de proteger dados
- Adequado para informação particularmente sensível (email, password, nº cartão de crédito, …)
- Pode ser considerada uma hierarquia:
  - Nível SO (e.g. Windows BitLocker)
  - Nível SGBD
  - Nível BD
- e adicionalmente considerar também a encriptação na transmissão dos dados (SSL)

Encriptação

Modos de Encriptação

- **Chaves Simétricas**
  o emissor e recetor partilham uma única e comum chave usada para encriptar e desencriptar os dados
  - Mais simples e preformante
  - Mas, a chave tem de ser transmitida!
  - e.g. DES, 3DES, AES



Encriptação

Modos de Encriptação

- **Chaves Assimétricas**
  Utilizam-se duas chaves uma para encriptar e outra para desencriptar

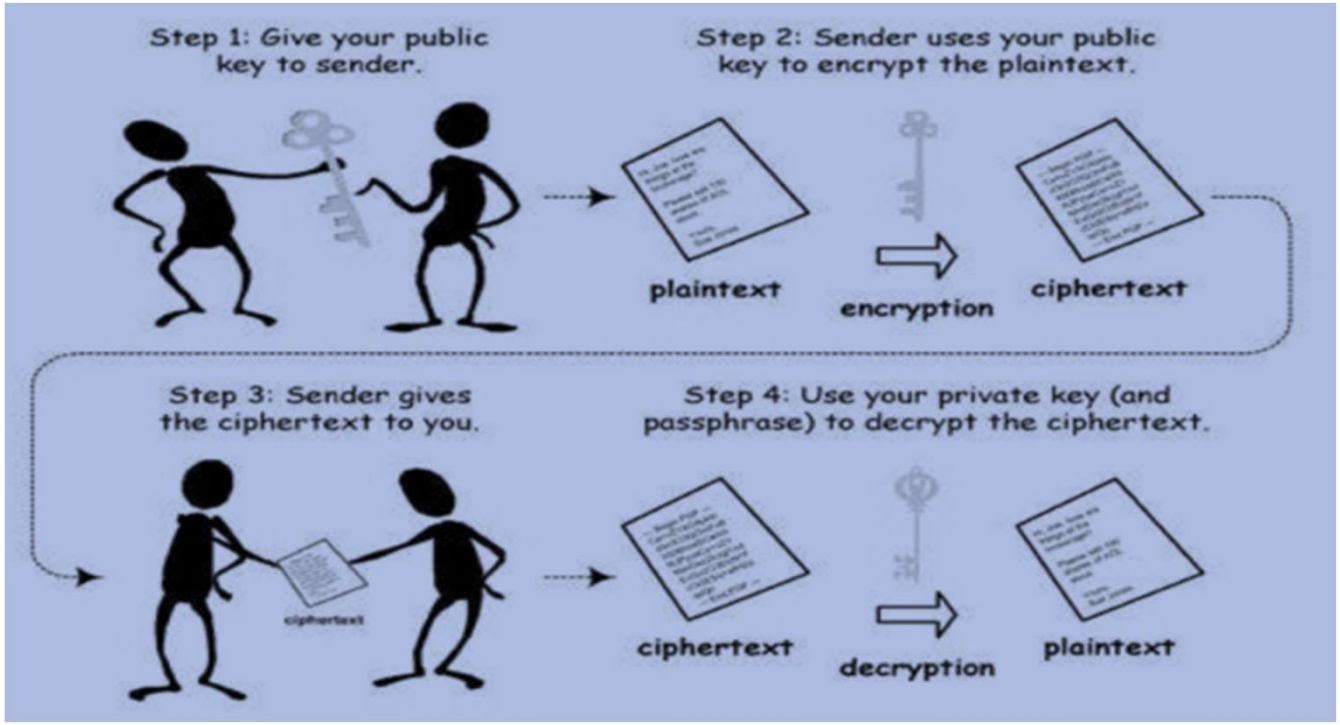
- O par é gerado conjuntamente e é constituído por:
  - uma chave *public-key* que será distribuída
  - uma chave *private-key*

- O emissor utiliza a chave publica para realizar a encriptação o recetor utiliza a sua chave privada correspondente para desencriptar
  - Assim a chave publica não permite a desencriptação !

Encriptação

Modos de Encriptação

- **Chaves Assimétricas**



Encriptação

Modos de Encriptação

Chaves Assimétricas

- **Exemplos algoritmos**
  - RSA, DSA (w/ SHA-1), …
- **Notas**
  - Mais poderoso
  - Mais complexo
  - Menos preformante!

