Borneo International Journal eISSN 2636-9826; Vol.8 (2); 2025; 34-39
Published by *Majmuah Enterprise*
www.majmuah.com

The Effectiveness of Automated Ethical Hacking Tools in Identifying Web Application Vulnerabilities

TAN XIAO QIN, MOHAMAD FADLI ZOLKIPLI
School of Computing, Universiti Malaysia Utara (UUM),
06010 Sintok, Kedah, MALAYSIA
Email: tan_xiao@ahsgs.uum.edu.my, m.fadli.zolkipli@uum.edu.my | Tel: +60135909362, 049285058

Received: June 20, 2025
Accepted: June 24, 2025
Online Published: June 27, 2025

Abstract

The increasing complexity and importance of web applications have made them main targets of cyberattacks. Ethical hacking plays a crucial role in identifying and fixing vulnerabilities before they can be exploited by attackers. Specifically, automated ethical hacking tools have gained widespread use due to their speed and effectiveness in identifying vulnerabilities. This study reviews the most studied tools in previous research and evaluates their effectiveness based on reported detection results. Using a literature-based approach, this paper analyzes the effectiveness of automated ethical hacking tools in identifying web applications vulnerabilities. By reviewing and synthesizing findings from previous research, the study examines each tool's ability to detect common web applications vulnerabilities. The discussion highlights pros such as flexibility of use and high detection accuracy for common vulnerabilities, as well as cons like high false positives rates and limited on detecting complex or logic-based vulnerabilities. According to the results, automated tools are beneficial for basic evaluations, but they work better when combined with manual testing. This study offers a deeper understanding of the role automated tools play in modern cybersecurity practices and provide guidance for enhancing vulnerability assessment methods.

Keywords: automated tools; ethical hacking; web application vulnerabilities; penetration testing

1. Introduction

In an increasingly digital world, cybersecurity plays a vital role in protecting individuals and organizations from the risks of hacking. While hacking is generally considered a negative activity due to its association with unauthorized access or control, it also has a legitimate role in cyber and information security. This authorized practice of hacking is known as ethical hacking or white-hat hacking (Yaacoub et al., 2021). Ethical hacking helps to identify and fix vulnerabilities in systems or networks before being exploited by malicious attackers. Ethical hackers use the same techniques and tools as malicious hackers but to improve security rather than compromise it. As technologies become more sophisticated, automated tools assist in identifying vulnerabilities and simulating attacks that malicious hackers might attempt. By automating tasks that would otherwise be performed manually, these tools save time, reduce resource consumption, and enhance overall efficiency and effectiveness. While manual testing remains essential, especially when spotting complex and specific vulnerabilities, it is often time consuming, costly and prone to human error. Additionally, it is not suitable for evaluating large or rapidly evolving web applications due to its lack of scalability (Divyaniyadav et al., 2018). Since, automated ethical hacking tools are now a crucial part of proactive cybersecurity practices, particularly for organizations managing large-scale or dynamic web environments.

Web applications are becoming increasingly essential in nearly every area, including e-commerce, social networking, project management, and healthcare. However, many web applications suffer from vulnerabilities or misconfigurations and often store sensitive data like personal information, login credentials, financial details. The 2024 Vulnerability Statistics Report states that a large percentage of security issues that reported were related to web applications, with vulnerabilities such as SQL injection, cross-site scripting (XSS) and authentication vulnerabilities being among the most detected. As a result, web applications are attractive targets for attackers to gain unauthorized access, disrupt services, or steal data (Alaoui & Nfaoui, 2022). Therefore, ensuring robust security for web applications is essential, not only to protect sensitive data, but also to preserve organizational reputation, meet legal and regulatory obligations, ensure business continuity, and maintain user trust. This paper aims to investigate whether automated ethical hacking tools can effectively identify vulnerabilities in web applications by analyzing and synthesizing findings from existing research. By comparing the strengths, limitations, and detection capabilities of these tools, the study contributes to a deeper understanding of their role in modern cybersecurity practices. The paper is organized as follows: the next

section presents a review of relevant literature, followed by a discussion of key findings, and concludes with a summary and recommendations for future research.

