Lab 18: Path traversal vulnerabilities

Objective:

- This lab is focused on exploring Path Traversal vulnerabilities, a type of security flaw where an attacker can access and potentially manipulate files on a server that are not intended to be accessible through the web application. Participants will interact with a simulated environment featuring a web application with path traversal vulnerabilities. The objective is to understand how these vulnerabilities are exploited, their potential impact on system security, and the strategies for mitigating them.

In this lab, students need to:

- Answer the following questions:
  - What is a Path Traversal vulnerability, and how does it differ from other types of web application vulnerabilities? Explain the concept of path traversal, including how attackers can manipulate input to access or manipulate files outside of the intended directory.
  - Describe the process for exploiting a Path Traversal vulnerability in a web application. What are the common methods or techniques used by attackers to manipulate file paths, and how can these methods compromise the security of a server?
- Perform challenge:
  - File path traversal, traversal sequences blocked with absolute path bypass
- Explain and capture all steps (full windows screen capture).

Submit a report addressing all the questions mentioned above in either **PDF** or **Markdown** format. Additionally, include a **video** demonstrating the detailed process of your work to ensure the authenticity of your lab exercise.

Lab 18: Path traversal vulnerabilities

What is a Path Traversal vulnerability, and how does it differ from other types of web application vulnerabilities? Explain the concept of path traversal, including how attackers can manipulate input to access or manipulate files outside of the intended directory.

A Path Traversal vulnerability, also known as Directory Traversal or File Path Manipulation, is a type of security flaw that allows an attacker to access files or directories on a web server that are outside the intended directory or scope. This vulnerability arises when an application improperly sanitizes or validates user-supplied input used to construct file paths, allowing attackers to manipulate the input to traverse directories and access sensitive files or resources.

Here's how Path Traversal differs from other types of web application vulnerabilities:

1. **Cross-Site Scripting (XSS)**: Cross-Site Scripting involves injecting malicious scripts into web pages, which are then executed within the context of the victim's browser. In contrast, Path Traversal vulnerabilities focus on manipulating file paths on the server-side to access files or directories that are not intended to be accessible.

2. **SQL Injection (SQLi)**: SQL Injection attacks exploit vulnerabilities in web applications that use SQL databases by injecting malicious SQL queries. This can lead to unauthorized access to the database, data leakage, or data manipulation. Path Traversal, on the other hand, involves accessing files or directories on the server's file system rather than directly interacting with the database.

3. **Server-Side Request Forgery (SSRF)**: Server-Side Request Forgery vulnerabilities allow attackers to manipulate the server into making unintended requests to internal or external resources. While SSRF may involve accessing internal resources, Path Traversal specifically focuses on accessing files or directories on the server's file system.

Path Traversal occurs when an application does not properly validate or sanitize user-supplied input used to construct file paths, such as file names, directory names, or URL parameters. Attackers can exploit this vulnerability by manipulating the input to traverse directories and access files outside of the intended directory structure.

Here's how attackers can manipulate input to exploit a Path Traversal vulnerability:

1. **Traversal Sequences**: Attackers may use traversal sequences, such as "../" (dot-dot-slash), to navigate up the directory tree from the current directory and access files or directories in higher-level directories.

2. **Absolute Paths**: Attackers may provide absolute paths to access files or directories outside of the application's intended directory structure. For example, "/etc/passwd" on Unix-based systems or "C:\Windows\System32\config" on Windows systems.

3. **URL Encoding**: Attackers may use URL encoding or other encoding techniques to obfuscate traversal sequences and evade input validation or filtering mechanisms implemented by the application.

4. **Null Byte Injection**: Attackers may exploit vulnerabilities in the application's handling of null bytes (\x00) to truncate file paths and bypass input validation or filtering.

Once exploited, Path Traversal vulnerabilities can lead to unauthorized access to sensitive files or directories, data leakage, privilege escalation, or remote code execution, depending on the permissions and capabilities of the compromised application and server. To mitigate Path Traversal vulnerabilities, developers should implement proper input validation and sanitization techniques, enforce access controls, and avoid directly exposing file system paths to user-supplied input.

