XSS ATTACK DETECTION AND PACKET ANALYSIS USING MONOSEK

[1]Chaya P [2]Anjali N Menon [3]Madhurya Aithal A [4]Pratheeksha J [5]Varsha S
[1]Assistant Professor [2,3,4,5]Student
Department Of Information Science and Engineering
[1,2,3,4,5]GSSS Institute of Engineering and Technology for Women, Karnataka, India.

Abstract - The advent of network in all kinds of business technologies has made every individual more dependent on the internet for all the purposes. So are the threats for the same is increasing and the network security has become a major issue. Our project aims in detecting on the most popular attacks, the xss attack in the websites using the Monosek- a Network Processor based Network Packet Processing and Network Session Analysis system. Also the traffic generated in this attack produces packets which a recollected in the database and analyzed for further use.

***Keywords:* Xss attacks, networks, cross-site scripting, packet analysis, Monosek, network security.**

I INTRODUCTION

People are relying on the internet for almost everything now a day. Web applications are used for variety of purposes like simple thing as accessing a blog to banking related difficult tasks. These web applications handle large amount of user's personal data. The importance of this information like banking related information, personal information, etc. interests the attacker in these web applications. web applications have become major targets of hackers who take advantage of web developers' poor coding practices, weaknesses in the application code, inappropriate user input authorization, or no adherence to security standards by the software developers. This leads to Cross site scripting attack. Cross-site scripting (XSS) is a form of web security attack which involves the injection of malicious codes into web applications from untrusted sources. There are three actors in this attack (XSS) the attacker, the website and the victims can take use of JavaScript, Flash, VBScript and ActiveX. But mostly used is JavaScript. XSS occurs even when the servers and the database engine contains no vulnerability themselves, and it is arguably one of the most predominant web application exposures today (Fig 1).

The chart below shows the relative share of vulnerability types found and the sum equals 100%. Out of all vulnerabilities discovered XSS attack and Information leakage are the largest share because they occur often and in some cases multiple times per application.

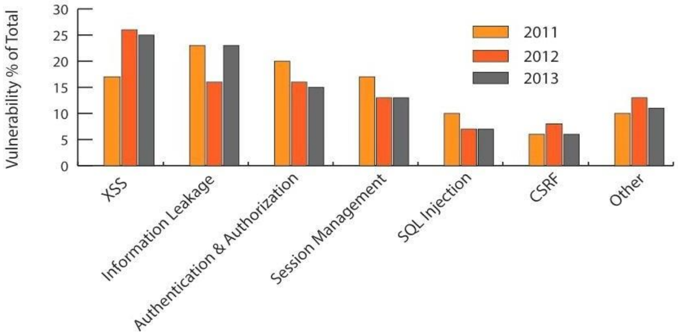

Fig 1: Cenzic Application Vulnerability Trends Report (2013).

A. INTRUSION DETECTION SYSTEM (IDS)

MONOSEK is intrusion detection software that monitors high speed network traffic by developing own traffic pattern with API calls. This software is embedded software for packet analysis, session analysis and deep packet inspection. MONOSEK plays a major role in order to analyze each packet that is transmitted in the network traffic and to detect the occurrence of XSS attack in the network. XSS attack detection is the major aim of the project where we have an attacker system and victim system along with a MONOSEK server to monitor the packet transmission. As the attacker floods the victim system by enormous packets by forging the victim IP address, attack occurs and victim is denied of the service. In order to detect the attack occurrence, we use MONOSEK server which alerts the user as soon as the XSS attack occurs.

The main objective of the project is to

- Trace out cross site scripting vulnerabilities in the web application to steal user's authentication details.
- To provide user's confidential data and user's profile.
- Avoiding malwares to interact and misuse user data.

II. LITERATURE REVIEW

Various methodologies have been implemented till date on different platforms. Even though no IDS are 100 % secure. In this section we will take a look at previously proposed systems. A static string analyzer [1] checks the string output of a program with context free grammar. This technique checks the presence of "<script>" tag in the whole document. As web applications more often have their own scripts and also there are several other ways to invoke a JavaScript interpreter, the approach is not at all practical to find XSS vulnerabilities.

