Journal of Information Systems Engineering and Management
2025, 10(4)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Evaluating the Efficacy of Automated Penetration Testing Tools in Identifying Vulnerabilities in Modern Web Applications

Fawaz A. Mereani[1], Emad Shafie[2]
[1]*Corresponding Author | Department of Computer and Applied Science, Applied College, Umm Al-Qura University, Mecca, Saudi Arabia | Email: famereani@uqu.edu.sa*
[2]*Department of Engineering and Applied Science, Applied College, Umm Al-Qura University, Mecca, Saudi Arabia | Email: Eashafie@uqu.edu.sa*

ARTICLE INFO

Received: 10 Nov 2024
Revised: 25 Dec 2024
Accepted: 22 Jan 2025

ABSTRACT

Considering the increasing and rapidly evolving security vulnerabilities of modern web applications, numerous research studies can be undertaken. This study aimed to evaluate the efficacy of five automated penetration tools to detect SQL injection vulnerabilities in modern web applications. An experimental study was done in which five tools were used to test SQL injection, XSS and CSRF. To test the efficiency, the detection rate, precision, recall, scan time and false positive rate were used. Overall, the results suggest that the most robust approach for evaluating the security of web applications involves integrating both automated and manual penetration testing strategies. By combining the strength of automated tools in rapidly scanning and identifying potential vulnerabilities and the insight of manual analysis to verify and investigate the context and impact of these findings, organisations can ensure a more comprehensive security posture. The implications of these findings are pivotal for cybersecurity strategies, encouraging a balanced and holistic approach to vulnerability assessment. Further scope of research lies in testing genetic fuzzy algorithms and combining detection and prevention techniques using single studies.

Keywords: SQL injection attacks, modern web applications, automated penetration testing tools, and manual testing.

Introduction

Evaluation of the efficacy of automated penetration testing tools to identify vulnerabilities of modern web applications

Automated penetration testing tools use software to simulate cyberattacks and identify weaknesses in systems, networks, and applications. They are effective for identifying vulnerabilities in modern web applications, especially when used in conjunction with manual testing, as they can quickly scan for common issues and simulate real-world attacks. There are many types of tools, such as Dynamic Application Security Testing (DAST) and vulnerability scanners (examples: Burp Suit, Core Impact, Nessus). Their efficiency is tested using speed, precision, accuracy and repeatability. Automated tools are most effective when used along with manual penetration testing, focusing on more complex and nuanced vulnerabilities. However, these tools have limitations in the identification of false positives, the complexity of some tools and the possibility of missing some complex vulnerabilities. SQL injection, Cross-site Scripting (XSS), server-side request forgery (SSRF) and security misconfiguration are some common vulnerabilities which can be detected using these tools.

Based on the above information, this study aimed to test the efficacy of five automated penetration testing tools (combined with manual testing) in two test environments. In this study, five tools were tested to test SQL injection, XSS and CSRF. To test the efficiency, the detection rate, precision, recall, scan time and false positive rate were used.

Journal of Information Systems Engineering and Management
2025, 10(4)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Literature Review

The findings obtained by Yadav, Rounak, and Sharma (2024) indicated that their newly developed automated scanner, created using Python and Selenium, outperformed conventional techniques in identifying various vulnerabilities, such as SQL injection, cross-site scripting (XSS), and new threats, particularly when it came to recognising intricate and evolving vulnerabilities.

A system designed to automate the identification of vulnerabilities in web applications demonstrated its ability to utilise the advantages of combining the two forms of automation within the tool. It successfully automated the detection of vulnerability risks and presented the findings to the user in a straightforward manner, proving to be cost-effective and needing minimal user involvement (Moreira, Seara, Pavia, & Serrão, 2024).

