XSS Attack Prevention over Code Injection Vulnerabilities in Web Applications

K. Joylin Bala, E.Babu Raj, A. M. Anusha Bamini[3]

Abstract*:*In modern web applications Cross Site Scripting (XSS) Attacks are creating the maximum popular security harms. XSS happens at the time of accessing information or data in intermediary confidential sites. The millions of websites are attacked by this cross site scripting attacks. Malicious scripting codes are injected into applications using XSS, and also it sends back the modified code to the customer side. If the user tries to visit the web browser to check the injected portions of the malicious code, then the code will perform directly to the client's computer. The keywords of XSS was detected by using javascript engine or the malicious codes are filtered in the server side using javascript. But the attackers are creating various types of malicious codes in a very fast manner. Therefore it is difficult to detect, list key words and avoid XSS attacks. In this proposed work provide a server side solution, by providing step by step procedure for preventing XSS. The proposed detection structure recorded accuracy improvement of 10% and the false-positive rate reduction is 0.1%. By considering existing works the proposed work gives better results.

Index Terms*:*XSS, Vulnerability, Malicious, Attack, Defend, Injection.

I. INTRODUCTION

Nowadays the network usage increases rapidly, especially Web 2.0. Internet is a common platform for business, game, communication and Chat applications. Commonly Google is used as a search engine, Amazon or E-Bay is used for online shopping, My-Space is used to communicate with friends. Internet is now becoming an unavoidable part of our life. Hence it is necessary to provide a beneficial and safe networking for the users. Many users can be affected in an internet via single vulnerability. Before ten years maximum of the websites were fixed and less web based security is required. But now the hackers increases in this field. Cross site scripting attacks are the attacks in browser side scripting language. In XSS attack the malicious script is inserted, the user cant able to identify it when they are inspecting the site. This is typically done by an attacker by submitting particularly created values into the board site's URL or web forms, or somewhere. There are two types of XSS attack.

1. Misleading a user to click on a link by viewing an email or by viewing different unwanted sitesbeneathhacker'scontrol.This sites may contain advertisements, posts or image tags like <imgsrc=badcode.html/>.
2. In the next type the XSS attacks are created and stored into the server side. Such as user profiles, social sites and other forums. These attacks are in the form of self-propagating, creating and it automatically creating and replicating XSS worm.zXSS attacks are occurred in variety of forms and they are hard to identify.They are like usual human understandable text, or particularlyencryptedletterings. It is a commonly used vulnerability, but the hackers inoculate the code into the output application. The injected code will accomplishrepeatedly or rip-off sensitive data from the people input. Same SQL Injection code is used. Stored XSS, Reflected XSS and Dom- based XSS are the three common techniques used in Web Application Security.

II.LITERATURE REVIEW

To prevent XSS attacks many solutions have been proposed.AntiSamy is one of the conceptvia OWSAP (Open Web Application Security Project) to prevent XSS. Here many APIs are used tom filter and validate the input beside the attack for input validation and output encoding. The tool practicesNekoHTML and Policy file aimed at authenticating HTML plus CSS inputs. Given HTML to XML document is parsed using NekoHTML. Collective attributes, regular expressions, general tag and commands are the general rules aimed at authentication. It should stand changed according to the need of web administrator. The technique of HTML Purifier remains a HTML filter library standards engraved in PHP. By conducting absolute analysis the HTML Purifier will reject all certain malicious code. In XSS-GUARD, major hint for distinguishing among gentle and malicious content to produce anidentical answer for each HTTP response formedvia the web application. The reason at back of generation ofreplacementreply to generate the preferredregular set of accredited script progression equivalent to the HTTP response. The set of scripts present in real HTTP response is identified using XSS-GUARD. The modified web browser code is createdfor identification of malicious script. XSS-GUARD verify for occurrence of script which is not accreditedvia web application. This is made by using duplicate response that contains the scripts which are projected by web application.

Revised Manuscript Received on June 05, 2019

K.JoylinBala,,Research Scholar, Department of Computer Science, Bharathiar University, India.

E. Babu Raj, Professor,Department of Computer Science and Engineering, Marian Engineering College, Thiruvananthapuram,India.

A.M.AnushaBamini, Assistant Professor,Noorul Islam Centre for Higher Education, India.

