Penetration Testing Report

Full Name: Aashutosh Thakur
Program: HCPT
Date: 01/03/2025

Introduction

This report document hereby describes the proceedings and results of a Black Box security assessment conducted against the **Week 3 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 3 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	**Cross-Origin Resource Sharing Labs, Cross-Site Request Forgery Labs**

3. Summary

Outlined is a Black Box Application Security assessment for the **Week 3 Labs**.

Total number of Sub-labs: 13 Sub-labs

High	Medium	Low
5	4	4

High - **5**

Medium - **4**

Low - 4

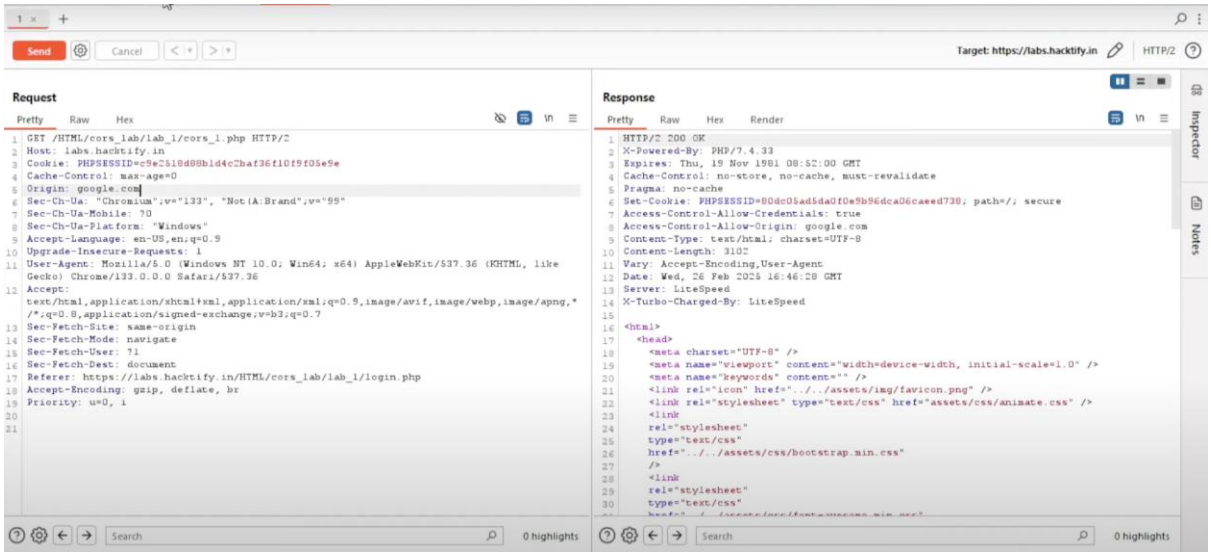
1. Cross-Origin Resource Sharing Labs

1.1. CORS With Arbitrary Origin

Reference	Risk Rating
CORS With Arbitrary Origin	Low
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
Insecure CORS configuration allows any origin to access server resources.	
How It Was Discovered	
Automated tools like Burp Suite or manual inspection of CORS headers.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_1/cors_1.php	
Consequences of not Fixing the Issue	
Data theft, CSRF attacks, privilege escalation, and reputation damage.	
Suggested Countermeasures	
Restrict allowed origins, validate origin headers, use SameSite cookies, and log requests.	
References	
Https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html	

