Title

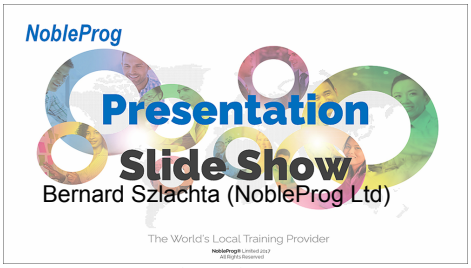
MariaDB Administration Part 2 - Security and Privilege System

Author

Bernard Szlachta (NobleProg Ltd)

Subfooter

MariaDB Administration Part 2 - Security and Privilege System



Slide Show

Contents

Privilege System Overview

MariaDB privilege system:

- authenticates a user who connects from a given host
- associates that user with privileges on a database such as SELECT, INSERT, UPDATE, and DELETE.

Anonymous user handling is deprecated and will be skipped in this presentation.

CHECKING USER PRIVILEGES

show grants for a specific account

```
SHOW GRANTS FOR hitra@localhost;;
```

Show grants for a current account

```
SHOW GRANTS;
```

What you cannot do with the privilege system

- You cannot explicitly specify that a given user should be denied access.
- You cannot specify that a user has privileges to create or drop tables in a database but not to create or drop the database itself.
- A password applies globally to an account. You cannot associate a password with a specific object such as a database, table, or routine.

MariaDB Permission System

- There are only three statements CREATE USER, GRANT, and REVOKE
- privilege information are stored in the grant tables of the mysql database
- server reads the contents of these tables into memory when it starts and bases access-control decisions on the in-memory copies of the grant tables (you can synchronize them with flush statement)
- The MySQL privilege system ensures that all users may perform only the operations allowed to them
- During connection your identity is determined by:
    - the host from which you connect
    - the user name you specify

Users and hosts

The same username can have totally different permission depend on the host they connect from:

```
SHOW GRANTS FOR 'joe'@'office.example.com';
SHOW GRANTS FOR 'joe'@'home.example.com';
```

Permission Control Stages

Stage 1 (Authentication, Connection Verification)
: The server accepts or rejects the connection based on your identity and whether you can verify your identity by supplying the correct password.

Stage 2 (Authorization, Request Verification)
: Assuming that you can connect, the server checks each statement you issue to determine whether you have sufficient privileges to perform it.

MySQL Privileges

Administrative

- enable users to manage operation of the MySQL server.
- are global, not specific to a particular database.

Database

- apply to a database and to all objects within it
- can be granted for specific databases, or globally so that they apply to all databases.

Privileges for database objects such as tables, indexes, etc.. can be granted:

- for specific objects within a database,

- for all objects of a given type within a database (for example, all tables in a database)
- globally for all objects of a given type in all databases).

Detailed Privileged

- ALL or ALL PRIVILEGES specifier is shorthand for “all privileges available at a given privilege level” (except GRANT OPTION)

* For example, granting ALL at the global or table level grants all global privileges or all table-level privileges.

http://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html

Showing All Users

```
select distinct concat(user,'@',host) from mysql.user;
```

Specifying Account Names

appears in CREATE USER, GRANT, SET PASSWORD

```
'user_name'@'host_name'
```

- 'me' is equivalent to 'me'@'%'
- Quotes are necessary to specify:
  - a user_name string containing special characters (such as “-”),
  - a host_name string containing special characters or wildcard characters (such as “%”); for example, 'test-user'@'%.com'
- backticks (“`”), single quotes (“'”), or double quotes (“"”) can be used
- Write 'me'@'localhost', not 'me@localhost'; the latter is interpreted as 'me@localhost'@'%'
- The anonymous account is *@'localhost'*.

Specifying Host

- Host name or an IP number 'localhost','127.0.0.1'
- Wildcard characters “%” and “_”, '%.mysql.com' ,'192.168.1.%'
- Someone could try to exploit this capability by naming a host 192.168.1.somewhere.com
- To foil such attempts, host names that start with digits and a dot are not valid. Hosts like 1.2.example.com never matches the host part of account names. An IP wildcard value can match only IP numbers, not host names
- host_ip/netmask,'david'@'192.58.197.0/255.255.255.0'
- client_ip & netmask = host_ip
- Only A, B or C network classes are valid
- The following netmask (28 bits) will not work 192.168.0.1/255.255.255.240

Multiple Matches

When multiple matches are possible, the server must determine which of them to use.