Also occurrence of proposed script remains nothing but XSS attack vector, hence XSS-GUARD will take away those scripts and send the resulting response to the client [18].

Using various analysis deDacota gives better approach to automatically take apart of code and data in a web application. Consuming statistical information it's intend is to renovate a certain web application to the novel description by means of statically transformation. To preserve the semantics of application and yield web pages it will take all its inline JavaScript code which is conveyed to external JavaScript file. These JavaScript files are simply basis of scripts implemented by web application observing to Content Security Policy (CSP). The remaining JavaScripts are unobserved at the time of interpreting the webpage [16].

The elevated performance and high reliability is achieved in XSS auditor by making the boundary among the browser HTML parser and JavaScript engine. By default Google Chrome is enabled and embedded with this security implementation. XSS Auditors Post-parser scrutinizes the semantics of HTTP response, as interpreted by the browser, without the need for an error-prone time overriding simulation. This will block the malicious attacks and avoid injected script from the JavaScript engine rather than risk making modifications in the HTML code as revealed in [14].

II. PROBLEM STATEMENT

At the time of transferring data from client side to server side lot of script changes are done in the original code to hack the data. Therefore it important to propose a better mechanism for perverting this attack. Lot of prevention techniques are proposed earlier in the survey. Therefore the proposed scheme concentrates in preventing attacks in server side. The enormous modifications are performed in the proposed XSS attack prevention method. The drawbacks in the existing technique are overcome in this proposed model. The major challenge in designing a better solution is, the proposed technique must block each and every insertion of malicious script injection in addition with new attacks. To filter the malicious script various encoding techniques are applied. This XSS is detected by the proposed code injection mechanism. The server side solution for XSS attack is provided using the injection of java script code. At the time of preventing server side data the response time of accessing data should be decreased. And also, the problem is in the accuracy, false positive rate and vulnerability detection.

III. PROPOSED SOLUTION

The proposed modified web application will separate the user provided contents in different way. By inserting boundary tag the original content of web application is hided. At first the user inserted data is extracted from the reply and then the malicious scripts is checked as well as altered. Afterwards the filtered data is fixed in user response. The below proposed algorithm 1 shows the server service filtration procedure. This will filter the unwanted cross sited scripts from the server.

Algorithm: Server Service Filtration Algorithm

Step 1: Remove different encodings using preprocessing.
Step 2: Parse the content by Jsoup HTML parser to DOM.
Step 3: Search event attributes in the resulting DOM
Step 3.1: If the attributes access any DOM properties procured in section
Step 3.2:Then filtered out the event attributes from content.
Step 4: Then presence of attributes is searched from filtered DOM.
Step 5: The filtered DOM is verified in embedded CSS in <style>tag and styleattribute.
If the CSS is having malicious scripts then these CSS contents are dropped otherwiserecollected in DOM.
Step 6: The DOM is verified for <svg>tag and if it mention to some active HTMLcontent then the attribute of <svg>tag is blocked.
Step 7: Then the filtered DOM is patterned for occurrence of grouping the tag-attribute pair.
Step 8: The data URI is searched in the resulted DOM. The values present in the DOM altered with suitable decoding scheme and checked for malicious code.
Step 9: The special tags are checked from the resultedDOM.

The below figure 1, shows the flowchart of detecting XSS attacks in step by step procedure. First step is to start the web server. Then enter the URL address of victim server. If the victim comes across its login page, then submit the details of the web server. The web server transfer the cookies to victim's browser. Victim cookie will transfer the service to attack. XSS has exploited on the victim web browser.

