BÁO CÁO THỰC HÀNH

**Môn học: Bảo mật web và ứng dụng**

**Kỳ báo cáo: Buổi 02 (Session 02)**

**Tên chủ đề: Top 10 OSWAP**

*GVHD: Ngô Khánh Khoa*

*Ngày báo cáo: 09/04/2024*

1. **THÔNG TIN CHUNG:**

*(Liệt kê tất cả các thành viên trong nhóm)*

Lớp: NT213.O21.ANTN

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Email** |
| 1 | Phạm Ngọc Thơ | 21522641 | 21522641@gm.uit.edu.vn |

1. **NỘI DUNG THỰC HIỆN:[[1]](#footnote-1)**

|  |  |  |
| --- | --- | --- |
| **STT** | **Công việc** | **Kết quả tự đánh giá** |
| 1 | Kịch bản 01 | 100% |
| 2 | Kịch bản 02 | 100% |
| 3 | Kịch bản 03 | 100% |
| 4 | Kịch bản 04 | 100% |
| 5 | Kịch bản 05 | 100% |

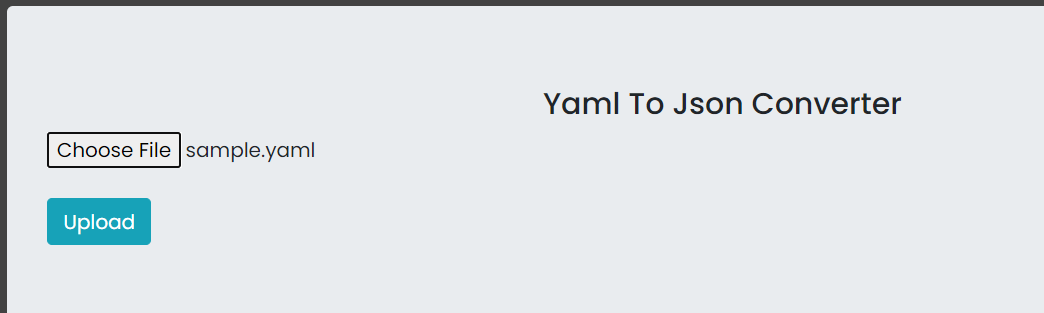
**Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.**

BÁO CÁO CHI TIẾT

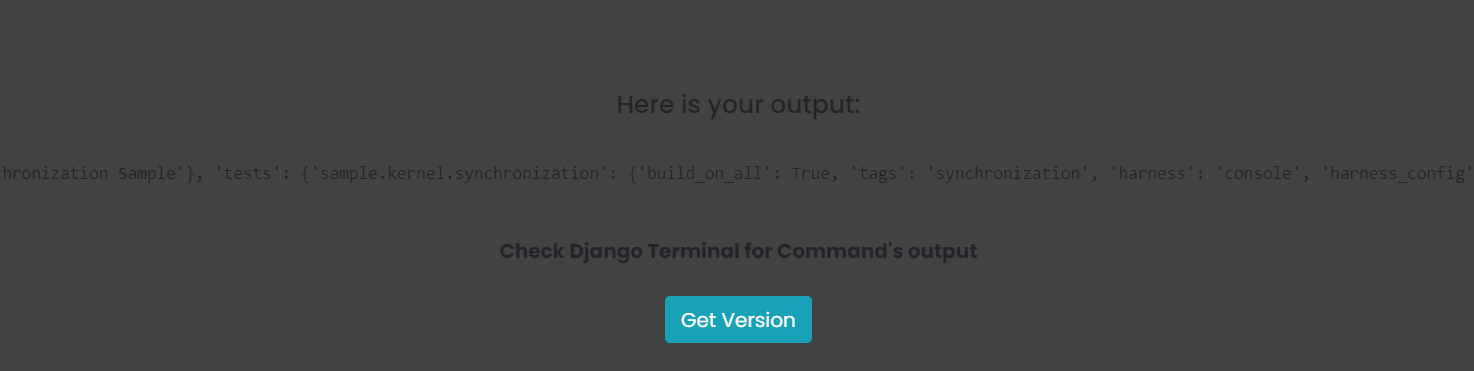
## **A06:2021 – Vulnerable and Outdated Components**

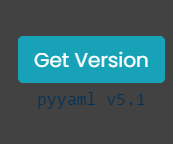
1. Chậm lại và suy nghĩ 1: Lỗ hổng tồn tại ở đây là gì, kiểm tra và khai thác như thế nào?

