Penetration Testing Report

Full Name: Aurelia Anthony
Program: HCPT
Date: 11/02/2025

Introduction

This report describes the proceedings and results of a Black Box security assessment conducted against the **Week 1 Labs**. The report hereby lists the findings and corresponding best practice mitigation actions and recommendations.

1. Objective

The objective of the assessment was to uncover vulnerabilities in the **Week 1 Labs** and provide a final security assessment report comprising vulnerabilities, remediation strategy and recommendation guidelines to help mitigate the identified vulnerabilities and risks during the activity.

2. Scope

This section defines the scope and boundaries of the project.

Application Name	**HTML and Cross site scripting Injections**

3. Summary

Outlined is a Black Box Application Security assessment for Week **1 Labs**.

Total number of Sub-labs: 17 Sub-labs

High	Medium	Low
Two(2)	Twelve(12)	Three(3)

High - Number of Sub-labs with hard difficulty level
Medium - Number of Sub-labs with Medium difficulty level
Low - Number of Sub-labs with Easy difficulty level

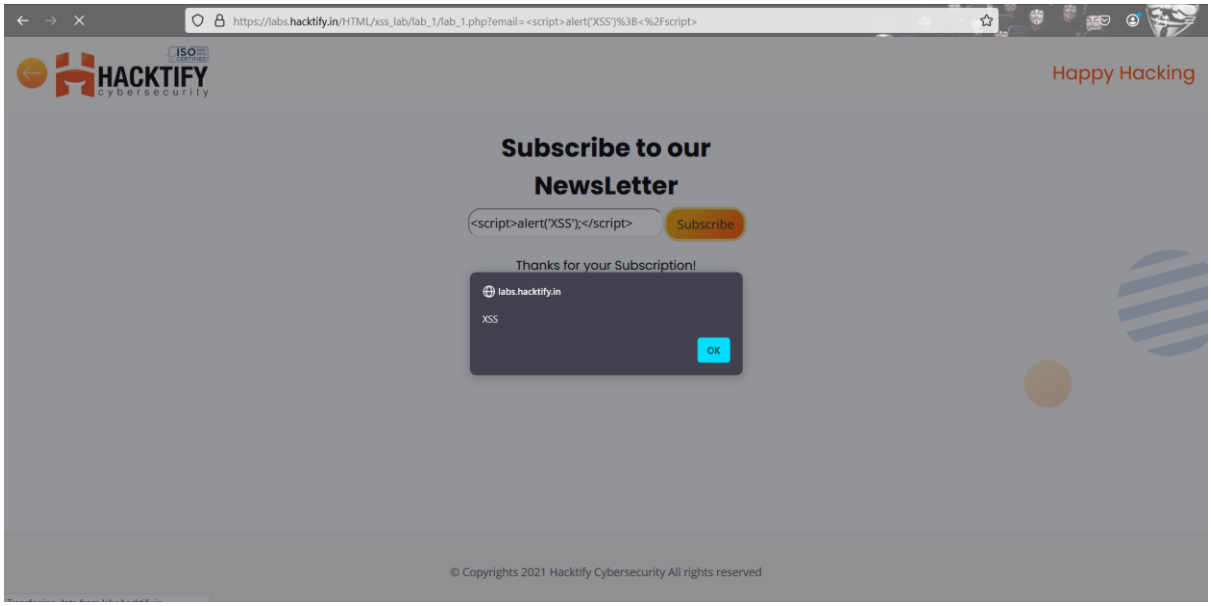
1. Cross site Scripting

1.1. Let's do it!

Reference	Risk Rating
Let's do it!	**Low**
Tools Used	
XSS payloads using javascript	
Vulnerability Description	
Cross-Site Scripting (XSS) is a type of security vulnerability that allows attackers to inject malicious scripts (usually JavaScript) into web pages viewed by other users	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_1/lab_1.php?email=%3Cscript%3Ealert%28%27XSS%27%29%3B%3C%2Fscript%3E	
Consequences of not Fixing the Issue	
Validate and sanitize all user inputs.	
Suggested Countermeasures	
XSS vulnerabilities can allow the manipulation of content on a web page, compromising the integrity of the application.	
References	
https://owasp.org/www-community/attacks/xss/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

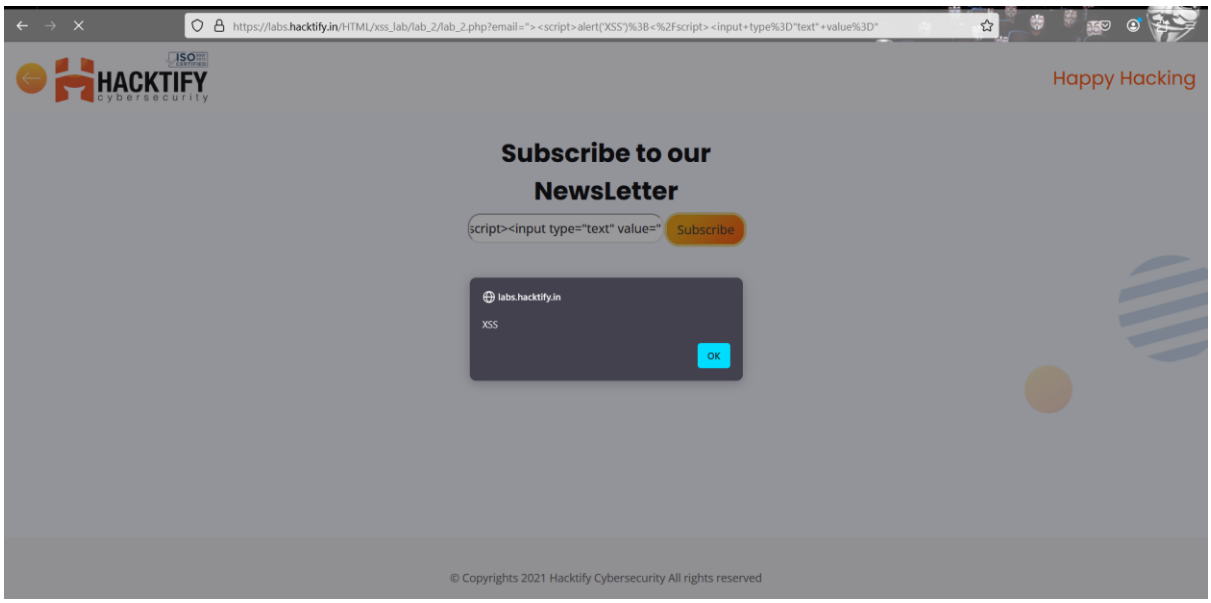


1.2. Balancing is important in life

Reference	Risk Rating
Balancing is important in life	**Medium**
Tools Used	
XSS payloads using javascript	
Vulnerability Description	
Balancing a payload for an XSS injectiozn attack refers to crafting the payload in such a way that it escapes the existing JavaScript context in the application, allowing the injected code to execute correctly.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_2/lab_2.php?email=%22%3E%3Cscript%3Ealert%28%27XSS%27%29%3B%3C%2Fscript%3E%3Cinput+type%3D%22text%22+value%3D%22	
Consequences of not Fixing the Issue	
Attackers can inject malicious scripts that download malware onto users' devices.	
Suggested Countermeasures	
Always escape dynamic content inserted into JavaScript to prevent interpretation as code.	
References	
https://owasp.org/www-community/attacks/xss/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

