Varnostno središče

Zašita SQL strežnika

Zaščito SQL strežnika izvajamo na več nivojih

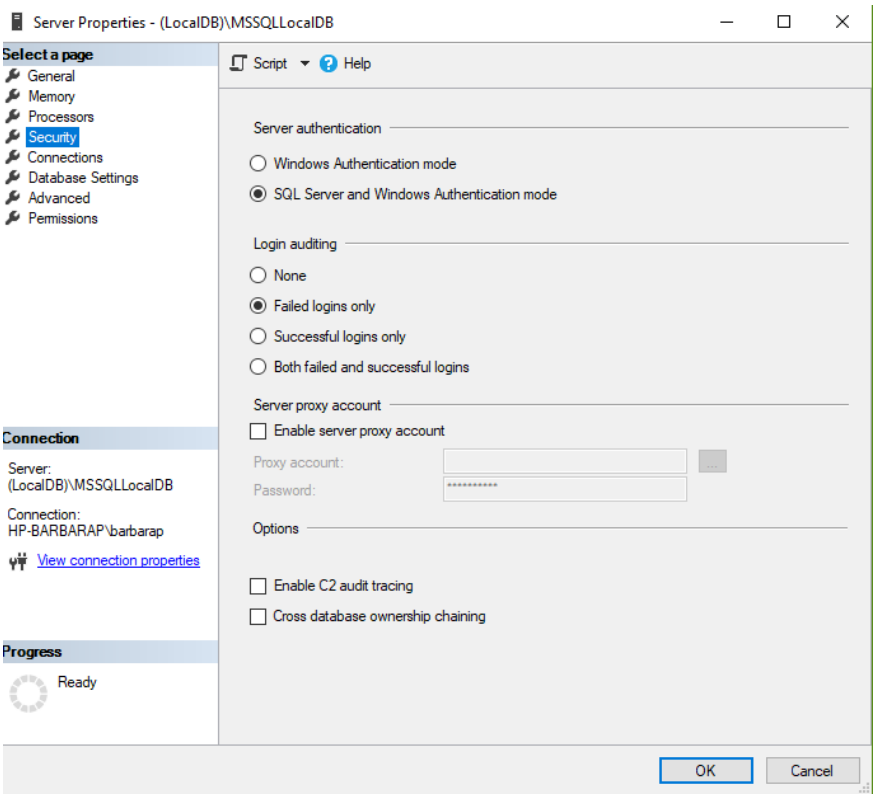
1. Zaščita platforme in omrežja
    a. Fizično varovanje
    b. Varovanje na nivoju operacijskega sistema (poskrbimo za redne posodobitve operacijskega sistema in varnostne posodobitve, omogočimo požarni zid (dostop do podatkovne baze je omogočen prek privzetih vrat 1433, ostale nastavitve po potrebi https://docs.microsoft.com/en-us/sql/sql-server/install/configure-the-windows-firewall-to-allow-sql-server-access))
    c. Varovanje SQL server programskih in podatkovnih datotek na nivoju OS
2. Zaščita podatkov in objektov v podatkovni bazi (v nadaljevanju podrobneje)
3. Varnost aplikacij, ki dostopajo do SQL strežnika

Za zaščito uporabljamo orodja kot so SQL Server Management studio, sqlcmd Utility, SQL Server Configuration Manager in druga.

Avtentikacija – kdo si?

- Kdo avtenticira?
    - Windows avtentikacija
    - SQL server avtentikacija
- Kje si avtenticiran?
    - V podatkovni bazi master
    - V uporabniški podatkovni bazi
- Druge identitete
    - Poverilnice
    - Izvajanje pod drugo prijavo
    - Izvajanje kot drugi uporabnik podatkovne baze

Med namestitvijo strežnika izberemo način avtentikacije. Lahko izberemo Windows avtentikacijo ali mešano avtentikacijo. (Windows + SQL server). Način avtentikacije lahko tudi naknadno spremenimo. To storimo v lastnostih strežnika, zavihek Varnost (Security).



