Investigation of Detection and Mitigation of Web Application Vulnerabilities

Shekhar Disawal
School of Computer Science & IT
Devi Ahilya University,
Indore, M.P., India

Ugrasen Suman, PhD
School of Computer Science & IT
Devi Ahilya University,
Indore, M.P., India

Maya Rathore, PhD
Associate Professor
Christian Eminent College Indore
M.P., India

ABSTRACT
Web applications are the backbone of technology in the global era of information. In this digital world connecting many commercial organizations that utilize the internet for financial transactions, education, and other activities. In recent days, web applications have been exploited by attackers frequently. Most web developers and website owners have limited awareness of the vulnerabilities in their websites,which are prone to web vulnerability attacks. Many researchers are working to detect and mitigate the vulnerability and provide differentmethods to resolve the various types of web vulnerabilities. However, these solutions are insufficient since they often have restrictions and areinefficient to prevent all vulnerabilities. This paper aims to reviewexisting detection and mitigation methodsfor web application vulnerabilities.This will helppractitioners to develop practices and solve issues related to web vulnerabilities.

Keywords
Web application vulnerability, Detection, and Mitigation, Web attacks

1. INTRODUCTION
For many years, the world has witnessed various applications developed for communication, meetings, e-business, and various purposes. Mostly users use their online platforms for business and other purposes. As the usage of the internet grows, there is a need to provide secure and crucial data communications. Many unwanted activities happen with users due to insecure web applications because of the weaknesses or loopholes in web applications.

The attackerexploitthe loophole and ultimately controls the web applications. Any weakness or loophole in a web application is known as vulnerability. Some vulnerabilities are SQL Injection, XPath Injection, Cross-Site Scripting, Broken Access Control, Buffer Overflow, and many more. Every exposure has a different impact on the web application.

According to CERT-IN in 2017 report, approximately 53089 security events were addressed, including 29500 and 18 website defacements [1]. According to Statista, 953 thousand online assaults were stopped daily in 2018, increasing from 611 thousand daily blocked attacks [2]. In February 2021, over a year after the pandemic began, there were 377.5 million brute-force attacks, a far cry from the 93.1 million seen at the start of 2020. In February 2021, India alone saw 9.04 million assaults. The total number of assaults registered in India during January and February 2021 was around 15 million [3].

The internetisfullof attackers with harmful and criminal motives who target inexperienced users using freshly devised attack vectors to compromise security. Some people perform it for enjoyment, while others have malicious intentionssuch as exploiting data to make quick money, theft, extortion, criminal activity.

Continuous efforts are needed to identify the attack methods, goals, and remedies for these newly developed attack vectors. Various profitable and non-profit consortiums, groups, or agencies are working in this direction. They create a database of identified attacks, complete with attack description, procedure, goal, risk factors, and prevention measures. MITRE's Common Weakness Enumeration (CWE), CERT-In, WASC Threat Classification, SANS Institute, National Cyber Security, CDAC, OWASP (Open Web Application Security Project), and Centre for Internet Security (CIS) are examples of national and worldwide vulnerability database standard agencies.

Security researchers make numerous efforts to identify these vulnerabilities and implement mitigation techniques from time to time. However, comprehensive vulnerability assessments are required to deal with rising web application vulnerability risks. Continuous efforts are necessary to solve web vulnerability issues. Various security issues can be resolved by implementing a proper web vulnerability detection and mitigation strategy.Although researchers have developed the techniques and methods to prevent web application vulnerabilities,the problem persists in web applications and is still very harmful.

This paper isorganised as follows. Section 2 presents a brief background and related work. Section 3 deals with a research approach for this study. Section 4 describes the results obtained from the literature review. Section 5 describes research questions along with finding. Section 6 presents the discussion, and finally, Section 7 provides the concluding remarks.

2. BACKGROUND
Various techniques have been devised for detecting various forms of XSS attacks. There exist limited strategies for mitigating and preventing all sorts of assaults. For the prevention of client-side and server-side attacks, several approaches such as static analysis, dynamic analysis, proxy firewalls, and sanitization can be used [4].