2. Literature Review

2.1 Web application vulnerabilities

According to the Edgescan (2024 Vulnerability Statistics Report 9th Edition, 2024), SQL Injection (SQLi) accounted for 19.47% of critical web application vulnerabilities, making it the most common. This was followed by cross-site scripting (XSS) attacks – both stored and reflected – at 10.5%, and malicious file upload vulnerabilities at 7.25%. SQL Injection is a security vulnerability that occurs when an attacker inserts a malicious SQL query into an application input field, allowing unauthorized access to sensitive data in the database. The root cause of SQLi lies in the construction of dynamic SQL queries without the use of parameterized queries or proper input validation. Attackers can manipulate these inputs to change the structure of the query, enabling actions such as authentication bypasses, data modification or deletion, and execution of a remote command (Hasan et al., 2022). Despite much study on detection and prevention, SQL injection attacks remain common due to the continued lack of proper input validation and secure coding practices, particularly in legacy systems or rapidly developed applications (Wimukthi et al., 2022).

Cross-Site Scripting (XSS) is another common and dangerous vulnerability found in many web applications, where attackers inject malicious scripts into websites, which are then executed by the browser when other users visit the website. There are two primary forms of XSS: stored and reflected (Alsaffar et al., 2022). In a stored XSS attack, the malicious script is permanently saved on the server, such as in a comment section or user profile, and is executed automatically when another user visits the page. In contrast, reflected XSS occurs when the malicious script is included in a URL, input field or cookie and is immediately reflected in the server's response. This often requires the user to click on a malicious link (Rodríguez et al., 2020). When successful, XSS attacks can result in the theft of session cookies, user impersonation, unauthorized actions on behalf of the user, and redirection to malicious websites. Malicious file upload vulnerabilities occur when attackers upload files containing harmful content, such as scripts, executables, or backdoors, to a server. If successful, this can lead to remote code execution or unauthorized control over the server. Although not listed as a standalone item in the OWASP Top 10, malicious file upload is addressed under broader categories such as Security Misconfiguration and Broken Access Control. Security misconfiguration refers to improperly maintained system settings that can be exploited by attackers, while broken access control allows unauthorized users to gain access to restricted resources due to insufficient access restrictions (Abdulghaffar et al., 2023).

2.2 Automated ethical hacking tool

In the current digital landscape, web applications are essential to business operations, public services, and daily communication. However, this increasing dependence has also made them prime targets for cyberattacks. As a result, web application security has become a critical concern, and ethical hacking has emerged as a key approach for proactively identifying and addressing security vulnerabilities. A wide range of automated tools exist for web applications penetration testing, each offering varying levels of effectiveness and efficiency in detecting vulnerabilities. Albahar et al. (2022) conducted simulation tests to evaluate the effectiveness of both commercial and non-commercial web application security tools. In the non-commercial category, the tools compared were Arachni, Wapiti3, and OWASP ZAP, while the commercial tools included Qualys WAS, Fortify WebInspect, and Burp Suite Professional. The results showed that Burp Suite Professional had the strongest performance among commercial tools, while OWASP ZAP was the most effective among non-commercial (open-source) tools. Both tools also demonstrated strong crawling capabilities, which are essential for thoroughly scanning web applications. Using multiple vulnerability scanners is a recommended approach to improve the accuracy of vulnerability detection in web applications (Abdulghaffar et al., 2023). The study showed that combining the results from multiple tools produced more accurate outcomes compared to relying on a single scanner, such as Arachni or OWASP ZAP. Although both tools identified similar vulnerabilities, OWASP ZAP detected most of the true positives and contributed significantly to the combined results. In contrast, Arachni missed several true positives, resulting in lower performance. As a result, OWASP ZAP outperformed Arachni in terms of recall and F-measure, demonstrating its greater effectiveness in identifying vulnerabilities.

(Shahid et al., 2022) evaluated eleven ethical hacking tools by scanning intentionally vulnerable applications such as DVWA. The selected tools were accessed based on criteria such as detection rate, precision, vulnerability coverage, and severity level. The study found that OWASP ZAP had a higher detection rate than other tools, while Acunetix and NetSparker, both commercial scanners, demonstrated fewer false positives, indicating better accuracy. To improve

open-source tools further, the researchers proposed developing custom plugins and algorithms to enhance crawling capabilities, authentication support, and reducing false positives.

OWASP ZAP was effectively used to identify vulnerabilities in both test environment and a university website (Ivanov & Atanasova, 2022). Cross-Site Scripting (XSS), external redirects, path traversal, Remote File Inclusion (RFI) and OS command injection are among the significant vulnerabilities it discovered. The severity of each vulnerability was clearly marked, assisting administrators in identifying and resolving the most critical problems first.

