INTERNATIONAL JOURNAL OF INFORMATION SECURITY

IAEME PUBLICATION

Plot: 03, Flat- S 1, Poomalai Santosh Pearls Apartment, Vaiko Salai 6th Street,
Jai Shankar Nagar, Palavakkam, Chennai - 600 041, Tamilnadu, India.

E-mail: editor@iaeme.com, iaemedu@gmail.com Website: www.iaeme.com Mobile: +91-9884798314

https://iaeme.com/Home/journal/IJIS

International Journal of Information Security (IJIS)
Volume 4, Issue 1, January-June 2025, pp. 98-110, Article ID: IJIS_04_01_005
Available online at https://iaeme.com/Home/issue/IJIS?Volume=4&Issue=1
ISSN Online: 2273-1307; Journal ID: 1230-2805
Impact Factor (2025): 8.50 (Based on Google Scholar Citation)
DOI: https://doi.org/10.34218/IJIS_04_01_005

© **IAEME** Publication

SCOPE DATABASE INDEXED

OPEN ACCESS

COMPARATIVE PERFORMANCE ANALYSIS OF WEB VULNERABILITY SCANNERS

Tenzin Yarphel
Lovely Professional University, Phagwara, Punjab, India.

ABSTRACT

As the reliance on web-based services increases, attackers increasingly target web applications. Therefore, Web Vulnerability Scanners (WVS) are necessary to identify vulnerabilities prior to exploitation. This study conducted a comparative analysis regarding several commonly available automated WVS, namely: OWASP ZAP, Nessus, Nikto, and Burp Suite. Each tool was compared, and objective criteria were established to analyze detection when comparing relevant precision and recall rates. A controlled test case was produced to quantify the vulnerability detection capability of the scan tools which used intentionally vulnerable web applications such as DVWA and a live host, to ensure consistency and repeatability. The research showed that all the scans can detect known and common web-based vulnerabilities such as SQL injection and Cross site Scripting (XSS) but proved to be substantial variations in the level of efficacy of each of the tools across many metrics. Burp Suite had the highest accuracy of the detection rates, while OWASP ZAP provided a sufficient middle ground between usability and scan scope. Overall, this study provided security practitioners with an examination of the application and failings of commonly used WVS tools, allowing practitioners to make decisions based off informed knowledge of the tools used in vulnerability assessment.

Keywords: web vulnerability scanner, web scanner, black-box testing, white-box testing, gray box testing.

Cite this Article: Tenzin Yarphel. (2025). Comparative Performance Analysis of Web Vulnerability Scanners. *International Journal of Information Security (IJIS)*, 4(1), 98-110.

https://iaeme.com/MasterAdmin/Journal_uploads/IJIS/VOLUME_4_ISSUE_1/IJIS_04_01_005.pdf

I. Introduction

In this digital age, a plethora of activities have moved to the web which must therefore be safeguarded from bad actors. Fortinet's 2022 Web Application Security Report indicated that a staggering 56% of participating organizations had experienced web application breaches or compromises in the previous 12 months. This marks a 6% turnaround from the previous year's survey where 50% of organizations experienced these same issues [1]. The trend clearly demonstrates the uptick across all industries in both sophisticated and frequent threats directed at web applications. In order to address the growing security threat, a group of volunteers established the Open Web Application Security Project (OWASP), which publishes the OWASP Top 10 list of the most critical web vulnerabilities on a regular basis. While many organizations use these reports to bolster their security posture, smaller organizations with limited breadth of knowledge struggled to interpret the recommendation and use it correctly. As a result, organizations have opted for very basic automated web vulnerability screening (WVS) methods that are simple, faster, and often more intuitive enough.

There are many web vulnerability scanning (WVS) products currently on the market, with different performance characteristics and capability to identify vulnerabilities. This research provides a comparative study of many commercial and open-source web vulnerability scanners, and reviews their effectiveness within several different test contexts, to identify which, if any, tools provide the most reliable results on web application security vulnerabilities. This study focuses primarily on gray-box testing, where the tester has some prior knowledge of the program. We will primarily use precision, recall, and the Youden Index to evaluate the final results. This document is divided in five main sections: Introduction, Background, Related Work, Research Methodology, and Conclusion.