A static string analyzer uses context-free grammar to verify the string output of a program. This method examines the entire page for the existence of the < “script”> tag[5]. The Pixy tool discovers the web application vulnerability using static analysis. Static source code analysis is helpful to solve the problem of a vulnerable online application. The Data flow analysis has been utilized to find weak points in the software using flow sensitivity, inter-procedural, and contextsensitivity. Static analysis might result in false positives. If the number of false positives is large,the site is vulnerable to XSS attack [6]. Non-persistent attacks are identified by comparing incoming data and departing Javascript with simple measures like matching incoming data to HTML Javascript code. Separating the code and data, deDacota statically rewrites the current application. The static analysis method identifies all inline

JavaScript code in web pages. The dynamic inline script is a second-order issue. deDacota slightly solved this problem. [7].

Dynamic analysis has been conducted by inspecting the code through multiple input values with different encoding methods, then identifying which kind of input was the source of the security breach. Using a machine-learning algorithm based on the URL and Javascript code features, they identify regular and dangerous pages. The J48 provides better performance forfalse positive rates. But the time required for file creation was more significant than the Naïve Bayes classifier but less than the Support Vector Machine [8].

NOXES was the first client-side solution to protect against cross-site scripting attacks. By analyzing the flow of data through the browser, Noxes focuses on preserving the secrecy of sensitive data [9]. The web proxy prevents data leaking from the user environment [10]. The three-step model provides a client-side solution that does not degrade the performance of an application. It offers efficient security from the XSS attacks with optimized web browsing [11].

A server-side sanitizer prevents XSS attacks by validating current sanitization. In web applications, effective sanitization is a severe issue. It is crucial to ensure that all components of web applications are covered [12]. Initially, identify all of the possible attack vectors. The same input may appear distinctly in the application's output [13]. An HTML input filter is used on the browser side to damage the security of web applications. There is a server-side solution to XSS attacks; this solution is not dependent on the online application provider. The authors minimize information leaking on the browser side. This approach makes it simple to include a Javascriptfilter into Java-based applications [14].

The vulnerabilities in the context of web services are very harmful. In addition, the author discussed the general countermeasures forthe prevention and mitigation of SQLIA [15]. Furthermore,it aimed to provide a web developer with the potential impact of knownweb application attacks based on attackcategory, level, spreading, size, deviation, dependencies, findability, and amplification [16]. However,review the effective SQLIA detection and prevention techniquesand illustrate the prevalent input validation attacks, including SQL injection and Cross-Site Scripting [17]. Although various review papers are available about the types, processes, and tools of SQLIA, there is a lack of systematic literature reviews on SQLIA to keep investigators up to date with the styles, techniques, and tools of SQLIA[18-20].

3. RESEARCH APPROACH

This paper analyzes existing studies on detecting and mitigating web application vulnerabilities published in journals/conferences. The research questions, data sources, requirement process, paper inclusion standard, paper exclusion standard,and data collection are discussed in the following subsections.

3.1 Research Questions

We have formed the following research questions to complete the study of web application vulnerability detection and mitigation technique issues.

Q1: How much research has conducted on detecting and mitigating web application vulnerabilities?

Q2: What are the web service vulnerabilities issues that are identified in the research papers?

Q3: What are the methods proposed to solve web service vulnerability issues?

Q4: Which web service vulnerability areas are focused on in the research papers?

3.2 Data Sources

There are various sources of research databases. We have chosen the following database, which has been considered relevant for web application vulnerability.

- ACM Digital Library
- IEEE Xplore
- Elsevier Science Direct
- International Journal/Conference

3.3 Requirement Process

We want to search all necessary literatureto analyze web vulnerabilities detection and mitigation techniques, anda data repository search has been performed. Relevant articles havebeen obtained from each of the repositories. The search string that we have used is as follows:

((“web service attacks OR “web service vulnerability” OR “web vulnerabilitymitigation” OR “web application vulnerability” OR “cross-site scripting vulnerability” OR “SQL injection vulnerability” OR “XPath injection vulnerability” OR “spoofing attack” OR “web service attack detection”) AND (“detection” OR “mitigation” OR “prevention” OR “web app vulnerabilities detection method”) AND (“issues” OR “problem” OR “challenges” OR “tool”)).

3.4 Paper Inclusion Standard

Research Papers with the following criteria have been included:

- Papers that describe the solution to address the web service security problem.
- Papers that use methodologies for vulnerability detection in web services.
- Have discussed vulnerability detection and mitigation techniques.
- Papers that address web service attacks such as Cross-Site Scripting, SQL injection,Buffer Overflow, Spoofing, and XPath injection.
- Papers that were published before March 2021.