Windows avtentikacija pomeni, da SQL strežnik avtenticira uporabniško ime in geslo z uporabo Windows žetona v operacijskem sistemu. Identiteto uporabnika potrdi operacijski sistem. SQL strežnik ne sprašuje po geslu in ne izvaja preverjanja identitete. Tak način avtentikacije je zelo varen. SQL strežnik zaupa poverilnicam Windows sistema, zato taki povezavi rečemo tudi »zaupna« povezava (trusted connection). Pri SQL avtentikaciji imamo prijave definirane v SQL strežniku in niso povezane z operacijskim sistemom.

Slabosti SQL avtentikacije:

- Uporabnik Windowsov mora vseeno vnašati geslo, kar pomeni, da si mora zapomniti še eno uporabniško ime in geslo.
- SQL avtentikacija ne more uporabljati Kerberos varnostnega protokola (več o tem https://en.wikipedia.org/wiki/Kerberos_(protocol))
- Windows omogoča pravila za gesla, ki niso omogočena na SQL strežniku
- Kriptirano SQL geslo mora biti poslano po omrežju med povezavo. Nekatere aplikacije, ki se avtomatično povežejo, bodo geslo shranile na odjemalca, kar pomeni varnostno tveganje.

Prednosti SQL avtentikacije:

- Omogoča podporo starejšim aplikacijam
- Omogoča podporo okoljem z mešanimi operacijskimi sistemi
- Omogoča prijavo iz neznanih ali nezaupanja vrednih omrežji
- Omogoča SQL podporo spletnim aplikacijam, kjer uporabniki sami izdelajo svojo identiteto

Kjer je le mogoče uporabljamo Windows avtentikacijo

Avtorizacija – kaj lahko delaš?

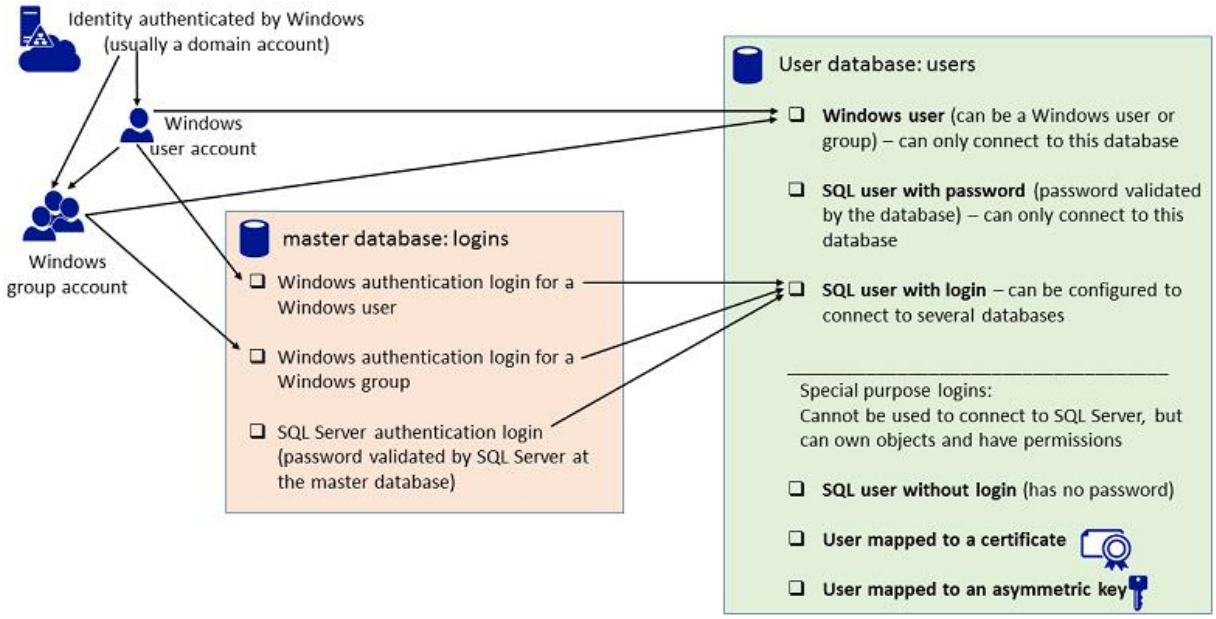
- Dodelitev, preklic in prepoved pravic (grant, revoke, deny)
- Varnost prek vlog (roles)
- Omejevanje dostopa do posameznih delov podatkov

