Penetration Testing Report

Full Name: Rayen Gaied
Program: HCS - Penetration Testing Internship Week-2
Date: 01/03/2024

Introduction

This report document hereby describes the proceedings and results of a Black Box security assessment conducted against the **Week 2 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 2 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	**Lab 1:** **Cross Site Scripting** • **Cross-site scripting (also known as XSS) is a web security vulnerability that allows an attacker to compromise the interactions that users have with a vulnerable application. Cross-site scripting vulnerabilities normally allow an attacker to masquerade as a victim user, to carry out any actions that the user is able to perform, and to access any of the user's data. If the victim user has privileged access within the application, then the attacker might be able to gain full control over all the application's functionality and data.** **Lab 2:** **Insecure direct object reference** • **An insecure direct object reference (IDOR) is an access control vulnerability where invalidated user input can be used for unauthorized access to resources or operations. It occurs when an attacker gains direct access by using user-supplied input to an object that has no authorization to access. Attackers can bypass the authorization mechanism to access resources in the system directly by exploiting this vulnerability.**

3. Summary

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

Total number of Sub-labs: 15 Sub-labs

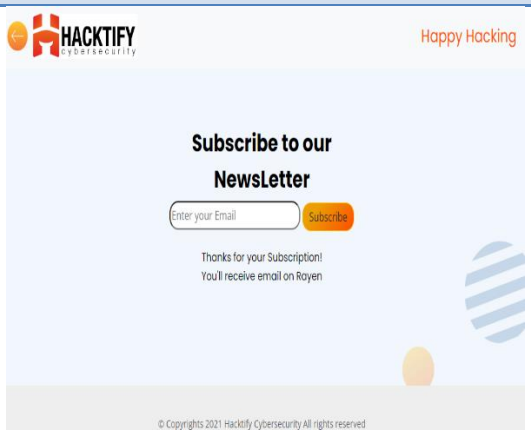
High	Medium	Low
4	5	6

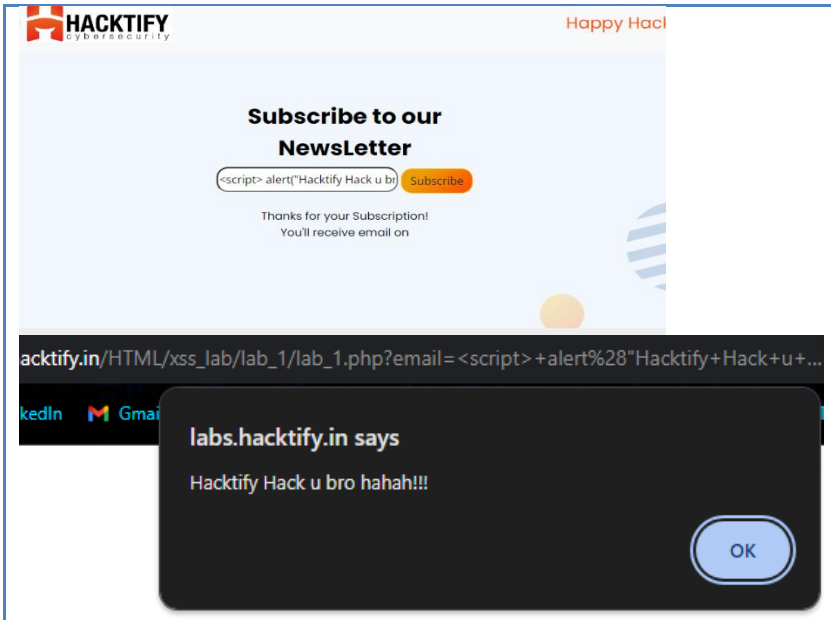
High	-	**Number of Sub-labs with hard difficulty level**
Medium	-	**Number of Sub-labs with Medium difficulty level**
Low	-	**Number of Sub-labs with Easy difficulty level**

1. Cross Site Scripting

1.1. Let's Do IT!

Reference	Risk Rating
Let's Do IT!	Low
Tools Used	
HTML Payload	
Vulnerability Description	
when we entered 'alert(“Hacktify Hack u bro hahah!!!”)' JavaScript treated ‘Hacktify Hack u bro hahah!!! ‘ as a variable and started a journey to find the variable and which being my input obviously won't be any variable so will not give us an alert box.	
How It Was Discovered	





Vulnerable URLs

- https://labs.hacktify.in/HTML/xss_lab/lab_1/lab_1.php?email=%3Cscript%3E+alert%28%22Hacktify+Hack+u+bro+hahah%21%21%21%22%29+%3C%2Fscript%3E

Consequences of not Fixing the Issue

- Session hijacking: Attackers can steal session cookies and hijack legitimate user accounts, potentially leading to unauthorized access to sensitive information or systems.
- Data theft: XSS attacks can be used to steal sensitive data such as login credentials, credit card information, and personally identifiable information (PII).
- Malicious redirects: Attackers can redirect users to malicious websites or perform other malicious operations on the user's machine under the guise of the vulnerable site.
- Account compromise: If an attacker gains access to an account with administrative privileges, they can perform unauthorized actions, potentially leading to severe damage to the web application.
- Reputation damage: XSS vulnerabilities can undermine the trust users have in a company, leading to negative publicity and potential loss of customers.

Suggested Countermeasures

- Input validation: Validate and sanitize all user inputs to ensure they do not contain malicious scripts that could be executed on the website.
- Output encoding: Encode user-generated content before displaying it on the website to prevent browsers from interpreting it as executable code.
- Content Security Policy (CSP): Implement a CSP to restrict the sources from which certain types of content can be loaded on your website, reducing the risk of XSS attacks.
- Use security libraries: Utilize security libraries like OWASP ESAPI to help prevent common security vulnerabilities, including XSS attacks.

- Regular security audits: Conduct regular security audits and penetration testing to identify and address any vulnerabilities in your web application.

References

- https://owasp.org/www-community/attacks/xss/

1.2. Balancing Is Important In Life!

Reference	Risk Rating
Balancing Is Important In Life!	**Low**

Tools Used

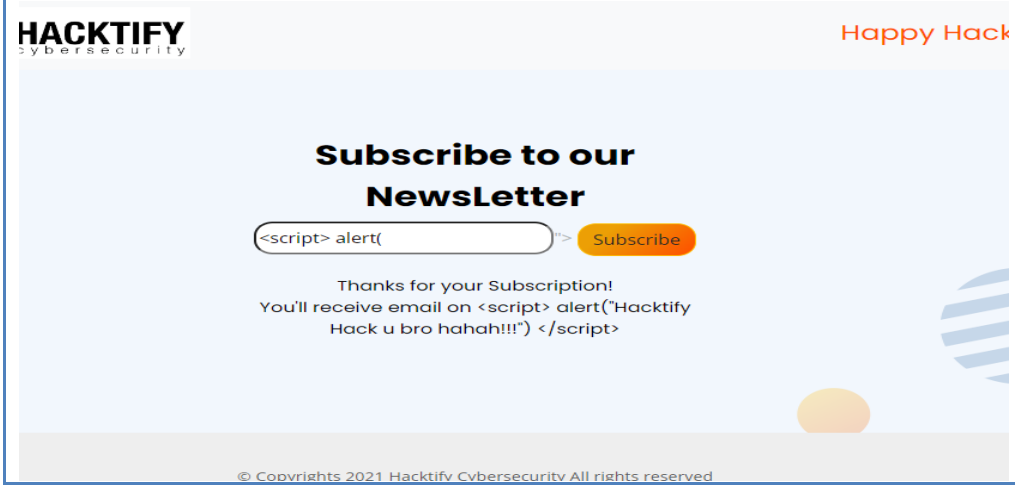
HTML Payload

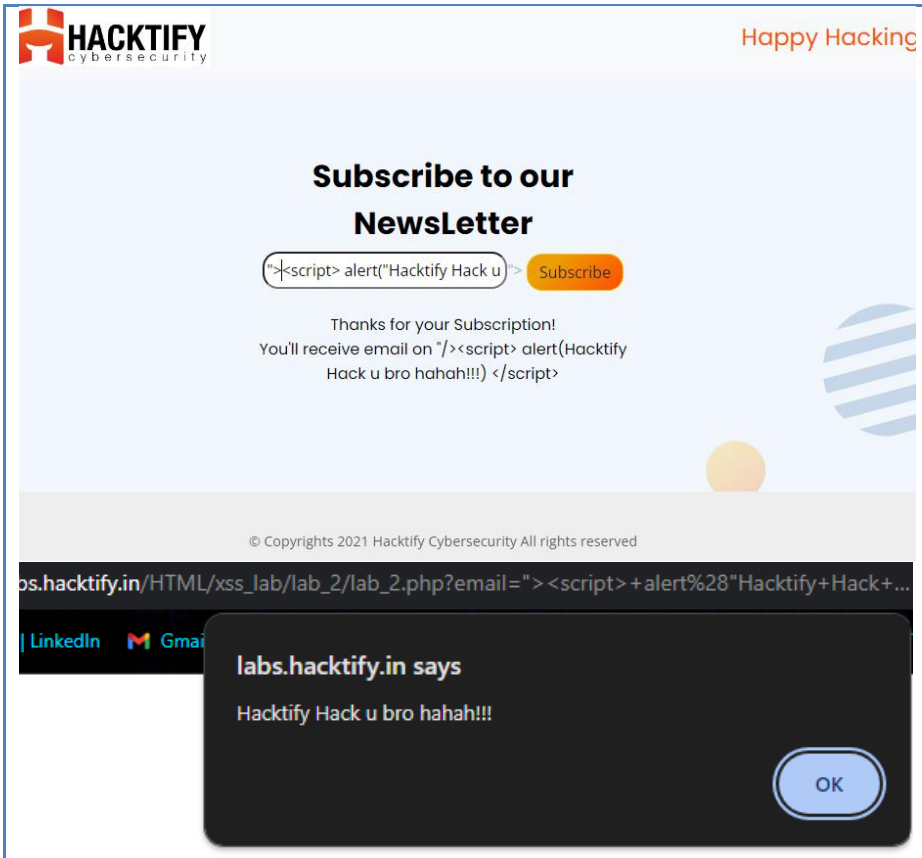
Vulnerability Description

- our value parameter is vulnerable to XSS. we should always check the content displayed on frontend or over the user interface, but also the parameters or attributes in the page source.

