



MySQL Security, Privileges & User Management

Kenny Gryp <kenny.gryp@percona.com>
Percona Live Washington DC / 2012-01-11

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ Network Security
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ DoS
- ◆ PCI Compliance

Security, Privileges & User Management

- ◆ **Privilege System**
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ Network Security
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ DoS
- ◆ PCI Compliance

Privilege System

- ◆ Users
- ◆ Grants
- ◆ mysql database
- ◆ Resource Limits
- ◆ Default Permissions

Users

- ◆ Identify users based on: user@host
 - ◆ user: username
 - ◆ host: hostname/ip of the client that connects
 - ◆ different host, different user, different 'grants'
- ◆ Examples:
 - 'fred'@'localhost', 'root'@'localhost'
 - 'lefred'@'app0001', 'kampen'@'192.168.%'
 - 'lekampen'@'192.168.1___', 'fred'@'app.fq.dn'
- ◆ Creating A User:
 - >CREATE USER 'lefred'@'app0001';
- ◆ Drop user: change CREATE into DROP

Grants

- ♦ Grant the user some kind of privilege
- ♦ Grant ... to:
 - server, column, view,
 - database, trigger, index
 - table, stored procedure,
- ♦ Example: `INSERT, SELECT, UPDATE, DELETE`
- ♦ SQL Command:
 - >`GRANT SELECT ON db.* TO 'lefred'@'app0001';`
 - >`GRANT INSERT ON *.* TO 'lefred'@'app0001';`
- ♦ Revoking privileges: change `GRANT` into `REVOKE`

Table/Column Level Grants

- ◆ Possible:
 - > GRANT SELECT ON db.table TO 'lefred'@'app';
 - > GRANT SELECT (col) ON db.table to 'fr'@'app';
- ◆ Too much columns might make authentication slower

Password

◆ Examples:

```
> SET PASSWORD FOR 'lfred'@'app0001' =  
PASSWORD('pass');
```

```
> SELECT PASSWORD('pass')\G  
PASSWORD('pass'):  
*196BDEDE2AE4F84CA44C47D54D78478C7E2BD7B7
```

```
> SET PASSWORD FOR 'lfred'@'app0001' =  
'*196BDEDE2...';
```

```
> CREATE USER 'lfred'@'app0001' IDENTIFIED  
BY 'pass';  
> CREATE USER 'fred'@'app' IDENTIFIED BY  
PASSWORD '*196BDEDE2...';
```

```
> GRANT SELECT ON db.* TO 'fred'@'app'  
IDENTIFIED BY 'pass';
```

Grants

- ◆ Complete list of grants:

CREATE	ALTER ROUTINE
DROP	CREATE ROUTINE
GRANT OPTION	EXECUTE
LOCK TABLES	FILE
EVENT	CREATE USER
ALTER	PROCESS
DELETE	PROXY
INDEX	RELOAD
INSERT	REPLICATION CLIENT
SELECT	REPLICATION SLAVE
UPDATE	SHOW DATABASES
CREATE TEMPORARY TABLES	SHUTDOWN
TRIGGER	SUPER
CREATE VIEW	ALL [PRIVILEGES]
SHOW VIEW	USAGE

Grants

CREATE
DROP
GRANT OPTION
LOCK TABLES
EVENT
ALTER
DELETE
INDEX
INSERT
SELECT
UPDATE
CREATE TEMPORARY TABLES
TRIGGER
CREATE VIEW
SHOW VIEW

ALTER ROUTINE
CREATE ROUTINE
EXECUTE
FILE
CREATE USER
PROCESS
PROXY
RELOAD
REPLICATION CLIENT
REPLICATION SLAVE
SHOW DATABASES
SHUTDOWN
SUPER
ALL [PRIVILEGES]
USAGE

SHOW GRANTS

♦ > **SHOW GRANTS;**

```
+-----+
| Grants for root@localhost
+-----+
| GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' IDENTIFIED BY
| PASSWORD '*81F5E21E35407D884A6CD4A731AEBFB6AF209E1B' WITH GRANT
| OPTION
+-----+
```

♦ > **SHOW GRANTS FOR 'lefred'@'app0001';**

```
+-----+
| Grants for lefred@app0001
+-----+
| GRANT INSERT ON *.* TO 'lefred'@'app0001'
| GRANT SELECT ON `db`.* TO 'lefred'@'app0001'
+-----+
```

GRANT OPTION

- ♦ User with 'GRANT OPTION' can give grants to other users
- ♦ only for the grants he has already

FILE

- ◆ Read/Write Files with:
 - > `SELECT ... INTO OUTFILE`
 - > `LOAD DATA INFILE ...`
- ◆ Are you sure you want to give FILE?
- ◆ Restrict with `secure_file_priv=/path/`