Devi & Kumar (2020) stated that Nikto tool detected a greater number of vulnerabilities compared to OWASP ZAP when both tools were applied across 10 different application domains to identify security weaknesses. According to the results of the testing, OWASP ZAP successfully identified medium and low-level vulnerabilities across all 10 domains, whereas Nikto only detected vulnerabilities in a few of those domains. However, when comparing results within the same domain, Nikto was able to identify additional vulnerabilities that OWASP ZAP failed to detect.

Althunayyan et al. (2022) tested the effectiveness of black-box web scanners in detecting SQL injection (SQLi), NoSQL, and server-side template injection (SSTI) vulnerabilities. Precision, recall, and F-measure were used to evaluate the vulnerability detection rates of the scanners, with OWASP ZAP and Burp Suite Professional outperforming the others. The researchers stated that their limited ability to crawl dynamic web applications and their incomplete support for detecting all types of injection vulnerabilities.

Kiran Gandikota et al. (2023) compared several vulnerability scanning tools using real data, including Nessus (on Windows), Burp Suite Professional, and Wapiti (on Kali Linux). The study used the OWASP framework as a benchmark to evaluate the tools. The result shows that Burp Suite Professional performed the best, as it could detect various types of web application vulnerabilities and works on both Windows and Linux. Each tool had its pros and cons: Nessus was easy to use but could not scan ".org" domains, while Wapiti was only available on Kali Linux, making it less accessible.

Varghese & Kurian (2021) reported that Uniscan is outperform Nikto and Grabber by performing more in-depth analysis, including checks for SQL injection, which make it more effective for database related vulnerabilities. Although Grabber can detect the same vulnerabilities as Uniscan, it performs slower and provides less detailed results in more comprehensive security testing.

3. Discussion

This paper evaluates the effectiveness of automated ethical hacking tools based on previous studies. OWASP ZAP, Burp Suite, Wapiti, and Arachni are among the most evaluated tools due to their widespread use and reliability. Each tool has its own pros and cons across different aspects of vulnerability detection. For instance, OWASP ZAP is free of charge, open source, widely supported by the security community, and capable of automatically detecting common web application vulnerabilities (Carlos P. Flores Jr., 2024). Most of the research paper evaluated OWASP ZAP as the most effective automated tools due to * Burp Suite Professional is also frequently selected in research papers for its effectiveness in identifying web application vulnerabilities. It has demonstrated strong performance in detecting sophisticated input-based vulnerabilities, such as cross-site scripting (XSS) and SQL injection (Althunayyan et al., 2022) (Alazmi & De Leon, 2022). This is mainly because it provides full control to users by combining automated and manual testing capabilities to efficiently discover and exploit vulnerabilities. However, unlike OWASP ZAP, Burp Suite requires a commercial license for full functionality, which may limit accessibility for some users. Tools like Wapiti and Arachni have also been examined in several studies, but their results are typically based on the scope and configuration of the testing environment. Wapiti is known for its ability to detect a range of common web applications vulnerabilities, including SQL injections, cross-site scripting (XSS), and file disclosure. However, its effectiveness is limited by a simpler crawling process and a lack of flexibility compared to more advanced tools. Arachni, on the other hand, offers broader coverage and higher detection accuracy, especially for modern web applications. Nevertheless, its heavy resource usage and discontinued development reduces its performance and reliability compared to regularly updated tools.

Through the reviewed studies, one of the most common limitations is that automated tools often have significant false-positive and false-negative rates. Various automated tools commonly report vulnerable that do not exist, which reduces user confidence in the tools and makes manual validation necessary to filter actual vulnerabilities (Nnaemeka & Ehichoya, 2022). Automated tools produce false positives, and even with improved detection logic, they still struggle to fully eliminate inaccurate results (Rennhard et al., 2022). This shows the ongoing challenge of improving detection accuracy and highlights the importance of manual review by ethical hackers or penetration testers to validate automated

results (Qadir et al., 2025). Those tools often lack context awareness, which makes it difficult for them to understand dynamically generated content, user-specific responsibilities or business logic (Shahid et al., 2022). Automated tools are generally most effective in identifying common web applications vulnerabilities but fail to identify more complex vulnerabilities like chained exploits or logical errors. At this point, manual testing becomes crucial because it enables ethical hackers to interpret web applications specifically, conduct custom workflows, and find complex vulnerabilities that cannot be identified by automated tools. To achieve the most comprehensive vulnerability assessment, the reviewed literature continuously recommends a combination of automated tools with manual testing.