A team developed [2] DEXTERJS tool to detect and prevent the DOM-based XSS vulnerability on the web application by using the taint tracking mechanism. The tool is evaluated by the Alexa top 1000 sites which contain 820 distinct zero-day DOM-XSS. Raghuveer and Chandrasekhar proposed a model combining techniques like Support Vector Machine classifier, fuzzy neural network and K-means. The input dataset is clustered using K- means algorithm in to k clusters, which are trained with the help of neuro fuzzy logic. Vectors are generated by passing each of the data generated through neuro fuzzy classifier and at last classification based on redial SVM(Support Vector Machine) is done to detect intrusion in the system. [3]. Paper[4] uses the methodology Web Vulnerability Scanners (WVS) which has three major components: A crawling component, an attack component and an analysis component. It merging the mechanisms provided from XSS and SQL Injection. But the detection rate of certain type of XSS vulnerabilities is disappointing. In particular, scanners face problem in detecting stored XSS properly. Nonce spaces tool to prevent XSS attacks by using the Instruction Set Randomization (ISR) techniques to differentiate between benign and malicious contents for thwart the XSS vulnerabilities exploitation. But Doesn't contain any defensive mechanism for inserted JavaScript code when downloading from the remote web site.[5]. H B Kuan Tan and L K Sharet described a method to detect and prevent XSS attacks. This model eliminates XSS vulnerabilities from the code in two parts. In first detection part taint-based analysis is done to track the flow of user data in HTML output statements and see if there are any vague statements or comments. In second part pattern matching mechanism and data dependency is checked to prevent the injection in code due to XSS.

III METHODOLOGY

A. CROSSSITE SCRIPTING ATTACKS

The XSS attack is a malicious JavaScript code executed in the victim's browser to steal the user's credentials such as cookies, credit card numbers, and passwords… etc. Fig. 2 shows a high-level view of a typical XSS attack. The XSS attacks are divided into three types namely; Reflected XSS attack, Stored XSS attack, and DOM-based XSS attack.

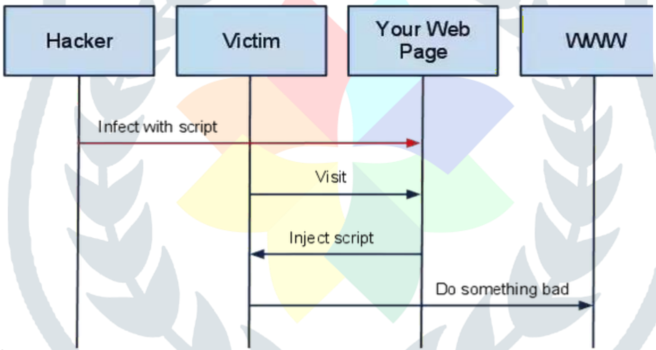

Fig 2 .A high-level viewing of typical XSS attack

The XSS attack process, the taxonomy of XSS attack, the statistics of XSS attacks serious vulnerability and the incident XSS attack are discussed in the following subsections. A. XSS attack process: The hackers inject a malicious script code into the web page and when the user visits this website, the malicious script exploits the credentials of the user by hijacking session ID, password, credit card number or cookies [3, 4, 6, 8].

B. The taxonomy of XSS attacks:

XSS is classified into two main classes, server-side vulnerability and client-side vulnerability.

1. Server-side vulnerability:

The server-side vulnerabilities occur more frequently than the client-side vulnerability since most of the web applications are generated by dynamically scripts and contain the user input and responses to user's requests. The server-side vulnerability has two types; Reflected XSS attack and Stored XSS attack.

a. Reflected XSS attack (Non-persistent attack):

The purpose of this attack is to steal the session cookie of the user. This type of XSS attack requires a more interaction between the victim and the attacker [3, 5]. The steps of the Reflected XSS attack are illustrated in Fig. 3. The steps are as follows:

1. The attacker sends a link containing malicious script code to the user via email or any similar webpage.
2. When the user clicks on this link, the malicious code is sent to the server without being detected by the web application
3. The server sends the HTTP response to the user with the malicious script code
4. The attacker's domain receives the user's cookies after executing the script contained in the response.
5. The attacker can store these cookies for future use.

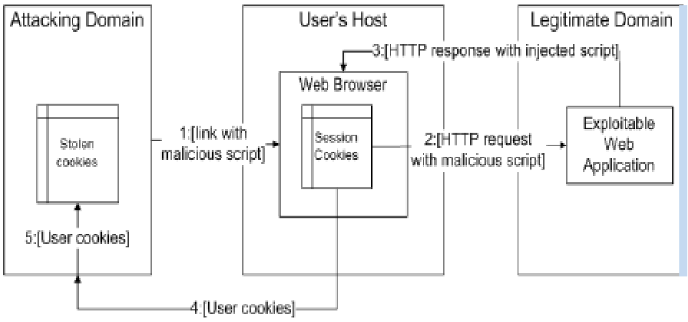
Attacking Domain
User's Host
Legitimate Domain
3:[HTTP response with injected script]
Web Browser
Stolen cookies
1:[link with malicious script]
Session Cookies
2:[HTTP request with malicious script]
Exploitable Web Application
5:[User cookies]
4:[User cookies]

