Protection Against Cross-Site Scripting (XSS) Attacks

Julianne Cox
Department of Computer Science

Introduction

- XSS attacks involve stealing cookies by injecting malicious scripts through user input
- Once injected, the scripts covertly deliver the website’s cookies to the attacker’s desired location
- We deployed two versions of a website: vulnerable and protected
- Users can submit comments
- XSS attacks succeed on the vulnerable site and fail on the protected one

Types of XSS Attacks

- Reflected XSS (AKA Non-Persistent) – the browser “reflects” malicious script when a user clicks on an attacker’s link
- Stored XSS (AKA Persistent) – script from an attacker is stored on the server; whenever the server content is loaded, so is the script
- DOM-Based XSS – when a link with script in it is clicked, the script is populated in the URL property of the DOM which executes the attack [2]

Methods of Protection

- Input Validation – allowing or disallowing input based on its presence/absence from a white/blacklist
- Input Sanitization – eliminating unwanted characters by “sanitizing” the input submitted
- Disabling HTTP Trace – a method which echoes input back to the user and could execute malicious script
- Escaping Control Characters – changing certain characters into text to prevent script execution
- Using an Automated Scanner – tools which can scan code to identify vulnerabilities [1]
- Performing Code Reviews – regularly review your code to ensure it properly handles user input

Implementation and Results

- The document displayed in Figure 1 is populated by script injections being loaded
- The script in Figure 3 includes a pop-up so users can immediately identify if attack was successful, which is displayed in Figure 4

![Figure 1](image1.png) – Stolen cookie sent to a document filled by the script injection
![Figure 2](image2.png) – Attack script stored in a MySQL database as a comment.

Acknowledgment: This project is funded by the QEP EDGE Curriculum Grant from Tennessee Technological University.