Securing User Browsing: Comparing Browsers and Extensions Effectiveness against XSS Attacks

CSCI 5742: Cyber Programming and Analysis – Project Proposal

Ishit Vasoya, 110961681
Anmol Singhal, 111021672

Spring 2024

Introduction:

Cross-site scripting (XSS) attacks pose a significant threat to web application security, with reflected XSS vulnerabilities presenting a particularly challenging problem. In these attacks, malicious scripts injected into web applications are at once reflected to users' browsers without proper validation. This project addresses the critical need to evaluate the effectiveness of different browser configurations in mitigating reflected XSS attacks.
Our primary aim is to empirically decide which browser configurations offer the most robust defense against reflected XSS attacks. This assessment will involve comprehensive testing of browsers both with and without extensions, as well as an examination of the impact of Content Security Policy (CSP) headers implemented on target websites.
Through rigorous testing protocols, we will find browsers that prove inherent resilience against reflected XSS attacks, even in the absence of extensions. Additionally, we will assess the efficacy of various browser extensions, such as ScriptBlock, NoScript, and Counter XSS, in enhancing browser security and mitigating XSS vulnerabilities.
Furthermore, we will examine the practical implementation of CSP headers on target websites to confirm their role in XSS mitigation. By analyzing browser responses to XSS payloads under different configurations, our aim is to offer conclusive insights into the relative effectiveness of browser security features, extensions, and CSP implementations.
In essence, this project is a proactive step towards strengthening web application security and safeguarding users' online experiences.

Project Overview:

The project involved building a vulnerable website using HTML, CSS, and Python for the backend. User data, uploaded files, code snippets, and uploaded URLs were stored in separate text files. The website had five vulnerabilities that could be exploited to perform XSS attacks.

Attack Vectors:

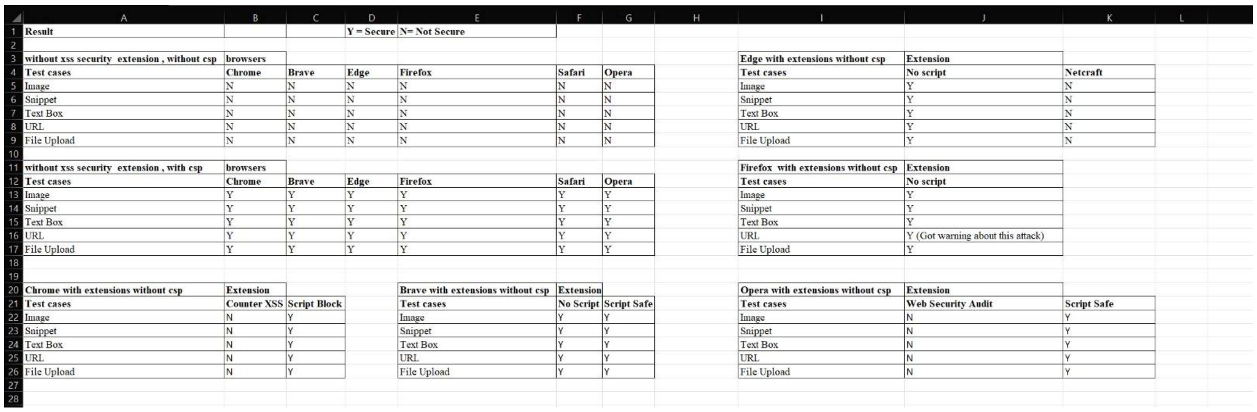
1. File Upload Attack: Uploading a malicious file and clicking the provided URL executed a script on the user's system due to a lack of proper control and sanitization.
2. URL Attack: Adding a malicious script to the end of the login page URL executed the script due to a lack of URL sanitization.
3. Snippets Attack: Adding a malicious script to the "New Snippets" page and hovering over the text on the "My Snippets" page executed the script due to a lack of input validation and sanitization.
4. Text Box Attack: Adding a malicious script to the "Favorite Color" text box on the "Update Profile" page and hovering over the color on the "Dashboard" page executed the script due to a lack of input validation and sanitization.
5. Image Attack: Adding a malicious script to the "First Name" text box on the "Update Profile" page and visiting the "Dashboard" page executed the script due to a lack of input validation and sanitization.

