**BÁO CÁO THỰC HÀNH – LAB 2**

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

|  |  |
| --- | --- |
| **Họ và tên** | **MSSV** |
| Phan Nguyễn Nhật Trâm | 22521501 |

**GVHD:** Ngô Đức Hoàng Sơn

**Lớp:** NT213.P12.ANTT

# Bài tập 1: Báo cáo lỗ hổng đang được thực hành.

## Tiêu đề: Lỗ hổng Vulnerable and Outdated Components trong tính năng chuyển đổi tập tin yaml thành định dạng json, khiến kẻ tấn công lấy được thông tin của server.

## Mô tả:

Lỗ hổng Vulnerable and Outdated Components trong hệ thống xảy ra do phần mềm, hệ điều hành, máy chủ ứng dụng, cơ sở dữ liệu, API hoặc các thành phần khác không được hỗ trợ hoặc không được cập nhật thường xuyên.

## Kịch bản:

Trong trường hợp này, lỗ hổng xảy ra khi sử dụng thư viện PyYAML 5.4 để phân tích và xử lý các tệp YAML trên server. Phiên bản PyYAML này dễ bị tấn công bởi lỗ hổng **arbitrary code execution.** Kẻ tấn công tải lên một tệp YAML độc hại chứa các lệnh code injection và web thực thi mã độc từ tệp này trên server. Từ đó, có thể chiếm quyền điều khiển máy chủ, thực thi các lệnh nguy hiểm, gây ra tình trạng từ chối dịch vụ (DoS) hoặc đánh cắp dữ liệu nhạy cảm.

## Các bước để thực hiện:

1. **Tạo file sample.yaml:**

!!python/object/apply:subprocess.Popen

- ls

1. **Upload file lên hệ thống, quan sát kết**A screenshot of a computer

   Description automatically generated

*Hình 1. Kết quả upload file sample.yaml*

A screenshot of a computer

Description automatically generated

*Hình 2. Kết quả upload file sample.yaml ở giao diện web*

**Nhận xét**: Lệnh ls đã được thực hiện trên server và liệt kê các tập tin của server trên terminal

1. **Chỉnh sửa file sample.yaml để lấy thông tin user:**

!!python/object/apply:subprocess.check\_output

- ["whoami"]

1. **Upload file, kiểm tra kết quả:**

A screenshot of a computer

Description automatically generated

*Hình 3. Kết quả truy xuất username*

**Nhận xét:** Kẻ tấn công đã lấy được thông tin user

## Tài liệu hỗ trợ và tham khảo:

<https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/>

## Mức độ ảnh hưởng của lỗ hổng:

Hệ thống sẽ dễ dàng bị khai thác bởi các lỗ hổng đã được công bố và có sẵn. Điều này có thể dẫn đến việc bị tấn công, đánh cắp dữ liệu, chiếm quyền điều khiển hệ thống, hoặc phá hoại dịch vụ.

Nếu không quét lỗ hổng bảo mật định kỳ và không kiểm tra tính tương thích của các bản vá, hệ thống có thể trở nên không ổn định hoặc tiếp tục tồn tại lỗ hổng đã biết. Điều này kéo dài thời gian hệ thống dễ bị tổn thương, tạo cơ hội cho kẻ tấn công khai thác và gây thiệt hại nghiêm trọng, ảnh hưởng đến an ninh, hiệu suất và uy tín.

## Khuyến cáo khắc phục:

* Theo dõi phiên bản các thành phần: Liên tục kiểm kê phiên bản của tất cả các thành phần phía máy khách và máy chủ, bao gồm cả thư viện phụ thuộc, bằng cách sử dụng các công cụ như OWASP Dependency Check, retire.js, v.v. Đồng thời, theo dõi các nguồn như Common Vulnerability and Exposures (CVE) và National Vulnerability Database (NVD) để phát hiện lỗ hổng bảo mật.
* Sử dụng công cụ phân tích thành phần phần mềm: Tự động hóa quy trình kiểm tra lỗ hổng bằng cách sử dụng các công cụ phân tích thành phần phần mềm và đăng ký nhận thông báo qua email về các lỗ hổng bảo mật liên quan.
* Chỉ sử dụng thành phần từ nguồn chính thức: Tải các thành phần từ nguồn chính thức qua các liên kết bảo mật, ưu tiên các gói phần mềm đã được ký số để giảm thiểu nguy cơ sử dụng thành phần bị thay đổi hoặc độc hại.
* Theo dõi thư viện không được duy trì: Giám sát các thư viện và thành phần không còn được duy trì hoặc không còn phát hành bản vá bảo mật cho các phiên bản cũ. Nếu không thể vá lỗi, xem xét triển khai giải pháp vá ảo để giám sát, phát hiện hoặc bảo vệ trước các vấn đề bảo mật.
* Duy trì quy trình cập nhật liên tục: Đảm bảo có kế hoạch liên tục theo dõi, đánh giá và áp dụng các bản cập nhật hoặc thay đổi cấu hình trong suốt vòng đời của ứng dụng hoặc danh mục sản phẩm.

