Chapter 7: MySQL Usage & Hardening

In this laboratory exercise, the student will be introduced to MySQL hardening as well as familiarized with perform MySQL queries.

7.1 Laboratory Exercise

MySQL Hardening & Basics

7.1.1 Specifications

The variant of Linux being used for this laboratory is Ubuntu Server 16.04.6. For this lab, it is be used as a database server.

7.1.2 Learning Objectives

- Hardening a MySQL Database
- MySQL Basic Commands

7.1.3 Mapping to NIST Nice Framework

This laboratory exercise is intended to increase the student's familiarity with Linux and MySQL. The student should be familiar with accessing, managing, and querying a MySQL database. This laboratory exercise maps to the following KSAs from the NIST NICE Framework:

- Cybersecurity and Privacy Principles (K0004)

- Cyber Threats and Vulnerabilities (K0005)
- Data Administration (K0020)
- Database Management Systems (K0023)
- Query Languages (K0069)
- Basic System and OS Hardening Techniques (K0205)
- Database Theory (K0420)
- Generate Queries (S0037)
- Recognizing Types of Vulnerabilities (S0078)
- System, Network, and OS Hardening Techniques (S0121)
- Maintain Databases (A0176)

7.1.4 Necessary Background and Expected Completion Time

This laboratory exercise can be completed by students with varying background and experience. The following categories should help identify approximately how much time (in minutes) will be necessary to complete the laboratory exercise, for a student meeting the criteria for the respective experience level.

- Beginner: A student in this category has little to no experience using Linux and the Linux terminal. Additionally, this student has little to no experience using MySQL.
- Intermediate: A student in this category has experience with Linux and the Linux terminal but has little to no experience using MySQL.
- Advanced: A student in this category has experience with Linux, the Linux terminal and has experience using MySQL.

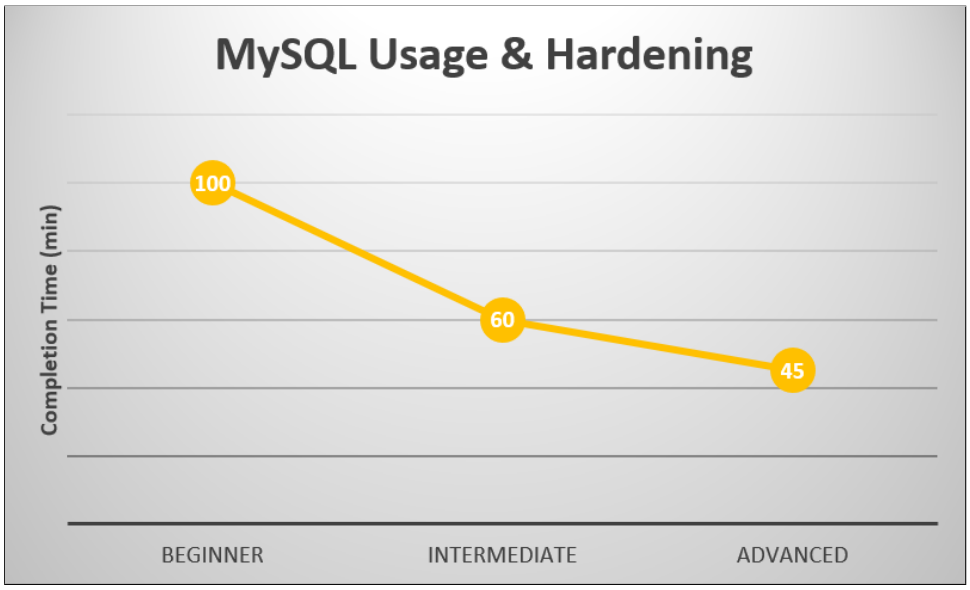

Figure 7.1: *MySQL Hardening & Basics Laboratory Exercise Expected Completion Time (min)*

7.1.5 Configuration and Setup

The machine in this laboratory exercise is an installation of CentOS 7. The machine has been configured with vulnerabilities listed in the vulnerability overview. The vulnerabilities include default or weak passwords, insecure MySQL root accounts, poorly configured MySQL permissions, default port usage, unnecessary databases and users, and automated startup tasks. MySQL is installed as a part of this laboratory exercise. The initialization script for this laboratory exercises reaches out to a GitHub repository to clone necessary files for the exercise. The initialization script also performs the creation and population of the MySQL databases and tables. The machine is configured using the initialization script, listing **7.1**.