Methodology:

Two versions of the website were built: one without Content Security Policy (CSP) and one with CSP. The five attacks were tested on both websites across six different browsers (Chrome, Edge, Brave, Opera, Firefox, and Safari) with and without extensions and CSP. Various browser extensions, including CounterXSS, ScriptBlock, NoScript, Netcraft, Web Security Audit, and ScriptSafe, were used in the testing process.

Result (Data insights):

1. Without extensions and CSP, none of the six browsers were secure against even simple attacks.
2. With CSP, all attacks were mitigated by all browsers, making the website safe.
3. Netcraft, Web Security Audit, and CounterXSS, which claim to mitigate JavaScript code, did not work against attacks, even against the attacks with clear <script> tag.
4. NoScript, supported by most browsers, mitigated all attacks on Brave, Edge, and Firefox. However, on Firefox, the URL attack showed a special warning, making NoScript work best on Firefox.
5. Chrome's ScriptBlock and Brave's NoScript and ScriptSafe performed well.
6. Safari iOS claims to be the most secure browser, but without extensions and CSP, all attacks were successful, refuting its security claims. Moreover, available extensions were limited and paid.
7. Below is the table with the results of each of the test vectors. We did a total of 105 tests for introducing these above claims.



Result			Y = Secure	N= Not Secure

without xss security extension , without csp	browsers					
Test cases	Chrome	Brave	Edge	Firefox	Safari	Opera
Image	N	N	N	N	N	N
Snippet	N	N	N	N	N	N
Text Box	N	N	N	N	N	N
URL	N	N	N	N	N	N
File Upload	N	N	N	N	N	N

without xss security extension , with csp	browsers					
Test cases	Chrome	Brave	Edge	Firefox	Safari	Opera
Image	Y	Y	Y	Y	Y	Y
Snippet	Y	Y	Y	Y	Y	Y
Text Box	Y	Y	Y	Y	Y	Y
URL	Y	Y	Y	Y	Y	Y
File Upload	Y	Y	Y	Y	Y	Y

Chrome with extensions without csp	Extension	
Test cases	Counter XSS	Script Block
Image	N	Y
Snippet	N	Y
Text Box	N	Y
URL	N	Y
File Upload	N	Y

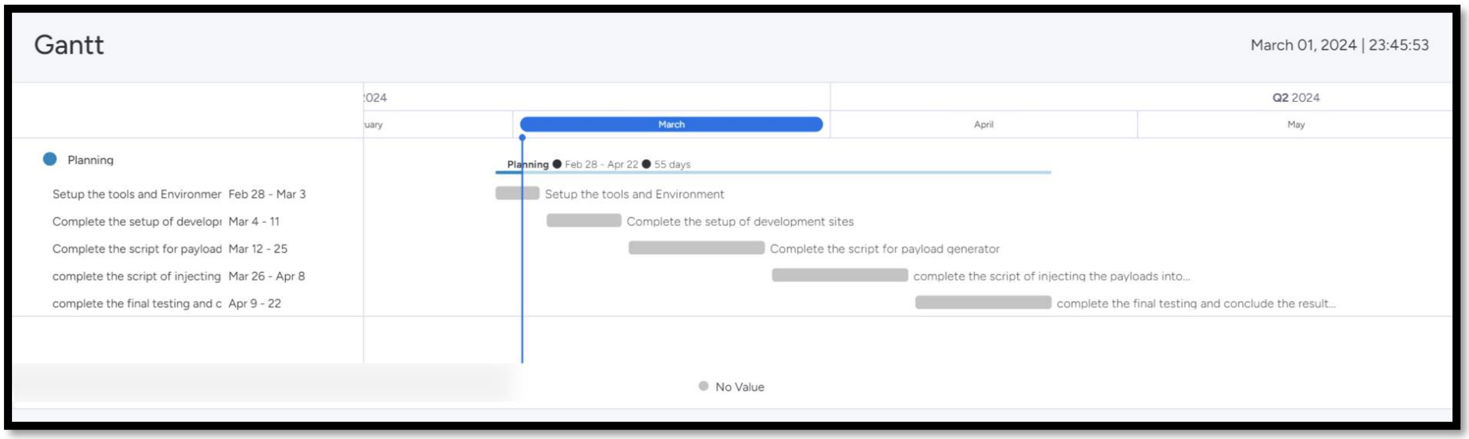
Brave with extensions without csp	Extension	
Test cases	No Script	Script Safe
Image	Y	Y
Snippet	Y	Y
Text Box	Y	Y
URL	Y	Y
File Upload	Y	Y