Describe the process for exploiting a Path Traversal vulnerability in a web application. What are the common methods or techniques used by attackers to manipulate file paths, and how can these methods compromise the security of a server?

Exploiting a Path Traversal vulnerability in a web application involves manipulating user-controlled input, typically file paths, to access files or directories outside of the intended scope. Here's a general process for exploiting a Path Traversal vulnerability:

1. **Identify the Vulnerability**: The attacker first needs to identify a vulnerable input field or parameter in the web application that is susceptible to Path Traversal. This could include parameters in URL paths, file upload forms, or any other input mechanism where file paths are used.

2. **Understand the File System**: The attacker needs to understand the file system structure of the server hosting the web application. This includes knowing the directory structure, file naming conventions, and permissions of files and directories.

3. **Craft Traversal Sequences**: The attacker crafts traversal sequences, such as "../" (dot-dot-slash), to navigate up the directory tree from the current directory and access files or directories outside of the application's intended scope. The number of traversal sequences needed depends on the depth of the directory traversal required to reach the target file or directory.

4. **Inject Traversal Sequences**: The attacker injects the crafted traversal sequences into the vulnerable input field or parameter. This could be done by modifying URL parameters, form fields, or any other input mechanism that accepts file paths.

5. **Submit the Request**: The attacker submits the manipulated request containing the injected traversal sequences to the web application server.

6. **Access Files or Directories**: If the Path Traversal vulnerability is successfully exploited, the server will process the manipulated file path and return the contents of the accessed file or directory to the attacker.

Common methods or techniques used by attackers to manipulate file paths and exploit Path Traversal vulnerabilities include:

- **Traversal Sequences**: Injecting traversal sequences such as "../" to navigate up the directory tree from the current directory and access files or directories outside of the intended scope.

- **Absolute Paths**: Providing absolute paths to access files or directories directly without navigating through the directory structure. For example, "/etc/passwd" on Unix-based systems or "C:\Windows\System32\config" on Windows systems.

- **URL Encoding**: Using URL encoding or other encoding techniques to obfuscate traversal sequences and evade input validation or filtering mechanisms implemented by the application.

- **Null Byte Injection**: Exploiting vulnerabilities in the application's handling of null bytes (\x00) to truncate file paths and bypass input validation or filtering.

Exploiting Path Traversal vulnerabilities can compromise the security of a server in several ways:

- **Unauthorized Access**: Attackers can access sensitive files containing confidential information such as passwords, configuration files, or databases.

- **Data Leakage**: Attackers can read or download files containing sensitive data, leading to data leakage or exposure.

- **File Manipulation**: Attackers can modify or delete critical files, leading to system instability or compromise.

