UNIVEZITET U NIŠU

ELEKTRONSKI

FAKULTET

Seminarski rad

Sigurnost baza podataka u MySQL

Predmet: Sistemi za upravljanje bazama podataka

Mentor: Aleksandar Stanimirović

Student: Marija Stojanović
1034

Niš, maj 2020. God

Sadržaj:

1. Uvod

Sigurnost baza podataka odnosi se na upotrebu širokog spektra kontrola za zaštitu informacija kako bi se zaštitila baza podataka (podaci, aplikacije baze podataka ili sačuvane funkcije, sistem baza podataka, serveri baza podataka i sve povezane mreže) od ugrožavanja njihove poverljivosti, integriteta i dostupnosti.[1]

Sigurnosni rizici za sisteme baza podataka uključuju:[1]

- Neautorizovane ili nenamerna aktivnost, kao i zloupotreba od strane ovlašćenih korisnika baze podataka, administratora baza podataka ili menadžera mreže/sistema ili neovlašćenih korisnika ili hakera (npr. neprimereni pristup osetljivim podacima, metapodacima ili funkcijama unutar baza podataka ili neprimerene promene programa, struktura ili konfiguracije sigurnosti);
- Infekcije zlonamernim softverom koji može izazvati incidente kao što su neovlašćeni pristup, curenje ili otkrivanje ličnih podataka, brisanje ili oštećenje podataka ili programa, prekid ili uskraćivanje ovlašćenog pristupa bazi podataka, napadi na druge sisteme i nepredviđeni neuspeh usluga koje pruža baza podataka;
- Preopterećenja, ograničenja performansi i problemi sa kapacitetom što rezultira nesposobnošću ovlašćenih korisnika da koriste baze podataka onako kako je to planirano;
- Fizičko oštećenje servera baza podataka uzrokovano požarima ili poplavama u računarskoj sobi, pregrevanjem, grmljavinom, slučajnim izlivanjem tečnosti, statičkim pražnjenjem, elektronskim kvarovima\greškama na opremi i zastarevanjem iste;
- Dizajnerske i programerske greške u bazama podataka kao i sistemima i programima koji im pripadaju, stvarajući time različite bezbedosne ranjivosti, gubitak\korupciju podataka, propadanje performansi itd.;
- Korupcija podataka i/ili gubitak podataka uzrokovan unosom nevažećih podataka ili naredbi, greškama u procesima administracije baze podataka ili sistema, sabotažama/kriminalnim oštećivanjem itd.;

U poglavlju 2 biće govora o faktorima koji utiču na sigurnost baza podataka i temama koje je potrebno razmotriti ako govorimo o sigurnosti.

U poglavlju 3 biće detaljnije objašnjena kontrola pristupa i rukovođenje korisničkim nalozima. Upravo je ovo oblast iz sigurnosti baza podataka kojom ćemo se baviti u ovom radu.

2. Sigurnost

Zbog svega navedenog u poglavlju 1, kada se razmišlja o sigurnosti unutar MySQL-a, potrebno je razmotriti širok spektar mogućih tema i kako one utiču na bezbednost MySQL servera i povezanih aplikacija:[2]

- Opšti faktori koji utiču na bezbednost. Oni uključuju izbor dobrih lozinki, nedavanje nepotrebnih privilegija korisnicima, osiguravanje bezbednosti aplikacija sprečavanjem SQL injection, korupcije podataka i drugo.
- Sigurnost same instalacije. Fajlovi podataka, log fajlovi kao i svi fajlovi aplikacije u instalaciji potrebno je da budu zaštićeni kako bi se izbeglo da budu dostupni za čitanje i upis neautorizovanim korisnicima.
- Kontrola pristupa i sigurnost unutar samog sistema baze podataka, uključujući korisnike i baze podataka kojima je dodeljen pristup bazama podataka, prikazima i programima koji se koriste u bazi podataka.
- Funkcije koje su omogućene korišćenjem dodataka (plugins) za sigurnost
- Sigurnost mreže koja povezuje MySQL i vaš sistem. Sigurnost je povezana sa privilegijama individualnih korisnika, ali ukoliko se želi ograničiti MySQL tako da je dostupan samo lokalno na host MySQL serveru ili ograničenom skupu drugih hostova.
- Osigurati da su dostupne adekvatne kopije fajlova baze podataka, konfiguracionih i log fajlova. Takođe osigurati se da je dostupno rešenje za oporavak podataka iz rezervnih kopija, i naravno tesirati dato rešenje.