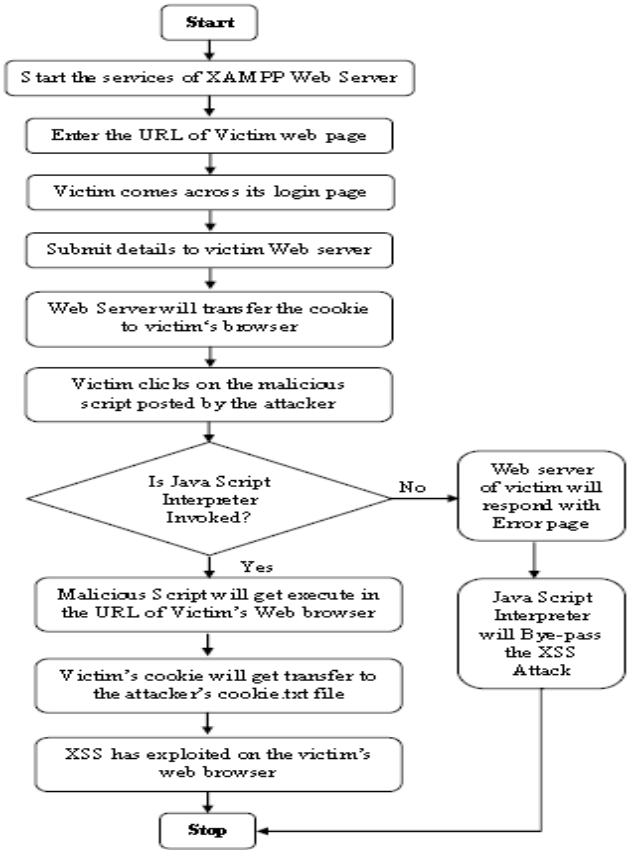

Figure 1. Flowchart to find victim server

IV. MITIGATING CROSS-SITE SCRIPTING ATTACKS

The proposed technique differs from the general methods.

- The XSS payload type is analyzed first.
- Isolate the minimal technical capabilities from the systematical analysis, to posse's payload to function.
- Finally, the approaches to remove the attacker are identified for the capabilities.

The SID should be protected from the javascript without embedding it into web page. To protect this SID, it is stored in cookies and these cookies are kept in different domain. Over the secure domain we used two modest server scripts: getCookie.ext and setCookie.ext. The cookie data are transported using these two scripts.

A) Getting the cookie data

From client side to server side the procedure of transferring an existing cookie is conventionalone. For the subsequent situation the client web browser previously retains a cookie for the domain secure.example.org. The cookies are set for loading the webpage. It comprises of the subsequent steps.

1. First request RQ1 is generated by the client's web browser. For example www.example.org/index.ext.
2. For this request RQ1 the web server answers with a small HTML page that consists of the PageLoader. This request is named as RP1.
3. In the DOM tree of web page the PageLoadercomprises the getCookie.ext image. From the server, request for image is generated to the client's web browser known as RQ2. The cookie comprising the SID which is recorded in secure.example.org.
4. The webpage's body is requested by the PageLoadervia the XMLHttpRequest called RQ3. In parallel the HTTP request also generated. The HTTP request is termed asgetCookie.ext image.
5. Web server generate the response to RQ3 and wait for receiving the next request called getCookie.ext image. Based on cookies data the request is processed and the web server to calculate and propel the webpage's body (RP2).
6. The body of the webpage is received byPageLoader and it uses the document.write technique to demonstrate the data.

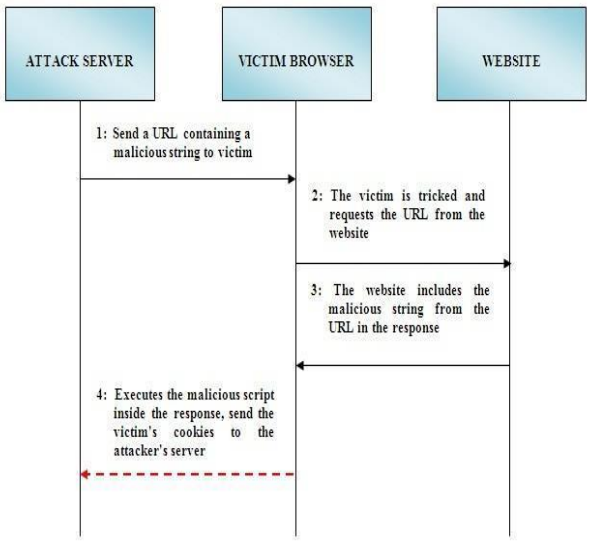

Figure 2. Getting cookies

B)Setting a cookie

The frequently used procedure of exchanging cookies data from client side to server side islabeled previously. The previously mentioned steps are simplified into following steps to set the cookies. The steps followed to set cookies are,

