Simran P, 2022, 10:2
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

Automating Cross Site Scripting

Simran P, Dr. A. Rengarajan
Dept of MCA, School of CS & IT,
Jain Deemed-to-be University, Bengaluru,India
Email: simrangiriya@gmail.com, a.rengarajan@jainuniversity.ac.in

Abstract- Cross site scripting (XSS) is a type of scripting attack on web pages and account as one of the unsafe vulnerabilities existed in web applications. As prevention against such attacks like, it is essential to implement security controls that certainly block the third-party intrusion. Recently the most dangerous attacks are reflected, and DOM based cross-site scripting attacks because in both cases attacker's attack using server-side scripting and do forgery over the network, it is very hard to detect and therefore it must be prevented. Vulnerabilities of the websites are exploited over the network through web request using GET and POST method

Keywords- XSS attack, web application, XSS vulnerabilities, phishing attack, fake maliciouswebsite, cross-site scripting, security of

I. INTRODUCTION

Cross-Site Scripting, commonly also known as XSS, is a type of attack that gathers malicious information about a user, typicallyin the form of a hyperlink that will save the users credentials. Cross-site scripting (XSS) is a web security vulnerability where the attacker injects malicious client-side script into a web page. When a user visits a web page, the script code is downloaded and run by the web browser.

XSS represents most web-based security vulnerabilities. One reason of the popularity of XSS vulnerabilities is that developers of web-based applications often have little or no security background. The result is that it can be poorly developed code, riddled with security flaws, is deployed, and made accessible to the whole Internet. Currently, XSS attacks are dealt by fixing the server-side vulnerability, which is usually the result of improper input validation routines.

XSS protection can be configured for multiple types of request and response data – URL query parameters – URL encoded input ("POST data") – HTTP headers, Cookies. The possibilities to manipulate HTML documents displayed by the browser with JavaScript or influence the operation of the browser itself are dangerous features if misused. Document and access credentials JavaScript also provides access possibilities to this information. We are using two types of xss in this project as follow:

Reflected XSS Attacks

Reflected attacks are the ones where the injected script is reflected off the web server, such as in an error message, search result, or any other response that includes some of the input sent to the server as part of the request only. Reflected attacks are sent to victims via another route, such as in an e-mail message, or on some other website.

When a user is clicking on a malicious link, submitting a specially crafted form, or even just browsing to a malicious sitethen injected code travels to the vulnerable web site, which reflects the attack back to the user's browser. The browser then executes the code because it came from a "trusted" server. Reflected Cross site scripting is also referred to as non-Persistent.

Stored XSS Attacks

Stored attacks are those when the injected script is permanently stored on the targets servers, such as in a database, message forum, visitor log, comment field, etc. The victim then recoversthe malicious script

International Journal of Science, Engineering and Technology

An Open Access Journal

from the server when it requests the stored information. Stored XSS is also sometimes referred to as Persistent.

II.SYSTEM REQUIREMENTS

Requirements Specifications specifies the usage of Software and Hardware with them corresponding version, modules, etc...., which are necessary for the overall of the project.

1.Hardware requirements:

Processor – i5 0r i7
Hard Disk –200 GB
Memory – 4GB RAM

2.Software Requirements:

OS: Windows 10, VMware, Kali Linux
Go Language

III.SYSTEM ANALYSIS

For the following system to run this project you need the following requirements as follow-

1. Kali Linux –Kali Linus is an open-source, Debian-based Linux distribution geared towards various information security tasks, such as Penetration Testing, Security Research, Computer Forensics etc.
2. Installed ParamSpider - ParamSpider, a new open-source tool, automates the discovery of parameters in URL addresses, a key step in probing websites and applications for vulnerabilities.
3. Installed Dalfox- It is a fast, powerful parameter analysis and XSS scanner, based on a golang/DOM parser.

The above-mentioned requirements should be there to run this project smoothly. With the help of this we will be able to find the list of URLs which are vulnerabilities and we can proceed with the next step of finding the XSS so we cansee the result of the vulnerable

IV.PROJECT DESCRIPTION

XSS attacks are those type of attacksthat the web applications which is often used to steal the cookies from a web browser 's database. The following figure 1 is an architecture which shows the steps of exploiting the XSS vulnerability by a malicious attacker. This architecture contains three partsas follows- Attacker Domain, Victim Domain and Vulnerable Web Application. Here are some steps which will explain for exploiting the XSS attack: -

- The attacker has found that theweb application is vulnerable to Cross-Site Scripting attack. After this, attacker will post a malicious Java Script Code on the Vulnerable Web Application whose function is to steal cookies of the victim 's account session.
- Then the victim logs into the vulnerable web application by giving the user-id and password. As a result, the web server of web application will generate and transfer the cookie of that particular session to victim 's web browser.
- The victim browsesthe malicious Java Script Code and gets executed on its browser.•The Script Interpreter of the victim 's browser gets invoked and transfers the cookies of the victim 's session to the attacker 's domain.
- Now lastly, these cookies will be utilized by the attacker to get into the account of victim and attack.

