ENHANCING XSS VULNERABILITY DETECTION AND REMOVAL IN WEB APPLICATIONS USING GENETIC ALGORITHMS

By

ISATOU HYDARA

Thesis Submitted to the School of Graduate Studies, Universiti Putra Malaysia, in Fulfilment of the Requirements for the Degree of Doctor of Philosophy

February 2024

FSKTM 2024 7

DEDICATION

To my mother Oumie Kah and late father Sheriff Sulayman Hydara who raised me with unconditional love.

To my wonderful husband, Lamin Saidykhan, who supported and encouraged me throughout this research journey.

To my dear brother, Ebrima Hydara, and my sister, Ndey Awa Hydara, for their endless love, prayers and support.

And to everyone else who has provided me guidance, support, and encouragement.

Abstract of thesis presented to the Senate of Universiti Putra Malaysia in fulfilment of the requirement for the degree of Doctor of Philosophy

ENHANCING XSS VULNERABILITY DETECTION AND REMOVAL IN WEB APPLICATIONS USING GENETIC ALGORITHMS

By

ISATOU HYDARA

February 2024

Chairman : Professor Abu Bakar bin Md Sultan, PhD
Faculty : Computer Science and Information Technology

Cross-site scripting (XSS) vulnerabilities are a major security threat for both desktop and mobile web applications. They occur due to lack of proper verification of the user inputs, which enables hackers to inject and execute malicious scripts in the web pages of an application. Successful XSS attacks can lead to serious security violations such as account hijacking, denial of service, cookie theft, and web content manipulations. Current approaches to addressing this problem are limited by large number of false positives in their analysis results, non-inclusion of all types of XSS, lack of focus on removing XSS vulnerabilities, and non-inclusion of mobile web applications.

Static analysis techniques are good at detecting XSS vulnerabilities in the source codes of web applications, and especially when combined with other techniques. However, they tend to generate a lot of false positives since they are conservative techniques. Another limitation is the limited or lack of focus on the removal of XSS vulnerabilities after their detection in the source code. Consequently, an approach

called XSS-DETREM has been proposed with the objectives of combining genetic algorithms with static analysis, and a code replacement technique to detect and remove XSS vulnerabilities, respectively, to address the problem of XSS at the source code level. The research used a quantitative research methodology and randomised complete block design in the experimentation design whereby new improvements were implemented in a software tool.

XSS-DETREM has been evaluated empirically using a data set of JSP and Android web applications that have been used in previous studies. Comparisons of the evaluation results have shown improvements in the detection and removal of XSS vulnerabilities in desktop and mobile web applications. These improvements focused on reducing the rate of false positives generated by static analysis, increasing the vulnerability coverage for all types of XSS on both the server-side and client-side. Consequently, the objectives of the research have been met and the expected results were achieved. This new improved approach is significant in helping web application developers to test their applications for all types of XSS and remove any detected vulnerabilities before releasing them to the public. Also, as more users are browsing the Internet through their mobile applications, this approach will help in protecting their private data and make browsing safer for them with both Desktop and Mobile web applications.

Keywords: cross-site scripting attack; cross-site scripting vulnerability, software security, XSS vulnerability detection

SDG: GOAL 4: Quality Education

Abstrak tesis yang dikemukakan kepada Senat Universiti Putra Malaysia sebagai memenuhi keperluan untuk ijazah Doktor Falsafah

MEMPERTINGKATKAN PENGESANAN DAN PENGHAPUSAN KERENTANAN PENSKRIPAN SILANG-TAPAK KE ATAS APPLIKASI WEB MENGGUNAKAN ALGORITMA GENETIK

Oleh

ISATOU HYDARA

Februari 2024

Pengerusi : Profesor Abu Bakar bin Md Sultan, PhD
Fakulti : Sains Komputer dan Tekknologi Maklumat

Kerentanan pengskripan silang-tapak (XSS) merupakan ancaman keselamatan utama kepada aplikasi web dan desktop. Ianya berlaku disebabkan kelemahan verifikasi input yang betul, membolehkan penggodam untuk menyuntik dan melaksanakan skrip jahat kepada aplikasi di tapak web. Serangan XSS yang berjaya boleh membawa kepada pelanggaran keselamatan yang serius seperti rampasan akaun, penafian servis, kecurian kuki dan pemanipulasian kandungan web. Kaedah semasa menangani masalah ini terhad akibat jumlah besar bilangan positif palsu dihasilkan dalam analisis keputusan, dan tidak merangkumi semua jenis XSS, kurang tumpuan untuk penghapusan kerentanan XSS, dan tidak merangkumi aplikasi mobile web.

