Đại học Quốc gia Thành phố Hồ Chí Minh
Trường Đại học Công nghệ Thông tin (UIT)

BÁO CÁO THỰC HÀNH

Môn học: Bảo mật web và ứng dụng

Lab 2: Tổng quan các lỗ hổng bảo mật web thường gặp (phần 2)

Ngày báo cáo: 03/04/2024

1. THÔNG TIN CHUNG:

Lớp: NT213.O22.ATCL.2

STT	Họ và tên	MSSV	Email
1	Hoàng Gia Bảo	21521848	21521848@gm.uit.edu.vn
2	Phạm Hoàng Phúc	21521295	21521295@gm.uit.edu.vn
3	Nguyễn Đại Nghĩa	21521182	21521182@gm.uit.edu.vn

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

BÁO CÁO CHI TIẾT

Bài tập 1:

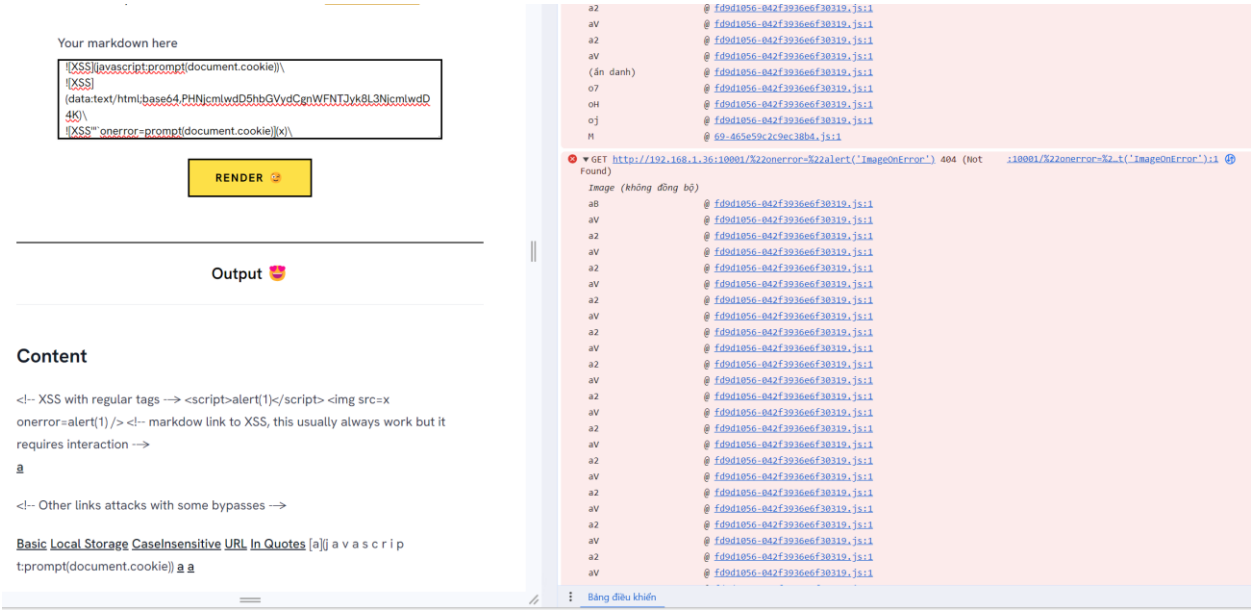
LỖ HỔNG : XSS stored

Trước hết thì ta có thể thấy trang web này đang sử dụng thư viện react-markdown và gray-matter để render ra output