Cloud security testing involves assessing the cloud infrastructure and applications for weaknesses and ensuring the protection of sensitive data. These tools, known for their effectiveness, precision, and cost-efficiency, mark a major improvement over conventional manual testing methods. They effectively handle the complexity and scale of cloud operations, lessening manual effort and reducing human errors, which are vital in the intricate and ever-changing landscape of cloud computing. Nonetheless, the deployment and proper utilisation of these automated tools face challenges due to the intricacies of cloud environments, the necessity for ongoing updates and improvements to combat emerging threats, and integration hurdles. These obstacles require a strategic plan that encompasses continuous maintenance, adherence to best practices, and keeping up to date with the latest trends and techniques in cloud security (Ghazizadeh, Tamm, & Creutzburg, 2024).

Through a case study involving interviews and experiments, Alkhurayyif and Almarshdy (2024) found that affordable automated penetration testing tools can protect small organisations from cybersecurity threats. The penetration testing tools revealed that the organisation's website possessed several vulnerabilities. The Nessus tool detected at least 37 vulnerabilities on the web application. The ZAP testing tool indicated that the web application was facing critical failures, resulting in multiple vulnerabilities. The system was found to have three medium-risk, 12 low-risk, and four informational-risk vulnerabilities. By evaluating open ports, the NMAP tool uncovered various vulnerabilities. These results hold significant importance for small organisations. Firstly, automated penetration testing tools can be easily utilised by small organisations to enhance their cybersecurity without the need for expensive expert assistance. Secondly, based on these findings, it is advisable to use automated penetration testing tools in various combinations, as different tools offer unique benefits to cybersecurity.

This study highlighted the limitations of relying solely on one scanning tool by using evidence from penetration testing methods, tools, and OWASP risk methodologies, as shown by the varying results obtained from different techniques and tools. The most successful approach for detecting and addressing web application vulnerabilities is to employ a thorough testing strategy that integrates various types of vulnerability scanners and techniques. These issues become particularly clear when using grey box testing techniques alongside both manual and automated scanning tools like Acunetix, Invicti, Burp Suite Professional, and OWASP ZAP, which assess factors such as vulnerability coverage, scanning speed, vulnerability detection, and false positive rates. By implementing the described method, the security community can gather trustworthy information to aid in making educated choices when selecting penetration testing techniques and tools to effectively safeguard information in websites and applications. According to these findings, a suggested approach is a combination of manual testing and automated scanning due to its high effectiveness (Echefunna, et al., 2024).

A groundbreaking architecture utilises the powerful features of the Metasploit Framework and OWASP ZAP, enabling organisations to proactively detect and address vulnerabilities in their web applications. The Metasploit Framework (a deliberately vulnerable virtual machine) provides a controlled setting for mimicking actual cyberattacks. Its wide array of vulnerabilities, ranging from basic misconfigurations to intricate exploits, creates an optimal environment for evaluating the strength of web applications. OWASP ZAP (a prominent open-source security tool) enhances this framework with its extensive set of scanning and testing capabilities. By automating the vulnerability detection and analysis process, OWASP ZAP simplifies the workflow of penetration testing, facilitating the efficient identification of potential threats (Samgir, Gutte, Kolhe, & Patil, 2024).

Journal of Information Systems Engineering and Management
2025, 10(4)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

A model proposed by Alhogail and Alkahtani (2024) consisted of a collection of information followed by vulnerability assessment to test and exploit. The results are generated automatically as a report. A tool, Kashef model, was also developed to examine the accuracy and effectiveness of the model. Tests showed the proposed model as an effective tool to identify vulnerabilities of the application.

A comparison study of two emerging tool types, Interactive Application Security Testing (IAST) and Runtime Application Self-Protection (RASP), with well-established tools like Dynamic Application Security Testing (DAST) and Static Application Security Testing (SAST), showed that IAST performed relatively well compared to other tools, performing second-best in both efficiency and effectiveness. IAST detected eight Top-10 OWASP security risks compared to nine by SMPT and seven for EMPT, DAST, and SAST. IAST found more vulnerabilities than SMPT. The efficiency of IAST (2.14 VpH) is second to only EMPT (2.22 VpH). These findings imply that our study benefited from using IAST when conducting black-box security testing. RASP prevented only Injection attacks in Open MRS. Thus, in the context of a large, enterprise-scale web application such as Open MRS, RASP does not replace vulnerability detection, while IAST is a powerful tool that complements other techniques (Seth, Bhattacharya, Elder, Zahan, & Williams, 2025).