Proof of Concept

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

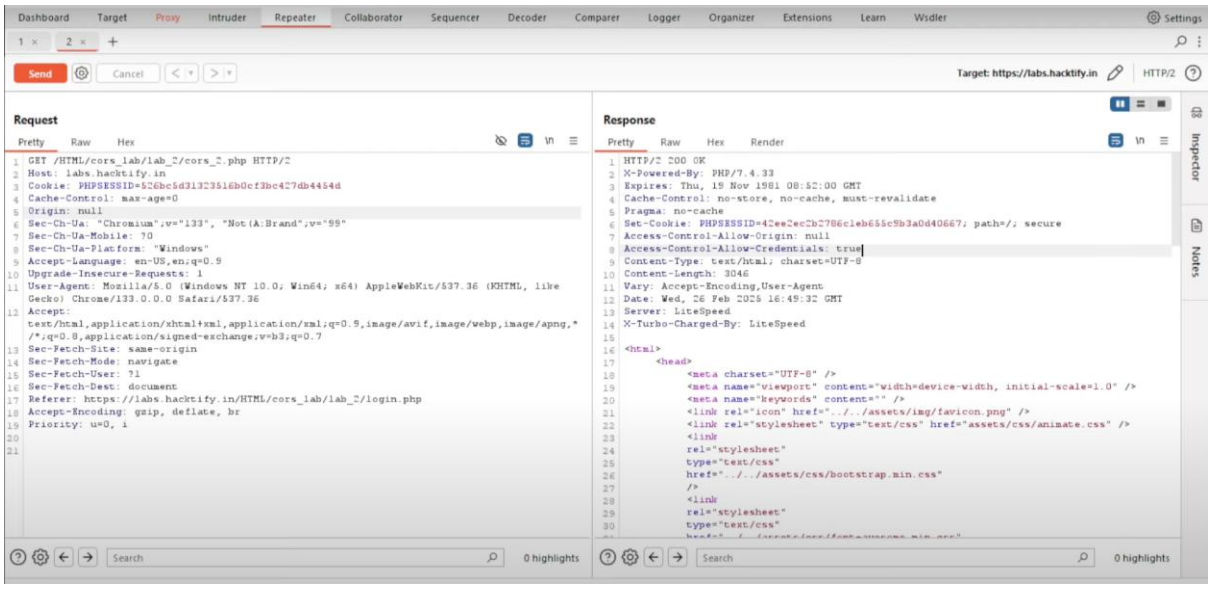


1.2. CORS with Null origin

Reference	Risk Rating
CORS with Null origin	Low
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
Insecure CORS configuration allows any origin to access server resources.	
How It Was Discovered	
Automated tools like Burp Suite or manual inspection of CORS headers.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_2/cors_2.php	
Consequences of not Fixing the Issue	
Data theft, CSRF attacks, privilege escalation, and reputation damage.	
Suggested Countermeasures	
Restrict allowed origins, validate origin headers, use SameSite cookies, and log requests.	
References	
Https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html	

Proof of Concept

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

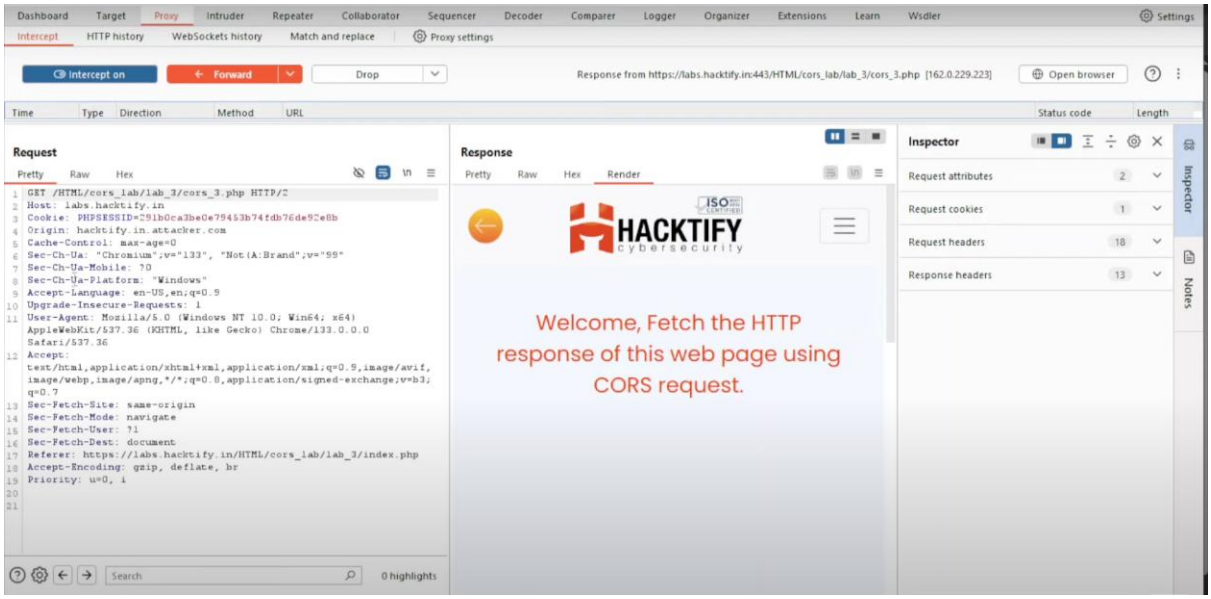


1.3. CORS with prefix match

Reference	Risk Rating
CORS with prefix match	Medium
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
Insecure CORS configuration allows any origin to access server resources.	
How It Was Discovered	
Automated tools like Burp Suite or manual inspection of CORS headers.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_3/cors_3.php	
Consequences of not Fixing the Issue	
Data theft, CSRF attacks, privilege escalation, and reputation damage.	
Suggested Countermeasures	
Restrict allowed origins, validate origin headers, use SameSite cookies, and log requests.	
References	
Https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html	