➢ Tipicamente utiliza-se modo assimétrico para encriptar chaves do modo simétrico, estas ultimas usadas então na encriptação dos dados

Encriptação

Desempenho

Symmetric Key

Results | Messages

```
(5000 row(s) affected)

(5000 row(s) affected)
Symmetric Key Time Difference (ms): 550
```

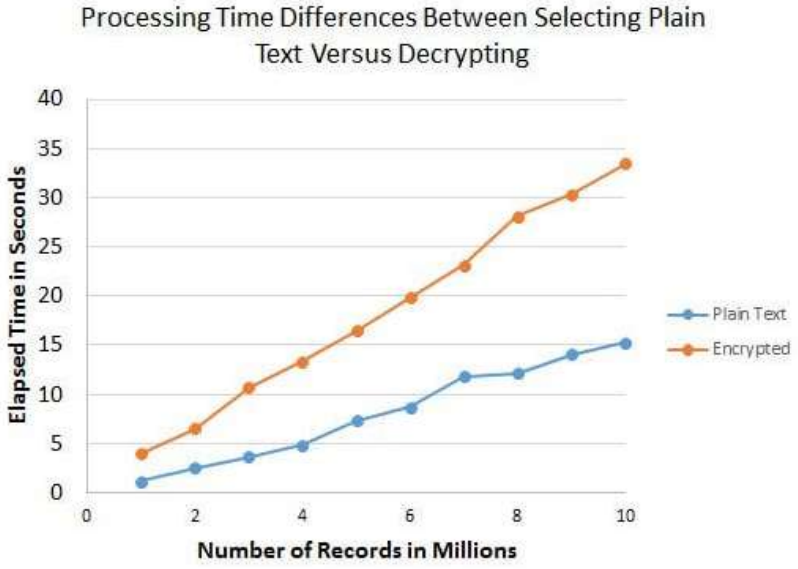
Asymmetric Key

Results | Messages

```
(5000 row(s) affected)

(5000 row(s) affected)
Asymmetric Key Time Difference (ms): 7466
```

Texto livre *vs* encriptado



https://www.mssqltips.com

Encriptação

MS SQL

- **Encriptação ao nível da BD** (*MS SQL Enterprise – TDE*)
  Possibilita a encriptação da totalidade dos dados (*data files*)
  - Ler e Escrever precisa do processo de desencriptação/encriptação
  - ➢ Degradação da performance

- **Encriptação ao nível da(s) coluna(s)**
  Somente colunas que persistem dados sensíveis são encriptadas
  - Melhor performance, mas
    » Atenção se a(s) coluna(s) em causa são chaves, ou alvo das clausulas *where* ou *Join on* de *queries*; degradará a performance dessas *queries*
  - ✓ Mais comumente utilizado

Encriptação

MS SQL

Exemplo

```
/* Create Database  */
USE master
GO
CREATE DATABASE EncryptTest
ON PRIMARY ( NAME = N'EncryptTest', FILENAME = N'C:\EncryptTest.mdf')
LOG ON ( NAME = N'EncryptTest_log', FILENAME =
N'C:\EncryptTest_log.ldf')
GO
```

```
CREATE TABLE TestTable (FirstCol INT, SecondCol VARCHAR(50))
```

TestTable

	FirstCol	SecondCol
1	1	First
2	2	Second
3	3	Third
4	4	Fourth
5	5	Fifth

Encriptação

MS SQL

Exemplo

Hierarquia

- DB MasterKey
  utilizada para proteger as *asymmetric-key* e certificados
  (rely on 3DES/AES w/ user-provided password)
  (DMK not directly encripting data, but keys!)
  (stored on both DB and master)
- Certificate (ou *asymmetric key*)
  Objeto (modo assimétrico) digitalmente assinado por autoridade emissora confiável que é utilizado para encriptar chaves ou dados
- Symetric Key || Data

```
/* Create Database Master Key */
USE EncryptTest
GO
CREATE MASTER KEY ENCRYPTION
BY PASSWORD =    'Pwd123'
GO
```

Self-signed MS SQL certificates

```
/* Create Encryption Certificate */
USE EncryptTest
GO
CREATE CERTIFICATE EncryptTestCert
WITH SUBJECT =   'Protect Data'
GO
```