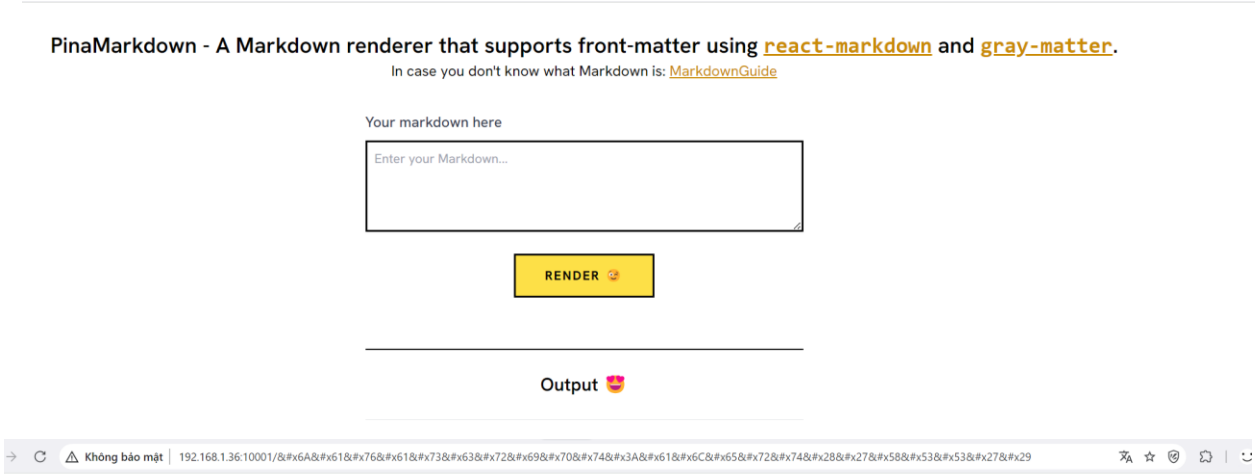
Sử dụng react-markdown giúp đảm bảo tính an toàn và tin cậy khi render markdown trong ứng dụng React, đồng thời giúp tránh được các vấn đề về bảo mật như tấn công XSS.

Ở đây ta sẽ thử sử dụng 1 số payload trên mạng để fuzzing web

Nguồn : https://book.hacktricks.xyz/pentesting-web/xss-cross-site-scripting/xss-in-markdow



Kết quả sau khi nhấn vào các link :



Mức độ ảnh hưởng của lỗ hổng XSS trong markdown

Tiềm ẩn rủi ro bảo mật: XSS cho phép tin tặc chèn mã JavaScript độc hại vào các trang web hoặc ứng dụng, khiến cho những người dùng không may mắn có thể bị tấn công.

Lộ thông tin cá nhân: Mã JavaScript độc hại có thể truy cập và lấy thông tin cá nhân của người dùng, như tên người dùng, mật khẩu, cookie hoặc dữ liệu phiên.

Phá hoại hoặc can thiệp vào trang web: Tin tặc có thể sử dụng lỗ hổng XSS để thay đổi nội dung của trang web, thêm các liên kết độc hại, hoặc thậm chí làm hỏng trang web hoàn toàn.

Đánh cắp phiên làm việc của người dùng: XSS có thể được sử dụng để đánh cắp phiên làm việc của người dùng, cho phép tin tặc thực hiện các hành động thay mặt người dùng trên trang web.

Phá vỡ tính toàn vẹn của dữ liệu: Nếu dữ liệu bị biến đổi hoặc phá vỡ tính toàn vẹn bởi mã JavaScript độc hại, điều này có thể gây ra hậu quả nghiêm trọng đối với tính toàn vẹn và tin cậy của dữ liệu.

Khuyến cáo khắc phục : Sử dụng thư viện hỗ trợ kiểm tra file markdown như trang

Bài tập 2:

Tiêu đề: Identification and Authentication Failures

Tóm tắt lỗ hổng: Ta được cung cấp admin account và password ở dạng hash để đăng nhập vào 1 trang web. Lợi dụng cơ chế bảo mật của trang web này, ta có thể nhập password sai liên tục để làm tài khoản bị khóa 1 ngày.