How It Was Discovered

```
r-section">
ainer">
page-titlee">Hacktify</h2> -->

containers">
scribe to our NewsLetter</h1>

m action="lab_2.php" method="GET">
ut type="text" name="email" class="field" placeholder="Enter your Email" value="<script> alert("Hacktify Hack u bro hahah!!!") </script>">
ut type="submit" value="Subscribe" class="btn btn-warning">
rm>
nter><center><br><h2>Thanks for your Subscription!<br> You'll receive email on <b>&lt;script&gt; alert(&quot;Hacktify Hack u bro hahah!!!&q
```





Vulnerable URLs

- https://labs.hacktify.in/HTML/xss_lab/lab_2/lab_2.php?email=%22%3E%3Cscript%3E+alert%28%22Hacktify+Hack+u+bro+hahah%21%21%21%22%29+%3C%2Fscript%3E

Consequences of not Fixing the Issue

• Session hijacking: Attackers can steal session cookies and hijack legitimate user accounts, potentially leading to unauthorized access to sensitive information or systems.
• Data theft: XSS attacks can be used to steal sensitive data such as login credentials, credit card information, and personally identifiable information (PII).
• Malicious redirects: Attackers can redirect users to malicious websites or perform other malicious operations on the user's machine under the guise of the vulnerable site.
• Account compromise: If an attacker gains access to an account with administrative privileges, they can perform unauthorized actions, potentially leading to severe damage to the web application.
• Reputation damage: XSS vulnerabilities can undermine the trust users have in a company, leading to negative publicity and potential loss of customers.

Suggested Countermeasures

• Input validation: Validate and sanitize all user inputs to ensure they do not contain malicious scripts that could be executed on the website.
• Output encoding: Encode user-generated content before displaying it on the website to prevent browsers from interpreting it as executable code.
• Content Security Policy (CSP): Implement a CSP to restrict the sources from which certain types of content can be loaded on your website, reducing the risk of XSS attacks.
• Use security libraries: Utilize security libraries like OWASP ESAPI to help prevent common security vulnerabilities, including XSS attacks.
• Regular security audits: Conduct regular security audits and penetration testing to identify and address any vulnerabilities in your web application.

References

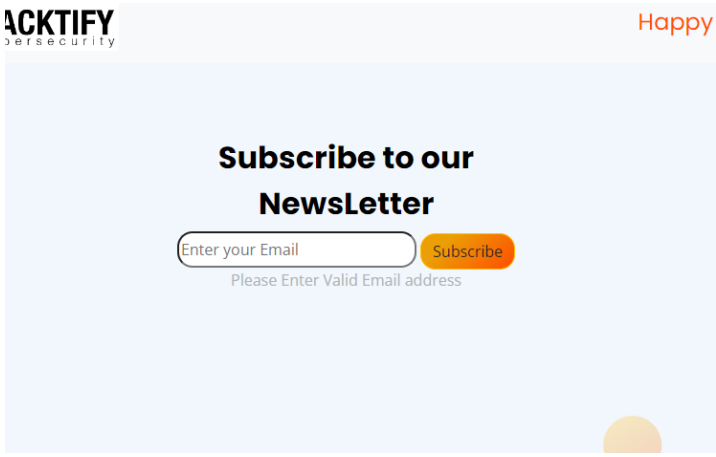
1.3. XSS Is Everywhere!

Reference	Risk Rating
XSS Is Everywhere!	**Low**

Tools Used

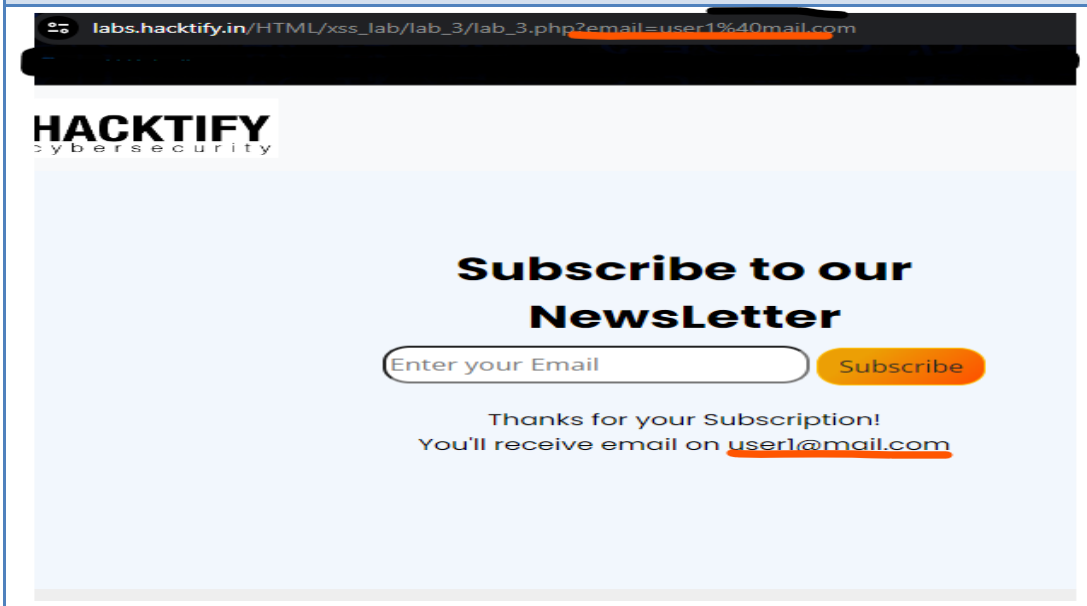
- Payload : user1@mail.com<script> alert("Hacktify Hack u bro hahah!!!") </script>

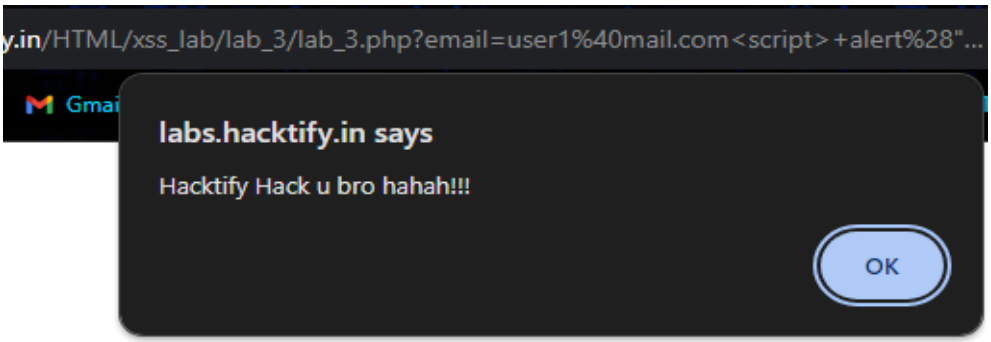
Vulnerability Description



- the warning or response thrown in the response, we can think of one of the major reasons for our payload not working is not the payload itself, but a thing called **input validation**, basically in simple words the backed code is checking the input of the search box that it should be in the form of a email

How It Was Discovered





Vulnerable URLs

- https://labs.hacktify.in/HTML/xss_lab/lab_3/lab_3.php?email=user1%40mail.com%3Cscript%3E+alert%28%22Hacktify+Hack+u+bro+hahah%21%21%21%22%29+%3C%2Fscript%3E

Consequences of not Fixing the Issue

- Session hijacking: Attackers can steal session cookies and hijack legitimate user accounts, potentially leading to unauthorized access to sensitive information or systems.
- Data theft: XSS attacks can be used to steal sensitive data such as login credentials, credit card information, and personally identifiable information (PII).
- Malicious redirects: Attackers can redirect users to malicious websites or perform other malicious operations on the user's machine under the guise of the vulnerable site.
- Account compromise: If an attacker gains access to an account with administrative privileges, they can perform unauthorized actions, potentially leading to severe damage to the web application.
- Reputation damage: XSS vulnerabilities can undermine the trust users have in a company, leading to negative publicity and potential loss of customers.

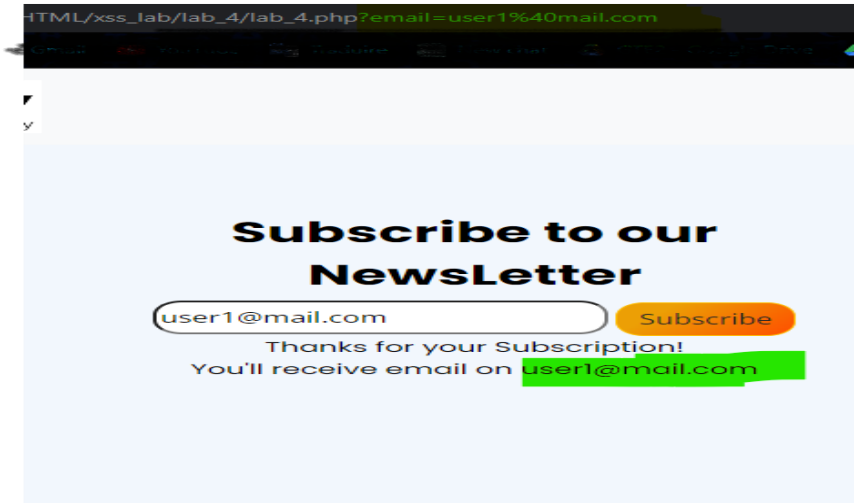
Suggested Countermeasures

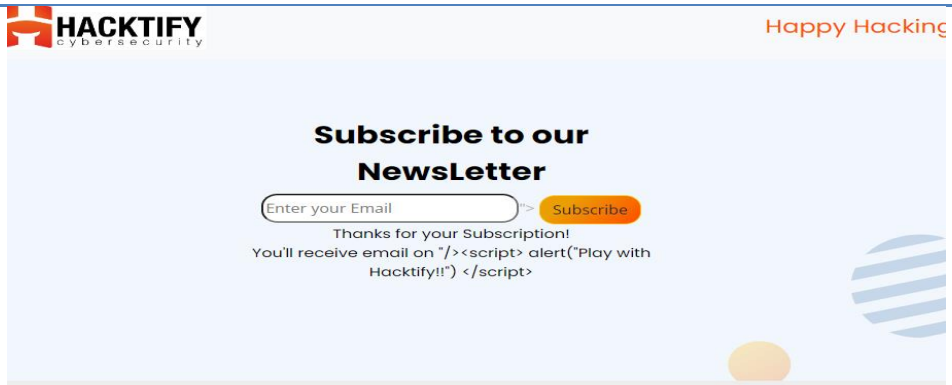
- Output encoding: Encode user-generated content before displaying it on the website to prevent browsers from interpreting it as executable code.
- Content Security Policy (CSP): Implement a CSP to restrict the sources from which certain types of content can be loaded on your website, reducing the risk of XSS attacks.

