Development of Mathematical Algorithm for Detecting XSS Attacks on Web Applications

KOMIL FIKRATOVICH KERIMOV[1], ZARINA ILDAROVNA AZIZOVA[2]
[1]Department of System and Application Programming,
Tashkent University of Information Technologies named after Muhammad al-Khwarizmi,
108 Amir Temur Avenue, Yunusabad paradise, 100084, Tashkent,
REPUBLIC OF UZBEKISTAN

[2]Department of Information Security,
Tashkent University of Information Technologies named after Muhammad al-Khwarizmi,
108 Amir Temur Avenue, Yunusabad paradise, 100084, Tashkent,
REPUBLIC OF UZBEKISTAN

Abstract: - The widespread usage of web applications has led to an increase in security threats, with Cross-Site Scripting (XSS) attacks being one of the most prevalent and damaging. Detecting and mitigating XSS attacks is crucial to ensure the integrity and confidentiality of sensitive user data. This article presents the mathematical algorithm and a way to identify XSS attacks using a bounded function from below, which depends on the input string, and highlights its potential impact in bolstering web application security. To construct this function, we used special characters and keywords that are frequently found in the construction of XSS attacks.

Key-Words: - XSS, vulnerability, signatures, information security, artifacts, testing.

Received: March 2, 2024. Revised: August 26, 2024. Accepted: October 2, 2024. Published: November 5, 2024.

1 Introduction

Nowadays the internet has become an integral part of our daily lives, with web applications serving as the backbone of various online services. Web applications are increasingly targeted by malicious actors, particularly through Cross-Site Scripting (XSS) attacks. These attacks involve injecting harmful scripts into web pages, enabling data theft, session manipulation, and further exploitation. Successful XSS attacks can result in financial losses, reputational damage, and compromised trust.

However, this increased reliance on web applications has also made them attractive targets for malicious actors seeking to exploit vulnerabilities and compromise user data. Among the various attack vectors, Cross-Site Scripting (XSS) attacks have emerged as a significant security concern.

Traditional approaches to detecting XSS attacks have relied on signature-based methods or pattern-matching techniques. While these techniques can be effective to a certain extent, they often struggle to detect sophisticated and obfuscated XSS payloads. Furthermore, the dynamic nature of modern web applications poses challenges for static analysis-based detection methods. To address these limitations, we propose a novel mathematical algorithm for detecting XSS attacks on web applications. Our approach combines the principles of XSS vulnerability detection based on the specific weight of each attack symbol signature.

In this paper, we propose a mathematical modeling and a method to identify XSS attacks using a bounded function from below, which depends on the input string. To construct this function, we used special characters and keywords that are often found in the construction of XSS attacks. In the proposed method, it is possible to detect XSS attacks using a single special character or a single keyword. However, it can be shown experimentally that the proposed detection method using a set of multiple characters and words can detect the vulnerability of a type of XSS attack more accurately.

1.1 Related Research Papers on XSS Attack Detection

Article [1] proposes a machine learning approach for detecting XSS attacks, which relies on diverse and representative training data for reliable results. Static analysis methods [2] detect XSS vulnerabilities during development but require manual verification and adjustment of rules for improved accuracy. Combining dynamic analysis and machine learning [3] enhances detection accuracy but poses challenges in scalability and adapting to evolving attack techniques.

Applying NLP techniques to enhance XSS detection shows promise but faces challenges with obfuscated payloads, [4]. Symbolic execution detects DOM-based XSS attacks by analyzing JavaScript code but can be computationally expensive and requires careful handling of complex code, [5]. According to [6], signature-based detection systems are based on predefined patterns or signatures of known XSS attacks. Although signature-based methods are optimal and, under certain conditions, effective in detecting known attacks when the signature database is regularly updated, they are ineffective in detecting new or evolving attack patterns. Regular updates and customization of WAF rules based on known attack patterns are essential [7]. XSS attacks inject malicious code into HTTP requests. Existing prevention techniques may not work against unknown attacks, emphasizing the need for new prevention methods, [8]. In [9], a mathematical method for detecting XSS-type vulnerabilities is proposed based on the specific weight of each attack symbol signature.

