Project proposal on -

Cross-Site Scripting (XSS) Attack

Michelle Ramsahoye, Mohammad Imrul Jubair, Mohsena Ashraf, Waad Alharthi

CSCI 5403 (Team H)

Abstract

Cross Site Scripting (XSS) is a type of vulnerability attack that allows attackers to inject malicious scripts into a legitimate website or web application. These malicious scripts are executed when the user or victim visits the website, and unknowingly triggers them. In return, it allows the threat actor to steal user data, cookies, sessions, and even take over the entire website, which can lead to critical security issues. In this project, we plan to set up our testing lab, perform a vulnerability scan to find out the potential XSS vulnerabilities for hijacking session and cookies of users, and conduct penetration testing while demonstrating against different types of XSS attacks. Also, we want to explore the possible ways to find the prevention techniques and defend against them.

I. Introduction

Cross Site Scripting is a type of vulnerability that can impact web applications and gives cybercriminals the ability to insert malicious programs into a target website. The subsequent execution of these scripts by users who are unaware of their presence can result in a variety of security issues, including the stealing of data, the disruption of sessions, and others. This vulnerability can be really fatal as it empowers the attacker to imitate the victim user, access the user's data and perform tasks as the victim user, capture the user's credentials, and can even inject trojans in the website while taking over the website's control (Gupta & Gupta, 2017).

In this report we propose a project to implement and analyze XSS vulnerabilities to demonstrate the procedure of identifying it, exploiting an application with it, and mitigating the associated risks. We mainly target to simulate a scenario to steal sessions and cookies via XSS. If a website is compromised by session and cookie hijacking, attackers could hijack users' login credentials. Among other types of confidential information, they can also obtain credit card data.

In this project, we plan to cover the following aspects -

- A background on XSS, including few examples of relevant vulnerability events listed in CVE.
- A method to perform XSS vulnerability scanning and attacking to steal user session and cookies on our testing lab environment.
- A presentation of our experimental results.
- Analysis of the techniques for preventing and defending XSS attacks, along with the best practices for secure web development.

II. Background

Classified under "Injection" as the 3rd most critical cybersecurity risk according to the OWASP Top Ten, cross-site scripting (XSS) is a web application vulnerability that can have severe consequences for website owners and users. XSS attacks are typically carried out by injecting code into vulnerable fields such as search boxes or comment sections, and then tricking unsuspecting users into executing that code. Some of the largest XSS attacks within the last decade included the 2018 British Airways data breach affecting 380,000-500,000 customers (BBC News, 2020), 2019 Fortnite/Epic Games data breach with 200 million affected players (Ng, 2019), and the eBay data breach which affected 145 million customers (Reuters, 2014). Figure 1 demonstrates the process of the execution of an XSS attack.
There are three types of XSS attacks (*OWASP Foundation*), which are described below:

1. **Reflected XSS:** These types of XSS attacks are also known as non-persistent or Type-1 attacks. In a reflected XSS attack, the injected script will reflect back into the user's browser and show some or all of the input sent to the server. These are often delivered to the victims through an email message or another website.
2. **Stored XSS:** Stored XSS, also known as persistent or Type-2 attacks, refers to the type of attacks where the injected script is located permanently on the target server (like in a visitor log or comment field). The victim will then retrieve this script when they request the stored information.
3. **DOM Based XSS:** DOM XSS attack stands for Document Object Model-based Cross-site Scripting. It is also known as Type-0 attack, where the payload is executed after modifying the DOM "environment" in the victim's browser, causing the client side code to run in an "unexpected" manner. There will be no changes in regards to the response page in this case, but the client side code will execute differently.

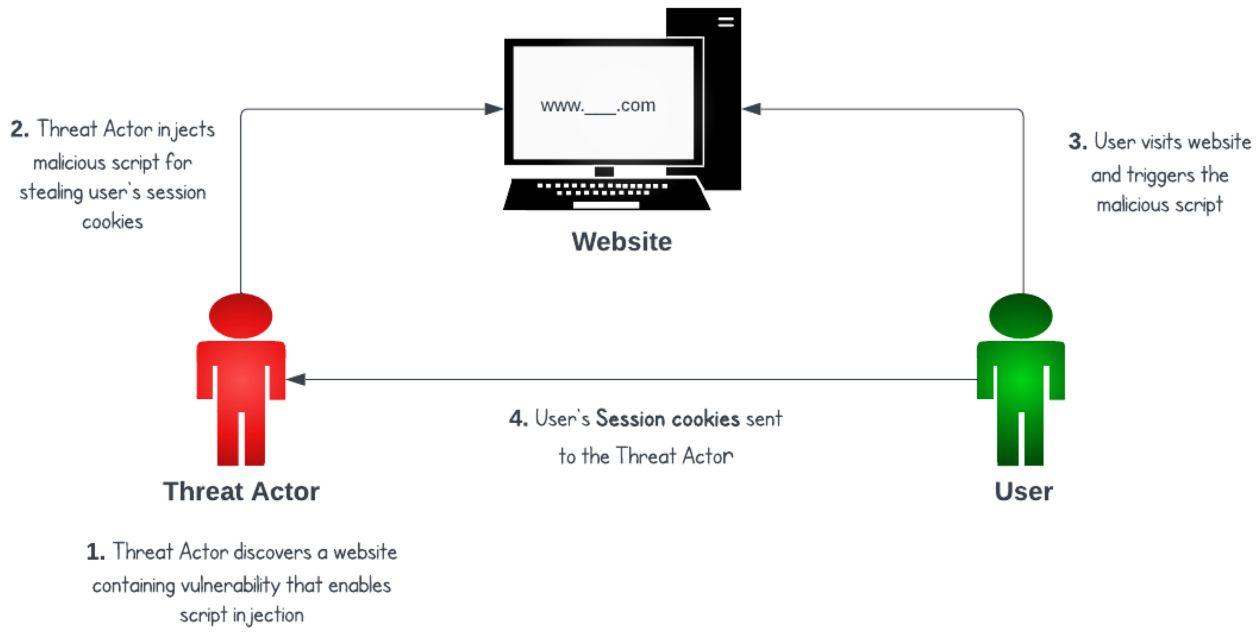
2. Threat Actor injects malicious script for stealing user's session cookies
www.___.com
Website
3. User visits website and triggers the malicious script
Threat Actor
4. User's Session cookies sent to the Threat Actor
User
1. Threat Actor discovers a website containing vulnerability that enables script injection