A non-intrusive systematic review of 81 papers by Adeniran, et al. (2024) on various knowledge-based authentication techniques, vulnerabilities, and attacks, stressed the significance of data, the effects of vulnerabilities over data, and the tools used in detection and prevention. It also attempts to create awareness of the importance of adopting the latest security measures to be protected from attacks. The authors used many charts to describe the categorisation of knowledge-based authentication (KBA) methods, their applications, their historical progression, key performance criteria, vulnerability categories and the pros and cons of each.

Chaturvedi, Lakhani, Agarwal, Moharir, and Kumar AR (2024) examined the capabilities of incorporating OpenVAS, Wireshark, Nmap, and Metasploit for a thorough evaluation and analysis of vulnerabilities in IT environments as well as in both public and private sectors. OpenVAS acts as a powerful platform for vulnerability scanning, while Wireshark analyses network traffic for possible threats. Nmap detects open ports and associated vulnerabilities, and Metasploit enables ethical hacking and penetration testing. Collectively, these tools empower organisations to proactively identify and remediate security weaknesses, thereby enhancing defences against cyber threats. The paper recommends regular integration within a cohesive vulnerability management framework, fostering a proactive and efficient strategy for protecting against cybersecurity issues.

The above review of papers shows that the efficacies of automated penetration testing tools can widely differ and depend on the comparison methods and the penetration test platforms. The methods used in this study were similar to many of these papers.

Methodology

The study aimed to evaluate the efficacy of automated penetration testing tools in identifying vulnerabilities in modern web applications. The methodology was structured into several key phases: tool selection, test environment setup, vulnerability testing, and result analysis. Each phase involved specific algorithms and systematic approaches to ensure a comprehensive evaluation.

1. Tool Selection

- Criteria Definition: We defined criteria for selecting automated penetration testing tools, focusing on popularity, ease of use, comprehensiveness, and support for current web technologies.
- Tool Selection: Five tools, Tool A, Tool B, Tool C, Tool D, and Tool E, were selected for analysis based on the criteria. These tools are:
    - Tool A: OWASP ZAP
    - Tool B: Burp Suite Community Edition
    - Tool C: Acunetix Free Edition
    - Tool D: Netsparker Community Edition
    - Tool E: Arachni

Journal of Information Systems Engineering and Management
2025, 10(4)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

First, the comparative advantages and disadvantages of automated and manual tools need to be evaluated when considering the selection of automated tools for web vulnerability testing (Singh, Meherhomji, & Chandavarkar, 2020). To reduce false positives, Awang and Manaf (2013) suggested a framework consisting of an automatic Blackbox testing followed by a manual method. The combined method detected five different vulnerabilities. For their experiments, Abdulghaffar, Elmrabit, and Yousefi (2023) selected the latest versions of OWASP (Arachni and Zap) as the automated targeting software. They tested a union list and an intersection list of these two tools. Union list performed best for true positives, False positives were the highest for OWASP ZAP, but closely followed by Union list with the highest precision and recall and the lowest false negatives. The reputation of tools was the basis of tools selection of the two tools (Burp Suite and OWASP ZAP) in the studies of Shah (2020). False positive rates were 0% and 0.02% for them, respectively. Alkhurayyif and Almarshdy (2024) found that some tools are very effective and affordable in identifying vulnerabilities of small business web applications. Therefore, the selection of tools for testing web vulnerabilities needs to be based on effectiveness and affordability.

All the factors identified in the above papers were used for the selection of the five tools of this research. The criteria listed above demonstrate this.

2. Test Environment Setup

- Web Application Selection: We selected five modern web applications with diverse tech stacks, including the following:
  - DVWA (Damn Vulnerable Web App) – Classic PHP/MySQL stack for basic vulnerabilities.
  - NodeGoat – Node.js/Express-based purposely vulnerable app.
  - Juice Shop – An intentionally insecure Angular-based web application.
  - Vulnerable Django Application – Created in-house based on public open-source templates for Django.
  - ReactGoat – A custom-built React-based vulnerable app following guidelines from the OWASP Benchmark project.
- Mock Vulnerabilities: Using algorithmic approaches, specific known vulnerabilities were embedded into these applications, ensuring they represented real-world scenarios like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).