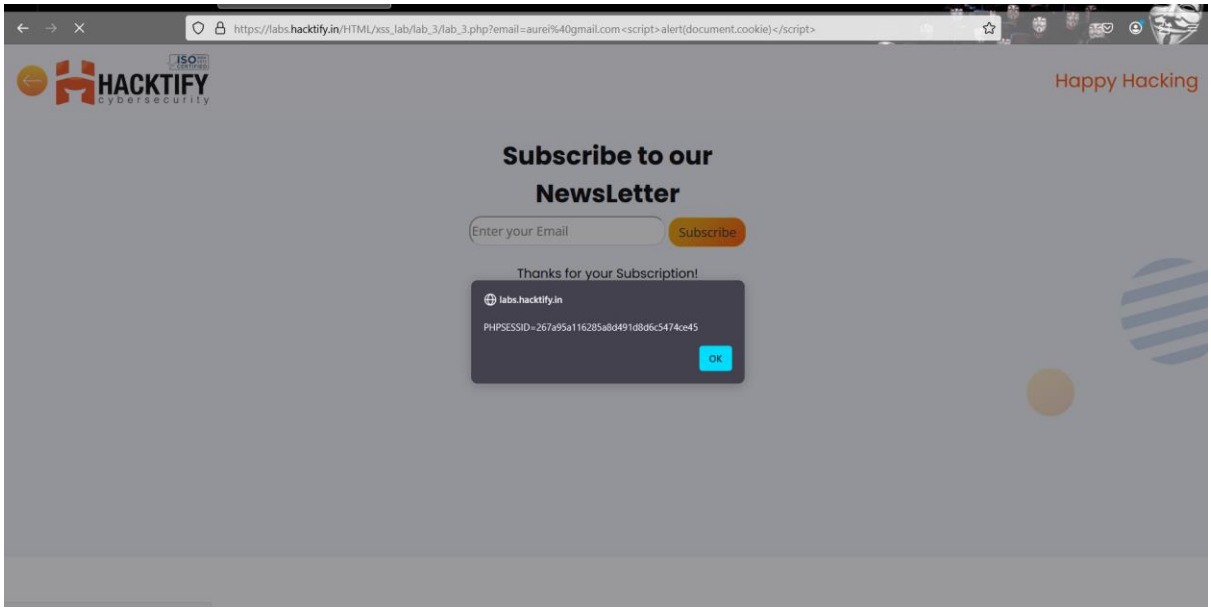


1.3. XSS is everywhere

Reference	Risk Rating
XSS is everywhere	**Medium**
Tools Used	
XSS payloads using javascript	
Vulnerability Description	
Reflected XSS attacks are those where the injected script is reflected off the web server, such as in an error message, search result, or any other response that includes some or all of the input sent to the server as part of the request.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_3/lab_3.php?email=aurei%40gmail.com%3Cscript%3Ealert(document.cookie)%3C/script%3E	
Consequences of not Fixing the Issue	
By stealing session cookies, attackers can impersonate users and gain unauthorized access to their accounts.	
Suggested Countermeasures	
Deploy a Web Application Firewall to detect and block XSS payloads in real-time.	
References	
https://owasp.org/www-community/attacks/xss/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

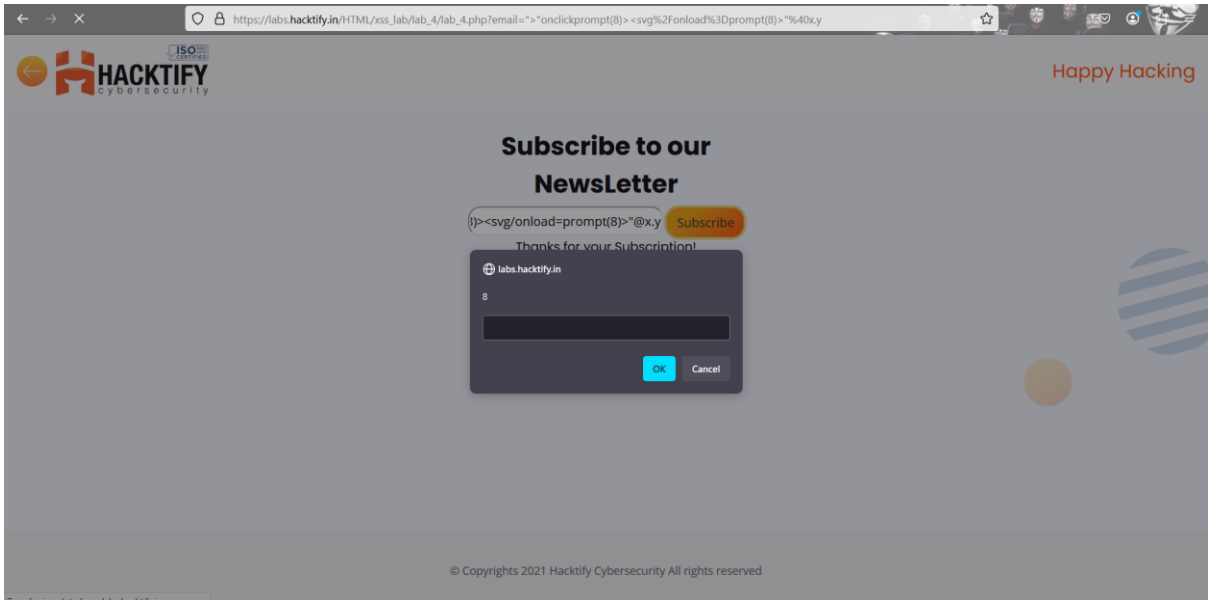


1.4. Alternatives are a must!

Reference	Risk Rating
Alternatives are a must!	Medium
Tools Used	
XSS payloads using javascript	
Vulnerability Description	
Reflected XSS attacks are those where the injected script is reflected off the web server, such as in an error message, search result, or any other response that includes some or all of the input sent to the server as part of the request.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_4/lab_4.php?email=%22%3E%22onclickprompt%288%29%3E%3Csvg%2Fonload%3Dprompt%288%29%3E%22%40x.y	
Consequences of not Fixing the Issue	
Attackers can modify the content of the website, leading to reputational damage.	
Suggested Countermeasures	
Use allowlists to accept only expected and safe input patterns.	
References	
https://owasp.org/www-community/attacks/xss/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

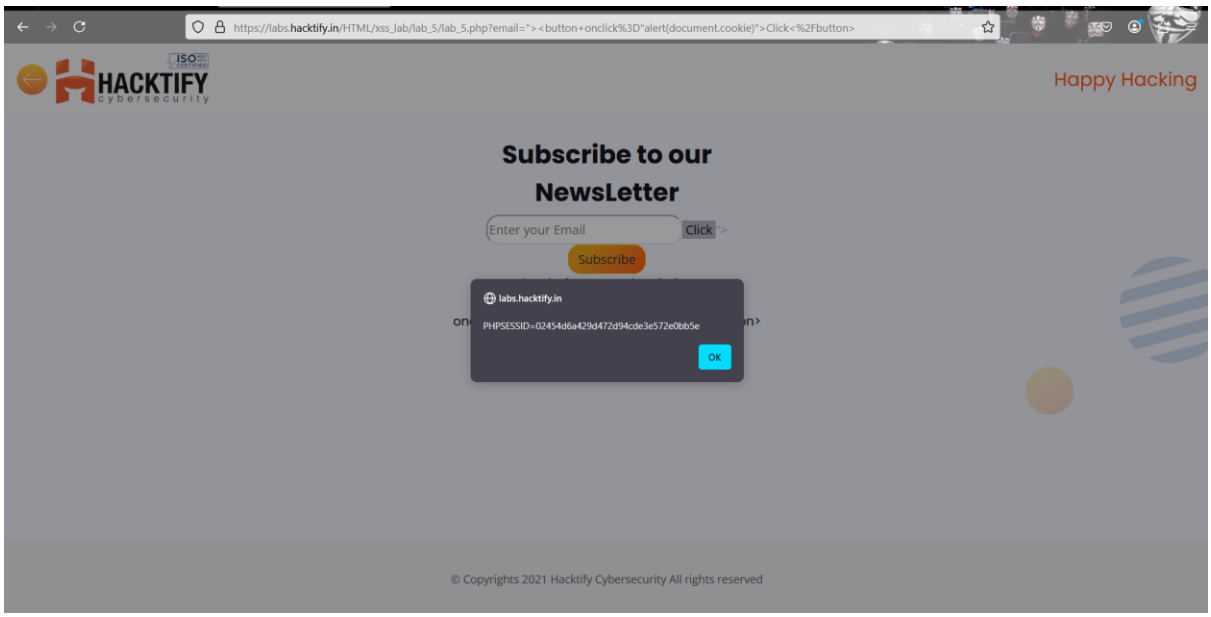


1.5. Developer hates scripts!

Reference	Risk Rating
Developer hates scripts!	Medium
Tools Used	
XSS payloads using javascript and burp suite for encoding	
Vulnerability Description	
Reflected attacks are those where the injected script is reflected off the web server, such as in an error message, search result, or any other response that includes some or all of the input sent to the server as part of the request.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_5/lab_5.php?email=%22%3E%3Cbutton+onclick%3D%22alert%28document.cookie%29%22%3EClick%3C%2Fbutton%3E	
Consequences of not Fixing the Issue	
Attackers can modify the content of the website, leading to reputational damage.	
Suggested Countermeasures	
Use context-specific encoding (e.g. JavaScript, URL) depending on where the data is being inserted.	
References	
https://owasp.org/www-community/attacks/xss/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

