National College of Ireland

Evaluation of Open-Source Vulnerability Scanners for Web Applications and WordPress Websites

MSc Research Project
MSc Cybersecurity

Muzammil Hussain
Student ID: x23218029

School of Computing
National College of Ireland

Supervisor: Kamil Mahajan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Muzammil Hussain.....

Student ID: x23218029.....

Programme: MSc Cybersecurity..... **Year:** 2024.....

Module: Thesis.....

Supervisor: Mr. Kamil Mahajan.....

Submission Due Date: 12-Dec-2024.....

Project Title: Evaluation of Open-Source Vulnerability Scanners for Web Applications and WordPress Websites

Word Count: .....8,937..... **Page Count**.....23.....

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: .....Muzammil Hussain.....

Date: .....11-Dec-2024.....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Evaluation of Open-Source Vulnerability Scanners for Web Applications and WordPress Websites

Muzammil Hussain
x23218029

Abstract

Web applications are more vulnerable ever than before. Everyone is using web applications to somehow and businesses are more worried about the confidential details i.e. informational or financial of their valued customers to expose into the wrong hands. Cyber-criminals always try to break down the security of these applications to get unauthorized access. So, it is better to find the weak points and secure them prior to attack with the help of automatic vulnerability scanners because manual evaluation is pretty much difficult and time consuming. Mostly vulnerability scanners are commercial and expensive to buy. The motive of this research is to evaluate the performance of open-source vulnerability scanners and propose a best open-source vulnerability scanner for web applications including WordPress websites based on evaluation criteria. Five open-source vulnerability scanners i.e. reNgine, Nessus (essentials), OWASP ZAP, Wapiti and Burp Suite (community) are tested to find the vulnerabilities in Open Web Application Security Project-Broken Web Application (OWASP-BWA) and WordPress website hosting on Amazon Web Service-Elastic Compute Cloud (AWS-EC2) instance. reNgine is proposed tool to find the vulnerabilities of web applications based on evaluation criteria i.e. free available, easy to integrate and use, find vulnerabilities as per OWASP-Top 10 vulnerabilities, provide detail compliance documentation, smart alerts capabilities with continuous asset monitoring and data correlation features.

1 Introduction

Most of the business industries especially retail sector, shopping, banking, tickets booking are dealing with the customers at online platform nowadays by using various applications. These web applications are made now as mandatory part of daily life. Some of these web applications are hosted on premises and some of them are managed by the cloud providers. So, continuously using these web applications need some safeguard as well against the potential threats because cyber-criminals are always trying to attack such applications to steal away the data and sensitive informations. Web application vulnerabilities are the main reasons of damage which is increasing day by day. It is extremely difficult to develop a such web application which is completely secure and protected. It is not possible to check all the web application vulnerabilities by hand because it requires a lot of time, efforts and heavy cost. So, demand for automated vulnerabilities scanners has increased now. There are several automated vulnerabilities scanners available, some of them are commercial and many are free for use. These tools play crucial roles to detect the vulnerabilities and provide the mitigation against them. This study will focus on the evaluation of open-source vulnerability scanners (OpenVASs) and their performance to find the vulnerabilities. Open Web Application Security Project (OWASP) has listed top 10 web applications security vulnerabilities and

risks.[1] In addition to this, scanning and evaluation of cloud-based application i.e. WordPress will also be tested.

The motivation is taken from increasing a large numbers of cyber-attacks on web applications which require more advanced OpenVASs tools to find the vulnerabilities timely and efficiently. Nearly all companies are concerned about escalating cyber threats and the evolving regulatory environment, and vulnerability scanners assist them in effectively managing their vulnerability management programs. Vulnerability scanners autonomously identify and report any detected vulnerabilities, and their utilization in an automated setting enhances the organization's security capabilities. Vulnerability scanners freely examine and detect the deficiencies in computer systems, online web applications, and networks. Security specialists at firms consistently utilize automated scanning technologies to ensure that systems and apps are updated against any cyber-attacks. Automated vulnerability scanning technologies produce reports identifying obsolete software, updates, and misconfigurations, enabling specialists to fix these issues promptly. Several studies have already been done on OpenVASs tools to find the vulnerabilities in web applications. Mostly tools are used to detect common vulnerabilities or known vulnerabilities i.e. SQL injection, cross-site scripting (XSS) or outdated softwares etc. However, there is a research gap to explore new attack vectors such as cloud-specific vulnerabilities, API vulnerabilities and zero-day detection. Scanners are good enough to find the common vulnerabilities and exposures based (CVE) attacks while zero-day detection is required an Artificial intelligence (AI)-driven model.[2] There is a lot of improvement as well to integrate the scanners with CI/CD pipelines but still there is a large gap with real-time scanning during the integration and deployment.

Research Questions:

(a) Evaluation of Open-Source Vulnerability Scanners for Web Applications and WordPress Websites.

(b) Recommendation of a latest advanced open-source tool for asset discovery, vulnerability assessment and continuous monitoring.

Commercial tools are not used and tested in this research study due to limitation of budget. In addition to this, I have limited access of AWS account which may affect the evaluation and results of chosen OpenVASs.