* Thực hiện chuyển đổi thử từ tập tin \*.yaml sang định dạng json. Tập tin sample.yaml được cung cấp kèm theo:



* Kết quả:





* Ở đây, ta có thể xem được version mà web đang sử dụng là pyyaml v5.1 - một phiên bản dễ bị thực thi mã. Đây cũng chính là lỗ hổng của lab này vì sử dụng thư viện dễ bị tấn công.
* Chúng ta có thể khai thác lỗ hổng bằng cách chèn nội dung mong muốn vào file \*.yaml. Các bước thực hiện chi tiết, em sẽ trình bày trong bài tập 1.

1. Bài tập 1: Báo cáo lỗ hổng đang được thực hành. Có thể sử dụng format theo mẫu sau:

**#Tiêu đề**: Vulnerable and Outdated Components làm tiết lộ thông tin nhạy cảm của người dùng và phá hủy danh tiếng của nhà cung cấp ứng dụng.

**#Mô tả lỗ hổng**:

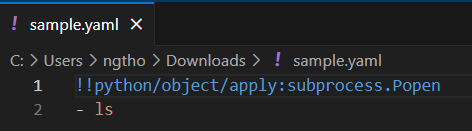
### Tóm tắt: Các ứng dụng web được xây dựng bằng cách sử dụng kết hợp các thành phần phần mềm, bao gồm thư viện, framework và OS. Các thành phần này có thể được phát triển nội bộ hoặc lấy từ các nhà cung cấp bên thứ ba.

* Các thành phần dễ bị tổn thương chứa các lỗ hổng bảo mật mà kẻ tấn công có thể khai thác để truy cập trái phép vào dữ liệu nhạy cảm hoặc chiếm quyền kiểm soát hệ thống. Những lỗ hổng này có thể tồn tại trong nhiều năm trước khi chúng được phát hiện và vá lỗi.
* Các thành phần lỗi thời không còn được nhà phát triển hỗ trợ nữa. Điều này có nghĩa là nó sẽ không nhận được các bản cập nhật bảo mật, khiến nó càng dễ bị khai thác hơn.

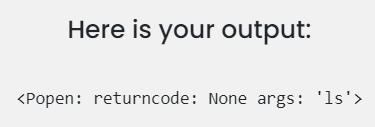
Việc sử dụng các thành phần có lỗ hổng đã biết sẽ khiến ứng dụng dễ bị tấn công nhắm vào bất kỳ phần nào của ngăn xếp ứng dụng. Ví dụ: code injection, buffer overflow, XSS, command injection.

### Các bước để thực hiện lại và bằng chứng:

1. bước 1: Theo hướng dẫn tại “Lab 1 Detail”, em sẽ sửa đổi nội dung file “sample.yaml” thành:



2. bước 2: Sau khi convert từ .yaml sang .json, trang web phản hồi:



3. bước 3: Từ bước 3, ta có thể thấy rằng chương trình đã thực thi thành công lệnh ls, và dòng output là kết quả của lệnh này nhưng không có giá trị trả về (popen: returncode).

### Tài liệu hỗ trợ và tham khảo: [link](https://medium.com/@nidhirohilla.nr/vulnerable-and-outdated-components-47d907f8aa09)

**#Mức độ ảnh hưởng của lỗ hổng**: Tác động bảo mật nào mà kẻ tấn công có thể đạt được?

* Vi phạm dữ liệu: Kẻ tấn công có thể khai thác lỗ hổng để đánh cắp dữ liệu nhạy cảm của người dùng, chẳng hạn như số thẻ tín dụng và thông tin cá nhân.
* Xâm phạm hệ thống: Việc xâm nhập có thể dẫn đến việc chiếm quyền kiểm soát hoàn toàn hệ thống, cho phép kẻ tấn công làm gián đoạn hoạt động, cài đặt phần mềm độc hại hoặc tiến hành các cuộc tấn công tiếp theo.
* Thiệt hại về danh tiếng: Vi phạm bảo mật có thể làm xói mòn lòng tin của người dùng và gây tổn hại đến danh tiếng thương hiệu.

**#Khuyến cáo khắc phục**: Làm thế nào để vá lỗ hổng này?

* Duy trì kho phần mềm: Tạo danh sách đầy đủ tất cả các thành phần được sử dụng trong ứng dụng, bao gồm cả các phiên bản và thành phần phụ thuộc của chúng.
* Thực hiện quy trình quản lý bản vá: Thường xuyên cập nhật các thành phần với các bản vá bảo mật mới nhất. Ưu tiên các lỗ hổng nghiêm trọng và yếu tố ảnh hưởng đến ứng dụng khi cập nhật.
* Sử dụng các phương pháp mã hóa an toàn: Các nhà phát triển nên được đào tạo về các phương pháp mã hóa an toàn để giảm thiểu việc tạo ra các lỗ hổng ngay từ đầu.
* Cập nhật thông tin: Đăng ký nhận tư vấn bảo mật cho các thành phần đang sử dụng. Điều này sẽ giúp chúng ta cập nhật các lỗ hổng mới nhất và có hành động kịp thời.
* Xem xét các lựa chọn thay thế: Nếu một thành phần không còn được hỗ trợ, hãy tìm kiếm những sự lựa chọn thay thế tốt hơn.
* Chỉ sử dụng các thành phần từ các nguồn chính thức thông qua một liên kết an toàn.

