

# **File path traversal, traversal sequences blocked with absolute path bypass –**

## **Lab Report**

---

Submitted By:

**Name:** Arsalan Khan

**Position/Role:** Internee

**Date:** July 25, 2025

---

### **Platform:**

PortSwigger

### **Objective:**

Bypass path traversal filtering and retrieve the contents of the `/etc/passwd` file by supplying an absolute path instead of using traversal sequences.

### **Tools Used:**

- Burp Suite Community

## 1. Access the Lab

- Clicked on one of the product images to trigger an image request.

The screenshot shows the Burp Suite interface on the left and a web browser on the right. The Burp Suite 'HTTP history' tab is active, displaying a list of intercepted requests. The browser shows the 'Web Security Academy' lab page with a 'File path traversal, traversal sequences blocked with absolute path bypass' challenge. A congratulatory message 'Congratulations, you solved the lab!' is visible.

Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Ti
https://0aab002303b22c96...	GET	/			200	10442	HTML		Fi
https://0aab002303b22c96...	GET	/resources/images/shop.svg			200	7258	XML	svg	
https://0aab002303b22c96...	GET	/resources/labheader/js/labHead...			200	1673	script	js	
https://0aab002303b22c96...	GET	/resources/labheader/images/log...			200	8852	XML	svg	
https://0aab002303b22c96...	GET	/resources/labheader/images/ps-l...			200	942	XML	svg	
https://0aab002303b22c96...	GET	/academyLabHeader			101	147			
https://0aab002303b22c96...	GET	/resources/labheader/images/ps-l...			200	707	XML	svg	
https://0af00020464755a8...	GET	/			200	11016	HTML		In
https://0af00020464755a8...	GET	/resources/labheader/js/labHead...			200	1673	script	js	
https://0af00020464755a8...	GET	/resources/images/shop.svg			200	1333	script	js	
https://0af00020464755a8...	GET	/resources/labheader/images/log...			200	7258	XML	svg	
https://0af00020464755a8...	GET	/resources/labheader/images/ps-l...			200	8852	XML	svg	
https://0af00020464755a8...	GET	/resources/labheader/images/ps-l...			200	942	XML	svg	
https://0af00020464755a8...	GET	/academyLabHeader			400	130	text		
https://0af00020464755a8...	GET	/			200	10929	HTML		In
https://0af00020464755a8...	GET	/academyLabHeader			400	130	text		
https://0af00020464755a8...	GET	/			200	10929	HTML		In
https://0af00020464755a8...	GET	/academyLabHeader			400	130	text		
https://0af00020464755a8...	GET	/			200	10929	HTML		In
https://0af00020464755a8...	GET	/academyLabHeader			101	147			
https://0af00020464755a8...	GET	/product?productId=1			200	4337	HTML		In
https://0af00020464755a8...	GET	/academyLabHeader			400	130	text		
https://0af00020464755a8...	POST	/submitSolution			200	100	JSON		
https://0af00020464755a8...	GET	/product?productId=1			200	7246	HTML		In
https://0af00020464755a8...	GET	/resources/labheader/js/complete...			200	175	script	js	
https://0af00020464755a8...	GET	/resources/labheader/images/ps-l...			200	707	XML	svg	
https://0af00020464755a8...	GET	/academyLabHeader			400	130	text		
https://go.portswigger.net	GET	/p.js			200	6004	script	js	
https://no.mozilla.net	GET	/sanitizer?src=3&sanitizeId=ACB7			200	1301	script	js	

## 2. Intercept the Image Request.

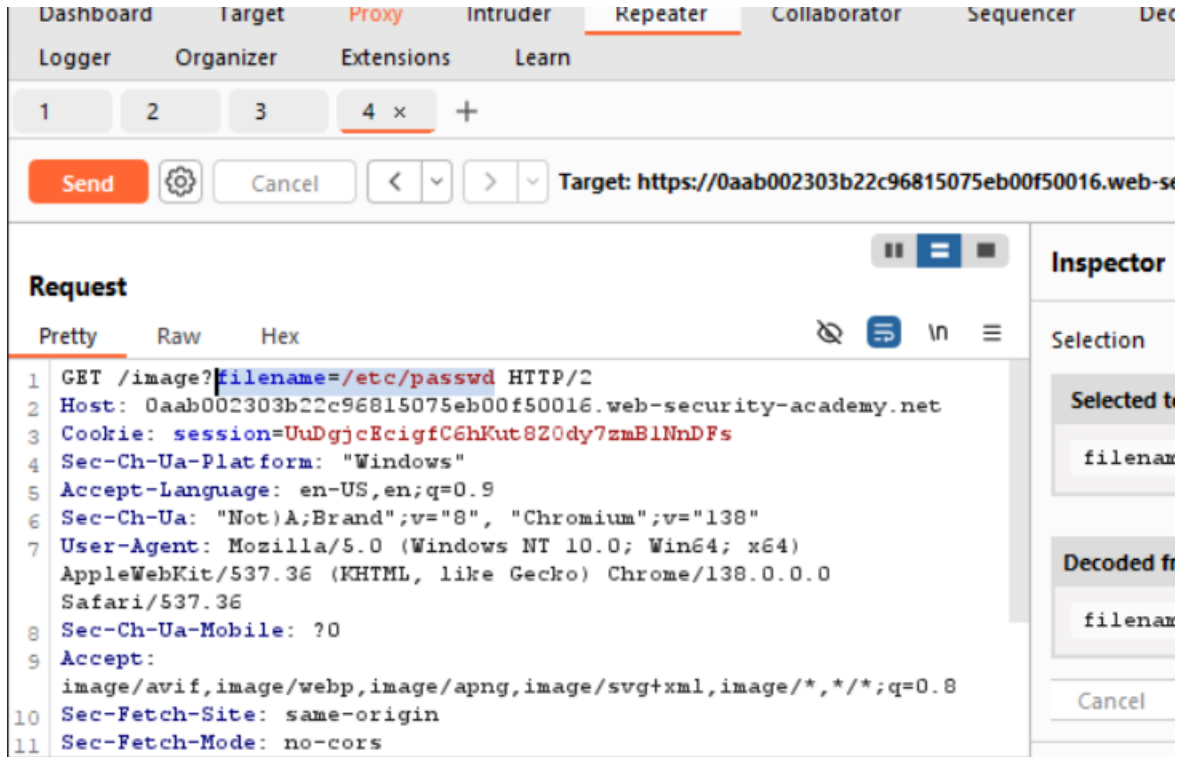
- Enabled Intercept and clicked a product image to capture the request.