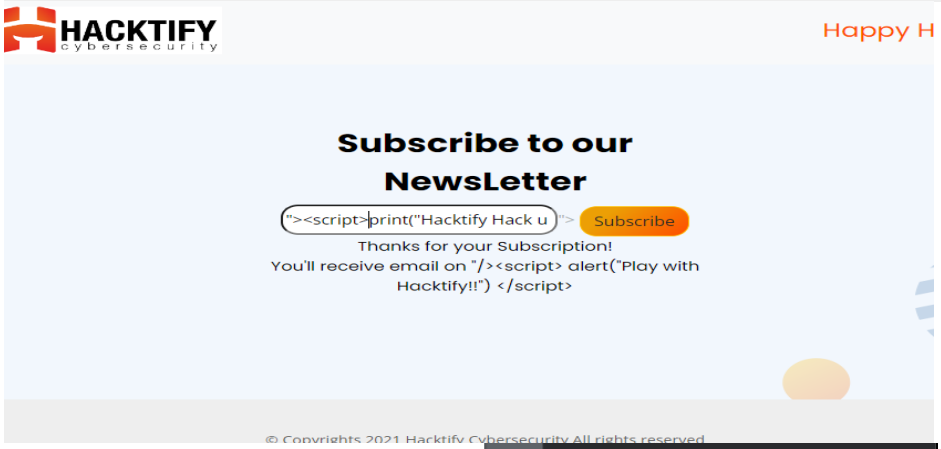
• Use security libraries: Utilize security libraries like OWASP ESAPI to help prevent common security vulnerabilities, including XSS attacks. • Regular security audits: Conduct regular security audits and penetration testing to identify and address any vulnerabilities in your web application
References

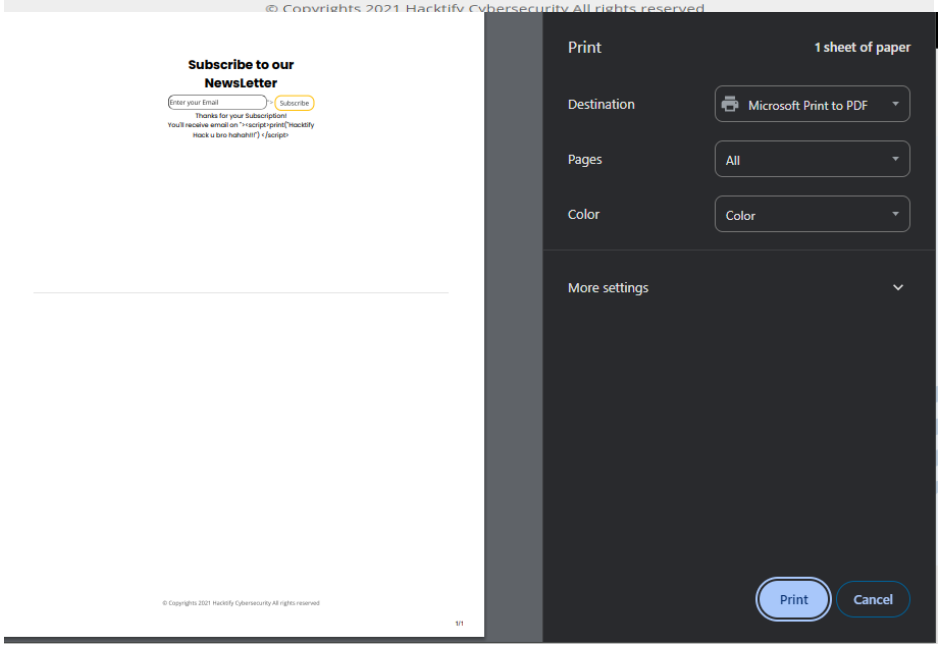
1.4. Alternatives Are Must!

Reference	**Risk Rating**
Alternatives Are Must!	**Medium**
Tools Used	
Payloads	
Vulnerability Description	
• this lab didn't throw us the response of invalid email as the last one we can conclude that no input validation. The alert is being blocked and now when we focus on the name of the lab we can understand what it means by alternative • Payloads: "><script>print("Hacktify Hack u bro hahah!!!") </script>	
How It Was Discovered	









- Payload: "><script>**print**("Hacktify Hack u bro hahah!!!") </script>

Vulnerable URLs

- https://labs.hacktify.in/HTML/xss_lab/lab_4/lab_4.php?email=%22%3E%3Cscript%3Eprint%28%22Hacktify+Hack+u+bro+hahah%21%21%21%22%29+%3C%2Fscript%3E

Consequences of not Fixing the Issue

- The same as the previous ones.

Suggested Countermeasures

- The same as the previous ones.

References

1.5. Developer Hates Scripts!

Reference	Risk Rating
Developer Hates Scripts!	**High**

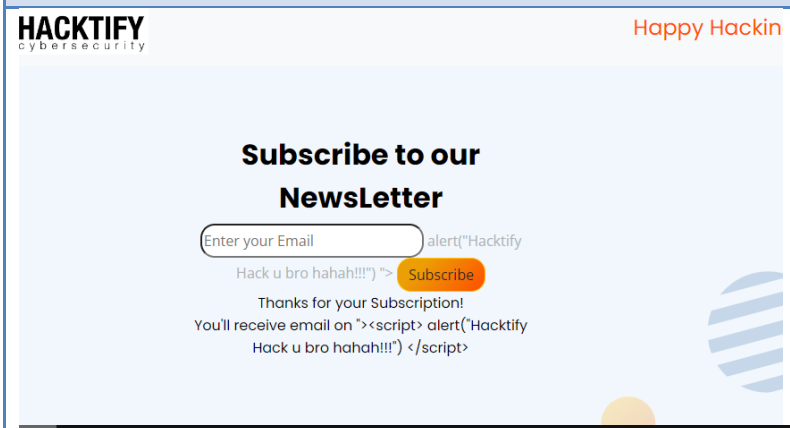
Tools Used

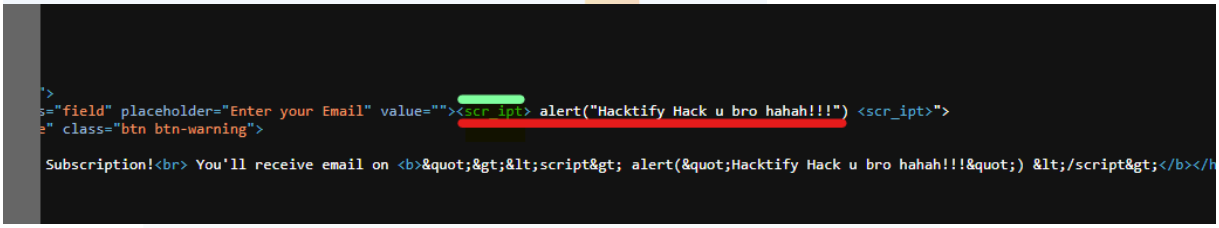
Payload that does not have a script tag in it.

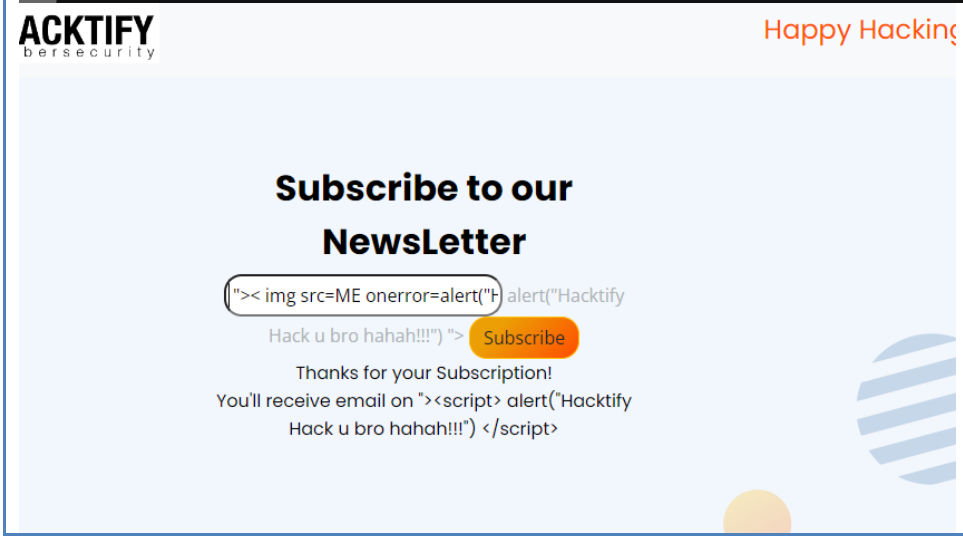
Vulnerability Description

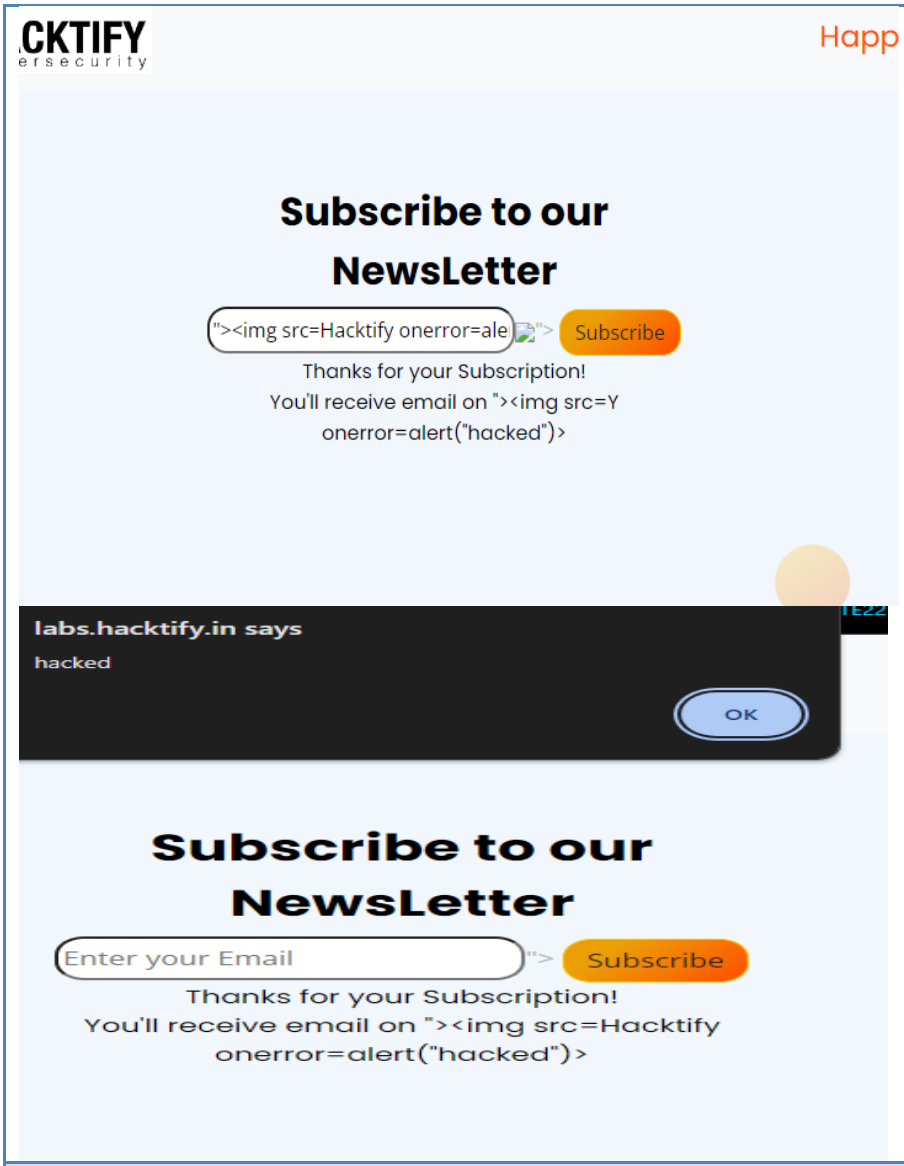
- the '<script>' is being changed to '<scr_ipt>'. we need to find a payload that does not have a script tag in it.