Pravice se dodeljujejo za prijave (logins) in vloge (roles). Glavni varnostni objekti (security principals) so (objekti, ki jim lahko dodeljujemo pravice):

- Prijave (logins, lahko so Windows ali SQL server)
- Fiksne strežniške vloge (vnaprej pripravljene vloge, ki že imajo dodeljene določene pravice)
- Uporabniško definirane vloge strežnika
- Uporabniki podatkovne baze – prijavam dovolimo dostop do podatkovne baze tako, da v bazi ustvarimo uporabnika. Navadno ima isto uporabniško ime kot tisto, s katerim se je prijavil. Uporabniki podatkovne baze nimajo nujno pripadajoče prijave (logina) v streniku.
- Fiksne vloge podatkovne baze (vnaprej pripravljene vloge, ki že imajo določene pravice na podatkovni bazi)
- Uporabniško definirane vloge za podatkovno bazo
- Druge možnosti (aplikacijske vloge, prijave in uporabniki z uporabo certifikatov in asimetričnih ključev)

Priporočena metoda za konfiguracijo pravic:

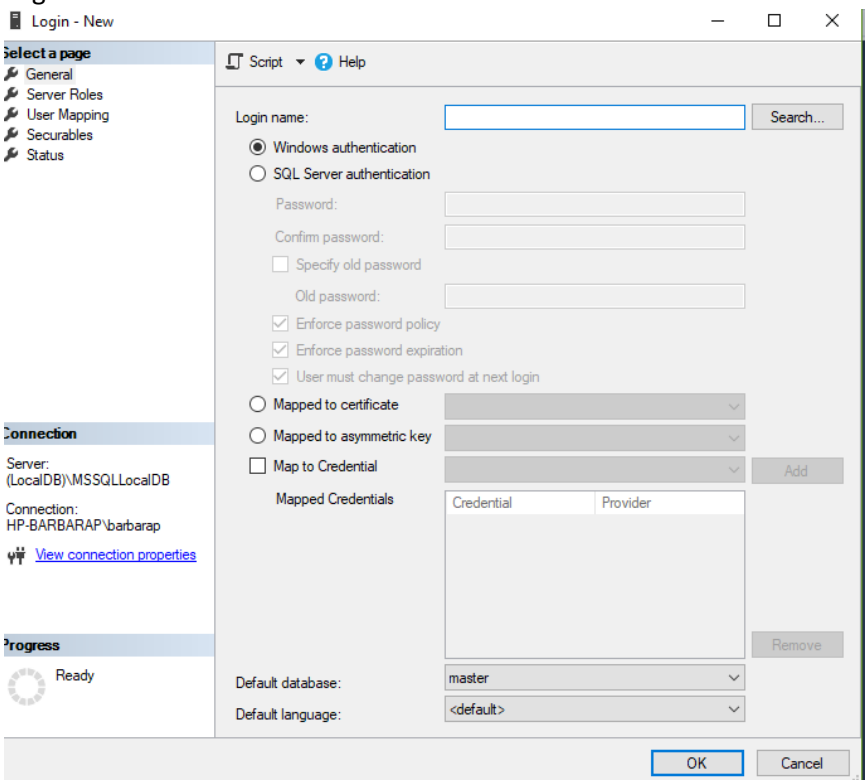
1. Na strežniku Windows: V aktivnem imeniku ustvarimo uporabnika, dodelimo ga grupi (glede na delovno mesto ali oddelek kjer dela).
2. Če bo uporabnik dostopal do več podatkovnih baz:
    a. Ustvari prijavo (login) za Windows grupo (če uporabljamo SQL avtentikacijo pa tu ustvarimo SQL server avtentikacijo) na SQL strežniku
    b. V podatkovni bazi ustvari uporabnika baze za prijavo, ki si jo ustvaril v točki a (avtenticiran si v master podatkovni bazi)
    c. V podatkovni bazi ustvari eno ali več uporabniško definiranih vlog, ki predstavljajo podobne funkcionalnosti (na primer finančnik, prodajalec,…)
    d. Dodaj uporabnike podatkovne baze eni ali več vlogam
    e. Dodaj pravice uporabniško definiranim vlogam.
3. Če bo uporabnik dostopal samo do ene podatkovne baze:
    a. Ustvari prijavo za Windows grupo (če uporabljamo SQL avtentikacijo pa tu ustvarimo SQL server avtentikacijo) na SQL strežniku
    b. V podatkovni bazi ustvari vsebovanega uporabnika podatkovne baze za Windows grupo (avtenticiran si v uporabniški podatkovni bazi)
    c. V podatkovni bazi ustvari eno ali več uporabniško definiranih vlog
    d. Dodaj uporabnika podatkovne baze eni ali več vlogam
    e. Dodaj pravice uporabniško definiranim vlogam.