Teknik-teknik analisis statik amat baik untuk mengesan kerentanan XSS pada kod sumber aplikasi web, dan terutamanya bila digabungkan dengan teknik lain. Bagaimanapun ianya cenderung untuk menjana banyak positif palsu kerana ianya teknik konservatif. Kelemahan lain adalah teknik semasa kurang menumpu kepada

penghapusan kerentanan XSS selepas dikesan pada kod sumber. Seterusnya, kaedah yang dipanggil XSS-DETREM telah dicadangkan dengan objektif untuk mengabungkan algoritma genetik bersama analisis statik, dan teknik pengantian kod untuk mengesan dan menghapuskan kerentanan XSS, masing-masing untuk menagani masalah XSS di paras kod sumber. Penyelidikan ini menggunakan metodologi kuantitatif dan rekabentuk blok rawak lengkap untuk pengujian yang mana penambahbaikan baru dijalankan melalui alatan perisian.

XSS-DETREM telah dinilai secara empirikal mengunakan set data JSP dan aplikasi web android yang digunakan dalam kajian sebelum. Peningkatan ini memfokus kepada mengurangkan kadar positif palsu yang dijana oleh analisis static, meningkatkan liputan kerentanan kepada semua jenis XSS dikedua-dua pelayan dan pelanggan. Perbandingan ke atas hasil penilaian telah menunjukkan peningkatan pengesanan dan penghapusan kerentanan XSS untuk aplikasi desktop dan aplikasi mobil web. Seterusnya , objektif penyelidikan ini dicapai dan jangkaan keputusan dicapai. Kaedah penambahbaikan baru amat signifikan untuk membantu pembangun aplikasi web menguji aplikasi mereka sebelum digunakan oleh publik. Juga, semakin ramai pengguna melayari internet melalui aplikasi mobil, kaedah ini akan melindungi data peribadi dan menjadikan pelayaran selamat untuk mereka bagi keduanya iaitu desktop dan aplikasi mobil web.

Kata Kunci: Serangan penskripan silang-tapak, kerentanan penskripan silang-tapak, keselamatan perisian, Pengesanan kerentanan XSS

SDG: MATLAMAT 4: Kualiti Pendidikan

ACKNOWLEDGEMENTS

Alhamdulillah, praise be to Allah in Whose blessings I found the strength to accomplish this research work. I wish to thank all those who have contributed to the success of this venture.

First of all, I would like to thank my supervisor, Prof. Dr. Abu Bakar bin Md Sultan, and my co-supervisors Assoc. Prof. Dr. Hazura Zulzalil and Assoc. Prof. Ts. Dr. Novia Admodisastro for their invaluable support and guidance from the inception of my research work to its completion. Their constructive criticisms, comments, and suggestions have gone a long way in making this research successful.

A sincere and special thank you goes to my loving family for their endless encouragement, moral and financial support, prayers, and understanding.

A big thank you goes to the Ministry of Education, Malaysia for providing me with a scholarship to pursue a PhD degree in Universiti Putra Malaysia. Their financial support has gone a long way in alleviating any financial burden I could face and enable me to focus on my research work.

Last, but not the least, I extend my gratitude to the Faculty of Computer Science and Information Technology and all its staff and students for providing a conducive and friendly environment that enabled me work and study in peace. My gratitude also goes to my friend and closest lab mate Najla'a Ateeq Draib with whom I shared this research journey. I am also thanking all those who, in one way or the other, have contributed to the successful completion of this research work.

This thesis was submitted to the Senate of Universiti Putra Malaysia and has been accepted as fulfillment of the requirement for the degree of Doctor of Philosophy. The members of the Supervisory Committee were as follows:

Abu Bakar bin Md Sultan, PhD
Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Chairman)

Hazura binti Zulzalil, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

Novia Admodisastro, PhD
Associate Professor, Ts
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

ZALILAH MOHD SHARIFF, PhD
Professor and Dean
School of Graduate Studies
Universiti Putra Malaysia

Date: 10 October 2024

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

OWASP	Open Web Application Security Protocol
SDLC	System Development Life Cycle
SBSE	Search-Base Software Engineering
GA	Genetic Algorithm
EA	Evolutionary Algorithm
XSS	Cross Site Scripting
SLR	Systematic Literature Review
HTML	Hyper Text Markup Language
WWW	World Wide Web
DOM	Document Object Model
RQ	Research Question
QA	Quality Assessment
JGAP	Java Genetic Algorithm Package
IDE	Integrated Development Environment
CFG	Control Flow Graph
ESAPI	Enterprise Security API
LOC	Line Of Code
JDBC	Java Data Base Connector

CHAPTER 1

INTRODUCTION

1.1 Background of the Research

Web applications, both desktop and mobile versions, have become an integral part of our lives. We use them for accessing information, conducting business transactions online, and interacting with family and friends on social media (CWE, 2024; Krishnan et al., 2024; OWASP, 2024a). Many businesses and organizations also use web applications to provide many of their services, not only on Desktop versions but also on mobile versions as more people use their mobile phones to access some of those services. However, as web applications become very important to the success of businesses and organizations, their securities have increasingly become more complex (OWASP, 2024a). Hence, more security issues have emerged due to the increasing number of security threats affecting web applications (Thajeel et al., 2023).