Fig 3. The Reflected XSS attack

b. Stored XSS attack (persistent attack)

This kind of attack occurs frequently in the social networks and other similar applications. The steps of the Stored XSS vulnerability are illustrated in Fig. 4. These steps are as follows [3,5]:

- The attackers insert a malicious script code on a web application that has vulnerability.
- When the user sends HTTP request, the web page content which include the malicious code
- The malicious script is sent to the user by the HTTP response.
- The script is executed in the web browser and sends the session cookies to the attackers.
- These stolen cookies are stored on the attacker's domain

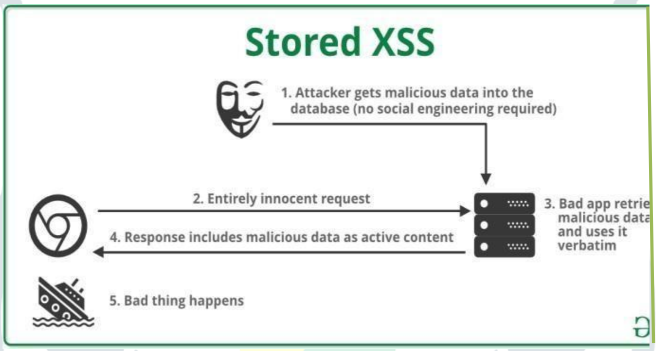
Stored XSS
1. Attacker gets malicious data into the database (no social engineering required)
2. Entirely innocent request
3. Bad app retrie malicious data and uses it verbatim
4. Response includes malicious data as active content
5. Bad thing happens

Fig 4. The Stored XSS attack

C. Proposed Work:

Our motive is to detect most common attack of all time i.e. Web Application attack (XSS). Here we are proposing rules to identify XSS (Web Application) attack with the help of IDS, and we will monitor our incoming & outgoing packets which will further match with our database rules. The XSS attack process is illustrated in Fig. 5.

The process consists of the following steps:

1. When the attacker finds vulnerability in a web page, he/she injects a malicious script code (JavaScript code) into this vulnerable web application to steal the sensitive credentials from the victim’s web browser.
2. The victim visits the infected web page.
3. The website sends the requested page with the malicious code as part of the HTML body.
4. Once the malicious script code is executed inside the victim’s browser, the attackers steal the personal information from the victim’s cookies.

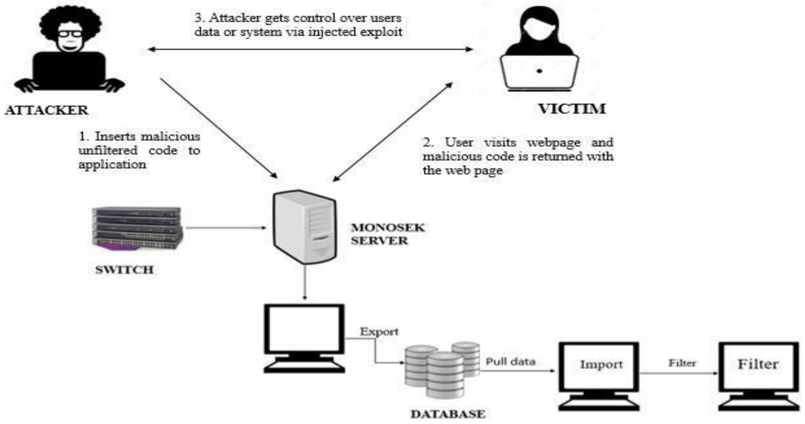
3. Attacker gets control over users data or system via injected exploit
ATTACKER
VICTIM
1. Inserts malicious unfiltered code to application
2. User visits webpage and malicious code is returned with the web page
MONOSEK SERVER
SWITCH
Export
Pull data
Import
Filter
Filter
DATABASE

Fig 5. The XSS attack

System Modules:

- Attacker: The attacker is the one who will insert the malicious unfiltered code to the server to get the required information for him. Attacker inserts the malicious code to the web page where the victim visits. Whenever the victim visits the web page he will be under attacked by the attacker and will get the information which is needed. And also, attacker get the control over the user data or system via injected exploit.
- Victim: The victim module is the one where he will be affected by the attacker once he get into the malicious page and the malicious data is sent to get required information. Once this has been done by the attacker, the victim will be in the control of the attacker.
- Server: This is the module where the unfiltered code is stored and sent to victim unknowingly.