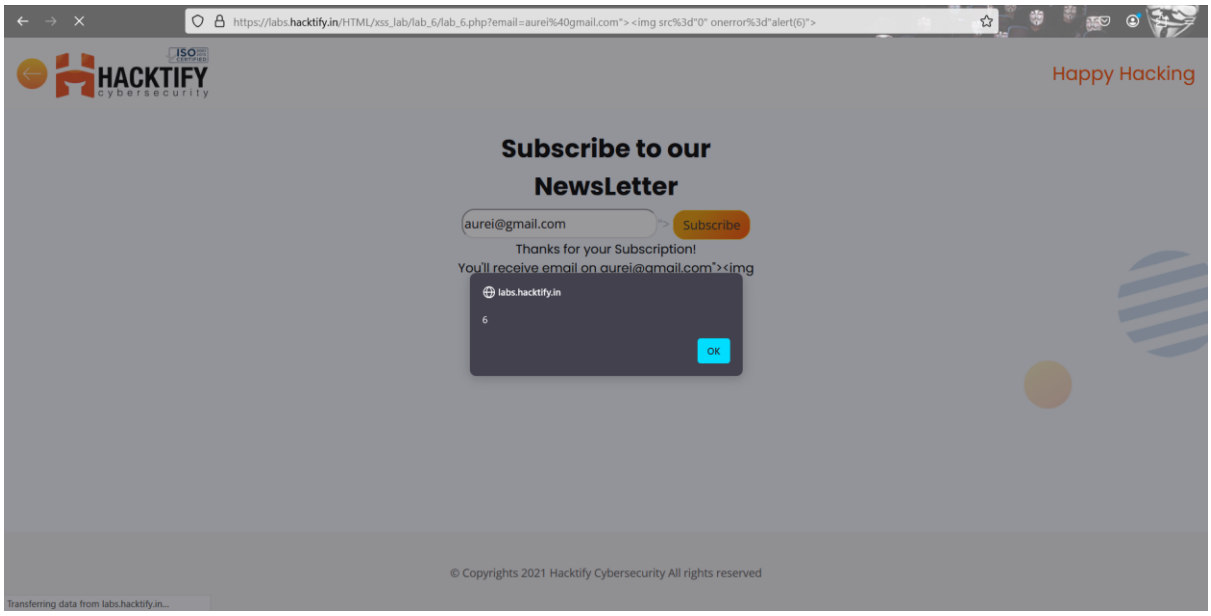


1.6. Change the variation!

Reference	Risk Rating
Change the variation!	**Medium**
Tools Used	
XSS payloads using javascript and burp suite for encoding	
Vulnerability Description	
Reflected XSS occurs when an attacker injects malicious scripts into a web application, which are then reflected back to the victim's browser and executed. This typically happens when user input is included in the server's response without proper validation or encoding.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_6/lab_6.php?email=aurei%40gmail.com%22%3E%3Cimg%20src%3d%220%22%20onerror%3d%22alert(6)%22%3E	
Consequences of not Fixing the Issue	
Attackers can inject scripts that redirect users to malicious websites or download malware onto their devices.	
Suggested Countermeasures	
Validate all user inputs on the server side to ensure they conform to expected formats (e.g., alphanumeric characters, email format, etc.).	
References	
https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability

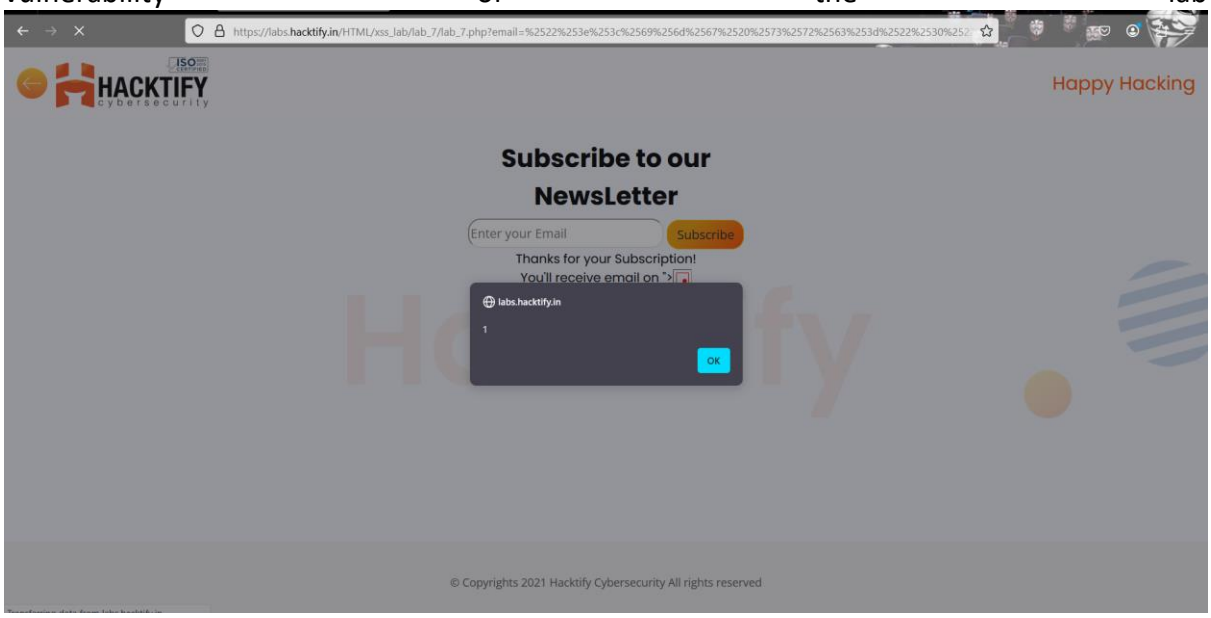


1.7. Encoding is the key?

Reference	Risk Rating
Encoding is the key?	**Medium**
Tools Used	
Xss encoded payload	
Vulnerability Description	
Reflected XSS occurs when an attacker injects malicious scripts into a web application, which are then reflected back to the victim's browser and executed. This typically happens when user input is included in the server's response without proper validation or encoding.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_7/lab_7.php?email=%2522%253e%253c%2569%256d%2567%2520%2573%2572%2563%253d%2522%2530%2522%2520%256f%256e%2565%2572%2572%256f%2572%253d%2522%2561%256c%2565%2572%2574%2528%2531%2529%2522%253e	
Consequences of not Fixing the Issue	
Attackers can inject scripts that redirect users to malicious websites or download malware onto their devices.	
Suggested Countermeasures	
Leverage frameworks and libraries that automatically handle XSS protection, such as React, Angular, or Django.	
References	
https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html#xss-prevention-rules-summary	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