3.5 Paper Exclusion Standard

Research Papers with the following criteria have beenexcluded:

- Was duplicate or repeated content.
- Were papers not related to the research objective.

3.6 Data Collection

The following data has been collected from every paper:

- Title of the article publication
- Name of vulnerability
- Summary of the paper
- Type of method/technique
- Area of focus (vulnerability detection, mitigation/prevention)
- Issues in method or technique

4. RESULTS OBTAINED

The search string is used to search the electronic databases mentioned earlier. The search string has been modified under the database. The title and abstract of the article are assessed in

the initial review to determine whether or not it should be included. The papers chosen in the initial study are evaluated by reading their introduction, opening a few pages, and conclusion. Then, a subset of the publications that were judged to be relevant was chosen. In the final review, whole papers were reviewed and checked to ensure that they met the inclusion requirements. Table 1 displays the summary and findings of the selected articles reviewed.

Table 1. Summary of review papers

Name of Vulnerability	Summary of the Paper	Method	Area of focus Prevention/ Detection/Both	Issues
SQL Injection [21]	Model-Based Approach to Prevent SQL Injection Attacks on .NET Applications	Model-based approach	Attack prevention	If runtime generates queries found to be malicious code, it will not match the static query model. Thus, it will be rejected.
SQL Injection [22]	On predictive errors of SQL injection attack detection by the feature of the single character	Feature of single character	Vulnerability detection	False Positive Result
SQL Injection [23]	Obfuscation-based Analysis of SQL Injection Attacks	Obfuscation-based	Both	Manual Approach
SQL Injection [24]	Automated testing for SQL injection vulnerabilities: an input mutation approach	Input mutation approach	Vulnerability detection	Not work in complex condition
SQL, XML, XPath injection [25]	A novel approach complements existing vulnerability detection by forming sound and precise slices, thus identifying false and true positives.	Novel approach	Vulnerability detection	False and true positive values
SQL injection, XPath injection [26]	An approach to prevent SQL/XPath Injection attacks on web services by combining statement learning and service protection.	Combine statement learning	Attack prevention	Still problem not solved
SQL Injection [27]	Using ASCII-Based String Matching, an Efficient Technique for Detection and Prevention of SQL Injection Attacks.	Code conversion method	Both	Not effective in a complex query
XPath Injection [28]	In a database-centric web services context, a model-based system design is used to prevent XPath injection.	Model-based system architecture	Attack prevention	Still problem not solved
Cross-Site Scripting [29]	Detection of an XSS vulnerability in PHP web application.	Genetic algorithm	Vulnerability detection	Manual Approach
Cross-Site Scripting [30]	An algorithmic approach to detect and remove XSS vulnerability using data mining techniques.	Data mining	Vulnerability detection	False Positive Result

Cross-Site Scripting [31]	Gray box mechanism HTML output cooperates with the database context-sensitive XSS flaws HTTP e OWASP Zed Attack Proxy	Gray box mechanism	Vulnerability detection	Intercept traffic to non-opensource databases
Cross-Site Scripting [32]	Detection for XSS Attacks Based on Generative Adversarial Networks	MCTS, GAN Deep learning CICIDS2017 dataset	Vulnerability detection	Suitable only for adversarial examples
Cross-Site Scripting [33]	A Multilayer Perceptron-Based Integrated XSS-Based Attack Detection Scheme in Web Applications	ANN, MLP, DFE dynamic-features extraction technique	Vulnerability detection	High precision and low complexity
Cross-Site Scripting [34]	The novel taint tracking based dynamic detection framework for DOM Cross-Site Scripting	DOM-based XSS attack Taint tracking analysis	Vulnerability detection	lower rate of false-positive and false-negative
Cross-Site Scripting [35]	A Server-Side Approach to Automatically Detect XSS Attacks	S2XS2	Vulnerability detection	It takes lots of time; low detection capability
Cross-Site Scripting [36]	A Micro Benchmark for Evaluating DOM-Based Cross-Site Scripting Detection	DomXss Micro	Vulnerability detection	only detect one type of XSS attack.
Cross-Site Scripting [37]	A new server-side solution for detecting Cross-Site Scripting attack	Server-side solution	Vulnerability detection	requests to load from the server of the open network.
Cross-Site Scripting [38]	Automated discovery of JavaScript code injection attacks in PHP web applications	An automatic detection system	Vulnerability detection	Cannot detect DOM-based XSS attack
Cross-Site Scripting [39]	Using Randomization to Enforce Information Flow Tracking and Thwart Cross-site	Scripting Attacks	Vulnerability detection	Needs to modify the browser
Cross-Site Scripting [40]	Precise Dynamic Prevention of Cross-Site Scripting Attacks	XSS-GUARD	Vulnerabilitypreventio n	Do not prohibit the benign HTML allowed
Cross-Site Scripting [41]	Enhanced browser defense for reflected Cross-Site Scripting	XSS-ME	Vulnerability detection	Only one attack can be detected and defended.
Spoofing [42]	To avoid a similar attack on web services, a usage pattern termed Spoofing web services was proposed.	Misuse pattern	Vulnerability prevention	Still problem not solved