U nastavku rada skoncentrisaćemo se na bezbednost MySQL-a koja je omogućena ograničavanjem kontrole pristupa podacima od strane korisnika, kao i upravljanjem korisničkim nalozima.

3. Kontrola pristupa i upravljanje korisničkim nalozima

MySQL omogućava kreiranje naloga koji omogućavaju korisnicima klijentima da se povežu na server i pristupe podacima kojima on upravlja. Primarna funkcija MySQL sistema koji omogućava dodeljivanje privilegija korisnicima je autentifikacija korisnika koji se povezuju sa datog hosta i povezivanje datog korisnika sa privilegijama na bazu podataka tj. na naredbe SELECT, INSERT, UPDATE i DELETE. Dodatna funkcionalnost uključuje mogućnost dodeljivanja privilegija koje se tiču administrativnih operacija. [2]

Kako bi se kontrolisalo koji korisnik se može povezati, svakom nalogu moraju biti dedeljeni autentifikacijski kredencijali kao što je lozinka i korisničko ime. Korisnički interfejs ka MySQL nalozima sastoji se od SQL naredbi kao što su CREATE USER, GRANT i REVOKE.[2]

MySQL system za dodeljivanje privilegija omogućava da svi korisnici mogu izvršavati jedino operacije koje su im dozvoljene. Kao korisnik, prilikom konektovanja na MySQL server, vaš identitet je određen hostom sa koga se konektujete i korisničkim imenom koje ste specificirali. Nakon konektovanja, kada izdajete zahteve, sistem daje privilegije u skladu sa vašim identitetom i onim što želite da uradite.[2]

MySQL grant tabele

Kada se MySQL po prvi put instalira, MySQL instaler automatski kreira dve baze podataka: *test* bazu, koja služi kao poštansko sanduče za nove korisnike i *mysql* bazu podataka koja sadrži nekoliko MySQL grant tabela koje sadrže informacije o korisničkim nalozima i privilegijama koje su im dodeljene. To možemo videti pomoću sledećih komandi:[3]

```
mysql> SHOW tables FROM mysql;
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| columns_priv              |
| db                        |
| engine_cost               |
| event                     |
| func                      |
| general_log               |
| gtid_executed             |
| help_category             |
| help_keyword              |
| help_relation             |
| help_topic                |
| innodb_index_stats        |
| innodb_table_stats        |
| ndb_binlog_index          |
| plugin                    |
| proc                      |
| procs_priv                |
| proxies_priv              |
| server_cost               |
| servers                   |
| slave_master_info         |
| slave_relay_log_info      |
| slave_worker_info         |
| slow_log                  |
| tables_priv               |
| time_zone                 |
| time_zone_leap_second     |
| time_zone_name            |
| time_zone_transition      |
| time_zone_transition_type |
| user                      |
+---------------------------+
31 rows in set (0.01 sec)
```