The screenshot shows the Burp Suite 'HTTP history' tab with a list of intercepted requests. The 'Request' tab is selected, showing the raw request for an image file. The 'Inspector' tab is also visible.

Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension
https://0aab002303b22c96...	GET	/resources/labheader/images/ps-l...			200	707	XML	svg
https://0aab002303b22c96...	GET	/			200	13499	HTML	
https://0aab002303b22c96...	GET	/resources/labheader/js/complete...			200	175	script	js
https://0aab002303b22c96...	GET	/image?filename=36.jpg						
https://0aab002303b22c96...	GET	/image?filename=44.jpg						
https://0aab002303b22c96...	GET	/image?filename=4.jpg						
https://0aab002303b22c96...	GET	/image?filename=74.jpg						
https://0aab002303b22c96...	GET	/image?filename=54.jpg						
https://0aab002303b22c96...	GET	/image?filename=17.jpg						
https://0aab002303b22c96...	GET	/image?filename=51.jpg						
https://0aab002303b22c96...	GET	/image?filename=9.jpg						
https://0aab002303b22c96...	GET	/image?filename=20.jpg						
https://0aab002303b22c96...	GET	/image?filename=58.jpg						
https://0aab002303b22c96...	GET	/image?filename=52.jpg						

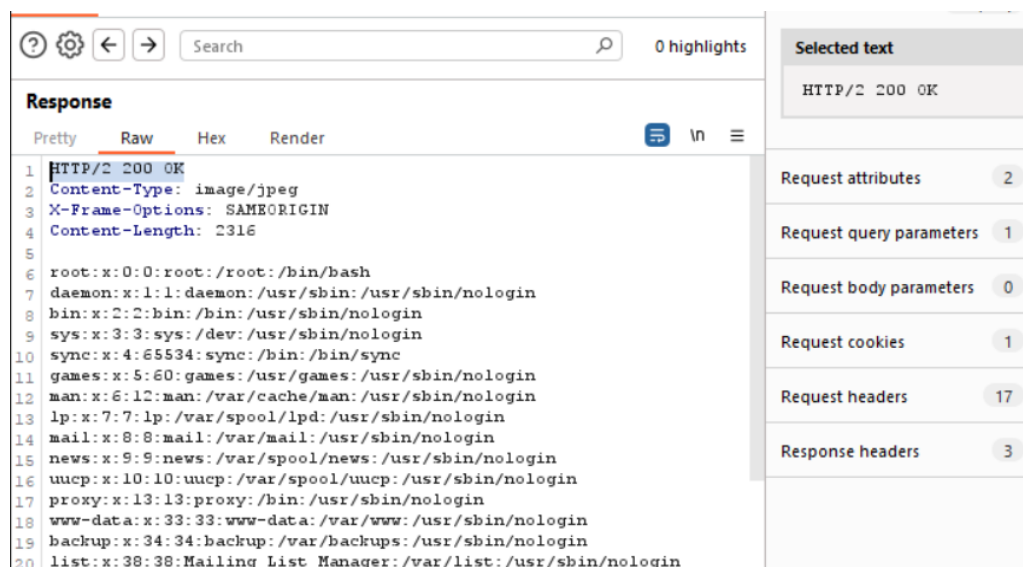
### 3. Modify the Request with Absolute Path

- Sent the intercepted request to Burp Repeater.
- Replaced the filename parameter with an absolute path to the /etc/passwd file:



### 4. Observe the Server Response

- The server responded with the contents of the /etc/passwd file.
- Confirmed that path traversal filtering was bypassed using an absolute path.



## 5. Submit the Solution



File path traversal, traversal sequences blocked with absolute path bypass

LAB

Solved



[Back to lab description >>](#)

## Vulnerability Analysis:

- **Issue:** The application attempts to block traversal sequences like `../`, but it fails to account for the use of absolute paths.
- **Exploit Method:** Supplying `/etc/passwd` directly as the filename bypasses the filter.
- **Impact:** Unauthorized access to sensitive files on the server's filesystem.
- **Risk Level:** High

## Mitigation Recommendations:

- Normalize all user input paths (resolve symlinks, absolute paths, etc.) before access.
- Enforce strict allow-lists of permitted filenames and directories.
- Disallow absolute paths and reject any inputs containing `/`, `\`, or starting with `/`.
- Run the application in a sandbox or chroot jail to isolate it from sensitive system files.

## Conclusion:

This lab demonstrated a bypass of path traversal protections by using an absolute file path instead of traversal sequences. The `/etc/passwd` file was accessed directly, proving that the application failed to properly validate or sanitize the filename input.

End...