Tackling Threats: A Study of Vulnerability Testing and Mitigation in Web Applications

Muhammed Ismaeel[1], Prashant Lokhande[2]

[1] Computer Department, Pillai College of Engineering, New Panvel, Mumbai, Maharashtra - 410206, India

[2] Professor, Computer Department, Pillai College of Engineering, New Panvel, Mumbai, Maharashtra - 410206, India

ABSTRACT

The abstract of the study "Tackling Threats: A Study of Vulnerability Testing and Mitigation in Web Applications" delves into the pressing significance of web application security within today's digital environment cannot be overstated. Because online applications are used for more and more purposes, there is a greater chance that malicious attacks and cyber threats that target weaknesses may occur. The study emphasizes the necessity of proactive measures, including vulnerability assessment, penetration testing, and effective mitigation strategies, to safeguard digital assets against cyber-attacks. Advanced techniques such as deep learning methods, algorithms, and machine learning solutions are recommended to enhance the accuracy and scalability of identifying and addressing web application vulnerabilities. The paper also highlights the significance of continuous research and innovation in strengthening security measures, developing modern defenses, and expanding the scope of vulnerability detection to combat evolving cyber threats. Collaboration among researchers and ongoing efforts to enhance web application security are crucial in protecting against the ever-evolving cyber threats in the digital realm. The abstract underscores the importance of robust security practices and the need for continuous improvement in web application security to mitigate risks effectively.

Keywords: Web Application Scanner tool, Penetration Testing, Mitigation Strategies, Web site secure, vulnerabilities, Ethical Hacking, Cyber Security, Digital Assets Protection, Online Security, Data Privacy.

1. Introduction

Web applications are becoming more and more popular as a trend in technology. They allow users to access information from anywhere at any time and save time and effort while creating enterprises, learning, or communicating [2]. For this reason, web apps start to play a bigger part in our lives. When individuals use web apps, they provide the organization with personal information, which is subsequently stored on them. However, some unscrupulous and self-serving attackers take use of the online application to obtain illegal access and carry out additional crimes like identity theft, invasions of privacy, and other cyberattacks. Because of the web application's vulnerabilities, these unlawful points give the attackers the freedom to create anything they wish [2].

Malicious actors are always seeking for those opportunities. malicious actors can easily plant malicious content on your server, plant backdoors allowing them later access, thus exposing your own internal network [3].

The creation of automatic methods for identifying web vulnerabilities is essential given the quantity and variety of web applications[5].

As more people use it, the Internet's penetration power grows dramatically, allowing it to become more widely used worldwide and ingrained in society's cracks and crevices[8]. This is demonstrated by the fact that between 2021 and 2022, the global user base increased annually from 59.5% to 62.5% [8], [13]. Thus, the quantity of web designers and online applications as built is likewise increasing in an effort to match user demands[8]. There were 537 million active websites and 26.2 million web developers globally in 2020, according to [8], [14]. By 2023, there will be 27.2 million web developers worldwide, according to predictions [8].

Therefore, this review of the literature explores the practices of vulnerability assessment and mitigation in web application security, offering an understanding of the methods, resources, and strategies that professionals employ. Our study intends to equip stakeholders to proactively protect their digital assets against increasing cyber threats by evaluating strengths, weaknesses, and emerging trends. We discuss testing procedures, common vulnerabilities, and practical mitigation techniques in the sections that follow. By means of this examination, we augment the robustness of web applications in the current hostile digital landscape.

2. Literature Review

Jaydeep R.Tadhani1 , and VipulVekariya (2024) authors said a hybrid deep learning approach to protect online applications from SQLi and XSS assaults, combining CNNs and LSTMs. The model outperforms conventional machine learning techniques in detecting assaults with a high degree of precision and little false positive rates. The method entails decoding HTTP requests and payloads and standardizing them. The paper recommends that in order to increase the models' accuracy and robustness and to enable them to detect more security threats, future research should investigate other architectures and data kinds. The authors emphasize how CNNs and LSTMs can protect web applications from different types of cyberattacks.