Figure 1: Demonstration of the execution process of Cross-Site Scripting (XSS) Attack

As our primary plan is to exploit the XSS attack for stealing session cookies, we found several CVE entries regarding this vulnerability, which works as a motivation behind this project. Table 1 depicts the related information.

CVE Entry	Procedure	Impact	CVSS Score
CVE-2022-29241	Brute-forcing the PID of the server	Session hijacking, Gaining control over the system	9.0
CVE-2021-45813	Injection of malicious Javascript code	Session hijacking, Credential theft	4.3
CVE-2021-45812	Injection of malicious Javascript code	Session Hijacking	4.3
CVE-2021-42703	Injection of malicious Javascript code	Hijacking session tokens, Unintended Browser Action	4.3
CVE-2020-1673	Clicking a crafted URL sent via phishing email	Hijacking target user's HTTP/ HTTPS session	7.6

Table 1: CVE entries focusing on stealing session cookies via XSS attack

III. Proposed Method

The project will be conducted over several weeks and will consist of different steps. Below we explain the testing lab we will be using in this project followed by the steps we plan to take.

Setting up a testing lab

To examine different methods of cross-site scripting we will use our testing lab, and we will use several tools for this purposes, which are -

- **Environment:** Linux distribution on Virtual machines.
- **Victim website:** DVWA or Damn Vulnerable Web Application [1]
- **Vulnerability scanner:** Automated and manual tools - Nmap, and OWASP Zap [2].
- **Attacking program:** JavaScript

Project plan

Using the above configuration, we plan to follow the steps below to accomplish our goal.

I. To gain a better understanding of XSS vulnerabilities and how they can be exploited, we will create our **environment** and configure our **victim website** for testing purposes. We will use our **vulnerability scanner** to scan and identify any potential vulnerabilities on our victim's side that

[1] https://github.com/digininja/DVWA
[2] https://www.zaproxy.org/

could be exploited. This will allow us to gain a better understanding of the lab's structure and potential vulnerabilities.

II. Once we identify a vulnerability, we will demonstrate how a threat actor can use it to exploit the chosen vulnerability using our **attacking program**. We will show how the potential impact it can have on a web application and its users. Our main target is to show the impact of cross site scripting attack via hijacking sessions and cookies of a visitor of the victim website. We will be demonstrating these scenarios for reflected, stored, and DOM XSS attacks.

III. Finally, we will analyze the possible ways of the prevention mechanism**.** We plan to implement and test mitigation strategies, i.e., input validation, sanitization, and output encoding, and we will try to develop secure web applications that are resistant to XSS attacks on hijacking of sessions and cookies. We will conduct a second vulnerability scan to verify that the vulnerability is no longer present.

IV. Conclusion

This project proposal outlined our approach to a practical implementation of cross-site scripting (XSS) attacks with an analysis of identifying and exploiting it, and mitigating related risks. In this report, we presented a background study on cross-site scripting and related vulnerabilities listed in CVE. We mentioned our project plans including setting up our testing lab, performing a vulnerability scans, and conducting penetrations. We also proposed to demonstrate the attacks for different types of XSS for stealing session and cookies of visitors. Moreove, we also exhibited our plans of exploring the possible ways to find the prevention techniques and defend against them.

References

BBC News. (2020, October 16). *British Airways fined £20m over data breach*. https://www.bbc.com/news/technology-54568784

OWASP Foundation. *Cross Site Scripting (XSS).* Retrieved March 14, 2023. https://owasp.org/www-community/attacks/xss/

Gupta, S., & Gupta, B. B. (2017). Cross-Site Scripting (XSS) attacks and defense mechanisms: classification and state-of-the-art. *International Journal of Systems Assurance Engineering and Management*, *8*(S1), 512–530. https://doi.org/10.1007/s13198-015-0376-0

Ng, A. (2019, January 16). *Fortnite had a security vulnerability that let hackers take over accounts*. CNET. https://www.cnet.com/tech/gaming/fortnite-had-a-security-vulnerability-that-let-hackers-take- over-accounts/

Reuters. (2014, May 22). *Hackers raid eBay in historic breach, access 145M records.* CNBC. https://www.cnbc.com/2014/05/22/hackers-raid-ebay-in-historic-breach-access-145-mln-records.html