Slika 1: Tabele u mysql bazi

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| diligent_db        |
| diligent_db_new    |
| mu_mining          |
| mysql              |
| performance_schema |
| smartone_db        |
| sys                |
| test               |
+--------------------+
```

Slika 2: Podrazumevane baze podataka

Sledeće tabele sadrže garancijske (grant) informacije:

- user: Korisnički nalozi, globalne privilegije, kao i još neke kolone
- db: Privilegije na nivou baze podataka.
- tables_priv: Privilegije na nivou tabela.
- columns_priv: Privilegije na nivou kolona.
- procs_priv: Privilegije nad sačuvanim procedurama i funkcijama.
- proxies_priv: Proksi-korisničke privilegije

Grant tabele se ne mogu modifikovati direktno. Modifikacija ovih tabela dešava se indirektno prilikom korišćenja već pomenutih naredbi za upravljanje korisničkim nalozima a to su CREATE USER, GRANT i REVOKE, kako bi se oformili nalozi i dodelile im se privilegije. Kada se koriste navedene naredbe prilikom manipulacije korisničkim nalozima, server sam modifikuje grant tabele.[2]
Direktna manipulacija grant tabelama koristeći naredbe kao što su INSERT, UPDATE ili DELETE nije preporučljiva i može se raditi na sopstveni rizik. Server je u ovakvim slučajevima slobodan da ignoriše redove tabela koji su potencijalno rizični, a koji su rezultat ovakve vrste modifikacija.

Tabele *user* i *db*

Tabela *user* predstavlja najbitniju od šest grant tabela. Server koristi tabelu *user* zajedno sa tabelom *db* u mysql bazi podataka u prvoj i drugoj fazi kontrole pristupa. Ova tabela određuje koji korisnici mogu pristupiti kojim bazama i sa kojih hostova.[2] Kolone iz tabela *user* i *db* su prikazani na sledećoj slici: [2]

Table 6.3 user and db Table Columns

Table Name	user	db
Scope columns	Host	Host
	User	Db
		User
Privilege columns	Select_priv	Select_priv
	Insert_priv	Insert_priv
	Update_priv	Update_priv
	Delete_priv	Delete_priv
	Index_priv	Index_priv
	Alter_priv	Alter_priv
	Create_priv	Create_priv
	Drop_priv	Drop_priv
	Grant_priv	Grant_priv
	Create_view_priv	Create_view_priv
	Show_view_priv	Show_view_priv
	Create_routine_priv	Create_routine_priv
	Alter_routine_priv	Alter_routine_priv
	Execute_priv	Execute_priv
	Trigger_priv	Trigger_priv
	Event_priv	Event_priv
	Create_tmp_table_priv	Create_tmp_table_priv
	Lock_tables_priv	Lock_tables_priv
	References_priv	References_priv
	Reload_priv	
	Shutdown_priv	

Slika 3: Kolone tabela *user* i *db*

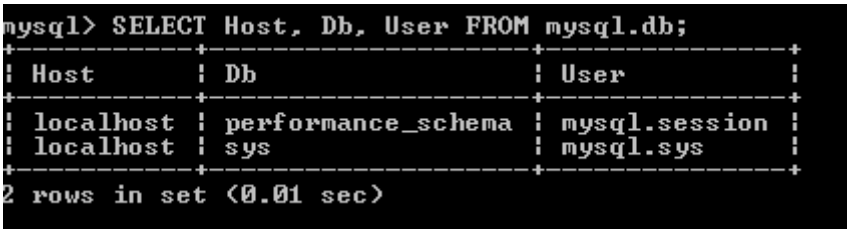
	Process_priv	
	File_priv	
	Show_db_priv	
	Super_priv	
	Repl_slave_priv	
	Repl_client_priv	