The algorithm known as XSS_SQL_Scanning_Algorithm, developed by Thinzar Aung and Zin Thu Thu Myint (2023), was utilized in the method for locating online application vulnerabilities that is detailed in this article, specifically SQL injection and Cross-site Scripting (XSS) attacks. The algorithm consists of multiple crucial steps, such as gathering all potential URLs by scanning the entire web application, parsing the URLs to determine input points, forwarding attack payloads to identify the web application, and utilizing the Naïve pattern matching algorithm to analyze the response and find potential vulnerabilities. With a 100% accuracy rate in vulnerability detection, the algorithm is engineered to identify vulnerabilities with a low rate of false positives and false negatives. The algorithm is a trustworthy and efficient tool for web application security because it uses less memory than the popular commercial program Acunetix. As part of the approach, the algorithm's performance is also assessed, comparing it to Acunetix and showcasing its precision and effectiveness in vulnerability detection.

Accuracy = ((TPR+TNR)/(TPR+FPR+TNR+FNR)) *100[2] is the formula used to compute accuracy, where True positive rate (TPR), false positive rate (FPR), true negative rate (TNR), and false negative rate (FNR) are the acronyms for these terms[2].

Andrei-Daniel ANDRONESCU and Ioana-Ilona BRĂSLAȘU et al. (Vol. X / 2023) state that to detect successful attacks, the scanner searches for particular signatures or anomalies in the response data it receives, such as unusual status codes or error messages.
Attack payloads based on HTTP are created via the malicious payload generator. These payloads are intentionally created to verify vulnerabilities without executing any harmful code. The scanner's goal is to harmlessly mimic the actions of actual attackers.
In the payload injection step, the payloads are injected into the web application after they have been generated. The scanner determines whether an attack was successful by examining the server answers. Anomalies or particular signs pointing to a successful attack are found using detection heuristics tailored to each kind of attack.
The document does not provide specific equations or mathematical formulas related to the methodology[3].

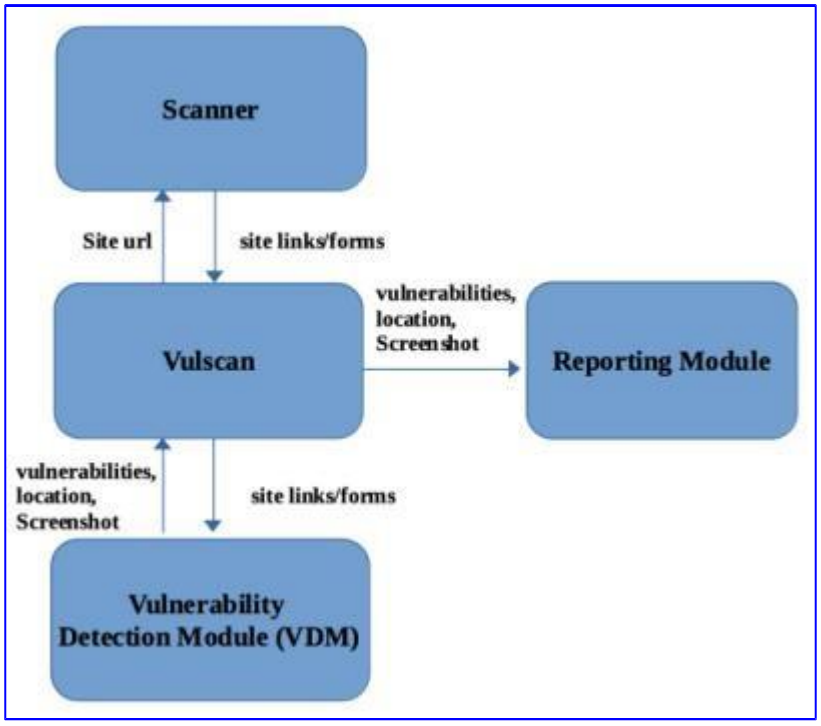
The paper's methodology leverages machine learning (ML) to determine web application vulnerabilities, including those related to Cross-Site Request Forgery (CSRF). The first machine learning method for detecting cross-site scripting vulnerabilities in a dark environment is dubbed Mitch, and its design is suggested by the DR. T. S. GHOUSE BASHA1, and P SANDHYA Vol.14 No.02(2023) authors.
The process entails using a collection of data over Six Thousand HTTP requests came from active webpages that were gathered and annotated by people specialists in order to train a machine learning classifier. Each of the 49 dimensions that make up the classifier's feature space X represents a distinct attribute of an HTTP request, such as the quantity, kind, and length of the request. After that, this classifier is employed to examine HTTP traffic by locating CSRF flaws in online applications.
In order to detect false positives and false negatives generated by the classifier, the process also includes manual testing. The work does not make explicit reference to equations or mathematical models. Nonetheless, two of the methodology's mathematical techniques may be identified: the use of feature space X and the ML classifier's training procedure.