1. Ustvarjanje prijave

Ustvarimo lahko prijavo za Windows uporabnika/grupa ali SQL server prijavo (če delamo v mešanem načinu).

a. V Raziskovalcu objektov izberemo strežnik in odpremo mapo Security, tam izberemo New, Login…



b. Zapišemo ime uporabnika (ali ga poiščemo)
c. Izberemo način avtentikacije (SQL bo omočena samo, če imamo nastavitev strežnika na mešano avtentikacijo)
d. Privzeta podatkovna baza je master
e. Privzet jezik za uporabnika je <defualt> = jezik SQL strežnika

V istem pogovornem oknu lahko dodamo uporabnika različnim strežniškim vlogam.

Lahko naredimo to s SQL stavkom. Na nivoju strežnika izdelamo novo okno za poizvedbe, vanj zapišemo

```
CREATE LOGIN [<domainName>\<loginName>] FROM WINDOWS;

GO
```

za Windows uporabnika ali

CREATE LOGIN shcooper

WITH PASSWORD = 'Baz1nga' MUST_CHANGE,

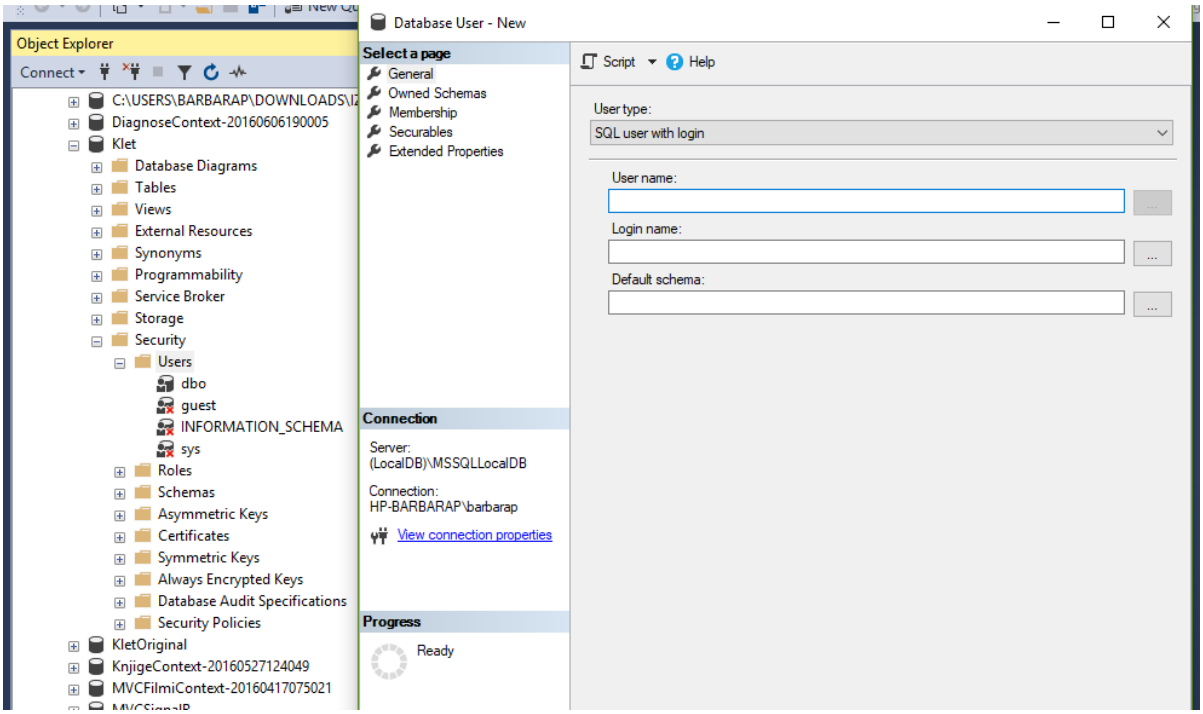
CREDENTIAL = RestrictedFaculty;