There are several OpenVAS tools available both commercially and free. In this study, I have tested and evaluated five OpenVAS tools and three of them are limited versions i.e. reNgine, OWASP ZAP, Wapiti, Nessus (essentials) and Burp Suite (community). Some of them are working only with Linux operating system (OS) while some are working with both Windows (OS) and Linux operating system (OS). As a part of testing environment, I have experimented these tools on OWASP-Broken web applications (BWA) and cloud-based web

[1] https://owasp.org/Top10/

[2] https://cve.mitre.org/

application i.e. WordPress hosting a website. I have used Amazon Web Services (AWS) as infrastructure as a service (IAAS).[3] Virtual machine (VM) instance i.e. Amazon Elastic Compute Cloud (Amazon EC2) has initiated to configure and host a website.

The structure of this research report is as follows: **Section II** mainly deals with background study along with some basic concepts of primary vulnerability testing approaches i.e. white-box testing and black-box testing. OWASP top 10 web application vulnerabilities are discussed in detail. **Section III** consists of the literature review of related research work which has already been done and identifies the research gaps in the literature review. **Section IV** details about methodology i.e. information gathering steps along with evaluation criteria and OpenVAS tools which are going to use in this research study. **Section V** comprises on design specifications and implementation i.e. architecture diagram and testing environments setup i.e. OWASP-BWA and Amazon-EC2 instance which are going to use during the research work. **Section VI** explains about the results and evaluations of OpenVAS tools. It shows comparison of the results of chosen OpenVAS scanners on selected testing environments. **Section VII** contains the conclusion that identifies the main findings and gaps of the research project. It also includes the information about the proposed OpenVAS scanner for web applications and WordPress sites.

2 Background Study

2.1 Open Web Application Security Project (OWASP)

Application security vulnerabilities are the deficiencies in your web applications that cybercriminals constantly work toward to exploit. These vulnerabilities may exist in code, design, plugins and tools. Their exploitation can lead to unauthorized access, theft of sensitive information, or disruption of the whole web application. OWASP is a non-profit organization that provides the guidance to the developers to develop and design the web applications and software solutions. OWASP top 10 is the list of top security issues in the web applications that is based on the consensus among the top developers. The list states the most important security vulnerabilities in the web applications and provides the solutions to deal with them.[4]

Broken Access Control: By these is the type of vulnerabilities attackers can easily evade the access control mechanism of any web application by making some changes in the permissions or in some other method. This will allow the attacker the unauthorized access to the sensitive data or systems. Broken access control was at position fifth in 2017 while now it is on top security risks in web applications. It is the mergence of two other categories i.e. missing function access control and insecure direct object reference.

Cryptographic Failures: These security risks rise when cryptographic methods are not properly implemented to protect the data. These vulnerabilities comprise on the use of cryptographic ciphers which are obsolete and cryptographic protocols are not properly

[3] What is Amazon EC2? - Amazon Elastic Compute Cloud

[4] https://owasp.org/www-project-top-ten/

implemented as well as cryptographic controls issues. Previously it was known as sensitive data exposure while OWASP changed its name to cryptographic failures to reflect the importance of cryptographic protocols.

Injection: These types of vulnerabilities allow attackers to inject the malicious data into the web applications by using the commands or redirect the users to the malicious websites or even totally changes the application. The most important attack vector of this type of category is Structured Query Language Injection (SQLi) which is used to access the web application database by substituting the malicious code to database query. The remedial action against such attacks is to completely authenticate all untrusted data specifically the data which is submitted by the end user. Cross-site scripting (XSS) is also included in this category now. It is the most common cyber-attacks. Malicious code is executed in the form of bits of JavaScript code. This code then executes by the victim's browser. These malicious code may also be written in Java, Hypertext Markup Language (HTML) and Ajax.[5]

Insecure Design: The security risk in this category arises from insecure architecture design of the web applications. When a web application is developed by using an authentication process which is not secure, or a website is not built securely to prevent the bots attacks which is a computer program that operates as an agent to execute some task automatically without instructions from humans.

Security Misconfiguration: Security misconfigurations in a web applications cause such attacks. For example, a web application does not properly filter input packets correctly and may be enable the default user ID, password or authorization.

Vulnerable and Outdated Components: These types of security vulnerabilities arise when developers use the vulnerable software components in applications. These vulnerable components may include libraries, frameworks, application programming interface (API) etc. These threats are also increase in case of outdated or unpatched softwares. It may also cause in case of unpatched the underlying operating system or out of date APIs.

Identification and Authentication Failures: Authentication issues that to steal the credential informations and cause the brute-force attacks are the main reasons of such security vulnerabilities. The applications which do not use the multifactor authentication and do not invalidate the expired sessions or inactive users are also included in this category.

Software and Data Integrity Failures: The application code or infrastructure that do not protect the software or data integrity may cause such attacks. For example, if the digital signatures are not used while updating the software packages then such danger may cause.

Security Logging and Monitoring Failures: These types of attacks occur when a system is not properly managing the logs of the events and fail to monitor and detect the attacks properly. Proper and detail examining is the key to avoid such cyber-attacks.

Server-Side Request Forgery: Web applications must be able to perform adequate validation of user-provided resources to protect against such cyber-attacks. These vulnerabilities can use by threat vectors to make the applications access malicious websites.