3. Vulnerability Testing

- Automated Scanning: Each selected tool was used to perform automated scanning on all five applications. Each scan was run thrice to ensure consistency, following standard operation procedures recommended by tool developers, including:
  - Using default or recommended scan profiles targeting web application vulnerabilities (SQLi, XSS, CSRF).
  - Ensuring all scans were unauthenticated (black-box), except where a tool required authentication setup for some tests.
  - Running each scan three times to account for tool variability/scan randomness and average the outcomes.
  - Resetting application state between scans to ensure tests were reproducible.
- Data Collection: Using algorithms, we systematically collected data on the number and type of vulnerabilities identified, scan duration, and false-positive rates.

4. Result Analysis

- Performance Metrics: We utilised several performance metrics, such as detection rate, precision, recall, and scan time efficiency.
- Data Analysis Algorithms: Statistical algorithms, including logistic regression and ANOVA, were used to analyse the collected data and determine the statistical significance of the results.

The pseudocode used for implementing the methodology for this work is shown below.

Journal of Information Systems Engineering and Management
2025, 10(4)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

```
BEGIN
1. Initialise Tool Performance Data Storage
  SET metrics for detection_rate[], precision[], recall[], scan_time[], false_positive_rate[], false_negative_rate[]
2. Tool Selection
  FOR each tool IN [Tool A, Tool B, Tool C, Tool D, Tool E]
    SELECT based on criteria: ["Popularity", "Ease of Use", "Comprehensiveness", "Compatibility"]
3. Test Environment Setup
  SELECT 5 diverse web applications
  FOR each application IN selected applications
    INJECT vulnerabilities: [SQL Injection, XSS, CSRF]
4. Vulnerability Testing
  FOR each tool IN selected tools
    FOR each application IN selected applications
      PERFORM scan 3 times
      RECORD: vulnerabilities_detected, scan_duration, false_positives, false_negatives
      CALCULATE: average_detections, precision, recall
5. Data Collection and Analysis
  STORE: detection_rate[], precision[], false_positives[], false_negatives[]
  CALCULATE metrics: recall[], scan_time[]
  PERFORM statistical analysis (logistic regression, ANOVA) FOR each metric
6. Evaluate Performance
  IDENTIFY the best tool with the highest performance based on all metrics
7. Conclusions
  PRINT results emphasizing the importance of combining automated and manual testing
END
```

The pseudocode begins with the initialisation phase, where data structures are established to store the performance metrics for each of the automated penetration testing tools under consideration. This step is crucial to organise and facilitate the subsequent data collection and analysis processes. Following initialisation, the tool selection and environment setup phase involves the criteria-based selection of tools, ensuring that the chosen tools align with pre-defined standards of popularity, ease of use, comprehensiveness, and compatibility with current web technologies. This phase also includes the embedding of known vulnerabilities into selected web applications, providing a representative test bed for evaluating the tools' capabilities.

During the testing and data collection phase, each selected tool undergoes multiple scans across various web applications. The objective here is to meticulously record the number of detected vulnerabilities, the time taken for each scan, and the incidence of false positives. This detailed data collection is essential for the generation of performance metrics, which include calculating the detection rate, precision, recall, and scan time efficiency. These metrics provide a quantitative measure of each tool's efficacy.

Journal of Information Systems Engineering and Management
2025, 10(4)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

In the statistical analysis phase, sophisticated statistical methods, such as logistic regression and ANOVA, are employed to scrutinise the collected metrics and establish their statistical significance. This analysis is vital for providing reliable insights into each tool's performance. Performance evaluation is then conducted to identify the tool with the highest efficacy across all metrics, with particular emphasis on tools that demonstrate superior performance, such as Tool D in this study.