Erik Trickel! , and Fabio Pagani (2023) authors stated that the Witcher is an online vulnerability discovery tool that finds Vulnerabilities related to SQL and command injection in web applications using grey-box coverage-directed fuzzy logic. enhance vulnerability detection also application coverage, it applies coverage-guided input generation and fault escalation. Witcher performs better than other scanners in terms of both vulnerability detection and web application coverage, while being restricted to detecting reflected injection vulnerabilities and requiring instrumentation of the target application. In future research, the authors propose to integrate Witcher with other scanners and investigate the effectiveness of fuzzers on online applications.

Chun Lin , Yijia Xu et al Appl. Sci. 2023 said that this study employs a unique vulnerability detection method for graph neural networks technique named VulEye as its methodology. The method is building the PHP source code's Program Dependence Graph (PDG), slicing the PDG into Sub-Dependence Graphs (SDGs) with sensitive functions, and utilizing the SDGs as the input that a Graph Neural Network model is trained on. Three units in stack form with a Graph Convolutional Network (GCN), pooling of top-k and attention layer make up the model of a graph neural network. These tiers are intended to identify and extract the characteristics of source code vulnerabilities. Equations are utilized to express concepts, such as the global soft attention score, the degree matrix, and the propagation mode in the GCN model. the degree matrix computation, and the global soft attention score calculation. The method makes use of the source code's graph structure and semantic properties to precisely detect the existence of vulnerabilities in PHP applications.

Deeptha R.1* , K.Sujatha 2 , et al. (16 May 2023) authors state that the creation and use of a vulnerability scanning tool intended to identify different kinds of vulnerabilities in web applications is probably the methodology used in the study article on Website Vulnerability Scanner. This could involve building a test-ready, vulnerable web environment, applying scanning algorithms to find security flaws like SQL injection, cross-site scripting, failed authentication, payload, and email disclosure, and evaluating how well the scanner works in terms of identifying and categorizing system flaws. The approach may also take into account network stability aspects, such as employing a stable Wi-Fi modem for dependable scanning activities, and make use of currently available technologies, such as AcuMonitor, to improve the accuracy of vulnerability data.

Tobias Osemegbe Odion,and Ife Olalekan Ebo et al.(2023) authors said that the three primary modules of the VulScan methodology are the three modules: reporting, vulnerability detection (VDM), and scanner. These modules provide an authenticated scanning strategy. While the VDM tests for vulnerabilities including SQL injection, DDoS, DoS assaults, XSS, CSRF, SSL stripping, clickjacking, and Scanner module navigates the online site, finding links, forms, and input fields. To evaluate the security posture of the online application, equations for vulnerability detection (e.g., SQL syntax analysis for SQL injection detection) may be used in the VDM. The Reporting Module gives users a thorough an image showing the web's security state application by producing comprehensive reports on the vulnerabilities found, their severity, and suggested mitigation techniques.



The proposed VulScan detection system is shown in Figure 1 [5].

Ann Zeki Ablahd (vol.13, n°2, January 2023) author described that using the Flask and Django frameworks in conjunction with the Python 3.7 programming language, the paper's methodology develops the SCANSCX tool, which is capable of identifying web application vulnerabilities such the execution of commands problems as well as injection of Cross-Site Scripting (XSS). The program makes use of Python
3.7's features to evaluate different web apps and spot possible security issues. The presented passages do not specifically address any equations or mathematical models related to the methodology.

Jingyu Zhang et al (2021) stated that a Cross Site Request Forgery through a browser (CSRF) detection mechanism is described in the document. The suggested model finds and stops cross-site request forgeries by examining HTTP requests and web page content. A web server, CSRF handler, content analysis, and HTTP request analysis are some of the components that make up the model architecture. In order to identify potential CSRF attacks and notify the user, the model analyses the HTTP request page before to loading each page. It also intercepts suspicious HTTP requests. There are no particular equations pertaining to the approach or technique in the paper.

Ines Jemal1 , Omar Cheikhrouhou2 (2020) described and discussed new proposed solutions, including ontology- and machine learning-based solutions.
Machine learning based SQLIA countermeasures:

The paper discussed the application of machine gaining knowledge techniques to identify and prevent attacks using SQL injection. It classifies machine learning methods into supervised, unsupervised, and reinforcement learning.
Equation: The paper discusses how accuracy and precision are used as important metrics to test machine learning models. Equations for accuracy and precision are given as follows.
Specifically = (TP + TN) / (TP + FP + FN + TN) . Specificity = TP / (TP + FP) 1 .
The paper also includes a confusion matrix for defining features needed to evaluate machine learning classifiers.
Furthermore, the paper presents a comparison between different SQL injection solutions, focusing on their capacities for reporting, preventing, and detecting generation, mapping and classes.