These research works provide valuable insights into the different approaches and techniques for XSS attack detection. However, it is important to consider the limitations, challenges, and potential trade-offs associated with each approach.

Also noteworthy is the work [10], which presents a method based on the subsequence parliamentary algorithm. The key diagram of the algorithm for compiling a sequence in a common substring between the input parameters and the generated data, with setting a threshold to limit the length of the common substring. In addition, the analysis of the advantages and disadvantages of the static and algorithmic XSS detection methods used in the article carries valuable information load.

The paper [11] described a web vulnerability assessment methodology and a risk distribution model of vulnerabilities, supplemented by a survey of web developers. The results of the testing and the survey allowed us to define a "triangular model" of the causes and consequences of web vulnerabilities, as well as the necessary measures to reduce their number and frequency. In turn, in [12], the author's attention is drawn to the use of methods for detecting XSS attacks by applying various tools, including the XSS Detector developed by them.

The difference between our proposed method is that it is based on processing existing types of XSS vulnerability types. A mathematical algorithm for detecting attacks on a web application using information related to the frequency and importance factor of attack symbols does not require complex mathematical techniques like optimization etc. Since the proposed method is based on the processing of available types of real-world vulnerabilities of XSS kind, and is general in nature, we can expect that our proposed algorithm can be used as an online cross-site scripting attack detection platform.

The proposed method involves creating a character set that distinguishes between attack and normal strings using a predefined threshold. Experiments with synthetic data showed that a character set containing a space, a semicolon, and a right parenthesis performed well over a range of weights for attack and normal strings. However, it should also be noted that there are problems in collecting samples of real attacks and conventional data, as well as in capturing the entire pattern of attack methods.

2 Mathematical Algorithm for Detecting XSS Attacks

To ensure the protection of web resources, XSS attack detection is a relevant solution. XSS attacks actively exploit vulnerabilities in web applications to execute malicious scripts, which affects the integrity of system and user data. This section contains information about XSS attack detection methods that are aimed at improving the security level of web applications.

A comprehensive approach to protecting web applications includes several methods for effectively detecting XSS attacks. Static analysis allows you to identify vulnerabilities in a web application at the development stage, while dynamic analysis and machine learning-based detection methods allow you to observe changes in real time and regularly monitor application performance. Signature-based XSS attacks and WAF detection using strictly defined rules also allow to monitor network traffic. Combining these methods allows you to implement in practice an effective and reliable strategy of defence against XSS attacks.

In this paper, to validate the proposed algorithm, we need to determine whether the input string is an XSS attack or not, namely to model an information object, i.e., a query that consists of a sequence of special characters from Table 1 and keywords from Table 2 respectively.

The following characters and keywords are often used to construct XSS attacks:

Table 1. Special characters for defining XSS attacks

Variable	Special symbols	Importance ratio	Variable	Special symbols	Importance ratio
A_1	“	0,561	A_{17}	}	0,043
A_2	>	0,331	A_{18}	#	0,552
A_3	/	0,511	A_{19}	+	0,033
A_4	<	0,331	A_{20}	!	0,047
A_5	Spacebar	0,485	A_{21}	,	0,067
A_6	=	0,609	A_{22}	@	0,017
A_7	‘	0,318	A_{23}	?	0,047
A_8	:	0,174	A_{24}	[	0,013
A_9	.	0,997	A_{25}	]	0,013
A_{10}	(	0,532	A_{26}	-	0,144
A_{11}	)	0,532	A_{27}	~	0,003
A_{12}	-	0,144	A_{28}	*	0,023
A_{13}	;	0,622	A_{29}	|	0,003
A_{14}	(yen sign)	0,023	A_{30}	^	0,003
A_{15}	&	0,622	A_{31}	%	0,037
A_{16}	{	0,043	A_{32}	$	0,003