Các bước thực hiện:

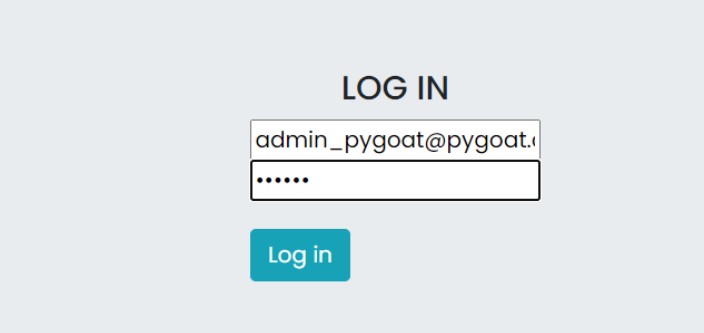
Tài khoản admin được cung cấp: **admin_pygoat@pygoat.com**

Here is a admin pannel of the application. After some recon we got the username `admin_pygoat@pygoat.com`
& password hash `$argon2id$v=19$m=65536,t=3,p=4$Ub40KHiEbH9I3Bsd4VHQDA$4zsIHDmAbejFJmaZq8a2yVIJdHvfylDlQ85w3YRLMSQ`

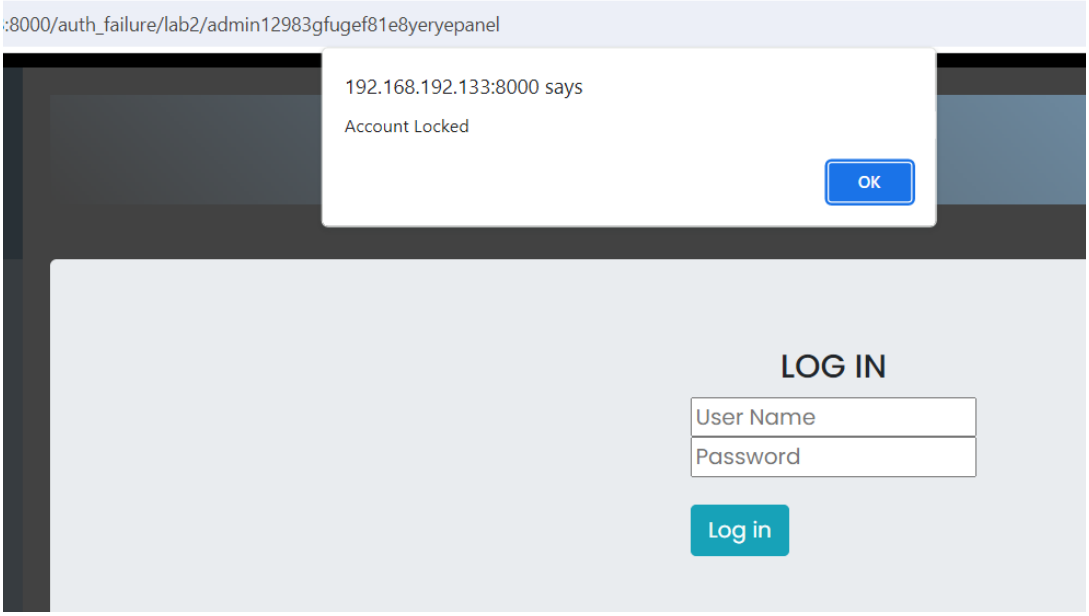
Nếu mật khẩu nhập sai 5 lần thì tài khoản sẽ bị khóa 1 ngày

```
try:
 ph = PasswordHasher()
 ph.verify(user.password, password)
 if user.is_locked == True and user.lockout_cooldown < datetime.date.today():
  user.is_locked = False
  user.last_login = datetime.datetime.now()
  user.failattempt = 0
  user.save()
 return render(request,"Lab_2021/A7_auth_failure/lab2.html", {"user":user, "success":True,"failure":False})
except:
 fail_attempt = user.failattempt + 1
 if fail_attempt == 5:
  user.is_active = False
  user.failattempt = 0
  user.is_locked = True
  user.lockout_cooldown = datetime.datetime.now() + datetime.timedelta(minutes=1440)
  user.save()
```

Tiến hành nhập mật khẩu:



Kết quả:



Mức độ ảnh hưởng: Cao

Khuyến cáo khắc phục:

Đảm bảo mật khẩu đủ mạnh và khó đoán, nên bao gồm cả chữ hoa, chữ thường, số và ký tự đặc biệt. Thay đổi mật khẩu định kỳ.