Security testing of web applications has, therefore, become a crucial issue to the software security industry as well as governments, businesses, and organizations. Static application security testing (SAST) is an automated static analysis testing method for finding security vulnerabilities in web applications source code (Felderer et al., 2016). It can be applied very early in the software development life cycle and has an advantage over dynamic security testing as it can analyse all the control flows of a program. Study of the major security threats in web applications has shown that XSS vulnerabilities are among the top ten vulnerabilities, as reported by the Open Web Application Security Project (OWASP) (OWASP, 2024a). OWASP identifies

and documents the most common of these security issues, known as input validation vulnerabilities, that affect web applications. They keep an updated list of the top ten of these vulnerabilities (OWASP, 2024c).

Cross-site scripting (XSS) vulnerabilities are input validation vulnerabilities found in web applications and can be exploited through XSS attacks when such applications are deployed and running online (CWE, 2024; OWASP, 2024a). XSS attacks are of three types namely reflected, stored and DOM-based. Reflected XSS is executed by the victim's browser and occurs when the victim provides input to the web application such as username and password. The second type, Stored XSS attacks cab be stored in the web application's databases where information is saved, message forums, and comments fields. The malicious code is executed every time users open it thereby passing their privileges to the attacker. Both reflected and stored XSS take place on the application side. On the other hand, DOM-based XSS attacks are executed on the client side. Attackers are able to collect sensitive or important information from the user's computer.

Injecting malicious scripts where these applications accept user inputs can result to serious security breaches such as cookie theft, account hijacking, manipulation of web content and theft of private information. Many security solutions have been proposed, but the problem of XSS still remains and continues to affect many web applications. Attention on software security is increasing and progress on detection is being made but still more work needs to be done. New improvements on the existing approaches and techniques need to be added in order to tackle the expanding problem of XSS.

Genetic algorithms (GAs) have been the most commonly used of all optimization algorithms in SBSE, although there have been few studies to establish any practical performance differences among the algorithms. Genetic Algorithms (GAs) are a subset of Evolutionary Algorithms (EAs), which are metaheuristic optimization algorithms based on population and inspired by biology (Weise, 2009). They employ mechanisms of natural evolution such as mutation, crossover, natural selection, and survival of the fittest (Streichert, 2002) to find optimal solutions in a search space. GAs are different from other EAs in that they have a crossover (recombination) operation and use binary coding in bits or bit-strings to represent a population (Streichert, 2002).

GAs have proven to be good solutions to many software engineering problems since their discovery. Their successful use in software security testing (Avancini & Ceccato, 2010) and intrusion detection systems (Bankovic et al., 2008) enlighten the possibility of their usage in detecting and removing XSS vulnerabilities in web applications. In addition GAs have some advantages compared to other machine learning techniques (Bankovic et al., 2008), which are stated below:

- GAs are intrinsically parallel, since they have multiple offspring, they can explore the solution space in multiple directions at once. If one path turns out to be a dead end, they can easily eliminate it and continue work on more promising avenues.
- Due to the parallelism that allows them to implicitly evaluate many schemas at once, genetic algorithms are particularly well-suited to solving problems where the space of all potential solutions is truly huge - too vast to search exhaustively in any reasonable amount of time, as network data is.
- Working with populations of candidate solutions rather than a single solution, and employing stochastic operators, to guide the search process

permit GAs to cope well with attribute interactions and to avoid getting stuck in local maxima, which together make them very suitable for dealing with identifying different vulnerabilities such as XSS.

- A system based on GA can easily be re-trained. This property provides the adaptability of a GA-based system, which is an imperative quality of a vulnerability detection application bearing in mind the high rate of new attacks emerging.

1.2 Problem Statement

Cross-Site Scripting (XSS) vulnerabilities are a major security threat for both web and mobile applications (CWE, 2024; OWASP, 2024a). XSS attacks can lead to loss of private data, user account/session hijacking, web content manipulation, and financial losses for both individuals and businesses. Business impacts of XSS attacks can include system disruption, damaged reputation, legal issues, and financial losses (OWASP, 2024c). One of the most known real world examples of XSS attacks was the "Sammy worm" on the MySpace website in 2005 (Store, 2024). Since then, many other big corporations including FaceBook, Fortnite, British Airways, the CIA, and eBay have been victims of XSS attacks over the years (Store, 2024).

Many solutions have been proposed to address this problem at different stages of an application development life cycle, such as the design, coding and testing stages. Recent approaches to the mitigation of this problem at the testing stage include the integration of static analysis and genetic algorithm (M. A. Ahmed & Ali, 2016; Z. Liu et al., 2022; Tariq et al., 2021), the combination of static analysis and data mining (Medeiros et al., 2016a), the use of program slicing and pruning techniques (Thome et al., 2017), dynamic taint tracking technique (R. Wang, Xu, et al., 2017)

and the use of Genetic algorithm with other data mining techniques (Kareem Thajeel et al., 2023; Krishnan et al., 2024; Tariq et al., 2021; Younas et al., 2024). Although these solutions have proved effective in mitigating the problem of XSS to some extent, they have limitations that open some research gaps to be further addressed.