GO

za SQL server uporabnika.

2. Ustvarjanje uporabnika podatkovne baze

Uporabnika dodamo podatkovni bazi. Lahko gre za uporabnika, ki smo mu prej ustvarili prijavo za strežnik (SQL user with login), lahko pa dodamo uporabnika neposredno za eno podatkovno bazo. Izberemo bazo in v mapi Security dodamo uporabnika.



Lahko uporabimo SQL stavek

```
CREATE USER Test FOR LOGIN Test;
```

3. Priključevanje vlogam

Uporabnika lahko priključimo vlogam na strežniku ali vlogam podatkovne baze. Ko dodamo uporabnika vlogam, smo mu dodelili tudi pravice (druga možnost je, da mu pravice dodeljujemo posebej, glej spodaj).

Vnaprej definirane strežniške vloge:

Strežniška vloga	Opis
sysadmin	Člani **sysadmin** imajo vse pravice na strežniku.
serveradmin	Člani **serveradmin** lahko spremenijo nastavitve strežnika in ga ugasnejo.
processadmin	Člani **processadmin** lahko končajo procese, ki tečejo v trenutnem izvodu SQL strežnika.
setupadmin	Člani **setupadmin** lahko odstranijo povezane strežnike
bulkadmin	Člani **bulkadmin** lahko izvedejo `BULK INSERT` ukaz.
diskadmin	**diskadmin** je za upravljanje z diskom.
dbcreator	Člani **dbcreator** lahko ustvarijo, spreminjajo, brišejo in restavrirajo katerokoli bazo.
public	Vsaka prijava v SQL Server pripada **public** vlogi. Če uporabniku ne dodelimo drugih pravic ima pravice vloge public. Članstva v vlogi public ni mogoče spreminjati

Vnaprej definirane vloge podatkovne baze:

Vloga pod. baze	Opis
db_owner	Člani **db_owner** vloge lahko konfigurirajo in vzdržujejo PB, lahko jo tudi izbrišejo.
db_securityadmin	Člani **db_securityadmin** lahko spreminjajo članstvo v vlogah in upravljajo s pravicami.
db_accessadmin	Člani **db_accessadmin** vloge lahko dodvolijo ali prekličejo dostop do PB različnim prijavam.
db_backupoperator	Člani **db_backupoperator** vloge lahko izvedejo backup PB.
db_ddladmin	Člani **db_ddladmin** vloge lahko poganjajo katerikoli DDL ukaz v PB.
db_datawriter	Člani **db_datawriter** vloge lahko dodajajo, brišejo, spreminjajo podatke v vseh tabelah.
db_datareader	Člani **db_datareader** vloge lahko berejo podatke iz vseh tabel..
db_denydatawriter	Člani **db_denydatawriter** vloge ne smejo dodati, spreminjati ali brisati podatkov.
db_denydatareader	Člani **db_denydatareader** vloge ne smejo brati podatkov iz tabel.