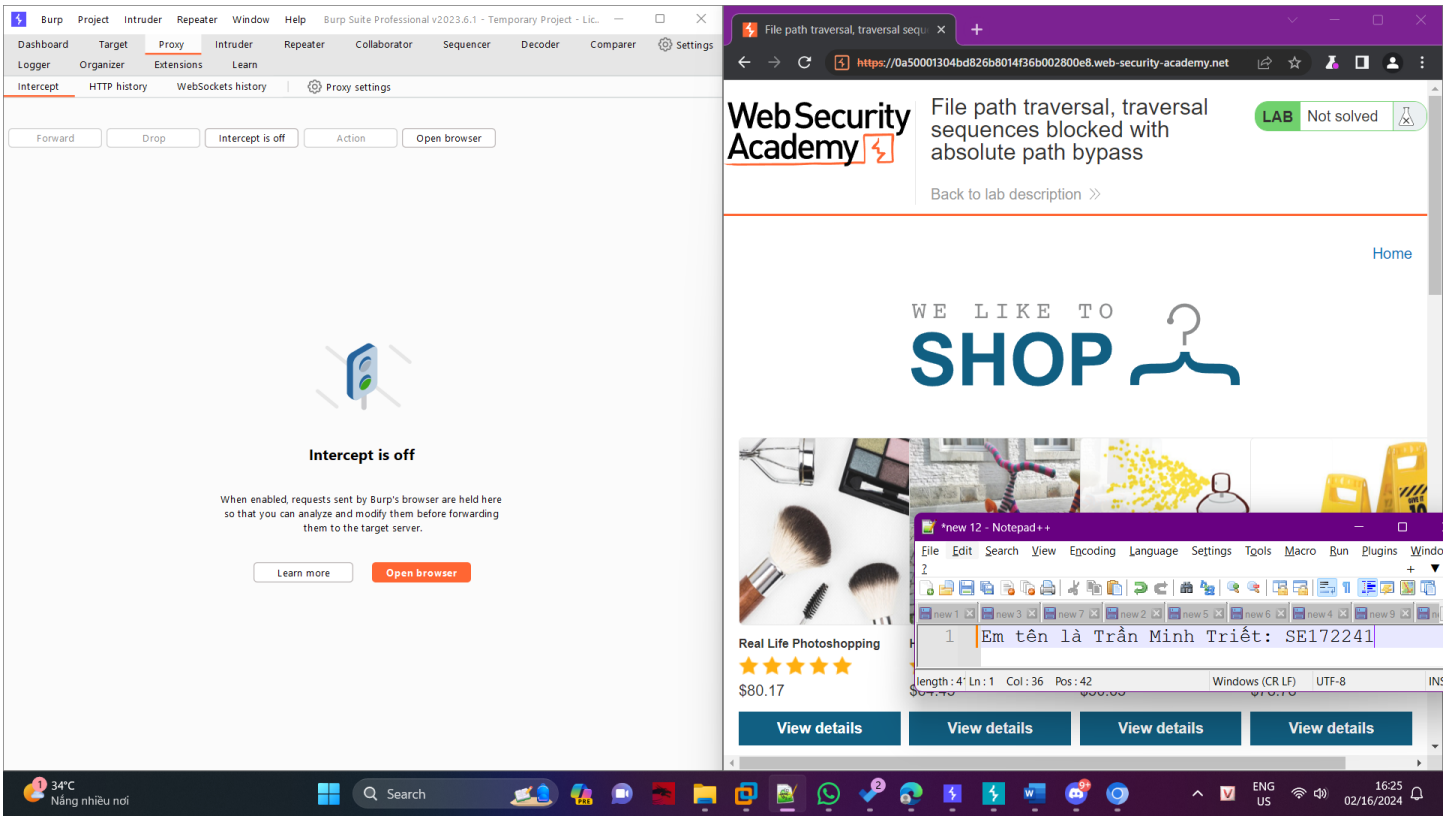
- **Privilege Escalation**: Exploiting Path Traversal vulnerabilities may allow attackers to escalate privileges and gain administrative access to the server.

- **Remote Code Execution**: In some cases, attackers may leverage Path Traversal vulnerabilities to execute arbitrary code on the server, leading to complete compromise of the system.

