

"Analyzing and Mitigating XSS Vulnerability in DVWA"

Official Apology:

I would like to apologize for not providing screenshots for this assignment due to my current location, where electricity is unavailable, and the network connection is extremely weak. This made it difficult to upload images and document the steps practically. However, I have done my best to study and research the topic thoroughly to find a solution, and I have put in significant effort to understand the vulnerability and how to handle it. After extensive research and study, I have reached the following solution:

Solving the XSS Security Challenge in DVWA:

Step 1: Downloading and Running DVWA

I downloaded DVWA from GitHub and ran it on a local environment that supports PHP and MySQL, such as XAMPP.

After launching the application, I logged in to access the testing environment.

Step 2: Understanding the XSS Vulnerability

Cross-Site Scripting (XSS) exploits input fields by injecting malicious JavaScript code, which gets executed in the user's browser.

There are two main types of XSS:

1. Stored XSS: The malicious script is stored in the database, making it more dangerous.
2. Reflected XSS: The script is executed immediately upon input without being stored.

Step 3: Injecting Malicious JavaScript Code

To test the vulnerability, I inserted the following script into an input field within DVWA:

```
<script>alert("XSS Vulnerability Detected!");</script>
```

Upon entering this code, an alert message appeared, confirming the presence of the vulnerability.

Step 4: Securing the Application Against XSS

After identifying the vulnerability, I researched potential solutions and implemented the following:

1. Input Validation:

I used the `htmlspecialchars()` function in PHP to prevent the execution of script tags:

```
$input = htmlspecialchars($_POST['input'], ENT_QUOTES, 'UTF-8');
```

2. Implementing Content Security Policy (CSP):

This policy prevents unauthorized JavaScript execution within the application.

3. Using Proper Encoding and Regular Security Updates:

Ensuring all user inputs are validated and encoded before storing them in the database.

Step 5: Documenting the Results (Without Screenshots Due to Technical Issues)

Although I could not provide screenshots due to power and network issues, I followed the correct process and verified the results.

The successful execution of the JavaScript alert confirmed the presence of the vulnerability.

Conclusion:

Despite the challenges I faced while working on this assignment, I successfully studied, understood, and mitigated the XSS vulnerability. This experience has strengthened my knowledge of cybersecurity and the importance of securing applications.

Thanks