5. EXPLORATION

In this research paper, we have studied various research papers related to web application vulnerability detection and mitigation methods mentioned in Table 1.We describe our research work through the following questions:

5.1 How much research has conducted on the detection and mitigation of web application vulnerabilities?

This systematic study identified relevant papers, as shown in Table 1. Paper publications are spread across journals and conference proceedings. Out of the 22 papers,8 have been published in journals covering computer science, information, and web applicationvulnerability. The remaining 14 have been published at various international conferences and others.

Table 2. Web Vulnerability Detection and Mitigation Method

Web Vulnerability	Detection	Prevention	Combination
SQL Injection	2	1	2
SQL, XML, XPath Injection	1	0	0
SQL Injection, XPath Injection	0	1	0
XPath Injection	0	1	0
Cross-Site Scripting	12	1	0
Spoofing	0	1	0

5.2 What are the web service vulnerabilities issues that are identified in the research papers?

Web services have various vulnerabilities, ranging from injections to Cross-Site Scripting attacks. The following sections elaborate on them.

5.2.1 SQL Injection

An SQL Injection attack is a vulnerability that injectsa database of a related web application with malicious code and exploits users' data. Table 2 identified that 32% (7) of papers address SQL injection attacks. Among the 7 papers, 71.42% of them (5) discuss other attacks such as XPath, XML injection, cross-site, and spoofing.

5.2.2 XML Injection Vulnerability

An XML injection vulnerability exists when a service fails to verify malicious XML information. Injecting malicious XML material into any service mightcause it to malfunction.As perTable 2, only paper (01) discusses XML injection attacks along with other attacks.

5.2.3 Cross-Site Scripting

Cross-Site Scripting (XSS) is a security flaw in which an attacker may inject malicious code (JavaScript, VBScript, ActiveX, HTML, or Flash) into a vulnerable dynamic page and gather data by running the script on their machine. The usage of XSS may compromise sensitive information, modify or steal cookies, generate requests that might be misinterpreted as those of a legitimate user, or execute malicious malware on end-user computers. XSS vulnerabilities are addressed in 59.09% of the papers (13) in Table 2.

5.2.4 Spoofing

Spoofing vulnerabilities, also known as content injection or arbitrary text injection, are attacks that target a user and are enabled via an injection vulnerability in a web application. An attacker can submit material to a web application, generally via a parameter value, which is mirrored back when the application does not correctly manage user-supplied data. In Table 2, spoofing vulnerabilities are addressed only in 1 paper.

5.3 What are the methods proposed to solve web service vulnerability issues?

This study mainly focuses on web application vulnerabilities. Atotal of 22 studies are reviewed in Table 1. 07 papers focus on SQL Injection vulnerability methods used for detection and mitigation are model-based approachesto prevent SQL Injection attacks on .NET applications.Predictive errors of SQL Injection attack detection by the feature of the single character.Obfuscation-based analysis of SQL Injection attacks.Input mutation approach automated testing for SQL Injection vulnerabilities.Code Conversion Method is an efficient technique for detecting and preventing SQL Injection attacks using ASCII-based string matching.