How It Was Discovered









Vulnerable URLs

- https://labs.hacktify.in/HTML/xss_lab/lab_5/lab_5.php?email=%22%3E%3Cimg+src%3DHacktify+onerror%3Dalert%28%22hacked%22%29%3E

Consequences of not Fixing the Issue

- Execution of JavaScript: Removing the script tag from the DOM does not necessarily stop the execution of the JavaScript code contained within it. The script can continue to run even after the tag is removed, as demonstrated in tests where scripts persist and execute despite tag removal.

Suggested Countermeasures

- Comprehensive Input Validation: Implement robust input validation mechanisms that go beyond simple keyword filtering to detect and sanitize potentially malicious inputs, regardless of variations like "scri_pt" tag
- Output Encoding: Apply proper output encoding techniques to all user-generated content before displaying it on the website to prevent script execution and protect against XSS attacks.

1.6. Change The Variation!

Reference	Risk Rating
Change The Variation!	**High**

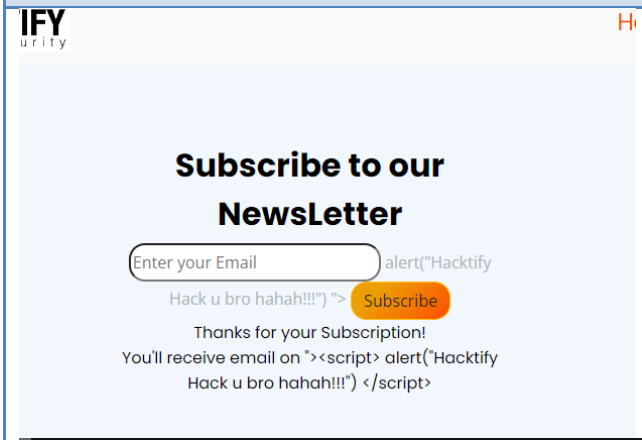
Tools Used

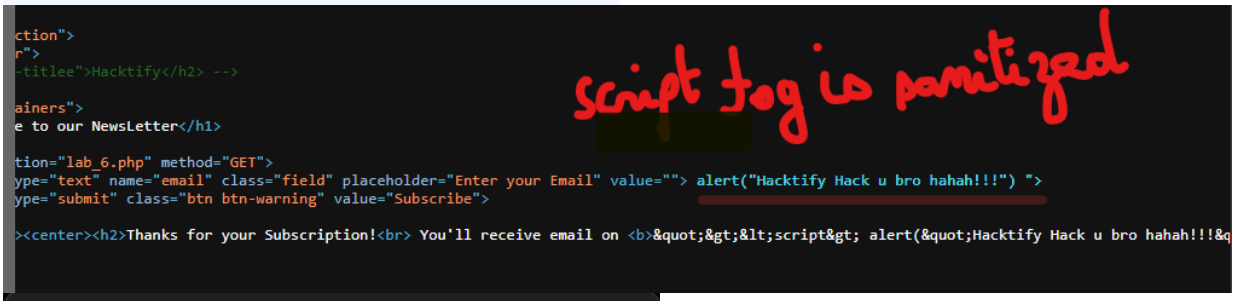
Payload

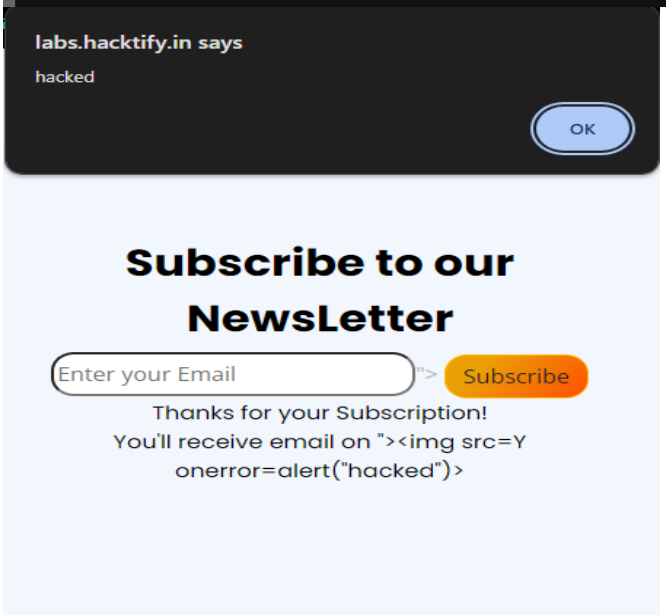
Vulnerability Description

- "<script>" tags are being sanitized in a web application, it means that any script elements within the HTML content are being removed or altered to prevent the execution of potentially malicious scripts.

How It Was Discovered







Vulnerable URLs

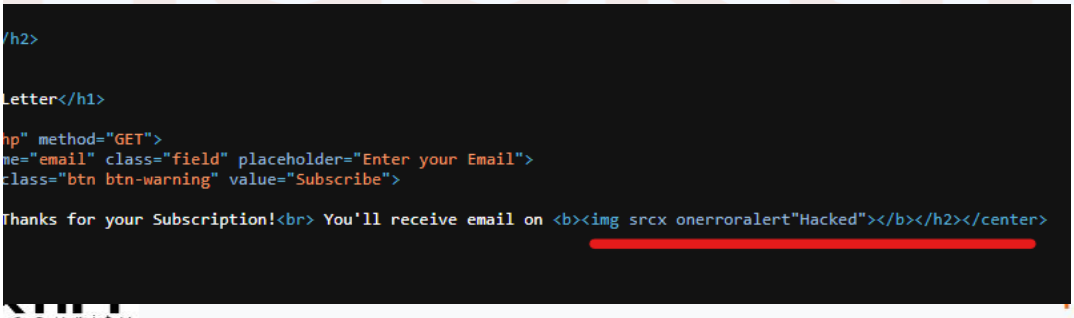
• https://labs.hacktify.in/HTML/xss_lab/lab_6/lab_6.php?email=%22%3E%3Cimg+src%3DY+onerror%3Dalert%28%22hacked%22%29%3E
Consequences of not Fixing the Issue
• Session Hijacking: Attackers can exploit XSS vulnerabilities to steal session cookies, enabling them to hijack user accounts and gain unauthorized access to sensitive information or systems. • Data Theft: XSS attacks can result in the theft of sensitive data like login credentials, credit card details, and personally identifiable information (PII), putting users at risk of identity theft and financial loss. • Account Compromise: The most severe XSS attacks can lead to complete account compromise, allowing attackers to access user accounts, manipulate content, install malware, or redirect users to malicious websites
Suggested Countermeasures
• Enhanced Input Validation: Strengthen input validation processes to detect and sanitize malicious scripts effectively, ensuring that all user inputs are thoroughly validated and sanitized before being processed. • Output Encoding: Apply proper output encoding techniques to all user-generated content to prevent script execution and protect against XSS attacks, even if "<script>" tags have been sanitized. • Content Security Policy (CSP): Implement a robust CSP to restrict the sources from which scripts can be loaded, reducing the risk of unauthorized script execution and enhancing overall web application security.

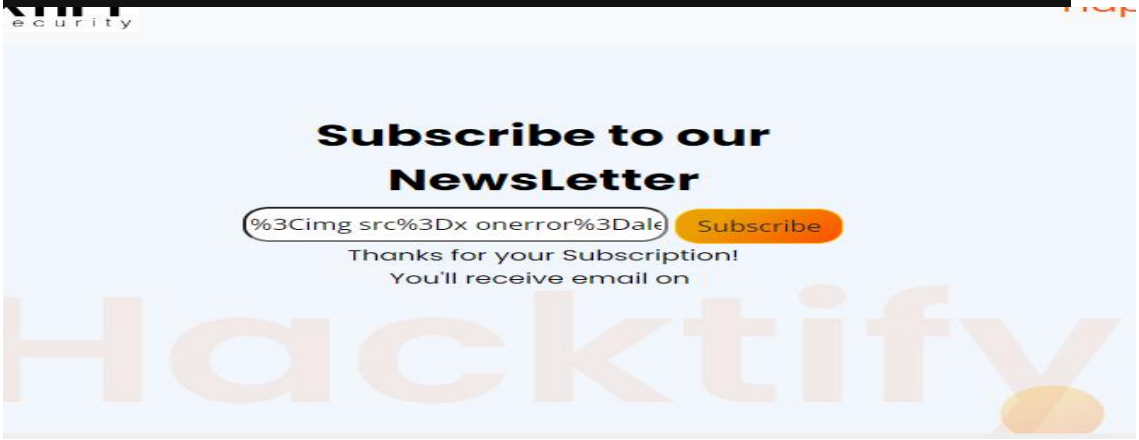
1.7. Encoding Is The Key?

Reference	**Risk Rating**
Encoding Is The Key?	**Medium**
Tools Used	
Payload & URL Encoding	
Vulnerability Description	
• Our payload reflects just in a single position, which is in the body of the page not in the 'input' tag as it does not have a 'value' attribute, also the other thing we can observe is it sanitized our '<', '>', '/' , '=' , '(' and ')'	
How It Was Discovered	


HACKTIFY
cybersecurity
Subscribe to our NewsLetter
%3Cimg src=x onerror=alert("Ha
Subscribe
Thanks for your Subscription!
You'll receive email on Hacked
Subscribe to our NewsLetter
Enter your Email
Subscribe
Thanks for your Subscription!
You'll receive email on


Subscribe to our NewsLetter
%3cu%3eHacked%3c%2fu%3e
Subscribe
Thanks for your Subscription!
You'll receive email on Hacked

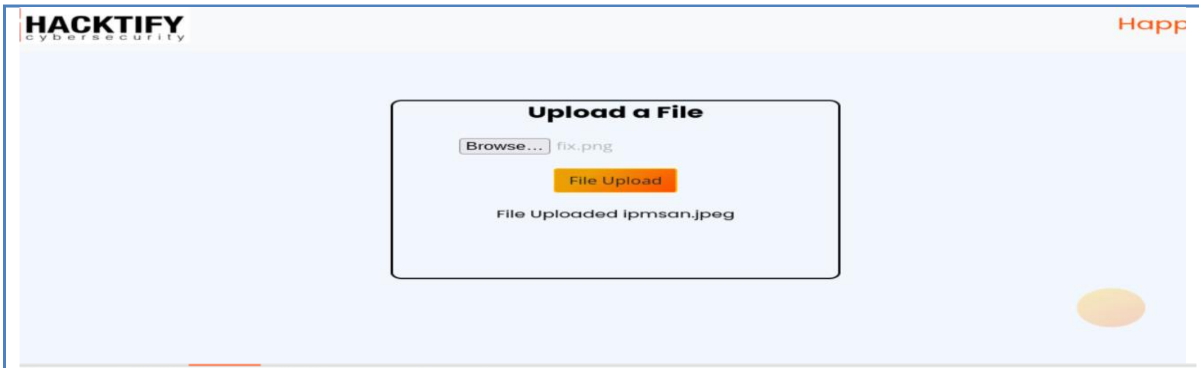

/h2>
Letter</h1>
hp" method="GET">
me="email" class="field" placeholder="Enter your Email">
class="btn btn-warning" value="Subscribe">
Thanks for your Subscription!
 You'll receive email on <img srcx onerroralert"Hacked"></h2>


