1 Uživatele

1.1 Databázová bezpečnost teorie

Základní princip oprávnění, který MySQL používá je v zásadě totožný s většinou jiných databázových systémů. Funguje tak, že pokud chcete s databází pracovat, je třeba se nejprve identifikovat pomocí **uživatelského jména**. Toto uživatelské jméno je samozřejmě **unikátní**, takže se nemůže stát, že by dva uživatelé na jednom databázovém serveru měli stejné uživatelské jméno.

Pozn.: Na druhou stranu je možné, že se pod jedním uživatelským účtem vytvoří několik připojení k databázi. Ty mají potom z hlediska oprávnění rovnocenné podmínky.

Dalším bezpečnostním prvkem je heslo. Tady už teoreticky je možné, aby dva různí uživatelé měli heslo stejné; i když v praxi to nastane zřídkakdy. Je rovněž teoreticky možné heslo nepoužívat, ale to podstatně snižuje bezpečnost systému. Na základě kombinace uživatelského jména a hesla je rozumným způsobem zajištěno, že se přihlašuje opravdu ten, kdo heslo obdržel. I když s jistotou se to tvrdit nedá, heslo moho být zcizeno.

Následně se přihlášený uživatel pokusí vykonat na databázi nějaký příkaz - vybrat data, manipulovat s daty nebo spravovat databázi. Před každým takovým příkazem jsou zkontrolována oprávnění a buď je akce povolena, nebo je zakázána.

Uvedený přístup funguje obdobně u naprosté většiny databází. MySQL má ještě rozšíření v tom smyslu, že jako součást přihlašovacích informací může DBMS brát v úvahu počítač, z něhož se k databázi připojujete. Tak tedy například uživatel "franta" může se svým heslem přistupovat k serveru z adresy 192.168.0.1, ale tentýž uživatel nemůže přistupovat ze stroje 192.168.0.2. Tím se možnosti zabezpečení dosti rozšiřují - lze například zakázat nebo povolit přístup k databázi z místní sítě nebo z internetu.

1.1.1 Rozlišení práv

Co není povoleno, je zakázáno

V moderních databázových systémech samozřejmě nejde jen o to uživateli něco povolit nebo zakázat, ale jde i o to, jak jemné může takové nastavení být. Proto se dají oprávnění vymezit nejen pro celý server, ale i pro jednotlivé drobnější celky. MySQL nezůstává v tomto trendu pozadu, takže pro uživatele lze zajistit následující věci:

- Přiřadit mu práva pro celý server
- Přiřadit mu práva pro jednu nebo více databází (z mnoha existujících)
- Přiřadit mu práva pro jednu nebo více tabulek v databázi (z mnoha existujících)
- Přiřadit mu práva pro jeden sloupec v tabulce
- Přiřadit mu práva vytvářet a používat **pohledy** (jen MySQL 5.x)
- Přiřadit mu práva vytvářet, měnit nebo spouštět uložené **procedury** (jen MySQL 5.x)

O jaká práva se konkrétně jedná? Pokud jde o data, lze povolit nebo zakázat, že uživatel může:

- Prohlížet (používat SELECT)
- Vkládat (používat INSERT)
- Upravovat (používat UPDATE)
- Odstraňovat (používat DELETE)

Zvláštní práva:

- Měnit práva uživatelů
- Nastavovat systém

V případě databází existují práva vytvářet či měnit tabulky, měnit sloupce v tabulkách či vytvářet dočasné tabulky. Existují práva pro správu, umožňující měnit práva ostatním uživatelům nebo měnit nastavení systému. Je toho hodně, ale je v tom systém. Idea je ta, že prostřednictvím kombinace práv by mělo být možné povolit nebo zakázat prakticky cokoli.

Databáze MySQL ukládá oprávnění - jak asi tušíte - do zvláštní, systémové databáze. Ta se **jmenuje *mysql* a je vytvořena při instalaci serveru**. Tato databáze obsahuje několik tabulek, které ukládají oprávnění uživatelů pro globální přístup, pro jednotlivé databáze, tabulky a sloupce. Jak uvidíme následně, oprávnění jsou z tabulek načítána jako klasická data, což znamená, že přímou změnou těchto tabulek pomocí akčních dotazů můžeme oprávnění spravovat.

Uvidíme ale také, že existuje sada příkazů SQL, která to udělá za nás s mnohem větším komfortem!