The XPath and SQL Injection vulnerability methods used for detection and mitigation are model-based system architectures for preventing XPath injection in a database-centric web services environment. A novel approach is jointly used for SQL, XML, and XPath injection to complement existing vulnerability detection by forming sound and precise slices, thus identifying false and true positive values. This approach is used for preventing SQL and XPath injection attacks on web services by combining statement learning with service protection.

Cross-Site Scripting (XSS) vulnerability methods used for detection and mitigation are a genetic algorithm is used to detect an XSS vulnerability in a PHP web application. An algorithmic approach to detect and remove XSS vulnerability using data mining techniques. Gray box mechanism HTML output cooperates with the database context-sensitive XSS flaws HTTP e OWASP Zed Attack Proxy. MCTS, GAN deep learning CICIDS2017 dataset detection for XSS attacks based on Generative Adversarial Networks. ANN, MLP, DFE dynamic-features extraction technique an integrated XSS-based attack detection scheme in web applications using multilayer perceptron technique. DOM-based XSS attack Novel taint tracking analysis is used for the dynamic detection framework.The misuse pattern method is used for the prevention of Spoofing vulnerability.

5.4 Which are the web servicevulnerability areas focused on in the research papers?

This study mainly focuses on web application vulnerability detection and prevention. 20 studies (90.90%) focused on web vulnerability detection and prevention, and 02 papers (9.09%) concentrated on combined techniques for detection and prevention. Furthermore, the strategies are further classified as vulnerability detection and prevention andblended.

Table 2 shows how the approaches are classified into four groups. Having 68.18 percent, attack detection is the vital individual area, followed by attack prevention, which has a paper contributing 22.72 percent. It is a beneficial strategy

since it requires not only the implementation of contingency measures once an attack has occurred, but also a thorough investigation into how to avoid the online vulnerability itself.

6. DISCUSSION

This paper summarises the existing detection and mitigation methods. From Table 1, we have mentioned web vulnerability detection and prevention methods.SQL Injection is a significant issue that affects SQL databases. Innumerable ways are used to detect and prevent SQL Injection, including Model-based approach, feature of Single character, Obfuscation-based, Input mutation approach, Novel approach, and Code conversion method, all of which have some severe issues which can harm user data directly.

Cross-site scripting is the most dangerous web vulnerability.It is used to exploit and harm web applications and users. In our assessment,Genetic algorithm, Gray box mechanism, MCTS, GAN deep learning CICIDS2017 dataset, ANN, MLP, DFE dynamic-features extraction technique, and DOM-based XSS attack Taint tracking analysis methods are used for detection and prevention.All methods have some limitations which the attackers can take advantage of and harm the users data.

It is also required to improve the system for detecting and preventing XML/XPath injection and spoofing attacks. The vulnerability detection technique suffers from false positive and false negative problems. It needs to be updated for detection and newly coming vulnerabilities.

As per Table 1, although researchers have developed the methods to detect and prevent web application vulnerability attacks, we observed that issues are still present because millions of websites are hacked each year and victim to many attacks [3]. There is a necessity for more investigation and analysis of web application vulnerability detection and mitigation techniques.

7. CONCLUSION

Web services have emerged as the primary means for vital information to be effortlessly shared across web applications. As a result, web service security is a critical component, and web service attacks pose a noteworthy danger to data integrity and availability. We have systematically reviewed 22 research papers related to web vulnerability detection and mitigation approaches. After analyzing all the methods, this review found 15 research papers associated with detecting the vulnerability, 05 research papers preventing the vulnerability, and 02 research papers that are combined approaches to detect and prevent web vulnerabilities. A brief overview of detection and mitigation methods used in web vulnerability assessment is also presented. The results obtained from this investigation can be helpful for researchers and practitioners. The results of this investigation provide a comprehensive table of issues for detection and mitigation methods. It would help researchers find new research opportunities in the vulnerability detection and mitigation domain. Practitioners can apply these results to their development practices and resolve the problem related to web vulnerabilities. In the future, we will develop a new approach to detect and mitigate web application vulnerabilities.

8. REFERENCES

[1] Indian Computer Emergency Response Team (CERT-In), “Annual Report-2017”, Ministry of Electronics & Information Technology, Government of India, 2018.

[2] Statista, “Number of web attacks blocked daily worldwide 2015-2018”,[Online], Available:https://www.statista.com/statistics/494961/web-attacksblocked-per-day-worldwide/, 2019.