Proof of Concept

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

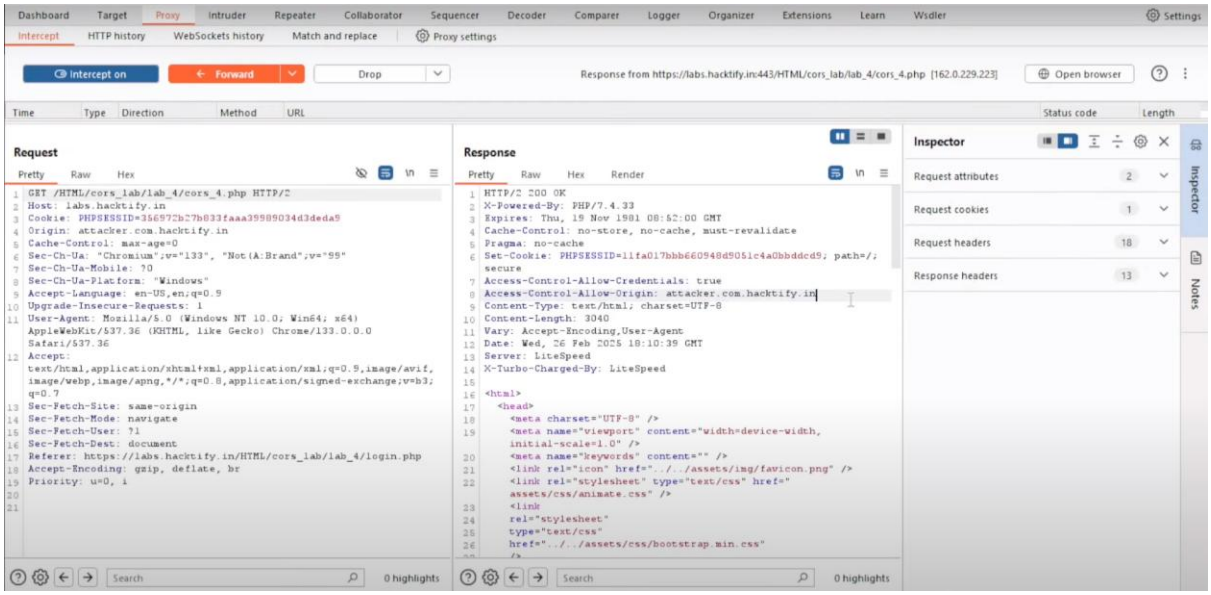


1.4. CORS with suffix match

Reference	Risk Rating
CORS with suffix match	Medium
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
Insecure CORS configuration allows any origin to access server resources.	
How It Was Discovered	
Automated tools like Burp Suite or manual inspection of CORS headers.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_4/cors_4.php	
Consequences of not Fixing the Issue	
Data theft, CSRF attacks, privilege escalation, and reputation damage.	
Suggested Countermeasures	
Restrict allowed origins, validate origin headers, use SameSite cookies, and log requests.	
References	
Https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html	

Proof of Concept

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



1.5. CORS with Escape dot

Reference	Risk Rating
CORS with Escape dot	High
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
Insecure CORS configuration allows any origin to access server resources.	
How It Was Discovered	
Automated tools like Burp Suite or manual inspection of CORS headers.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_5/cors_5.php	
Consequences of not Fixing the Issue	
Data theft, CSRF attacks, privilege escalation, and reputation damage.	
Suggested Countermeasures	
Restrict allowed origins, validate origin headers, use SameSite cookies, and log requests.	
References	
Https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html	