Table 2. Special keywords for defining XSS attacks

Variable	Key -words	Importance ratio	Variable	Key-words	Importance ratio
A_{33}	FScommand	0,003	A_{56}	Jscript	0,171
A_{34}	&lt;script&gt;	0,074	A_{57}	Wscript	0,012
A_{35}	&lt;/script&gt;	0,164	A_{58}	Vbscript	0,003
A_{36}	on\w*	0,003	A_{59}	Vbs	0,003
A_{37}	style	0,097	A_{60}	Ilayer	0
A_{38}	xmlns:xdp	0,003	A_{61}	Iframe	0,015
A_{39}	Formaction	0,010	A_{62}	Applescript	0
A_{40}	Form	0,020	A_{63}	Jar	0
A_{41}	xlink:href	0,003	A_{64}	Eval	0,006
A_{42}	seekSegment Time	0,003	A_{65}	Document	0,064
A_{43}	FSCommand	0,003	A_{66}	base64	0,018
A_{44}	Applet	0,003	A_{67}	</script>	0,164
A_{45}	Audio	0,003	A_{68}	<script	0,161
A_{46}	Basefont	0,030	A_{69}	Keygen	0
A_{47}	Base	0,023	A_{70}	1Object	0,012
A_{48}	Behavior	0,003	A_{71}	Plaintext	0
A_{49}	Bgsound	0,003	A_{72}	Mochae	0
A_{50}	Blink	0	A_{73}	Style	0,097
A_{51}	view-source	0	A_{74}	Javascript	0,171
A_{52}	Embed	0,020	A_{75}	Xml	0,047
A_{53}	Livescript	0	A_{76}	Math	0
A_{54}	Mocha	0,003	A_{77}	Source	0,006
A_{55}	Bechavior	0,003	A_{78}	Svg	0,017

Suppose some input string is observed L and let $x_1, x_2, ..., x_{32}$ the frequency of occurrence in L pecial characters from Table 1 and let $x_{33}, x_{34}, ..., x_{78}$ the frequency of occurrence of special keywords from Table 2, x_{79} frequency of occurrence of all other signs and numbers 0,1,2,...,9 in the line L. From the point of view of defining XSS attacks, common characters $a, b, ..., z$ and numbers 0, 1, … , 9 do not play an important role. So in this paper, we always assume that the frequency of occurrence of all these symbols and numbers in the observed string L is equal to1, i.e. $x_{79} \ = 1$. In this way, any string L can be defined with certain characteristics as follows: $L = (x_1, x_2, ..., x_{32}, x_{33}, ..., x_{78}, x_{79})$, as an element of some phase space X. From the definition L it is seen that any element L from the constructed space X lies on the hyperplane $G = \{L = (x_1, x_2, ..., x_{32}, x_{33} ..., x_{78}, x_{79}) : x_{79} = 1\}$.

Using this hyperplane equation, we can assume that the greater the frequency of occurrence of special characters and keywords in the input string, the more obvious is the proximity of the input string L to XSS attacks. Therefore, it is natural to assume that the attack definition function should be increasing in the variables $x_1, x_2, ..., x_{32}, x_{33}, ..., x_{78}$, and decreasing in the variable x_{79} . Based on these considerations, the following function is proposed to define XSS attacks, which is an increasing function on the variables $x_1, x_2, ..., x_{32}, x_{33}, ..., x_{78}$ and decreasing in x_{79} :

$$f(L) = f(x_1, x_2, ..., x_{32}, x_{33}, ..., x_{78}, x_{79}) = \frac{\sum_{i=1}^{78} x_i}{\sum_{i=1}^{78} x_i + x_{79}}$$

As in this paper, we always assume that the frequency of occurrence of all other characters and numbers 0,1, 2,..., 9 in the line L is equal to 1, then from the last equality we obtain

$$f(L) = f(x_1, x_2, ..., x_{32}, x_{33}, ..., x_{78}, x_{79}) = \frac{\sum_{i=1}^{78} x_i}{\sum_{i=1}^{78} x_i + 1} \qquad (1)$$