[3] Businesses standard, India becomes favourite destination for cyber criminals amid Covid-19, 6 April 2021. [Online], Available: https://www.business-standard.com/article/technology/india-becomes-favourite-destination-for-cyber-criminals-amid-covid-19-121040501218_1.html.

[4] Monika Rohilla et. al., “XSS Attack: Detection and Prevention Techniques”, International Journal of Scientific & Engineering Research, Volume 7, Issue 12, December-2016.

[5] Minamide, Y., “Static Approximation of Dynamically Generated Web Pages”, WWW '05 Proceedings of the 14th international conference on World Wide (pp. 432-441). New York, NY, USA: ACM, 2005.

[6] Jovanovic, N., Kruegel, C., &Kirda, E., “Pixy: a static analysis tool for detecting Web application vulnerabilities”, IEEE Symposium on Security and Privacy (S&P'06), (pp. 6pp.-263), Berkeley/Oakland, CA: IEEE, 2006.

[7] Doupe, A. et. al., “deDcota: Toward Preventing Server - Side XSS via Automatic Code and Data Seperation”, CCS'11, Berlin Germany, ACM, 2013.

[8] A, V. B., & P, J. K., “Prediction of Cross Site Scripting Attack Using Machine Learning Algorithm”, ICONIAAC, Amritapuri India: ACM, 2014.

[9] Kirda E. et. al., “Client - Side Cross Site Scripting protection”, 2009.

[10] Duraisamy, Kannan, &Selvamani., “Protection of Web Application from Cross Site Scripting Attack in Browser Side”, IJCSIS, 229-236, 2010.

[11] Shalini, & Usha, “Prevention of Cross Site Scriptig Attack (XSS) on Web Application in The Client Side”, IJCSI International Journal of Computer Science Issue, 2011.

[12] Balzarotti D. et al., “Paper Review: Saner: Composing Static and Dynamic Analysis to Validate Sanitization in Web Applications”, Security and Privacy, IEEE symposium (pp. 378-401), Oakland CA: IEEE, 2008.

[13] Akhawe B., Saxsena, F., &Weinberge, S., “A Systematic Analysis of XSS Sanitization in Web Application Framework”, ESORICS'11 Proceeding of the 16th European Conference on Research in Computer Security, ACM, 2011.

[14] Duraisamy A., Sathiyamoorthy M., & Chandrasekar S., “A Server-Side Solution for Protecting of Web Application from Cross-Site Scripting Attack”, International Journal of Innovative Technology and Exploring Engineering (IJITEE), 2(4), March - 2013.

[15] M. Jensen et. al., “A survey of attacks on web services”, Computer Science-Research and Development, 2009. 24(4): p. 185-197.

[16] P. Kumar and R. Pateriya, “A survey on SQL Injection attacks, detection and prevention techniques”, In Computing Communication & Networking Technologies (ICCCNT), Third International Conference on 2012, IEEE.

[17] X.G.R. Chaudhariand M.V. Vaidya, “A Survey on Security and Vulnerabilities of Web Application”, IJCSIT, 2014.

[18] S. Srivastava, “A Survey On: Attacks due to SQL Injection and their prevention method for web application”, 2012.

[19] X. Liand Y. Xue, “A survey on server-side approaches to securing web applications”, ACM Computing Surveys, 2014. 46(4): p. 1-29.

[20] U. Agarwal et. al., “A Survey of SQL Injection Attacks”, International Journal of Advanced Research in Computer Science and Software Engineering, 2015.

[21] Shikhar Jain & Alwyn R. Pais, "Model-Based Approach to Prevent SQL Injection Attacks on .NET Applications", International Journal of Computer Science & Informatics, Volume-I, Issue-H, 2011.

[22] Takeshi Matsuda et al., "On predictive errors of SQL injection attack detection by the feature of the single character" Systems, Man, and Cybernetics (SMC), 2011, IEEE International Conference on 9-12 Oct 2011, On Page 1722-1727.

[23] Raju Halder and Agostino Cortesi, "Obfuscation-based Analysis of SQL Injection Attacks", IEEE, 978-1-4244-7755-5/10/$26.00, 2010.

[24] D. Appelt et al., “Automated testing for SQL injection vulnerabilities: an input mutation approach”, International Symposium on Software Testing and Analysis; p. 259-26, ACM, 2014.