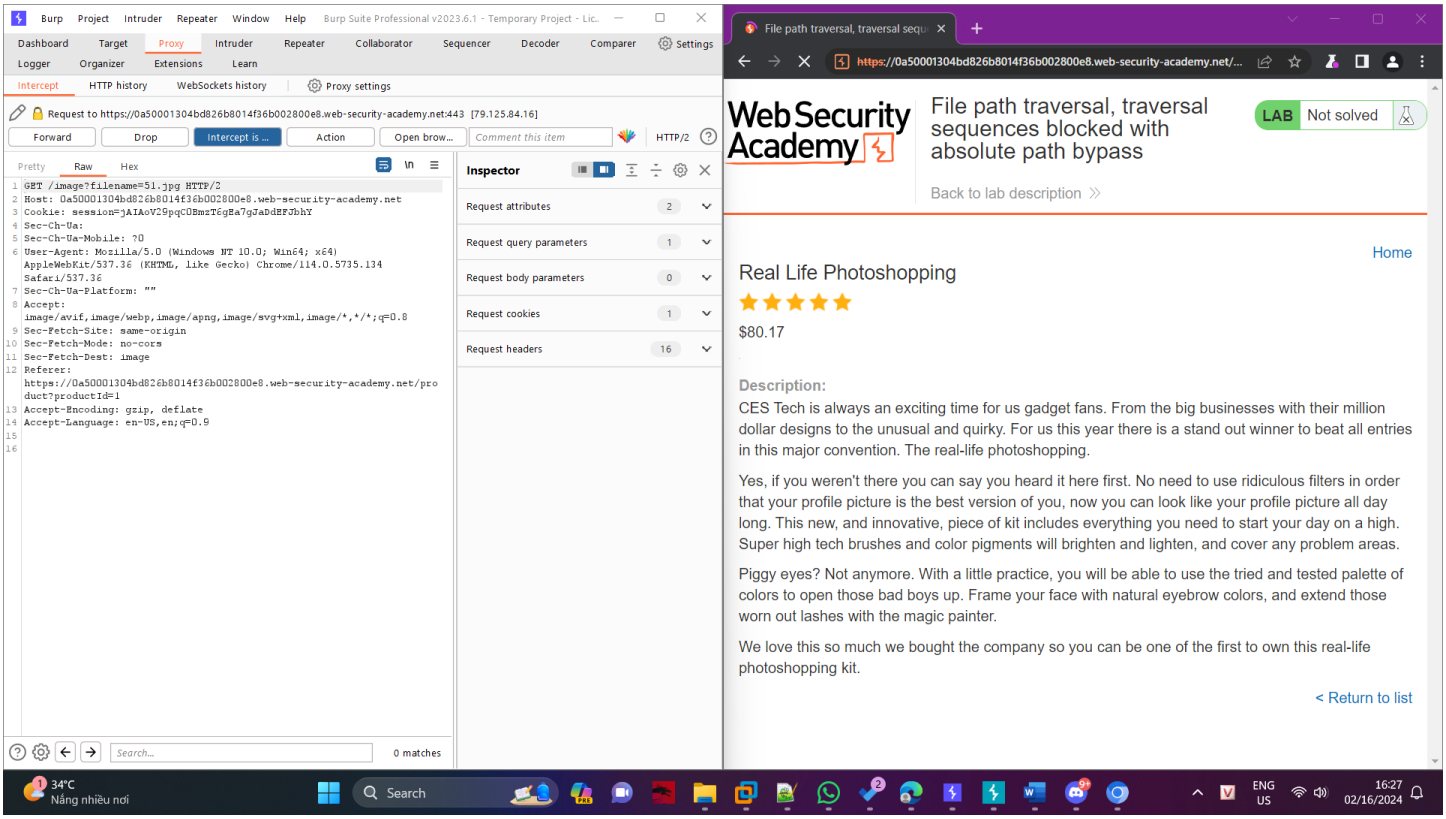
To mitigate Path Traversal vulnerabilities, developers should implement proper input validation and sanitization techniques, enforce access controls, and avoid directly exposing file system paths to user-