# Bài tập 2: Báo cáo lỗ hổng đang được thực hành.

## **Tiêu đề:** Lỗ hổng Identification and Authentication Failures trong trang đăng nhập bằng OTP, kẻ tấn công brute – force mã OTP và đăng nhập với quyền admin.

## Mô tả:

Lỗ hổng Identification and Authentication Failures có thể khiến ứng dụng trở thành mục tiêu của các cuộc tấn công xác thực như tấn công nhồi thông tin đăng nhập (credential stuffing), tấn công brute force, hoặc lạm dụng thông tin đăng nhập yếu. Các lỗ hổng này có thể xuất hiện nếu ứng dụng cho phép:

* Thực hiện tấn công tự động với danh sách tên người dùng và mật khẩu hợp lệ.
* Tấn công brute force hoặc các cuộc tấn công tự động khác.
* Sử dụng mật khẩu mặc định, yếu, hoặc dễ đoán như "Password1" hoặc "admin/admin".
* Quy trình khôi phục mật khẩu hoặc quên mật khẩu yếu hoặc không an toàn (như OTP).
* Lưu trữ mật khẩu ở dạng văn bản thuần, mã hóa yếu, hoặc hash yếu.
* Thiếu hoặc không có xác thực đa yếu tố (MFA).
* Không vô hiệu hóa đúng cách Session ID sau khi người dùng đăng xuất hoặc không hoạt động trong một khoảng thời gian dài.

## Kịch bản:

Trong trường hợp này, lỗ hổng xảy ra khi đăng nhập bằng OTP, kẻ tấn công sẽ brute-force OTP có dạng xxx là 3 digit

## Các bước để thực hiện:

1. **Nhập email và gửi yêu cầu nhận OTP:**

* Email: admin@pygoat.com

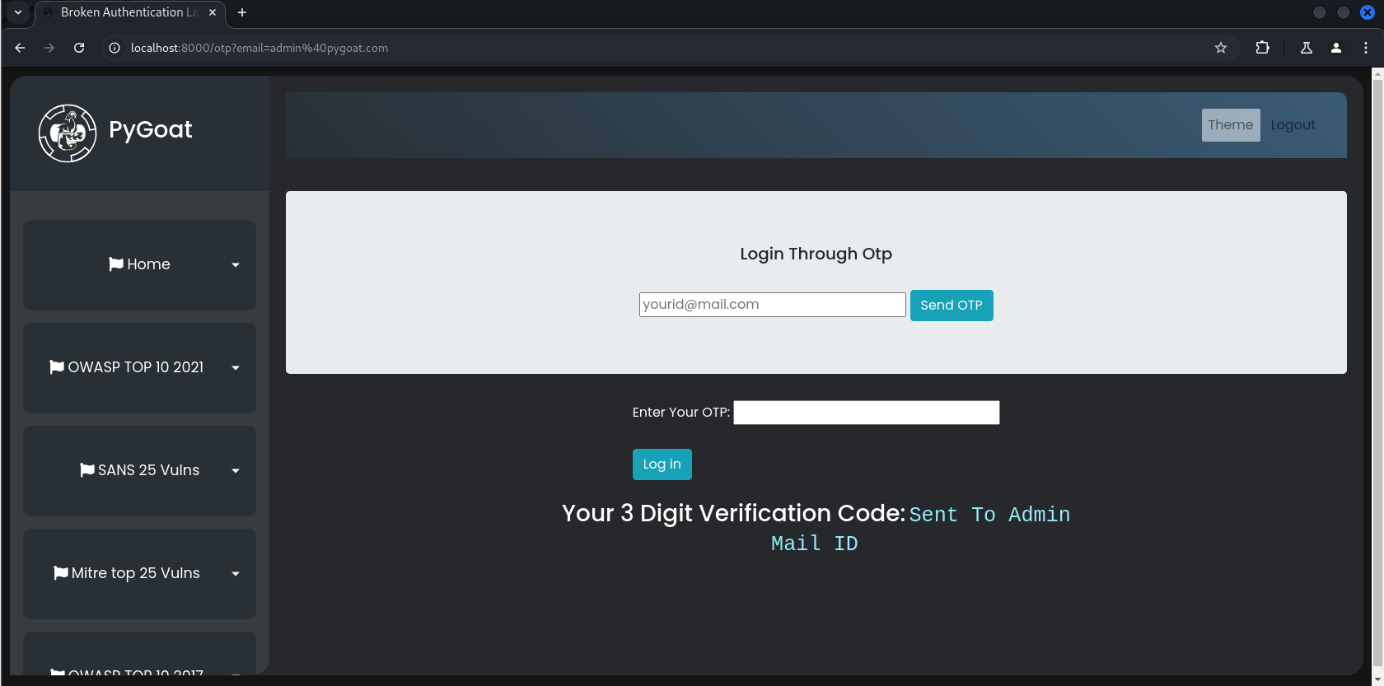
1. **Sử dụng Burpsuite, tiến hành brute-force**