Table 1: Strength and Weakness of Automated Ethical Hacking Tools from Previous Paper

Tool	Strength	Weakness
Acunetix	⁾ High detection accuracy ⁾ Support for modern web technologies	⁾ Commercial tool with high cost ⁾ Heavy resources usage
Arachni	⁾ High detection accuracy ⁾ Have good support for modern web applications ⁾ Provide detailed reports	⁾ Discontinued development ⁾ Heavy resource usage ⁾ Slow performance for scanning
Burp Suite Pro	⁾ High accuracy and precision in both detection and exploitation ⁾ Well perform in SQLi, NoSQL, SSTI detection ⁾ Support for complex scan modes	⁾ Commercial tool which requires license ⁾ Resource-intensive application ⁾ Configuration and expertise are needed
Nessus	⁾ Well perform in network vulnerabilities detection ⁾ Able to detect both network and web application vulnerabilities	⁾ Designed for basic web application scanning, limited detection of complex web application vulnerabilities
Nikto	⁾ Fast performance for server misconfigurations and outdated components ⁾ Suitable used in surface-level scanning ⁾ Lightweight and easy to deploy	⁾ No suitable for detecting complex web vulnerabilities ⁾ Higher false positive rate in detecting vulnerabilities of modern web applications
OWASP ZAP	⁾ Open-source and free ⁾ Actively maintained with regular updates ⁾ High detection rate, can detect a wide range of common vulnerabilities ⁾ Suitable for integration with CI/CD (Continuous Integration/Continuous Deployment) pipelines ⁾ Large community and documentation	⁾ Slow performance for deeper scanning ⁾ Possibility of producing false positives ⁾ Requires tuning for optimal accuracy
Paros	⁾ Lightweight and simple ⁾ Suitable for common SQL injection	⁾ Outdated and not maintained ⁾ Low detection rate, although for common vulnerabilities
Uniscan	⁾ Detect a wide range of vulnerabilities ⁾ Well perform in database testing ⁾ Multi-threaded crawler	⁾ Heavier resources usage compares with lighter tools ⁾ Slow performance for scanning
Wapiti	⁾ Open-source and free ⁾ CLI (Command-Line Interface) based ⁾ Suitable for common SQL injections	⁾ Lower detection rates, especially NoSQL and SSTI detection ⁾ Struggles with complex web structures and newer injection types ⁾ Show inaccurate results

4. Conclusions

According to the previous research studied automated ethical hacking tools are highly effective in detecting common vulnerabilities in web applications. Tools like OWASP ZAP and Burp Suite Pro regularly delivered good results due to their high detection accuracy and continuous improvement. They are highly beneficial for common vulnerabilities assessment and improve the efficiency of penetration testing processes. However, there are several limitations such as false positives, insufficient handling of authentication and failure to comprehend application specific logic. Although tools like Wapiti and Arachni contain scanning abilities, they are limited by issues like failure to support the latest web technologies also for Arachni is discontinued development. A hybrid testing approach is recommended to overcome these limitations because it offers deeper insights and makes up for the shortcomings of automation. Many studies limit the generalizability of their results by using audited benchmarks, unreliable techniques, or overly basic testing configurations. Future studies should focus on realistic testing scenarios, use standard evaluation frameworks, and stimulate the combination of automated and manual methods to improve the depth and accuracy of assessments.

Acknowledgments

The authors would like to express great appreciation to thank all members of the School of Computing who participated in the study. This study was carried out as part of the Hacking and Penetration Testing Project and was supported by Universiti Utara Malaysia.