FILE

◆ Example:

```
> CREATE TABLE passwd(user varchar(255),pass varchar(255),userid integer,`group` integer,gecos varchar(255),home varchar(255),shell varchar(255));
```

```
Query OK, 0 rows affected (0.05 sec)
```

```
> LOAD DATA INFILE '/etc/passwd' INTO TABLE passwd FIELDS TERMINATED BY ":";
```

```
Query OK, 40 rows affected (0.05 sec)
```

```
> SELECT user, pass, userid, `group`, gecos FROM passwd;
```

user	pass	userid	group	gecos
root	x	0	0	root
daemon	x	1	1	daemon
bin	x	2	2	bin
sys	x	3	3	sys
sync	x	4	65534	sync
games	x	5	60	games
man	x	6	12	man
lp	x	7	7	lp
mail	x	8	8	mail
news	x	9	9	news

LOAD DATA LOCAL

- ◆ Just like `LOAD DATA`, but takes a file from the client
- ◆ Must have config on server: `local-infile=0`
- ◆ More a security problem on the client:
 - ◆ `local-infile=0` to `[client]`
 - ◆ recompile library with `DENABLED_LOCAL_INFILE=1`

PROCESS

See complete **SHOW PROCESSLIST** for every user

```
> SHOW GRANTS;
```

```
+-----+
| Grants for process@localhost |
+-----+
| GRANT PROCESS ON *.* TO 'process'@'localhost' |
+-----+
```

```
>SHOW FULL PROCESSLIST\G
```

```
...
```

```
***** 3. row *****
```

```
    Id: 6163
```

```
    User: root
```

```
    Host: localhost
```

```
    db: test
```

```
Command: Query
```

```
    Time: 63
```

```
    State: Locked
```

```
    Info: insert into passwd values ('lefred','iLikeDim0',
null,null,null,null)
```

PROCESS

And....:

```
> SHOW ENGINE INNODB STATUS\G
```

```
...
```

```
=====
```

```
120110 21:44:00 INNODB MONITOR OUTPUT
```

```
=====
```

```
Per second averages calculated from the last 37 seconds
```

```
...
```

```
-----
```

```
TRANSACTIONS
```

```
-----
```

```
...
```

```
---TRANSACTION 0, not started, process no 955, OS thread id  
140712801937152
```

```
mysql tables in use 1, locked 1
```

```
MySQL thread id 6163, query id 273 localhost root Table lock
```

```
insert into passwd values ('lefred', 'iLikeDim0',
```

```
null,null,null,null)
```

```
...
```

RELOAD

- ◆ Reload all kinds of log files, not so bad...
- ◆ But:
 - ◆ `FLUSH MASTER`: remove all binary logs
 - ◆ `FLUSH SLAVE`: remove all slave configuration
 - ◆ `FLUSH TABLES WITH READ LOCK`: lock tables

REPLICATION CLIENT

- ◆ `SHOW MASTER STATUS;`
- ◆ `SHOW SLAVE STATUS\G`

REPLICATION SLAVE

- ♦ Required for slave to fetch binlogs

Also gives:

```
SHOW BINLOG EVENTS\G
```

```
...
```

```
  Log_name: mysql-bin.000001
```

```
    Pos: 175
```

```
Event_type: Query
```

```
  Server_id: 9999
```

```
End_log_pos: 312
```

```
  Info: use `test`; insert into passwd  
values ('lefred', 'iLikeDim0',  
null,null,null,null)
```

SHUTDOWN

- ◆ # mysqladmin shutdown

SUPER

- ◆ Known to be given to app users & monitoring users
- ◆ However, it is very powerful:
 - ◆ CHANGE MASTER TO, STOP SLAVE, START SLAVE
 - ◆ KILL any thread
 - ◆ SET GLOBAL ...
 - ◆ BINLOG
 - ◆ When `read_only=on` SUPER users can still write
 - ◆ Set DEFINER with Stored Procedures/Views to account of choice
 - ◆ Have the extra login when `max_connections` is reached

ALL

- ◆ Gives ALL privileges possible (on a certain level):

- > GRANT ALL ON *.*

- > GRANT ALL ON db.*

- > GRANT ALL ON db.table

- ...