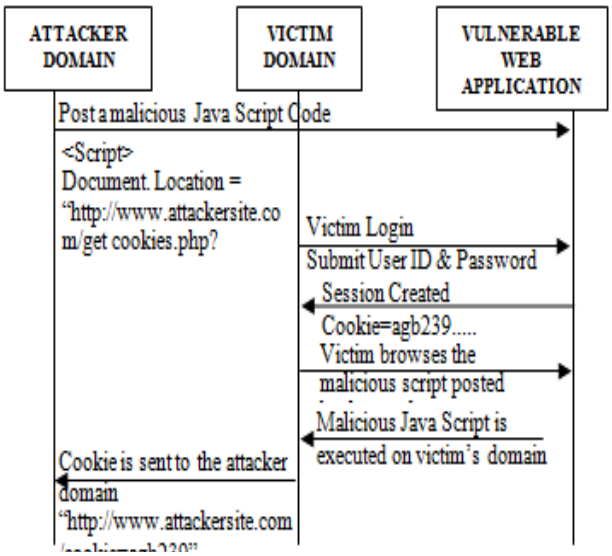

Fig 1 Working of XSS

V.EXISTING SYSTEM

In the existing system we do manual reconnaissance for Cross Site-Scripting (XSS). Manual testing may involve entering sentinelXSS inputs into form fields and parameter values in HTTP Requests and look for resulting pop-ups in subsequent responses. It is very difficult to go through numbers of URL's and find the vulnerability in them, has it been time consuming. So, if we use this tool, it saves time and increases accuracy as well.

International Journal of Science, Engineering and Technology

An Open Access Journal

VI.PROPOSED SYSTEM

In the proposed system we are using automated tool, The complexity of today's websites and web-applications practically mandates the use of security testing tools to keep their data safe. There are several automated tools, including some Browser Plugins that can be useful in detecting Cross-Site Scripting (XSS) vulnerabilities. The use of automated tools can lend a false sense of security to developers and testers since the tools can be blind to certain variations of Cross-Site Scripting (XSS) defects.

We are using two tools as follows:

1. ParamSpider-ParamSpider, a new open-source tool, automates the discovery of parameters in URL addresses, a key step in probing websites and applications for vulnerabilities.
2. DalFox- It is a fast, powerful parameter analysis and XSS scanner, based on a golang/DOM parser.

VII.FLOW CHART DIAGRAM

The flowchart diagram shows the complete flow of data from the beginning till the end of the project. The verifications, which are done by the system to acknowledge the identity of the user and give him/her, access to the system.

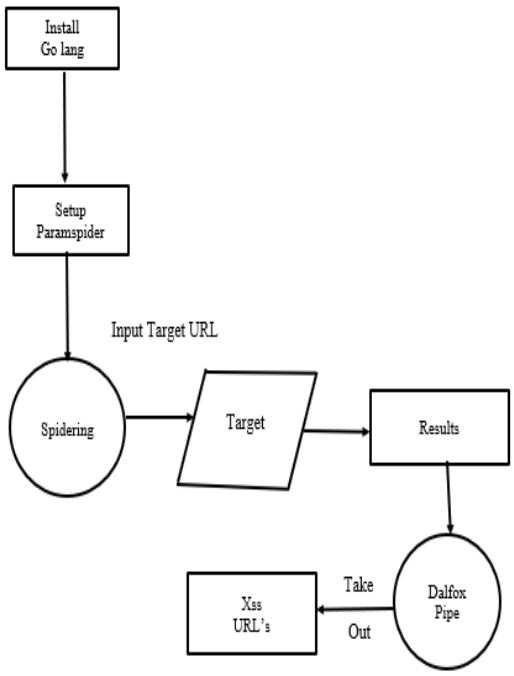

Fig 2 Flow chart .

VIII.CONCLUSION

Cross-site scripting attacks, while a passive attack, but because there are many sites on this loophole, likely to cause leaks and illegal data server to steal, the danger is very large, if ordinary users browse the web carefully, WEB applications developers tight design, this attack is difficult to achieve. This project will help us to find the cross-site scripting vulnerabilities using automated tools which is much faster than the manual method. It helps the user or the person to use the tools such as ParamSpider and Dalfox for finding the parameters and procedure further.

REFERENCES

1. Shielding Cross-Site Scripting Attacks Using the State of Art Techniques Megala Manickam and Uma Maheshwari Govindasamy
2. A study on detection of Cross Site Scripting (XSS) attacks, Vinayak Pai, Govardhan Hegde K
3. Cross-Site-Scripting Attacks and Their Prevention during Development Ms. Daljit Kaur, Dr. Parminder Kaur
4. BLUEPRINT: Robust Prevention of Cross-site Scripting Attacks for Existing Browsers Mike Ter Louw mter V.N. Venkatakrishnan
5. Detecting Cross-Site Scripting in Web Applications Using Fuzzy Inference System Bakare K. Ayeni, Junaidu B. Sahalu, and Kolawole R. Adeyanju
6. A Survey on Cross-Site Scripting Attacks Joaquin Garcia-Alfaro and Guillermo Navarro-Arribas
7. A Study of Existing Cross-Site Scripting Detection and Prevention Techniques Using XAMPP and VirtualBox Jalen Mack, Yen-Hung (Frank) Hu, and Mary Ann Hoppa
8. Cross-site Scripting Research: A Review PMD Nagarjun, Shaik Shakeel Ahamad
9. A Survey on Detection and Prevention of Cross-Site Scripting Attack V. Nithya, S. Lakshmana Pandian and C. Malarvizhi3
10. Prevention Of Cross-Site Scripting Attacks (XSS) On Web Applications in The Client-SideS. SHALINI, S. USHA