1. **A07:2021 – Identification and Authentication Failures**

Bài tập 2: Báo cáo lỗ hổng đang được thực hành. Có thể sử dụng format theo mẫu sau:

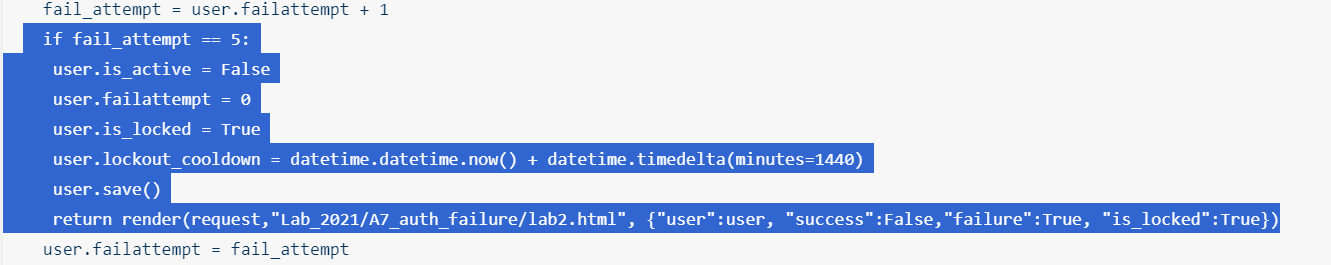
**#Tiêu đề**: Identification and Authentication Failures - lỗi xác nhận và xác thực tài khoản.

**#Mô tả lỗ hổng**:

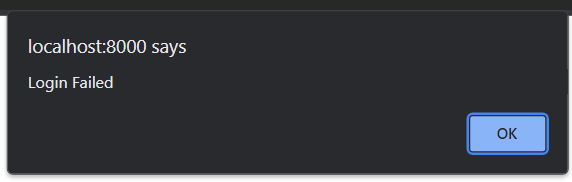
### Tóm tắt: Identification and Authentication Failures trước đây là Broken Authentication, xảy ra khi hệ thống không thực hiện đúng các biện pháp xác định (identification) và xác thực (authentication) người dùng.

### Các bước để thực hiện lại và bằng chứng:

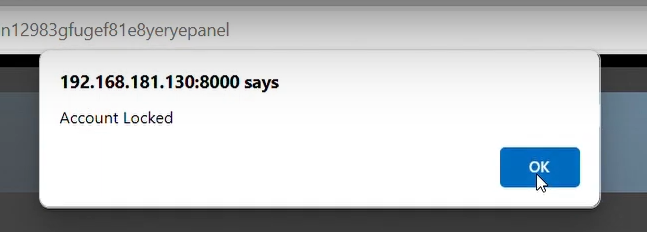
1. bước 1: lab 2 là một trang đăng nhập, yêu cầu người dùng nhập tên người dùng và mật khẩu của họ. Email của quản trị viên là admin\_pygoat@pygoat.com. Password hash là $argon2id$v=19$m=65536,t=3,p=4$Ub40KHiEbH9I3Bsd4VHQDA$4zsIHDmAbejFJmaZq8a2yVIJdHvfylDlQ85w3YRLMSQ. Việc cần làm không phải mà đăng nhập mà chặn quyền truy cập của admin trong một thời gian cố định. Sau khi xem source, nếu đăng nhập quá 5 lần thì sẽ bị khóa trong 1440 phút (24h):



2. bước 2: Đầu tiên, em sẽ nhập ngẫu nhiên một password như 123456789. Hiển thị thông báo đăng nhập thất bại:



3.Bước 3: Lặp lại thao tác trên 4 lần với email admin đã cung cấp. Cuối cùng, được kết quả:



Lỗi này sẽ làm cho admin thật cũng không thể đăng nhập vào hệ thống sau 24h tiếp theo.