It resolves this issue as follows:

- Whenever the server reads the user table into memory, it sorts the rows.
- When a client attempts to connect, the server looks through the rows in sorted order.
- The server uses the first row that matches the client host name and user name.
- It orders the rows with the most-specific Host values first. Literal host names and IP addresses are the most specific.
- Anonymous user is the least specific user, therefore is the last match

Multiple Matches and Sorting

```
+-----------+----------+
| Host      | User     |
+-----------+----------+
| %         | root     |
| %         | jeffrey  |
| localhost | root     |
| localhost |          |
+-----------+----------+
```

```
+-----------+----------+
| Host      | User     |
+-----------+----------+
| localhost | root     |
| localhost |          |
| %         | jeffrey  |
| %         | root     |
+-----------+----------+
```

Quiz - which user is going to be chosen

```
+----------------+----------+-
| Host           |     User |
+----------------+----------+-
| %              | bernard  |
| nobleprog.com  |          |
+----------------+----------+-
bernard@nobleprog.com
```

When Privilege Changes Take Effect

- When MariaDB starts, it reads all grant table contents into memory.
- GRANT, REVOKE, or SET PASSWORD, the server loads the grant tables into memory again immediately.
- If you modify the grant tables directly using statements such as INSERT, UPDATE, or DELETE, your changes have no effect on privilege checking until you either restart the server or tell it to reload the tables.
- Flushing privileges

```
FLUSH PRIVILEGES
mysqladmin flush-privileges
mysqladmin reload
```

When Privilege Changes Take Effect

When the server reloads the grant tables, privileges for each existing client connection are affected as follows:

- Table and column privilege changes take effect with the client's next request
- Database privilege changes take effect the next time the client executes a USE db_name statement
- Global privileges and passwords are unaffected for a connected client. These changes take effect only for subsequent connections.
- If the server runs with --skip-grant-tables option, it does not read the grant tables or implement any access control. Anyone can *connect and do anything. Flush the privileges to enforce privileges

Making privileges simple

If not sure use CURRENT_USER() to determine which row has been matched

MODULE 5 - ACCOUNT MANAGEMENT

MariadDB User Account Management

- Users and Passwords
- Creating New Users
- Deleting User Accounts
- Limiting User Resources
- Changing Passwords

User Names and Passwords

Account is made of:

- user name
- client host or hosts
- password

- Most MySQL clients by default try to log in using the current Unix user name as the MySQL user name
- MySQL user names can be up to 16 characters long
- MySQL encrypts passwords using its own algorithm - PASSWORD() SQL function
- Differs from the Unix password - see ENCRYPT()

```
mysql --user=monty --password=guess db_name
mysql -u monty -pguess db_name
```

Adding User Accounts

You can create MySQL accounts in the following ways:

- CREATE USER or GRANT. These statements cause the server to make appropriate modifications to the grant tables (preferred)
- By manipulating the MySQL grant tables directly with statements such as INSERT, UPDATE, or DELETE
- GUI or other tools (MySQL Admin, phpMyAdmin)

User Creation Examples

```
GRANT ALL  ON *.* TO 'monty'@'localhost' IDENTIFIED BY 'some_pass';
```

Is equivalent to:

```
CREATE USER monty@localhost;
GRANT ALL ON *.* TO monty@localhost
SET PASSWORD FOR localhost = PASSWORD('some_pass');
```

Manual User Creation

```
INSERT INTO user
    VALUES('%','monty',PASSWORD('some_pass'),
    'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y',
    'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y',
    ,,,,0,0,0,0);
INSERT INTO user SET Host='localhost',User='admin',
```

```
    Reload_priv='Y', Process_priv='Y';
INSERT INTO user (Host,User,Password) VALUES('localhost','dummy',);
FLUSH PRIVILEGES;
```

Removing Users

- DROP USER user [, user] ..
- Each user@host pair should be removed
- DROP USER user is equivalent to DROP USER user@'%'
- DROP USER does not automatically close any open user sessions
- DROP USER does not automatically delete or invalidate any database objects that the user created.

Killing User Sessions

```
MariaDB [mysql]> show processlist;
+----+----------------+-----------------+-------+---------+------+-------+----------
| Id | User           | Host            | db    | Command | Time | State | Info
+----+----------------+-----------------+-------+---------+------+-------+----------
|  2 | root           | localhost:50946 | mysql | Query   |    0 | init  | show proc
|  5 | bernard_select | localhost:51044 | test  | Sleep   |  531 |       | NULL
+----+----------------+-----------------+-------+---------+------+-------+----------
2 rows in set (0.00 sec)
```

```
MariaDB [mysql]> drop user bernard_select@localhost;
Query OK, 0 rows affected (0.02 sec)
```

```
MariaDB [mysql]> show processlist;
+----+----------------+-----------------+-------+---------+------+-------+----------
| Id | User           | Host            | db    | Command | Time | State | Info
+----+----------------+-----------------+-------+---------+------+-------+----------
|  2 | root           | localhost:50946 | mysql | Query   |    0 | init  | show proc
|  5 | bernard_select | localhost:51044 | test  | Sleep   |  549 |       | NULL
+----+----------------+-----------------+-------+---------+------+-------+----------
2 rows in set (0.00 sec)
```

```
MariaDB [mysql]> kill 5
    -> ;
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [mysql]> show processlist;
+----+------+-----------------+-------+---------+------+-------+------------------+-
| Id | User | Host            | db    | Command | Time | State | Info             |
+----+------+-----------------+-------+---------+------+-------+------------------+-
|  2 | root | localhost:50946 | mysql | Query   |    0 | init  | show processlist |
+----+------+-----------------+-------+---------+------+-------+------------------+-
1 row in set (0.00 sec)
```