The conclusions drawn from the analysis stress the importance of manual verification alongside automated testing due to the potential for false positives. The study advocates for an integrated approach, combining both automated and manual penetration testing, to achieve the most thorough security evaluation. Overall, this pseudocode outlines a systematic strategy for assessing the efficiency of automated penetration testing tools through a combination of statistical analysis and comprehensive performance metrics.

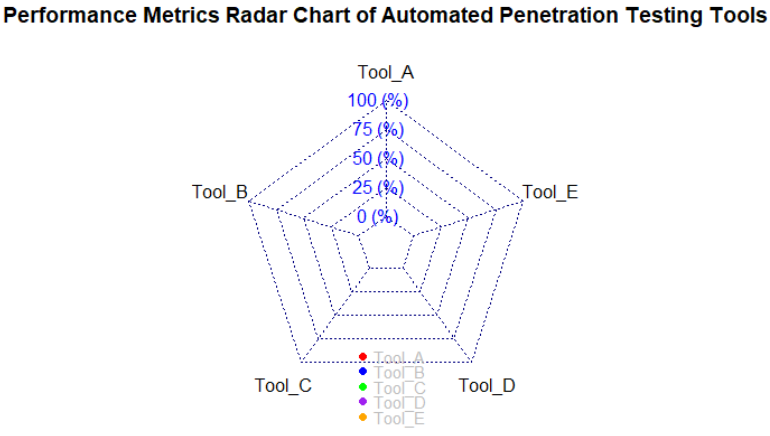
Results

The results section of this study provides a comprehensive analysis of the performance of various automated penetration testing tools employed to identify vulnerabilities in modern web applications. This section begins by presenting the performance metrics obtained from a series of thorough and systematic evaluations of the selected tools, including Tool A, Tool B, Tool C, Tool D, and Tool E. These metrics—detection rate, precision, recall, scan time efficiency, and false positive rate—were carefully calculated and compiled into a detailed table for ease of comparison. The evaluation results demonstrated varying effectiveness across different tools. The performance metrics are detailed below (Table 1 and Figures 1 and 2), summarised for each tool across all tested applications.

Table 1
Performance metrics

Metric	Tool A	Tool B	Tool C	Tool D	Tool E
Detection Rate	85%	73%	78%	90%	67%
Precision	82%	75%	70%	88%	64%
Recall	84%	70%	76%	92%	62%
Scan Time Efficiency (min per scan)	25	30	40	20	35
False Positive Rate	15%	20%	30%	10%	25%
False Negative Rate	5%	10%	15%	4%	12%

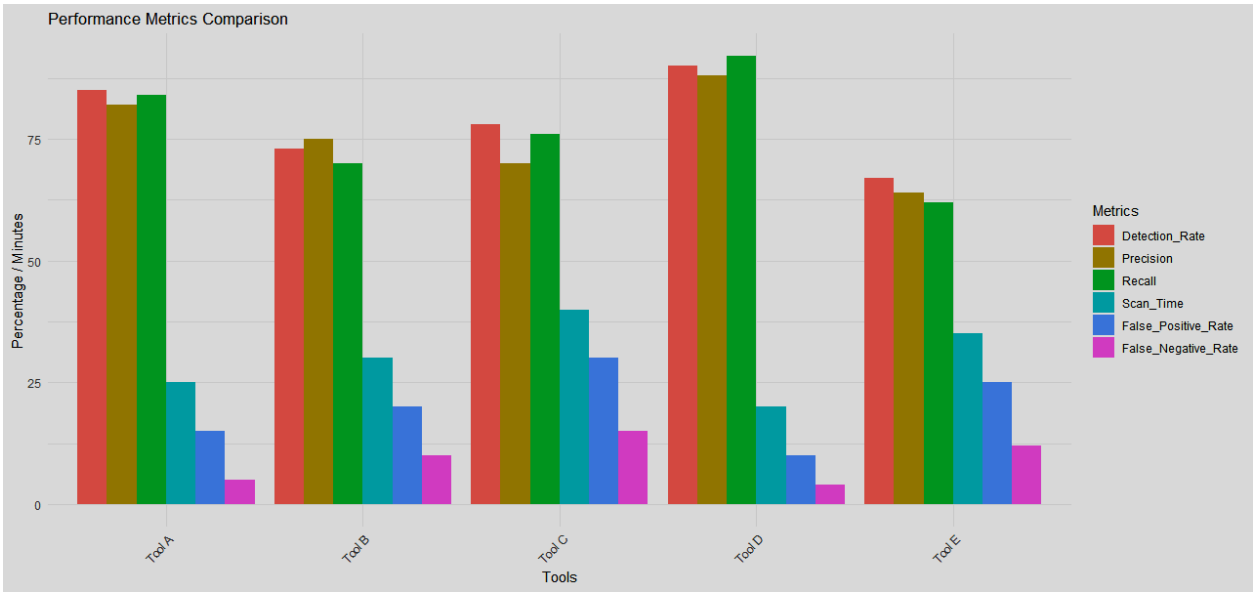
Figure 1
Performance metrics radar chart