Proof of Concept

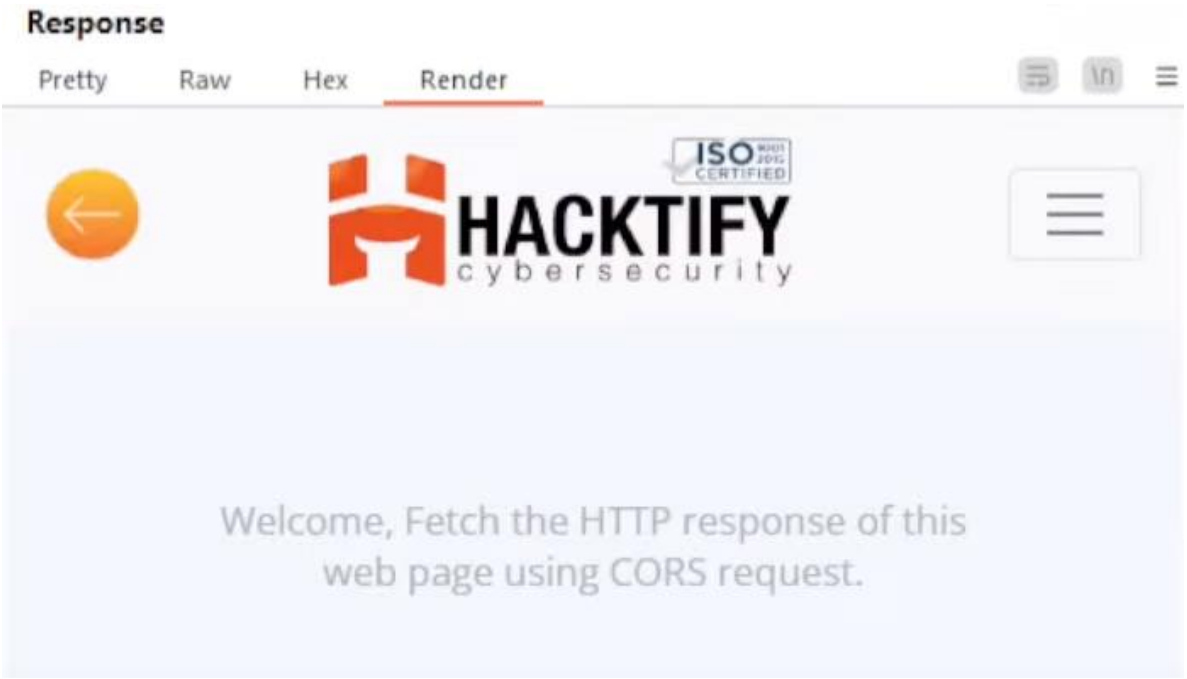
This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab

1.6. CORS with Substring match

Reference	Risk Rating
CORS with Substring match	**High**
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
Insecure CORS configuration allows any origin to access server resources.	
How It Was Discovered	
Automated tools like Burp Suite or manual inspection of CORS headers.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_6/cors_6.php	
Consequences of not Fixing the Issue	
Data theft, CSRF attacks, privilege escalation, and reputation damage.	
Suggested Countermeasures	
Restrict allowed origins, validate origin headers, use SameSite cookies, and log requests.	
References	
Https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html	

