Discussion Topic

Open-source tools are available to create UML diagrams; some are listed below. This list is not exhaustive. The benefit of using such tools is that they ensure that the recognised UML components correctly represent the parts of the model.

- **Visual Paradigm**
- **Sequence Diagram**
- **Umbrello**

Choose an open-source UML tool from the list above. Select one of the coding weaknesses OWASP has identified and create a flowchart of the steps that may have led to the weakness. Which UML models might you use to present the design of your proposed software, and why are they the most appropriate choice(s)?

My Initial Post:

In this discussion, and for my initial post, I have chosen the OWASP coding weakness, "Cross-Site Scripting (XSS)".

Gupta et al. (2015) stated that an application-level code injection security flaw is called cross-site scripting (XSS). It happens whenever a server programme (i.e., dynamic web page) includes unchecked input from an HTTP request, a database, or files in its response. It enables a hacker to steal confidential data and do other evil deeds.

Synopsys (N.D.) defined XSS as a malicious attack where an attacker injects harmful scripts into a trusted website or app by sending a malicious link to a user.

XSS attacks involve stealing sessions, taking over accounts, bypassing MFA, replacing or defacing DOM nodes and executing malicious code on the user's browser (OWASP, 2017). For instance, XSS attacks can occur when a program modifies a webpage with user data or inserts unverified data into a new page. This allows hackers

to run scripts in the victim's browser, hijack user sessions, or redirect users to malicious sites.

OWASP (2021) stated that applications are vulnerable to attack when they fail to verify, filter, or sanitise user-supplied data.

To represent the weaknesses of XSS, I have chosen a Sequence Diagram based on the ideas of Jindal (2017) because I can include more details, such as lifeline bars, return messages, and other representations.

Flowchart of Steps Leading to XSS attack:

1. The hacker creates a malicious link beforehand, which will be used in the online attack later.

2. The hacker then persuades other people via email to click the URL and follow the link.

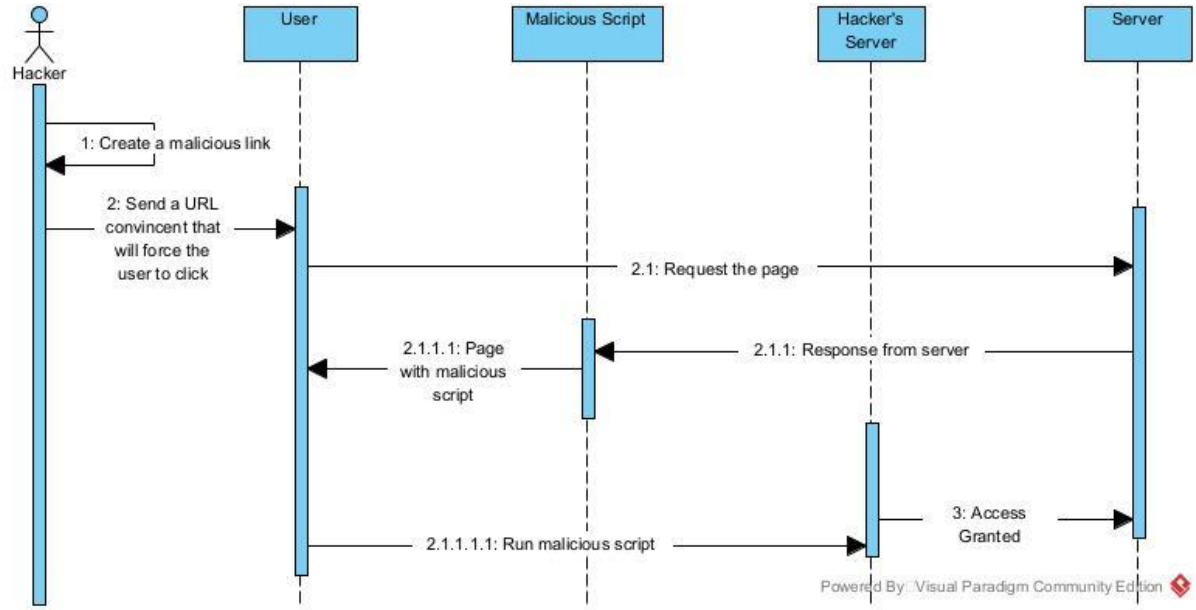
2.1. The user requests the web page from the server after clicking the malicious link.

2.1.1. The client receives a login page from the server.

2.1.1.1. The malicious script now functions and attaches to the response page.

2.1.1.1.1. The user enters his login information and clicks a malicious script button, unintentionally sending his information to the hacker’s server.

3. The hacker has now gained access to the main server.


Hacker
User
Malicious Script
Hacker's Server
Server
1: Create a malicious link
2: Send a URL convincent that will force the user to click
2.1: Request the page
2.1.1: Response from server
2.1.1.1: Page with malicious script
3: Access Granted
2.1.1.1.1: Run malicious script
Powered By: Visual Paradigm Community Edition

References:

Gupta, M.K., Govil, M.C. & Singh, G. (2015). *Predicting Cross-Site Scripting (XSS) security vulnerabilities in web applications*. [online] IEEE Xplore. doi:https://doi.org/10.1109/JCSSE.2015.7219789.

Synopsys (N.D.). *What Is Cross Site Scripting (XSS) and How Does It Work?* | *Synopsys*. [online] Available at: https://www.synopsys.com/glossary/what-is-cross-site-scripting.html#:~:text=Cross%2Dsite%20scripting%20(XSS)%20is%20an%20attack%20in%20which.

OWASP (2017). *OWASP Top 10 – 2017 The Ten Most Critical Web Application Security Risks.* [online] Available at: https://owasp.org/www-pdf-archive/OWASP_Top_10-2017_%28en%29.pdf.pdf

OWASP (2021). *A03 Injection - OWASP Top 10:2021*. [online] owasp.org. Available at: https://owasp.org/Top10/A03_2021-Injection/.

Jindal, C. (2017). *XSS & Its Types*. [online] Available at: https://astrologer.chetanjindal.com/2017/06/xss-its-types.html.