References

2024 Vulnerability Statistics Report 9th Edition. (2024). www.first.org/cvss/

Abdulghaffar, K., Elmrabit, N., & Yousefi, M. (2023). Enhancing Web Application Security through Automated Penetration Testing with Multiple Vulnerability Scanners. *Computers*, *12*(11). https://doi.org/10.3390/computers12110235

Alaoui, R. L., & Nfaoui, E. H. (2022). Deep Learning for Vulnerability and Attack Detection on Web Applications: A Systematic Literature Review. In *Future Internet* (Vol. 14, Issue 4). MDPI. https://doi.org/10.3390/fi14040118

Alazmi, S., & De Leon, D. C. (2022). A Systematic Literature Review on the Characteristics and Effectiveness of Web Application Vulnerability Scanners. In *IEEE Access* (Vol. 10, pp. 33200–33219). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2022.3161522

Albahar, M., Alansari, D., & Jurcut, A. (2022). An Empirical Comparison of Pen-Testing Tools for Detecting Web App Vulnerabilities. *Electronics (Switzerland)*, *11*(19). https://doi.org/10.3390/electronics11192991

Alsaffar, M., Aljaloud, S., Mohammed, B. A., Al-Mekhlafi, Z. G., Almurayziq, T. S., Alshammari, G., & Alshammari, A. (2022). Detection of Web Cross-Site Scripting (XSS) Attacks. *Electronics (Switzerland)*, *11*(14). https://doi.org/10.3390/electronics11142212

Althunayyan, M., Saxena, N., Li, S., & Gope, P. (2022). Evaluation of Black-Box Web Application Security Scanners in Detecting Injection Vulnerabilities. *Electronics (Switzerland)*, *11*(13). https://doi.org/10.3390/electronics11132049

Carlos P. Flores Jr. (2024). Evaluation of Common Security Vulnerabilities of State Universities and Colleges Websites Based on OWASP. *Journal of Electrical Systems*, *20*(5s), 1396–1404. https://doi.org/10.52783/jes.2471

Devi, R. S., & Kumar, M. M. (2020). Testing for Security Weakness of Web Applications using Ethical Hacking. *Proceedings of the 4th International Conference on Trends in Electronics and Informatics, ICOEI 2020*, 354–361. https://doi.org/10.1109/ICOEI48184.2020.9143018

Divyaniyadav, Gupta, D., Singh, D., Kumar, D., & Sharma, U. (2018, December 1). Vulnerabilities and security of web applications. *2018 4th International Conference on Computing Communication and Automation, ICCCA 2018*. https://doi.org/10.1109/CCAA.2018.8777558

Hasan, M., Al-Maliki, A., & Jasim, N. (2022). Review of SQL injection attacks: Detection, to enhance the security of the website from client-side attacks. *Int. J. Nonlinear Anal. Appl*, *13*, 2008–6822. https://doi.org/10.22075/ijnaa.2022.6152

Ivanov, I., & Atanasova, M. (2022). *Network and Web Applications Vulnerability Testing Using Ethical Hacking*.

Kiran Gandikota, P. S. S., Valluri, D., Mundru, S. B., Yanala, G. K., & Sushaini, S. (2023). Web Application Security through Comprehensive Vulnerability Assessment. *Procedia Computer Science*, *230*, 168–182. https://doi.org/10.1016/j.procs.2023.12.072

Nnaemeka, C. C., & Ehichoya, O. (2022). *Evaluating security vulnerabilities in web-based Applications using Static Analysis*.

Qadir, S., Waheed, E., Khanum, A., & Jehan, S. (2025). Comparative evaluation of approaches & tools for effective security testing of Web applications. *PeerJ Computer Science*, *11*, 1–42. https://doi.org/10.7717/peerj-cs.2821

Rennhard, M., Kushnir, M., Favre, O., Esposito, D., & Zahnd, V. (2022). Automating the Detection of Access Control Vulnerabilities in Web Applications. *SN Computer Science*, *3*(5). https://doi.org/10.1007/s42979-022-01271-1

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

Shahid, J., Hameed, M. K., Javed, I. T., Qureshi, K. N., Ali, M., & Crespi, N. (2022). A Comparative Study of Web Application Security Parameters: Current Trends and Future Directions. *Applied Sciences (Switzerland)*, *12*(8). https://doi.org/10.3390/app12084077

Varghese, S., & Kurian, R. (2021). *Identifying Vulnerabilities in a Website Using Uniscan and Comparing Uniscan, Grabber, Nikto*. *3*(1). https://doi.org/10.5281/zenodo.5091326

Wimukthi, Y., Sri, H. R., Kottegoda, H., Andaraweera, D., & Palihena, P. (2022). *A comprehensive review of methods for SQL injection attack detection and prevention*. https://www.researchgate.net/publication/364935556

Yaacoub, J.-P. A., Noura, H. N., Salman, O., & Chehab, A. (2021). *A Survey on Ethical Hacking: Issues and Challenges*. http://arxiv.org/abs/2103.15072