Thực hiện thông báo đăng nhập qua email hoặc thiết bị đã đăng nhập trước. Áp dụng cơ chế báo động hoặc khóa tài khoản khi nhập mật khẩu sai nhiều lần hoặc đăng nhập từ thiết bị lạ.

Sử dụng hệ thống xác thực đa yếu tố.

Tham khảo:

https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures

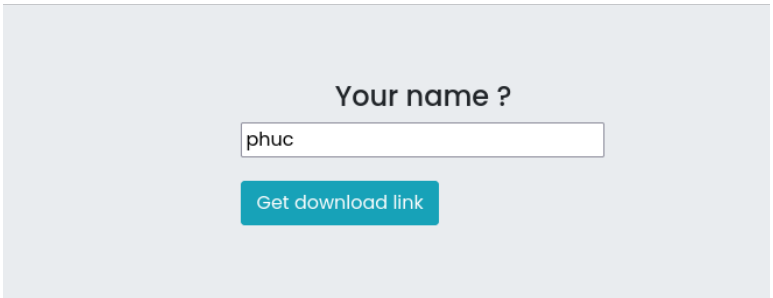
Bài tập 3:

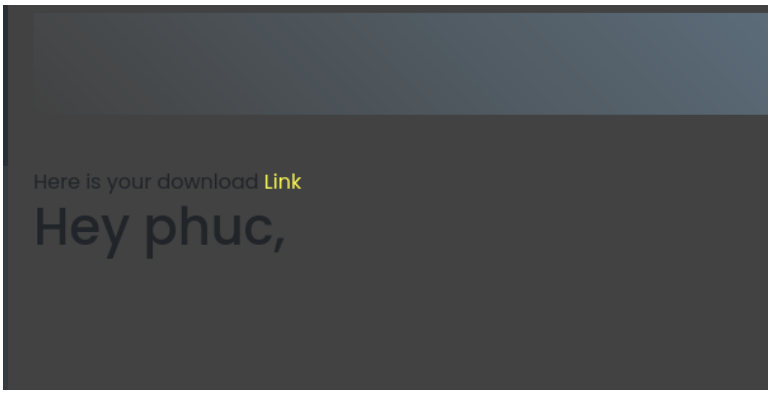
Tiêu đề: Software and Data Integrity Failures

Mô tả lỗ hổng: Trang web cho phép người truy cập tải file về. Ta có thể làm ảnh hưởng đến tính toàn vẹn dữ liệu bằng cách chèn script thông qua input để thay đổi file được tải về.

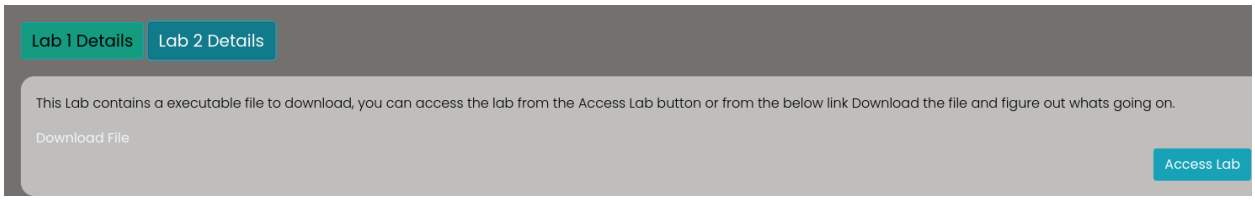
Các bước thực hiện:

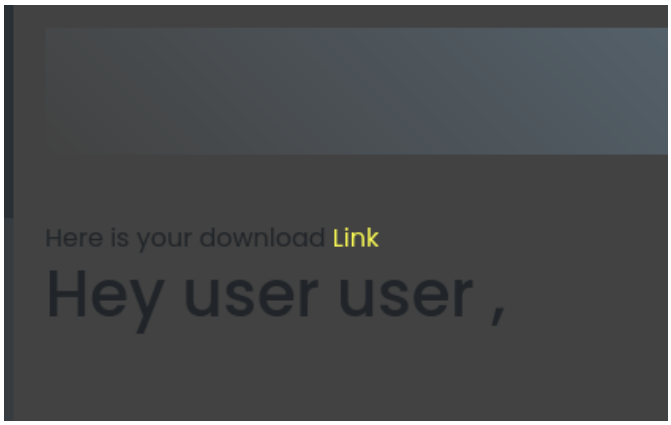
Truy cập vào trang web, nhập tên và tải file về. File được tải về là **real.txt**



