

XSS Attack Lab

This document explains the implementation and demonstration of CrossSite Scripting (XSS) vulnerabilities using a Node.js and Express based lab environment. The project shows Stored XSS, Reflected XSS, DOM Based XSS and a Secure version to demonstrate prevention techniques.

Stored XSS Demonstration

Stored XSS occurs when user input is stored on the server and executed whenever the page loads. In this lab, comments are saved and rendered without sanitization, allowing scripts to execute.

The screenshot shows the 'Stored XSS Lab' interface. On the left, a sidebar lists 'XSS LAB' with options: 'XSS (Stored)', 'XSS (Reflected)', 'XSS (DOM)', and 'Secure Version'. The main area has a dark background with a light blue header 'Stored XSS Lab'. Below it, a text box says: 'This page demonstrates Stored Cross-Site Scripting where user input is saved on the server and executed whenever the page loads.' A text input field contains 'Enter your comment' and a blue 'Post Comment' button. Underneath, a 'Comments' section shows a single entry: '('. At the bottom, there's a 'Try Payloads' section with three items, a 'More Information' section with three links, and a footer 'Cybersecurity Mini Project | Kritika'.

The screenshot shows the 'Stored XSS Lab' interface after posting a malicious payload. The main area now displays the posted payload: '<script>alert("Stored XSS")</script>'. The 'Comments' section shows the output of the payload: '('. The rest of the interface remains the same, including the sidebar and footer.

localhost:3000 says

Stored XSS

OK

XSS LAB

- XSS (Stored)
- XSS (Reflected)
- XSS (DOM)
- Secure Version

Stored XSS Lab

This page demonstrates Stored Cross-Site Scripting where user input is saved on the server and executed whenever the page loads.

Enter your comment

Post Comment

Comments

{
{
{
{ hello

Try Payloads

- <script>alert('Stored XSS')</script>
- <h1 style="color:red">HACKED</h1>
-

More Information

- [OWASP XSS Guide](#)
- [PortSwigger Stored XSS](#)
- [MDN XSS Docs](#)

- XSS (Reflected)
- XSS (DOM)
- Secure Version

This page demonstrates Stored Cross-Site Scripting where user input is saved on the server and executed whenever the page loads.