[5] https://www.techtarget.com/searchsecurity/definition/cross-site-scripting

2.2 Top 7 Cloud Vulnerabilities

Cloud environments have been dramatically increased in used in recent years due to its various potential benefits to organizations in the sense of flexibility, availability, cost effective and security. However, if we investigate its downside there are some weaknesses in it that cyber-criminals take advantages to steal secure informations. I will discuss them precisely.

Misconfiguration: There are many security configurations which should be properly implemented to secure the data and applications i.e. to access the file companies share a URL which is link-based file sharing mechanism which creates complexity on cloud.[6]

Shadow IT: It is the use of cloud assets without the permission of IT department. It increases the workload on cloud to create the access of personal usage. It also causes to data loss by unauthorized access and communication.[7]

Insecure Interfaces and Application Programming Interfaces (API): Insecure API can be vulnerable to exploitation. For example, lack of access control and not limiting the access to API can cause many span requests and attacker may exploit API misconfiguration to get unauthorized access.

Zero Day Vulnerabilities: It is a type of vulnerability which is identified and attacked by the cyber-criminals before the software manufacturer. These are extremely dangerous because many customers are using same cloud environment which can lead to access the sensitive data.

Access Management: Mostly cloud environments are easily accessible via public internet which makes it easy to access to vulnerable cloud infrastructure. Issues like weak passwords, failure to use multi-factor authentication (MFA) and granting excessive access to users are common.

Lack of Visibility: Mostly companies mix and match the cloud technologies from several service providers which creates a room for vulnerabilities to exploit due to lack of visibility.

Malicious Insiders: insiders are most dangerous than all other vulnerabilities because any individual that some have access to, and knowledge of company's IT environment is riskier in case to leave the company. It may include any third-party vendor or partner.

2.3 Vulnerability Scanning Methods

Vulnerability scanning is the most important step to discover the possible threats and weaknesses in a web application, network or any computer system network before cyber-criminals exploit them to make a cyber-attack. Vulnerability scanners use the databases of know vulnerabilities to identify and detect the security risks i.e. misconfigurations or missing updates etc. There are two different types of testing approaches stated below.

White Box Testing: It is a type of application testing technique that focuses on the internal logic of the application code and structure of the program to point out the coding errors and

[6] Top 7 Cloud Vulnerabilities In 2024 - Check Point Software

[7] 8 All-Too-Common Cloud Vulnerabilities | Wiz

designing flaws. It enables developers to understand the application's inner working, infrastructure and integrations. Aikido, Veracode are the famous tools for white box testing.[8]

Black Box Testing: It is the technique which is used to test and analyse the application's functionality without having the complete knowledge about the internal design, architecture and code. Various automated software tools i.e. OWASP ZAP, Nessus etc. are used to scan the web applications to find the vulnerabilities and generate the reports at the end of the testing.[9]

Grey Box Testing: It is the mixture of white box testing and black box testing. It means that tester have also some knowledge about the internal architecture of application but mainly focuses on the output rather than how the output is generated. Burp Suit and Wireshark have features of both testing mechanisms.[10]

2.4 Web Application Vulnerability Scanners

There are three main components of a web application security scanner i.e. a crawler, attacker and analyzer.[11] Crawler is the responsible to gather the website data and find the web pages that are accessible of the scanned application and identifies the entry points such as HTML forms input, GET or POST parameters etc. While on the other hand, attacker component sends the random and invalid inputs to the web application to analyze it for another component. Finally, analyzer component analyzes the return data, detects the vulnerabilities and generates the report.[12]

3 Literature Review

As a part of the literature review, several research papers, articles and journals has been studied to understand the up-to-date work on the subject title. Literature review also helped me to find the research gaps.

I started my literature review by three papers, (Elshheibia, Mohamed and Almahdi, 2024), (Alya Geogiana Buja *et al.*, 2024) and (Sllame, Tomia and Rahuma, 2024). Focusing on third paper, researchers used some well-known open tools i.e. Nikto, Acunetix, ZAP, Nessus and AI-powered tool ImmuniWeb to find the potential vulnerabilities in the system. They applied these all tools together as multi-layered security assessment strategy which is also called a holistic approach. A case study is presented in the paper in which various web sites are used as a vulnerability assessment i.e. Hackthissite, CTFlearn, Google Gruyere, OWASP Mutillidae II. In addition, researchers implemented all tools on Hackthissite and OWASP Mutillidae. Netcraft used to find the IP address. Further details collected by using NMAP and

---

[8] What is White Box Testing? (Example, Types, & Techniques) | BrowserStack

[9] Black Box Testing vs. White Box Testing (spiceworks.com)

[10] What Is Grey Box Testing? | Coursera

[11] Stages of Scanning | Invicti

[12] What Is a Web Crawler? | How Do Crawlers Work? | Akamai

Nessus used later as a vulnerability assessment enhanced by AI-powered tool ImmuniWeb. In conclusion, this research report did not address the WordPress related vulnerabilities. The holistic approach is not fit to find all web application vulnerabilities. The advanced tools with evaluations are still required for moving forward.