II. Background

This section provides the background for the study. It includes an introduction to web application vulnerability scanners (WAVS), different approaches for scanning, and benchmarking framework.

A. Web Application Vulnerability Scanner

Web vulnerability scanners are automated security tools that discover vulnerabilities in web applications while in execution [8]. Vulnerabilities are discovered by using predefined test cases, though human judgement is necessary to accurately analyse the results. Scanners can be configured by the user to meet their required testing criteria. They form an important part of cybersecurity and allow users to discover vulnerabilities, before too late, before they are exploited by an evil entity. After that Most Web Application Vulnerability Scanners (WAVS) have three basic components: crawler, scanner engine, and analysis. Each component has a function to perform in the vulnerability discovery process.

- Crawler: The crawler is responsible for exploring and mapping out the structure of the target web application. It systematically browses through online pages, forms, and links, discovering the available endpoints and inputs. This ensures that the scanner understands the application’s complete attack surface. [9].
- Attacker (Scanner Engine): Once the application has been mapped, the scanner engine uses the detected inputs to systematically attack the target using a range of payloads and attack methods. This emulates a real attack using methods such as SQL injections, cross-site scripting (XSS), and other vulnerabilities. This tests how the application responds to bad input [3].
- Analysis Module: The last step analyses the application’s response to the injected payloads. This identifies if the experience demonstrates a vulnerability, a false positive, or a good input. Advanced analysis modules may also provide a risk assessment, quantify the vulnerability, and make recommendations for remediation [9].

B. Types of Scanning

Before starting the testing, it’s critical to understand what type of security testing you are going to do. Generally, it will depend on the goals of the testing, the resources available, and what level of access you have to the application in question. When it comes to web vulnerability scanning, it will generally fall into three categories: black-box testing, white-box testing, and grey-box testing.

- Black-box Testing: In this type of testing, the scanner does not know anything about the application's source code, the technology that was used, or any internal infrastructure. The scanner does crawl to discover content and fuzzing to check for weaknesses. Black-box testing is generally associated with Dynamic Application Security Testing (DAST) [12].
- White-Box Testing: This type of testing gives the scanner complete access to the application's internal structures, including the source code. White-box testing can analyse all application logic and is generally categorized as Static Application Security Testing (SAST) [13].
- Grey-Box Testing: This type of testing gives slightly limited visibility into the application: some URLs, technologies that were used, pre-determined things that the application does, etc. Grey-box testing is a compromise between black-box testing and white-box testing, since it combines elements from both, thus improving the chances of discovering vulnerabilities [8].

C. Benchmarking frameworks

Precision: It signifies the percentage of true vulnerabilities that are correctly identified by a Web Application Vulnerability Scanner (WAVS) from all of the vulnerabilities it reported. This result is defined as true positive accuracy and is used to measure the extent to which a scanner reduces false-positives or at least is able to demonstrate when it has accurately identified a vulnerability from the other non-vulnerabilities that made up its reporting [3].

$$Precision = \frac{TP}{TP+FP} \qquad (1)$$

Recall: Recall can be thought of as the percentage of True Positive (TP) examples out of all true positive examples in the data set [13].

$$Recall = \frac{TP}{TP+FN} \qquad (2)$$

The Youden's Index can be used as an overall measure for how effective a diagnosis tool is, and is expressed as a single value that can range from -1 to 1:

- Score of 1: Represents perfect performance where the tool identified all true vulnerabilities, and did not produce any false positives.

- Score of 0: Represents random performance; aka there is no diagnostic benefit from the tool beyond random guessing, and as a diagnostic tool for vulnerabilities it is completely ineffective.
- Score of -1: Represents totally misplaced performance, i.e. the tool only produced false alerts and missed identifying any true vulnerabilities [6].

$$J = \frac{TP}{TP+FN} + \frac{TN}{TN+FP} - 1 \qquad (3)$$

The Youden Index effectively balances sensitivity and specificity, making it particularly valuable for assessing the overall quality of vulnerability detection tools.