Enter your comment

Post Comment

Comments

{
{
{ hello

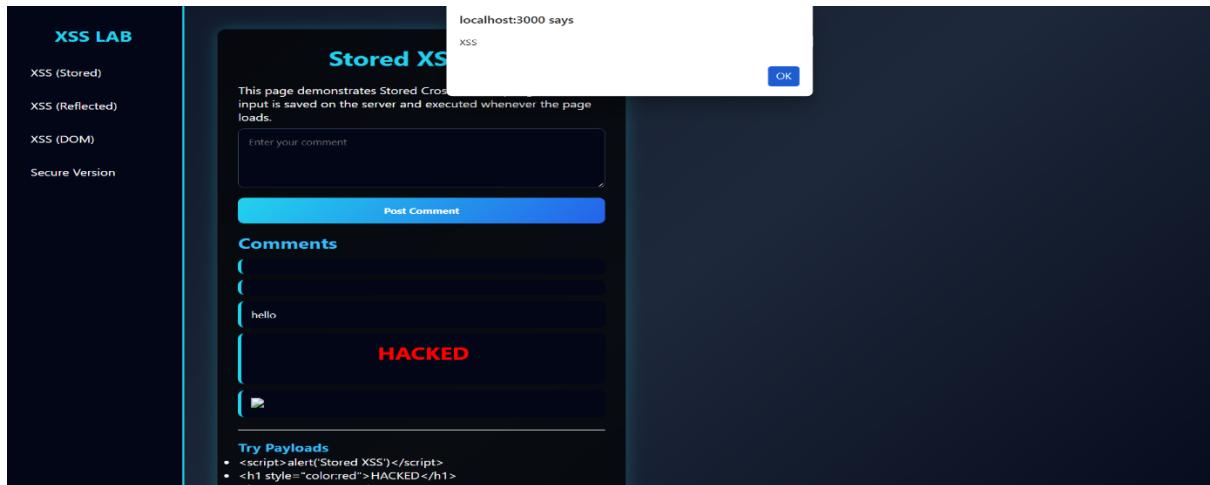
HACKED

Try Payloads

- <script>alert('Stored XSS')</script>
- <h1 style="color:red">HACKED</h1>
-

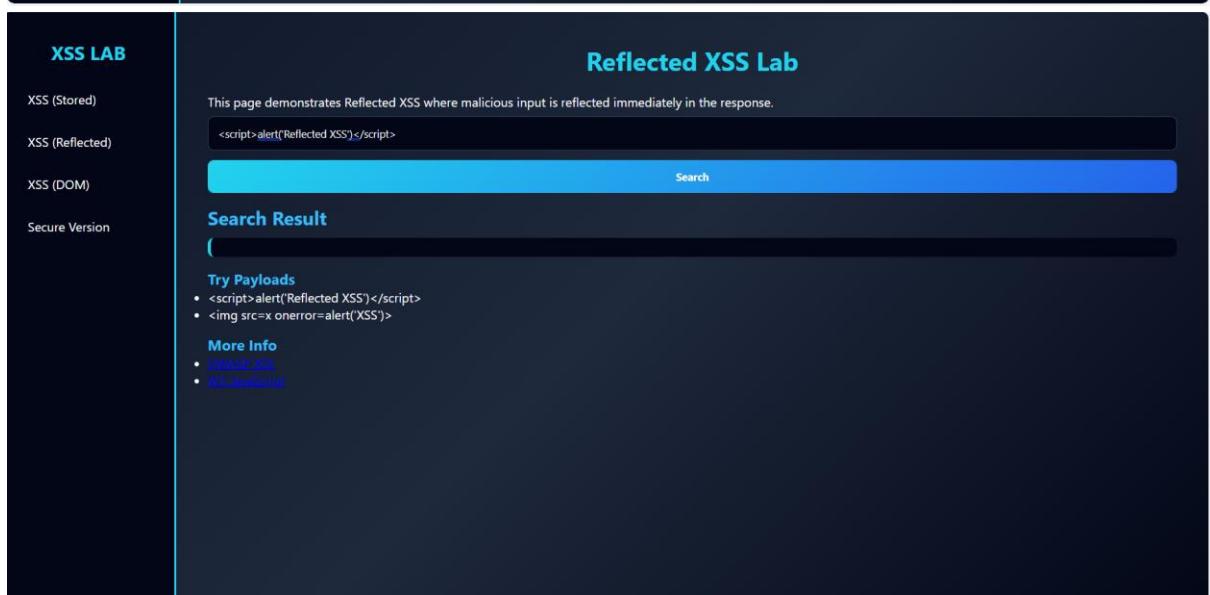
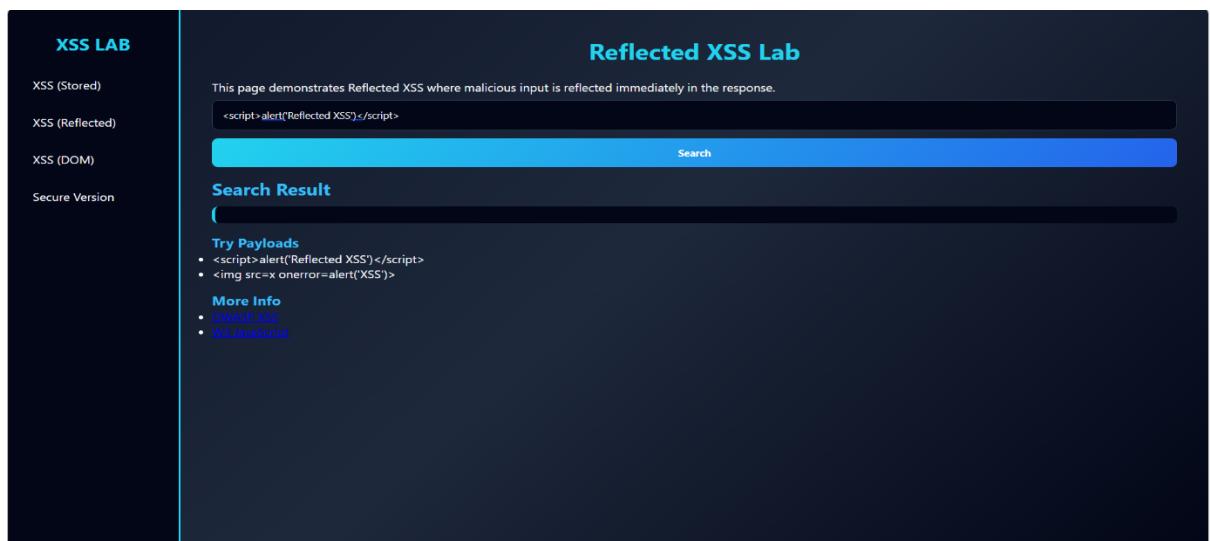
More Information