Subscribe to our NewsLetter
%3Cimg src%3Dx onerror%3Dale
Subscribe
Thanks for your Subscription!
You'll receive email on

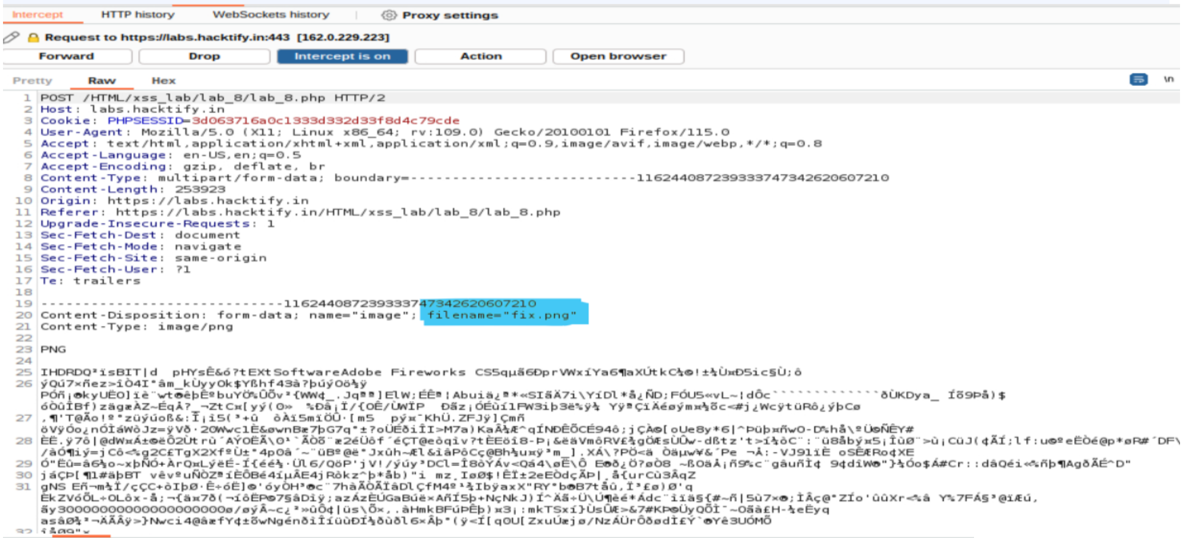
Vulnerable URLs
• https://labs.hacktify.in/HTML/xss_lab/lab_7/lab_7.php?email=%253Cimg+src%253Dx+onerror%253Dalert%2528%22Hacked%22%2529%253E
Consequences of not Fixing the Issue
• The same consequences as the previous one.
Suggested Countermeasures
• The same as the previous one.

1.8. XSS With File Upload (File name)

Reference	Risk Rating
XSS With File Upload (File name)	**Low**
Tools Used	
Burp suite	
Vulnerability Description	
• our parameter is not rendering or processing the close tags and tags are only working before the file name, so we needed a payload that just has an opening tag and this requirement is fulfilled by our 'img' tag.	
How It Was Discovered	

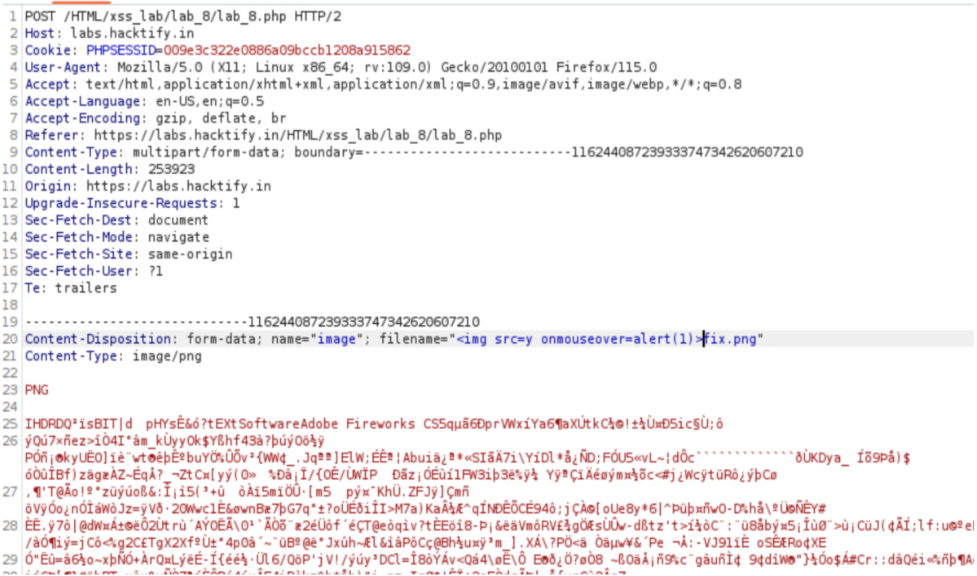


File Uploaded ipmsan.jpeg

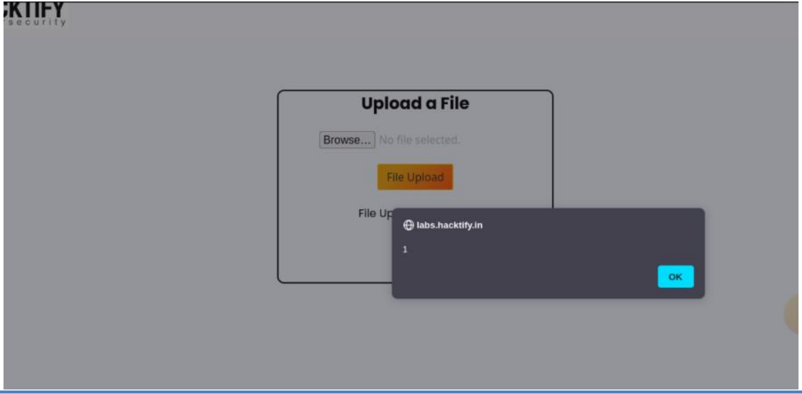


Request to https://labs.hacktify.in:443 [162.0.229.223]

```
POST /HTML/xss_lab/lab_8/lab_8.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=3d063716a0c1333d332d33f8d4c79cde
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: multipart/form-data; boundary=---------------------------116244087239333747342620607210
Content-Length: 253923
Origin: https://labs.hacktify.in
Referer: https://labs.hacktify.in/HTML/xss_lab/lab_8/lab_8.php
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Te: trailers

-----------------------------116244087239333747342620607210
Content-Disposition: form-data; name="image"; filename="fix.png"
Content-Type: image/png

PNG

IHDRDQ²isBIT|d  pHYsÊ&ó?tEXtSoftwareAdobe Fireworks CS5qµã6ÐprVWxíYa6¶aXÚtkC¼®!±¾Ù¤Ð5ic§Ù;ô
```



```
POST /HTML/xss_lab/lab_8/lab_8.php HTTP/2
Host: labs.hacktify.in
Cookie: PHPSESSID=009e3c322e0886a09bccb1208a915862
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://labs.hacktify.in/HTML/xss_lab/lab_8/lab_8.php
Content-Type: multipart/form-data; boundary=---------------------------116244087239333747342620607210
Content-Length: 253923
Origin: https://labs.hacktify.in
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Te: trailers

-----------------------------116244087239333747342620607210
Content-Disposition: form-data; name="image"; filename="<img src=y onmouseover=alert(1)>fix.png"
Content-Type: image/png

PNG

IHDRDQ²isBIT|d  pHYsÊ&ó?tEXtSoftwareAdobe Fireworks CS5qµã6ÐprVWxíYa6¶aXÚtkC¼®!±¾Ù¤Ð5ic§Ù;ô
```

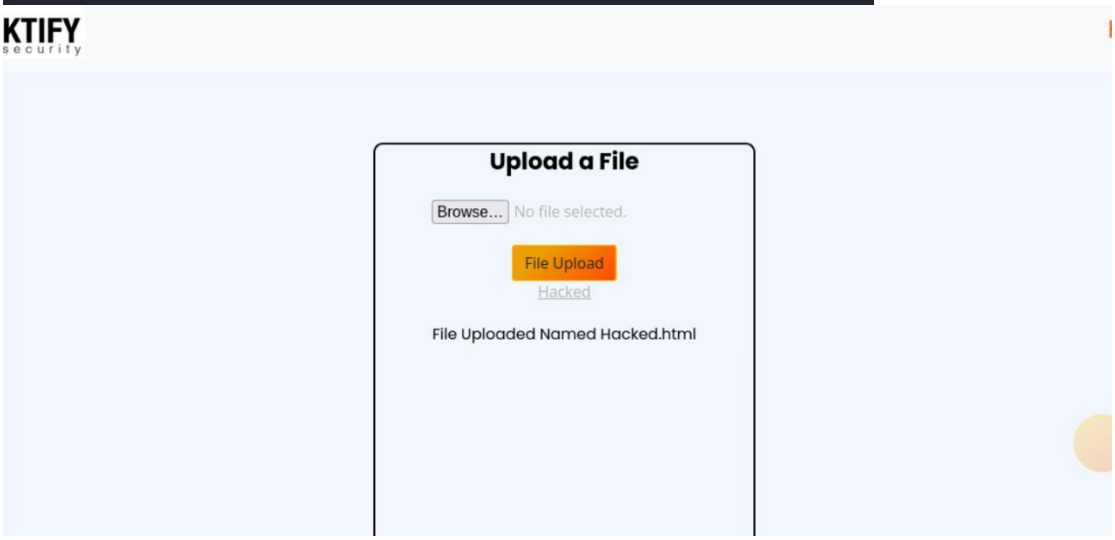


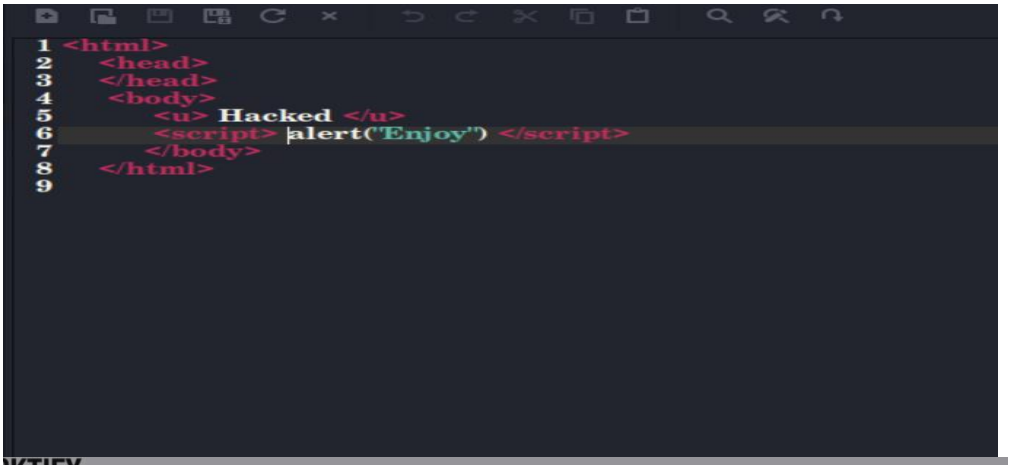
Vulnerable URLs
• https://labs.hacktify.in/HTML/xss_lab/lab_8/lab_8.php
Consequences of not Fixing the Issue
• the same consequences as other sub-labs
Suggested Countermeasures
• the same as other sub-labs