Encriptação

MS SQL

Exemplo

```
/* Create Symmetric Key */
USE EncryptTest
GO
CREATE SYMMETRIC KEY TestTableKey
WITH ALGORITHM = TRIPLE_DES ENCRYPTION
BY CERTIFICATE EncryptTestCert
GO
```

```
/*  Encrypt Data using Key and Certificate
Add Columns which will hold the encrypted data in binary */
USE EncryptTest
GO
ALTER TABLE TestTable
ADD EncryptSecondCol VARBINARY(256)
GO
```

```
/* Update binary column with encrypted data created by certificate and key
USE EncryptTest
GO
OPEN SYMMETRIC KEY TestTableKey DECRYPTION
BY CERTIFICATE EncryptTestCert
UPDATE TestTable
SET                          EncryptSecondCol                          =
ENCRYPTBYKEY(KEY_GUID('TestTableKey'),SecondCol)
GO
```

1

2

Encriptação

MS SQL

Exemplo

```
/* DROP original column which was encrypted for protect the data */
USE EncryptTest
GO
ALTER TABLE TestTable
DROP COLUMN SecondCol
GO
```

```
/* Check the content of the TestTable */
USE EncryptTest
GO
SELECT *
FROM TestTable
GO
```

	FirstCol	EncryptSecondCol
1	1	0x003B444CCFA3B040AEE7FF980C76B82301000000CEFE6F...
2	2	0x003B444CCFA3B040AEE7FF980C76B823010000007C11369...
3	3	0x003B444CCFA3B040AEE7FF980C76B82301000000BA407B...
4	4	0x003B444CCFA3B040AEE7FF980C76B8230100000004623AC...
5	5	0x003B444CCFA3B040AEE7FF980C76B8230100000002886EB...

Encriptação

MS SQL

Exemplo

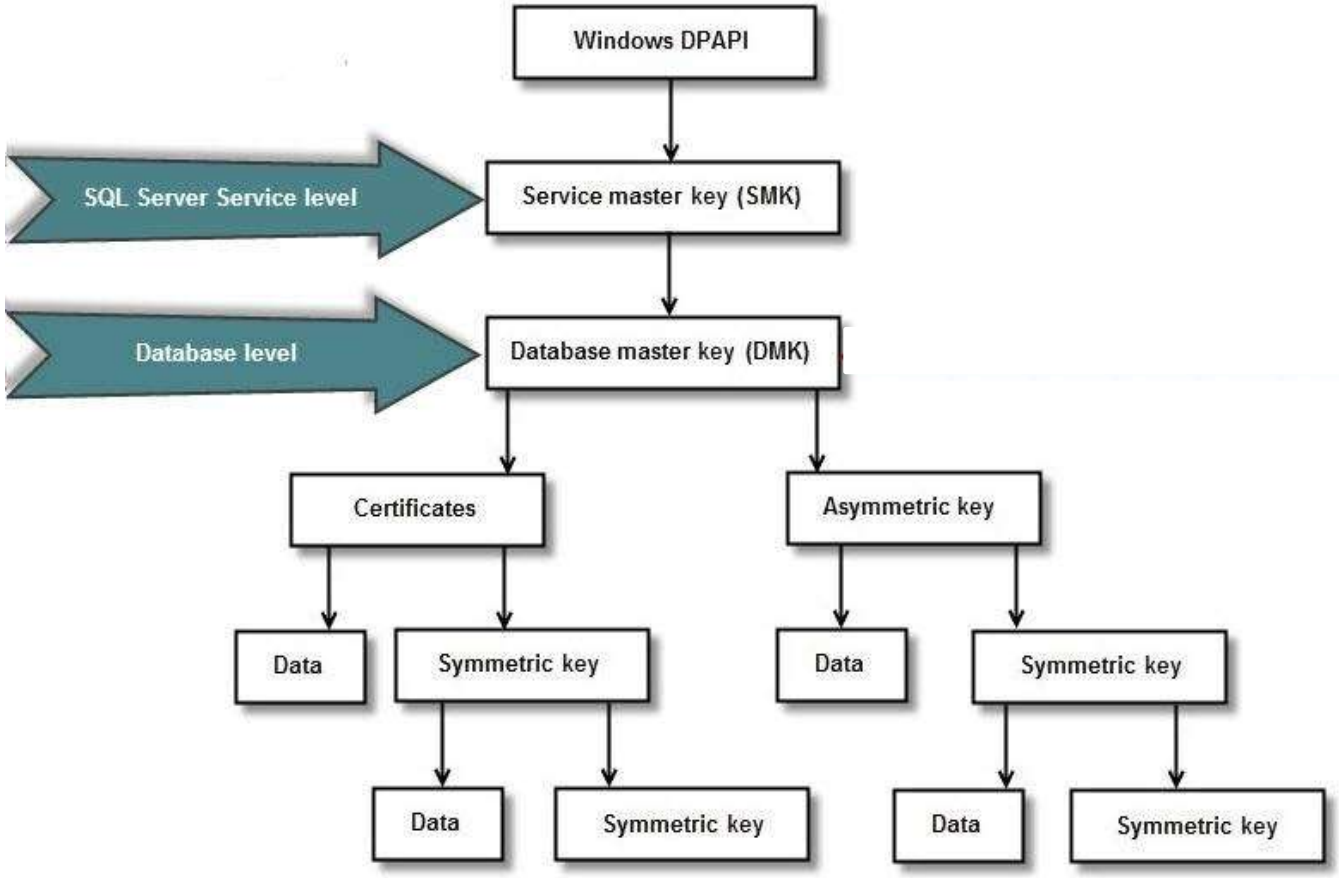
Requires an authorized user!

```
/* Decrypt the data of the SecondCol  */
USE EncryptTest
GO
OPEN SYMMETRIC KEY TestTableKey DECRYPTION
BY CERTIFICATE EncryptTestCert
SELECT   CONVERT(VARCHAR(50),DECRYPTBYKEY(EncryptSecondCol))
AS DecryptSecondCol
FROM TestTable
GO


USE EncryptTest
GO
CLOSE SYMMETRIC KEY TestTableKey
GO
```