1.1.2 Praxe

Běžný uživatel při práci s MySQL přiliš nenarazí na oprávnění. U hostingu jemnohdy jedna databázi, a v ní bude moci provozovat jakoukoli myslitenou akci (obvykle kromě smazání databáze samotné). Uživatelé jsou tedy omezeni na "svoji" databázi a mohou v ní vytvářet a měnit tabulky, pracovat s daty a vytvářet pohledy, funkce a procedury. Zároveň většinou nemají přístup ke konfiguraci systému.

I firem vám nastaví správci.

Proto se uživatelům budeme věnovat jen z rychlíků.

Na FI jsou určitě k dispozici kurzy jen na nastavení a správu databází.

1.2 Tabulky nad databázi MySQL

Veškerá oprávnění jsou v MySQL uložena ve zvláštní databázi, která se jmenuje mysql. V ní je pět tabulek, které s oprávněním tak či onak souvisejí. MySQL při požadavku na provedení nějaké operace tyto tabulky dotazuje, a podle výsledku se rozhodne, zda dané oprávnění přidělí či nikoli. Těmito pěti tabulkami jsou:

- **user**
- **db**
- **host**
- **func**
- **columns_priv**
- **tables_priv**

V zásadě se dá říci, že se s jejich obsahem dá za určitých okolností manipulovat přímo.

Pro experimenty je dobré v tomto případě mít k dispozici testovací server s administrátorským přístupem.

MySQL provádí v podstatě dvojí ověřování.

- První se nazývá ***ověření při připojení*** a stanovuje kdo se odkud smí nebo nesmí k našemu serveru připojit. Pokud to projde nastupuje
- druhá fáze, nazývaná ***ověření při požadavku***. Ta poskytuje uživatelům ověřeným při připojení práva pro konkrétní akce na serveru (jako je třeba vkládání dat do tabulek).

Pozn.: V těchto dvou věcech je podstatný rozdíl. Je klidně možné nakonfigurovat správu uživatelů tak, aby se někdo mohl připojit, ale potom nemohl provést prakticky žádnou myslitelnou akci na serveru.

1.2.1 tabulka user

V databázi mysql v tabulce **user** jsou uložena oprávnění pro uživatele, která platí na **úrovni celé databáze**, proto dávejte pozor, co zde komu dovolíte.

- **host** - IP adresa, nebo název počítače, odkud se uživatel může připojit, pokud je vám to jedno, použijte znak **%** jako zástupný symbol pro cokoliv.
- **User** - Jméno uživatele.
- **Password** - Heslo zahashované funkcí PASSWORD().
- Následující položky můžou nabývat hodnot **Y** a **N**, jako výchozí hodnota je nastaveno **N**.
- **Select_priv** - Vypisování řádků z tabulky.
- **Insert_priv** - Vkládání řádků do tabulky.
- **Update_priv** - Update řádků v tabulce.
- **Delete_priv** - Odstraňování řádků z tabulky.
- **Create_priv** - Vytváření databází a tabulek.
- **Drop_priv** - Mazání databází a tabulek.
- **Reload_priv** - Znovunačtení tabulky přístupových oprávnění.
- **Shutdown_priv** - Vypínaní serveru.
- **Process_priv** - Prohlížení informací o vláknech serveru
- **File_priv** - Přístup k souborům na serveru, číst můžete jen soubory čitelné všemi a vytvářet jen nové soubory tzn. že nemůžete přepsat existující soubor.
- **Grant_priv** - Udělovaní oprávnění jiným účtům.
- **References_priv** - Rezervováno pro budoucí použití.
- **Index_priv** - Vytváření/odstraňování indexů.
- **Alter_priv** - Změny struktury tabulky.
- **Show_db_priv** - Zobrazovat informace o databázích *(SHOW DATABASES)*
- **Super_priv** - Superuživatelské operace, práce s vlákny apd...
- **Create_tmp_table_priv** - Vytváření dočasných tabulek
- **Execute_priv** - Spouštění uživatelských procedur *(zatím se nepoužívá, kdyžtak mě opravte ;-)*
- **Lock_tables_priv** - Povolí nastavení zámku proti čtení/zápisu.
- **Pár dalších** (dle verze) - další položky, ale ty se nevztahují k přímo k přístupovým právům.

1.2.2 Tabulka DB

Zde jsou uložena oprávnění k jednotlivým databázím, struktura je v podstatě stejná u jako u tabulky user.

1.2.3 Tabulka host

Zde můžete měnit oprávnění pro různé databáze, podle toho, odkud se uživatel přihlásil.

1.2.4 Tabulka columns_priv

Práva pro jednotlivé sloupce tabulek.

1.2.5 Tabulka tables_priv

Práva pro jednotlivé tabulky.

