MySQL security best practices

A 101 webinar presented by
Dimitri Vanoverbeke

Whoami

Dimitri Vanoverbeke
Solution Engineer in Percona

MySQL Security 101 overview

- Having a security mindset.
  - Infrastructure
  - Operating system
  - Applications
- MySQL privileges
- SSL communication
- Handling ransomware
- Encryption options

Have the correct mindset

- Applications should be written with **security** from the ground up.
- Work together with your **sysadmins** and **devteam** to make the correct choices.
- Disable and restrict remote access
- Understand the **cloud** means working on **other peoples computers**.
- **Restrictive** mindset

Infrastructure: Network

- **Segregate** your network
  - Only **application servers** should be able to connect to the DB **remotely**.
  - Dev **access/general access** should be **limited** by using a bastion/jumphost
  - **DO NOT OPEN IT UP TO THE INTERNET!!!** (or use strict firewall rules)
  - **Example** on https://www.shodan.io/search?query=mysql
- IPS/IDS appliance/software can be handy
  - Snort, Bro Network Security monitor, OSSEC

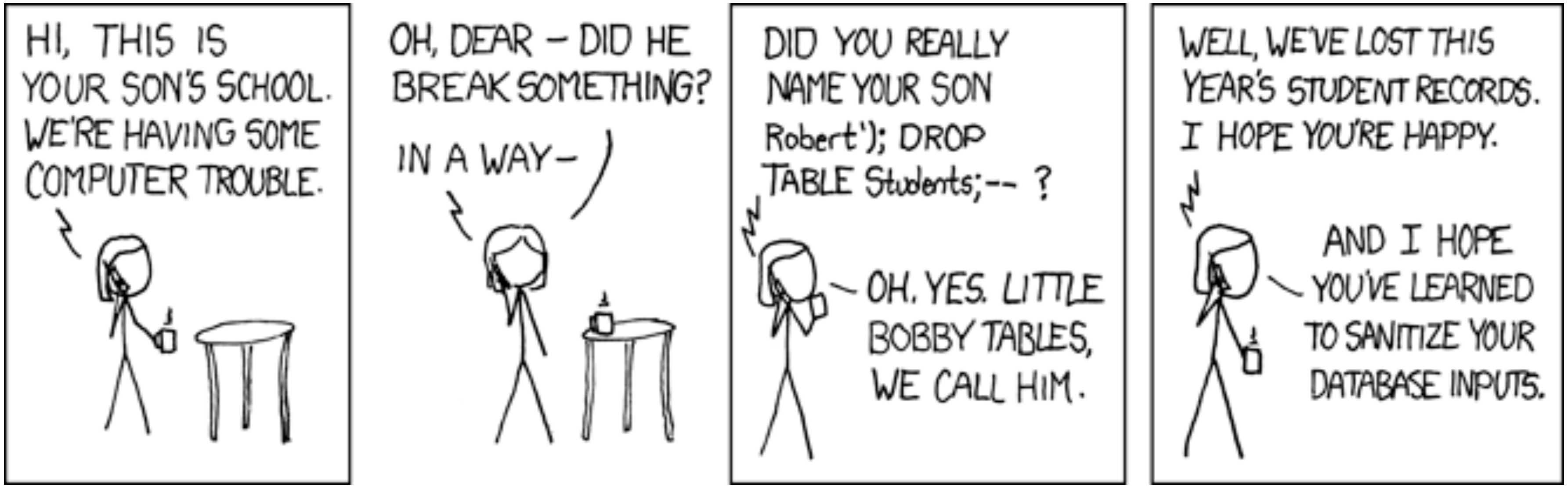
Be friends with your network engineer

Operating system security

- Deploy security patches as soon as possible.
- Make sure permissions are correct:
  - mysql should be the owner
  - Don’t use chmod 777 :-)
  - 0750 dirs , 0640 files
  - selinux setenforce 1
- If PCI compliant?
  - ecryptfs, luks, EBS encryption (starting at medium sizes)
- Use trusted package sources!
- Establishing a patching policy!

Applications

- Perform penetration tests on staging environments.
- Validate user inputs
- Watch out for SQL injections.