	FirstCol	DecryptSecondCol
1	1	First
2	2	Second
3	3	Third
4	4	Fourth
5	5	Fifth

Síntese


Windows DPAPI
SQL Server Service level
Service master key (SMK)
Database level
Database master key (DMK)
Certificates
Asymmetric key
Data
Symmetric key
Data
Symmetric key
Data
Symmetric key
Data
Symmetric key

Related Catalog Views

Encryption Views

sys.asymmetric_keys	sys.cryptographic_providers
sys.certificates	sys.key_encryptions
sys.column_encryption_key_values	sys.openkeys
sys.column_encryption_keys	sys.security_policies (Transact-SQL)
sys.column_master_key_definitions	sys.security_predicates (Transact-SQL)
sys.crypt_properties	sys.symmetric_keys

https://docs.microsoft.com

Encryption vs Hashing

- Diferença principal:
  - dados encriptado podem ser desencriptados
  - Dados sob *Hashing* não
  - Utilidade: Se dados não são processados e.g. password

- A encriptação pode produzir resultados diferentes para os mesmos dados.
  - Com *Hashing* não.
  - Assim apesar de irrecuperável é comparável.

- Fator de decisão:
  - é necessário ou não processar os dados encriptados guardados?
    e.g. cartão de crédito
  - Se é então segurança é via encriptação;
    Se não então via *Hashing*

Exemplo

```
Insert into testTable (usr, hashed_password)
  values (usr1, HashBytes('MD5', 'Pass1'));


-- Password check
If exists (
     SELECT *
     FROM testTable
     WHERE id= ‘usr1’ AND hashed_password = HashBytes('MD5', 'Pass1'))
      BEGIN
          SELECT 'Password Matched'
      END
     ELSE
      BEGIN
       SELECT 'Password Did Not Match'
       END
```

DEMO

mini Sumário

1. Conceitos de chaves publica e privada
2. Encriptação em SQL Server:

   hierarquia de Keys
3. Encriptação vs *Hashing*

10:00

Exercícios

1. Distinga a utlização de chaves simétricas *vs* chaves assimétricas

2. Em que situações é adequada a utilização de encriptação

3. Em que situações é adequada a utilização de *Hashing*

Recolha das Folhas de Resposta

CORREÇÃO DOS EXERCÍCIOS

Complementos de Bases de Dados

– Segurança –

(*MS SS*)

Engenharia Informática

2º Ano / 1º Semestre

Cláudio Miguel Sapateiro
claudio.sapateiro@estsetubal.ips.pt

DSI :: Escola Superior de Tecnologia de Setúbal :: Instituto Politécnico de Setúbal