* Bật Intercept on
* Click Login tại giao diện web
* Gửi request đến tab Intruder
* Lưu ý: sử dụng <http://localhost:8000> cho target
* Tại tab payload
  + Payload Type: Numbers
  + Payload Setting: From 1 To 999 Step 1 (có thể chia nhỏ để attack nhanh hơn)
  + Min integer digits và Max integer digits = 3
* Start Attack ‘Sniper’

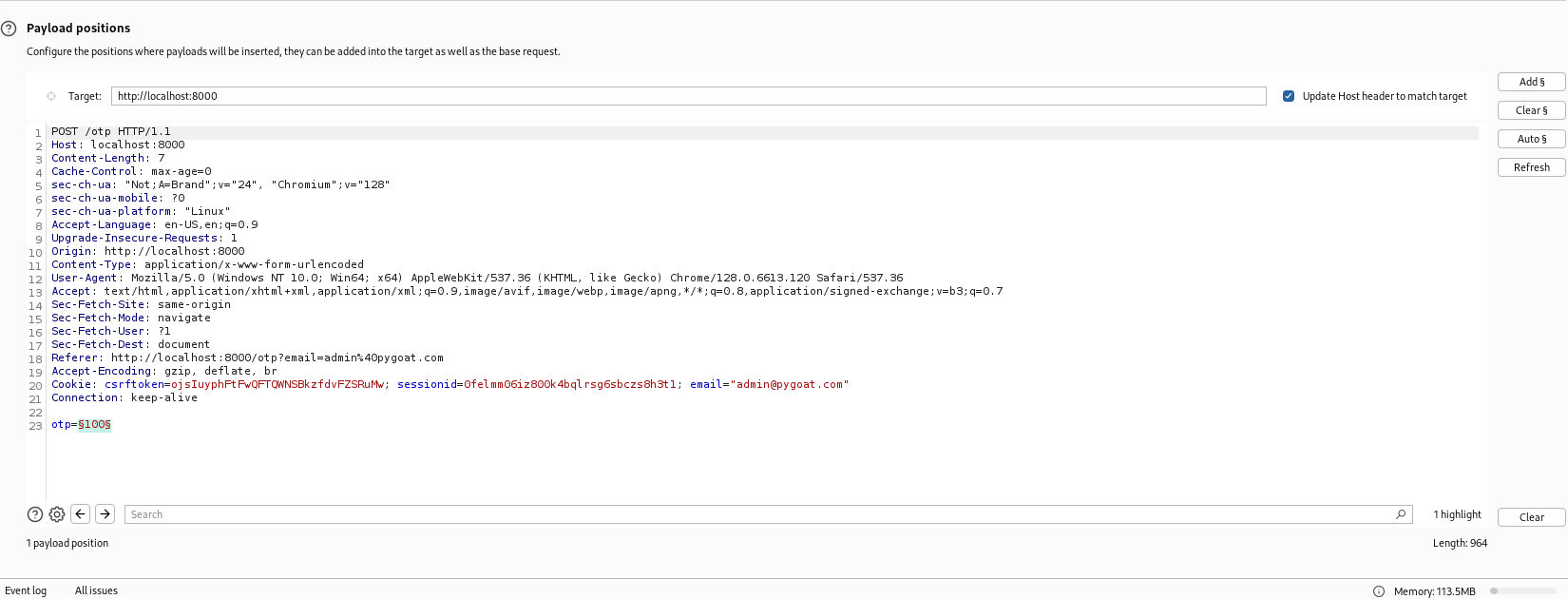
1. **Phân tích kết quả Attack:**

* Kiểm tra cột length, chú ý sự khác biệt

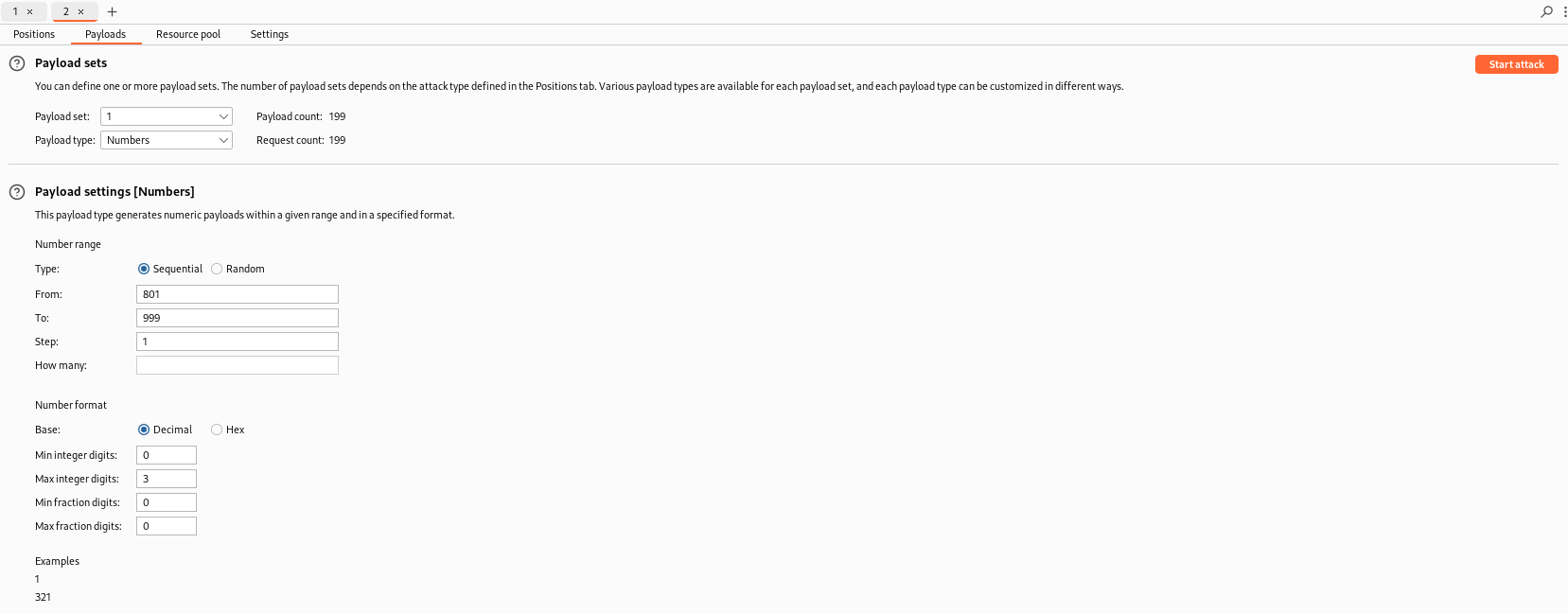
## Minh chứng:



*Hình 4. Dùng email của admin để nhận OTP*



*Hình 5. Request được chuyển qua tab Intruder*

**

*Hình 6: Cài đặt Payload để Attack*

*A screenshot of a computer

Description automatically generated*

*Hình 7: Quá trình Attack*

## Tài liệu hỗ trợ và tham khảo:

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

## Mức độ ảnh hưởng của lỗ hổng:

* Credential stuffing: Kẻ tấn công có thể tận dụng thông tin đăng nhập đã thu thập để đăng nhập tự động vào tài khoản người dùng.
* Tấn công brute force: Cho phép tấn công tự động thử nhiều tổ hợp tên người dùng và mật khẩu, dẫn đến việc đánh cắp thông tin người dùng. (Trường hợp đang thực hành)
* Xâm phạm tài khoản: Sử dụng mật khẩu yếu hoặc mật khẩu mặc định có thể dễ dàng bị khai thác, dẫn đến xâm phạm tài khoản.

## Khuyến cáo khắc phục:

* Áp dụng xác thực đa yếu tố (MFA) để ngăn chặn các cuộc tấn công brute force, và sử dụng lại thông tin đăng nhập đã bị đánh cắp.
* Không sử dụng mật khẩu mặc định khi triển khai hệ thống, đặc biệt là cho tài khoản quản trị viên.
* Kiểm tra mật khẩu yếu: Thực hiện kiểm tra mật khẩu mới hoặc thay đổi dựa trên danh sách các mật khẩu dễ đoán hoặc yếu.
* Tuân thủ các chính sách mật khẩu hiện đại: Điều chỉnh độ dài, độ phức tạp, và quy tắc thay đổi mật khẩu theo các tiêu chuẩn như NIST 800-63b.
* Bảo vệ quy trình đăng ký và khôi phục thông tin đăng nhập: Đảm bảo các quy trình này chống lại các cuộc tấn công liệt kê tài khoản bằng cách hiển thị thông báo giống nhau cho tất cả các trường hợp.

# Bài tập 3: Báo cáo lỗ hổng đang được thực hành.

## **Tiêu đề:** Lỗ hổng Software and Data Integrity Failures trong trang đăng nhập, kẻ tấn công có thể đăng nhập với quyền admin.

## Mô tả:

Lỗ hổng xảy ra khi mã nguồn và hạ tầng không được bảo vệ khỏi các hành vi vi phạm tính toàn vẹn. Ví dụ về lỗ hổng này bao gồm:

* Ứng dụng dựa vào các plugin, thư viện hoặc module từ các nguồn không tin cậy, kho lưu trữ không xác thực, hoặc mạng phân phối nội dung (CDN).
* Quy trình tích hợp liên tục và triển khai liên tục (CI/CD) không an toàn có thể dẫn đến truy cập trái phép, mã độc hại hoặc bị xâm phạm hệ thống.
* Chức năng tự động cập nhật không đảm bảo việc kiểm tra tính toàn vẹn, dẫn đến khả năng kẻ tấn công tải lên các bản cập nhật độc hại và phân phối tới tất cả các cài đặt.
* Dữ liệu hoặc đối tượng được mã hóa hoặc tuần tự hóa có thể bị kẻ tấn công xem và sửa đổi, dễ dẫn đến lỗ hổng deserialization không an toàn.