Static analysis techniques are good at detecting XSS vulnerabilities in the source codes of web applications, and especially when combined with other techniques. However, they tend to generate a lot of false positives since they are conservative techniques (Medeiros et al., 2016a; Thome et al., 2016, 2017). False positives occur when non-vulnerable parts of a piece of code are identified as vulnerable (Thajeel et al., 2023). In order to address this limitation, the underlying techniques and their algorithms need to be improved in terms of precision, coverage and effectiveness.

Another limitation of the existing approaches is the limited or lack of focus on the removal of XSS vulnerabilities after being detected in the source code. Detecting vulnerabilities in web applications is very important but if not properly removed the vulnerabilities will continue to cause security problems in the applications. Some research work that have addressed vulnerability removal include the use of the OWASP's escaping API and security guidelines (Lekies et al., 2017; Malviya et al., 2021; Rodríguez et al., 2020; Shanmugasundaram et al., 2015; Shar & Tan, 2012). The OWASP API (OWASP, 2024d) and security guidelines (OWASP, 2024f) do not include all XSS vulnerabilities in the context of their API. Therefore, these techniques will not be able to detect XSS vulnerabilities that are out of this context. There is need to extend these techniques in order to include all XSS vulnerabilities.

Another research work made use of data mining techniques (Medeiros et al., 2016a). These data mining techniques employed by Medeiros et al. are reported to be good at XSS vulnerability removal, but the techniques they used only included server-side and not client-side vulnerabilities.

A new dynamic approach to XSS detection has been proposed (R. Wang, Xu, et al., 2017). This approach proposed to use taint tracking technique on a browser's rendering process to derive and verify vulnerabilities automatically. The limitations of their work include the inability of the approach to handle two-order inputs and the long time taken to generate attack vectors. In addition, this approach only focuses on DOM-based XSS and not Reflected or Stored XSS vulnerabilities.

This research work, therefore, was focused on reducing false positives in the detection of XSS vulnerabilities, increasing the coverage of XSS types addressed, and eliminating detected vulnerabilities from the source to prevent future attacks.

1.3 Objectives of the Research

The main objective of this research is to propose an enhancement in the detection and removal of XSS vulnerabilities from web and mobile applications using Genetic Algorithms, static taint analysis and code replacements techniques. This research work is proposed to make improvements to the limitations identified in the previously proposed approaches on XSS vulnerability detection and removal. These improvements will focus on reducing the rate of false positives generated by static analysis, including both server-side and client-side XSS vulnerabilities and using

code replacement technique to remove vulnerabilities. Hence, our specific objectives are:

- To propose an improved GA based security testing approach to XSS detection in desktop and mobile web applications that reduces false positive rates
- To use an enhanced code replacement technique to remove the detected XSS vulnerabilities
- To empirically evaluate and measure the effectiveness of XSS detection and removal of the proposed approach

1.4 Scope of the Research

This research focuses on the security testing of web applications at the source code level, and hence employs white box testing techniques. There are many security vulnerabilities affecting web applications and XSS is among the top. Moreover, XSS attacks target not only the web applications but the users as well. Therefore, this research is focusing on improving the detection and removal of XSS vulnerabilities by reducing false positives and eliminating detected vulnerabilities. The detection and removal of all the three types of XSS vulnerabilities i.e., reflected, stored, and DOM-based on both the server sides and client sides of web applications are also addressed. In addition, the issue of cross-site scripting in mobile applications is also included. Our proposed approach is designed for testing Java-based web applications and Android mobile applications. Java Server Pages (JSP) are widely used for developing web applications, and the Android applications are based on Java. The limitation of the scope to Java-based web applications could impact the

generalisation of the approach. However, with some modification in future research, the approach can be adopted to other programming languages.

1.5 Contributions of the Research

This research produced an enhanced and improved approach to detecting and removing XSS vulnerabilities in Java-based web and Android mobile applications. Therefore, the main contributions of this research are:

- A comprehensive survey of existing XSS mitigation approaches
- An improved GA-based detection technique for all XSS types in Java-based web applications
- An improved removal technique for all XSS types in Java-based web applications
- A tool support for the automation of the detection and removal techniques of XSS
- Empirical evidence that the proposed approach is effective in detecting and removing XSS vulnerabilities from web applications
- The improvement on the detection and removal of XSS vulnerabilities in the source code of web applications
- The implementation of a tool for the automation of the proposed approach
- Empirical evidence to show that the proposed approach can be more effective in detecting and removing XSS vulnerabilities

1.6 Organization of the Thesis

This thesis consists of seven chapters that are organized as follows:

Chapter 1 introduces the thesis and provides the general overview of the thesis. It provides an overview of the research background and identifies the research problem, the objectives to be achieved, as well as the research scope and contributions.

Chapter 2 provides a detailed review of the literature related to this research work. It discusses the existing research on XSS vulnerabilities detection and removal. Different approaches and techniques to mitigate the XSS problems have been reviewed and the research gaps found have been identified in this chapter.

In Chapter 3, the methodology of the research is presented in general overview. It highlights the different steps of the methodology used in the research.