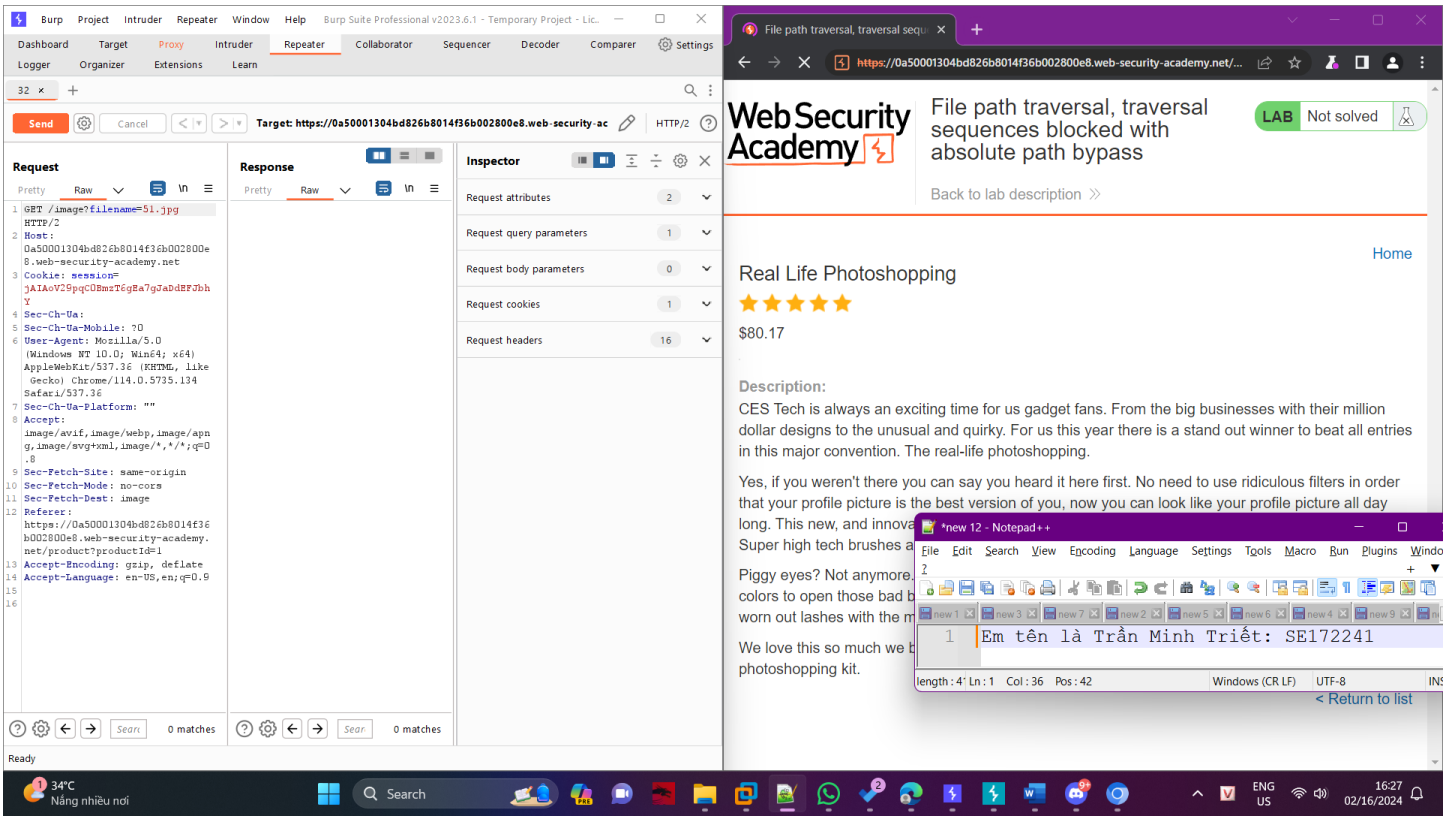
supplied input. Additionally, web application firewalls (WAFs) and security scanners can help detect and prevent Path Traversal attacks.