## Kịch bản:

Trong trường hợp này, lỗ hổng xảy ra khi token serialize bị kẻ tấn công xem và sửa đổi

## Các bước để thực hiện:

1. **Dùng Burpsuite chặn request đăng nhập**
2. **Phân tích request**

* Ta thấy token được giải mã theo Base64 có dạng như sau:

A screenshot of a phone

Description automatically generated

* Dự đoán \0 là quyền admin
* Đổi \0 thành 01

A screenshot of a chat

Description automatically generated

1. **Kết quả**

A screenshot of a computer

Description automatically generated

## Tài liệu hỗ trợ và tham khảo:

[https://owasp.org/Top10/A08\_2021-Software\_and\_Data\_Integrity\_Failures/](%20https://owasp.org/Top10/A08_2021-Software_and_Data_Integrity_Failures/)

## Mức độ ảnh hưởng của lỗ hổng:

* Truy cập trái phép: Kẻ tấn công có thể chiếm quyền kiểm soát hệ thống thông qua việc cài đặt mã độc hoặc thao tác dữ liệu.
* Mất mát dữ liệu: Việc sửa đổi hoặc xóa dữ liệu quan trọng có thể dẫn đến tổn thất về tài chính và uy tín.
* Độ tin cậy của phần mềm: Người dùng có thể mất niềm tin vào sản phẩm nếu phát hiện có lỗ hổng an ninh nghiêm trọng.
* Khả năng tương tác với các ứng dụng khác: Lỗ hổng có thể ảnh hưởng đến khả năng tích hợp và tương tác của ứng dụng với các dịch vụ bên ngoài.

## Khuyến cáo khắc phục:

1. Sử dụng chữ ký số: Áp dụng chữ ký số hoặc các cơ chế tương tự để xác minh rằng phần mềm hoặc dữ liệu đến từ nguồn mong đợi và không bị thay đổi.
2. Kiểm soát kho lưu trữ: Đảm bảo rằng các thư viện và phụ thuộc, chẳng hạn như npm hoặc Maven, chỉ tiêu thụ từ các kho lưu trữ đáng tin cậy. Đối với các tổ chức có mức độ rủi ro cao hơn, hãy xem xét việc lưu trữ một kho lưu trữ nội bộ đã được kiểm duyệt.
3. Sử dụng công cụ bảo mật chuỗi cung ứng phần mềm: Sử dụng các công cụ như OWASP Dependency Check hoặc OWASP CycloneDX để xác minh rằng các thành phần không chứa các lỗ hổng đã biết.
4. Quy trình xem xét: Đảm bảo rằng có một quy trình xem xét cho các thay đổi mã và cấu hình để giảm thiểu khả năng mã độc hoặc cấu hình độc hại được đưa vào quy trình phần mềm.
5. Tách biệt quy trình CI/CD: Đảm bảo rằng quy trình CI/CD có sự phân tách, cấu hình và kiểm soát truy cập phù hợp để đảm bảo tính toàn vẹn của mã trong quy trình xây dựng và triển khai.
6. Kiểm tra toàn vẹn dữ liệu: Đảm bảo rằng dữ liệu tuần tự hóa không được gửi đến các khách hàng không đáng tin cậy mà không có một dạng kiểm tra toàn vẹn hoặc chữ ký số để phát hiện việc sửa đổi hoặc phát lại dữ liệu tuần tự.

# Bài tập 4: Báo cáo lỗ hổng đang được thực hành.

## **Tiêu đề:** Lỗ hổng Security Logging and Monitoring Failures

## Mô tả:

Thiếu sót trong việc ghi nhật ký, phát hiện, giám sát và phản ứng sẽ dẫn đến việc không thể phát hiện ra các sự cố vi phạm bảo mật. Điều này xảy ra khi các sự kiện có thể kiểm tra, như đăng nhập, đăng nhập thất bại, và các giao dịch có giá trị cao không được ghi lại. Việc không có hoặc có nhật ký không đầy đủ, không rõ ràng hoặc không được giám sát thường xuyên cho các hoạt động nghi ngờ cũng là một lỗ hổng. Ngoài ra, thiếu sự giám sát hoặc cảnh báo cho các cuộc tấn công hoặc hoạt động đáng ngờ có thể dẫn đến sự rò rỉ thông tin và các vấn đề bảo mật nghiêm trọng khác.

## Kịch bản:

Trong trường hợp này, thông tin đăng nhập đã bị rò rỉ trong các nhật ký.