Journal of Information Systems Engineering and Management
2025, 10(4)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Figure 2
Performance metrics comparison



The results on performance metrics highlight the varying levels of effectiveness exhibited by each tool. Tool D emerged as the most effective, leading in terms of detection rate (90%), precision (88%), and recall (92%), and demonstrating the most efficient scan time at 20 minutes per scan. Tool D also has the lowest false positive (10%) and false negative rates (4%), with the values ranging from 15% to 30% for the other four tools. Tool E showed the least effective performance, with lower metrics across all categories, including a detection rate of 67%, precision of 64%, recall of 62%, and a high false positive rate of 25%. These measurements stressed Tool D's reliability and its statistically significant superior capability ($p < 0.05$) in identifying vulnerabilities, making it a clear standout among the tested tools. The range of these metrics for the other three tools was between these two extremes. (Figure 1 and 2).

However, the study also underlines an important caveat: while automated penetration testing tools like the presence of false positives remains a critical issue, underscoring the continued necessity for manual verification methods. The highest was for Tool C with 30% false positives. Equally, the false negative results of Tool C was the highest with 15%. The relatively significant presence of false negatives for all the five tools show that although automation can significantly aid in the efficiency and breadth of testing, it cannot fully replace the nuanced understanding provided by skilled human testers.

Discussion and Conclusion

The findings obtained in this study on the inclusion of manual testing and the use of multiple tools for better efficacy were supported by (Kollepalli, Natarajan, Mathi, and Ramalingam (2024). In a review of SQL injection attacks, Hajar, Jaafar, and Rahim (2024) recommended the same procedure followed in this study. Identifying and exploiting vulnerabilities in a controlled, safer environment is vital. A detailed analysis of SQL injection attacks and types, as well as possible solutions discussed by Sharma and Sharma (2016), agrees with many aspects of this study. Some of the papers reviewed and discussed by Wimukthi, Kottegoda, Andaraweera, and Palihena (2022) are similar to this study (example, Li et al., 2019; Kumar & Sujatha, 2022; Rankothge et al., 2020) in their methodological approaches. A systematic review by Lawal, Sultan, and Shakiru (2016) included the automated detection systems and found these methods useful for the efficient detection of SQL attacks, based on which preventive steps can be planned. The review does not contain current examples of tools, as it was published about a decade ago. In a survey of SQL injection attack countermeasures, Ladole and Phalke (2015) discussed the method followed in this study related to testing the efficacy of automated penetration testing tools. Thus, this review supports the methodology used in this study. The intrusion detection system used in some studies mentioned by Abdullayev and Chauhan

(2023) also supports many aspects of this study with similar observations like the efficacy advantage and false positive disadvantage.

Using a genetic fuzzy rule-based classification system (FRBCS) for SQLI detection, prioritising the accuracy, learning and flexibility of the obtained rules, a top-down software design approach, a web-based application software was written in C# programming language that runs on MySQL as the backend database. This was evaluated using many well-known malicious data sets. The tool restored security in our web-based transactions, assuring confidence, transparency, integrity, and privacy in our transactions (Agbakwuru & Njoku, 2021). Although the concept and the methods used in this paper were similar to those used in this study, performance metrics were not provided.