Listing 7.1: *ms-initializationscript.sh*

```
#!/bin/bash

sudo dpkg --configure -a

#install git
sudo apt-get install git -y;

#remove existing repo and clone git repo
sudo rm -r CYOTEE
sudo git clone https://github.com/CenterForSecureAndDependableSystems/CYOTEE.git

#install MySQL
sudo apt-get install -y mysql-server;

#create the MySQL users
sudo mysql -u root -e "CREATE USER 'randomuser'@'localhost' IDENTIFIED BY '
    password'";
sudo mysql -u root -e "CREATE USER 'redteamer'@'localhost' IDENTIFIED BY 'redteam
    '";
sudo mysql -u root -e "CREATE USER 'haxxor'@'localhost' IDENTIFIED BY 'haxxor'";
sudo mysql -u root -e "CREATE USER 'testuser'@'localhost' IDENTIFIED BY 'test'";

#create the unnecessary database
sudo mysql -u root -e "CREATE DATABASE dontlook";

#remove the vulnerable database if one already exists
sudo mysql -u root -e "DROP DATABASE vulndb";

#create the vulnerable database
sudo mysql -u root -e "CREATE DATABASE vulndb";

#grant all privileges to users
sudo mysql -u root -e "GRANT ALL PRIVILEGES ON vulndb.* TO 'root'@'localhost'";
sudo mysql -u root -e "GRANT ALL PRIVILEGES ON vulndb.* TO 'testuser'@'localhost'
    ";

#import vulndb sql file
sudo mysql -u root vulndb < CYOTEE/CYOTEE_Code/SQL/vulndb.sql;
```

7.1.6 Vulnerability List

1. MySQL Root Account
2. MySQL Permissions
3. Default Ports
4. Unnecessary Databases and Users
5. Automated Startup Tasks

7.1.7 Challenges

1. *Secure MySQL Root Account*

   Root accounts on machines are typically the account which has the highest privileges. It is critical that the root account be properly secured as the root user can perform many super user tasks that a normal user may not have permission to perform. MySQL comes preconfigured with a root account which is not secured in the default installation of MySQL. Because of this, one is able to access the root account simply by typing the command:

   `$ mysql -u root`

   For this task, set the root account password to be something secure.

2. *Harden MySQL Permissions*

   MySQL allows permissions to be configured for tables in a database on a user-by-user basis. A few of the most commonly used MySQL commands are SELECT (equivalent to read permissions), INSERT, and DELETE. Databases being used by a web page to populate fields, for example, likely only need read access on the database and therefore should only have the SELECT privilege. The user **testuser** has all privileges on all tables in the database *vulndb*. This was accomplished by using the command:

   `GRANT ALL PRIVILEGES ON vulndb.* TO 'testuser'@'localhost';` In a later exercise, you will learn about tailoring permissions based on the concept of least privilege as needed in a web application.

   For this task, modify the privileges so that testuser has only read access on all the tables in the database *vulndb*.

3. *Change Default Ports*

   Network ports can be thought of as doors. Each application on a computer has its own port, or door, which is used for data to flow to and from that application. MySQL uses port 3306 by default, however, this can be changed. Although changing the port MySQL uses does not inherently improve security, it does defend against automated attacks which specifically target port 3306.

For this task, change the port that MySQL uses to a different port which is currently unused.

4. *Remove Unnecessary Databases or Users*

   Often, a base installation of MySQL will already include multiple MySQL users and a few example databases. These databases and users can be a vulnerability which attackers may exploit to gain access to your MySQL server because they are often not considered when securing the server and databases on it. It can be best to remove or disable these accounts and databases in order to avoid forgetting they exist, thereby forgetting to secure them.

   For this task, first find and remove the account which seems least likely to be a valid account on the machine and then find and remove the database which seems most likely to be associated with the unnecessary user.

5. *Using MySQL*

   MySQL commands are performed by using a specific syntax.

   For this task, perform the following steps:

   (a) Create a new database named `television`

   (b) Create a table in the newly created database named `shows` with the following fields (also called columns)

      i. `name`

      ii. `startyear`

   (c) Add another field named `endyear`

   (d) Create an entry for the following television shows in the table

      i. Boy Meets World

      ii. That 70's Show

      iii. Saved by the Bell

   (e) Remove the entry for Boy Meets World

   (f) Add an entry for the show Girl Meets World

Performing these tasks should provide you with a basic understanding of using MySQL and enable you to create and modify a database.

6. *Querying MySQL Databases*

   Often, rather than being tasked with creating and managing a database, one will be asked to find data in a table. Various actions such as reading, inserting, and deleting data to and from a database are known as queries. Reading information from a table is done in MySQL by using the SELECT command along with specific parameters if the target data is known.

   For part A of this task, you will be querying the table *useraccounts* in the database *vulndb* to read the data.

   Part B of this task requires that you query the table *employees* in the database *vulndb* to read the data. Keep in mind that queries can be made where field values are specified, and multiple field-value pairs may be chained together for a more specific query.

   (a) Query to find the solutions to the following questions.

      i. What is the name of the individual whose locations is New York?

      ii. What is the salary of Jeremiah Houston?

      iii. What is the name of the individual whose location is Los Angeles and has a salary of $500,000?

   (b) Query to find the solutions to the following questions.

      i. How old is Tami Vasquez?

      ii. What does Wanda Lloyd do for work?

      iii. What are the occupations of the 40 year-olds?