Furthermore, five more research papers were analyzed on subject problem, four of them (Shaikh and Lokhande, 2024), (Premchand *et al.*, 2024), (Kyer, no date) and (Shivananjappa and Creutzburg, 2024), thoroughly discussed about the emergence of Artificial intelligence and machine learning techniques with the existing solutions to find vulnerabilities and mitigations. They highlighted the effective measures together i.e. vulnerability scanning including domain and subdomain enumeration tools, penetration testing, SAST tools i.e. J-WAVE and open-source tools i.e. OWASP ZAP, OSV scanner etc. empowering the organizations to enhance their security features. Fifth paper (Fadlil, Riadi and Mu'min, 2024) critically analyzed about the SQL injection attacks by using the OWASP open-source project. To conclude this, cloud-based environment was not considered in these papers deeply.

Besides this, another paper (Chaturvedi *et al.*, 2024) included in study which focused on the implementation of OpenVAS as scanning platform, Wireshark as a network traffic monitor, Nmap as a port identifier and Metasploit as an ethical penetration testing for robust vulnerability assessment and analysis for IT environment. Two more research papers studied deeply related to vulnerability assessment in web application. (Vemula, 2024) and (Pradhana, 2024) adopted the same methodologies for vulnerabilities assessment with different tools i.e. OWASPZAP and WPScan.

Likewise, one more paper which is thoroughly investigated for this research work, authors (Gajula and Vassilakis, 2024) used well known OpenVASs i.e. ZAP, Nessus, Nikto and Nmap used as a vulnerabilities assessment in various network environment i.e. local system and a cloud based architecture Libode. They did not use the Microsoft Azure and AWS for some limitations and complexities. First, they examined the capabilities and features of the chosen scanners to get the deep understanding of their performances. they tested tools to get the data collection based on the evaluation criteria. Mostly the data was about the network in or network out packets, percentage of CPU usage, RAM usage etc. In conclusion, the research paper is limited to test the network related vulnerabilities, no application is being used to test and analyzed throughout the study. In addition, paper is argued that there is no one scanner which is more effective instead use of multiple scanners is good to find the vulnerabilities in broader view.

Similarly, a paper focused on established a strong and efficient automated penetration testing architecture to find the security vulnerabilities in a web application (Samgir *et al.*, 2024) is also part of study. Metasploit which is powerful tool for exploitation is recommended with the automated tool OWASP ZAP to find the vulnerabilities in web applications. No test is performed during the research instead only case study is presented. Metasploit architecture is comprehensive solution for both detecting and exploit the web application vulnerabilities. OWASP ZAP scans the application to identify the vulnerabilities i.e. XSS, SQL injection etc. then Metasploit automatically trigger to exploit the weaknesses. In conclusion, the proposed architecture with automated tool has some limitations like depend on single tool is risky and over reliance without human oversight. In addition to this, some latest vulnerabilities require some sophisticated tools and analysis.

Correspondingly, another paper discussed about the open source tools for vulnerability assessment with broader scope of application security (Cruz, Almeida and Oliveira, 2023). The primary objective of the research paper is to study various tools for vulnerability assessment across several stages of Software Development Life Cycle (SDLC). Three types of application security testing tools i.e. Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST) and Software Composition Analysis (SCA) discussed in the paper both open-source tools and commercial tools are compared and presented in detail. Comparative analysis is exhibited based on programming language support, CI/CD integration, interoperability, easy setup and budget and based on analysis some tools also recommended i.e. Bandit, Semgrep, OWASP ZAP, Wfuzz, Trivy etc. In conclusion, deep study is being done to find a suitable tools for security vulnerabilities but practically they are not implemented and tested.

Moreover, additional paper which is gone through deeply is a thesis work which is emphasized on vulnerability assessment of web applications specifically SQL injection and cross-site scripting by using the OpenVASs (Matti, 2021). Three tools OWASP Zap, Vega and Wapiti used in the research study to find the desired vulnerabilities. Researcher used OWASP broken web applications to test the mentioned tools. WAVSEP, Bricks, Mutillidae and Security Shepherd are being tested applications. The results show that these scanners use the same techniques to identify the vulnerabilities while the differences are only due to pre-defined payloads and detection approaches. In conclusion, these scanners have some limitations in terms of accuracy and occasionally they highlighted such vulnerabilities that did not exit. In addition to this, these tools are not good for web applications having AJAX or complex JavaScript frameworks.

Further, for better understanding about chosen research topic, three more research papers (Sagar *et al.*, 2018), (Makino and Klyuev, 2015) and (Abdullah, 2020) studied deeply. In all research papers same methodology is used by the researchers. They tested various OpenVASs such as OWASP ZAP, Paros, w3af, Skipfish and OWASP Zed for vulnerability scanning of web applications. They used OWASP broken web application i.e. buggy web application (bWAPP), Damn Vulnerable web application (DVWA) and web application vulnerability scanner evaluation (WAVSEP). The studies emphasized increasing the usage of web application scanners due to growing importance of security issues in web applications. Scanners evaluated based on their abilities to detect the vulnerabilities and in all papers OWASP ZAP is considered as best scanners in terms of finding the vulnerabilities. In conclusion, the researchers approach is good but is limited to less OpenVASs. Some of them were outdated and some of them were limited to find the general vulnerabilities. Mostly scanners were tested with default configurations which may not reflect the full capabilities if they configured optimally.