## Phân tích log:

1. **Cách truy cập:**

<http://0.0.0.0:8000/debug>

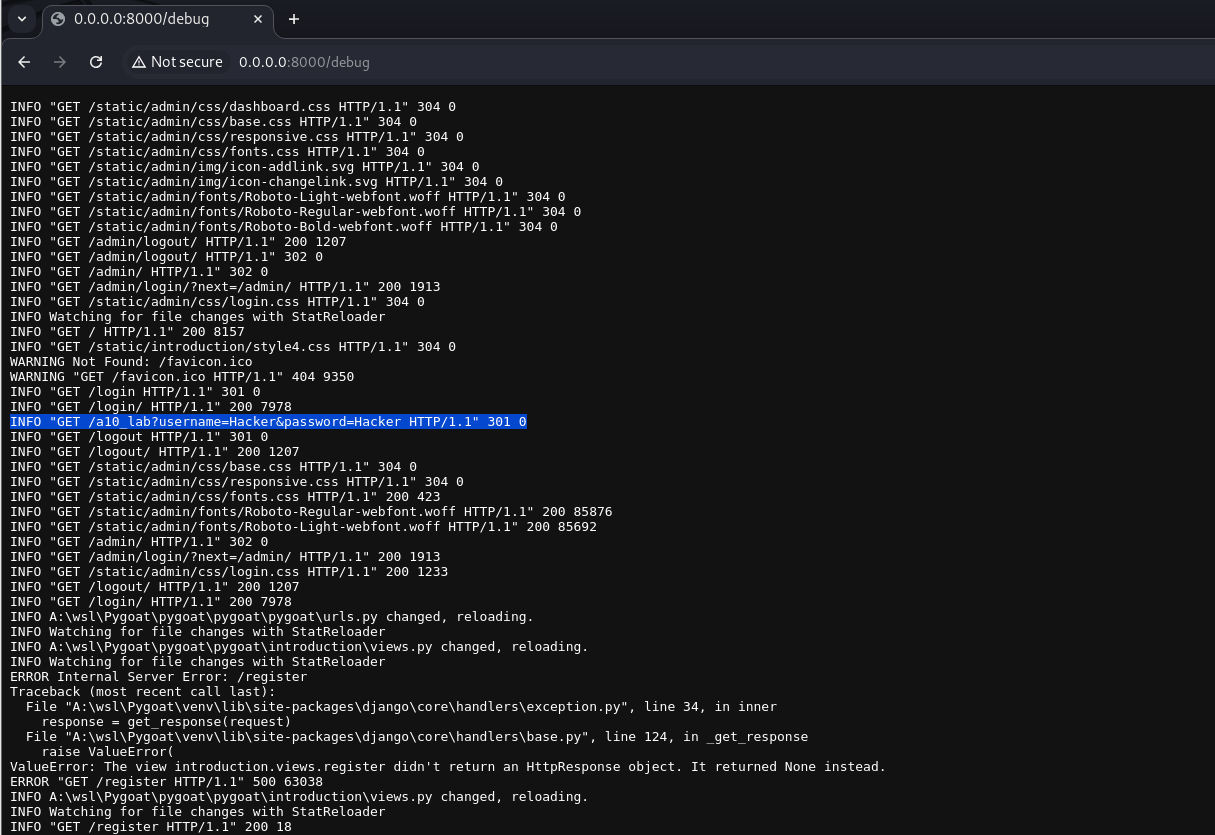
1. **Bài thực hành ghi log ở đâu?**

* Log truy cập từ ứng dụng Django. Đây là các bản ghi ghi lại các yêu cầu HTTP mà máy chủ nhận được và phản hồi tương ứng từ máy chủ.

1. **Thông tin nhạy cảm có thể được tiết lộ từ vị trí nào của log?**

* Yêu cầu đến các đường dẫn nhạy cảm:
  + Các đường dẫn như ***/login*** và ***/register*** có thể tiết lộ rằng người dùng đang cố gắng đăng nhập hoặc đăng ký tài khoản. Điều này có thể được xem là thông tin nhạy cảm nếu có nhiều người dùng truy cập vào trang web.
* Thông tin lỗi:
  + Các thông báo lỗi như ***ValueError:*** ***The view introduction.views.register didn't return an HttpResponse object*** có thể tiết lộ cách mà ứng dụng xử lý các yêu cầu, giúp kẻ tấn công biết thêm về cấu trúc ứng dụng và các lỗ hổng tiềm ẩn.
* Đường dẫn đến các tệp và thư mục:
  + Các đường dẫn tệp như ***A:\wsl\Pygoat\pygoat\pygoat\introduction\views.py*** có thể tiết lộ cấu trúc thư mục và tệp trong máy chủ, có thể cho phép kẻ tấn công tìm kiếm các tệp quan trọng khác.
* Thông tin từ các yêu cầu GET có chứa tham số nhạy cảm:
  + Ví dụ, đường dẫn như ***/a10\_lab?username=Hacker&password=Hacker*** có thể tiết lộ thông tin đăng nhập giả mạo. Nếu đây là thông tin thật, nó có thể dẫn đến việc xâm nhập tài khoản.
* Thông báo 404 Not Found:
  + Thông báo này có thể cho thấy rằng có các đường dẫn hoặc tài nguyên không tồn tại, có thể cho phép kẻ tấn công thử nghiệm các đường dẫn khác.
* Thông báo về CSRF token:
  + Nếu thông báo nào đó đề cập đến CSRF token, điều này có thể tiết lộ cách thức mà ứng dụng bảo vệ chống lại các cuộc tấn công giả mạo yêu cầu giữa các trang (CSRF).
* Các địa chỉ IP và User-Agent:
  + Nếu log ghi lại địa chỉ IP và User-Agent của người dùng, thông tin này có thể được sử dụng để theo dõi hoặc tấn công

## Minh chứng:



## Tài liệu hỗ trợ và tham khảo:

<https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/>

## Mức độ ảnh hưởng của lỗ hổng:

Khi không có hệ thống giám sát và ghi nhật ký đủ mạnh, tổ chức sẽ gặp khó khăn trong việc phát hiện và phản hồi kịp thời các sự cố bảo mật, dẫn đến nguy cơ thông tin bị đánh cắp, các hoạt động bất hợp pháp được tiến hành mà không bị phát hiện, và khó khăn trong việc điều tra pháp lý sau khi xảy ra sự cố. Ngoài ra, nếu thông tin nhật ký bị rò rỉ hoặc lộ ra cho người dùng hoặc tin tặc, nguy cơ bị tấn công và khai thác thêm các lỗ hổng sẽ tăng cao, ảnh hưởng nghiêm trọng đến uy tín và hoạt động của tổ chức.