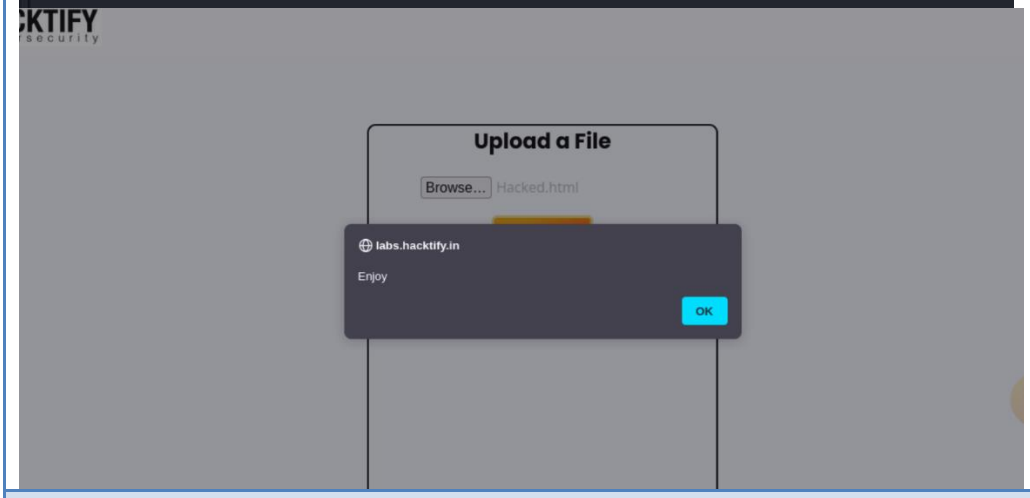
1.9. XSS With File Upload (File Content)

Reference	Risk Rating
XSS With File Upload (File Content)	**Medium**
Tools Used	
HTML Injection	
Vulnerability Description	
• the file content can be an attack vector to find the XSS.	
How It Was Discovered	

```
1 <html>
2   <head>
3   </head>
4   <body>
5   <u> Hacked </u>
6   </body>
7   </html>
8
```







Vulnerable URLs

- https://labs.hacktify.in/HTML/xss_lab/lab_9/lab_9.php

Consequences of not Fixing the Issue

- Data Theft: XSS vulnerabilities can lead to the theft of sensitive data, such as session cookies, user credentials, and personal information, allowing attackers to access and misuse this data.
- Identity Impersonation: Attackers can exploit XSS vulnerabilities to impersonate users, gaining unauthorized access to accounts and performing actions on behalf of the victim without their consent.
- Website Defacement: Unaddressed XSS vulnerabilities can result in website defacement, where attackers modify the appearance and content of a website to spread malicious messages or misinformation.

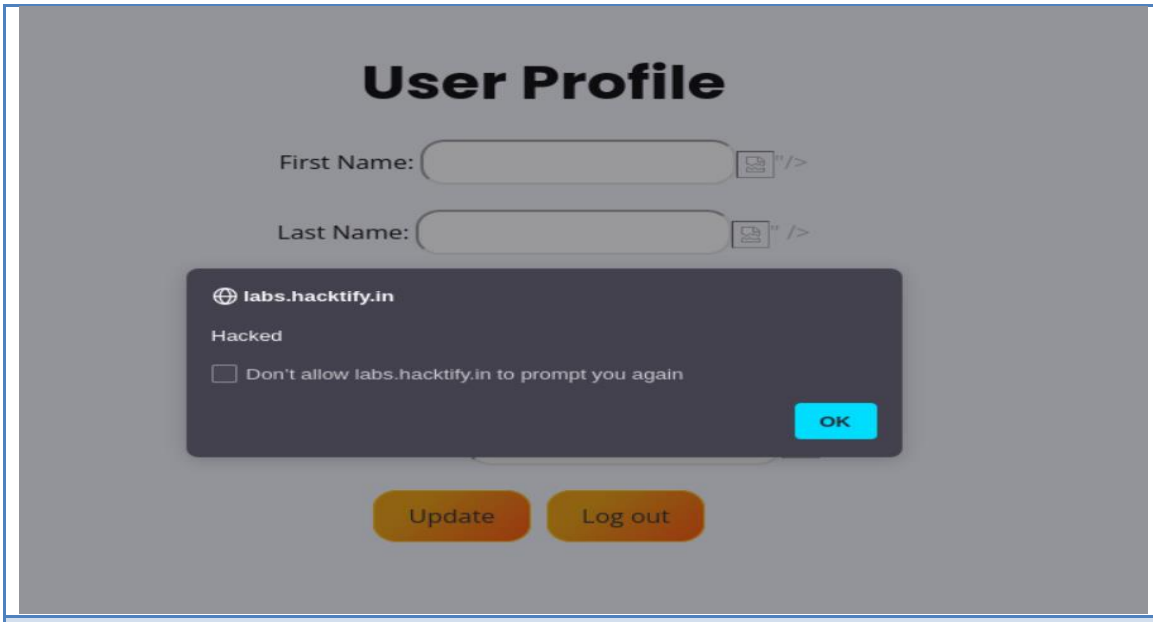
Suggested Countermeasures

- HTTPOnly and Secure Flags for Cookies: Set the HTTPOnly flag on cookies to prevent client-side scripts from accessing them and use the Secure flag to ensure that cookies are only transmitted over secure HTTPS connections.

1.10. Stored Everywhere!

Reference	Risk Rating
Stored Everywhere!	**Low**
Tools Used	
Payload	
Vulnerability Description	
• Our lab had stored XSS vulnerability in it.	
How It Was Discovered	

- As we used 'onmouseover' function we got the following result when we hovered our mouse pointer over the image icons,



Vulnerable URLs
• https://labs.hacktify.in/HTML/xss_lab/lab_10/profile.php
Consequences of not Fixing the Issue
• The same as other Labs
Suggested Countermeasures
• The same as other Labs

1.11. DOM's are love!

Reference	Risk Rating
DOM's are love!	**High**
Tools Used	
Payloads	
Vulnerability Description	
• Our lab is vulnerable to XSS for '?name=' and '?coin=' parameters and it is also vulnerable to open redirect on its '?redir=' parameter as it enables the attacker to craft and redirect to a random web page.	
How It Was Discovered	
I found a file named 'dom.js' as observed before and as we can look into the file, we can locate quite a few parameters to play with, listed as ?name= ?redir= ?coin= I try to play with this parameters and that’s what I get :	