Prakhar Tripathi , and Rahul Thingla (SUSCOM-2019) authors described the document's technique focuses on using vulnerability scanning tools to identify and stop XSS and SQL-injection attacks on websites applications. In order to find and fix security flaws, the method comprises examining input fields, forms, and keywords for possible vulnerabilities, looking for special characters and Boolean values, and comparing the collected data. This paper presents an algorithm for SQL-Injection free Secure Algorithm that includes form data input, vulnerability checking, keyword and special character compilation, and vulnerability data collection. The algorithm's flow chart illustrates the procedure in visual form.

3. Observation

1) Further research examining various CNN and LSTM model architectures and fusion methods can improve accuracy and robustness. Including other data types, such as network metadata, has the potential to enhance the identification of security vulnerabilities. There are several research opportunities in expanding model applications to detect bad URLs, phishing attempts, and botnet attacks. With continued improvement, the hybrid CNN and LSTM technique has the potential to be an important tool for protecting online applications from cyber attacks. It is worthwhile to look at model training efficiency optimizations such as scaling and normalization techniques. The method to identify DDoS and zero-day assaults may be expanded through additional study using fresh datasets.

2) Using Naïve pattern matching algorithm, we can identify more flaws including command injection, overflowing buffer, and cross-site request forgery additionally also suggest possible mitigation strategies.

3) While there have been advances in vulnerability testing for web applications, there are differences in the

breadth and depth of attacks that such software can successfully detect, detect and report Current tools may not cover an emerging threat or surprise attack potential weaknesses go unnoticed.
Subsequent investigations may concentrate on augmenting automated vulnerability testing software's functionalities by broadening the spectrum of attacks it can accommodate. This could entail creating fresh methods or algorithms to efficiently identify and counteract changing cyber threats. Furthermore, there exists opportunities for reframing current initiatives in order to optimize scalability and augment the tool's capacity to evade increasingly sophisticated bot detection systems. Additionally, efforts ought to be focused on making sure that these technologies are developed and used ethically, with a dedication to protecting the security and integrity of online applications without jeopardizing live servers.

4) There is still a gap in the development of machine learning (ML) solutions for detecting several classes of vulnerabilities beyond CSRF, even though ML shows promise in tackling web application vulnerabilities. While current research may concentrate on particular weaknesses, it does not provide a thorough analysis of all risks. Furthermore, the complexity and diversity of online applications might affect how successful machine learning (ML) solutions are, which emphasizes the need for more research into optimization strategies.
Subsequent investigations may broaden the scope of ML solutions developed and assessed to identify vulnerabilities in web applications other than CSRF.
In order to improve detection accuracy and scalability, this may entail investigating cutting-edge techniques, algorithms, and procedures. Furthermore, efforts must to be directed on verifying the efficacy of machine learning solutions in various web application scenarios and environments. Working together, researchers may be able to exchange expertise and implement best practices for ML-based vulnerability detection, which could eventually result in the development of more powerful and trustworthy automated web application inspection tools.

5) The Witcher prototype's shortcomings draw attention to a number of areas that still need investigation in the area of web application vulnerability detection. First off, the existing prototype can only identify vulnerabilities related to SQL injection and command injection; it is unable to identify additional serious flaws like Path traversal, local file inclusion, remote code evaluation, and cross-site scripting. Furthermore, Witcher has serious limitations because to its incapacity to identify second-order weaknesses like SQL injection stored in memory and its lack of state reasoning for online applications. The prototype's inability to comprehend multi-state activities in online apps is further limited by its single URL fuzzing methodology.
Subsequent investigations may tackle these constraints and enhance the functionalities of vulnerability detection systems like as Witcher. A wider range of vulnerabilities such as cross-site scripting and local file insertion could be detected by expanding detection capabilities, improving simultaneous crawling and fuzzing to share results and interweave URL executions, and creating methods to automate initial setup and configuration. Furthermore, investigating methods for comprehending and reasoning about the status of online applications could enhance the efficacy of vulnerability detection technologies. Additionally, improving methods that produce inputs that are both syntactically and semantically correct may result in more thorough and precise web vulnerability identification. Overall, there is a lot of room for study into automating the process of finding vulnerabilities in web applications, which will help close the gap that exists between web application security and fuzzing techniques.

