Protection Against Cross-Site Scripting (XSS) Attacks

Julianne Cox Dr. Denis Ulybyshev
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 exist which can scan code to identify vulnerabilities [1]
- **Performing Code Reviews –** regularly review your code to ensure it properly handles user input

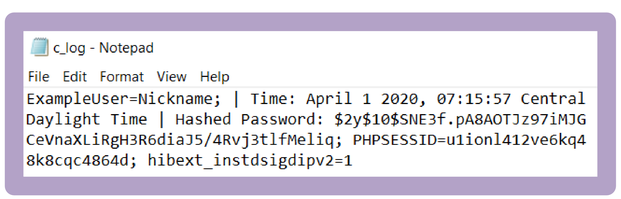

Figure 1 – Stolen cookie sent to a document filled by the script injection

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**

```
Julianne  | <script>document.location='cookieStealer.php?cookie_data='+document.cookie;</script>
```

Figure 2 – Attack script stored in a MySQL database as a comment.

```
humble comment <script> alert("The fun begins now"); document.location='cookieStealer.php?cookie_data='+document.cookie; </script>
```

Figure 3 – A comment with malicious script.

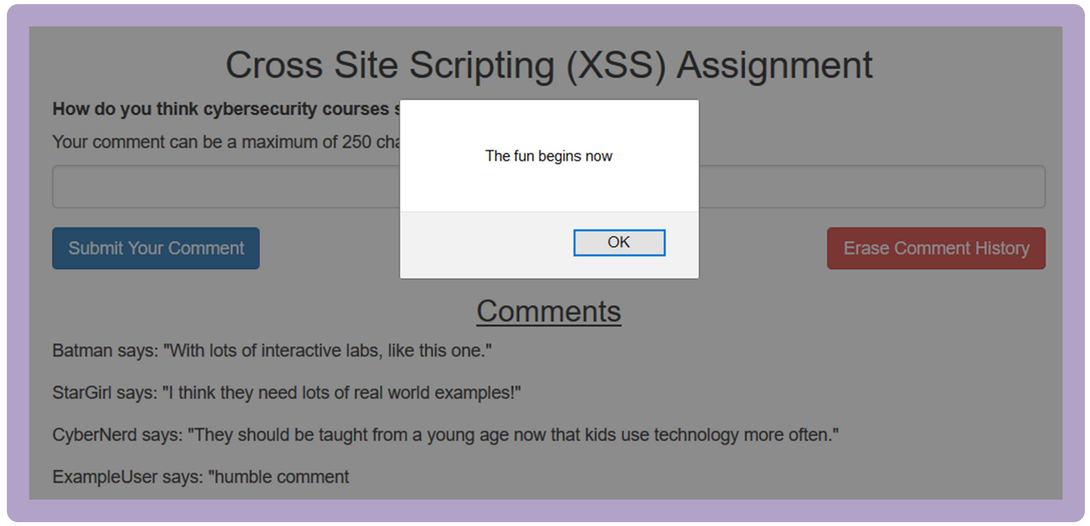

Figure 4 – The vulnerable version falling victim to a script injection.

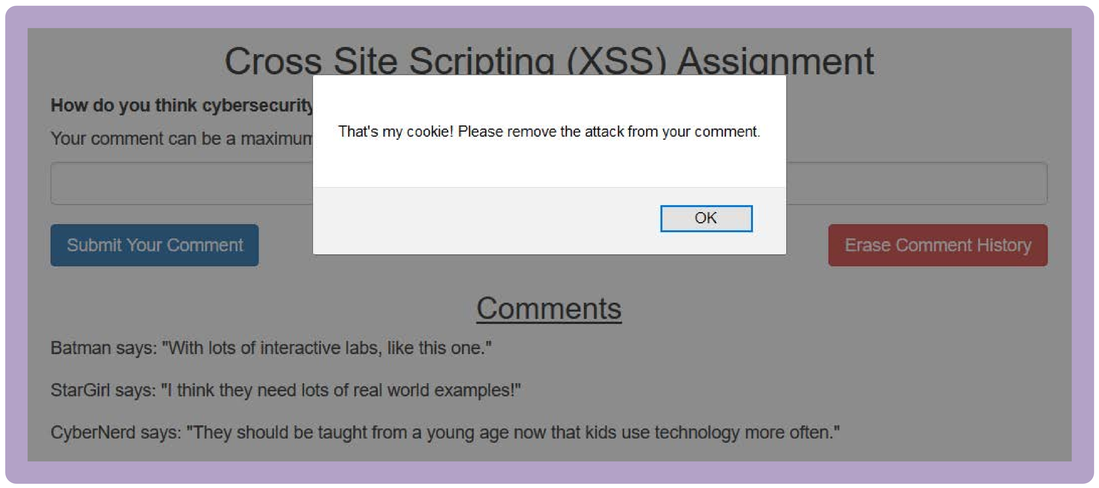

Figure 5 – The protected version successfully stopping an XSS attack.

Figure 5 contains one of many errors the user can receive based on which type of script they injected

Website Attributes

- **Back End:** MySQL database to host usernames, hashes of passwords, and comments; PHP, HTML, and JavaScript code used to create the websites.
- **Front End**: Users must go through registration and login pages to access the main discussion forum
- There are enforced parameters on the username and password (length, character requirements)
- **Chosen Attack Type:** Stored XSS
- **Chosen Protection Method:** Input Validation
- The PHP function "stristr" searches strings to identify common characters <, >, or /
- It also searches for key phrases "script", "document.cookie" present in script injections
- When a comment is posted, these functions scan it and look for those characters and key words
- If a comment is deemed malicious, it is blocked from being entered into the MySQL database and returns a warning to the user

Conclusions

- There are **15543** XSS vulnerabilities currently reported in the Common Vulnerabilities and Exposures (CVE) Public Database [8]
- XSS protection mechanism successfully implemented, using Input Validation method
- Solution source code is minimal which meets the original goal of being easily implementable
- Further research could be done on combining the function variations for ease of access

References

[1] Acunetix, "Preventing XSS Attacks," *Acutenix Web Security Zone.* Sep. 5, 2011.
[2] A. Klein, "DOM Based Cross Site Scripting or XSS of the Third Kind," *Web Application Security Consortium (WASC)*, v. 0.2.8, July 2005.
[3] D. Wichers et al, "Types of XSS," *The OWASP Foundation*,
[4] G. Rama Koteswara Rao et al, "Cross Site Scripting Attacks and Preventative Measures," *International Research Journal of Engineering and Technology (IRJET)*, vol. 4, no. 3, Mar. 2017. Feb. 9 2020.
[5] J. Manico et al, "Cross Site Scripting Prevention Cheat Sheet," *Cheat Sheet Series, OWASP,* no. 19, 2014.
[6] J. Manico et al, "DOM Based XSS Prevention Cheat Sheet," *Cheat Sheet Series, OWASP,* no. 21, 2014.
[7] N. Gupta, "Cross-Site Scripting (XSS)," IBM MSS. Research and Intelligence Report. Dec. 15, 2014.
[8] "Common Vulnerabilities and Exposures", 2020, [Online], Available: https://cve.mitre.org/, Last Accessed: 18 Apr. 2020

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