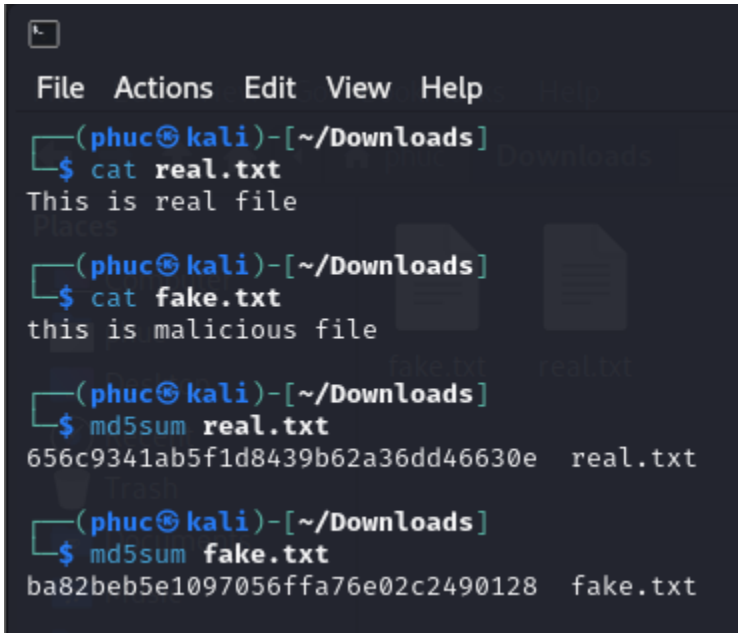


Ngoài ra, ở phần Lab 2 Details, ta tìm được một đường dẫn để tải về file **fake.txt**