HACKTIFY cybersecurity

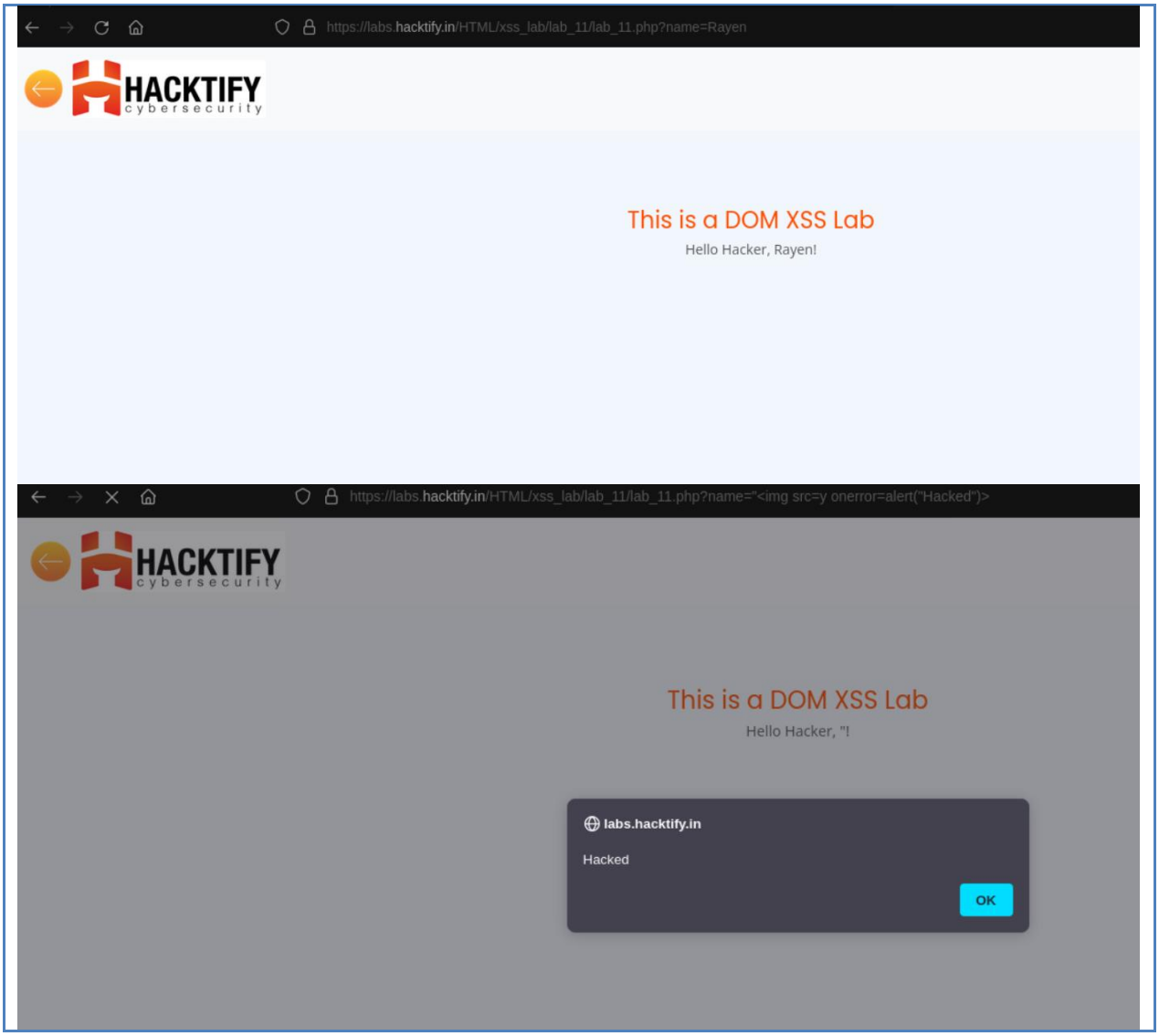
Happy Hacking

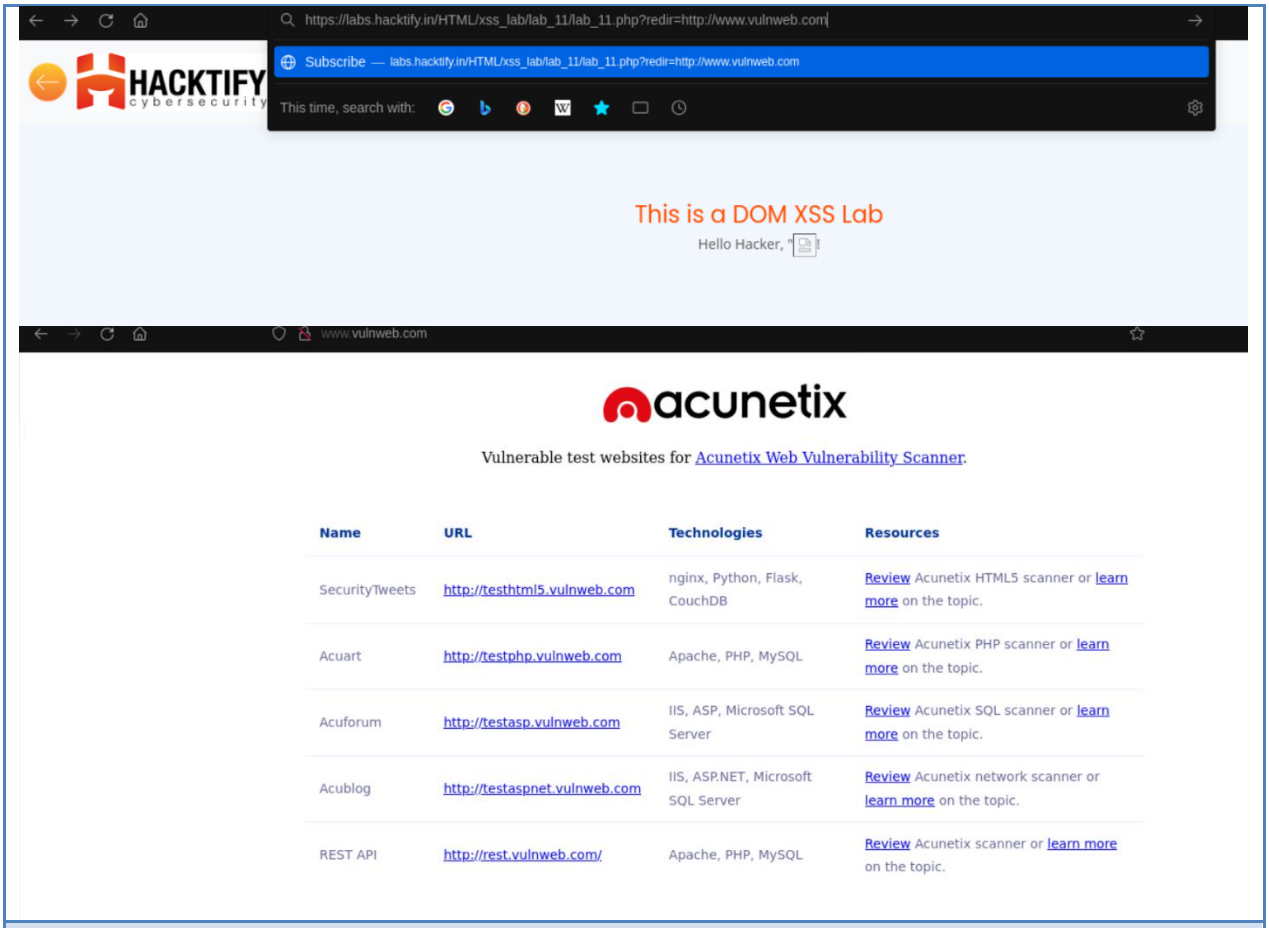
This is a DOM XSS Lab

Hello,__!

```
            </li>
            </ul>
        </div>
    </nav>
    <section class="pager-section">
        <div class="container">
        <!-- <h2 class="page-titlee">Hacktify</h2> -->
          <center>
            <div class="containers">
            <h3>This is a DOM XSS Lab</h3>
            <p id="p1">Hello,__!</p>
            <script src="dom.js"></script>
            </div>
          </center>
      <header>
    </div>
    </section>
    <hr />
      <div class="footer">
          <p>© Copyrights 2021 Hacktify Cybersecurity All rights reserved</p>
      </div>
    </div>
```

```
var currentSearch = document.location.search;
var searchParams = new URLSearchParams(currentSearch);

/*** Document Sink ***/

var username = searchParams.get("name");

if (username !== null) {
  document.getElementById("p1").innerHTML = "Hello Hacker, " + username + "!";
}

/*** Location Sink ***/

var redir = searchParams.get("redir");

if (redir !== null) {
  document.location = redir;
}

/*** Execution Sink ***/

var btc = "$40000";
var eth = "$2000";
var doge = "$1";

var market = [];
var coin = searchParams.get("coin").toLowerCase();
if (coin !== null) {
  document.getElementById("p1").innerHTML =
    "The 3 coins are btc, eth, doge, " + coin + "!";
}

eval("market.coin=" + coin);

document.getElementById("p1").innerHTML =
  "Current Hyped coin is " + market.coin + ".";
```


https://labs.hacktify.in/HTML/xss_lab/lab_11/lab_11.php?name=Rayen
HACKTIFY
cybersecurity
This is a DOM XSS Lab
Hello Hacker, Rayen!
https://labs.hacktify.in/HTML/xss_lab/lab_11/lab_11.php?name="<img src=y onerror=alert("Hacked")>
HACKTIFY
cybersecurity
This is a DOM XSS Lab
Hello Hacker, "!
labs.hacktify.in
Hacked
OK



Vulnerable URLs

- https://labs.hacktify.in/HTML/xss_lab/lab_11/lab_11.php

Consequences of not Fixing the Issue

- Session Hijacking: Attackers can steal session cookies through XSS attacks, allowing them to impersonate users and gain unauthorized access to their accounts.
- Credential Theft: XSS vulnerabilities can lead to the theft of sensitive information like usernames, passwords, bank account numbers, and personally identifiable information (PII).
- Data Disclosure: Attackers can disclose end-user files, install Trojan horse programs, redirect users to malicious sites, or modify the presentation of content through XSS attacks.

Suggested Countermeasures

- Web Application Firewalls (WAF): Deploy WAF solutions to monitor and filter incoming traffic, detecting and blocking malicious payloads that could exploit XSS vulnerabilities.
- Security Headers: Implement security headers like X-XSS-Protection, X-Content-Type-Options, and X-Frame-Options to enhance browser security and protect against various types of attacks, including XSS.

References

- https://security.stackexchange.com/questions/206520/how-dangerous-is-xss

2. IDOR

2.1. Give me my amount!!

Reference	Risk Rating
Give me my amount!!	**Low**
Tools Used	
• Changing id	
Vulnerability Description	
• in this URL as marked, we can see the "/profile.php?id=" id parameter, and that too has a numerical value telling us a lot about the backend architecture such as the database at the backend where user profiles are created would have following fields (id, Email, Password, Transaction 1, Transaction 2, Transaction 3), and my user is in the 528 row or is the 528 user.	
How It Was Discovered	
• Changing user id	
Vulnerable URLs	
• https://labs.hacktify.in/HTML/idor_lab/lab_1/profile.php?id=52	
Consequences of not Fixing the Issue	
• Unauthorized Data Access: Attackers can exploit IDOR vulnerabilities to access sensitive information belonging to other users by manipulating the object references in URLs, potentially leading to data breaches and privacy violations. • Privilege Escalation: IDOR vulnerabilities can result in both horizontal and vertical privilege escalation, allowing attackers to gain unauthorized access to resources or perform actions beyond their intended privileges within the application. • Data Manipulation: Attackers may modify or delete critical data by exploiting IDOR vulnerabilities, leading to data corruption, financial losses, or disruption of services. • Account Takeover: IDOR vulnerabilities can facilitate account takeovers, enabling attackers to impersonate users, perform unauthorized actions on their behalf, or compromise sensitive accounts with elevated privileges.	
Suggested Countermeasures	
• Web Application Firewalls (WAF): Deploy WAF solutions to monitor and filter incoming traffic, detecting and blocking malicious payloads that could exploit IDOR vulnerabilities. • Security Headers: Implement security headers like X-XSS-Protection, X-Content-Type-Options, and X-Frame-Options to enhance browser security and protect against various types of attacks, including XSS.	