1.3 Různá oprávnění

Z*zde je malý výpis oprávnění, která můžete použít*

- ***ADMINISTRATORSKA***
  - ALL PRIVILEGES- jak jsme předvedli výše – MySQL uživatel má přístup do všech databází (kromě systémových databází)
  - GRANT OPTION- povolí uživateli přiřazovat, nebo odstraňovat oprávnění
- UŽIVATELSKÁ
  - CREATE TEMPORARY TABLES – Vytvářet dočasné tabulky
  - FILE - Práce se soubory na serveru
  - LOCK TABLES - Povoluje explicitní použití příkazu LOCK TABLES
  - PROCESS - Získávání informací o vláknech.
  - RELOAD - Znovunačítání přístupových oprávnění apd.
  - SHOW DATABASES - Povoluje prohlížet si všechny databáze.
  - SHUTDOWN - Ukončovat práci serveru
  - SUPER – Ukončovat vlákna

Právo	**Vztahuje se na**	**Popis**
SELECT	tabulky, sloupce	Povoluje vybírat záznamy z tabulek.
INSERT	tabulky, sloupce	Povoluje vkládat nové řádky do tabulky.
UPDATE	tabulky, sloupce	Povoluje obnovovat hodnoty záznamů v tabulkách.
DELETE	Tabulky	Povoluje odstraňovat záznamy (řádky) z tabulek.
INDEX	Tabulky	Povoluje vytvářet a odstraňovat indexy z tabulek.
ALTER	Tabulky	Povoluje měnit strukturu stávajících tabulek (přejmenovávat nebo přidávat sloupce, měnit datové typy sloupců).
CREATE	databáze, tabulky	Povoluje vytvářet nové databáze a tabulky.
DROP	databáze, tabulky	Povoluje odstranit databáze a tabulky.

1.4 Aktualizace dat

Teď by jste asi rádi věděli, jak se s tím pracuje? Mohlo by vás napadnout, že , když je to všechno v tabulkách, že by jsme s nimi mohli pracovat obyčejně, jako s jinými daty. Ano samozřejmě, že to taky jde, ale server MySQL si tyto informace udržuje v paměti, pokud provedete změnu v tabulce příkazem

UPDATE, musíte spusit příkaz **FLUSH PRIVILEGES** *(dostupný od mysql verze 3.22)*, který serveru řekne, aby si znovu načetl tabulky přístupových oprávnění.

1.5 Tipy Triky

- V tabulce **user** smíte mít jednoho uživatele vícekrát. Například budete chtít, aby se franta mohl připojit i z hostitele 10.0.0.5. Tudíž je jedině logické, že pro jednoho uživatele a dva hostitele budou v tabulce user dva řádky. To není chyba
- Ve výše popsaném případě smí mít tentýž uživatel přistupující k serveru z různých míst různá hesla. Je to sice hezká funkce, ale moc se to nepoužívá. Důvodem je to, že by si uživatel musel na každém stroji pamatovat jiné heslo, což může být poněkud náročné.
- MySQL dovolí nechat políčko *host* prázdné. V zásadě to znamená, že daný hostitel se pak může připojit odkudkoli. Nedoporučuji to. Jednak to asi není to, co chcete, a jednak pak není jasné, zda jste hostitele pouze zapomněli specifikovat nebo zda má být prázdný.
- MySQL dovolí nechat prázdné políčko pro heslo. Význam je takový, že heslo pak není pro spojení vůbec požadováno. To sice rovněž nelze doporučit, ale v praxi se to někdy používá, třeba pro lokální přístup.
- Hostitele lze zadat mnoha způsoby, příklady jsou dostupné v dokumentaci. Já jsem si například zapamatoval, že při připojení z místního PC lze zadat jako název hostitele *localhost*. Za určitých podmínek lze rovněž zadat více hostitelů - například jednu podsíť.

- Pochopitelně nesmíte zadat stejnou kombinaci uživatele a hostitele do tabulky user vícekrát. Jednak je to logický nesmysl, a jednak je na tabulce user na sloupcích user a host primární klíč - a ten jak víte musí být unikátní.
- Smíte zadat prázdný název uživatele. To má význam ten, že pak povolujete anonymní přihlášení. Moc se to nepoužívá. Lze samozřejmě povolit anonymní přihlášení pouze z určitých hostitelů pomocí prázdného jména a zadaného hostitele.