6) VulEye's shortcomings—most notably, its incapacity to manage function calls among several PHP files—draw attention to a research void in the area of Graph Neural Network (GNN) vulnerability detection. This restriction limits VulEye's ability to analyze more intricate and large-scale PHP programs. Furthermore, even though VulEye performs well on the SARD datasets, more research is necessary to determine whether or not it can be applied in the real world and how well it can identify vulnerabilities in projects that are in production.
Subsequent investigations can concentrate on improving Vulnerability identification in graph neural networks methods for PHP scripts and resolving VulEye's shortcomings. This can entail creating processes to manage

function calls among various PHP files so that VulEye can more efficiently assess larger PHP projects. Furthermore, carrying out comprehensive trials in actual production projects would offer important insights regarding VulEye's functionality and effectiveness in real- world situations. It would also be helpful to investigate methods to raise VulEye's general accuracy and efficiency as well as its capacity to identify particular vulnerabilities. Overall, further study is needed to improve and optimize Graph Neural Network vulnerability detection techniques like VulEye in order to provide more accurate and dependable security analysis of PHP scripts.

7) Neglected Vulnerabilities: Researchers haven't paid much attention to some online application vulnerabilities, such as Invalidated Redirects and Forwards/Under Protected APIs, which suggests that in order to strengthen web security, more attention needs to be paid to these dangers that are often ignored.
Information Overload: Ineffective mitigation is hampered by the fact that current vulnerability scanning tools frequently produce too detailed corrective reports. For more effective vulnerability management, these reports must be improved and streamlined.
Enhanced Security Measures: Future studies can look into ways to strengthen security strategies and create cutting-edge defenses against constantly changing cyberthreats. Innovative Solutions: To more effectively lessen online threats, web scanning and security solutions can benefit from innovation. This entails investigating fresh approaches, structures, and technological advancements to fortify web security protocols.
Education on Cybersecurity: Higher education's cybersecurity curriculum can be improved by creating practical labs and instructional materials that concentrate on vulnerability scanning, producing cybersecurity professionals who are better equipped. Limited Scope of Vulnerability Detection: Although major online application vulnerabilities including DDoS, DoS attacks, XSS, CSRF, SSL stripping, SQL injection, and clickjacking are addressed by the proposed framework, VulScan, there might be more new vulnerabilities that are not addressed. This suggests that more study is needed to broaden the scope of vulnerability identification to include more potential threats.

8) Absence of a Complete Approach: VulScan provides a multimodal approach to vulnerability identification via active and passive operations, however it might not fully handle every part of web application security. Subsequent investigations may concentrate on creating more comprehensive strategies for efficiently identifying and mitigating risks.
Extension of Attack Detection: Beyond the six already recognized forms of assaults, future development may involve expanding VulScan's detection capabilities. To improve web application security, this can entail investigating new risks and creating appropriate detection systems.
Integration of Machine Learning: Using machine learning techniques to improve VulScan offers a promising path toward increasing the precision of vulnerability identification and performance assessment. Future studies could investigate how one can use machine learning methods to enhance VulScan's functionality and improve its ability to respond to changing threats. Complete Security Solutions: It is possible to create complete security solutions that offer strong mitigation techniques in addition to vulnerability detection. To improve overall web application security, future research could concentrate on combining VulScan with proactive security measures and automated remediation methods.

9) Despite the SCANSCX tool's efficacy in identifying XSS and command execution vulnerabilities, there may be a research gap in the study due to its limited attention on assessing the tool's scalability and performance when applied to large-scale online applications.
SCANSCX's effectiveness and usability can be greatly increased by enhancing its detection capabilities through the use of machine learning algorithms or sophisticated heuristics; integrating the tool with current security

frameworks for comprehensive web application security; automating vulnerability scanning and report generation; guaranteeing cross-platform compatibility; and encouraging community collaboration for ongoing development and addressing evolving threats in web application security.