Proof of Concept

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

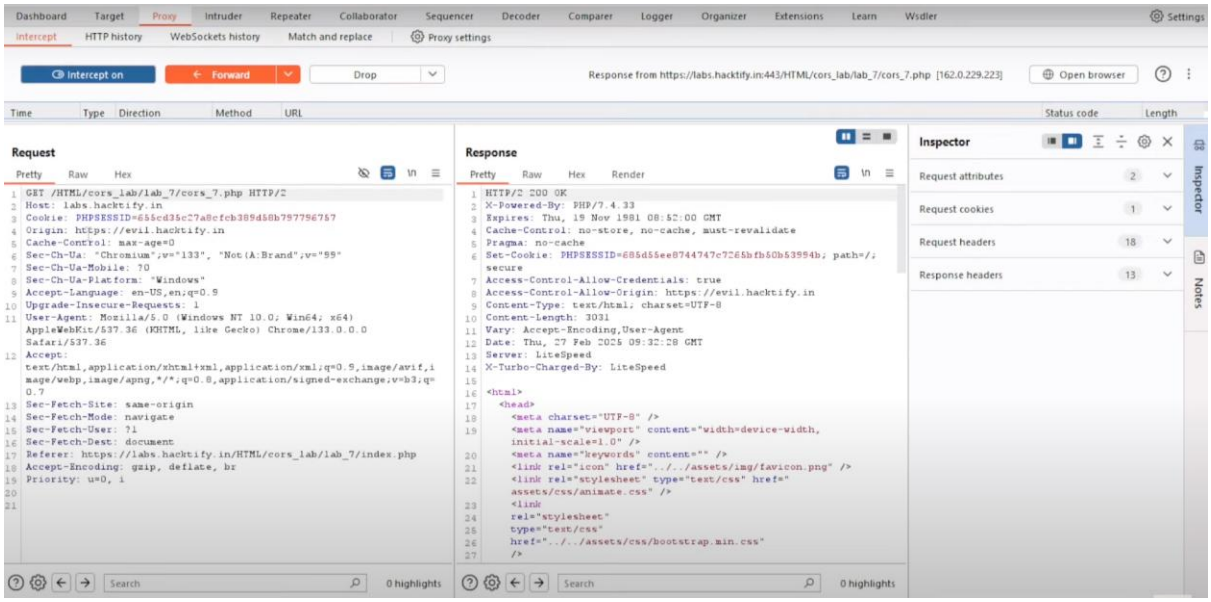


1.7. CORS with Arbitrary Subdomain

Reference	Risk Rating
CORS with Arbitrary Subdomain	High
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
Insecure CORS configuration allows any origin to access server resources.	
How It Was Discovered	
Automated tools like Burp Suite or manual inspection of CORS headers.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_7/cors_7.php	
Consequences of not Fixing the Issue	
Data theft, CSRF attacks, privilege escalation, and reputation damage.	
Suggested Countermeasures	
Restrict allowed origins, validate origin headers, use SameSite cookies, and log requests.	
References	
Https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html	

Proof of Concept

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



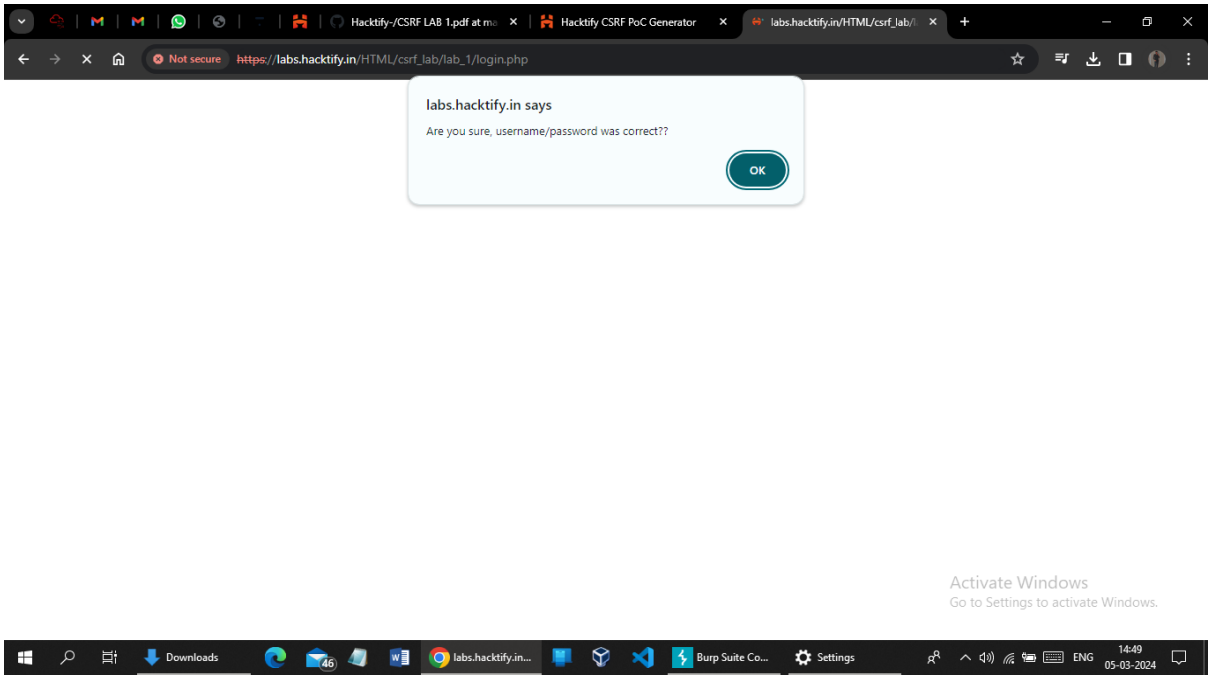
2. Cross-Site Request Forgery Labs

2.1. Eassyy CSRF

Reference	Risk Rating
Eassyy CSRF	Low
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
CSRF allows attackers to make unauthorized requests on behalf of authenticated users.	
How It Was Discovered	
Automated tools or manual testing by submitting requests from unauthorized sites.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_1/passwordChange.php	
Consequences of not Fixing the Issue	
Unauthorized actions, data manipulation, account compromise, and financial loss.	
Suggested Countermeasures	
Implement CSRF tokens, use SameSite cookies, and validate referer headers.	
References	
Https://developer.mozilla.org/en-US/docs/Glossary/CSRF	