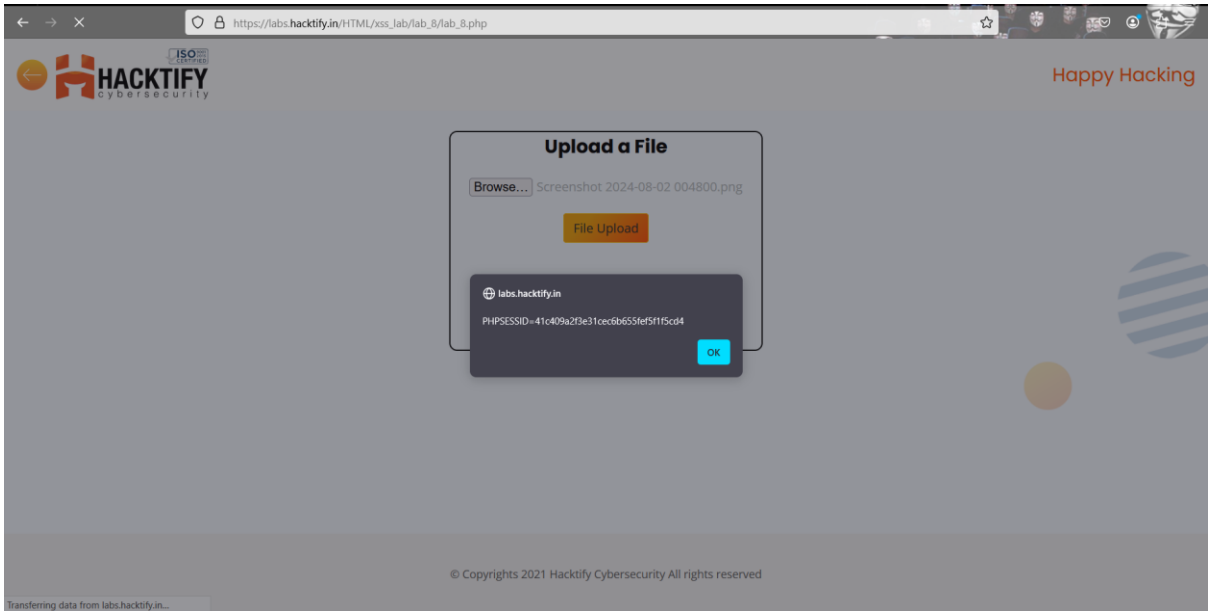


1.8. XSS with file upload (file name)

Reference	Risk Rating
XSS with file upload (file name)	Medium
Tools Used	
Xss encoded payload and burp suite	
Vulnerability Description	
Reflected XSS occurs when an attacker injects malicious scripts into a web application, which are then reflected back to the victim's browser and executed. This typically happens when user input is included in the server's response without proper validation or encoding.	
How It Was Discovered	
Automatic Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_8/lab_8.php	
Consequences of not Fixing the Issue	
Attackers can modify the content of the website, leading to reputational damage.	
Suggested Countermeasures	
Deploy a Web Application Firewall to detect and block XSS payloads in real-time.	
References	
URLs to the sources used to know more about this vulnerability	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability

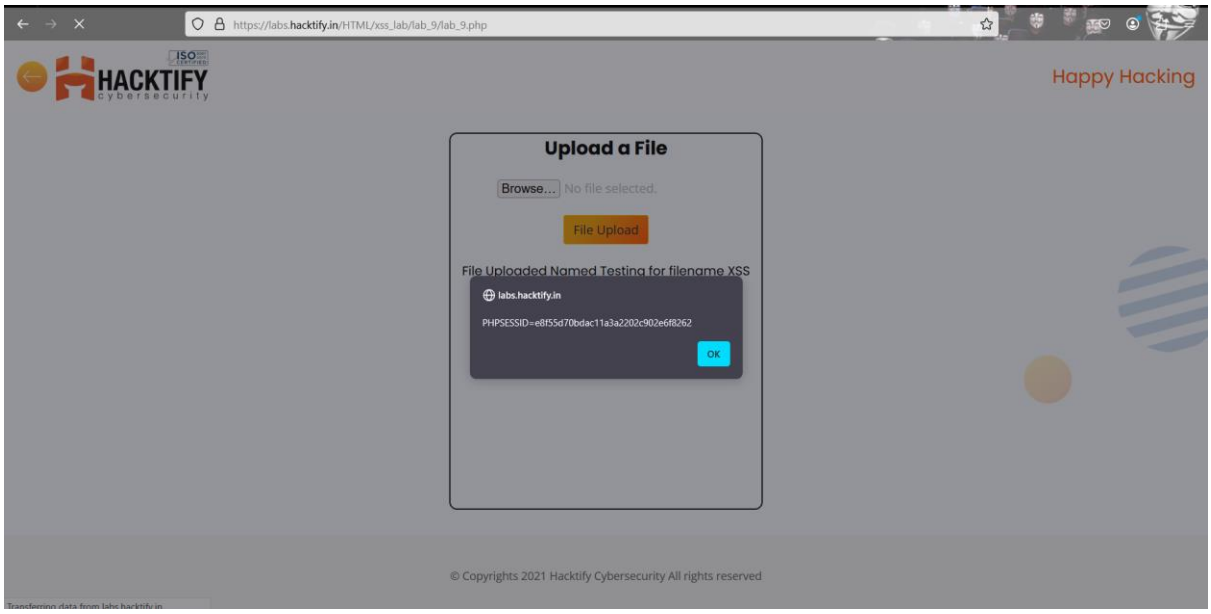


1.9. XSS with file upload (file content)

Reference	Risk Rating
XSS with file upload (file content)	**Medium**
Tools Used	
Xss encoded payload	
Vulnerability Description	
Reflected XSS occurs when an attacker injects malicious scripts into a web application, which are then reflected back to the victim's browser and executed. This typically happens when user input is included in the server's response without proper validation or encoding.	
How It Was Discovered	
Automatic Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_9/lab_9.php	
Consequences of not Fixing the Issue	
Attackers can inject scripts that redirect users to malicious websites or download malware onto their devices.	
Suggested Countermeasures	
Use a strong Content Security Policy to restrict the sources from which scripts can be loaded and executed.	
References	
https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability

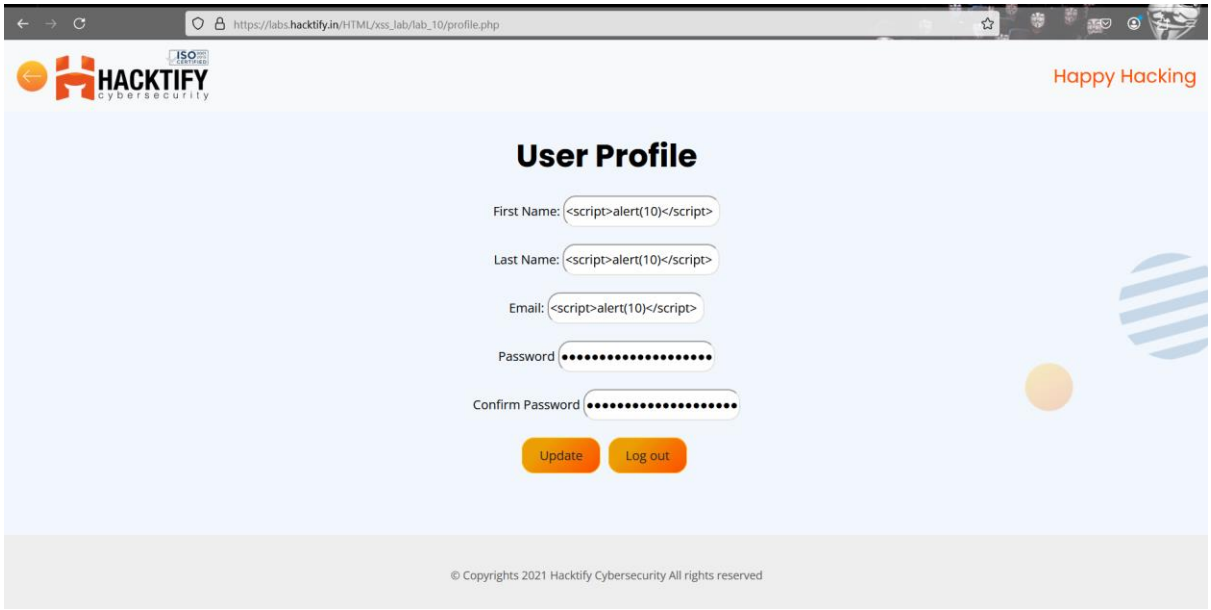


1.10. Stored Everywhere!

Reference	Risk Rating
Stored Everywhere!	**Medium**
Tools Used	
Xss payload	
Vulnerability Description	
Occurs when an attacker injects malicious scripts into a web application, which are then stored on the server (e.g., in a database, comment section, or user profile).	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_10/profile.php	
Consequences of not Fixing the Issue	
Attackers can create fake login forms or other deceptive content to trick users into revealing sensitive information.	
Suggested Countermeasures	
Validate and sanitize all user inputs on both the client and server sides.	
References	
https://cwe.mitre.org/data/definitions/79.html	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