Chapter 4 describes our proposed XSS-DETREM approach for the detection and removal of XSS vulnerabilities, demonstrating how the tool detects and removes XSS vulnerabilities. It gives an illustrative example of how the proposed approach works.

Chapter 5 presents the detailed implementation of the proposed approach to XSS detection and removal. It describes the design and implementation of the tool support.

Chapter 6 provides the experiment results after the evaluation of XSS-DETREM approach. The effectiveness of the approach is shown by the comparison results against other approaches. The chapter also discusses the implications of the results and findings.

Chapter 7 gives the conclusion of the thesis and discusses the new knowledge gained from this research work. The shortcomings and limitations of the research work are stated, and recommendations and suggestions are given for future research work.

REFERENCES

Acunetix. (2024). *Cross Site Scripting (XSS)*. Acunetix. http://www.acunetix.com/websitesecurity/cross-site-scripting/

Ahmed, M. A., & Ali, F. (2016). Multiple-path testing for cross site scripting using genetic algorithms. *Journal of Systems Architecture*, *64*, 50–62. https://doi.org/10.1016/j.sysarc.2015.11.001

Ahmed, M. a., & Hermadi, I. (2008). GA-based multiple paths test data generator. *Computers & Operations Research*, *35*(10), 3107–3124. https://doi.org/10.1016/j.cor.2007.01.012

Aljahdali, S. H., Ghiduk, A. S., & El-Telbany, M. (2010). The limitations of genetic algorithms in software testing. *ACS/IEEE International Conference on Computer Systems and Applications - AICCSA 2010*, 1–7. https://doi.org/10.1109/AICCSA.2010.5586984

Antunes, N., & Vieira, M. (2015). Assessing and Comparing Vulnerability Detection Tools for Web Services: Benchmarking Approach and Examples. *IEEE Transactions on Services Computing*, *8*(2), 269–283. https://doi.org/10.1109/TSC.2014.2310221

Avancini, A., & Ceccato, M. (2010). Towards Security Testing with Taint Analysis and Genetic Algorithms. *Proceedings of the 2010 ICSE Workshop on Software Engineering for Secure Systems, Section 5*, 65–71.

Avancini, A., & Ceccato, M. (2011). Security Testing of Web Applications: A Search-Based Approach for Cross-Site Scripting Vulnerabilities. *2011 IEEE 11th International Working Conference on Source Code Analysis and Manipulation*, 85–94. https://doi.org/10.1109/SCAM.2011.7

Avancini, A., & Ceccato, M. (2013). Comparison and integration of genetic algorithms and dynamic symbolic execution for security testing of cross-site scripting vulnerabilities. *Information and Software Technology*, *55*(12), 2209–2222. https://doi.org/10.1016/j.infsof.2013.08.001

Bankovic, Z., Bojanic, S., Nieto, O., & Badii, A. (2008). Unsupervised Genetic Algorithm Deployed for Intrusion Detection. In E. Corchado, A. Abraham, & W. Perdrycz (Eds.), *Lecture Notes in Computer Science* (vol 5271, pp. 132–139). Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/978-3-540-87656-4_17

Banković, Z., Stepanović, D., Bojanić, S., & Nieto-Taladriz, O. (2007). Improving network security using genetic algorithm approach. *Computers & Electrical Engineering*, *33*(5–6), 438–451. https://doi.org/10.1016/j.compeleceng.2007.05.010

Caturano, F., Perrone, G., & Romano, S. Pietro. (2021). Discovering reflected cross-site scripting vulnerabilities using a multiobjective reinforcement learning environment. *Computers and Security*, *103*, 102204. https://doi.org/10.1016/j.cose.2021.102204

Chakraborty, R. C. (2010). *Fundamentals of Genetic Algorithms*. http://www.myreaders.info/09_Genetic_Algorithms.pdf

Chen, Y. L., Lee, H. M., Jeng, A. B., & Wei, T. E. (2015). DroidCIA: A novel detection method of code injection attacks on HTML5-based mobile apps. *Proceedings - 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2015*, *1*, 1014–1021. https://doi.org/10.1109/Trustcom.2015.477

CWE. (2024). *CWE - CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') (2.5)*. The MITRE Corporation. http://cwe.mitre.org/data/definitions/79.html

Doupé, A., Cui, W., Jakubowski, M. H., Peinado, M., Kruegel, C., & Vigna, G. (2013). DeDacota: Toward preventing server-side XSS via automatic code and data separation. *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, 1205–1216. https://doi.org/10.1145/2508859.2516708

Dua, M., & Singh, H. (2018). Detection & prevention of website vulnerabilities: Current scenario and future trends. *Proceedings of the 2nd International Conference on Communication and Electronics Systems, ICCES 2017*, 429–435. https://doi.org/10.1109/CESYS.2017.8321315

Duchene, F., Groz, R., Rawat, S., & Richier, J.-L. (2012). XSS Vulnerability Detection Using Model Inference Assisted Evolutionary Fuzzing. *2012 IEEE Fifth International Conference on Software Testing, Verification and Validation, Itea 2*, 815–817. https://doi.org/10.1109/ICST.2012.181