Another interesting research paper included in this study related to open-source tools which is based on performance based comparative assessment (Alsaleh *et al.*, 2017). Four scanners, i.e. Wapiti, Arachni, Skipfish and IronWasp, were selected initially based on their pre-defined performances such as speed, visualization features, scanning scope etc. Researchers scanned these scanners on three testing environments i.e. Web Scanner test site and two Acunetix sites. For accuracy testing of scanners, they utilized Altoro and WAVSEP, and to know about the crawling capabilities of scanners they tested and set benchmark web

input vector extractor teaser (WIVET) project. After evaluation of all results, they selected two scanners to compare their evaluation with a case study i.e. Arachni and Wapiti. Case study was being done by researchers on AWS-EC2 virtual machines where these scanners were implemented and tested on various URLs. In conclusion, the paper was focused on SQL injections, cross-site scripting and cross-site request forgery (CSRF) related vulnerabilities but it should be more focused on latest security issues. Typical scanners were selected but now we have latest scanners with advanced properties.

In addition, another paper which is added to the research study is the survey about the common vulnerabilities associated with web application's included cloud services and recommended tools to detect them (Onukrane *et al.*, 2023). There is no practical testing of any tools in this study. Researchers went through theoretically various types of web & cloud vulnerabilities i.e. XSS, SQL injections, cross-site request forgery, brute force attack, dictionary attack, phishing attack etc. they also surveyed several detection methods i.e. static, dynamic and hybrid highlighted their weaknesses and strengths. In conclusion, they did not compare any results with real world examples. While they discussed the need for new techniques to identify these threats but did not mention how emerging vulnerabilities will be integrated the existing solutions. Even they did not mention any benchmark of selection of the tools.

Lastly, third last paper which is considered for research is about machine learning techniques to find the vulnerabilities in web application (Hossain Hadi and Hashim Al-Saedi, 2024). Two machine learning algorithms, i.e. Random Forest and Gradient Boosting were used as analysis on a dataset. The other two research papers are theoretical (Vaish *et al.*, 2024) and (Jimmy, no date). They discussed in detail different types of vulnerabilities and remedial actions focused on cloud tools to overcome them. One paper also proposed an online platform, i.e. CySecLearn to promote collaborative R&D and education related to cyber-security and related education.

3.1 Research Gaps.

Mostly all vulnerability scanners are focused on known vulnerabilities, i.e. SQL injections, XSS, outdated software etc. while there is a research gap related to several new attack vectors. For example, API, container and Kubernetes, which are related to clouds. In addition to this, zero-day vulnerabilities are still a big challenge for security professionals. Research is being done already to find these threats using AI-driven models based on behavior and patterns. Furthermore, there is a significant improvement to integrate the scanners with CI/CD pipelines for real time scanning while some scanners are still struggling. Such scanners are needed which should be fast as well as should capable enough to detect the vulnerabilities in highly dynamic DevOps environments. Finally, automated scanners with more creative interfaces could be more effective without knowing too much technical knowledge for non-experts. In conclusion, continuous research in tools to find vulnerabilities in web applications is more important to detect the threats before exploitation by criminals. I will evaluate open-source vulnerability scanners in terms of finding the vulnerabilities related to web applications including cloud-based applications i.e. WordPress website.

4 Methodology

4.1 Information Gathering

The foremost step of the research study is information gathering about the research topic which involves massive background study and previous research papers related to the topic. For this purpose, 23 research papers have been thoroughly studied. In addition, several books, websites, social media and artificial intelligence are considered too. The primary objective was to find information about the critical vulnerabilities based on OWASP Top 10 in web applications and most importantly to find the advanced open-source tools with latest techniques and mechanisms not only to identify the vulnerabilities in web applications but also for continuous monitoring and reconnaissance.

4.2 Evaluation Criteria

As per Table 1, there are seven basic evaluation criteria for the open-source vulnerability scanners which are considered in the research study to propose the best one for scanning of web applications [13] [14] [15].

Table 1: Evaluation Criteria

Evaluation Criteria	Description
Total Number of Vulnerabilities	It is the ability to detect the total no. of vulnerabilities including critical, high, medium, low and informational focusing on OWASP Top 10 i.e. SQL Injections, XSS etc. Also Include the ability to scan different types of applications and environments i.e. web apps, APIs, network services etc.
Ease of Use and Integration	User interface interaction, automation capabilities i.e. CLI usage, API support for CI/CD pipelines, plugin capabilities and Type of testing
Performance and Speed	Speed of scanning specially for large applications for frequent testing, usage of memory and CPU while scanning, minimize false positive
Reporting and Documentation	It is the capability of explained reporting of findings, severity ratings and suggested fixes in HTML, PDF, GUI etc. Detailed documentation is another good feature
Community and Support	Regular updates of vulnerability databases and patches, availability of community support
Compliance and Security Standards	Alliance with standards like OWASP Top 10, compliance regulations i.e. PCI-DSS, HIPAA and produce compliance reports
Cost and Licensing	Open-source or Paid, Usage restrictions,

[13] Best Web Application Vulnerability Scanner | Indusface Blog
[14] web-app-vulnerability-scanners-benchmark-2024.pdf
[15] OWASP Top Ten | OWASP Foundation

4.3 Open-Source Vulnerability Scanners

Based on information gathering and evaluation criteria , five open-source vulnerability scanners have been selected for web application scanning .