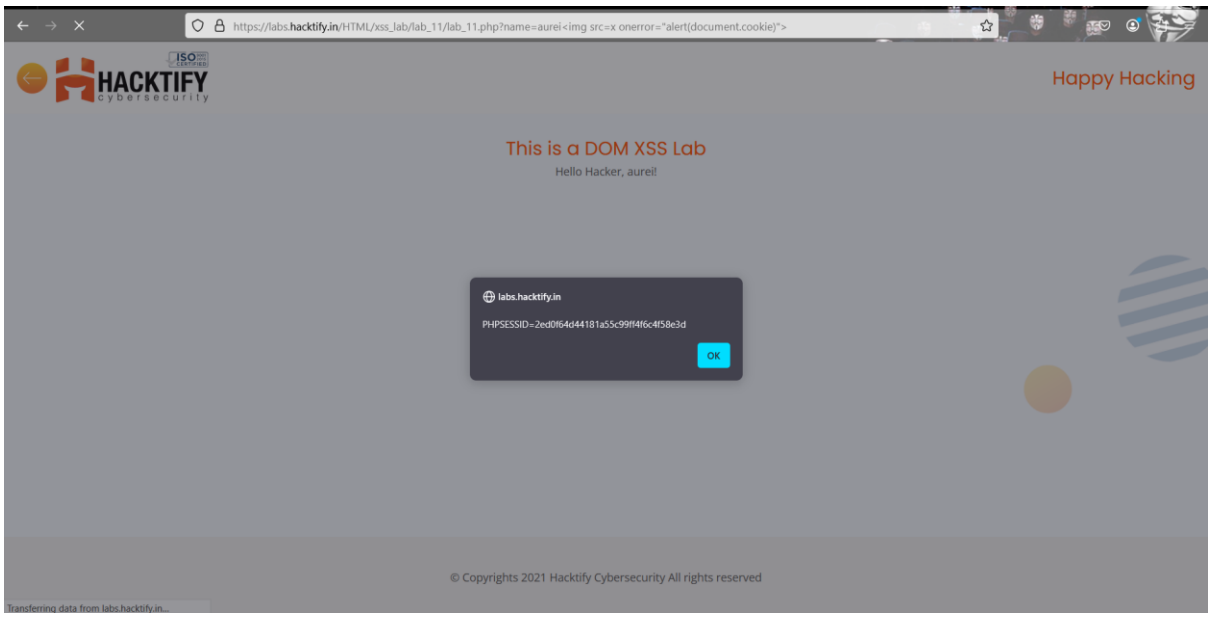


1.11. DOM's are love!

Reference	Risk Rating
DOM's are love!	**High**
Tools Used	
Xss payload	
Vulnerability Description	
DOM-based XSS is a type of XSS vulnerability where the attack payload is executed as a result of modifying the Document Object Model (DOM) in the victim's browser.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/xss_lab/lab_11/lab_11.php?name=aurei%3Cimg%20src=x%20onerror=%22alert(document.cookie)%22%3E	
Consequences of not Fixing the Issue	
Attackers can steal sensitive information such as session cookies, login credentials, or personal data.	
Suggested Countermeasures	
Use the Secure flag to ensure cookies are only sent over HTTPS.	
References	
https://cheatsheetseries.owasp.org/cheatsheets/DOM_based_XSS_Prevention_Cheat_Sheet.html	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab



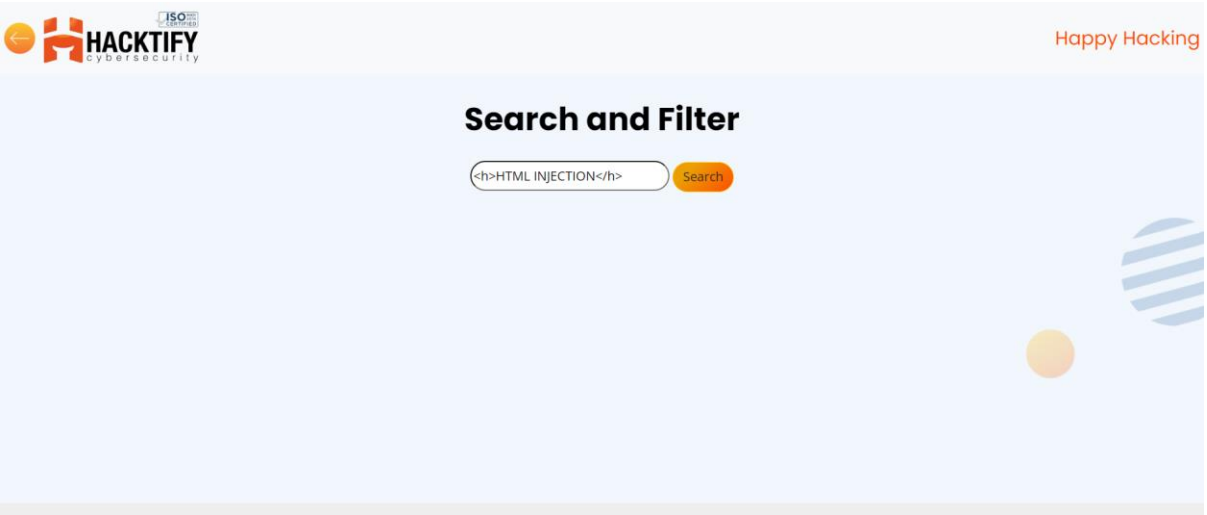
2. HTML INJECTION

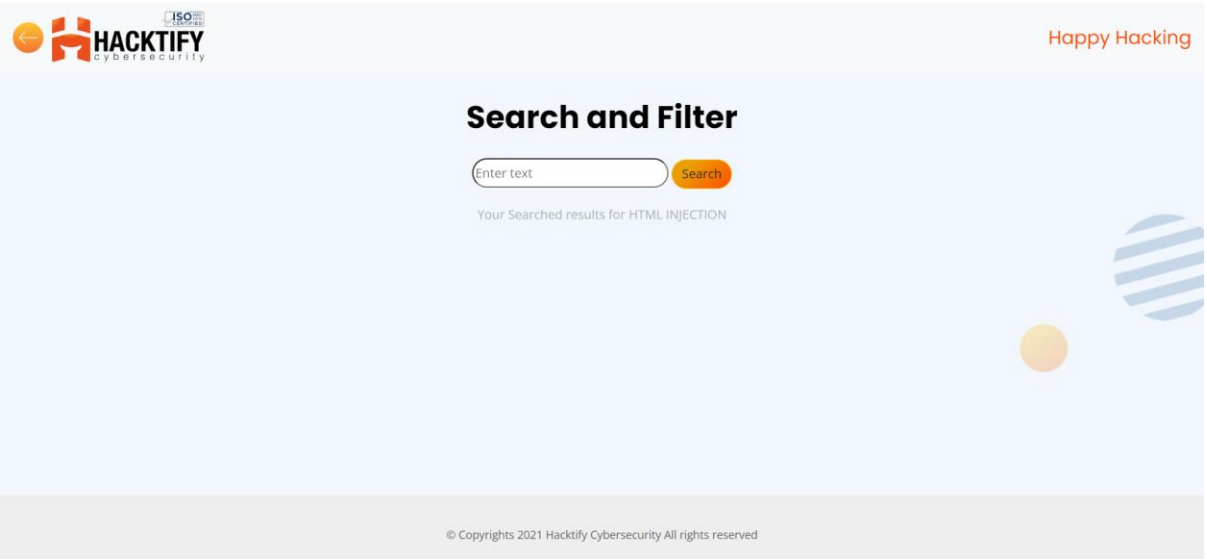
2.1. HTML's are easy!