Challenge

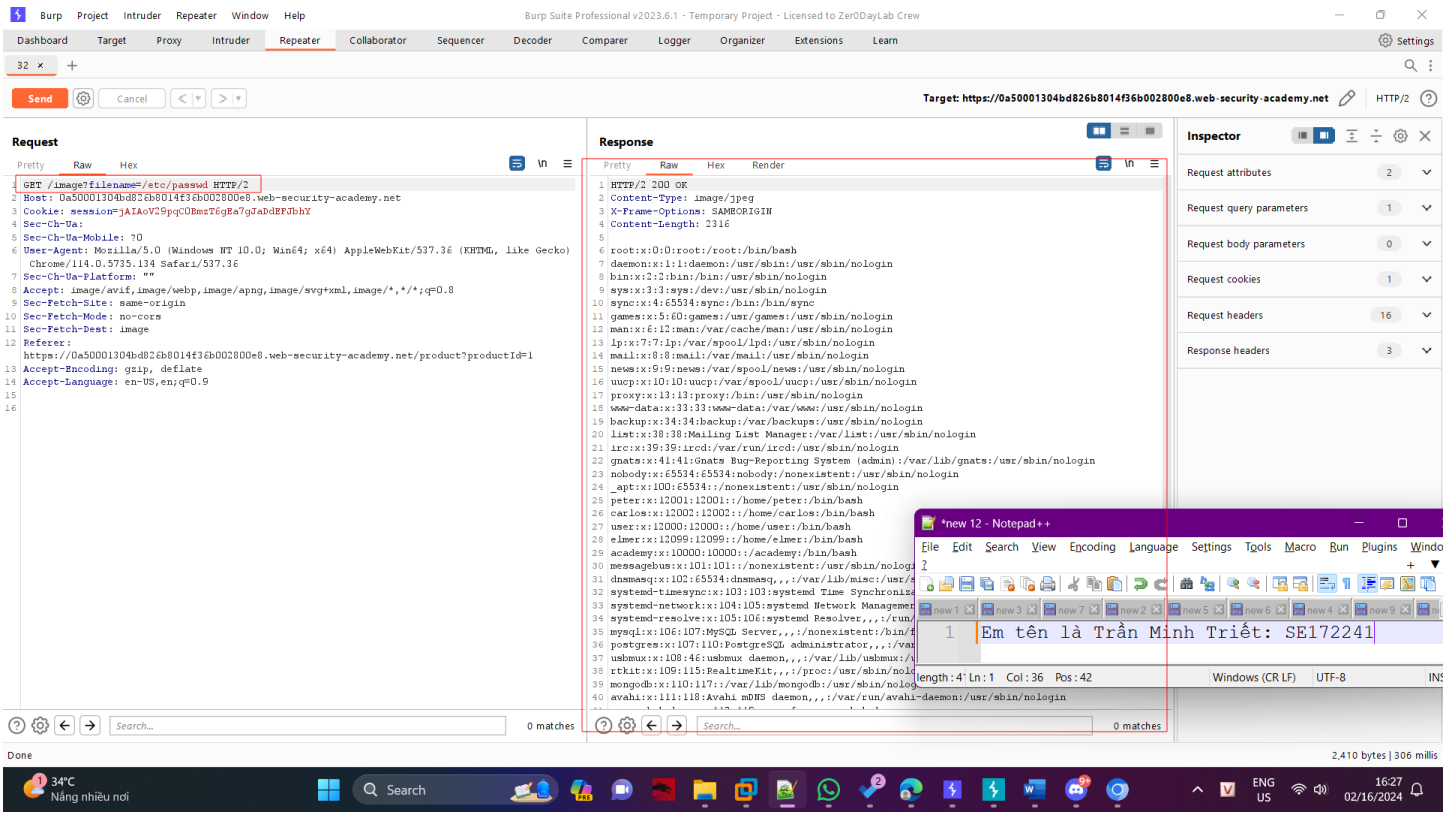


Use Burp Suite to intercept and modify a request that fetches a product image.