Proof of Concept

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

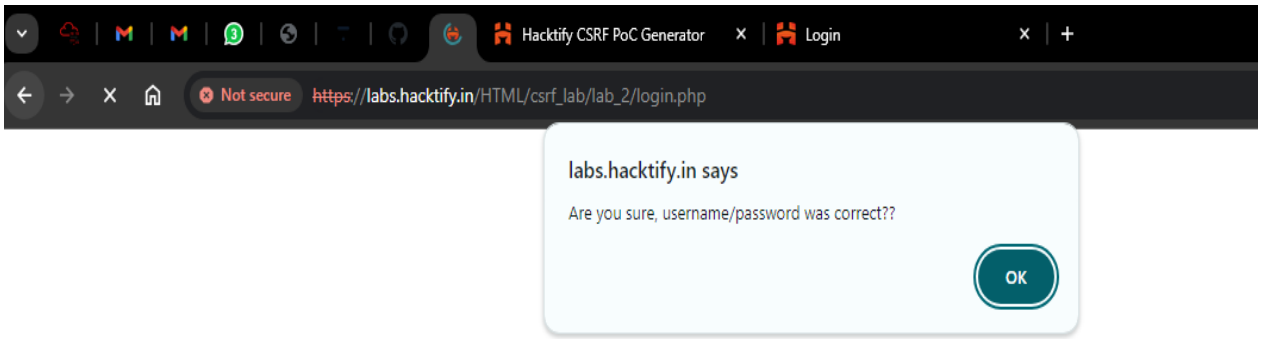


2.2. Always Validate Tokens

Reference	Risk Rating
Always Validate Tokens	**Medium**
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
CSRF allows attackers to make unauthorized requests on behalf of authenticated users.	
How It Was Discovered	
Automated tools or manual testing by submitting requests from unauthorized sites.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php	
Consequences of not Fixing the Issue	
Unauthorized actions, data manipulation, account compromise, and financial loss.	
Suggested Countermeasures	
Implement CSRF tokens, use SameSite cookies, and validate referer headers.	
References	
Https://developer.mozilla.org/en-US/docs/Glossary/CSRF	

Proof of Concept

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



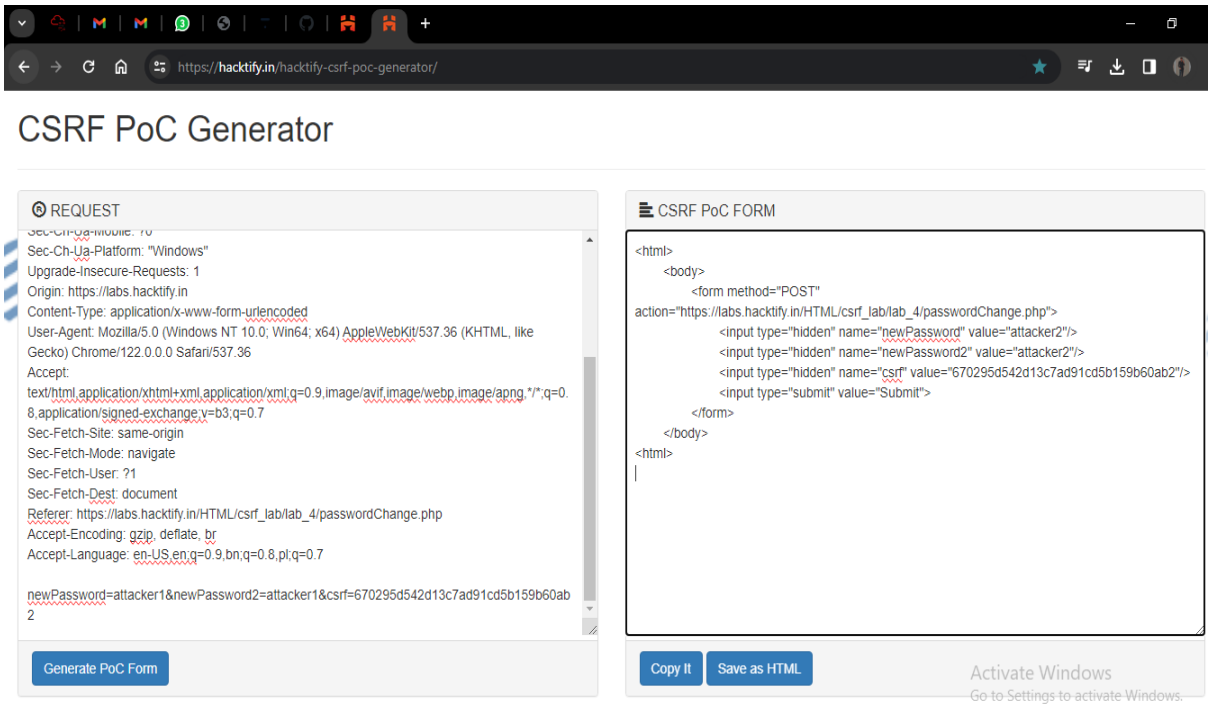
2.3. I hate when someone uses my tokens!