1.6 Prohlídnout uživatelé a práva

1.6.1 Uživatelé

Tady platí, že pro ověření při připojení se používají údaje, které jsou uloženy v tabulce user. Pokud jste přihlášen jako superuživatel, můžete si je prohlédnout pomocí

Prohlédnout uživatele výpisem

```
use mysql;
select * from user;
```

nebo ještě stručněji - protože nás v tuto chvíli zajímají pouze tři pole - pomocí něčeho jako

```
select Host, user, … from user;
```

Jak asi tušíte, ověření při připojení zjišťuje, zda daný uživatel se smí připojit z daného hostitele - a zda přitom použil správné heslo. Pokud je všechno v pořádku, pustí MySQL uživatele dál. Pokud ne, skončíte při přihlašování chybou ve stylu:

ERROR 1045 (28000): Access denied for user 'xxx'@'host' (using password: NO)

1.6.2 Opravnění GRANT – jaká jsou

Jaká oprávnění

```
show grants;
SHOW GRANTS FOR 'root'@'localhost';
SHOW GRANTS FOR CURRENT_USER;
SHOW GRANTS FOR CURRENT_USER();
```

Jaká oprávnění můžeme nastavovat

```
SELECT * FROM INFORMATION_SCHEMA.USER_PRIVILEGES WHERE GRANTEE LIKE
"%"
```

Jaká oprávnění můžeme nastavovat – lépe a jednodušeji

```
show privileges;
```

1.7 Vytvořit uživatele

Uživatele si volte své učo. Opatrně, už zasahujeme moc do databáze, tak asi brzo shodíme!
Omezení:

- Nemůžeme nastavit (můžeme a prosím nenastavovat) root přes heslo, takže si nevyzkoušíme přihlášení přes heslo.
- Jsou omezená práva (už nejsou tak benevolentní jak před pár lety), lze změnit pokud člověk ví co a jak, ale slušnost! Pořad v IT funguje

Pokud člověk dělá pro open komunitu tak se mu prodejní věci nehackujou !

1.7.1 Uživatel pod local host

Vytvoření uživatele je tak jednoduché, že na to zase stačí jeden příkaz:

Uživatel tvorba

```
CREATE USER 'uco'@'localhost' IDENTIFIED BY 'heslo';
select Host, user, … from user;
```

V tuto chvíli máte vytvořeného uživatele, který se může **z lokálního stroje(!)** přihlásit a...to je skoro vše. Ještě jsme mu nedali práva k žádné databázi. Heslo je heslo

1.7.2 Uživatel odudkoli nebo jinak

Odkud se uživatel přihlašuje nám určuje řetězec za @ - pokud je
zde '**localhost**' nebo '**127.0.0.1**' je přihlášení možné jen z lokálního stroje.
Zadat zde můžeme jak

- *jméno počítače*, tak
- *IP adresu*. Chceš-li vytvořit uživatele, který se může přihlašovat
- *odkudkoliv*, je třeba příkaz trošku upravit

Uživatel tvorba

```
CREATE USER 'uzivatel'@'%' IDENTIFIED BY 'heslo';
```

Právě znak procenta (%) MySQL serveru říká, že tento uživatel se může přihlásit z jakéhokoliv počítače.

1.7.3 Uživatel zapsat přímo do tabulky user

Do tabulky *user* může superuživatel (jsme máme práva) zapisovat přímo. Můžete tak například založit uživatele *franta* s heslem *123*, který bude smět přistupovat k serveru z hostitele 192.168.0.100. První, co Vás asi napadne je provést to pomocí INSERT:

```
insert into user (user, host, password) values
('franta','192.168.0.100','atnarf');
```

To nebude fungovat!

- MySQL neukládá hesla jako prostý text, ale - zjednodušeně řečeno - ukládá jejich hash. Ten lze vytovřit pomocí MySQL funkce PASSWORD (MD5),
- MySQL přejmenovává sloupce v tabulkách.
- Třetím důvodem je tentokrát mechanismus, jakým MySQL obsluhuje mezipaměť. Je třeba přikázat databázi, aby nový obsah tabulky user zapsala na disk. To provedete pomocí příkazu flush

Prosím netestovat!

```
FLUSH PRIVILEGES;
```

K čemu dojde? MySQL vyprázdní cache a znovunačte oprávnění.