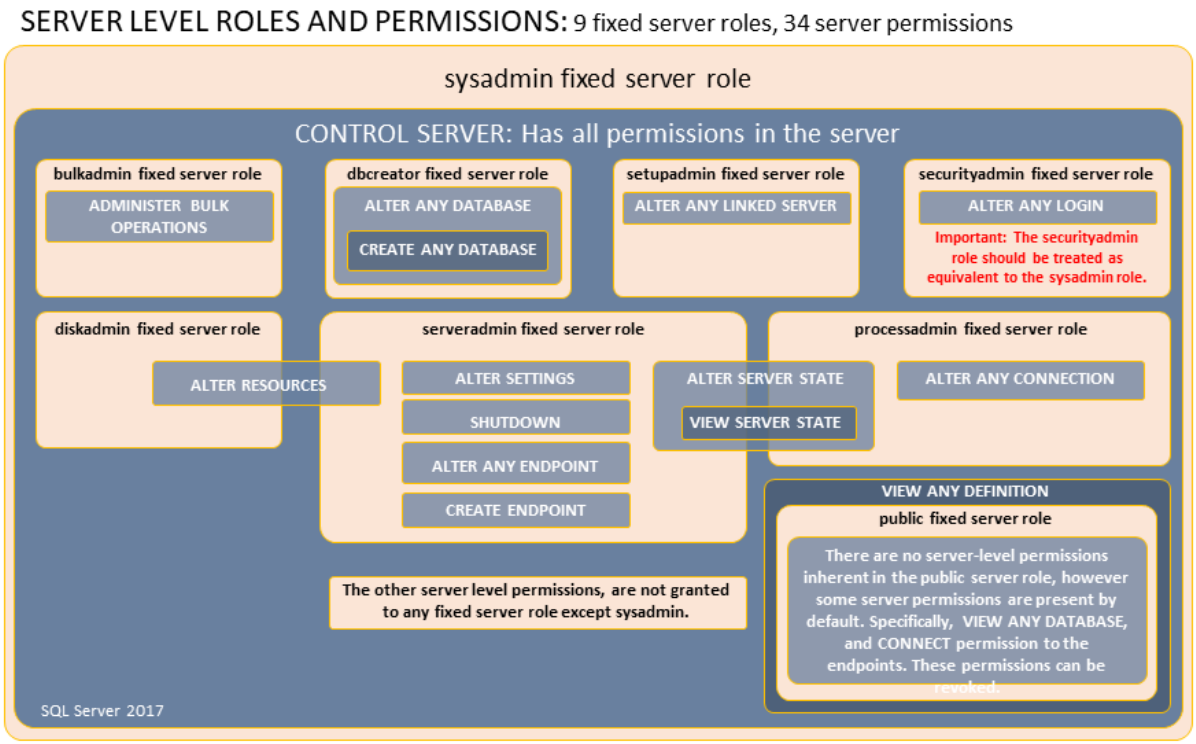
Lahko ustvarimo tudi svoje strežniške vloge in vloge podatkovne baze.

Posebej ima SQL strežnik tudi aplikacijske vloge, ki jih uporabljamo, ko želimo da neka aplikacija, ki dostopa do SQL strežnika teče v kontekstu uporabnika s svojo prijavo v SQL strežnik.