2.2. Stop polluting my params!

Reference	Risk Rating
Stop polluting my params!	**Medium**

Tools Used

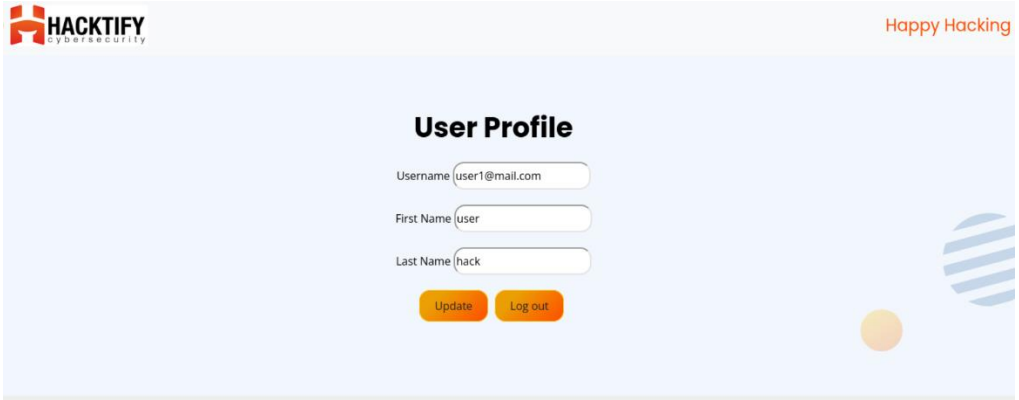
- HTTP Parameter Pollution

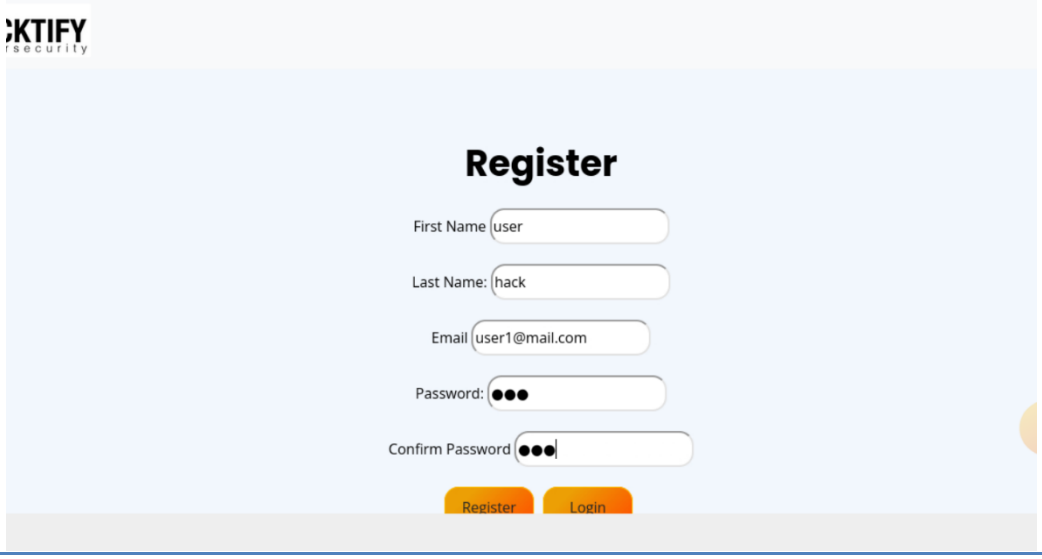
Vulnerability Description

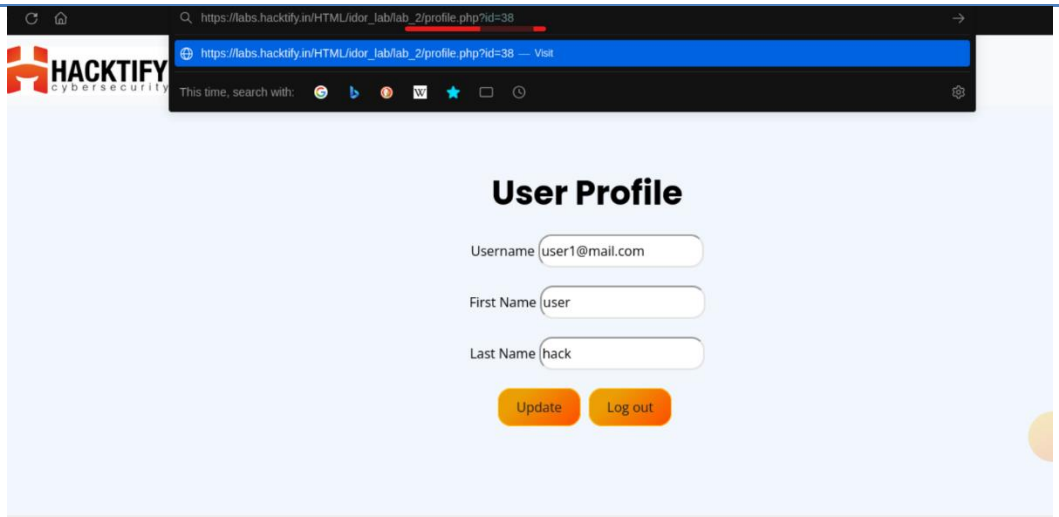
- HTTP Parameter Pollution (HPP) is a type of injection attack where an attacker manipulates existing HTTP parameters to trick the application into performing unintended actions. This technique can be used to override existing hardcoded HTTP parameters, modify application behavior, access and potentially exploit uncontrolled variables, and bypass input validation mechanisms and WAF rules

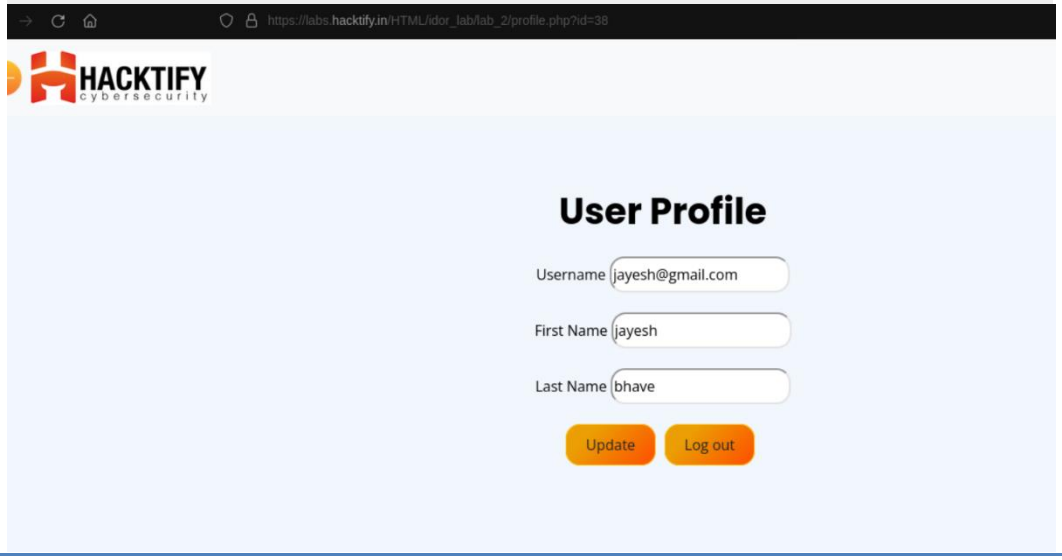
How It Was Discovered

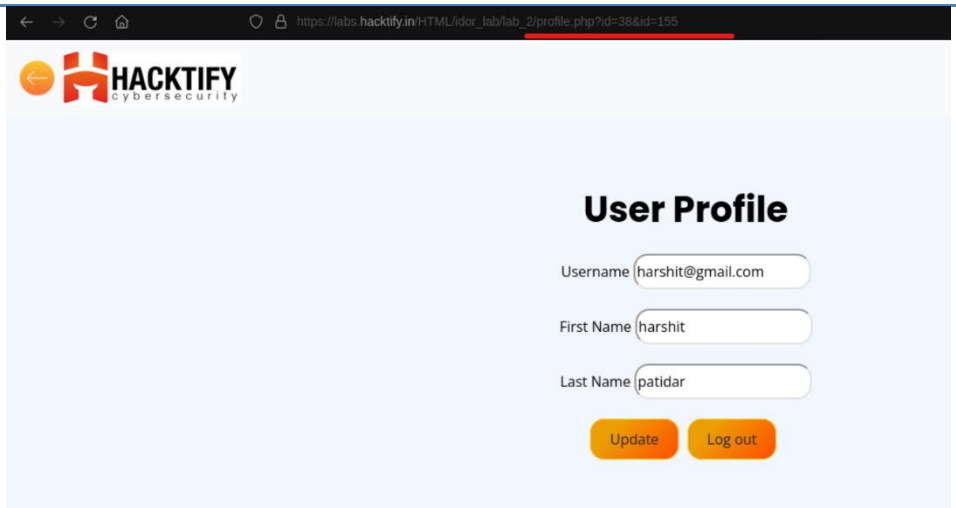
- Modifying id numbers

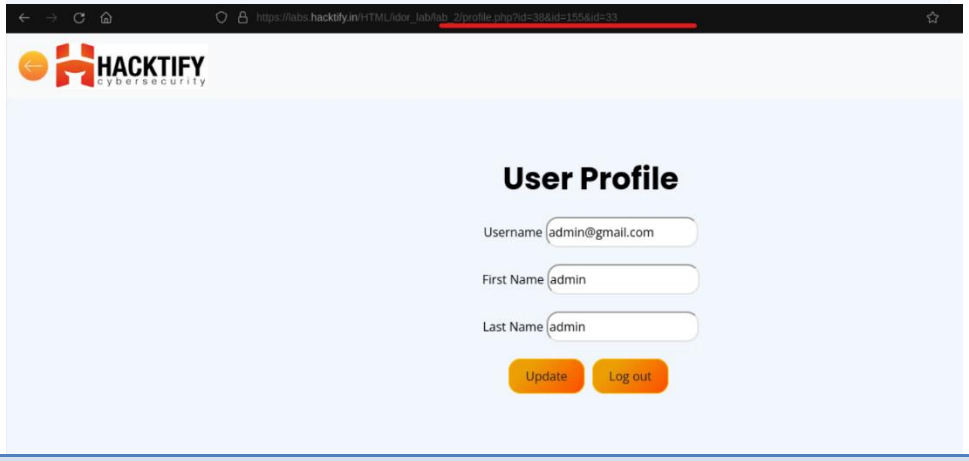





https://labs.hacktify.in/HTML/idor_lab/lab_2/profile.php?id=38
https://labs.hacktify.in/HTML/idor_lab/lab_2/profile.php?id=38 — Visit
This time, search with:
HACKTIFY
cybersecurity
User Profile
Username user1@mail.com
First Name user
Last Name hack
Update
Log out


https://labs.hacktify.in/HTML/idor_lab/lab_2/profile.php?id=38
HACKTIFY
cybersecurity
User Profile
Username jayesh@gmail.com
First Name jayesh
Last Name bhave
Update
Log out





Vulnerable URLs

- https://labs.hacktify.in/HTML/idor_lab/lab_2/profile.php?id=38&id=33

Consequences of not Fixing the Issue

- Data Integrity Issues: HPP can result in data corruption or manipulation, affecting the accuracy and reliability of information processed by the application.
- Security Breaches: Exploiting HPP vulnerabilities can enable attackers to bypass security controls, access unauthorized resources, and potentially compromise sensitive data.

Suggested Countermeasures

- Input Validation: Implement thorough input validation mechanisms to ensure that user-supplied data is sanitized and validated before processing, preventing malicious payloads from manipulating HTTP parameters.
- Parameter Whitelisting: Utilize parameter whitelisting to define and restrict the acceptable values for each parameter, allowing only authorized inputs and rejecting any unauthorized or unexpected values.
- Avoid Parameter Duplication: Avoid scenarios where the same parameter can be duplicated in a request, as this can lead to ambiguity and potential exploitation by attackers manipulating the order or presence of parameters.

References

- https://www.imperva.com/learn/application-security/insecure-direct-object-reference-idor/

- Unfortunately, I had an issue while I try to login after registering, but the principle of the labs is to intercept while changing password using Burp suite and try to generate a New password. My idea is to create 2 accounts with different passwords and try to switch between them.

In conclusion, addressing XSS and IDOR vulnerabilities requires a comprehensive approach that includes proactive testing, adherence to best practices, continuous monitoring, and swift remediation of identified issues. By following recommended mitigation strategies and staying informed about evolving threats, organizations can enhance the security posture of their web applications and protect user data from malicious exploitation.