### Tài liệu hỗ trợ và tham khảo: [Top10/A07\_2021-Identification\_and\_Authentication\_Failures/](https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/)

**#Mức độ ảnh hưởng của lỗ hổng**: Tác động bảo mật nào mà kẻ tấn công có thể đạt được?

* Đánh cắp thông tin cá nhân.
* Truy cập trái phép vào hệ thống.
* Tiết lộ thông tin nhạy cảm.
* Thực thi mã độc hại.
* Quản lý phiên không được vận hành đúng cách.

**#Khuyến cáo khắc phục**: Làm thế nào để vá lỗ hổng này?

* Kích hoạt và thực thi MFA (Multi-factor Authentication).
* Đảm bảo rằng mật khẩu mặc định chỉ được sử dụng một lần và được cập nhật khi người dùng đăng nhập.
* Thực thi chính sách mật khẩu để ngăn người dùng đặt mật khẩu yếu.
* Giới hạn số lần đăng nhập khi người dùng đăng nhập không chính xác.
* Mã thông báo phiên nhưng phải đủ an toàn, mã thông báo không được truyền trong URL, được lưu trữ an toàn và bị vô hiệu hóa sau khi đăng xuất, không sử dụng hoặc khi hết thời gian chờ.
* Khuyến nghị pentester cần kiểm thử cẩn thận ứng dụng.
* Ghi lại tất cả các lỗi và cảnh báo cho quản trị viên khi phát hiện việc nhồi thông tin xác thực, brute force hoặc các cuộc tấn công khác.
* Sử dụng trình quản lý phiên tích hợp, an toàn, phía máy chủ để tạo ID phiên ngẫu nhiên mới với entropy cao sau khi đăng nhập.

1. **A08:2021 – Software and Data Integrity Failures**

Bài tập 3: Báo cáo lỗ hổng đang được thực hành. Có thể sử dụng format theo mẫu sau:

**#Tiêu đề**: Software and Data Integrity Failures - Lỗi toàn vẹn dữ liệu và phần mềm.

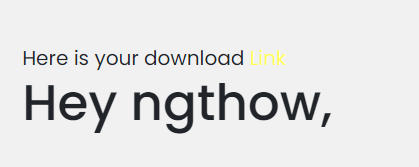
**#Mô tả lỗ hổng**:

### Tóm tắt: Lỗi toàn vẹn dữ liệu và phần mềm (Software and Data Integrity Failures) xảy ra khi dữ liệu hoặc phần mềm bị thay đổi hoặc bị ảnh hưởng mà không có sự cho phép hoặc kiểm soát của người quản trị hoặc người dùng cuối. Điều này có thể làm suy giảm tính toàn vẹn của dữ liệu và tính toàn vẹn của phần mềm. Các lỗi về tính toàn vẹn của phần mềm và dữ liệu liên quan đến code và cơ sở hạ tầng không bảo vệ khỏi các vi phạm tính toàn vẹn.

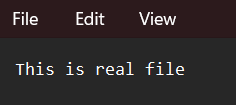
* Ứng dụng dựa vào các plugin, thư viện hoặc mô-đun không đáng tin cậy, không an toàn. Dẫn đến truy cập trái phép, thực thi các mã độc hại hoặc xâm nhập hệ thống
* Tự động cập nhật các bản cập nhật mà không xác minh tính toàn vẹn đầy đủ và được áp dụng cho phiên bản trước đó

### Các bước để thực hiện lại và bằng chứng:

1. bước 1: Kịch bản này bao gồm một trang chức năng là hiển thị trang download tài liệu cho người dùng:

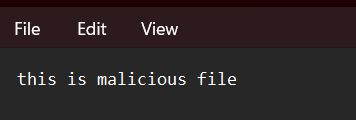


Sau khi nhập tên user, sẽ được cấp 1 đường link và file text tải về là *real.txt* với nội dung:



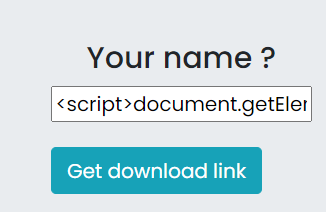
2. bước 2: Ngoài ra, page còn cung cấp cho ta một file *fake.txt*:

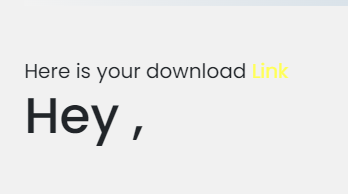




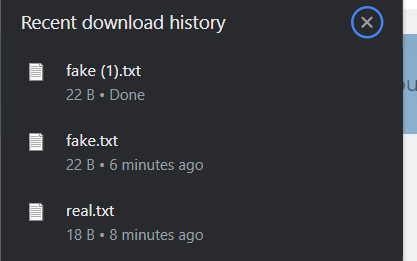
3. bước 3: Nhiệm vụ của lab này là khi đăng nhập vào, user thay vì tải về file *real.txt*, sẽ tải về file *fake.txt*. Chúng ta có thể thực hiện được điều này bằng cách sử dụng một đoạn Javascript:

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





- Kết quả: thành công thay thế được file:



### Tài liệu hỗ trợ và tham khảo: <https://viblo.asia/p/tong-quan-ve-top-ten-owasp-2021-6J3ZgR7BKmB>

**#Mức độ ảnh hưởng của lỗ hổng**: Tác động bảo mật nào mà kẻ tấn công có thể đạt được?

* Rủi ro bảo mật: Lỗ hổng toàn vẹn dữ liệu và phần mềm có thể tạo ra rủi ro bảo mật nghiêm trọng. Nó có thể cho phép kẻ tấn công thực hiện các cuộc tấn công như thực thi mã độc hại hoặc lợi dụng dữ liệu nhạy cảm.
* Mất mát dữ liệu.
* Gây ra tình trạng không ổn định, làm gián đoạn hoạt động kinh doanh và dịch vụ.
* Thất thoát tài chính và uy tín: Sự mất mát dữ liệu, thất bại hệ thống và các vấn đề liên quan có thể dẫn đến thất thoát tài chính đáng kể, bao gồm chi phí để khắc phục sự cố, mất mát doanh thu, và các hậu quả pháp lý.

**#Khuyến cáo khắc phục**: Làm thế nào để vá lỗ hổng này?

* Kiểm tra và lọc đầu vào để chặn việc chèn script.
* Không chấp nhận đối tượng serialized từ nguồn không tin cậy.
* Thực hiện kiểm tra toàn vẹn như chữ ký số trên bất kỳ đối tượng serialized nào để ngăn chặn việc tạo đối tượng độc hại hoặc sửa đổi dữ liệu: Sử dụng các phương pháp như chữ ký số để đảm bảo tính toàn vẹn của dữ liệu serialized và ngăn chặn việc tạo ra đối tượng độc hại hoặc sửa đổi dữ liệu.
* Thực hiện các ràng buộc kiểu nghiêm ngặt trong quá trình deserialize trước khi tạo ra đối tượng.
* Cô lập và chạy mã deserialize trong môi trường có đặc quyền thấp khi có thể để giảm thiểu nguy cơ tấn công.
* Ghi log các ngoại lệ và sự cố trong quá trình deserialize, chẳng hạn như khi loại đầu vào không phải là loại mong đợi hoặc quá trình deserialize gây ra ngoại lệ.
* Hạn chế hoặc giám sát kết nối mạng vào và ra từ các container hoặc máy chủ thực hiện deserialize.

1. **A09:2021 – Security Logging and Monitoring Failures**

Bài tập 4: Báo cáo lỗ hổng đang được thực hành. Có thể sử dụng format theo mẫu sau:

**#Tiêu đề**: Security Logging and Monitoring Failures - Các lỗi trong quá trình theo dõi và ghi nhật ký bảo mật.

**#Mô tả lỗ hổng**:

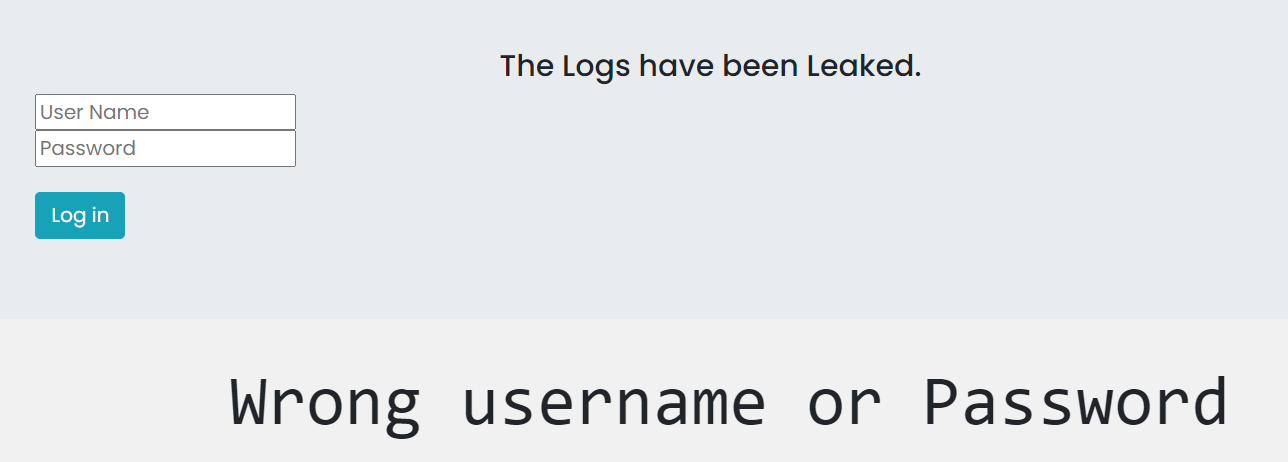
### Tóm tắt: Lỗi trong việc ghi nhật ký bảo mật và giám sát (Security Logging and Monitoring Failures) xảy ra khi hệ thống không thể hoặc không ghi nhật ký hoạt động bảo mật một cách đầy đủ hoặc không thực hiện giám sát hiệu quả đối với các hoạt động quan trọng. Đây là nguyên nhân nền tảng của hầu hết mọi sự cố lớn. Những kẻ tấn công dựa vào việc thiếu giám sát và phản ứng kịp thời để đạt được mục tiêu mà không bị phát hiện.