Elhakeem, Y. F. G. M., & Barry, B. I. A. (2013). Developing a security model to protect websites from cross-site scripting attacks using ZEND framework application. *Proceedings - 2013 International Conference on Computer, Electrical and Electronics Engineering: "Research Makes a Difference", ICCEEE 2013*, 624–629. https://doi.org/10.1109/ICCEEE.2013.6634012

Felderer, M., Büchler, M., Johns, M., Brucker, A. D., Breu, R., & Pretschner, A. (2016). Chapter One – Security Testing: A Survey. In A. M. Memon (Ed.), *Advances in Computers* (1st Editio, Vol. 101, pp. 1–51). Elsevier Inc. https://doi.org/10.1016/bs.adcom.2015.11.003

Fogie, S., Grossman, J., Hansen, R., Rager, A., & Petkov, P. D. (2007). *XSS Attacks: Cross Site Scripting Exploits and Defense* (S. Forgie (ed.)). Elsevier, Inc/Syngress Publishing, Inc.

Gol, D., & Shah, N. (2015). Detection of Web Appication Vulnerability Based on RUP Model. *National Conference on Recent Advances in Electronics & Computer Engineering (RAECE)*, 96–100. https://doi.org/10.1109/RAECE.2015.7510233

Grabowski, R., Hofmann, M., & Li, K. (2012). Type-Based Enforcement of Secure Programming Guidelines — Code Injection Prevention at SAP. *FAST 2011, Lecture Notes in Computer Science*, *7140*, 182–197.

Gupta, S., & Gupta, B. B. (2016). CSSXC: Context-sensitive Sanitization Framework for Web Applications against XSS Vulnerabilities in Cloud Environments. *Procedia Computer Science*, *85*(Cms), 198–205. https://doi.org/10.1016/j.procs.2016.05.211

Gupta, S., & Gupta, B. B. (2018). XSS-secure as a service for the platforms of online social network-based multimedia web applications in cloud. *Multimedia Tools and Applications*, *77*(4), 4829–4861. https://doi.org/10.1007/s11042-016-3735-1

Gupta, S., Gupta, B. B., & Chaudhary, P. (2018). Hunting for DOM-Based XSS vulnerabilities in mobile cloud-based online social network. *Future Generation Computer Systems*, *79*, 319–336. https://doi.org/10.1016/j.future.2017.05.038

Hanif, H., Nasir, M. H. N. M., Ab Razak, M. F., Firdaus, A., & Anuar, N. B. (2021). The rise of software vulnerability: Taxonomy of software vulnerabilities detection and machine learning approaches. *Journal of Network and Computer Applications*, *179*(November 2020), 103009. https://doi.org/10.1016/j.jnca.2021.103009

Hermadi, I. (2004). *Genetic algorithm based test data generator* [King Fahd university of Petroleum and Minerals]. https://doi.org/10.1109/CEC.2003.1299560

Islam, A. B. . A. Al, Azad, M. A., Alam, M. K., & Alam, M. S. (2007). Security Attack Detection using Genetic Algorithm (GA) in Policy Based Network. *2007 International Conference on Information and Communication Technology*, 341–347. https://doi.org/10.1109/ICICT.2007.375407

Jin, X., Hu, X., Ying, K., Du, W., Yin, H., & Peri, G. N. (2014). Code Injection Attacks on HTML5-based Mobile Apps: Characterization, Detection and Mitigation. *CCS '14 Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 66–77. https://doi.org/10.1145/2660267.2660275

Johns, M., Beyerlein, C., Giesecke, R., & Posegga, J. (2010). Secure Code Generation for Web Applications. *Lecture Notes in Computer Science*, *5965*, 96–113.

Kallin, J., & Valbuena, I. L. (2019). *Excess XSS: A comprehensive tutorial on cross-site scripting*. Excess-Xss.Com. https://excess-xss.com/

Kareem Thajeel, I., Samsudin, K., Jahari Hashim, S., & Hashim, F. (2023). Dynamic feature selection model for adaptive cross site scripting attack detection using developed multi-agent deep Q learning model. *Journal of King Saud University - Computer and Information Sciences*, *xxxx*. https://doi.org/10.1016/j.jksuci.2023.01.012

Kaur, G., Pande, B., Bhardwaj, A., Bhagat, G., & Gupta, S. (2018). Efficient yet Robust Elimination of XSS Attack Vectors from HTML5 Web Applications Hosted on OSN-Based Cloud Platforms. *Procedia Computer Science*, *125*, 669–675. https://doi.org/10.1016/j.procs.2017.12.086

Korać, D., Damjanović, B., Simić, D., & Choo, K. K. R. (2022). A hybrid XSS attack (HYXSSA) based on fusion approach: Challenges, threats and implications in cybersecurity. *Journal of King Saud University - Computer and Information Sciences*, *34*(10), 9284–9300. https://doi.org/10.1016/j.jksuci.2022.09.008

Krishnan, M., Lim, Y., Perumal, S., & Palanisamy, G. (2024). Detection and defending the XSS attack using novel hybrid stacking ensemble learning-based DNN approach. *Digital Communications and Networks*, *10*(3), 716–727. https://doi.org/10.1016/j.dcan.2022.09.024