	Create_user_priv	
	Create_tablespace_priv	
Security columns	ssl_type	
	ssl_cipher	
	x509_issuer	
	x509_subject	
	plugin	
	authentication_string	
	password_expired	
	password_last_changed	
	password_lifetime	
	account_locked	
Resource control columns	max_questions	
	max_updates	
	max_connections	
	max_user_connections	

Slika 4: Kolone tabela *user* i *db* (nastavak)

Polja *Host, User, authentication_string* iz tabele *user* definišu kojim korisnicima je dopušteno da se konektuju na server baze podataka, njihove šifre kao i hostove sa kojih mogu da se konektuju – MySQL koristi kombinaciju user i host identifikacije kao osnovu za sistem sigurnosti baza podataka. U sledećoj tabeli možemo videti da kada se korisnik konektuje sa localhost mašine, loguje se kao root korisnik sa svojom šifrom. Po osnovnim podešavanjima root korisnik nema šifru, tj authentication_string ali u sledećem primeru vidimo da je šifra ipak postavljena tj. promenjena za root korisnika. Root korisnik može pristupiti kompletno svim bazama u sistemu. Takođe korisnici sa drugih hostova nemaju pristup nijednoj bazi u sistemu.[3]

Slika 5: Kolone Host, User, authentication_string iz user tabele

U tabeli db u poljima Host, Db i User imamo određeno koji korisnik sa kog hosta ima pristup kojoj bazi podataka. Slika



Slika 6: Kolone Host, User, Db iz db tabele

Tabele *tables_priv* i *columns_priv*

Tabele *tables_priv* i *columns_priv* su slične kao i *db* tabela, ali su malo više specijalizovane: ove tabele se odnose na nivo tabela i pojedinačnih kolona, a ne na nivo celokupne baze podataka. Preciznije rečeno *tables_priv* se odnosi na privilegije nad celim tabelama u bazi podataka dok se *columns_priv* odnosi na privilegije nad pojedinačnim kolonama.

Table 6.4 tables_priv and columns_priv Table Columns

Table Name	tables_priv	columns_priv
Scope columns	Host	Host
	Db	Db
	User	User
	Table_name	Table_name
		Column_name
Privilege columns	Table_priv	Column_priv
	Column_priv	
Other columns	Timestamp	Timestamp
	Grantor	

Slika 7: Kolone tabela *tables_priv* i *columns_priv*

Tokom drugog stepena kontrole pristupa, server vrši verifikaciju zahteva kako bi se osiguralo da svaki klijent ima dovoljno privilegija za svaki zahtev koji izda. Pored *user* i *db* grant tabela, server može uzeti u obzir i *tables_priv* kao i *columns_priv* tabele za zahteve koji uključuju tabele.

U sledećem primeru imamo rekord iz tabele *tables_priv* u kome vidimo da je korisniku *logger* dozvoljeno da izvrši Select, Insert i Update operacije samo nad tabelom *logs* i to sa hosta *lost.soul.com*