### Các bước để thực hiện lại và bằng chứng:

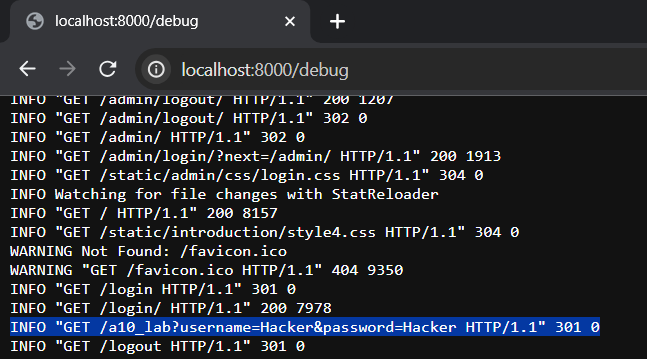
1. bước 1: Trong bài lab này, người dùng được cung cấp một trang đăng nhập cho phép họ tìm hiểu về cách một việc ghi nhật ký không đúng cách có thể dẫn đến tiết lộ thông tin. Người dùng cần tìm ra thông tin nhật ký đã bị rò rỉ và cố gắng lấy được các thông tin đăng nhập đã bị rò rỉ trong các nhật ký.

Để tìm thông tin nhật ký, người dùng cần truy cập vào đường dẫn /debug, có thể là thông qua brute force với các subdomain hoặc chỉ đơn giản là đoán đường dẫn. Sau khi tìm ra thông tin nhật ký, họ có thể cố gắng lấy thông tin đăng nhập đã rò rỉ do việc ghi nhật ký không đúng cách.

2. bước 2: Đăng nhập ngẫu nhiên, đương nhiên là sai bét nhè rùi ạ



3. bước 3: Theo gợi ý của lab thì em sẽ tìm trong /debug, kết quả tìm được username=Hacker và password=Hacker:



### Tài liệu hỗ trợ và tham khảo: <https://viblo.asia/p/tong-quan-ve-top-ten-owasp-2021-6J3ZgR7BKmB>

**#Mức độ ảnh hưởng của lỗ hổng**: Tác động bảo mật nào mà kẻ tấn công có thể đạt được?

* Giảm khả năng phát hiện sự tấn công: Khi hệ thống không ghi lại hoặc giám sát các hoạt động bất thường, kẻ tấn công có thể thực hiện các cuộc tấn công mà không bị phát hiện.
* Phản ứng và phục hồi không hiệu quả: Nếu không có hệ thống ghi nhật ký hiệu quả và giám sát, tổ chức sẽ không thể phản ứng kịp thời và hiệu quả khi xảy ra sự cố bảo mật. Điều này làm giảm khả năng phục hồi sau các sự kiện xâm nhập hoặc các vấn đề bảo mật khác.
* Mất mát dữ liệu quan trọng: Khi không ghi nhật ký một cách đầy đủ, các hoạt động quan trọng hoặc các sự kiện bảo mật có thể không được ghi lại. Điều này có thể dẫn đến việc mất mát dữ liệu quan trọng hoặc không thể khôi phục được.
* Thiếu chứng cứ và phân tích sau cùng: Việc không có dữ liệu ghi nhật ký đầy đủ có thể làm suy giảm khả năng thu thập chứng cứ và thực hiện phân tích sau cùng về các sự kiện bảo mật. Điều này làm giảm khả năng phát hiện và hiểu biết về các mối đe dọa tiềm ẩn và các hoạt động xâm nhập.
* Không tuân thủ quy định pháp lý: Trong nhiều trường hợp, các tổ chức cần tuân thủ các quy định pháp lý hoặc tiêu chuẩn ngành về việc ghi nhật ký và giám sát. Việc thất bại trong việc tuân thủ các quy định này có thể dẫn đến hậu quả pháp lý và tài chính nghiêm trọng.