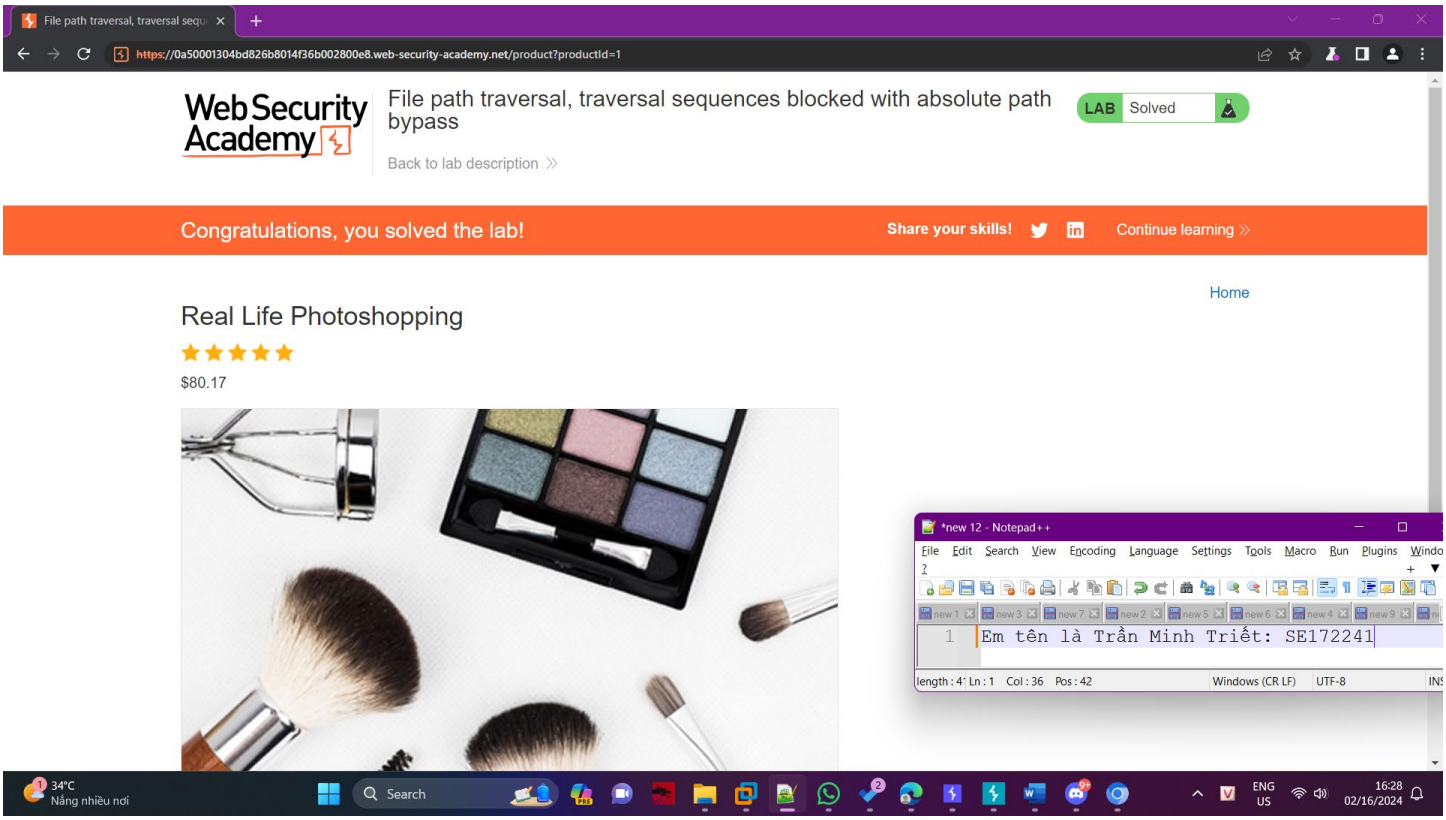




Modify the filename parameter, giving it the value /etc/passwd.



Observe that the response contains the contents of the /etc/passwd file.


File path traversal, traversal sequ
https://0a50001304bd826b8014f36b002800e8.web-security-academy.net/product?productId=1
Web Security Academy
File path traversal, traversal sequences blocked with absolute path bypass
LAB Solved
Back to lab description
Congratulations, you solved the lab!
Share your skills!
Continue learning
Home
Real Life Photoshopping
$80.17
*new 12 - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Windo
Em tên là Trần Minh Triết: SE172241
length : 41 Ln : 1 Col : 36 Pos : 42 Windows (CR LF) UTF-8
34°C
Nắng nhiều nơi
Search
ENG
US
16:28
02/16/2024