Reference	Risk Rating
HTML's are easy!	Low
Tools Used	
HTML code tags	
Vulnerability Description	
HTML Injection is a vulnerability where an attacker injects malicious HTML code into a web page by exploiting improper handling of user input.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_1/html_injection_1.php	
Consequences of not Fixing the Issue	
Attackers can inject spammy content or links, negatively impacting the website's search engine ranking.	
Suggested Countermeasures	
Validate and sanitize all user inputs on both the client and server sides. Ensure that inputs conform to expected formats (e.g., alphanumeric, no special characters).	
References	
https://www.imperva.com/learn/application-security/html-injection/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab



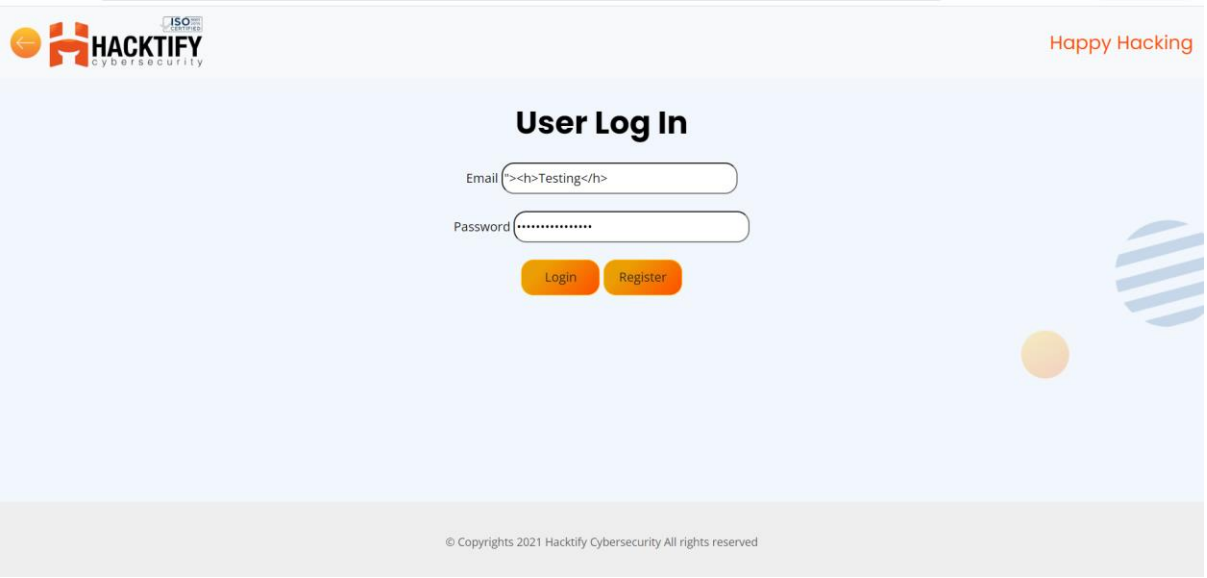


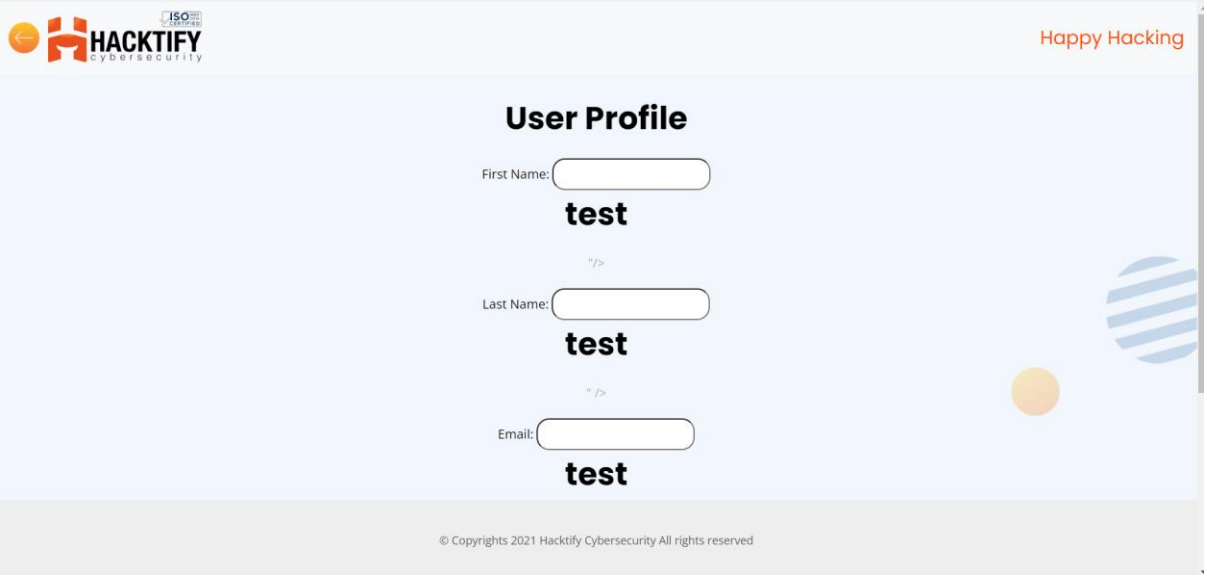
2.2. Let me store them

Reference	Risk Rating
Let me store them	**Low**
Tools Used	
HTML code tags	
Vulnerability Description	
HTML Injection is a vulnerability where an attacker injects malicious HTML code into a web page by exploiting improper handling of user input.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_2/profile.php	
Consequences of not Fixing the Issue	
Attackers can inject fake forms or content to steal sensitive information like login credentials, credit card details, or personal data.	
Suggested Countermeasures	
Validate and sanitize all user inputs on both the client and server sides. Ensure that inputs conform to expected formats (e.g., alphanumeric, no special characters).	
References	
https://www.imperva.com/learn/application-security/html-injection/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab



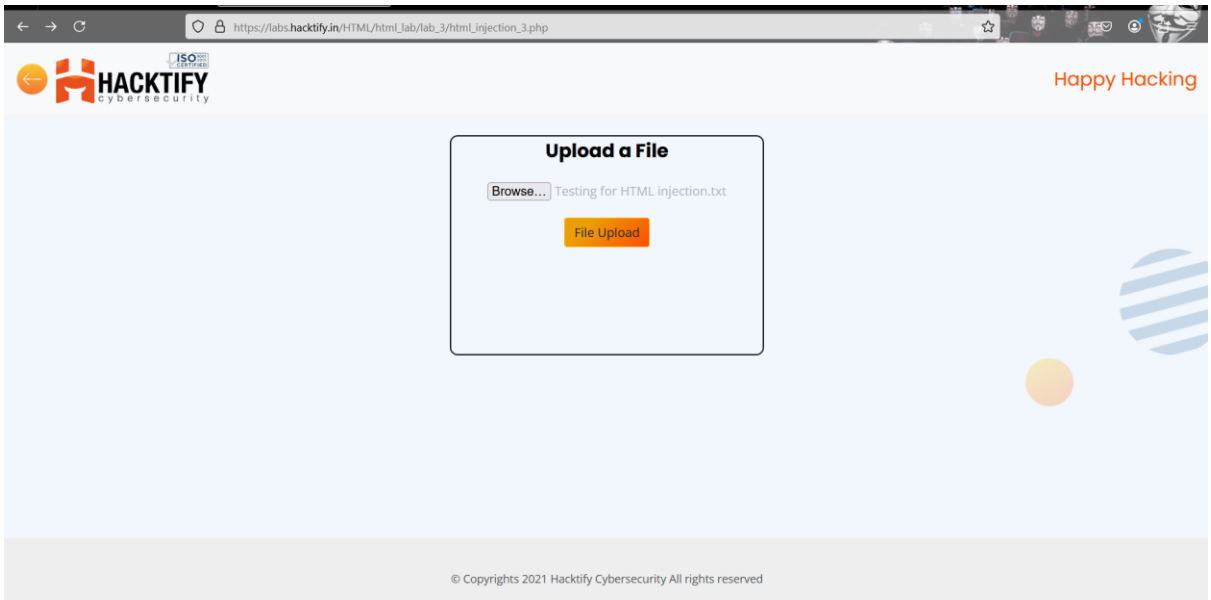


2.3. Filenames are also vulnerable

Reference	Risk Rating
Filenames are also vulnerable	**Medium**
Tools Used	
HTML payload and Burp suite	
Vulnerability Description	
Malicious code is permanently stored on the target server. This code is then served to users every time they access a particular page. Once the malicious code is in place, it can affect a large number of users without the attacker having to do anything further.	
How It Was Discovered	
Automated Tools	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_3/html_injection_3.php	
Consequences of not Fixing the Issue	
By injecting malicious scripts into web pages, attackers can force users' browsers to download and execute malware without their knowledge.	
Suggested Countermeasures	
Give some Suggestions to stand against this vulnerability	
References	
https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html https://www.imperva.com/learn/application-security/html-injection/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab



Burp Project Intruder Repeater View Help

Burp Suite Community Edition v2024.12.1 - Temporary Project

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn

Intercept HTTP history WebSockets history Match and replace Proxy settings

Intercept on | Forward | Drop

Request to https://labs.hacktify.in:443

Time	Type	Direction	Method	URL
01:21:00 17 Fe...	HTTP	→ Request	POST	https://labs.hacktify.in/HTML/html_lab/lab_3/html_injection_3.php

Request

Pretty Raw Hex

```
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17 Priority: u=0, i
18 Te: trailers
19
20 ------geckoformboundary648daacad38b086f5575b0f69ff2f168
21 Content-Disposition: form-data; name="image"; filename=<h1>Testing for HTML injection.txt</h1>
22 Content-Type: text/plain
23
24 Testing for filename XSS injection attack
25
26 ------geckoformboundary648daacad38b086f5575b0f69ff2f168
27 Content-Disposition: form-data; name="upload"
28
```

Search 0 highlights

Event log (97) All issues

https://labs.hacktify.in/HTML/html_lab/lab_3/html_injection_3.php

HACKTIFY cybersecurity

Happy Hacking

Upload a File

Browse... No file selected.

File Upload

File Uploaded

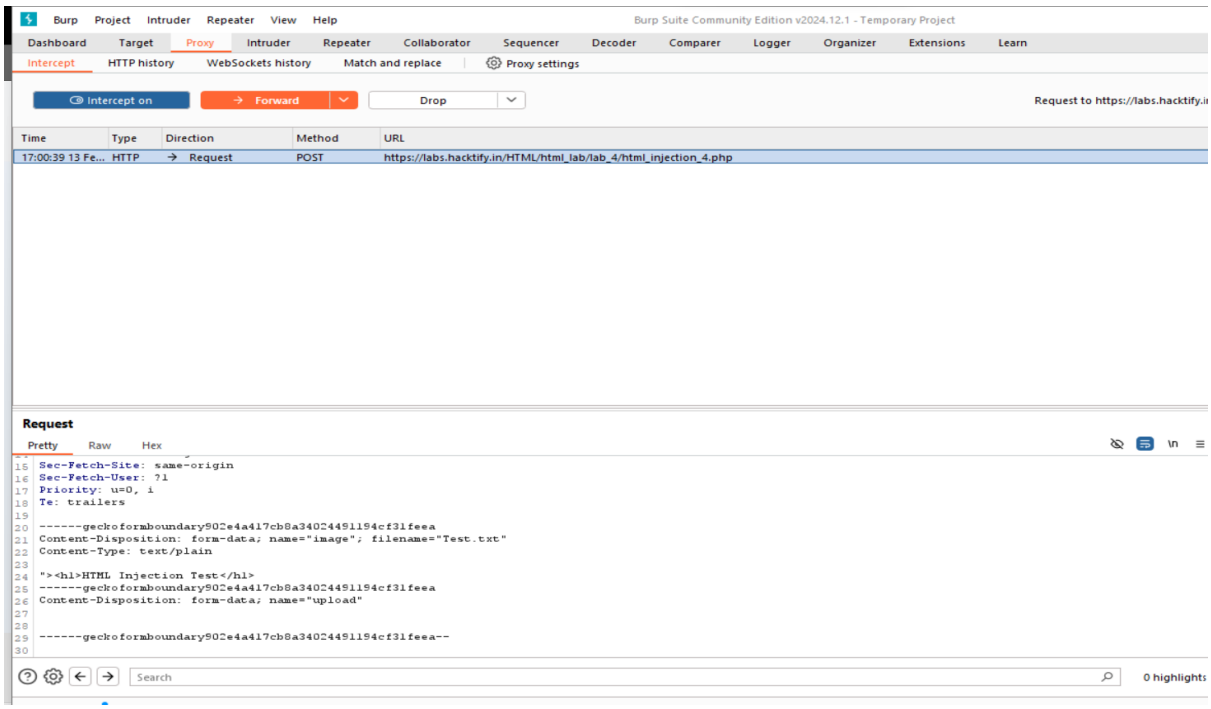
Testing

2.4. File content and HTML injection a perfect pair

Reference	Risk Rating
File content and HTML injection a prefect pair	**Medium**
Tools Used	
HTML payload and Burp suite	
Vulnerability Description	
HTML injection via file content occurs when the application processes or displays the content of an uploaded file without proper sanitization.	
How It Was Discovered	
Automated Tools	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_4/html_injection_4.php	
Consequences of not Fixing the Issue	
Attackers can modify the content of the website, leading to reputational damage and loss of user trust.	
Suggested Countermeasures	
Conduct regular code reviews and security audits to identify and fix vulnerabilities.	
References	
https://www.imperva.com/learn/application-security/html-injection/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab



HACKTIFY cybersecurity

Hap

Upload a File

Browse… No file selected.