If the input string is L is an XSS attack, then this string must at least contain one special character from Table 1 or one keyword from Table 2. Therefore $\sum_{i=1}^{78} x_i \geq 1$ and because the function $f(L)$ is increasing for each of the variables x_i its minimum at $\sum_{i=1}^{78} x_i \geq 1$ is reached at the point L_0

$\sum_{i=1}^{78} x_i = 1$ for which . Thus, if L random string and $f(L) \geq 1/2$, then L is probably an XSS attack, in which case it is built using a minimum of either 1 a special character from Table 1 and 1 keyword from Table 2 are used to build it, or 1 keyword from Table 2. If, on the other hand $f(L) < 1/2$ then the input string is probably normal. Therefore, function (1) can be used to recognize XSS attacks and normal strings constructed with special characters and keywords.

Further, let us refine the algorithm by using the importance coefficients of special characters in the construction of the recognition function. To do this, let us construct the function:

$$f_1(L) = f_1(x_1, x_2, ..., x_{32}, x_{33}, ..., x_{78}, x_{79}) = \frac{\sum_{i=1}^{78} k_i x_i}{\sum_{i=1}^{78} k_i x_i + 1} \quad (2)$$

where $k_i, 0 < k_i < 1,$ – sign importance coefficients from Table 1. The importance coefficients are calculated by examining 299 pieces of real XSS attacks. Using the values of importance coefficients and the form of function (2), it is easy to determine the minimum of the new function $f_1(L)$ under the condition $\sum_{i=1}^{78} x_i \geq 0{,}003$. The minimum of this function is reached at the point L_0, for the coordinates of which the equality $\sum_{i=1}^{78} x_i = 0{,}003$. Thus, from the new function $f_1(L)$ we have that if L arbitrary string and $f_1(L) \geq 0{,}003$ then L is probably an XSS attack, and in this case at least 1 special character from Table 1 and 1 keyword from Table 2 are used to build it, or 1 keyword from Table 2. If, however $f(L) < 0{,}003$ then the input string is normal. Thus, function (2) can be used to detect XSS attacks and regular strings formed with special characters and keywords.

The advantage of our proposed mathematical algorithm is its adaptability. It can be subjected to training and tuning based on both previously known data and real-time data, which leads to its continuous improvement and enhancement of XSS attack detection capabilities. To improve the accuracy of attack detection and reduce the number of false positives and false negatives, a complete dataset containing different scenarios and focus of XSS attacks is required.

This algorithm can be used in conjunction with other web application security methods, including input validation, content content security policy, etc. The proposed method does not require high computing power as the mathematical calculations are not difficult. Based on the application of static real-time processing of input data, the importance coefficients of the available special characters and keywords used in the construction of XSS attacks were obtained. The results are presented in Tables 1 and 2, respectively. The obtained results, taking into account the importance coefficients of special characters, were necessary to demonstrate the effectiveness of the new XSS attack recognition algorithm. Based on the available characteristics of XSS attacks using special characters and special keywords, an information object in the form of queries was modeled in the process. Thus this method can be used as an online tool to detect XSS attacks.

3 Conclusion

The mathematical algorithm for detecting XSS attacks on web applications, taking into account the frequency of occurrence and importance coefficient of characters involved in the construction of incoming requests presented in this research leverages various mathematical techniques and data analysis methodologies to identify potential XSS vulnerabilities. In this case, from Table 1 and Table 2 we can see that the importance coefficients for special characters (Table 1) have a significant difference, and the importance coefficients for keywords are very close to zero. Therefore, if we do not take into account some parameters with zero coefficients when constructing functions (1), and (2), the recognition of incoming queries is possible with some small error.

In conclusion, a mathematical algorithm for detecting XSS attacks on web applications is a promising approach to improving the security of web applications. Using mathematical principles and data analysis techniques, it provides an adaptable and systematic method for identifying potential XSS vulnerabilities. However, it must be used as part of a comprehensive security framework to provide a robust defense against XSS attacks.