Kurniawan, A., Abbas, B. S., Trisetyarso, A., & Isa, S. M. (2018). Static Taint Analysis Traversal with Object Oriented Component for Web File Injection Vulnerability Pattern Detection. *Procedia Computer Science*, *135*, 596–605. https://doi.org/10.1016/j.procs.2018.08.227

Lekies, S., Kotowicz, K., Groß, S., Vela Nava, E. A., & Johns, M. (2017). Code-Reuse Attacks for the Web: Breaking Cross-Site Scripting Mitigations via Script Gadgets. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security - CCS '17*, 1709–1723. https://doi.org/10.1145/3133956.3134091

Liu, Y., & Milanova, A. (2009). Practical Static Analysis for Inference of Security-Related Program Properties. *IEEE 17th International Conference on Program Comprehension*, 50–59. https://doi.org/10.1109/ICPC.2009.5090027

Liu, Z., Fang, Y., Huang, C., & Xu, Y. (2022). GAXSS: Effective Payload Generation Method to Detect XSS Vulnerabilities Based on Genetic Algorithm. *Security and Communication Networks*, *2022*, 1–15. https://doi.org/10.1155/2022/2031924

Livshits, B., & Lam, M. S. (2005). Finding Security Vulnerabilities in Java Applications with Static Analysis. *USENIX Security*, 18. http://portal.acm.org/citation.cfm?id=1251416

Malviya, V. K., Rai, S., & Gupta, A. (2021). Development of web browser prototype with embedded classification capability for mitigating Cross-Site Scripting attacks. *Applied Soft Computing*, *102*, 106873. https://doi.org/10.1016/j.asoc.2020.106873

Marashdih, A. W., Zaaba, Z. F., Suwais, K., & Mohd, N. A. (2019). Web application security: An investigation on static analysis with other algorithms to detect cross site scripting. *Procedia Computer Science*, *161*, 1173–1181. https://doi.org/10.1016/j.procs.2019.11.230

McGraw, G. (2006). *Software Security: Building Security In* (1st Editio). Addison Wesley Profesional.

Medeiros, I., Neves, N., & Correia, M. (2016a). Detecting and Removing Web Application Vulnerabilities with Static Analysis and Data Mining. *IEEE Transactions on Reliability*, *65*(1), 54–69. https://doi.org/10.1109/TR.2015.2457411

Medeiros, I., Neves, N., & Correia, M. (2016b). Equipping WAP with WEAPONS to detect vulnerabilities: Practical experience report. *Proceedings - 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2016*, 630–637. https://doi.org/10.1109/DSN.2016.63

Mohammadi, M., Chu, B., & Lipford, H. R. (2017). Detecting cross-site scripting vulnerabilities through automated unit testing. *Proceedings - 2017 IEEE International Conference on Software Quality, Reliability and Security, QRS 2017*, 364–373. https://doi.org/10.1109/QRS.2017.46

Møller, A., & Schwarz, M. (2014). Automated detection of client-state manipulation vulnerabilities. *ACM Transactions on Software Engineering and Methodology (TOSEM) - Special Issue International Conference on Software Engineering (ICSE 2012) and Regular Papers*, *23*(4), 29:1-29:30. https://doi.org/10.1145/2531921

OWASP. (2024a). *Cross Site Scripting (XSS)*. OWASP Foundation. https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

OWASP. (2024b). *DOM based XSS Prevention Cheat Sheet*. OWASP Foundation. https://www.owasp.org/index.php/DOM_based_XSS_Prevention_Cheat_Sheet

OWASP. (2024c). *Mobile Top 10 2024-M4 - OWASP*. OWASP Foundation. https://owasp.org/www-project-mobile-top-10/

OWASP. (2024d). *OWASP Enterprise Security API - OWASP*. OWASP Foundation. https://www.owasp.org/index.php/Category:OWASP_Enterprise_Security_API

OWASP. (2024e). *OWASP Java HTML Sanitizer*. OWASP Foundation. https://www.owasp.org/index.php/OWASP_Java_HTML_Sanitizer_Project

OWASP. (2024f). *XSS (Cross Site Scripting) Prevention Cheat Sheet*. OWASP Foundation. https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet

Rathore, A. (2011). Application of Genetic Algorithm and Tabu Search in Software Testing. *Proceedings of the Fourth Annual ACM Bangalore Conference*, 1–4.