4.3.1 reNgine

reNgine is an advanced web application reconnaissance and asset discovery scanner to assist the cyber-officials and penetration testers in identifying the vulnerabilities and attack surface of web applications. It is highly customizable scanner which provides powerful scanning capability through its engine which makes user tune the scanner as per their needs and requirements. reNgine scan engine can be configured to scan for multiple targets. It is the only reconnaissance tool that provides screen shots during the scanning process and view your desired results. It is also effective to find the subdomain discovery and open ports. By using the dashboard, you can easily find your desired subdomains against HTTP status, ports, content length, directory, file name, page title etc. It uses a web crawler to gather endpoints and alive URLs. It has good data correlation, vulnerability report generating, project management and role-based access capabilities. It is also beneficial in continuous monitoring by scheduling the scanning process of web applications as per requirements [16]. Its GUI interface is very friendly and easy to use [17]. It has prominent capability of reconnaissance which is a better alternative compared to the commercial tools which are expensive. Irrespective of your cyber-profession, it provides go-to solution for automating and improving the information gathering efforts. You can configure reNgine to send a notification to you and your development team after finishing the scan.

4.3.2 OWASP ZAP

Its is the mainly used web applications vulnerability scanner by professionals and is being managed by the security professional of OWASP free of cost. It is commonly used to identify the known and unknown vulnerabilities in the web applications [18]. It executes the passive scanning of the web requests. It scans all the files and folders on the server side. It uses crawler method to scan the site's structure completely to retrieve the whole links and URLs. It has full control of web requests between the browser and server. Keeping in view all the above-mentioned characteristics, OWASP ZAP is considered as a right choice to find the vulnerabilities i.e. XSS, compromised authentication, SQL Injection, sensitive data exposure and so on.

4.3.3 Nessus (Essentials)

Nessus has been claiming world's no. 1 vulnerability scanning tools developed by tenable. It is commonly used to scan vulnerabilities in devices, applications, operating systems, cloud

[16] reNgine Documentation
[17] Introducing reNgine — an automated reconnaissance framework. | by Bhavkaran Chahal | SECARMY | Medium
[18] What is OWASP Zed Attack Proxy (ZAP)?

services and other network resources [19]. It is a commercial tool now, but you can use a limited edition with some features free of cost. Nessus has capabilities to identify the software flaws, missing the patches, malware, vulnerabilities related to denial-of-service, and configuration issues. Nessus is famous sue to its plugin database which are automatically compiled in the tool to enhance the scanning performance and reduce the time of assessing the security flaws. Live results are another feature of Nessus which enables it to scan in offline mode with every plug-in update. Nessus can also generate configurable reports in various formats i.e. HTML, comma-separated and Nessus extensible markup language. These reports can be customized as vulnerability types, host vulnerabilities, client etc.

4.3.4 Wapiti

Wapiti is a python-based vulnerability scanner used to find security flaws in web applications. It supports HTTPS, HTTP and SOCKS5 proxies. It offers several features i.e. form-based login authentication, HTTP authentication, URL parameter remover and capability to set initial URLs for exploration [20]. It also has ability to import cookies from Chrome or Firefox and verify the SSL certificates. It is black box scanning tool which means it uses crawling when it discovers web pages. It does not have direct access to source code [21]. It is widely used to scan the web vulnerabilities i.e. SQL Injection, XSS, command injection, path reversal and server-side request forgery. So, wapiti uses fuzzing means to upload different payloads to identify the vulnerabilities.

4.3.5 Burp Suite (Community)

Burp Suite is a professional tool which is mostly used by penetration testers. It is optimised and significantly designed to meet the requirements of professional pentesters. It provides both manual and automatic testing mechanisms. It is available in different versions. Community edition which is part of the research study is a limited version. It has free and limited features as compared to professional edition. It is very easy to install and use. Besides this it has large active community support available which makes it prominent among others [22]. Same as OWASP ZAP, Burp Suite also act as a proxy between the web browser and server hosting the web application. Burp Suite intercepts all the exchange information between these two components to be able to analyze in detail and to modify them. It also has an ability to repeat the requests for visualization for deep analysis to view any changes and modifications. Burp Suite is capable to perform active and passive scanning to find the vulnerabilities in web applications. Its automatic scanner can only be used in professional version but in community edition u can find some vulnerabilities with limited access. Intruder is also included in professional version only. Community edition also does not report generating features.

---

[19] What is the Nessus vulnerability scanning platform? | Definition from TechTarget
[20] Wapiti review (vulnerability scanner for web applications) - Linux Security Expert
[21] Wapiti is a Free and Open-source Web Vulnerability Scanner
[22] Burp Suite, the Tool Dedicated to Web Application Security

5 Design Specifications

5.1 Architecture Diagram

The architecture diagram of practical implementation of the research study to view and analyze the results has been shown in Figure 1 below.