USAGE

- ◆ Gives you the possibility to... login
- ◆ Possible to run:
 - ◆ `SHOW GLOBAL STATUS;`
 - ◆ `SHOW GLOBAL VARIABLES;`
 - ◆ Set session buffers/variables (see next chapter)

mysql Database

```
> SHOW TABLES;
```

```
+-----+
| Tables_in_mysql |
+-----+
| columns_priv    |
| db              |
| event           |
| func            |
| general_log     |
| help_category   |
| help_keyword    |
| help_relation   |
| help_topic      |
| host            |
| ndb_binlog_index
```

```
plugin
proc
procs_priv
proxies_priv
servers
slow_log
tables_priv
time_zone
time_zone_leap_second
time_zone_name
time_zone_transition
time_zone_transition_type
user
+-----+
24 rows in set (0.00 sec)
```

mysql Database

- ◆ Do not give rights for app or general users
- ◆ DML statements are possible
 - ◆ use `FLUSH PRIVILEGES` to apply

Resource Limits

- ♦ For every user: `max_user_connections`
- ♦

```
>GRANT USAGE ON db.* TO 'lefred'@'localhost'  
WITH MAX_QUERIES_PER_HOUR 1000  
MAX_UPDATES_PER_HOUR 999  
MAX_CONNECTIONS_PER_HOUR 100  
MAX_USER_CONNECTIONS 5;  
FLUSH USER_RESOURCES;
```
- ♦ Not commonly used

Default Permissions

```
-- Grants dumped by pt-show-grants
-- Dumped from server Localhost via UNIX socket, MySQL 5.5.17-55-
log at 2012-01-11 03:36:25
-- Grants for 'root'@'127.0.0.1'
GRANT ALL PRIVILEGES ON *.* TO 'root'@'127.0.0.1' WITH GRANT
OPTION;
-- Grants for 'root'@'desktop'
GRANT ALL PRIVILEGES ON *.* TO 'root'@'desktop' WITH GRANT OPTION;
-- Grants for 'root'@'localhost'
GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' WITH GRANT
OPTION;
-- Grants for ''@'localhost'
GRANT USAGE ON *.* TO ''@'localhost';

# mysql_secure_installation
Set root password? [Y/n] y
Remove anonymous users? [Y/n] y
Disallow root login remotely? [Y/n] y
Remove test database and access to it? [Y/n] y
```

Security, Privileges & User Management

- ◆ Privilege System
- ◆ **User Management**
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ Network Security
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ DoS
- ◆ PCI Compliance

User Management

- ◆ Difficult to manage when having +1 MySQL server
- ◆ How to properly manage all those users?
 - ◆ Version Control
 - ◆ SecuRich
 - ◆ Configuration Management

Version Control

- ♦ Put all grants in a .txt file and put in VC
- ♦ use pt-show-grants:
 - ♦ orders grants, easy to VC
 - ♦ generates revoke statements

```
-- Grants dumped by pt-show-grants
-- Dumped from server Localhost via UNIX socket, MySQL 5.5.17-55-log at
2012-01-10 23:52:18
-- Grants for 'debian-sys-maint'@'localhost'
GRANT ALL PRIVILEGES ON *.* TO 'debian-sys-maint'@'localhost'
IDENTIFIED BY PASSWORD '*C86BAB1C913CE0D310B662846E830230C51DA954' WITH
GRANT OPTION;
-- Grants for 'lefred'@'app0001'
GRANT INSERT ON *.* TO 'lefred'@'app0001';
GRANT SELECT ON `db`.* TO 'lefred'@'app0001';
-- Grants for 'lefred'@'localhost'
GRANT SELECT, SELECT (user) ON `test`.`passwd` TO 'lefred'@'localhost';
GRANT USAGE ON *.* TO 'lefred'@'localhost';
```

<http://www.percona.com/doc/percona-toolkit/2.0/pt-show-grants.html>

SecuRich

- ◆ Tool (scripts/stored procedures) to facilitate user management
- ◆ Has some features MySQL does not have:
 - ◆ password expiry
 - ◆ block users (even throws out users)
 - ◆ password history
 - ◆ password complexity checks

Configuration Management

- ◆ Use your favorite configuration management tool
- ◆ Puppet example:
<https://github.com/DavidS/puppet-mysql>

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ **Pluggable Authentication**
- ◆ Application Security
- ◆ Network Security
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ DoS
- ◆ PCI Compliance

Pluggable Authentication

- ◆ Feature Since MySQL 5.5
- ◆ New Grant: PROXY: act like a user
- ◆ Percona PAM Plugin:
 - ◆ <http://www.mysqlperformanceblog.com/2011/12/05/announcing-pam-authentication-plugin-for-mysql-early-access-release/>
- ◆ Oracle PAM Plugin: commercial plugin
- ◆ Clear text password will be sent: use secure connections

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ **Application Security**
- ◆ Network Security
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ DoS
- ◆ PCI Compliance

Application Security

- ◆ SQL Injections
- ◆ use `mysql_real_escape_string()`
- ◆ Use Prepared Statements
- ◆ Use different users in the application (read/write/...)
- ◆ Don't give app users permissions they should not have (see this presentation)

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ **Network Security**
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ DoS
- ◆ PCI Compliance

Network Security

- ◆ Port protection
- ◆ Traffic encryption
- ◆ DNS

Port Protection

- ◆ Firewall
- ◆ `bind-address=127.0.0.1`
- ◆ No need for network connections (socket only):
`skip-networking`