Reference	Risk Rating

I hate when someone uses my tokens!	Medium
Tools Used	
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.	
Vulnerability Description	
CSRF allows attackers to make unauthorized requests on behalf of authenticated users.	
How It Was Discovered	
Automated tools or manual testing by submitting requests from unauthorized sites.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_3/passwordChange.php	
Consequences of not Fixing the Issue	
Unauthorized actions, data manipulation, account compromise, and financial loss.	
Suggested Countermeasures	
Implement CSRF tokens, use SameSite cookies, and validate referer headers.	
References	
Https://developer.mozilla.org/en-US/docs/Glossary/CSRF	

Proof of Concept

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



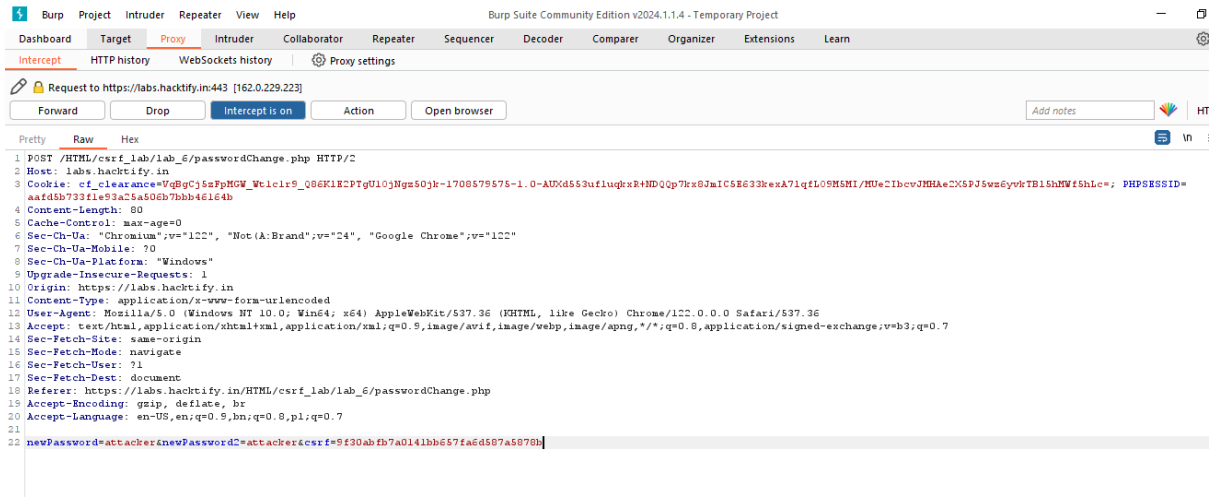
2.4. GET Me or POST ME

Reference	Risk Rating
GET Me or POST ME	Low

Tools Used
Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.
Vulnerability Description
CSRF allows attackers to make unauthorized requests on behalf of authenticated users.
How It Was Discovered
Automated tools or manual testing by submitting requests from unauthorized sites.
Vulnerable URLs
https://labs.hacktify.in/HTML/csrf_lab/lab_4/passwordChange.php
Consequences of not Fixing the Issue
Unauthorized actions, data manipulation, account compromise, and financial loss.
Suggested Countermeasures
Implement CSRF tokens, use SameSite cookies, and validate referer headers.
References
Https://developer.mozilla.org/en-US/docs/Glossary/CSRF