## Khuyến cáo khắc phục:

1. Đảm bảo rằng tất cả các sự kiện quan trọng như đăng nhập, kiểm soát truy cập, và các lỗi xác thực ở phía máy chủ đều được ghi lại với đủ thông tin để xác định các tài khoản đáng ngờ hoặc độc hại, đồng thời lưu trữ chúng đủ lâu để phân tích pháp lý chậm trễ.
2. Đảm bảo rằng nhật ký được tạo ra ở định dạng có thể dễ dàng tiêu thụ bởi các giải pháp quản lý nhật ký.
3. Mã hóa dữ liệu nhật ký đúng cách để ngăn chặn các cuộc tấn công injection hoặc tấn công vào hệ thống giám sát và ghi nhật ký.
4. Các nhóm DevSecOps nên thiết lập hệ thống giám sát và cảnh báo hiệu quả để phát hiện và phản hồi nhanh chóng các hoạt động đáng ngờ.
5. Xây dựng hoặc áp dụng kế hoạch phản ứng và phục hồi sự cố, chẳng hạn như NIST 800-61r2 hoặc phiên bản mới hơn.

# Bài tập 5: Báo cáo lỗ hổng đang được thực hành.

## **Tiêu đề:** Lỗ hổng Server-Side Request Forgery trong trang web blog

## Mô tả:

Server-Side Request Forgery là một lỗ hổng xảy ra khi ứng dụng web thực hiện yêu cầu đến tài nguyên từ xa mà không kiểm tra và xác thực kỹ lưỡng URL do người dùng cung cấp. Điều này cho phép kẻ tấn công buộc ứng dụng gửi các yêu cầu được tùy chỉnh đến các đích không mong muốn, ngay cả khi các đích này được bảo vệ bởi tường lửa, VPN, hoặc các danh sách kiểm soát truy cập mạng (ACL). Kẻ tấn công có thể lợi dụng SSRF để truy cập vào các hệ thống nội bộ, tấn công mạng nội bộ, hoặc tương tác với các dịch vụ bên trong mà thường không được công khai.

## Kịch bản:

Trong trường hợp này, lỗ hổng nằm ở việc kẻ tấn công có thể thay đổi tham số blog để trỏ đến các tệp nhạy cảm hơn hoặc các URL nội bộ của máy chủ.

A screenshot of a computer

Description automatically generated

## Các bước khai thác:

1. **Dùng Burpsuite chặn request**

A screenshot of a computer

Description automatically generated

1. **Gửi yêu cầu POST:**

Kẻ tấn công có thể gửi một yêu cầu POST với tham số blog, chỉ định tệp cần đọc. Mặc dù ứng dụng cho phép truy cập vào các tệp trong thư mục templates/Lab/ssrf/blogs/, nhưng không có kiểm tra chặt chẽ nào để đảm bảo rằng kẻ tấn công không thể truy cập vào các tệp khác trên hệ thống. Ví dụ blog có giá trị là ../../../../etc/passwd

