DUM 15 téma: Příkazy pro řízení přístupu

ze sady: 3 **tematický okruh sady:** III. Databáze
ze šablony: 7 – Kancelářský software **určeno pro:** 4. ročník
vzdělávací obor: 18-20-M/01 Informační technologie
vzdělávací oblast: odborné vzdělávání
metodický list/anotace: viz VY_32_INOVACE_07315ml.pdf

Důvěrnost a bezpečnost dat

Dala, uložená v relačním databázovém systému, jsou velmi zranitelná díky tomu, že k nim má centralizovaným způsobem přístup jakýkoliv uživatel sítě či přímo internetu. Úkolem systému řízení báze dat je tak zajistit ochranu dat před neoprávněným přístupem – ať čtením, či změnou a mazáním.

Pro zajištění tohoto úkolu je databázový server schopný rozlišovat uživatele mezi sebou, provádět procesy identifikace, autentizace a autorizace. Při každém příchozím požadavku prochází svou vnitřní evidenci oprávnění a zjišťuje, zda je možné požadavek realizovat. Teprve poté jen předá samotnému databázovému engine ke zpracování (případně odmítnutí jako chybně formátovaný či nerealizovatelný).

Nejčastějším způsobem identifikace uživatelů v databázovém systému jsou uživatelská jména a jako autentizační mechanismus užíváme textová hesla – byť řada databázových serverů podporuje i klíče či certifikáty.

Evidence uživatelů je většinou oddělena od samotného datového skladu a tak není nutné pro každé oprávnění přidávat dalšího uživatele. V některých databázových systémech jsou dokonce uživatelé uloženi, či převzati, fixně a správce databáze může pouze již existujícím uživatelům pouze přidávat či odebírat práva.

Pro řízení uživatelských oprávnění se v databázích užívá jazyk DCL – Data Control Language. V Jazyce SQL reprezentovaný příkazy GRANT a REVOKE

Přidělení práv

Systémy založené na SQL využívají pro přidělení práv kombinace údajů uživatel-databáze-tabulka-činnost a tento konglomerát při každém dotazu testují proti tabulce oprávnění, kterou si systém sám vede. Práva tak lze řídit na úrovni celé databáze, nebo jejích jednotlivých tabulek; dále i na úrovni jednotlivých prováděných příkazů.

Typickou syntaxí příkazu GRANT je:

GRANT ***práva*** **ON** ***tabulka_či_databáze*** **TO** ***uživatel@host***

Kdy jak sekce „práva“ může zahrnovat i několik dovolených příkazů.

Praktická ukázka:

GRANT SELECT,INSERT **ON** databaze1.test **TO** ‘frantisek‘@‘recepce‘;

V této podobě je uživateli „frantisek“ dovoleno provádět příkazy SELECT a INSERT v rámci tabulky „test“ uložené v databázi „databaze1“. Ovšem pouze v případě, že je přihlášen z počítače „recepce“. Místo označení počítače lze použít třeba jeho IP adresu, typicky u webových aplikací pak 127.0.0.1 (localhost) kdy prává mají pouze uživatelé na místním počítači – třeba PHP server a podobně. Pokud chceme práva přidělit bez ohledu na stanici, použijeme zástupný znak „%“:

GRANT SELECT,INSERT **ON** databaze1.test **TO** ‘frantisek‘@‘%‘;

poznámka: Některé implementace SQL vyžadují uvedení uživatelského jména v apostrofech, některé ne. Jejich uvedením ale většinou nic nezkazíme.

Další úpravou může být přidělení práv pro všechny (třeba zatím ještě neexistující) tabulky v celé databázi. Zde používáme zástupný znak „*“ (na rozdíl od % všude jinde v SQL):

GRANT SELECT,INSERT **ON** databaze1.* **TO** ‘frantisek‘@‘recepce‘;

Samozřejmě lze uvést i velkorysé přidělení na všechny tabulky všech databází:

GRANT SELECT,INSERT **ON** ***.*** **TO** ‘frantisek‘@‘recepce‘;

což je ale bezpečnostně velmi nevhodné a v ostrém provozu se nikdy nepoužívá.

Kromě explicitního výčtu práv (reprezentovaných přímo příkazy jazyka SQL) je možné uživateli přidělit všechna dostupná práva pomocí termínu „ALL PRIVILEGES“:

GRANT ALL PRIVILEGES **ON** databaze1.test **TO** ‘frantisek‘@‘recepce‘;

i tento postup je v praxi spíše výjimkou a bezpečnostním rizikem.

Obecně lze uživateli přidělit tato práva:

select, insert, update, delete, index, alter, create, drop, grant, references, reload, shutdown, process a file

(tučně označená jsou v praxi používána, ostatní zcela výjimečně)

Kromě přidělení práva pro celou tabulku, je v některých variantách SQL (třeba MySQL) možné rozlišovat práva až na úroveň jednotlivých sloupců. Například umožňuj uživateli editovat pouze některé části datových vět:

GRANT SELECT, UPDATE(datum_prijezdu) **ON** databaze1.test **TO** ‘frantisek‘@‘recepce‘;

V tomto případě může Fantišek v recepci prohlížet veškeré záznamy v tabulce „test“, ale pouze ve sloupci dat_prijezdu může provádět změny.