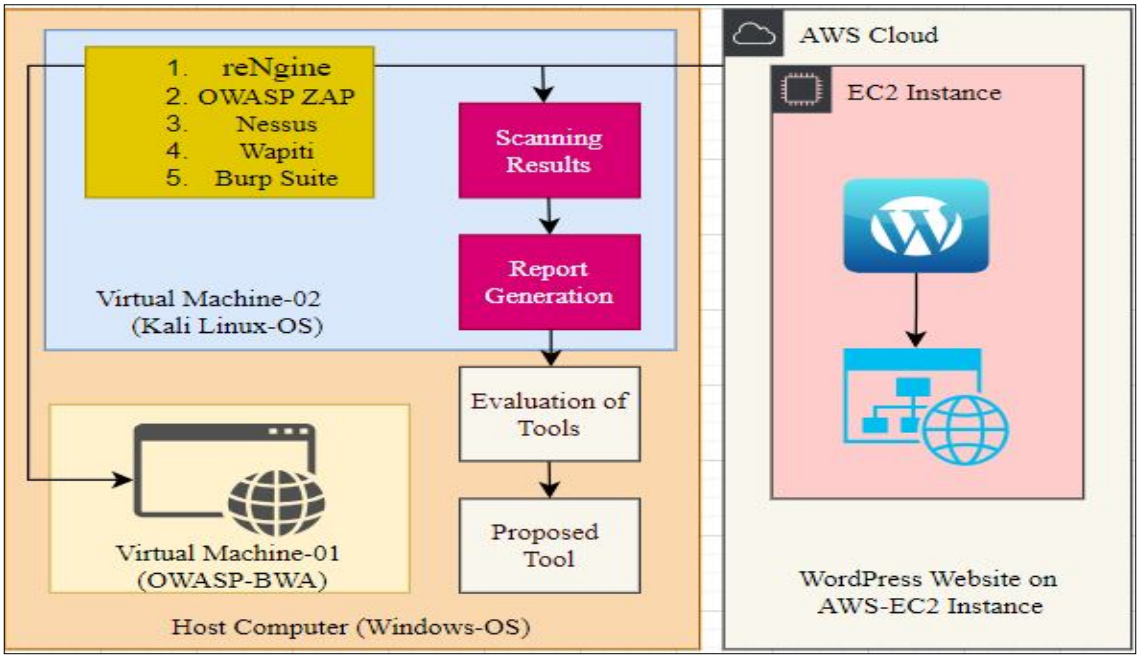

Figure 1 : Architecture Diagram

5.2 Implementation & Testing Environment Setup

I used HP Elitebook 850 G6 as a part of main laptop (Host) due to its powerful hardware features. 64-bit Windows 11 Pro version 24H2 has been installed on it. Its processing power is intel core i5 5th generation processor running @ 1.60GHz and 1.90GHz. 16 GB Ram has been installed in it. Its storage capacity is 1TB. It has 4 cores and 8 logical processors which make virtualization and multitasking very easy.

5.2.1 OWASP-Broken Web Application

As a part of testing environment setup, first virtual machine which has been installed is OWASP-BWA project which is a pre-configured VM consisting of deliberately vulnerable web applications designed to test the vulnerability scanners for professionals [23]. It consists of various vulnerable web applications with known OWASP Top 10 vulnerabilities to test the tools. First, I installed Virtual Box version 7.1.4 from its official link [24]. Then I downloaded the OWASP-BWA project version 1.2.7 and unzip it. Then I applied some settings in virtual

[23] OWASP Broken Web Applications Project download | SourceForge.net
[24] https://www.virtualbox.org/

box i.e. assigned video memory 50 MB, 4 MB RAM, 3 CPUs and configured the NAT network to access from other VM. After these settings, I attached the unzip file i.e. OWASP CL1 and started the VM. After some time, it has been successfully installed and can be connected through IP = 10.0.2.15 from second VM for testing the scanners.

5.2.2 Open-Source Vulnerability Scanners Installation

After setting up the OWASP-BWA virtual machine, I downloaded the kali Linux operating system from the official web site having version (kali-linux-2024.3-installer-amd64). Kali Linux comes up with extensive and built-in toolsets which make it ideal platform for cybersecurity research. It provides an environment with efficient performance and ease of use to conduct various tests i.e. testing and evaluation of open-source vulnerability scanners [25]. I made some settings in virtual box i.e. assigned 8 GB Ram and 4 CPUs, video memory assigned 64 MB, configured NAT network to access the OWASP-BWA machine and Bridge Adapter to access the WordPress website hosting on AWS-EC2 instance. I selected Debian (64-bit) from options and installed Kali Linux. After OS installation, I installed reNgine scanner from official link [26]. After installation, I configured root user and password for login and scanning. Then I installed Nessus from the official link [27]. After installation, I configured it with username and password. It took some time to get plugin updates. Third scanner which I downloaded and installed is OWASP-ZAP [28]. It is a straight way installation. Wapiti and Burp Suite both are pre-installed in Kali Linux Operating system.