Modes and User Creation

Some of the statements will fail if the server's SQL mode has been set to enable certain restrictions.

- strict mode (STRICT_TRANS_TABLES, STRICT_ALL_TABLES)
- NO_AUTO_CREATE_USER

will prevent the server from accepting some of the statements.

Changing Password

During user creation

```
CREATE USER 'jeffrey'@'localhost' IDENTIFIED BY 'biscuit';
```

Administrators for different user

```
SET PASSWORD FOR 'jeffrey'@'localhost' = PASSWORD('biscuit');
```

Users themselves

```
SET PASSWORD = PASSWORD('biscuit');
GRANT USAGE ON *.* TO 'jeffrey'@'localhost' IDENTIFIED BY 'biscuit';
```

From command line

```
mysqladmin -u user_name -h host_name password "newpwd"
```

Common User Creation Scenarios

```
GRANT ALL ON test.* TO user@localhost IDENTIFIED BY 'password' ;
```

- Creates an user@localhost account if it doesn't exists
- Grants ALL PRIVILEGES on all tables in the test database, even those create after the command was executed.

Exercise

1. Restore File:Tables.sql into NP database
2. Create user nobleproguser@localhost and grant all privilages on the test database
3. Which tables has been modified (in mysql database)?
4. Show the account privileges using "show grants for nobleproguser@localhost" command
5. Grant select permission to a *np* database
6. Login as nobleproguser@localhost and test it
7. Show the account privileges using "show grants for nobleproguser@localhost" command
8. Grant update permission to a NP.emp table
9. Test it by updating an employee
10. *Grant select permission to NP.emp.name column

Limiting Account Resources

max_user_connections system variable limits only the number of simultaneous connections made using a single account Individual accounts can have following limits:

- The number of queries that an account can issue per hour
- The number of updates that an account can issue per hour
- The number of times an account can connect to the server per hour
- The number of simultaneous connections to the server an account can have

Limiting Resources

```
CREATE USER 'francis'@'localhost' IDENTIFIED BY 'frank';
GRANT ALL ON customer.* TO 'francis'@'localhost'
    WITH
   MAX_QUERIES_PER_HOUR  2
    MAX_UPDATES_PER_HOUR 10
    MAX_CONNECTIONS_PER_HOUR 5
    MAX_USER_CONNECTIONS 2;
```

Modifying existing accounts:

```
GRANT USAGE ON *.* TO 'francis'@'localhost'
   WITH MAX_QUERIES_PER_HOUR 100;
```

To remove an existing limit, set its value to zero.

Limiting Resources

- To reset the current counts to zero for all accounts, issue a FLUSH USER_RESOURCES statement.
- The counts also can be reset by reloading the grant tables (for example, with a FLUSH PRIVILEGES statement or a mysqladmin reload command).
- Counter resets do not affect the MAX_USER_CONNECTIONS limit.
- All counts begin at zero when the server starts; counts are not carried over through a restart.

Common Pitfall

1. Create two accounts in the following order: user@localhost and user@127.0.0.0/255.255.255.0
2. Grant the users full permission on test table
3. Login as the user from the local host (mysql –u user)
4. Check the which user you are logged in: select current_user();
5. Drop both accounts and recreate them in the reverse order
6. Follow the step 3 and 4, is the result the same?

Column Privileges

```
grant select (name,sal), insert (sal) on NP.emp to test@localhost;
```

Global, Database, Column Privileges

mysql> SHOW GRANTS FOR hitra@localhost;;

```
+---------------------------------------------------------------------+
| Grants for hitra@localhost                                          |
+---------------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'hitra'@'localhost' IDENTIFIED BY PASSWORD  /
| GRANT ALL PRIVILEGES ON `hitra`.* TO 'hitra'@'localhost'            |
| GRANT SELECT, INSERT ON `hitrapl`.* TO 'hitra'@'localhost‘           |
| GRANT SELECT (nid), INSERT (vid, nid) ON `hitrapl`.`node` TO 'hitra'@'localhost' |
+---------------------------------------------------------------------+
```