Proof of Concept

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



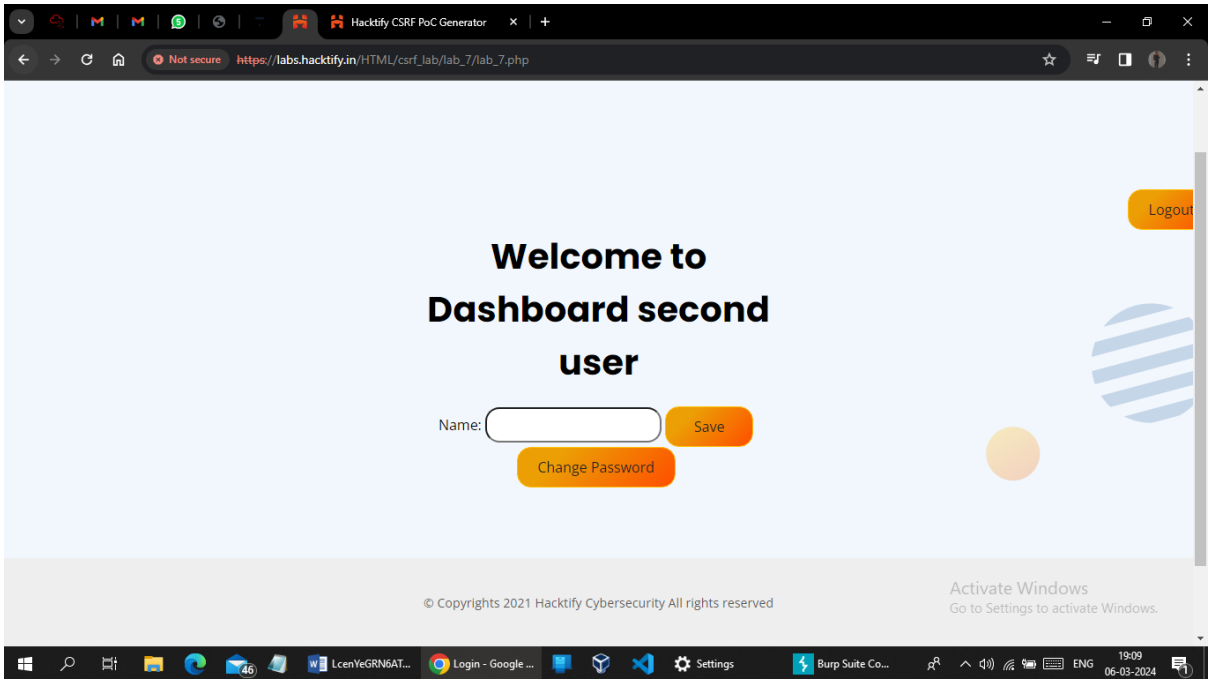
2.5. XSS the saviour

Reference	Risk Rating
XSS the saviour	High
Tools Used	

Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.
Vulnerability Description
CSRF allows attackers to make unauthorized requests on behalf of authenticated users.
How It Was Discovered
Automated tools or manual testing by submitting requests from unauthorized sites.
Vulnerable URLs
https://labs.hacktify.in/HTML/csrf_lab/lab_5/passwordChange.php
Consequences of not Fixing the Issue
Unauthorized actions, data manipulation, account compromise, and financial loss.
Suggested Countermeasures
Implement CSRF tokens, use SameSite cookies, and validate referer headers.
References
Https://developer.mozilla.org/en-US/docs/Glossary/CSRF

Proof of Concept

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



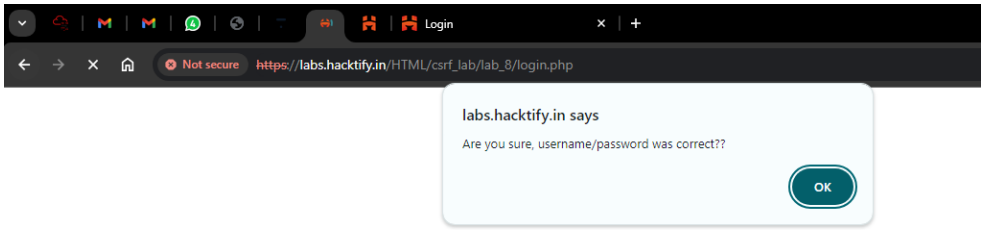
2.6. rm -rf token

Reference	Risk Rating
rm -rf token	High
Tools Used	

Burp Suite, OWASP ZAP, Postman, Browser Developer Tools.
Vulnerability Description
CSRF allows attackers to make unauthorized requests on behalf of authenticated users.
How It Was Discovered
Automated tools or manual testing by submitting requests from unauthorized sites.
Vulnerable URLs
https://labs.hacktify.in/HTML/csrf_lab/lab_6/passwordChange.php
Consequences of not Fixing the Issue
Unauthorized actions, data manipulation, account compromise, and financial loss.
Suggested Countermeasures
Implement CSRF tokens, use SameSite cookies, and validate referer headers.
References
Https://developer.mozilla.org/en-US/docs/Glossary/CSRF

Proof of Concept

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



NOTES:

- **Everything mentioned inside {} has to be changed based on your week, labs and sub-labs.**
- **If you have 2 labs in same week you need to mention that, if not ignore those mentions for lab 2.**
- **Here it is given with 2 Sub-labs vulnerability, you need to add all the sub-labs based on your labs.**
- **Don't forget to add the screenshot of the vulnerability in the proof of concept.**
- **This NOTE session is only for your reference, don't forget to delete this in the report you submit.**