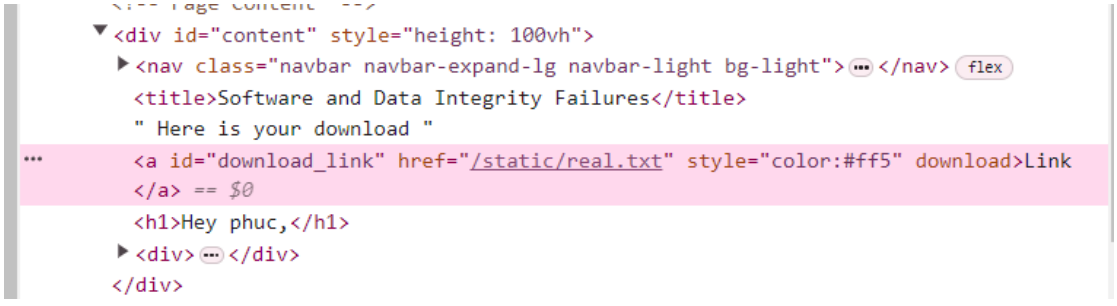




Kiểm tra file

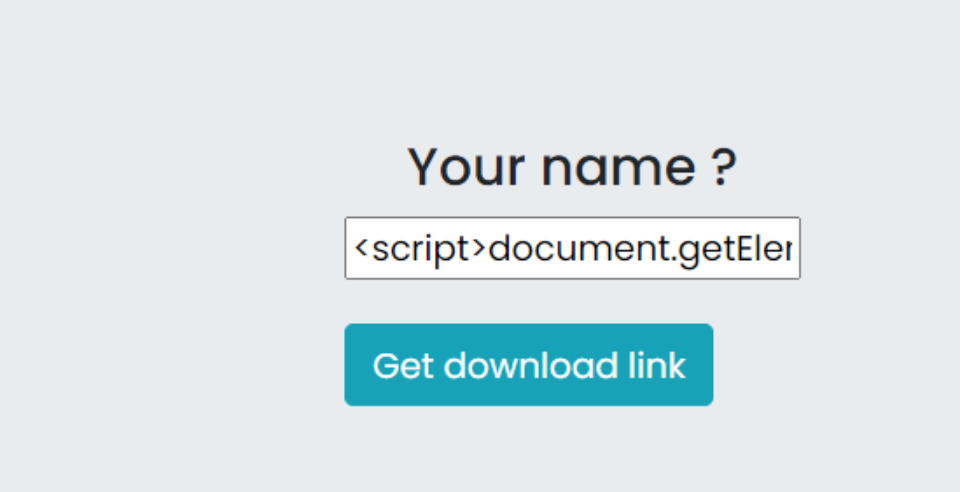


Dựa vào source code, ta biết được đường dẫn cho real file được lưu trong href của thẻ <a> có id = “download_link”

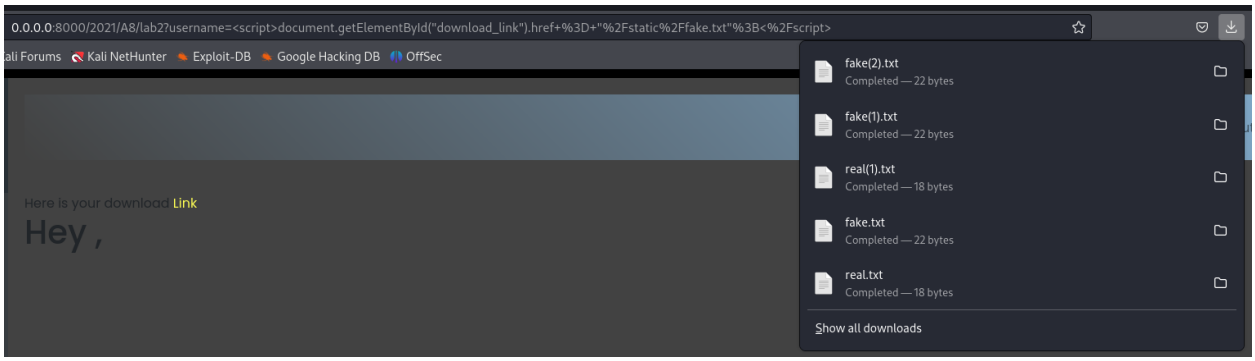


Nhập script khai thác vào input

<script>document.getElementById("download_link").href ="/static/fake.txt";</script>



Kết quả là nhấn vào đường link real file nhưng lại tải về fake file



Mức độ: rất cao

Khuyến cáo khắc phục: Thực hiện lọc các giá trị input để chặn việc chèn các lệnh thực thi vào chương trình

Tham khảo:

https://owasp.org/Top10/A08_2021-Software_and_Data_Integrity_Failures

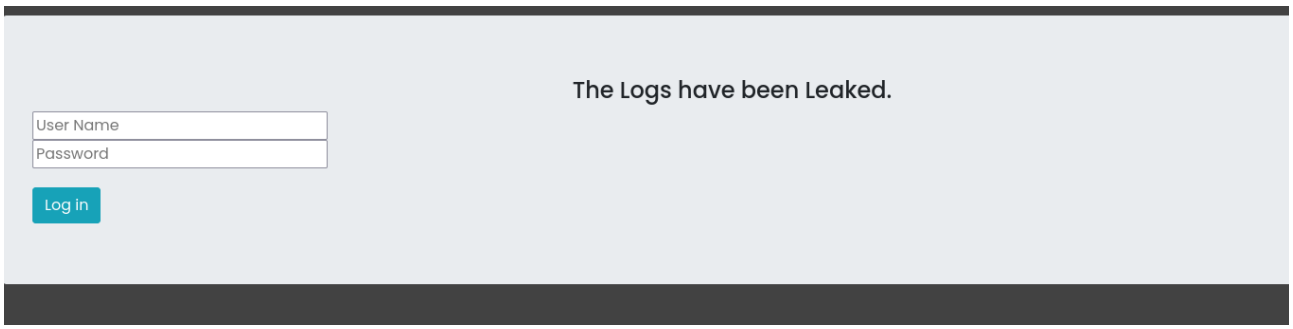
Bài tập 4:

Tiêu đề: Security Logging and Monitoring Failures

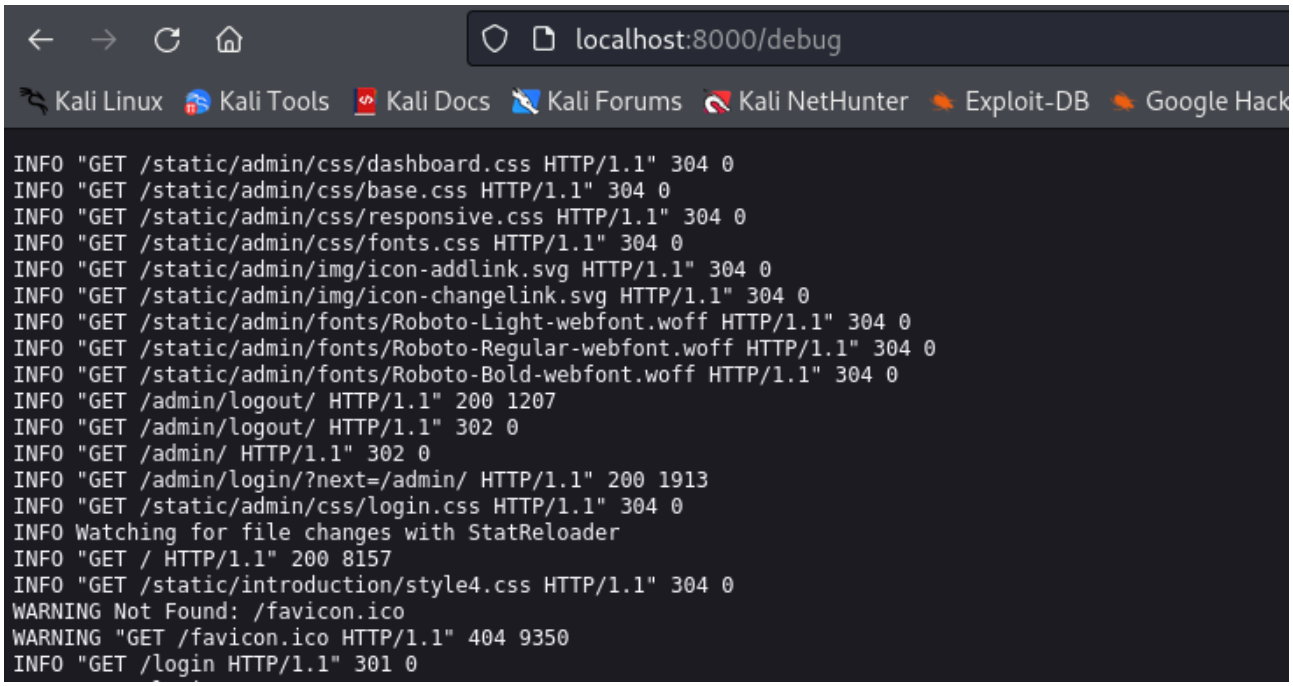
Mô tả lỗ hổng: Trang web đăng nhập này có thực hiện việc lưu log lại của những lần đăng nhập, nhưng mà nguy hiểm ở chỗ log này nó có ghi lại luôn cả username và password đăng nhập. Không những thế việc lưu log cũng vô cùng lỏng lẻo, bất kì ai cũng có thể đọc được log mà không cần là người quản trị.

Các bước thực hiện:

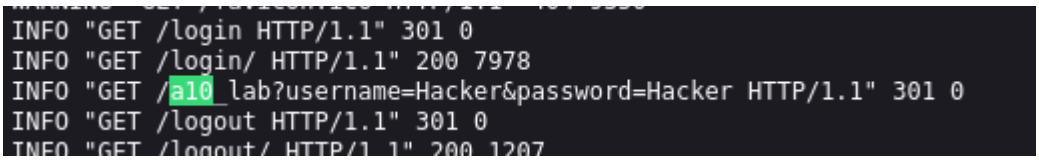
Khi truy cập vào trang, đây là giao diện nhận được:



Sau đó em tiến hành search trên thanh url để vào /debug (bởi gợi ý ở phần details):



Tiếp tục tìm đến từ khóa a10, do bài tập lần này là /a10



Tìm đến đây thì em đã thấy được thông tin đăng nhập gồm username và password, cả 2 đều là Hacker, em sẽ lấy thông tin này để đăng nhập thử.