10) Restricted Detection Scope: Although the suggested CSRF detection model concentrates on examining HTTP requests and web page content to spot possible CSRF assaults, it might miss some complex attack paths or CSRF attack variants. The necessity for ongoing improvement and development of detection capabilities is indicated by the possibility of gaps in the ability to identify new attack patterns or evasive tactics used by attackers.
Absence of Real-Time Protection: While detecting CSRF vulnerabilities is the main goal of the CSRF detection methodology, it does not offer real-time defense against CSRF assaults. In the future, research may focus on creating proactive defenses against CSRF attacks in real time, like behavior analysis methods or token-based authentication.
Improved Detection Methods: In order to respond to changing attack situations, future work may entail improving and fine-tuning the detection methods employed in the CSRF detection model. This can entail using sophisticated heuristics or machine learning methods for raising the precision and efficacy of CSRF identification. Real- Time Defense Mechanisms: To actively counteract CSRF assaults as they happen, real-time defense mechanisms can be developed within the CSRF detection paradigm. To stop harmful requests in real-time, this can entail implementing intrusion prevention systems or incorporating automatic response mechanisms.
Integration with Web Security Frameworks: To offer a thorough defensive mechanism against a variety of web-based threats, the CSRF detection model could be combined with currently available web security frameworks or tools. This integration may improve the web's overall security posture.

11) The SQLIA algorithm is not suitable for handling a large number of requests per second due to the lack of an updated and standardized datasets. Although these methods are straightforward and simple to use, it can be challenging to have all the legal query models. The web application may become inoperable due to a high false positive rate caused by an inadequate number of legal query models.
To improve SQL injection detection, researchers could combine machine learning with ontology-based or monitoring techniques. To evaluate practical effectiveness and performance in live web settings, real-world deployment and evaluation are essential. This includes thorough testing under varied traffic situations and attack scenarios to validate robustness and reliability.

12) The constraints of current methods and tools for identifying and stopping SQL injection and cross-site scripting (XSS) attacks within online applications represent the research gap in the paper. Although there are several vulnerability scanning programs accessible, more sophisticated and all-encompassing techniques are required to properly handle these security risks.
The subsequent studies in this field will focus on creating increasingly advanced instruments and methods for identifying and addressing XSS and SQL injection problems. When it comes to organizing and putting in place security safeguards for web applications, the industry needs to mature. Furthermore, there exists an opportunity for additional comparison analyses of current tools to assist users in selecting the best tool for their particular requirements. In general, the future scope entails deepening our knowledge of web vulnerabilities and strengthening defenses against XSS and SQL injection assaults.

4. Conclusion And Knowledge Gaps

The study emphasizes the critical importance of web application protection in brand new digital landscape due to the increasing range of cyber threats and malicious actors concentrated on vulnerabilities in web applications. It highlights the need for proactive measures together with vulnerability assessment, penetration checking out, and effective mitigation techniques to shield digital assets against cyber-attacks. The use of advanced techniques like deep studying methods, algorithms, and system mastering answers is recommended to decorate detection accuracy and scalability in figuring out and addressing internet software vulnerabilities. Additionally, the paper suggests the significance of non-stop studies and innovation in strengthening security measures, developing contemporary defenses, and increasing the scope of vulnerability detection to address evolving cyber threats. Education on cybersecurity and the improvement of complete techniques for risk identity and mitigation are also highlighted as vital additives in enhancing web utility safety.

Overall, the look at underscores the significance of robust safety practices, collaboration among researchers, and ongoing efforts to improve web application security to guard in opposition to the ever-evolving cyber threats within the digital realm.

Upon meticulously scrutinizing the about sixteen documented assessments concerning the effectiveness of web vulnerability scanners, we discovered six unforeseen and, in our opinion, highly significant discoveries:

1) Though machine learning (ML) holds great potential in mitigating web application vulnerabilities, there is still a significant lack of ML solutions for identifying vulnerabilities other than cross-site scripting vulnerabilities (CSRF). Present research frequently concentrates on particular flaws without providing a thorough examination of all hazards. Furthermore, the performance of machine learning solutions may be impacted by the complexity and diversity of online applications, highlighting the need for additional study into optimization techniques.

2) The Witcher prototype identifies important areas in web application vulnerability detection that need more research. Although it works well for command and SQL injection, it is unable to detect serious vulnerabilities like as XSS, remote code evaluation, local file inclusion, and path traversal. Furthermore, because of its single URL fuzzing approach, it is constrained and suffers from second-order vulnerabilities as well as a lack of state reasoning for online applications.

3) Notwithstanding the effectiveness of the SCANSCX tool in detecting XSS and command execution vulnerabilities, the study's scant focus to evaluating the tool's scalability and performance when used in large-scale web applications may indicate a research gap.