1. The HTTP request is send from the client web browser to the server known as RQ1. Example: www.example.org/index.ext (RQ1).
2. In second step the web servers accepts the request and make a replies to the PageLoader. After that the PageLoader consequently requests the complete data by sending the request RQ2.
3. Third request RQ2 computed by the web server. The server decides to set cookies because of the conclusion of the computation. The server answers with "SETCOOKIE" to the PageLoader's demand for the data.
4. The SETCOOKIE token is received from PageLoader and it is included in the setCookie.ext image in webpage, which is having DOM tree. Client web browser generates a call called RQ3 and ie requests the image from the server.
5. PageLoader creates a new request called RQ4 and it is requested by webpage's body. This request of HTTP is in the form of setCookie.ext image.
6. The new request received by the web server known as RP3 which receives the demand for image as well as cookie data.
7. The web server accepts RP3 and wait for the response to RQ4 till the setCookie.extimage is effectively delivered to client. The image request data after processing will be sent through the request RP4.

VI. PERFORMANCE ANALYSIS

Based on the implementation of proposed work the metrics considered are,

a) Number of vulnerability detected

b) Accuracy

c) False positive rate

A)Number of vulnerability detected

It is the standard used to find theadvanced application's abilities to determine XSS threads in requests.

B)Accuracy

It is a measure of test searching performed on established structure followed by the regular dataset.The Accuracy can be denoted by,

Accuracy = $\frac{TN+TP}{TN+FP+FN+TP}$

Where TN = true negative, TP = true positive, FP = false positive, FN = false negative.

C) False-Positive Rate

The FPR is considered the inability of the system where the number of negative occasions wrongly predicted as positive .

FP= $\frac{FP}{(FP+TN)}n$

In a trial website 100 visits are monitored. Time taken for estimation is 240 seconds.Specific websites file with more than one paths, it makecertain websites scan quicker than others.The selected method is well suited of most of the applications. Number of classifications detected for every website above a period of 100 visits is shown in Figure 3.

Table 1. Comparison of performance metrics over various methods

Parameter	Netsparker	Acunetix	Webcurier	Proposed method
Vulnerability detection	15	19	14	22
Accuracy	56	59	40	85
False positive	3	6	2	0.99

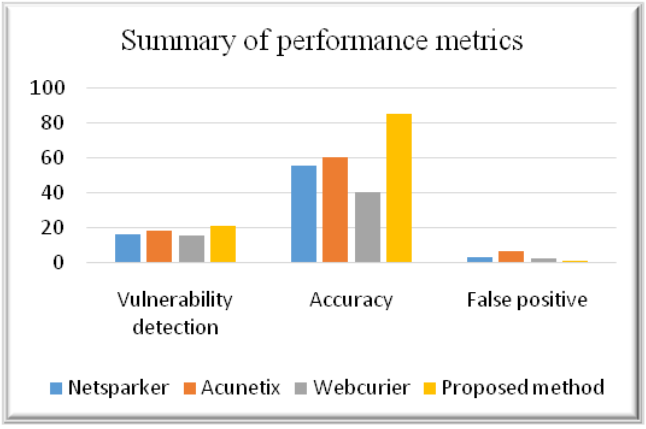

FIGURE 5. PERFORMANCE ANALYSIS

VII.CONCLUSION AND FUTURE ENHANCEMENT

Proposed work plays a major role in reducing XSS attack. From the comparative analysis of previous work, it is noticed that the proposed method outperforms than former web vulnerability scanners in relationships with accuracy, vulnerability detection and false-positive rate. It is identified the number of vulnerabilities noticed are greater than the existing methods. Input encoding can be protected beside more than just XSS. Before storing information in a database SQL injection and command injection can also be determined. Output encoding also plays a vital role and can be taken into consideration. The proposed detection structure recorded 10% enhancement of accuracy and 0.1% decrease the false-positive rate that is significantly less than the available method.

Future research work XSS data can be placed into the HTML document straight way by the server, and distributed to the end user with the XSS complete. Different the vulnerability described from the given terminology, server-side XSS authorizations will be further prevent this type of attack. The proposed method can be combined with soft computing methods to enhance the performance of the system.