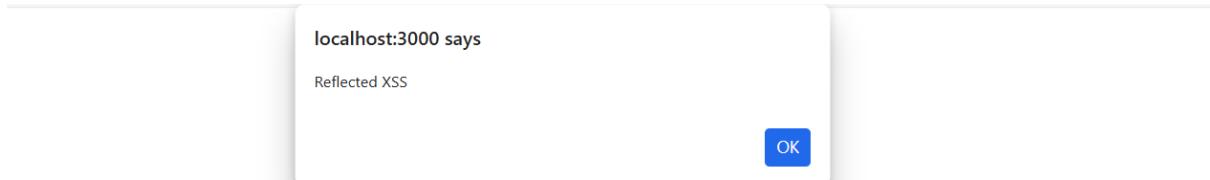
- [OWASP XSS Guide](#)
- [PortSwigger Stored XSS](#)
- [MDN XSS Docs](#)



Reflected XSS Demonstration

Reflected XSS happens when user input is immediately returned in the response. The attack runs instantly when a crafted input is submitted





XSS LAB

This page demonstrates Reflected XSS where malicious input is reflected immediately in the response.

```
<img src=x onerror=alert('XSS')>
```

Search

Search Result

(

Try Payloads

- <script>alert('Reflected XSS')</script>
-

More Info

- [OWASP XSS](#)
- [W3Schools XSS](#)

XSS LAB

This page demonstrates Reflected XSS where malicious input is reflected immediately in the response.

```
Search here
```

localhost:3000 says

XSS

OK

b

Search

Search Result

(

Try Payloads

- <script>alert('Reflected XSS')</script>
-

More Info

- [OWASP XSS](#)
- [W3Schools XSS](#)

DOM XSS Demonstration

DOM Based XSS occurs entirely on the client side when JavaScript dynamically modifies page content using unsafe user input

The screenshot shows the XSS LAB interface. On the left, a sidebar lists "XSS (Stored)", "XSS (Reflected)", "XSS (DOM)", and "Secure Version". The main area is titled "DOM Based XSS Lab" and contains the following content:

This page demonstrates DOM-Based XSS using client-side JavaScript.

```
<img src=x onerror=alert('DOM XSS')>
```

Submit

Output:

(

Try Payloads

-
- <script>alert('DOM')</script>

The screenshot shows the XSS LAB interface. On the left, a sidebar lists "XSS (Stored)", "XSS (Reflected)", "XSS (DOM)", and "Secure Version". The main area is titled "DOM Based XSS Lab" and contains the following content:

This page demonstrates DOM-Based XSS using client-side JavaScript.

Enter message here

Submit

Output:

(

Try Payloads

-
- <script>alert('DOM')</script>

XSS LAB

XSS (Stored)
XSS (Reflected)
XSS (DOM)
Secure Version

DOM Based XSS Lab

This page demonstrates DOM-Based XSS using client-side JavaScript.

```
<script>alert('DOM')</script>
```

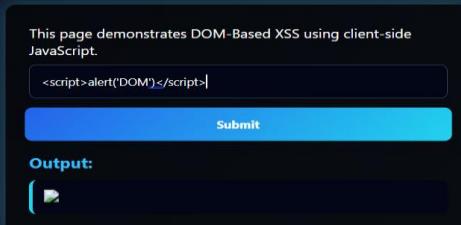
Submit

Output:

[]

Try Payloads

-
- <script>alert('DOM')</script>



Secure Version Demonstration

The secure page shows how output encoding and safe rendering prevent script execution

XSS LAB

XSS (Stored)
XSS (Reflected)
XSS (DOM)
Secure Version

Secure XSS Prevention Lab

This page demonstrates how XSS can be prevented using escaped output rendering.