III. Related Works

Azwar et al. [3] compares four Web Application Vulnerability Scanners (WAVS) OWASP ZAP, Wapiti, Arachni, and Burp Suite Professional against current Node.js applications. They initially tested the effectiveness of the WAVS against two vulnerable Node.js applications (DVNA and NodeGoat) and concluded the tools had moderate effectiveness, with F-measure values between 0.4-0.6. Burp Suite Professional had the better detection rates, while Arachni had the best precision with no false positives. The research reveals the current effectiveness of WAVS in detecting vulnerability in JavaScript applications is minimal, with no WAVS detecting 60% of the known vulnerabilities, and further studies are necessary using other JavaScript frameworks. This study [4] goes on to compare three open-source web vulnerability scanners, OWASP ZAP, Skipfish and w3af, using DVWA (Damn Vulnerable Web Application), to determine the effectiveness of identifying OWASP Top 10 vulnerabilities. The researchers compared each scanner on input vector coverage, audit ability, and efficiency and precision of vulnerability detection, as well as time spent scanning. Results demonstrate that OWASP ZAP was the best overall, with the greatest amount of true positives and reasonable scanning time (2m 50s) and Skipfish (1m 48s), while w3af had the greatest time scanning (5h 20m) and least effectiveness in performance. Their conclusion was that no scanner could sufficiently identify all OWASP Top 10 vulnerabilities completely. The author Marwam et al. [5] proposed a new benchmarking framework for testing the performance of scanners, and then compared the performance of a number of tools, including OWASP ZAP, Burp Suite Professional, Qualys WAS, Arachni, Wapiti3, and Fortify WebInspect. Of all the tools assessed, the most effective overall scanner was Burp Suite Professional. The study [2] concluded using both automated scanning and manual testing, using six vulnerable web applications (bWAPP, OWASP_Bricks, DVWA, WackoPicko, WebGoat, and Gruyere). The authors compared effectiveness using precision, recall, and F-measure to determine false positives and protocol. In terms of SQL injection, Paros Proxy was most effective with an F-measure of (79.95%), whereas in XSS detection, Skipfish was best with an F-measure of (73.3%). The findings of the research established scanner effectiveness varied significantly depending on vulnerability type, with SQL injection generally being more effectively scanned monitored than XSS attacks. The research study [6] focused on the effectiveness of multiple web application security scanners, including commercial solutions (Acunetix, HP WebInspect, IBM AppScan) compared to open-source tools (OWASP ZAP, Skipfish, Arachni, Vega, and Iron WASP). Through their methodology, comprehensive testing of the scanners revealed broad, significant performance variations of the scanning tools. They found no one scanning tool was better than another in every facet of vulnerability detection. The authors conclude organizations may achieve wider coverage in security by implementing a multiple-scanner model, where they can best utilize the strengths of multiple scanning tools together to help maximize vulnerability detection. The author Lyubka [7] generates a paper that accounts for what was. They perform comparative studies of Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST) as it uses open-source tools. The study investigates the performance of tools such as SonarQube (SAST) and OWASP ZAP (DAST) as they report vulnerabilities on web applications. The authors found both strengths and weaknesses of each strategy, identifying SAST was key for early detection in the security development lifecycle, whereas DAST was preferable for identifying runtime vulnerabilities. The proposal for SAST and DAST together is determined to provide a greater extent of security coverage.

IV. Rsearch Methodology

We conducted a comprehensive assessment of several web application vulnerability scanners, both commercial and open source, as part of our tool selection. Due to the high cost associated with many of the commercial tools, we intentionally chose a balanced portfolio of two open-source tools, OWASP ZAP [12] and Nikto [14], and considered two commercial scanners that did not exceed budget: Nessus [15] and Burp Suite Professional [5]. As shown in

Fig. 1. approach provides a thorough comparison and considers fiscal appropriateness and includes multiple scanning methods and detection capabilities across multiple areas of the web application security market.