**#Khuyến cáo khắc phục**: Làm thế nào để vá lỗ hổng này?

* Đảm bảo rằng nhật ký được tạo ở định dạng mà các công cụ quản lý nhật ký trung tâm có thể dễ dàng sử dụng.
* Các giao dịch có giá trị cao phải có quy trình kiểm tra với các biện pháp kiểm soát tính toàn vẹn để ngăn chặn việc thao túng hoặc xóa.
* Thiết lập việc giám sát và cảnh báo để có thể phát hiện và phản hồi kịp thời các hoạt động đáng ngờ.
* Không ghi lại các thông tin nhạy cảm như mật khẩu.

1. **A10:2021 – Server-Side Request Forgery (SSRF)**

Bài tập 5: Báo cáo lỗ hổng đang được thực hành. Có thể sử dụng format theo mẫu sau:

**#Tiêu đề**: Server-side Request Forgecy - Giả mạo yêu cầu phía máy chủ.

**#Mô tả lỗ hổng**:

### Tóm tắt: SSRF là lỗ hổng mà kẻ tấn công thường tận dụng các chức năng của ứng dụng web để tạo ra các yêu cầu từ phía máy chủ mà máy chủ không kiểm soát được hoặc không nên truy cập. Điều này cho phép kẻ tấn công tấn công vào các hệ thống nội bộ, các dịch vụ back-end, hoặc các hệ thống không được công khai.

### Các bước để thực hiện lại và bằng chứng:

1. bước 1: Truy cập vào lab:

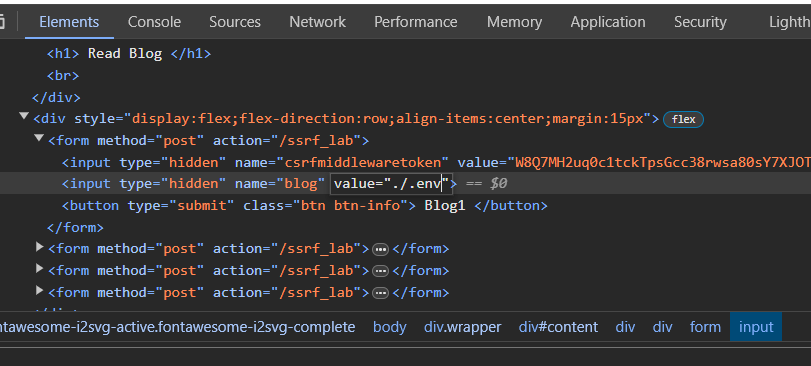


Gợi ý là tìm 1 file .env và được cung cấp source code viết bằng Python như trên.

2. bước 2: Một dự án Django thì sẽ có nhiều tệp và thư mục quan trọng như:

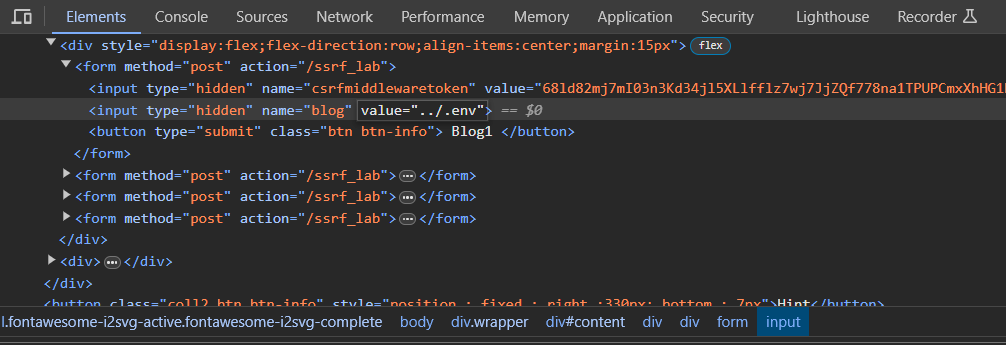


Do đó, em sẽ thử các thư mục này để tìm file .env. Với cây thư mục như trên, đầu tiên em sẽ thử với thư mục hiện tại của mã nguồn ./.env, nhưng không đúng path:

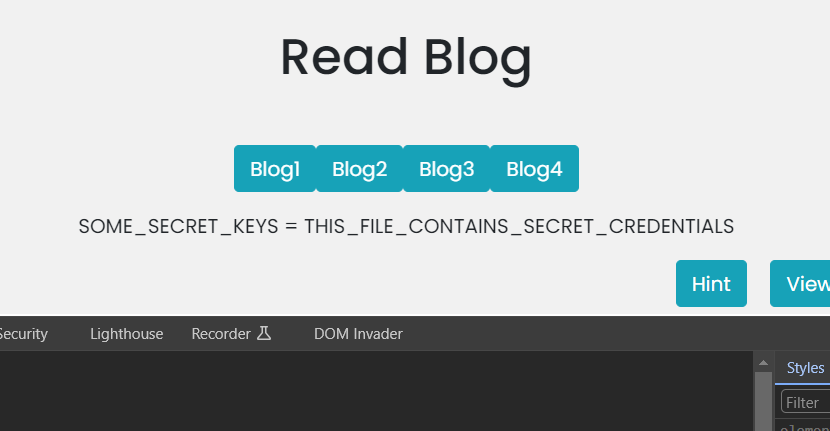




3. bước 3: Em sẽ tiến thêm một thư mục - kiểm tra trong thư mục cha của thư mục hiện tại ../.env:



Kết quả:



### Tài liệu hỗ trợ và tham khảo: <https://viblo.asia/p/tong-quan-ve-top-ten-owasp-2021-6J3ZgR7BKmB>

<https://viblo.asia/p/ssrf-la-gi-cach-phat-hien-va-ngan-chan-tan-cong-yeu-cau-gia-mao-tu-phia-may-chu-Eb85op08K2G>

**#Mức độ ảnh hưởng của lỗ hổng**: Tác động bảo mật nào mà kẻ tấn công có thể đạt được?

* Đọc tài nguyên mà công chúng không thể truy cập, chẳng hạn như trace.axd trong ASP.NET hoặc siêu dữ liệu API trong môi trường AWS.
* Quét mạng nội bộ mà máy chủ được kết nối đến.
* Đọc tệp từ máy chủ web.
* Xem trang trạng thái và tương tác với các API như máy chủ web.
* Truy xuất thông tin nhạy cảm như địa chỉ IP của máy chủ web sau proxy ngược.

**#Khuyến cáo khắc phục**: Làm thế nào để vá lỗ hổng này?

* Đảm bảo kiểm tra thông tin phản hồi cho người dùng: Thay vì trực tiếp trả về thông tin từ yêu cầu của người dùng, chúng ta cần thêm các bước kiểm tra tính hợp lệ của thông tin, nguồn gốc và nội dung trước khi phản hồi cho người dùng.
* Thống nhất các thông báo lỗi: Hạn chế khả năng tấn công bằng cách đồng nhất các thông báo lỗi, ngăn chặn việc kẻ tấn công sử dụng sự khác biệt giữa các thông báo lỗi để khai thác thông tin.
* Sử dụng một số biện pháp phòng chống SSRF kết hợp: Bao gồm các biện pháp dựa trên blacklist, whitelist, chặn các địa chỉ IP có dấu hiệu bất thường,...
* Ngăn chặn việc sử dụng các wrapper không cần thiết: Bao gồm các wrapper như file://, gopher://, ftp://, v.v., để ngăn chặn khả năng tấn công SSRF.

---

***Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này***

# **YÊU CẦU CHUNG**

* Sinh viên tìm hiểu và thực hành theo hướng dẫn.
* Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
* Sinh viên báo cáo kết quả thực hiện và nộp bài.

**Báo cáo:**

* File .DOCX và .PDF. Tập trung vào nội dung, không mô tả lý thuyết.
* Nội dung trình bày bằng Font chữ Times New Romans/ hoặc font chữ của mẫu báo cáo này (UTM Neo Sans Intel/UTM Viet Sach)– cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.
* Đặt tên theo định dạng: [Mã lớp]-SessionX\_GroupY. (trong đó X là Thứ tự buổi Thực hành, Y là số thứ tự Nhóm Thực hành đã đăng ký với GVHD-TH).

*Ví dụ: [*NT101.K11.ATCL*]-Session1\_Group3.*

* Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
* Không đặt tên đúng định dạng – yêu cầu, sẽ **KHÔNG** chấm điểm bài Lab.
* Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

**Đánh giá**: Sinh viên hiểu và tự thực hiện được bài thực hành. Khuyến khích:

* Chuẩn bị tốt.
* Có nội dung mở rộng, ứng dụng trong kịch bản phức tạp hơn, có đóng góp xây dựng bài thực hành.

*Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.*

**HẾT**

1. Ghi nội dung công việc, các kịch bản trong bài Thực hành [↑](#footnote-ref-1)