Traffic Encryption

- ◆ Problem:

```
# tcpdump -w - -i lo port 3306 | strings  
...  
insert into passwd values  
( 'lefred', 'dim00tjen', null, null, null, null )H
```

- ◆ Solution:

- ◆ Built-in SSL
- ◆ Secure Tunnels

DNS

- ◆ Remember: authentication is `user@host`
- ◆ MySQL does Reverse DNS Lookup
 - ◆ taking over DNS server can change grants
 - ◆ Killing DNS server can cause stalls (next to the default dns cache in MySQL or `nscd`): both security and performance problem
 - ◆ use `skip-name-resolve`

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ Network Security
- ◆ **OS Level Security**
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ DoS
- ◆ PCI Compliance

OS Level Security

- ◆ Security profiles:
 - ◆ AppArmor
 - ◆ SELinux
- ◆ Chroot
 - ◆ start mysqld with `--chroot`

Filesystem Encryption

- ◆ LUKS/ecryptfs/....:
 - ◆ Disk/File/Directory encryption
 - ◆ Protects against 'disk-stealing'
 - ◆ No protection for user 'root'
- ◆ Gazzang ezNcrypt (<http://www.gazzang.com>)
 - ◆ Commercial tool
 - ◆ Uses ecryptfs
 - ◆ Off-Site Key Management
 - ◆ kernel module
 - ◆ ACL
 - ◆ Only certain binary, with a certain hash can access the encrypted files

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ Network Security
- ◆ OS Level Security
- ◆ **Other MySQL Security Features**
- ◆ Data Security Functions
- ◆ DoS
- ◆ PCI Compliance

Other MySQL Security Features

- ◆ `old_passwords`: Insecure 4.1 hashing
 - ◆ set `secure-auth` to avoid
- ◆ `skip-symbolic-links`
- ◆ `max_connect_errors`:
 - ◆ default=10
 - ◆ Error: Host 'host_name' is blocked
 - ◆ `FLUSH HOSTS`
- ◆ `skip-grant-tables`:
 - ◆ ignore authentication
 - ◆ recover lost root password
- ◆ Audit Plugin interface (since 5.5)

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ Network Security
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ **Data Security Functions**
- ◆ DoS
- ◆ PCI Compliance

Data Security Functions

- ◆ `PASSWORD ()`
- ◆ Crypt: `ENCRYPT () / DECRYPT ()`
- ◆ AES: `AES_ENCRYPT () / AES_DECRYPT ()`
- ◆ DES: `DES_ENCRYPT () / DES_DECRYPT ()`
- ◆ Hashing: `MD5 () , SHA2 ()`
- ◆ Statement Based Replication includes the SQL statement in the binary log: Use Row Based
- ◆ Same counts for general/slowlog
- ◆ Maybe encrypt in application

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ Network Security
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ **DoS**
- ◆ PCI Compliance

DoS

```
mysql> show grants;
```

```
+-----+
| Grants for @localhost |
+-----+
| GRANT USAGE ON *.* TO '@localhost' |
+-----+
```

```
1 row in set (0.00 sec)
```

◆ Disk:

```
mysql> use information_schema;
```

```
mysql> select a.* FROM CHARACTER_SETS a, CHARACTER_SETS b,  
-> CHARACTER_SETS c, CHARACTER_SETS d, CHARACTER_SETS e;
```

◆ Memory:

```
mysql> SELECT REPEAT('a', 1024*1024) INTO @a1;
```

```
Query OK, 1 row affected (0.01 sec)
```

```
..
```

```
mysql> SELECT REPEAT('a', 1024*1024) INTO @a99;
```

```
Query OK, 1 row affected (0.01 sec)
```

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ Network Security
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ DoS
- ◆ **PCI Compliance**

PCI Compliance

1. Install and maintain a firewall configuration to protect cardholder data
2. Do not use vendor-supplied defaults for system passwords and other security parameters
3. Protect stored cardholder data
4. Encrypt transmission of cardholder data across open, public networks
5. Use and regularly update anti-virus software on all systems commonly affected by malware
6. Develop and maintain secure systems and applications
7. Restrict access to cardholder data by business need-to-know
8. Assign a unique ID to each person with computer access
9. Restrict physical access to cardholder data
10. Track and monitor all access to network resources and cardholder data
11. Regularly test security systems and processes
12. Maintain a policy that addresses information security

http://en.wikipedia.org/wiki/PCI_DSS

Security, Privileges & User Management

- ◆ Privilege System
- ◆ User Management
- ◆ Pluggable Authentication
- ◆ Application Security
- ◆ Network Security
- ◆ OS Level Security
- ◆ Other MySQL Security Features
- ◆ Data Security Functions
- ◆ DoS
- ◆ PCI Compliance



Kenny Gryp
<kenny.gryp@percona.com>
@gryp

We're Hiring!
www.percona.com/about-us/careers/