[25] Thome J, Shar LK, & Briand L., “Security slicing for auditing XML, XPath, and SQL injection vulnerabilities”, IEEE 26th International Symposium on Software Reliability Engineering, pp. 553-564, 2015.

[26] Laranjeiro N, Vieira M, & Madeira H, “A Learning-Based Approach to Secure Web Services from SQL/XPath Injection Attacks”, IEEE 16th Pacific Rim International Symposium on Dependable Computing, pp. 191-198, 2010.

[27] IndraniBalasundaram and E. Ramaraj "An Efficient Technique for Detection and Prevention of SQL Injection Attack using ASCII Based String Matching", International Conference on Communication Technology and System Design 2011 © 2011 Published by Elsevier Ltd. Selection and/or peer-review under responsibility of ICCTSD, 2011.

[28] A. Asmawi et al., “Model-based system architecture for preventing XPath injection in database-centric web services environment”, 7th International Conference on Computing and Convergence Technology, pp. 621-625, 2012.

[29] Marashdih Abdalla Wasef et al., “Web Security: Detection of Cross Site Scripting in PHP Web Application Using Genetic Algorithm”, International Journal of Advanced Computer Science and Applications, 8 (5): 64-75, 2017.

[30] Medeiros et al., “Detecting and Removing Web Application Vulnerabilities with Static Analysis and Data Mining”, IEEE Transactions on Reliability 65 (1): 54-69, 2016.

[31] Steinhauser Antonín, and Petr Tůma, “Database Traffic Interception for Graybox Detection of Stored and Context-Sensitive XSS”, arXiv preprint arXiv:2005.03322, 7 Aug, 2020.

[32] Zhang Xueqin, et al., “Adversarial Examples Detection for XSS Attacks Based on Generative Adversarial Networks”, IEEE Access 8: 10989-10996, 2020.

[33] Fawaz Mahiuob Mohammed Mokbal, et al., “MLPXSS: An Integrated XSS-Based Attack Detection Scheme in Web Applications Using Multilayer Perceptron Technique”, IEEE, 2019.

[34] Ran Wang, et al., “TT-XSS: A novel taint tracking based dynamic detection framework for DOM Cross-Site Scripting”, Journal of Parallel and Distributed Computing 118: 100-106, 2018.

[35] H. Shahriar and M. Zulkernine, "S2XS2: A Server-Side Approach to Automatically Detect XSS Attacks”, 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, Sydney, NSW, pp. 7-14, 2011.

[36] J. Pan and X. Mao, "DomXssMicro: A Micro Benchmark for Evaluating DOM-Based Cross-Site Scripting Detection", 2016 IEEE Trustcom/BigDataSE/ISPA, Tianjin,2016, pp. 208-215, 2016.

[37] Tawfiq S. Barhoom and Sarah N. Kohail, “A new server-side solution for detecting Cross Site Scripting attack”, International Journal of Computer Information Systems, Vol. 3, No. 2, 2011.

[38] Shashank Gupta and B. B. Gupta, “Automated discovery of JavaScript code injection attacks in PHP web applications”, International Conference on Information Security & Privacy (ICISP), Nagpur, INDIA, 11-12 December 2015, Elsevier, Procedia Computer Science, vol. 78, pp.82 – 87, 2016.

[39] M. Gundy and H. Chen, “Noncespaces: Using Randomization to Enforce Information Flow Tracking and Thwart Cross-site Scripting Attacks,” Proc. of NDSS, San Diego, Feb. 2009.

[40] Prithvi Bisht, V. N. Venkatakrishnan, "XSS-GUARD: Precise Dynamic Prevention of Cross-Site Scripting Attacks”, Detection of Intrusions and Malware, and Vulnerability Assessment Systems and Internet Security Lab, Department of Computer Science University of Illinois, Chicago, pp. 23-43, 2008.

[41] B. Mewara, et al., "Enhanced browser defense for reflected Cross-Site Scripting”, Proceedings of 3rd International Conference on Reliability, Infocom Technologies and Optimization, Noida, pp. 1-6, 2014.

[42] Muñoz-Arteaga J, et al., “Misuse pattern: spoofing web services”, 2nd Asian Conference on Pattern Languages of Programs, 2011.