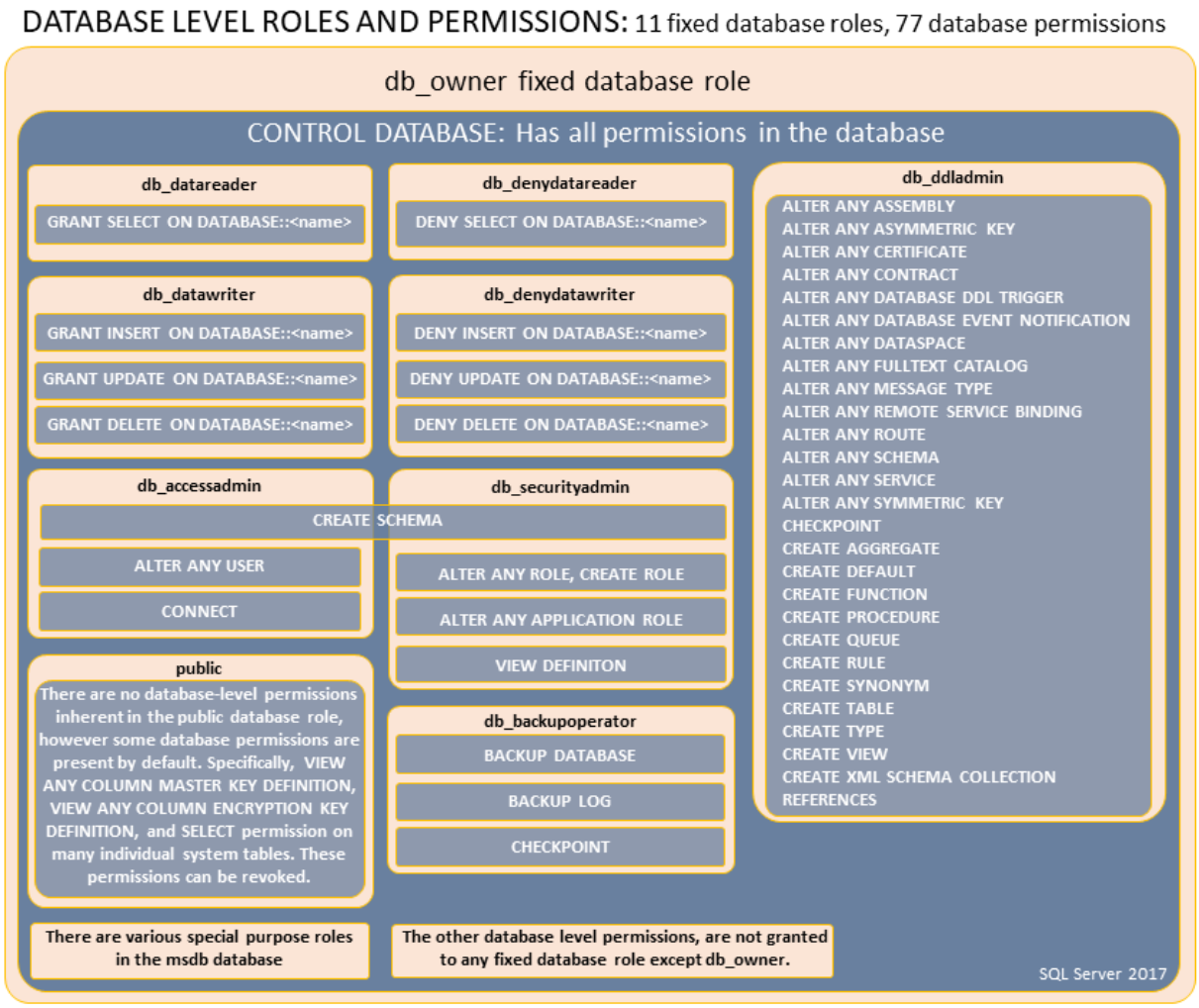
Dodeljevanje pravic

Pravice so povezane z vlogami. Vsaka izmed vlog je sestavljena iz ene ali več pravic.

Na spodnji sliki so pravice, ki jih imajo fiksne strežniške vloge:



Pravice fiksnih vlog podatkovne baze:



Oblika dodeljevanja pravic:

AUTHORIZATION PERMISSION ON SECURABLE::NAME TO PRINCIPAL;

AUTHORIZATION = GRANT, REVOKE ali DENY

PERMISSION je pravica, ki jo dodeljujemo (ALTER, CONTROL, DELETE, EXECUTE, IMPERSONATE, INSERT, SELECT, TAKE OWNERSHIP, UPDATE, VIEW DEFINITION,…)

ON SECURABLE::NAME tip objekta (server, server objekt, podatkovna baza, objekt v podatkovni bazi. Nekatere pravice uporabljajo preprostejšo sintakso, ker je jasno za kateri objekt gre. Na primer GRANT CREATE TABLE TO Mary;

PRINCIPAL =Komu dodeljujemo pravico (login, uporabnik, vloga)

Pravice dodeljujemo z GRANT, eksplicitno jih prepovedujemo z DENY. Pravico, ki smo jo dodelili prekličemo z REVOKE. Pravice so kumulativne, le DENY prekliče vse ostale.