File Upload

HTML Injection Test

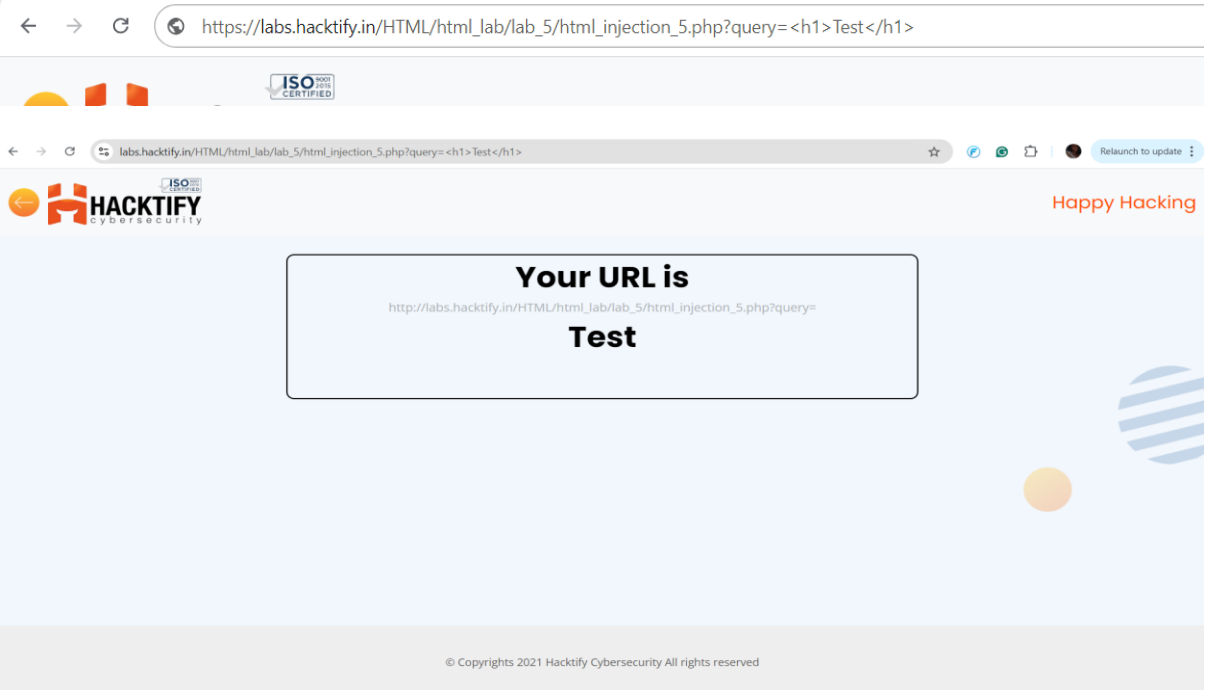
File Uploaded Named Test.txt

2.5. Injecting HTML using URL

Reference	Risk Rating
Injecting HTML using URL	**Medium**
Tools Used	
HTML payloads	
Vulnerability Description	
Injecting HTML using a URL involves manipulating URL parameters to include HTML	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_5/html_injection_5.php?query=%3Ch1%3ETest%3C/h1%3E	
Consequences of not Fixing the Issue	
Attackers can inject malicious scripts to redirect users to phishing sites or download malware onto their devices.	
Suggested Countermeasures	
Perform regular penetration testing and code reviews to identify and fix vulnerabilities.	
References	
https://www.imperva.com/learn/application-security/html-injection/	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

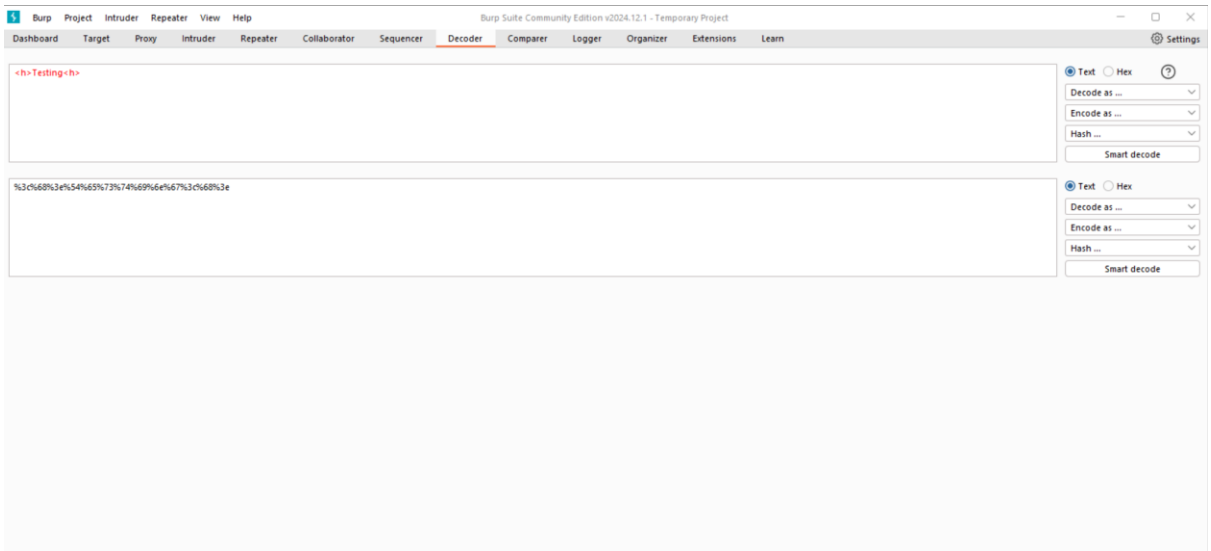


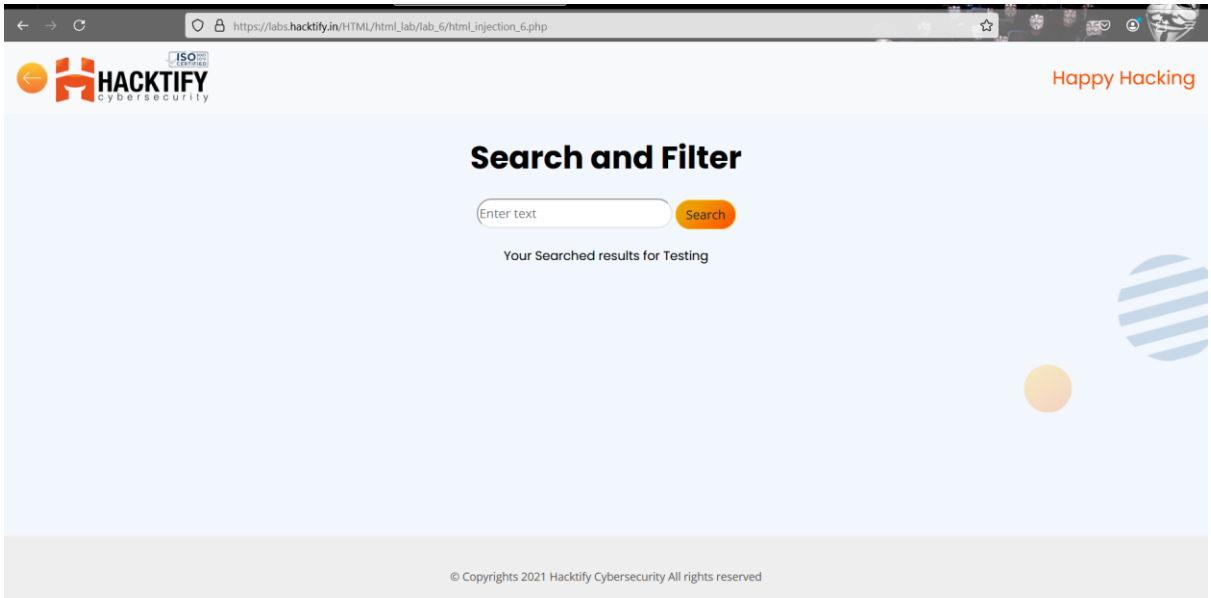
2.6. Encode IT!

Reference	Risk Rating
Encode IT!	**High**
Tools Used	
HTML encoded payloads	
Vulnerability Description	
HTML injection encoding payloads can help bypass basic input filters or sanitization mechanisms. Different encoding techniques, such as hex encoding, double hex encoding, and Base64 encoding, can be used to obfuscate the payload.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/html_lab/lab_5/html_injection_5.php?query=%3Ch1%3ETest%3C/h1%3E	
Consequences of not Fixing the Issue	
Attackers can inject malicious scripts to redirect users to phishing sites or download malware onto their devices.	
Suggested Countermeasures	
Perform regular penetration testing and code reviews to identify and fix vulnerabilities.	
References	
https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html	

Proof of Concept

This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab




https://labs.hacktify.in/HTML/html_lab/lab_6/html_injection_6.php
HACKTIFY
cybersecurity
Happy Hacking
Search and Filter
Enter text
Search
Your Searched results for Testing
© Copyrights 2021 Hacktify Cybersecurity All rights reserved