```
+-----------+-----+----------+------------+----------------------+
| Host      | Db  | User     | Table_name | Table_priv           |
+-----------+-----+----------+------------+----------------------+
| localhost | db1 | logger   | logs       | Select,Insert,Update |
+-----------+-----+----------+------------+----------------------+
```

Slika 8: Rekord iz tabele *tables_priv*

U sledećem primeru imamo rekord iz tabele *columns_priv* u kome vidimo da je korisniku *hr_supervisors* dozvoljeno da izvrši Select i Update operacije samo nad kolonama *name, dept* iz tabele *employees.* Takođe ovom korisniku je dozvoljena samo naredba Select nad kolonom *id* iz tabele *employees*

```
+-----+----------------+------------+-------------+-----------------+
| Db  | User           | Table_name | Column_name |  Column_priv    |
+-----+----------------+------------+-------------+-----------------+
| db1 | hr_supervisors | employees  | name        | Select, Update  |
| db1 | hr_supervisors | employees  | id          | Select          |
| db1 | hr_supervisors | employees  | dept        | Select, Update  |
| db1 | hr_users       | employees  | id          | Select          |
| db1 | hr_users       | employees  | name        | Select          |
| db1 | hr_users       | employees  | dept        | Select          |
+-----+----------------+------------+-------------+-----------------+
```

Slika 9: Rekord iz tabele *columns_priv*

Interakcija između grant tabela

Različite grant tabele međusobno interaguju jedna sa drugom kako bi kreirale obimna pravila za pristup koja MySQL koristi kada odlučuje na koji način će rukovoditi korisničkim zahtevima. Kontrola pristupa se odvija u dva stepena: prvi stepen koji podrazumeva povezivanje i drugi stepen koji podrazumeva korisničke zahteve.[3]

- **Stepen konekcije** – kada korisnik zahteva konekciju na server baze podataka sa specifičnog hosta, MySQL će prvo proveriti da li ulaz postoji za korisnika u tabeli *user*, ako je šifra korisnika validna, i ako je korisniku dopušteno da se konektuje sa specificiranog hosta. Ako je provera uspešna, konekcija će biti dozvoljena od strane servera.[3]
- **Stepen zahteva** – jednom kada je konekcija dozvoljena, svaki sledeći zahtev serveru – SELECT, DELETE, UPDATE, kao i drugi upiti će prvo biti provereni kako bi se utvrdilo da li korisnik ima potrebne privilegije kako bi izvršio datu akciju. Postoje različiti nivoi pristupa koji mogu biti omogućeni – neki korisnici mogu imati samo mogućnost da izvrše SELECT naredbu nad tabelama, dok drugi mogu imati mogućnost INSERT i UPDATE naredbi, ali ne i DELETE naredbe. [3]

Što se tiče same hijararhije MySQL grant tabela, tabela *user* je prva u hijararhiji, a odmah ispod nje su tabele *db* i *host* i tabelama *tables_priv*, *columns_priv*, *procs_priv*, *proxes_priv* koje su na dnu hijararhije. Tabele koje su na nižem nivou hijararhije proveravaju se jedino ako tabele na višem nivou ne uspeju da obezbede neophne privilegije.[3]

Kada se vrši odlučivanje o tome kom korisniku je dozvoljeno obavljanje kojih operacija nad bazom podataka, MySQL uzima u obzir polja koja sadrže potrebne privilegije iz prve tri tabele. Počinje sa tabelom user i proverava da li korisnik ima potrebne privilegije za operaciju koju pokušava da izvrši. Ako u ovoj tabeli ne postoje potrebne privilegije za datog korisnika, nakon toga se proveravaju tabele db i host kako bi se i u njima proverilo da li postoje potrebne privilegije. Tek nakon logičkog parsiranja privilegija na različite tabele, MySQL dozvoljava ili ne dozvoljava dati korisnički zahtev.[3]

Dodeljivanje, opozivanje i pregled privilegija korisnika

Kao što je već rečeno, sve informacije o korisnicima i njihovim privilegijama za pristup bazama, tabelama kao i pojedinačnim kolonama nalaze se u grant tabelama. Modifikacija i dodavanje korisničkih naloga kao i njihovih privilegija izvršava se upravo nad ovim tabelama. Da bi bila dozvoljena modifikacija ovih tabela mysql baza podataka zahteva superuser pristup MySQL serveru.

Ako je instaliran server na razvojnoj mašini, u toku instaliranja potrebno je upisati root lozinku. Ukoliko se lozinka ostavi kao prazno polje otvara se sigurnosna rupa u sistemu. [3]

Kako bismo proverili da li imamo potreban pristup, moramo se ulogovati na server kao root korisnik, kao u sledećem primeru:

```
[root@host] $ mysql -u root -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
mysql>
```

Slika 10: Logovanje root korisnika

Sledeća komanda pokazuje da li imamo pristup tabelama u mysql bazi podataka

```
mysql> USE mysql;
Database changed
```

Slika 11: Komanda USE

Nivo kontrole pristupa koji ima root korisnik je tipično distupna jedino administratoru baze podataka. Ostali korisnici imaju manji sigurnosni rejting, i samim tim limitirani pristup bazi.[3] Svaki od tih korisnika, koji nisu odministrator, se tipično povezuju na bazu koristeći svoje korisničko ime i lozinku. Kao što je već rečeno svrha MySQL grant tabela je da se omogući manipulisanje sigurnosnim podešavanjima nad ovim "običnim" korisnicima kako bi se za svakog od njih posebno kontrolisao nivo dozvoljenog pristupa.

Korišćenje GRANT i REVOKE naredbi

Način koji se preporučuje kada se radi o postavljanju korisničkih privilegija za pristup bazama podataka u MySQL grant tabelama je upravo pomoću GRANT i REVOKE komandi, koje su specijalno napravljene za ovu namenu. Ovako one izgledaju: [3]