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

Use configuration management

- Use your favourite configuration management solution. Tools like puppet, chef, ansible and salt are excellent tools to ensure compliancy:

```
$users = {
  'dim0@localhost' => {
    ensure                   => 'present',
    max_connections_per_hour => '0',
    max_queries_per_hour     => '0',
    max_updates_per_hour     => '0',
    max_user_connections     => '0',
    password_hash            => '*T5D3A5831A93829BE2468926B4132313728C250DBF',
  },
}
```

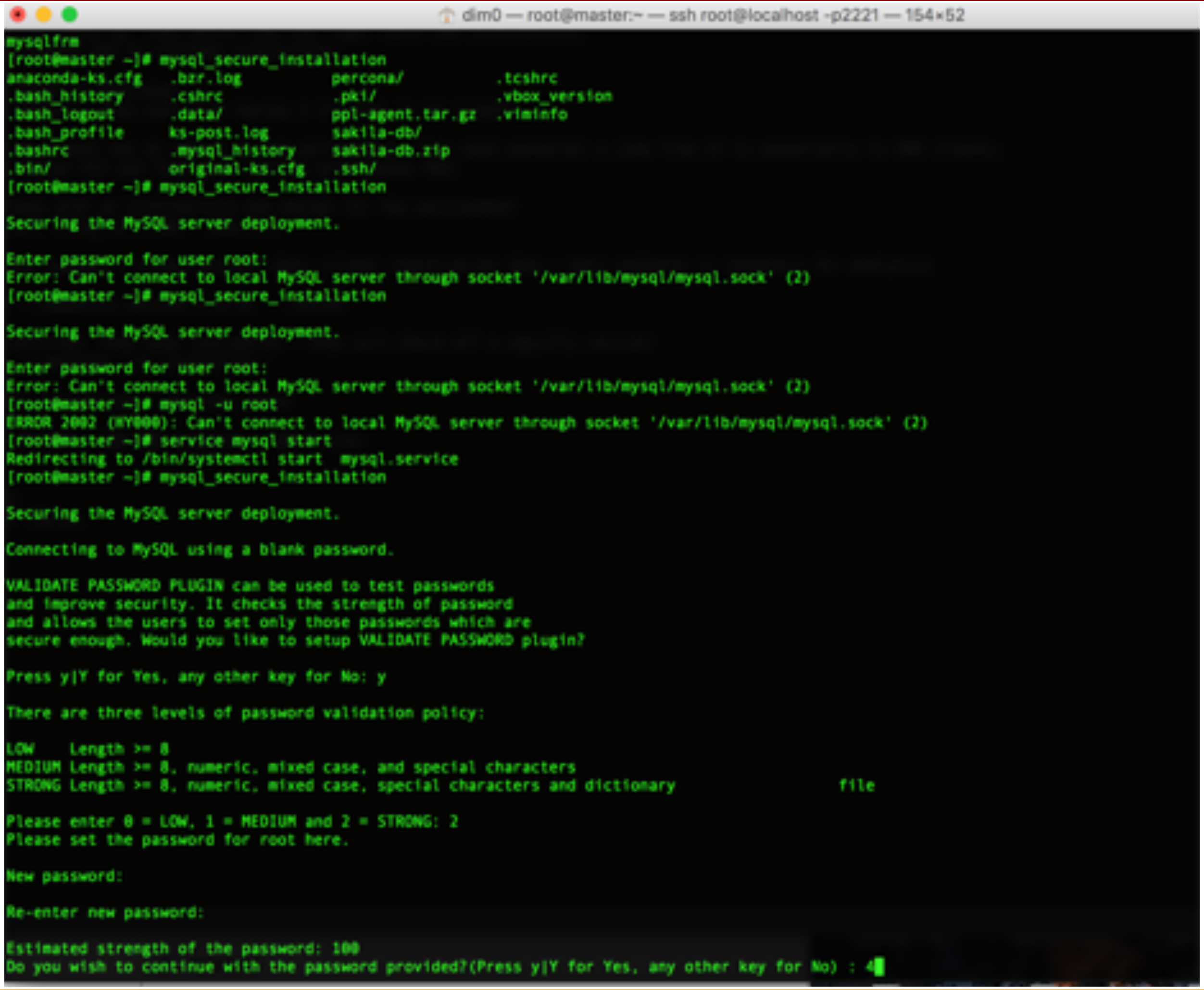
(Again) use configuration management (enforcement)

- Configuration management will help you with:
  - Consistent and effective rollout of your configuration files
  - Compliancy
  - MySQL database version (security patches, feature updates, etc)
  - OS security updates
  - User management
  - Resource limitations
  - Documents environments
  - Ensures the correct packages are installed
  - Less manual work

MySQL privileges

- Limit your user privileges to key application servers.
- Be restrictive for your users!
- Use complex passwords
  - Use the password validation plugin!