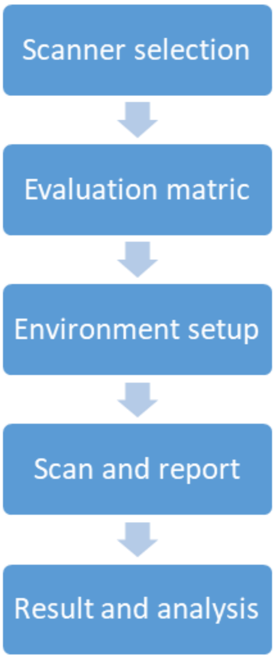

Fig. 1. Research workflow.

A. Scanner Selection

Regarding tool selection, we fully explored web application vulnerability scanners in both commercial and open-source space. Many commercial options came with steep licensing prices other have not been updated for a long time but we were still able to choose a reasonable mix including in total two open-source tools (OWASP ZAP [12], Nikto [14]) as well as two commercial scanners in our budget, Nessus [3] and Burp Suite Professional [5]. We have offered a comprehensive comparison, within budget limits, as well as a good mix of scanning methods and detection capabilities across various segments of the market.

Table I. Table Type Styles

S.no	Name	License	Latest version
1	OWASP zap	Open source	v2.16.1
2	Burpsuite pro	Commercial	2025.3.3
3	Nessus	Commercial	10.8.4
4	Nikto	Open source	2.5.0

B. Evaluation Metric

Following the process of tool selection, appropriate evaluation metrics were developed to ensure a complete performance evaluation. The benchmarking system includes precision, recall, and the Youden Index to evaluate performance. These metrics can be synergistically utilized to assess different aspects of scanner performance: that is, precision allows us to evaluate how close to actual cases the positive assertions of the tool are (by minimizing the number of false positives), recall allows us to evaluate how thorough the vulnerability detection is in terms of locating actual cases (by selecting the true positives), and Youden Index gives an overall impression of diagnostic performance. Using multiple metrics facilitates an overall evaluation of detection, precision, and discriminative power across various groups of vulnerabilities and provides a holistic view of the qualities of each scanners' performance.

C. Environment setup

In this experimental setup we have two susceptible machine scenarios, each running identical computing resources to ensure a fair, valid comparison analysis. The first machine is running locally in our controlled network environment, while the second machine is running as a live instance on the internet and offers various network conditions for testing purposes.

The susceptible applications for this configuration are testphp.vulnweb.com [11], and Damn Vulnerable Web Application (DVWA) [10]. The settings of DVWA were configured to a "low" security setting to allow scanning tools to discover as many vulnerabilities as possible. We also created authentication credentials and gave them to each scanning tool, to allow the full penetration of the application and some access to the authenticated sections of the apps. This two-environment approach allows us to test the scanning capabilities of the scanner in two different environments controlled local, and real-world accessible via the internet and the same susceptible apps provide an equal baseline for comparative evaluations across pertinent variations without outlier concerns.

D. Scan and report

After completing vulnerability scans across various designated tools, they produced detailed data that was gathered, examined and analysed. We systematically evaluated the performance of each scanner using standard formulas for precision, recall and the Youden Index, in order to obtain quantifiable measurements for comparative purposes.

In vulnerability detection systems, scan results are classified into four categories, based on the accuracy of the detection results.

- True Positive (TP): Occurs when the system correctly detects the existence of a true vulnerability. This is the successful identification of a real security vulnerability wherever it was accurately designated as having a security vulnerability in the scan.
- True Negative (TN): Occurs when the system correctly detects the absence of a vulnerability in a particular space or component. This is the appropriate distinction between secure and unsecure code.
- False Positive (FP): Occurs when the system incorrectly detects a vulnerability that is not there. These are false signals that can lead to wasted remedial effort and loss of confidence in the scanning tool.
- False Negative (FN): Occurs when the system fails to detect a vulnerability that is present. This is the worst category of error, as unrecognized vulnerabilities remain exploitable and create ongoing security risk.