Enter your comment

Post Secure Comment

Secure Comments

[]

Prevention Techniques

- Input Validation
- Output Encoding
- Content Security Policy (CSP)
- HttpOnly Cookies

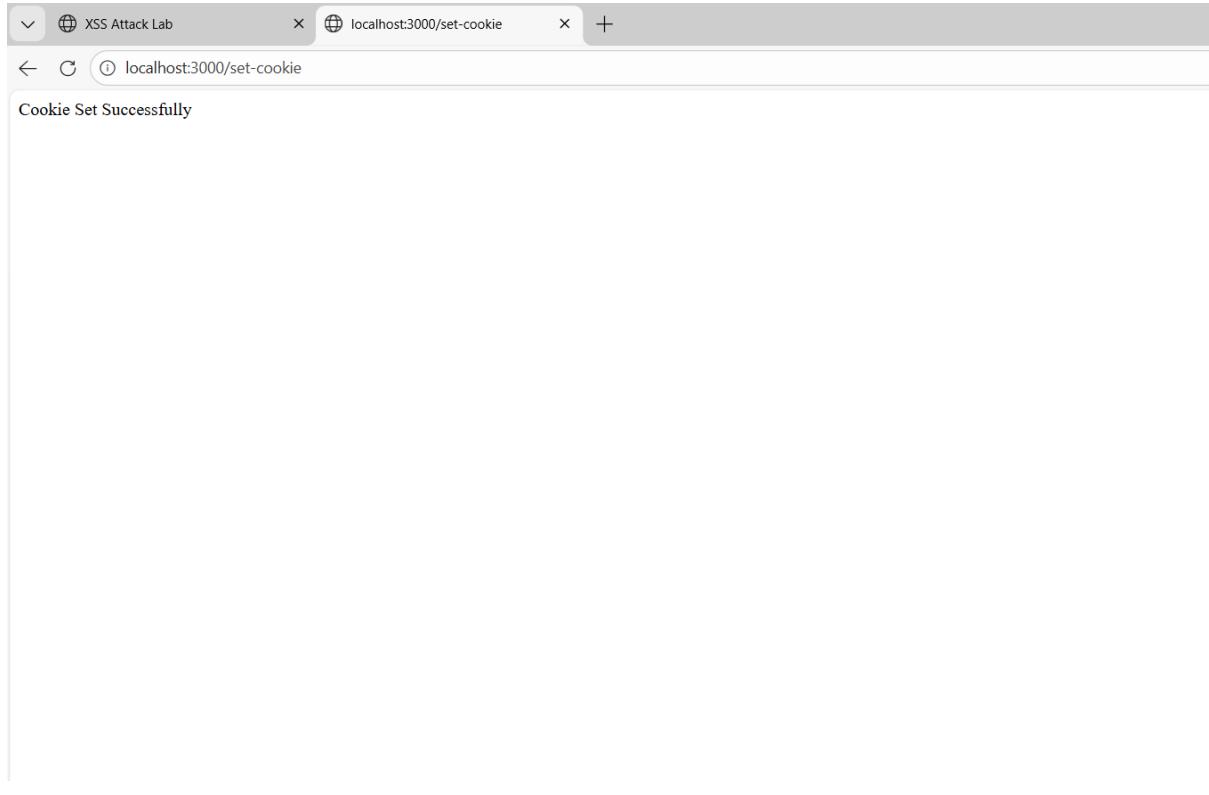
Learn More

- OWASP XSS-attacks
- MDN XSS



Cookie / Session Exposure Demonstration

This section demonstrates how XSS can expose cookies using document.cookie. A demo cookie is set and then accessed via JavaScript payload.



XSS LAB

XSS (Stored)
XSS (Reflected)
XSS (DOM)
Secure Version

Stored XSS Lab

This page demonstrates Stored Cross-Site Scripting where user input is saved on the server and executed whenever the page loads.

```
<script>alert(document.cookie)</script>
```

Post Comment

Comments

Try Payloads

- <script>alert('Stored XSS')</script>
- <h1 style="color:red">HACKED</h1>
-

More Information

- [OWASP XSS Guide](#)
- [PortSwigger Stored XSS](#)
- [MDN XSS Docs](#)

Cybersecurity Mini Project | Kritika

localhost:3000 says

sessionId=KRITIKA123

OK