Rodríguez, G. E., Torres, J. G., Flores, P., & Benavides, D. E. (2020). Cross-site scripting (XSS) attacks and mitigation: A survey. *Computer Networks*, *166*, 1–23. https://doi.org/10.1016/j.comnet.2019.106960

Shanmugasundaram, G., Ravivarman, S., & Thangavellu, P. (2015). A study on removal techniques of Cross-Site Scripting from web applications. *4th IEEE Sponsored International Conference on Computation of Power, Energy, Information and Communication, ICCPEIC 2015*, 436–442. https://doi.org/10.1109/ICCPEIC.2015.7259498

Shar, L. K., & Tan, H. B. K. (2012). Automated removal of cross site scripting vulnerabilities in web applications. *Information and Software Technology*, *54*(5), 467–478. https://doi.org/10.1016/j.infsof.2011.12.006

Shuai, B., Li, M., Li, H., Zhang, Q., & Tang, C. (2013). Software vulnerability detection using genetic algorithm and dynamic taint analysis. *2013 3rd International Conference on Consumer Electronics, Communications and Networks*, 589–593. https://doi.org/10.1109/CECNet.2013.6703400

Singh, M., Singh, P., & Kumar, P. (2020). An Analytical Study on Cross-Site Scripting. *2020 International Conference on Computer Science, Engineering and Applications (ICCSEA)*, *March*, 1–13. https://doi.org/10.1109/ICCSEA49143.2020.9132894

Srivastava, P. R., & Kim, T. (2009). Application of Genetic Algorithm in Software Testing. *Intenational Journal of Software Engineering and Its Applications*, *3*(4), 87–96.

Steinhauser, A., & Gauthier, F. (2016). JSPChecker: Static detection of context-sensitive cross-site scripting flaws in legacy web applications. *PLAS 2016 - Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security, Co-Located with CCS 2016*, 57–68. https://doi.org/10.1145/2993600.2993606

Store, W. S. (2024). *5 Real-World Cross Site Scripting Examples*. https://websitesecuritystore.com/blog/real-world-cross-site-scripting-examples/

Streichert, F. (2002). Introduction to Evolutionary Algorithms. *MathFinance Workshop*, 1–21.

Tariq, I., Sindhu, M. A., Abbasi, R. A., Khattak, A. S., Maqbool, O., & Siddiqui, G. F. (2021). Resolving cross-site scripting attacks through genetic algorithm and reinforcement learning. *Expert Systems with Applications*, *168*(November 2020). https://doi.org/10.1016/j.eswa.2020.114386

Thajeel, I. K., Samsudin, K., Hashim, S. J., & Hashim, F. (2023). Machine and Deep Learning-based XSS Detection Approaches: A Systematic Literature Review. *Journal of King Saud University - Computer and Information Sciences*, *35*(7), 101628. https://doi.org/10.1016/j.jksuci.2023.101628

Thome, J., Shar, L. K., & Briand, L. (2016). Security slicing for auditing XML, XPath, and SQL injection vulnerabilities. *2015 IEEE 26th International Symposium on Software Reliability Engineering, ISSRE 2015*, 553–564. https://doi.org/10.1109/ISSRE.2015.7381847

Thome, J., Shar, L. K., & Briand, L. (2017). Security slicing for auditing common injection vulnerabilities. *The Journal of Systems and Software*, *0*, 1–18. https://doi.org/10.1016/j.jss.2017.02.040

Van Rijsbergen, C. J. (1979). *Information Retrieval* (2nd Ed.). Butterworth-Heinemann.

Wang, C. H., & Zhou, Y. S. (2016). A New Cross-Site Scripting Detection Mechanism Integrated with HTML5 and CORS Properties by Using Browser Extensions. *Proceedings - 2016 International Computer Symposium, ICS 2016*, 264–269. https://doi.org/10.1109/ICS.2016.0060

Wang, R., Xu, G., Zeng, X., Li, X., & Feng, Z. (2017). TT-XSS: A novel taint tracking based dynamic detection framework for DOM Cross-Site Scripting. *Journal of Parallel and Distributed Computing*, 4–10. https://doi.org/10.1016/j.jpdc.2017.07.006

Wang, R., Zhu, Y., Tan, J., & Zhou, B. (2017). Detection of malicious web pages based on hybrid analysis. *Journal of Information Security and Applications*, *35*, 68–74. https://doi.org/10.1016/j.jisa.2017.05.008

Weise, T. (2009). *Global Optimization Algorithms – Theory and Application* – (2nd Ed.).

Wohlin, C., Runeson, P., Host, M., Ohlsson, M. C., Regnell, B., & Wesslen, A. (2000). *Experimentation in software engineering An Introduction* (V. R. Basili (ed.); 1st ed.). Springer Science+Business Media, LLC. https://doi.org/10.1007/978-1-4615-4625-2

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). Experimentation in Software Engineering. In *Experimentation in Software Engineering*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-29044-2

Younas, F., Raza, A., Thalji, N., Abualigah, L., Zitar, R. A., & Jia, H. (2024). An efficient artificial intelligence approach for early detection of cross-site scripting attacks. *Decision Analytics Journal*, *11*(January), 100466. https://doi.org/10.1016/j.dajour.2024.100466

Zhou, Y., & Wang, P. (2019). An ensemble learning approach for XSS attack detection with domain knowledge and threat intelligence. *Computers and Security*, *82*, 261–269. https://doi.org/10.1016/j.cose.2018.12.016

Zimmer, D. (2008). *Real World XSS*. XSSed.Com. http://www.xssed.com/article/21/Paper_Real_World_XSS/