Někdy může být nutné přidělit práva i uživateli neautentifikovanému (pokud databázi zpřístupňujeme předem neznámému okruhu uživatelů. V takovém případě místo uživatelského jména použijeme prázdné apostrofy ‘’:

GRANT SELECT,INSERT **ON** databaze1.test **TO ‘** ‘@‘bar‘;

Tento přístup je ale mnohem lépe nahradit zřízením univerzálního účtu (třeba „čtenář“), který pro takové účely využijeme. Případný zájemce o přístup k databázi (typicky externí firma, která potřebuje do naší databáze přístup) si o něj požádá.

Zatím nezmíněnou částí jsou hesla. Doposud uvedené příklady sice rozlišovaly uživatelská jména, avšak při jejich znalosti už dotazující se nemusel nijak svou identitu prokazovat. Způsob ochrany databáze pouze tím, že nezveřejníme jména uživatelů, je velmi chabý. Proto zásadně uživatelům přidělíme hesla, pomocí rozšíření „IDENTIFIED BY“:

GRANT SELECT,INSERT **ON** databaze1.test **TO ‘**frantisek‘@‘recepce‘ **IDENTIFIED BY** ‘tajne heslo‘;

poznámka: Místo zápisu hesla v otevřené podobě je možné použít direktivu „PASSWORD" a uložit přímo hash hesla:

… TO ‘frantisek‘@‘recepce‘ **IDENTIFIED BY PASSWORD** ‘F598A54D2165..’;

Pozor! Prvním použitím direktivy „IDENTIFIED BY“ zároveň uživateli nastavíte heslo, které bude platné pro všechna ostatní práva jemu dříve či později nastavená. Následné přidělení dalších práv bez „IDENTIFIED BY“ přidělí práva, ovšem již s požadavkem na použití hesla. V některým implementacích SQL je možné nastavit různá hesla v závislosti na místu, odkud se uživatel přihlašuje. kombinace uživatelského jména a hosta odkud posílá požadavek je pak spolu vlastně teprve uživatelským jménem, ke kterému se váže účet.

Toho se dá využít třeba při odlišení přístupu z firemní sítě (či lokálního stroje), kdy stačí heslo jednoduché či žádné a naopak při přístupu zvenčí je požadováno použití hesla.

Delegování práv

Má.li uživatel již přidělena nějaká práva, může za určitých okolností tato práva dále předávat jiným uživatelům. Záleží na správci databázového serveru, zda takový postup umožní a do jaké míry. Rozhodujícím faktorem je uvedení (či neuvedení) direktivy „WITH GRANT OPTION“, která umožňuje získaná práva dále předávat. Bez této direktivy uživatel přidělená práva nemá možnost nikomu dalšímu předat. Logicky také může dále předávat pouze taková práva, která sám má – případně jen jejich část.

V našem případě tedy:

GRANT SELECT,INSERT **ON** databaze1.test **TO ‘**frantisek‘@‘recepce‘ **WITH GRANT OPTION**;

způsobí, že František může dále předávat svá práva SELECT a INSERT jakémukoliv existujícímu uživateli. Pokud bychom chtěli, aby mohl sále delegovat pouze SELECT a ne INSERT, přidělíme práva nadvakrát – poprvé pouze SELECT s direktivou GRANT OPTION a poté INSERT bez ní.

Podstatné také je, že většina běžných uživatelů databáze není oprávněna v databázovém systému vytvářet další uživatele. Proto lze práva delegovat pouze na již existující a většinou jim samozřejmě ani nelze změnit či nastavit heslo. V jiných konfiguracích databázového systému je možné použití GRANT pro neexistujícího uživatele brát zároveň jako pokyn pro jeho vytvoření… pokud na to grantující má sám právo.

V ostatních případech musí správce systému nejprve uživatele vytvořit pomocí „CREATE USER“ a případně mu i přidělit heslo atd.

Odebrání práv

Kdekoliv lze práva přidávat, lze je i odebírat. V jazyce SQL k tomu slouží příkaz „REVOKE“. Odebrat práva může samozřejmě ten, kdo je smí i přidělovat (disponuje GRANT OPTION)-a to pouze v rozsahu těch práv, která sám k objektu má.

Syntaxe je obdobná jako u GRANT:

REVOKE INSERT **ON** databaze1.test **FROM** ‘frantisek‘@‘recepce‘;

odebere Františkovi právo INSERT, právo SELECT mu tak zůstává.

Případně lze použít i REVOKE ALL PRIVILEGES… k odebrání všech práv, ovšem pozor-i v takovém případě uživatel v systému zůstává, byť například nemá žádná práva k žádné tabulce.

Úkoly pro samostatnou práci

- sestavte příkaz, který umožní uživateli frantisek vkládat záznamy do tabulky „hoste“, ovšem pouze pokud je přihlášen na počítači „kancelář“
- zajistěte, aby uživatel „pavel“ mohl číst veškerá data na databázovém serveru, pokud je přihlášen na počítači „reditel“ tak bez hesla, v ostatních případech musí použít heslo (heslo zvolte dle uvážení)
- dovolte uživateli „jirka“ číst a mazat z tabulky „snidane“ a to tak aby on sám mohl dovolit čtení i jiným uživatelům

Zdroje:

- Archiv autora