Primer: Grupa Prodaja dobi pravico SELECT na tabeli Naročilo prek ukaza

GRANT SELECT ON OBJECT::Naročilo TO Prodaja;

Uporabnik Tadej je v grupi Prodaja. Tadeju je bila dodeljena pravica na tabeli Naročilo z ukazom

GRANT SELECT ON OBJECT::Naročilo TO Tadej;

Denimo, da želimo preklicati dovoljenje grupi Prodaja. Če uporabimo ukaz

REVOKE SELECT ON OBJECT::Naročilo TO Prodaja;

Bo Tadej obdržal pravico na tabeli, saj mu je bila pravica posebej dodeljena. Če pa izvedemo ukaz

DENY SELECT ON OBJECT::Naročilo TO Prodaja;

Bo Tadej brez pravic na tabeli Naročilo, saj smo z DENY preklicali dovoljenja vsem v grupi (DENY preglasi GRANT)

Pravice imajo povezavo roditelj/otrok, kar pomeni: Če dodelimo pravico SELECT na podatkovni bazi, smo isto pravico podelili tudi na vseh tabelah in ostalih objektih v podatkovni bazi.

V SQL serverju 2016 imamo 230 različnih pravic. Vse si lahko ogledate na https://social.technet.microsoft.com/wiki/contents/articles/11842.sql-server-database-engine-permission-posters.aspx

Priporočljivo je uporabljati dovoljenja in opuščati uporabo fiksnih vlog za podatkovne baze. Preverjanje pravic vključuje preverjanje članstva v skupinah, lastništva, eksplicitnih in implicitnih pravic na posameznem objektu. Splošni proces zbere vse pomembne pravice. Če ni nobene DENY pravice, poišče GRANT, ki omogoča dostop. Algoritem za preverjanje pravic uporablja tri glavne elemente:

a. Varnostni kontekst: Vsebuje prijavo, uporabnika, članstvo v vlogah, članstvo v Windows grupah
b. Prostor dovoljenj: Je varnostna entiteta in vsi razredi, ki vsebujejo to entiteto. Na primer: tabela – varnostna entiteta je vsebovana v shemi varnosti za tabelo in za podatkovno bazo, kar pomeni, da vplivajo na dostop do tabele pravice na nivoju tabele, sheme, podatkovne baze in strežnika.
c. Zahtevana pravica: Pravica, ki je zahtevana za izvedbo določenega SQL stavka

Glavni koraki algoritma:

1. Preskoči preverjanje pravic, če je prijava član strežniške vloge sysadmin ali je dbo za trenutno podatkovno bazo
2. Združi vse identitete na nivoju strežnika in podatkovne baze, ki so povezane s prijavo, da dobiš »varnostni kontekst«
3. Za ta varnostni kontekst zberi vsa dovoljenja v »prostor dovoljenj«
4. Določi zahtevano pravico
5. Prepovej dostop, če je zahtevana pravica neposredno ali posredno prek identitet povezana s prepovedjo DENY
6. Dovoli dostop, če je zahtevana pravica dovoljena v »prostoru dovoljenj«

SQL Injection

SQL injection imenujemo zlonamerno kodo, ki jo posredujemo strežniku. Na primer: Denimo, da bi radi izbrali vse stranke iz določenega mesta. Mesto si bo izbral uporabnik in bo posredovano naši poizvedbi, primer kode bi lahko bil:

```
var mesto;
```

```
mesto=Request.form("ShipCity");
var sql="select * from Naročila where ShipCity='"+mesto+"'";
```

Uporabnik namesto mesta (na primer Redmond) vnese `Redmond;drop table Naročila`
Ker gre za veljaven stavek, ga bo SQL strežnik izvedel. Zato moramo vedno verificirati vnos uporabnika, se izogibati izdelavi poizvedb v uporabniški aplikaciji, pregledati kodo in jo testirati, da ne bo dopuščala SQL injection-a.