ABSTRACT

This project is about Cross-Site Scripting (XSS) attacks, a common and serious web application security vulnerability. XSS enables attackers to insert malicious scripts into websites, exposing user data and web page security. Using the Damn Vulnerable Web Application (DVWA), a deliberately vulnerable application, the project illustrates how XSS attacks in different types take place and how they can be detected and mitigated.

**Problem Description and Summary**

Web applications typically take user input and output them without proper validation or modification. This can result in XSS attacks. XSS attacks allow attackers to steal sensitive information like cookies or session tokens and alter web content, which poses a threat to users and organizations. The challenge is to find and avoid these vulnerabilities, especially since there are still numerous web apps with such issues. This project is to test, confirm, and study XSS vulnerabilities with DVWA to understand how the attacks take place and enhance security consciousness.

**Tools and Apps Used**

1. Damn Vulnerable Web Application (DVWA): A testing environment with deliberate security vulnerabilities to build exploitation and defensive skills.
2. Docker: For containerizing DVWA so that it has an isolated and equivalent setup on different machines.
3. Browser Developer Tools: To test, inspect, and edit scripts and DOM objects within the browser.
4. Python HTTP Server:To capture and demonstrate cookie theft via malicious payloads in stored XSS scenarios.

Plan: We will execute DVWA under Docker so we can securely test weak web applications. We will test all three main categories of XSS attacks — Reflected, Stored, and DOM-Based. We will then observe the results and note how these attacks function and what type of security we can implement to prevent them.

Architecture: The configuration is DVWA in a Docker container on the host. The testing tools interact with DVWA via the browser. The environment is secure and controlled in a way that testing XSS payloads is risk-free.

**Design or Flow of the Project**

1. Install and configure DVWA using Docker.
2. Use the DVWA interface to identify input fields that could be vulnerable.
3. Inject malicious scripts for Reflected XSS, and observe how the server immediately reflects and executes them.
4. Perform Stored XSS by saving scripts in form fields like comments or messages, which are then executed when viewed.
5. Simulate DOM-Based XSS by manipulating client-side code directly via the browser.
6. Set up a local Python HTTP Server (python3 -m http.server 3000) to act as a trap to receive stolen cookies or redirected requests.
7. Use browser console tools to observe the effects of XSS payloads.
8. Record findings and propose defense mechanisms such as input validation, output encoding, and security headers.

**Conclusion**

It will demonstrate how improper input checking and output handling can lead to security issues. The objective is to know how to discover and exploit XSS vulnerabilities in a secure manner, and to design good methods to prevent such issues, such as cleaning input, encoding output, and using Content Security Policies. Ultimately, this project desires to increase knowledge regarding web application security and promote the use of best practices in order to protect users from XSS attacks.