A white background with black and red text

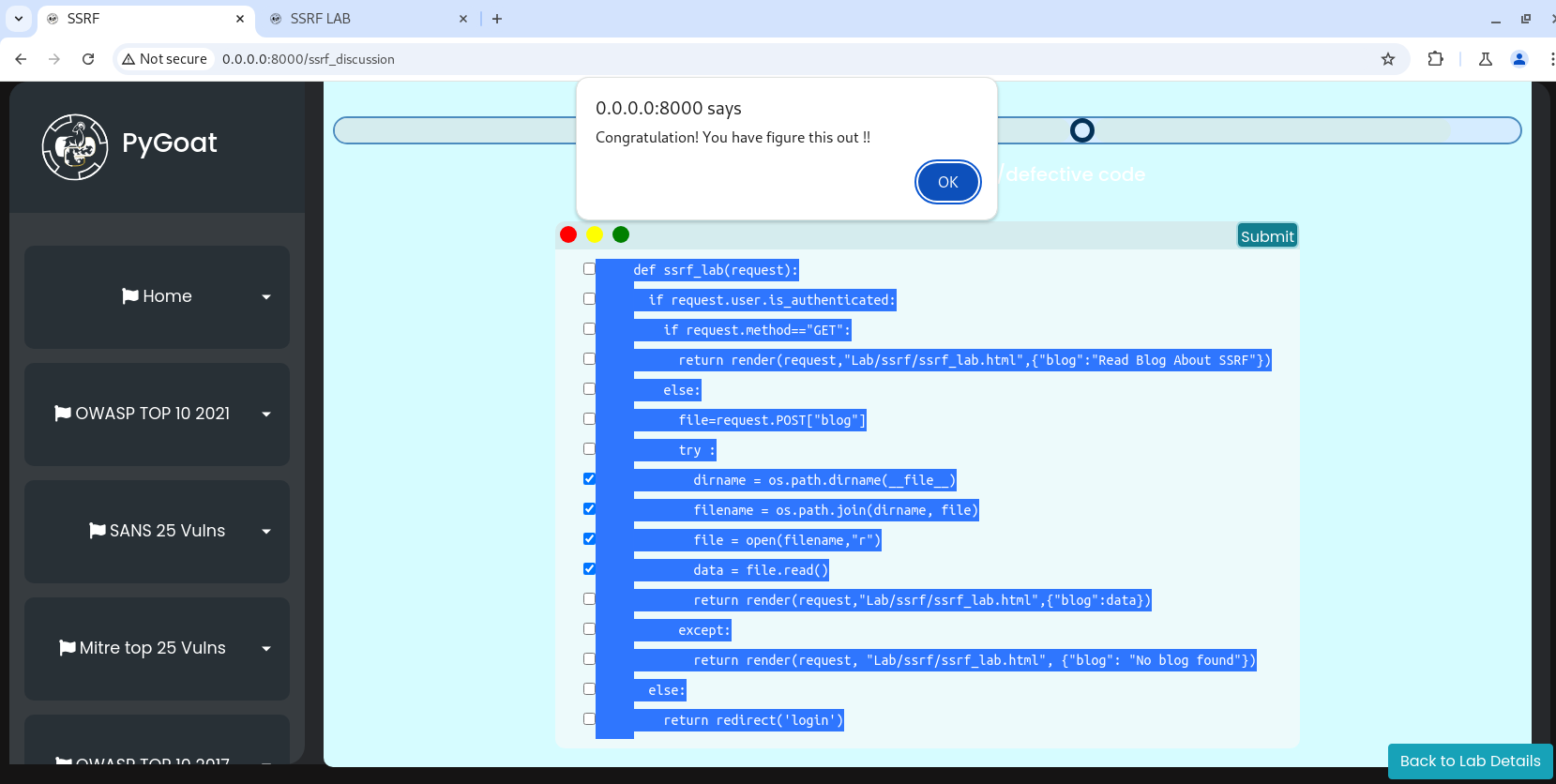
Description automatically generated

1. **Kết quả:**

A screenshot of a computer

Description automatically generated

## Minh chứng lỗ hổng hệ thống:



*Hình. Lỗi ở views.py*

A computer screen shot of a chat

Description automatically generated

*Hình. Lỗi ở ssrf\_lab.html*

## Tài liệu hỗ trợ và tham khảo:

<https://owasp.org/Top10/A10_2021-Server-Side_Request_Forgery_%28SSRF%29/>

## Mức độ ảnh hưởng của lỗ hổng:

* Truy cập trái phép vào hệ thống nội bộ: Kẻ tấn công có thể sử dụng SSRF để truy cập vào các hệ thống và dịch vụ nội bộ mà thường không thể tiếp cận từ bên ngoài, phá vỡ các rào cản bảo mật như tường lửa và VPN.
* Lấy thông tin nhạy cảm: Kẻ tấn công có thể lấy cắp thông tin quan trọng từ các dịch vụ nội bộ, bao gồm thông tin xác thực, dữ liệu cá nhân hoặc dữ liệu nhạy cảm khác.
* Thao tác dịch vụ nội bộ: Kẻ tấn công có thể gửi các yêu cầu được tùy chỉnh để thao tác các dịch vụ hoặc gây thiệt hại cho hệ thống nội bộ, từ việc làm gián đoạn dịch vụ đến xâm nhập và chiếm quyền kiểm soát các tài khoản hệ thống.
* Tấn công đến các hệ thống khác: SSRF có thể được sử dụng để làm bàn đạp thực hiện các cuộc tấn công tiếp theo, chẳng hạn như tấn công từ chối dịch vụ (DoS), tấn công vào hệ thống đám mây, hoặc chiếm quyền điều khiển các tài khoản quản trị hệ thống..

## Khuyến cáo khắc phục:

1. Kiểm tra và xác thực dữ liệu đầu vào: Kiểm tra và làm sạch toàn bộ dữ liệu do người dùng cung cấp, đặc biệt là các URL. Chỉ cho phép URL từ các nguồn tin cậy.
2. Sử dụng danh sách cho phép: Áp dụng danh sách cho phép (allow list) để xác thực schema URL, cổng và đích truy cập. Điều này đảm bảo chỉ có những URL hợp lệ được phép sử dụng.
3. Giới hạn quyền truy cập: Đảm bảo rằng ứng dụng web chỉ có quyền truy cập vào các tệp cần thiết, không cho phép truy cập vào các tệp hệ thống.
4. Ghi log và theo dõi: Ghi log các yêu cầu và theo dõi các hoạt động đáng ngờ để phát hiện các nỗ lực tấn công.

**HẾT**