1.8 Přiřadit práva GRANT

1.8.1 Teorie

Použití GRANT je mnohem pružnější než přímý zápis do tabulek s oprávněními nejméně z následujících důvodů:

1. Jeho použití nevyžaduje FLUSH PRIVILEGES. Nemělo by!
2. Heslo se nemusí hashovat pomocí PASSWORD.
3. Při změně formátu tabulek funguje příkaz GRANT konzistentně.
4. Narozdíl od vkládání do tabulky pomocí INSERT lze GRANT bezpečně libovolněkrát opakovat.
5. V jednom příkazu GRANT lze vložit více uživatelů nebo více kombinací uživatel+hostitel.
6. GRANT je mnohem rychlejší (připravené ukázky – hledám co chci)

1.8.2 GRANT syntaxe

Příkaz, který slouží pro přidávaní uživatelů a nastavení jejich práv. V MySQL je od verze 3.22.

```
GRANT práva [sloupce]
ON položka_kde
TO uživatel [IDENTIFIED BY 'heslo' ]
[REQUIRE volby_ssl]
[WITH [GRANT OPTION | volby_omezení] ]
```

Pro připomenutí:

- **ALL** - Může všechno.
- **ALTER** - Měnit strukturu tabulky.
- **CREATE** - Vytvářet tabulky a databáze.
- **DELETE** - Mazat záznamy z tabulek.
- **DROP** - Mazat tabulky a databáze.
- **FILE** - Práce se soubory na serveru.
- **INDEX** - Vytvářet a odstraňovat indexy.
- **INSERT** - Vkládat řádky do tabulek.
- **PROCESS** – Získávání informací o vláknech.
- **REFERENCES** - Nepoužívá se.
- **RELOAD** - Znovunačítání přístupových oprávnění apd.
- **SELECT** - Vybírat záznamy z tabulek.
- **SHUTDOWN** - Ukončovat práci serveru.
- **UPDATE** - Aktualizování řádků v tabulce.
- **USAGE** - Nemá žádná oprávnění.

1.8.3 Přidat jim práva databázi -prakticky

Už máme databázi i uživatele, takže nám zbývá jen je spojit neboli přiřadit uživateli **učo** práva k databázi **učo**:

Vytvořte si databazi

```
GRANT ALL ON uco.* TO 'uco'@'localhost'
```

Vytvoření uživatele a přiřazení práv můžeme spojit do jednoho příkazu:

```
GRANT ALL ON databaze.* TO 'uco'@'localhost' IDENTIFIED BY 'heslo';
```

S právy k databázím není žádná sranda a je dobré je přidělovat s rozumem. Pro bližší nastudování koukněte do dokumentace.

1.8.4 Omezení práv

```
GRANT SELECT, INSERT, DELETE ON database.* TO 'uco'@'localhost';
```

- `GRANT` – vytvoření uživatele a přidělení práv.
- `ALL PRIVILEGES` - - všechna práva, zde můžeme omezit.
- `dbtest.*` - pro které databázi a jejich tabulky – hvězdička všechno
- `TO 'user'@'hostname'` - 'user' - uživatel a kde (jak jsme si říkaly u tvorby užiivatele (u nás pro lokalhost)
- `IDENTIFIED BY 'password'` - heslo

```
databaze.studenti /* v databázi databaze tabulka studenti */
databaze.*     /* všechny tabulky v databaze */
*.studenti       /* tabulka studneti ve všech databázích */
```

Někdy je potřeba provést i FLUSH PRIVILEGE** … ač dokumentace uvádí něco jiného

1.9 Změnit práva uživatele

1.9.1 Práva uživatele

K tomu slouží příkaz **REVOKE**,

1.9.2 Syntaxe

```
REVOKE práva (sloupce) ON kde FROM kdo;
```

Položky práva, kde a kdo jsou stejné jako u příkazu **GRANT**.

Smazání

```
REVOKE ALL PRIVILEGES FROM 'uco…'@'localhost';
```

neumí však vymazat účet, to musíte provést ručně příkazem **DELETE**, potom nezapomeňte ještě zadat příkaz **FLUSH PRIVILEGES**, aby si server znovu načetl tabulku oprávnění do paměti.

1.9.3 Heslo uživatele

```
SET PASSWORD FOR 'uco…'@'localhost' = PASSWORD('tajne_heslo');
```

1.10 Smazat uživatele

No a stejně jako když mažete databázi příkazem DROP, tak i uživatele můžete smazat příkazem

Smazání

```
DROP USER 'uživatel'@'localhost';
```

1.11 Obnovení

```
FLUSH PRIVILEGES;
```

1.12 Uživatel v PHPMyAdmin

>> User accounts

>> Add user account
Uco…
Localhost
123
123
Zaškrtnout veškerá práva

2 Další podklady

http://dev.stderr.cz/2010/12/17/mysql-pridani-uzivatele-a-databaze-a-par-poznamek/