```
GRANT privilege (field-name, field-name, ...), privilege (field-name, field-name,
...), ... ON database-name.table-name TO user@domain IDENTIFIED BY password,
user@domain IDENTIFIED BY password, ...
REVOKE privilege (field-name, field-name, ...), privilege (field-name, field-name,
...), ... ON database-name.table-name FROM user@domain, user@domain, ...
```

Slika 12: Komande GRANT i REWOKE

Kako bismo ovo ilustrovali osvrnućemo se na sledeće primere:

U prvom primeru vrši se dodela SELECT, INSERT, UPDATE i DELETE privilegija za tabelu *db1.logs* korisniku *logger* koji se povezuje sa hosta *localhost* sa lozinkom *timber*:

```
mysql> GRANT SELECT, INSERT, UPDATE ON db1.logs TO logger@localhost
IDENTIFIED BY 'timber';
```

Slika 13: Komanda GRANT za dodelu privilegija korisniku logger

Na sledećoj slici vidimo šta se dešava kad se korisnik uloguje u MySQL i pokuša da pristupi različitim tipovima upita:

```
[user@host] $ mysql -h localhost -u logger -p
Enter password: ******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7 to server version: 4.0.14
Type 'help;' or '\h' for help. Type '\c' to clear the buffer
mysql> USE mysql;
ERROR 1044: Access denied for user: 'logger@localhost' to database 'mysql'
mysql> USE db1;
Database changed
mysql> SELECT * FROM employees;
ERROR 1142: select command denied to user: 'logger@localhost' for table 'employees'
mysql> SELECT * FROM logs;
+----+----------------------------+--------+
| id | message                    | sender |
+----+----------------------------+--------+
| 34 | Apache core error          | httpd  |
| 35 | Root login failure on tty1 | pamd   |
| 36 | Root login failure on tty1 | pamd   |
+----+----------------------------+--------+
3 rows in set (0.00 sec)

mysql> INSERT INTO logs VALUES (37, 'Sendmail restart, 102 messages in queue', 'sendmail');
Query OK, 1 row affected (0.00 sec)
mysql> DELETE FROM logs WHERE id = 37;
ERROR 1142: delete command denied to user: 'logger@localhost' for table 'logs'
```

Slika 14: Logovanje korisnika i pristup upitima

U prethodnom primeru možemo videti da su jedino komande specificirane GRANT komandom dozvoljene korisniku, a sve ostale komande su odbijene.

Komanda REVOKE ima potpuno suprotan efekat od GRANT naredbe. Ova naredba omogućava da se povuku privilegije korisnika. Na sledećem primeru videćemo kako se pomoću ove komande oduzimaju korisniku *logger* mogućnosti da izvrši INSERT i UPDATE komande nad *db1.logs* tabelom.

```
mysql> REVOKE INSERT, UPDATE ON db1.logs FROM logger@localhost;
```

Slika 15: Komanda REVOKE

Ova promena ima momentalni efekat. Nakon nje, kada korisnik *logger* pokuša da izvrši INSERT naredbu nad tabelom dešava se sledeća situacija:

```
mysql> INSERT INTO logs VALUES (38, 'System powerdown signal received', 'apmd');
ERROR 1142: insert command denied to user: 'logger@localhost' for table 'logs'
```

Slika 16: Pokušaj korisnika da izvrši INSERT naredbu

MySQL omogućava korišćenje * wildcard-a kada se radi o bazama podataka i tabelama – sledeći upit dodeljuje RELOAD, PROCESS, SELECT, DELETE i INSERT privilegije za bazu podataka korisniku *admin* na hostu pod imenom *medusa*:

```
mysql> GRANT RELOAD, PROCESS, SELECT, DELETE, INSERT ON *.* TO admin@medusa
IDENTIFIED BY 'secret';
```

Slika 17: Krišćenje wildcard-a u GRANT naredbi

Sledeći primer prikazuje dodelu SELECT privilegija za tabelu *employees.compensation* i to jedino korisniku *supervisor*:

```
mysql> GRANT SELECT ON employees.compensation TO supervisor;
```

Slika 18: Dodela SELECT privilegija korisniku *supervisor*

Sledeći primer predstavlja korak dalje, on dodeljuje SELECT i UPDATE privilegije za specifične kolone iz *grades* tabele korisnicima *harry* i *john* respektivno:

```
mysql> GRANT SELECT (id, name, subj, grade) ON db1.grades TO harry;
mysql> GRANT SELECT (id, name, subj, grade), UPDATE (name, grade) ON
db1.grades TO john;
Query OK, 0 rows affected (0.00 sec)
```

Slika 19: Dodeljivanje SELECT i UPDATE privilegija za specifične kolone

Sledeći primer oduzima korisniku *tim* pravo na korišćenje CREATE i DROP operacija nad bazom podataka db2003a:

```
mysql> REVOKE CREATE, DROP ON db2003a.* FROM tim@funhouse.com;
```

Slika 20: Primer korišćenja REVOKE naredbe

Važno je napomenuti da tabele i polja koja se pominju u GRANT naredbama moraju prethodno postojati u bazi podataka, kako bi se dodelile korisnicima privilegije za datu tabelu ili kolonu. Tako]e ovo pravilo ne važi kada su u pitanju privilegije nad celim bazama podataka. Naime, MySQL dozvoljava da se dodeljuju korisnicima privilegije nad bazom podataka iako ona još ne postoji. [3]

Sledeći primer prikazuje uklanjanje prava korisnika *sarah* da izvrši operaciju UPDATE nad *name* i *address* kolonama u *customer* bazi podataka:

```
mysql> REVOKE UPDATE (name, address) ON sales.customers FROM
sarah@work.domain.net;
```

Slika 21: Primer uklanjanja prava korisnika nad kolonama *name* i *address*

MySQL takođe omogućava da se dodele sve privilegije nekom korisniku pomoću komande ALL. U sledećem primeru dodeljuju se sve privilegije u bazi podataka *web* korisniku *www* koji se povezuje sa hosta u *melonfire.com* domenu:

```
mysql> GRANT ALL ON web.* TO www@'%.melonfire.com' IDENTIFIED BY
'abracadabra';
```

Slika 22: Korišćenje komande ALL

Ograničenje korišćenja resursa

Od MySQL verzije 4.x postoji mogućnost ograničenja korišćenja resursa na MySQL serveru, na bazi svakog korisnika pojedinačno. Ovo je omogućeno dodavanjem tri nova polja u *user* tabelu: *max_questions, max_updates* i *max_connections* koja se mogu koristiti kako bi se ograničio broj upita, ažuriranja tabela ili rekorda, kao i broj konektovanja od strane pojedinačnih korisnika u roku od sat vremena, respektivno. Ova tri polja se mapiraju u tri opcioonalne klauzule u GRANT naredbi.[3]

Prva od ovih klauzula je MAX_QUERIES_PER_HOUR, koja ograničava broj upita koji mogu biti izvršeni od strane korisnika u roku od sat vremena. Evo i primera:

```
mysql> GRANT SELECT ON *.* TO sarah WITH MAX_QUERIES_PER_HOUR 5;
```

Slika 23: Korišćenje klauzule MAX_QUERIES_PER_HOUR

Klauzula MAX_QUERIES_PER_HOUR kontroliše broj upita koji mogu biti prihvaćeni u roku od sat vremena, ovo se odnosi na SELECT, INSERT, UPDATE, DELETE i druge upite. Takođe moguće je ograničiti broj upita kojima se mogu izmeniti podaci u bazi, sa MAX_UPDATES_PER_HOUR klauzulom kao u sledećem primeru:

```
mysql> GRANT SELECT ON *.* TO sarah WITH MAX_UPDATES_PER_HOUR 5;
```

Slika 24: Korišćenje klauzule MAX_UPDATES_PER_HOUR

Broj novih konekcija koje mogu biti otvorene od strane imenovanih korisnika u roku od sat vremena može se kontrolisati klauzulom MAX_CONNECTIONS_PER_HOUR, kao što je pokazano u sledećem primeru:

```
mysql> GRANT SELECT ON *.* TO sarah WITH MAX_CONNECTIONS_PER_HOUR 3;
```

Slika 25: Korišćenje klauzule MAX_CONNECTIONS_PER_HOUR

Takođe, ove klauzule se mogu koristiti i u kombinaciji:

```
mysql> GRANT SELECT, INSERT, UPDATE, DELETE ON *.* TO supervisor WITH
MAX_QUERIES_PER_HOUR 50 MAX_UPDATES_PER_HOUR 10 MAX_CONNECTIONS_PER_HOUR 4;
```

Slika 26: Kombinovanje klauzula

Ovakva vrsta ograničenja ne može biti specificirana nad celokupnom bazom podataka ili tabelom. Jedino u globalnom kontekstu sa ON *.* klauzulom u GRANT komandi. U sledećem primeru MySQL obaveštava korisnika da je ograničenje isteklo:

```
mysql> INSERT INTO customers (id, name) VALUES (2892, 'Iola J');
ERROR 1226: User 'sarah' has exceeded the 'max_questions' resource
(current value: 5)
```

Slika 27: Obaveštavanje korisnika da je ograničenje isteklo

Server čuva interne brojače, na bazi svakog korisnika, i to za svaki od pomenuta tri ograničenja. Ovi brojači mogu biti resetovani u svakom trenutku novom FLUSH USER_RESOURCES komandom, kao u primeru:

```
mysql> FLUSH USER_RESOURCES;
Query OK, 0 rows affected (0.00 sec)
```

Slika 28: Komanda FLUSH_USER_RESOURCES

Pregled privilegija korisnika

MySQL omogućava pregled privilegija koje su dodeljene pojedinačnom korisniku komandom SHOW GRANTS, koja prihvata korisničko ime kao argument i prikazuje listu svih privilegija dodeljenih korisniku. [3] Sledeći primer to ilustruje:

```
mysql> SHOW GRANTS FOR sarah@localhost;
+-------------------------------------------+
| Grants for sarah@localhost                |
+-------------------------------------------+
| GRANT USAGE ON *.* TO 'sarah'@'localhost'  |
|  IDENTIFIED BY PASSWORD '4837a3954ee01ece' |
| GRANT SELECT ON sales.customers TO         |
| 'sarah'@'localhost'                        |
+-------------------------------------------+
2 rows in set (0.00 sec)
mysql> SHOW GRANTS FOR root;
+-----------------------------------------------------------+
| Grants for root@%                                         |
+-----------------------------------------------------------+
| GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT OPTION |
+-----------------------------------------------------------+
1 row in set (0.06 sec)
```

Slika 29: Korišćenje komande SHOW GRANTS

4. Zaključak

MySQL dolazi sa petoslojnim sistemom za kontrolu pristupa, pružajući potpunu kontrolu nad privilegijama svakog korisnika posebno koje poseduje nad bazama podataka, tabeama i individualnim poljima. Kroz prethodna poglavlja diskutovano je o kontroli pristupa i sistemu za dodelu privilegija korisnicima, objašnjavajući grant tabele kao i GRANT i REVOKE komande koje se koriste za vođenje istih. Takođe pomenuto je i kako se ograničava korišćenje resursa koje se može primeniti nad serverom za svakog korisnika posebno. Na kraju, prikazane su naredbe pomoću kojih se može izvršiti pregled svih privilegija za svakog pojedinačnog korisnika.

5. Literatura

[1] Web sajt: https://en.wikipedia.org/wiki/Database_security
[2] Web sajt: https://dev.mysql.com/doc/mysql-security-excerpt/5.7/en/security.html
[3] http://repository.root-me.org/Administration/Database/EN%20-%20MySQL%20security,%20access%20control%20and%20privileges.pdf