Edge with extensions without csp	Extension	
Test cases	No script	Netcraft
Image	Y	N
Snippet	Y	N
Text Box	Y	N
URL	Y	N
File Upload	Y	N

Firefox with extensions without csp	Extension
Test cases	No script
Image	Y
Snippet	Y
Text Box	Y
URL	Y (Got warning about this attack)
File Upload	Y

Opera with extensions without csp	Extension	
Test cases	Web Security Audit	Script Safe
Image	N	Y
Snippet	N	Y
Text Box	N	Y
URL	N	Y
File Upload	N	Y

Project Timeline:



Task	Start Date	End Date	Duration (Days)
Set up tools & environment	27/02/2024	3/03/2024	6
Develop testing sites	04/03/2024	11/03/2024	7
Payload generation script	12/03/2024	25/03/2024	14
Payload attack script	26/03/2024	08/04/2024	14
Final testing & conclusion	09/04/2024	22/04/2024	14
Presentation preparation	23/04/2024	29/04/2024	6

Conclusion:

The project proved the importance of proper input validation, sanitization, and the implementation of security measures like Content Security Policy (CSP) in mitigating reflected XSS attacks. It also highlighted the effectiveness of certain browser extensions, such as NoScript, in enhancing browser security against XSS vulnerabilities, especially in Firefox Browser. The findings of this project can serve as a valuable resource for developers, security professionals, and end-users in choosing right browser configurations and extensions to enhance protection against XSS attacks.

References (in APA[1] or ACM[2] citation format, must follow academic conventions)

1. Selenium Python Tutorial. (2020, June 9). GeeksforGeeks. https://www.geeksforgeeks.org/selenium-python-tutorial/?ref=lbp

2. dalfox/README.md at main · hahwul/dalfox. (n.d.). GitHub. Retrieved March 2, 2024, from
3. https://github.com/hahwul/dalfox/blob/main/README.md

4. XSS Filter Evasion - OWASP Cheat Sheet Series. (n.d.). Cheatsheetseries.owasp.org. https://cheatsheetseries.owasp.org/cheatsheets/XSS_Filter_Evasion_Cheat_Sheet.html

5. MITRE. (2019). CVE - Common Vulnerabilities and Exposures (CVE). Mitre.org. https://cve.mitre.org/

6. Offensive Security. (2019). Offensive Security's Exploit Database Archive. Exploit-Db.com. https://www.exploit-db.com/

7. Mitigate XSS Attacks with Content Security Policy. (2024, January 10). Discourse Meta. https://meta.discourse.org/t/mitigate-xss-attacks-with-content-security-policy/104243

8. What is Cross Site Scripting (XSS) ? (2019, May 14). GeeksforGeeks. https://www.geeksforgeeks.org/what-is-cross-site-scripting-xss/

9. HTTP headers | Content-Security-Policy. (2020, June 22). GeeksforGeeks. https://www.geeksforgeeks.org/http-headers-content-security-policy/
10. Mewara, B., Bairwa, S., & Gajrani, J. (2014). Browser's defenses against reflected cross-site scripting attacks. 2014 International Conference on Signal Propagation and Computer Technology (ICSPCT 2014),
11. Sangwan, S. (2020, December 6). s0md3v/XSStrike. GitHub. https://github.com/s0md3v/XSStrike

[1] https://www.mendeley.com/guides/apa-citation-guide/
[2] https://www.acm.org/publications/authors/reference-formatting