A different method of web penetration test in SQL injection attacks was used by Alanda, Satria, Ardhana, Dahlan, and Mooduto (2021). The method consisted of penetration testing using the black-box method to test web application security based on the list of most attacks on the Open Web Application Security Project (OWASP), namely SQL Injection. Ten websites were randomly tested using this method, and high levels of vulnerabilities to SQL injection attacks were detected.

Thus, the concept, methodology and results of this study are generally supported by the literature. A possible future research is using genetic fuzzy algorithms. There is also scope for research combining detection and prevention into single studies.

Ultimately, the results suggest that the most robust approach for evaluating the security of web applications involves integrating both automated and manual penetration testing strategies. By combining the strength of automated tools in rapidly scanning and identifying potential vulnerabilities and the insight of manual analysis to verify and investigate the context and impact of these findings, organisations can ensure a more comprehensive security posture. The implications of these findings are pivotal for cybersecurity strategies, encouraging a balanced and holistic approach to vulnerability assessment.

References

[1] Abdullayev, V., & Chauhan, A. S. (2023). SQL injection attack: Quick view. *Mesopotamian journal of Cybersecurity*, 30-34. doi:https://doi.org/10.58496/MJCS/2023/006,

[2] Adeniran, T. C., Jimoh, R. G., Abah, E. U., Faruk, N., Alozie, E., & Imoize, A. L. (2024). Vulnerability assessment studies of existing knowledge-based authentication systems: a systematic review. *Sule Lamido University Journal of Science & Technology, 8*(1), 34-61. doi:https://doi.org/10.56471/slujst.v7i.485

[3] Agbakwuru, A. O., & Njoku, D. O. (2021). SQL Injection Attack on Web Base Application: Vulnerability Assessments and Detection Technique. *International Research Journal of Engineering and Technology,*, 243-252. Retrieved March 12, 2025, from https://d1wqtxts1xzle7.cloudfront.net/105830034/IRJET-V8I345-libre.pdf?1695190941=&response-content-disposition=inline%3B+filename%3DSQL_Injection_Attack_on_Web_Base_Applica.pdf&Expires=1741747452&Signature=VqS-vNFXvL3yZEGDGGcZ06WNZtP30HwQ9RD0bgYNHlB5Gifj

[4] Alanda, A., Satria, D., Ardhana, M. I., Dahlan, A. A., & Mooduto, H. A. (2021). Web application penetration testing using SQL Injection attack. *JOIV: International Journal on Informatics Visualization, 5*(3), 320-326. doi:https://dx.doi.org/10.30630/joiv.5.3.470

[5] Alhogail, A., & Alkahtani, M. (2024). Automated extension-based penetration testing for web vulnerabilities. *Procedia Computer Science, 238*, 15-23. doi:https://doi.org/10.1016/j.procs.2024.05.191

[6] Alkhurayyif, Y., & Almarshdy, Y. S. (2024). Adopting automated penetration testing tools: A cost-effective approach to enhancing cybersecurity in small organisations. *Journal of Information Security and Cybercrimes Research, 7*(1), 51-66. doi:https://doi.org/10.26735/RJJT2453

[7] Chaturvedi, A., Lakhani, B., Agarwal, T., Moharir, M., & Kumar AR, A. (2024). A Comprehensive Vulnerability Tools Analysis for Security and Control in IT Environment and Organizations. *5th International Conference on Electronics and Sustainable Communication Systems (ICESC), 7-9 August 2024, Coimbatore, India* (pp. 612-618). IEEE. doi:https://doi.org/10.1109/ICESC60852.2024.10689860

[8] Echefunna, C. C., Osamor, J., Iwendi, C., Owoh, P., Ashawa, M., & Philip, A. (2024). Evaluation of Information Security in Web Application Through Penetration Testing Techniques Using OWASP Risk Methodology. *International Conference on Advances in Computing Research on Science Engineering and Technology (ACROSET), Indore, India, 27 September 2024* (pp. 1-21). IEEE. doi:https://doi.org/10.1109/ACROSET62108.2024.10743903