Our further research will focus on addressing the challenges of collecting real-world attack samples and conventional data and the development of a software tool based on the proposed algorithm and its evaluation. Further research and evaluation is needed to confirm the validity of the chosen method.

References:

[1] Banerjee Raima, Baksi Aritra, Singh Nidhi, Bishnu Sohan Kanti, Detection of XSS in web applications using Machine Learning Classifiers, *International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech) 2020 4th*, Kolkata, 2020, pp. 1-5, doi: 10.1109/IEMENTech51367.2020.9270052.

[2] Monali Shetty, Chirantar Nalawade, Hybrid approach for Detection and Analysis of SQL and XSS vulnerabilities, *International Journal of Engineering Trends and Technology*, Vol.59, 2018, pp. 37-41, doi: 10.14445/22315381/IJETT-V59P206.

[3] Wassermann Gary, Su Zhendong, Static detection of cross-site scripting vulnerabilities, *2008 ACM/IEEE 30th International Conference on Software Engineering*, Leipzig, Germany, 2008, pp. 171-180, doi: 10.1145/1368088.1368112.

[4] Fawaz Mokbal, Dan Wang, Xiaoxi Wang, (2022). Detect Cross-Site Scripting Attacks Using Average Word Embedding and Support Vector Machine, *International Journal of Network Security,* Vol.24, No.20-28, doi: 10.6633/IJNS.202201_24(1).03.

[5] Sanjukta Mohanty, Arup Abhinna Acharya, Detection of XSS Vulnerabilities of Web Application Using Security Testing Approaches, *Intelligent and Cloud Computing, 2021,* pp.267-275, doi: 10.1007/978-981-15-6202-0_27.

[6] Jasleen Kaur, Urvashi Garg, Gourav Bathla, Detection of cross-site scripting (XSS) attacks using machine learning techniques: a review. *Artificial Intelligence Review*, Vol. 56, 2023, pp. 1-45, doi: 10.1007/s10462-023-10433-3.

[7] Garn Bernhard, Sebastian Lang Daniel, Leithner Manuel, Richard Kuhn D., Kacker Raghu, Simos Dimitris, Combinatorially XSSing Web Application Firewalls, *2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW),* Porto de Galinhas, Brazil, 2021, pp. 85-94, doi: 10.1109/ICSTW52544.2021.00026.

[8] Domarev V.V., *Security of information technologies. Methodology of creation of protection systems*, DiaSoft, 2002. - 688 p., ISBN 966-7992-02-0.

[9] Sonoda Michio, Matsuda Takeshi, Koizumi Daiki, Hirasawa Shigeichi, On Automatic Detection of SQL Injection Attacks by the Feature Extraction of the Single Character, *Proceeding in the 4th International Conference on Security of Information and Networks*, SIN 2011, Sydney, NSW, Australia, , 2011, pp.81-86, https://doi.org/10.1145/2070425.2070440.

[10] Zhang Jingyu, Hu Hongchao, Huo Shumin an Li Huanruo, A XSS Attack Detection Method Based on Subsequence Matching Algorithm, *Proceeding in 2021 IEEE International Conference on Artificial Intelligence and Industrial Design (AIID),* Guangzhou, 2021, pp. 83-86, doi: https://doi.org/10.1109/AIID51893.2021.9456515.

[11] Blerim Rexha, Arbnor Halili, Korab Rrmoku and Dren Imeraj, Impact of secure programming on web application vulnerabilities, *Proceeding in 2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS),* Bhubaneswar, India, 2015, pp. 61-66, doi: https://doi.org/10.1109/CGVIS.2015.7449894 .

[12] D. Azshwanth, G. Sujatha, A novel automated method to detect XSS vulnerability in webpages, *Proceeding in 2022 International Conference on Computer Communication and Informatics (ICCCI)*, Coimbatore, India, 2022, pp. 1-4, doi: https://doi.org/10.1109/ICCCI54379.2022.9740937.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)