REFERENCES

1. Gupta S, Gupta B, "Automated discovery of JavaScript code injection attacks in PHP web applications", *ICIS*, vol. 78, pp. 82-87, 11–12 Dec 2015.
2. Sharma P, Johari R, and Sarma S, "Integrated approach to prevent SQL injection attack and reflected cross site scripting attack," Int Jour of System Assurance Engineering and Management, vol. 3, no. 4, pp. 343–351, 2012.
3. Martin and Justus Winter, "Protecting the Intranet against JavaScript Malware and Related Attacks", volume 4579 of LNCS, pp. 40-59. Springer, July 2007.
4. Isatou, S. Abubakr, Z. Hazura, and A. Novia, "An approach for cross site scripting detection and removal based on genetic algorithms," in Proceedings of the Ninth International Conference on Software Engineering Advances: France, pp. 227–232, Nice, France, October 2014.
5. Ishikawa Tomohisa, Kouichi Sakurai, "Parameter manipulation attack prevention and detection by using web application deception proxy",Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication ACM, *Jan 2017*.
6. *NITHYA, P. LAKSHMANA, AND C. MALARVIZHI*, "A SURVEY ON DETECTION AND PREVENTION OF CROSS-SITE SCRIPTING ATTACK," INTERNATIONAL JOURNAL OF SECURITY AND ITS APPLICATIONS, VOL. 9, NO. 3, PP. 139–152, 2015.
7. S.A. Gadhiya,K.H. Wandra , "The Research Perspective: XSS Attack and Prevention of XSS Vulnerability in Web Application", International Journal of Engineering Development and Research (IJEDR), ISSN:2321-9939, Vol.2, Issue 4, pp.3738-3741, Dec 2014.
8. P. S. Georgios and K. K. Sokratis, "Using fuzzy inference system to reduce false positive in intrusion detection," Elsevier Computer and Security Conference, vol. 29, no. 1, pp. 35–44, 2009.
9. Singh A, "A Survey on XSS web-attack and Defense Mechanisms", *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 4, no. 3, pp. 1160-1164, March 2014.
10. A AuardoNunan, "Automatic Classification of Cross-Site Scripting in Web Pages Using Document-based and URL-based Features", *IEEE Symposium on Computers and Communications (ISCC)*, pp. 000702-000707, July 2012.

11. B. Animesh and M. Debasish, "Genetic algorithm based hybrid fuzzy system for assessing morningness," Advances in Fuzzy Systems, vol. 2014, Article ID 732831, 9 pages, 2014.
12. A.M. AnushaBamini and Sharmini Enoch, "Dynamic Scheduling and Resource AllocationinCloud",InternationalJournalofControlTheoryandApplications, Volume 10, No. 3, pp. 63-72, 2017.
13. BakareK.Ayeni, "Detecting Cross-Site Scripting in Web Applications Using Fuzzy Inference System", Journal of Computer Networks and Communications, Volume 2018, https://doi.org/10.1155/2018/815
14. Klein, A., "DOM Based Cross Site Scripting or XSS of the Third Kind," Technical paper - Web Application Security Consortium, Describes DOM-based (Type 0) XSS.D.
15. Vandana, Y. Himanshu, and J. Anurag, "A survey on web application vulnerabilities," International Journal of Computer Applications, vol. 108, no. 1, pp. 25–31, 2014.
16. S.A. Gadhiya,K.H. Wandra , "The Research Perspective: XSS Attack and Prevention of XSS Vulnerability in Web Application", International Journal of Engineering Development and Research (IJEDR), ISSN:2321-9939, Vol.2, Issue 4, pp.3738-3741, Dec 2014
17. Artificial Neural Network Applications in Machining Process: A Review, International Journal of Control Theory and Applications, Vol 10, No 27,2017.

a. *Saleh, B. Rozalia, "A method for web application vulnerabilities detection by using Boyer-Moore string matching algorithm," Information Systems International Conference, vol. 72, no. 3, p. 112, 2015.*

18. A.M. AnushaBamini and Sharmini Enoch, "Optimized Scheduling and ResourceAllocationusingEvolutionaryAlgorithmsinCloudEnvironment",InternationalJournal of Intelligent Engineering and Systems, Volume 10, No. 5, pp. 125-133, 2017.
19. S.A. Gadhiya,K.H. Wandra , "The Research Perspective: XSS Attack and Prevention of XSS Vulnerability in Web Application", International Journal of Engineering Development and Research (IJEDR), ISSN:2321-9939, Vol.2, Issue 4, pp.3738-3741, Dec 2014