Password validation plugin



```
mysqlfrm
[root@master ~]# mysql_secure_installation
anaconda-ks.cfg   .bzr.log          percona/          .tcshrc
.bash_history     .cshrc            .pki/             .vbox_version
.bash_logout      .data/            ppl-agent.tar.gz  .viminfo
.bash_profile     ks-post.log       sakila-db/
.bashrc           .mysql_history    sakila-db.zip
.bin/             original-ks.cfg   .ssh/
[root@master ~]# mysql_secure_installation

Securing the MySQL server deployment.

Enter password for user root:
Error: Can't connect to local MySQL server through socket '/var/lib/mysql/mysql.sock' (2)
[root@master ~]# mysql_secure_installation

Securing the MySQL server deployment.

Enter password for user root:
Error: Can't connect to local MySQL server through socket '/var/lib/mysql/mysql.sock' (2)
[root@master ~]# mysql -u root
ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/lib/mysql/mysql.sock' (2)
[root@master ~]# service mysql start
Redirecting to /bin/systemctl start  mysql.service
[root@master ~]# mysql_secure_installation

Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD PLUGIN can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD plugin?

Press y|Y for Yes, any other key for No: y

There are three levels of password validation policy:

LOW    Length >= 8
MEDIUM Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary                  file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 2
Please set the password for root here.

New password:

Re-enter new password:

Estimated strength of the password: 100
Do you wish to continue with the password provided?(Press y|Y for Yes, any other key for No) : 4
```

Password requirement

- Use a strict policy along your organisation
- Make sure it’s not a dictionary word
- Give limitations!
- Don’t use root as a standard acces user to your database!

MySQL Grants

- Identify users based on: **user@host**
  - user: **username**
  - host: **hostname**/**ip**/network of the client that connects
  - different host, different user, different ‘grants’
  - use of **wildcards could be BAAAAD :)**
- Examples:

‘dim0’@‘localhost’, ‘root’@‘localhost’
‘tommeketoch’@‘app0001’, ‘kenju’@‘192.168.%’
‘ledijkske’@‘192.168.1.212’, ‘fredjen’@‘app.fq.dn’

- Creating A User:

```
> CREATE USER 'dim0'@'app0001';
```

- Drop user: change CREATE into DROP

```
> DROP USER [IF EXISTS] ‘dim0’@‘app0001’;
```

MySQL Grants (2)

- Grant the user some kind of privilege
- Grant ... to:

server,	trigger,
database,	stored procedure,
table,	view,
column,	index

- Example: INSERT, SELECT, UPDATE, DELETE
  - SQL Command:

```
> GRANT SELECT ON db.* TO ‘dim0’@‘app0001’;
> GRANT INSERT ON *.*  TO ‘dim0’@‘app0001’;
```

- Revoking privileges: change GRANT into REVOKE

MySQL grants (3)

- Password Expiration Policy
- Watch out for granting all privileges, the grant option or even super!

- User Account Locking

MySQL supports locking and unlocking user accounts using the ACCOUNT LOCK and ACCOUNT UNLOCK clauses

Grants (Limit your resources)

- For every user: max_user_connections

mysql> GRANT USAGE ON db.* TO ‘dim0'@'localhost' WITH **MAX_QUERIES_PER_HOUR** 1000 **MAX_UPDATES_PER_HOUR** 999 **MAX_CONNECTIONS_PER_HOUR** 100 **MAX_USER_CONNECTIONS** 5; FLUSH USER_RESOURCES;

It’s however not really popular… :-D

SSL connection

- SSL encryption to ensure in transit encryption.
- Requirement for PCI and other security compliance.
- Can give a slight performance penalty
- AWS/RDS users should definitely have a look at this

Handling ransomware

- Apply all updates!
- When is the last time you rebooted?
- Again limit access to trusted services and users.
- Make sure you have backups locally and offsite
- TEST

Encryption

- Encrypting your filesystem is still the most popular option.
- Since MySQL 5.7 table level encryption is included.

DON'T RUSH ME, SONNY. YOU RUSH A MIRACLE MAN, YOU GET ROTTEN MIRACLES.

Audit plugin

- Example of output

```
<AUDIT_RECORD
 "NAME"="Query"
 "RECORD"="23_2014-04-29T09:29:40"
 "TIMESTAMP"="2014-04-29T10:20:10 UTC"
 "COMMAND_CLASS"="select"
 "CONNECTION_ID"="49"
 "STATUS"="0"
 "SQLTEXT"="SELECT * from mysql.user"
 "USER"="root[root] @ localhost []”
 "HOST"="localhost"
 "OS_USER"=""
 "IP"=""
 />
```

- Output to syslog possible

Closing remarks

- Disable the use of the “LOAD DATA LOCAL INFILE”
- Maybe use the pam plugin!

Remember be restrictive!!!

Percona Live Europe Call for Papers & Registration are Open!

Championing Open Source Databases

- MySQL, MongoDB, Open Source Databases
- Time Series Databases, PostgreSQL, RocksDB
- Developers, Business/Case Studies, Operations
- September 25-27th, 2017
- Radisson Blu Royal Hotel, Dublin, Ireland

Submit Your Proposal by July 17th!

www.percona.com/live/e17