[9] Ghazizadeh, H., Tamm, G., & Creutzburg, R. (2024). Automated Tools for Cloud Security Testing. *Electronic Imaging, 36*, 1-7. doi:https://doi.org/10.2352/EI.2024.36.3.MOBMU-319

[10] Hajar, S., Jaafar, A. G., & Rahim, F. A. (2024). A Review of Penetration Testing Process For Sql Injection Attack. *Open International Journal of Informatics, 12*(1), 72-87. doi:https://doi.org/10.11113/oiji2023.11n2.256

[11] Kollepalli, R. P., Natarajan, A., Mathi, S., & Ramalingam, V. (2024). An Experimental Study on Detecting and Mitigating Vulnerabilities in Web Application. *International Journal of Safety & Security Engineering, 14*(2), 523-532. doi:ttps://doi.org/10.18280/ijsse.140219

[12] Ladole, A., & Phalke, D. A. (2015). A Survey on SQL Injection Attack Countermeasures Techniques. *International Journal of Science and Research, 4*(11), 1556-1566. Retrieved March 11, 2025, from https://d1wqtxts1xzle7.cloudfront.net/81549393/ae59eade974103be04ddda8ee29fafea6c71-libre.pdf?1646199951=&response-content-disposition=inline%3B+filename%3DA_Survey_on_SQL_Injection_Attack_Counter.pdf&Expires=1741709512&Signature=LL9j4aO7tiXXBab-Da2LRVnbe

[13] Lawal, M. A., Sultan, A. B., & Shakiru, A. O. (2016). Systematic literature review on SQL injection attack. *International Journal of Soft Computing, 11*(1), 26-35. Retrieved March 12, 2025, from https://www.researchgate.net/profile/Lawal-Muhammad-Aminu/publication/282377809_Systematic_literature_review_on_SQL_injection_attack/links/574191fc08ae9f741b36701c/Systematic-literature-review-on-SQL-injection-attack.pdf#page=1.00

[14] Moreira, D., Seara, J. P., Pavia, J. P., & Serrão, C. (2024). Intelligent platform for automating vulnerability detection in web applications. *Electronics, 14*(1), 79. doi:https://doi.org/10.3390/electronics14010079

[15] Samgir, A. B., Gutte, V., Kolhe, K., & Patil, D. R. (2024). Automated Penetration Testing Architecture Using Metasploit and OWASP ZAP for Web Applications. *2nd International Conference on Sustainable Computing and Smart Systems (ICSCSS), 10-12 July 2024, Coimbatore, India* (pp. 649-657). IEEE. doi:https://doi.org/10.1109/ICSCSS60660.2024.10625033

[16] Seth, A., Bhattacharya, S., Elder, S., Zahan, N., & Williams, L. (2025). Comparing effectiveness and efficiency of interactive application security testing (IAST) and runtime application self-protection (RASP) tools in a large java-based system. *Empirical Software Engineering, 30*(3), 67. doi:https://doi.org/10.1007/s10664-025-10621-5

[17] Sharma, C. J., & Sharma, A. K. (2016). Explorative study of SQL injection attacks and mechanisms to secure web application database-A. *International Journal of Advanced Computer Science and Applications, 7*(3), 79-87. doi:https//doi.org/10.14569/IJACSA.2016.070312

[18] Wimukthi, H. Y., Kottegoda, H., Andaraweera, D., & Palihena, P. (2022). A comprehensive review of methods for SQL injection attack detection and prevention. *International Journal of Scientific Research in Science and Technology*, 11 pp. Retrieved March 12, 2025, from https://www.researchgate.net/profile/Yasith-Wimukthi-Hr/publication/364935556_A_comprehensive_review_of_methods_for_SQL_injection_attack_detection_and_prevention/links/635f7f2c6e0d367d91e115d5/A-comprehensive-review-of-methods-for-SQL-injection-attack-det

[19] Yadav, N. S., Rounak, R., & Sharma, P. C. (2024). Web-based Vulnerability Analysis and Detection. *International Journal of Sensors, Wireless Communications and Control.* doi:https://doi.org/10.2174/0122103279319619241008221647