By applying equations (1), (2), and (3) to the data obtained from the scanners, the following results were obtained

TABLE II. DATA OF TESTPHP.VULNWEB.COM

WAS	testphp.vulnweb.com						
	TP	TN	FN	FP	precision	recall	YI
zap	23	18	7	22	0.51	0.77	0.22
nessus	19	14	11	26	0.63	0.63	-0.02
burpsuite	27	20	3	10	0.73	0.9	0.57
nikto	16	15	25	25	0.39	0.53	-0.09

TABLE III. DATA OF DVWA

WAS	DVWA						
	TP	TN	FN	FP	precision	recall	YI
zap	30	14	10	26	0.54	0.75	0.1
nessus	22	12	18	28	0.44	0.55	-0.15
burpsuite	35	18	5	16	0.69	0.88	0.41
nikto	18	11	22	29	0.38	0.45	-0.27

E. Results and analysis

This indicates the research conclusions from testing four web vulnerability scanning tools (OWASP ZAP, Burp Suite Professional, Nessus, and Nikto) on two vulnerable web applications testphp.vulnweb.com, and Damn Vulnerable Web Application (DVWA). The evaluation of application security is based on three measures, which are precision, recall, and Youden Index (YI).

The research findings show that a considerable disparity exists between the performance of the web vulnerability scanners under the two test conditions. Tables II and III outline all of the performance characteristics for each scanner with respect to both vulnerable applications.

Burp Suite Professional performed quite well under both test conditions, achieving the highest probe precision scores (0.73 and 0.69), recall scores (0.90 and 0.88), and Youden Index scores (0.57 and 0.41). This essentially meant that Burp Suite would have identified approximately, 90% of actual vulnerabilities and still maintained a high level of accuracy in its positive predictions. OWASP ZAP performed equally well against both applications with moderate precision (0.51 - 0.54) and good recall (0.75 - 0.77) which would make it a very valid option for comprehensively identifying vulnerabilities although it had a much higher rate of false positives than Burp Suite. Nessus had moderate performance overall as the precision ranged from 0.44 to 0.63 and the recall was 0.55 to 0.63, however there were both near-zero or negative Youden Index scores (-0.15 to -0.02) which means there was very little discriminative ability. Nikto was the lowest ranking scanner on all metrics, and had low precision (0.38 - 0.39), low recall (0.45 - 0.53), and negative Youden Index (-0.09 - -0.27), which indicated that it performed slightly better than random guessing.

From the results, it is clear that most scanners had slightly improved performance on testphp.vulnweb.com over DVWA, suggesting that the architecture of the applications and vulnerabilities affected the ability of the scanners themselves. The results are ranked as follow overall based on the assessments: Burp Suite Professional (best overall), OWASP ZAP (good performance), Nessus (low level of success), and Nikto (least successful).

V. Conclusion

In this investigation, four web vulnerability scanners (OWASP ZAP, Burp Suite Professional, Nessus, and Nikto) were analysed using two web applications with known vulnerabilities. Results demonstrated substantial differences in performance and all the scanners failed to find 100% of the vulnerabilities in every vulnerability with vulnerabilities found. Burp Suite Professional was the best overall scanner with accuracy, recall, and generally strong performance across the board. This commercial product showed a good combination of accuracy and the ability to detect vulnerabilities in an organization, making it the best tool for organizations that want to perform comprehensive vulnerability scans.

OWASP ZAP did the best among the open-source products, with respectable performance with a high recall rate and average accuracy, this made it a very affordable and available potential tool for organizations to use, because OWASP ZAP is free. Relative to ZAP, Nessus and Nikto delivered weaker results. Overall, the results show that commercial tools outperformed open-source products; however, OWASP ZAP could be a good tool for comprehensive scans.