CONCLUSION AND FUTURE SCOPE

The XSS attacks are still exploiting the web application vulnerabilities to steal the user credential. The techniques that are used to detect and prevent the XSS attack still needs more work to enhance the accuracy of XSS detection and prevention. Recently, OWASP developed a Web Application Firewall (WAF) model that can detect and prevent the Reflected XSS attack but the Stored XSS attack and the DOM-based XSS attack still requires more work.

The future work is to develop a defensive mechanism that uses data mining and machine learning techniques, to detect and prevent the Stored XSS attack and DOM based XSS attack in order to reduce the false negative and false positive.

REFERENCES

[1] Y. Minamide, “Static Approximation of Dynamically Generated Web Pages,” in WWW '05 Proceedings of the 14th International conference on World Wide, New York, NY, USA, 2005.

[2] Inian Parameshwaran , Enrico Budianto , Shweta Shinde ,Hung Dang, Atul Sadhu, Prateek Saxena “DEXTERJS: Robust Testing Platform for DOM-based XSS Vulnerabilities” 10th Joint Meeting on Foundations of Software Engineering(August 30-September 4), pp. 946-949 Bergamo, Italy, 2015.

[3] A. M. Chandrasekhar, K. Raghuveer “Intrusion Detection Technique by using K-means, Fuzzy Neural Network and SVM classifiers”, 2013 International Conference on Computer and Informatics(ICCCI),Coimbatore, INDIA, Jan04-06,2013.

[4] Punam Thopate, Purva Bamm, Apeksha Kamble, Snehal Kunjir, Prof S.M.Chawre “Cross Site Scripting Attack Detection & Prevention System” International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)Volume 3 Issue11, November 2014.

[5] Matthew Van Gundy and Hao Chen “Noncespaces: Using randomization to defeat cross-site scripting attacks” Computers & Security, No. 31, pp. 612 – 628, Elsevier, 2012.

[6] Piyushkumar A. Sonewar, Nalini A. Mhetre, "A Survey of Intrusion Detection System for Web Application", International Journal of Engineering Research and Technology Vol. 1 (02), ISSN 2278 – 0181, 2014.

[7] V. K. Malviya,S. Saurav, “On Security Issues in Web Applications through Cross Site Scripting (XSS)”,20th Asia-Pacific Software Engineering Conference,2013

[8] A. Kiezun,M. D. Ernst, “Automatic Creation of SQL Injection and Cross-Site Scripting Attacks”,ICSE, May 16- 24, 2009.

[9] Dukes, L.; Xiaohong Yuan; Akowuah, F., “A case study on web application security testing with tools and manual testing,” Southeastcon, 2013 Proceedings of IEEE , vol., no., pp.1,6, 4-7 April 2013.

[10] W. Alcorn, “Cross-site Scripting Viruses and Worms - A New Attack Vector,” Netw. Secur. Elsevier, vol. 2006, no. 7, pp. 7–8, 2006.

[7] L. Yang and D. Weng, "Snortbased Campus Network Security Intrusion Detection System Information Engineering and Applications." vol. 154, R. Zhu and Y. Ma, Eds., ed: Springer London, 2012, pp. 824- 831.

[11] T. Scholte, D. Balzarotti, and E. Kirda, “Have things changed now? An empirical study on input validation vulnerabilities in web applications,” Comput. Secur. Elsevier, vol. 31, no. 3, pp. 344–356, 2012.

[12] M.I.P. Salas and E. Martins, “Security Testing Methodology for Vulnerabilities Detection of XSS in Web Services and WS-security,” Electron. Notes Theor. Comput. Sci. Elsevier, vol. 302, pp. 133– 154, 2014.

[13] S. Gupta and B. B. Gupta, “Cross-Site Scripting (XSS) attacks and defense mechanisms: classification and state-of- theart,” Int. J. Syst. Assur. Eng. Manag. Springer, 2015.

[14] Johari, R.; Sharma, P., “A Survey on Web Application Vulnerabilities (SQLIA, XSS) Exploitation and Security Engine for SQL Injection,” Communication Systems and Network Technologies (CSNT), 2012 International Conference on, vol., no., pp.453,458, 11-13 May 2012.

[15] Hossein Jadidoleslamy “Weaknesses, Vulnerabilities And Elusion Strategies Against Intrusion Detection Systems” International Journal Of Computer Science & Engineering Survey (IJCSES) Vol.3, No.4, August 2012.