Drawing from these findings, we suggest the following courses of action for the future:

1) To enhance detection precision and expandability, this can involve exploring cutting-edge methods, procedures, as well algorithms. Verifying the effectiveness of artificial intelligence solutions in many web application settings and contexts should also be a priority. By collaborating, scientists may be able to share knowledge and put best practices for ML-based vulnerability identification into practice. This could ultimately lead to the creation of more potent and reliable automated web application inspection tools.

2) Future investigations should focus on enhancing vulnerability detection systems like Witcher by expanding detection capabilities to cover a wider range of vulnerabilities, improving automation, and enhancing methods for understanding the status of online applications.

3) SCANSCX's effectiveness and usability can be greatly increased by enhancing its detection capabilities through the use of machine learning algorithms or sophisticated heuristics; integrating the tool with current security frameworks for comprehensive web application security; automating vulnerability scanning and report generation; guaranteeing cross-platform compatibility; and encouraging community collaboration for ongoing development and addressing evolving threats in web application security.

REFERENCES

1. Tadhani, J. R., Vekariya, V., Sorathiya, V., Alshathri, S., & El-Shafai, W. (2024). Securing web applications against XSS and SQLi attacks using a novel deep learning approach. Scientific Reports, 14(1), 1803.

2. Aung, T., & Myint, Z. T. T. (2023, February). Effective Web Application Vulnerability Testing System Using Proposed XSS_SQL_Scanning_Algorithm. In 2023 IEEE Conference on Computer Applications (ICCA) (pp. 189-193). IEEE.

3. ANDRONESCU, A. D., BRĂSLAȘU, I. I., & NĂSTAC, D. I. (2023, May).

Vulnerability Scanner: Web-based Security Testing. In International Conference on Cybersecurity and Cybercrime (Vol. 10, pp. 43-48).

4. BASHA, D. T. G., SANDHYA, P., SUPRIYA, P., & NAVYA, P. (2023). MACHINE LEARNING FOR WEB VULNERABILITY DETECTION. Turkish
Journal of Computer and Mathematics Education (TURCOMAT), 14(2), 611-618.

5. Trickel, E., Pagani, F., Zhu, C., Dresel, L., Vigna, G., Kruegel, C., ... & Doupé, A. (2023, May). Toss a fault to your witcher: Applying grey-box coverage-guided mutational fuzzing to detect sql and command injection vulnerabilities. In 2023 IEEE symposium on security and privacy (SP) (pp. 2658-2675). IEEE.

6. Lin, C., Xu, Y., Fang, Y., & Liu, Z. (2023). VulEye: a novel graph neural network vulnerability detection approach for PHP application. Applied Sciences, 13(2), 825.

7. Deeptha, R., Sujatha, K., Sasireka, D., Neelaveni, R., & Guru, R. P. (2023). Website Vulnerability Scanner. Journal of Population Therapeutics and Clinical Pharmacology, 30(15), 43-53.

8. Odion, T. O., Ebo, I. O., Imam, R. M., Ahmed, A. I., & Musa, U. N. (2023, April). VulScan: A Web-Based Vulnerability Multi-Scanner for Web Application. In 2023

International Conference on Science, Engineering and Business for Sustainable Development Goals (SEB-SDG) (Vol. 1, pp. 1-7). IEEE.

9. Ablahd, A. Z. (2023). Using python to detect web application vulnerability. Res Militaris, 13(2), 1045-1058.

10. Zhang, J., Hu, H., & Huo, S. (2021). A browser-based cross site request forgery detection model. In Journal of Physics: Conference Series (Vol. 1738, No. 1, p. 012073). IOP Publishing.

11. Jemal, I., Cheikhrouhou, O., Hamam, H., & Mahfoudhi, A. (2020). Sql injection attack detection and prevention techniques using machine learning. International Journal of Applied Engineering Research, 15(6), 569-580.

12. Tripathi, P., & Thingla, R. (2019, February). Cross site scripting (XSS) and SQL- injection attack detection in web application. In Proceedings of International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM), Amity University Rajasthan, Jaipur-India.

13. DataReportal. Digital around the World, 2022 [Online]. Available of https://datareportal.com/global-digitaloverview#:~:text=4.95%20billion%20people%20around%20
the,of%20the%20world's%20total%20population. Accessed on Retrieved on 10/03/2022

14. Nick Galov. A Dive into the ocean of Web Design Stattstics in 2022, 2022 [Online]. Available on https://webtribunal.net/blog/web-design-statistics/#gref. Accesed on 17/5/2022.