References

[1] "2025 Web Application Security Report," Fortinet. [Online]. Available: https://www.fortinet.com/resources/reports/application-security-report. [Accessed: May 27, 2025].

[2] M. R. Mohammed, "Assessment of web scanner tools," Int. J. Comput. Appl., vol. 133, no. 5, pp. 1-4, Jan. 2016.

[3] Al Anhar, Azwar, and Y. Suryanto, "Evaluation of web application vulnerability scanner for modern web application," in Proc. 2021 Int. Conf. Artificial Intelligence and Computer Science Technology (ICAICST), 2021, pp. 200-204.

[4] D. Sagar et al., "Studying open source vulnerability scanners for vulnerabilities in web applications," IIOAB J., vol. 9, no. 2, pp. 43-49, 2018.

[5] M. Albahar, D. Alansari, and A. Jurcut, "An empirical comparison of pen-testing tools for detecting web app vulnerabilities," Electronics, vol. 11, no. 19, p. 2991, 2022.

[6] R. Amankwah et al., "An empirical comparison of commercial and open-source web vulnerability scanners," Softw. Pract. Exp., vol. 50, no. 9, pp. 1842-1857, 2020.

[7] L. Dencheva, "Comparative analysis of Static application security testing (SAST) and Dynamic application security testing (DAST) by using open-source web application penetration testing tools," M.S. thesis, National College of Ireland, Dublin, Ireland, 2022.

[8] P. S. Aarya et al., "Web scanning: existing techniques and future," in Proc. 2018 Second Int. Conf. Intelligent Computing and Control Systems (ICICCS), 2018.

[9] P. A. Sarpong et al., "Performance evaluation of open source web application vulnerability scanners based on OWASP benchmark," Int. J. Comput. Appl., vol. 174, no. 02, 2021.

[10] H. S. Abdullah, "Evaluation of open source web application vulnerability scanners," Academic J. Nawroz Univ., vol. 9, no. 1, pp. 47-52, 2020.

[11] A. Murzaeva and S. Akleylek, "An automated vulnerable website penetration," in Proc. Int. Conf. Advanced Technologies, Computer Engineering and Science (ICATCES'18), 2018.

[12] U. S. Potti, H. S. Huang, H. T. Chen, and H. M. Sun, "Security Testing Framework for Web Applications: Benchmarking ZAP V2. 12.0 and V2. 13.0 by OWASP as an example," arXiv preprint arXiv:2501.05907, 2025.

[13] K. Abdulghaffar, N. Elmrabit, and M. Yousefi, "Enhancing web application security through automated penetration testing with multiple vulnerability scanners," Computers, vol. 12, no. 11, p. 235, 2023.

[14] A. M. Sllame, T. E. Tomia, and R. M. Rahuma, "A Holistic Approach for Cyber Security Vulnerability Assessment Based on Open Source Tools: Nikto, Acunitx, ZAP,

Nessus and Enhanced with AI-Powered Tool ImmuniWeb," in Proc. 2024 IEEE 4th Int. Maghreb Meeting of the Conf. Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA), Tripoli, Libya, 2024, pp. 68-75, doi: 10.1109/MI-STA61267.2024.10599685.

[15] M. A. Muin, K. Kapti, and T. Yusnanto, "Campus website security vulnerability analysis using Nessus," Int. J. Comput. Information Syst., vol. 3, no. 2, pp. 79-82, 2022.

Citation: Tenzin Yarphel. (2025). Comparative Performance Analysis of Web Vulnerability Scanners. International Journal of Information Security (IJIS), 4(1), 98-110.

Abstract Link: https://iaeme.com/Home/article_id/IJIS_04_01_005

Article Link:
https://iaeme.com/MasterAdmin/Journal_uploads/IJIS/VOLUME_4_ISSUE_1/IJIS_04_01_005.pdf

Copyright: © 2025 Authors. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Creative Commons license: Creative Commons license: CC BY 4.0

✉ **editor@iaeme.com**