Dùng thông tin kiếm được để đăng nhập thì bị báo lỗi sau:

```
Wrong username or Password
```

Em cũng không biết tại sao, nhưng mà em khá chắc chắn rằng thông tim vừa tìm được là đã chính xác.

Mức độ ảnh hưởng: Cao

Khuyến cáo khắc phục:

- Xác định rõ ràng những loại thông tin cần được ghi log, bao gồm các hành động người dùng quan trọng, các lỗi hệ thống, truy cập không thành công, và các dấu hiệu của hoạt động đáng ngờ.
- Đảm bảo tính toàn vẹn và bảo mật cho các file log bằng cách hạn chế quyền truy cập và mã hóa. Sử dụng các giải pháp quản lý log trung tâm để thu thập, lưu trữ và phân tích log một cách hiệu quả.

Bài tập 5:

Tiêu đề: Server-Side Request Forgery (SSRF)

Mô tả lỗ hổng: Ở trang web này, cứ mỗi nút nhấn như blog1, blog2,… thì sẽ hiển thị ra tương ứng nội dung của blog1, blog2,… Nhưng người lập trình họ đã không kĩ lưỡng và để sơ xuất ở chỗ mỗi button blog đều có thể thay đổi được giá trị của input thông qua chức năng dev tool của web browser và rồi khi submit thì nó lấy nguyên giá trị từ input đó để thực hiện mà không tiến hành việc kiểm tra lại liệu đường dẫn file đó có phù hợp hay chưa (ví dụ như là nút nhấn blog1 thì tương ứng với file blog1, bất kì file khác blog1 đều sẽ trả về lỗi), từ đó người dùng họ có thể tự do đổi được đường dẫn để đọc được chính xác file mà họ muốn trong hệ thống thông qua việc thay đổi giá trị của input.

Các bước thực hiện:

Khi vừa vào trang web thì đây là giao diện em nhận được:

Bấm vào xem thử blog1 thì hiển thị ra như sau:

Lúc này đọc hint thì nhận được thông báo là tìm ra file .env để hoàn thành bài tập này.

Vậy thì em sẽ tiến hành việc tìm file .env thông qua việc thay đổi đường dẫn file trong input của blog1.

Mở dev tool lên, tìm đến code của nút nhấn blog1:

Ở đây có 2 trường input hidden, và trường input ở dưới có 1 giá trị value là đường dẫn file tới blog1.txt, vậy thì em sẽ tiến hành thay đổi giá trị này để tìm ra file .env

Em cũng không biết file này sẽ nằm ở đâu, thế nên em sẽ đoán và thử lần lượt thay đổi giá trị trường này thành ".env", "./.env", "../.env", "../../.env",…. Để tìm ra được kết quả đúng, đầu tiên là ".env":

Thông báo trên cho thấy rằng không tìm thấy blog, tương đương là không tìm được file .env, vậy thì em sẽ đổi sang là "./.env".

Kết quả thì cũng tương tự như ".env", vậy thì em sẽ đổi sang là "../.env" để xem sao:

Thông qua kết quả nhận được có thể thấy rằng đường dẫn "../.env" đã đúng và em đã hoàn thành được mục tiêu của bài lab

Mức độ ảnh hưởng: Cao

Khuyến cáo khắc phục:

- Kiểm tra và xác minh tất cả dữ liệu nhập từ người dùng để ngăn chặn việc chèn các yêu cầu độc hại. Sử dụng các phương pháp như ràng buộc loại dữ liệu và kiểm tra định dạng để loại bỏ các giá trị không hợp lệ.
- Trong các yêu cầu mà server gửi đi, sử dụng token chống CSRF (Cross-Site Request Forgery) để đảm bảo rằng yêu cầu được thực hiện là hợp lệ và đã được xác thực.