15. Dawadi, B. R., Adhikari, B., & Srivastava, D. K. (2023). Deep learning technique- enabled web application firewall for the detection of web attacks. Sensors, 23(4), 2073.

16. Falana, O. J., Ebo, I. O., Tinubu, C. O., Adejimi, O. A., & Ntuk, A. (2020, March). Detection of cross-site scripting attacks using dynamic analysis and fuzzy inference system. In 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS) (pp. 1-6). IEEE.

17. Abdullah, H. S. (2020). Evaluation of open source web application vulnerability scanners. Academic Journal of Nawroz University, 9(1), 47-52.

18. Amankwah, R., Chen, J., Kudjo, P. K., & Towey, D. (2020). An empirical comparison of commercial and open-source web vulnerability scanners. Software: Practice and Experience, 50(9), 1842-1857.

19. Lathifah, A., Amri, F. B., & Rosidah, A. (2022, September). Security Vulnerability Analysis of the Sharia Crowdfunding Website Using OWASP-ZAP. In 2022 10th International Conference on Cyber and IT Service Management (CITSM) (pp. 1-5). IEEE.

20. Jarupunphol, P., Seatun, S., & Buathong, W. (2023). Measuring Vulnerability Assessment Tools' Performance on the University Web Application. Pertanika Journal of Science & Technology, 31(6).

21. Rodríguez, G. E., Torres, J. G., Flores, P., & Benavides, D. E. (2020). Cross-site scripting (XSS) attacks and mitigation: A survey. Computer Networks, 166, 106960.

22. McDonald, S. (2002). SQL Injection: Modes of attack, defense, and why it matters. White paper, GovernmentSecurity. org.

23. ANDRONESCU, A. D., BRĂSLAȘU, I. I., & NĂSTAC, D. I. (2023, May).

Vulnerability Scanner: Web-based Security Testing. In International Conference on Cybersecurity and Cybercrime (Vol. 10, pp. 43-48).

24. Khanum, A., Qadir, S., & Jehan, S. (2023, November). OWASP-Based Assessment of Web Application Security. In 2023 18th International Conference on Emerging Technologies (ICET) (pp. 240-245). IEEE.

25. Sun, H., Cui, L., Li, L., Ding, Z., Hao, Z., Cui, J., & Liu, P. (2021). VDSimilar: Vulnerability detection based on code similarity of vulnerabilities and patches. Computers & Security, 110, 102417.

26. Fidalgo, A., Medeiros, I., Antunes, P., & Neves, N. (2020, October). Towards a deep learning model for vulnerability detection on web application variants. In 2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW) (pp. 465-476). IEEE.

27. Ablahd, A. Z., & Dawwod, S. A. (2020). Using flask for SQLIA detection and protection. Tikrit Journal of Engineering Sciences, 27(2), 1-14.

28. Fadlil, A., Riadi, I., & Mu'min, M. A. (2024). Mitigation from SQL Injection Attacks on Web Server using Open Web Application Security Project Framework. International Journal of Engineering, 37(4), 635-645.

29. Riepponen, M. (2024). Selection of open-source web vulnerability scanner as testing tool in continuous software development.

30. Regano, L., Canavese, D., & Mannella, L. (2024, April). A Privacy-Preserving Approach for Vulnerability Scanning Detection. In Proceedings of the Italian Conference on Cybersecurity (ITASEC 2024). CEUR-WS.

31. Lokhande, P. S. (2012). Performance and security measure of highly performed enterprise content management system.

32. Aslam, F. A., Mohammed, H. N., & Lokhande, P. S. (2015). Efficient Way Of Web Development Using Python And Flask. International Journal of Advanced Research in Computer Science, 6(2).

33. Lokhande, P. S. (2009). Learning from the Past Intrusion Attacks: Digital Evidence Collection to Make e-Commerce Systems More Secure. Conference ICL2009.

34. Lokhande, P. S., & Meshram, B. B. (2016, March). Analytic Hierarchy Process (AHP) to Find Most Probable Web Attack on an E-Commerce Site. In Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies (pp. 1-6).

35. Lokhande, P. S., Ingle, D. R., & Meshram, B. B. (2010). Consideration of critical elements. Active-X security concerns and risks for web development‖ RTSCIT-09.

36. Lokhande, P. S. (2016). SQL Injection Prevention Using Random4 Algorithm.

37. Lokhande, P. S., & Meshram, B. B. (2013). E-Commerce on Cloud: Issues, Attacks & Security. International Journal of Advanced Research in Computer Science, 4(2).