5.2.3 WordPress Installation on AWS-EC2 Instance

In addition to OWASP-BWA, I considered Amazon Web Services (AWS) to test the abilities of selected vulnerability scanners whether they can detect the cloud related vulnerabilities in cloud applications or not. AWS provides a wide range of services for computing, storage and much more. I registered as a student package with AWS academic learner lab [29]. I have limited access to resources. I have only 50 $ package on AWS student account to develop and test any cloud applications. AWS has restricted selecting the CPU and RAM capacity as per package which is quite challenging to achieve the desired results. I selected Amazon Linux 2023 AMI with 64-bit Linux OS with 1 vCPU and 1 GiB memory which is free tier version. Then I created a key pair and connected through SSH using power shell. I started to create security groups to authorize HTTP on (port 80), HTTPS (port 443) and SSH (port22) following instance information setup and checking networks and enabling Elastic IP. Then I installed LAMP stack and finally I downloaded and configured WordPress for hosting a web site for testing purposes [30] [31]. By default, AWS EC2 instance and WordPress are not secure.

---

[25] Get Kali | Kali Linux
[26] Installing reNgine on Linux/Windows/Mac - reNgine
[27] Download Tenable Nessus | Tenable®
[28] ZAP – Download
[29] https://awsacademy.instructure.com/courses/98290
[30] https://docs.aws.amazon.com/linux/al2023/ug/ec2-lamp-amazon-linux-2023.html
[31] https://docs.aws.amazon.com/linux/al2023/ug/hosting-wordpress-aml-2023.html

6 Results

The next step is to compile the results of all scanners in terms of defined evaluation criteria. It is crucial step to analyze the performance of all scanners and propose the best one among them. Total number of vulnerabilities in terms of critical, high, medium, low, informational and total in both testing applications (OWASP-BWA & WordPress) are shown in Table 2 below.

Table 2 : Types of Vulnerabilities of Each Tool

Tools	Critical	High	Medium	Low	Informational	Total
reNgine	0	0	1	10	24	35
Nessus (essentials)	3	5	27	4	39	78
ZAP	0	1	11	17	16	45
Wapiti	Not Defined	Not Defined	Not Defined	Not Defined	Not Defined	522
Burp Suite (community)	0	0	0	0	0	0

6.1 Evaluation

6.1.1 Total Number of Vulnerabilities

Vulnerability scanning results look very unusual. It is not as per expectations. Starting from reNgine reconnaissance framework, which can find only one medium vulnerability mostly focused on low and informational issues. It indicates that it focuses on reconnaissance and low priority vulnerabilities. If we investigate the reports, these vulnerabilities include SSH weak key exchange algorithms, self-signed SSL certificates, open ports including http (80), https (43), SSH (22), PHP information detection page and WordPress REST API user enumeration. Mostly informational issues are related to web application filtering (WAF), TLS/SSL certificates and IMAP detection. In addition to this, reNgine successfully scanned OWASP-BWA and WordPress but could not be able to find any cloud related vulnerabilities.

While on the other hand, Nessus results are pretty much impressive in terms of finding the critical and high vulnerabilities and these vulnerabilities are related to SQL injection attacks. Medium vulnerabilities align with the capability to find the configurations and outdated softwares issues. 3 critical vulnerabilities are detected by the Nessus related to old PHP versions which are installed in both applications and required immediate actions. It also detected 5 high vulnerabilities addressing the remote file inclusion and multiple SQL injections. In addition to this, 27 vulnerabilities are found related to cross-site scripting (XSS) in jQuery and HTTP trace methods allowed. 4 vulnerabilities are found pointing towards the

password auto-completion is enabled. Lastly, 39 vulnerabilities are related to information categories addressing the software versions and detection of servers etc.

ZAP scanner results are close to reNgine reconnaissance framework which is predominantly focusing on medium and low vulnerabilities. No doubt the ZAP is controlled by high professionals and free to use to find web application vulnerabilities, but you can see that it does not always detect critical and high severity issues. This could be due to its usage without fine tuning i.e. without payload testing etc. Zap failed to detect any critical vulnerability while successfully detecting 01 high vulnerability related to Open Redirect which can be used to manipulate the web application to redirect the user to a malicious URL. 11 vulnerabilities are related to absence of Anti-CRF tokens, missing anti-clickjacking headers, content security policy header not set and directory browsing. 17 low vulnerabilities are pointing towards the information about server version, cookies without HttpOnly flag, private IP disclosure and missing SameSite attribute. ZAP also detected 16 informational vulnerabilities about cookie poisoning, suspicious comments, session management, information disclosure and charset mismatch.

While testing both web applications using Burp Suite it is found that Burp Suit community edition is not allowed to perform the automatic scanning of web applications. So, the scan was not performed properly and did not run. Due to budget constraints, I cannot purchase the commercial version. Targets were added in the scope to scan but results and vulnerabilities can only be examined manually by forwarding the requests manually in the scanner and match the information with the known vulnerabilities manually which is a time taking process and could not continue at this stage of research. So, Burp Suite is failed to detect any vulnerability in both applications and is out of scope.

Finally, scanning the web applications using Wapiti is little bit time consuming in terms of main domain scanning. Further, it does not define any severity of vulnerabilities which shows it is lack in parsing or classifying results. No doubt that wapiti is identified 522 vulnerabilities relate to path traversal, reflected cross-site scripting, internal server error, content security policy and secure flag cookie. It looks like wapiti is focused on ross-site scripting vulnerabilities in the web applications which other tools might not be or may be the databases used by wapiti might contain some unique patterns to detect such vulnerabilities. Graphical comparison of tools is mentioned below in Figure 2 below.

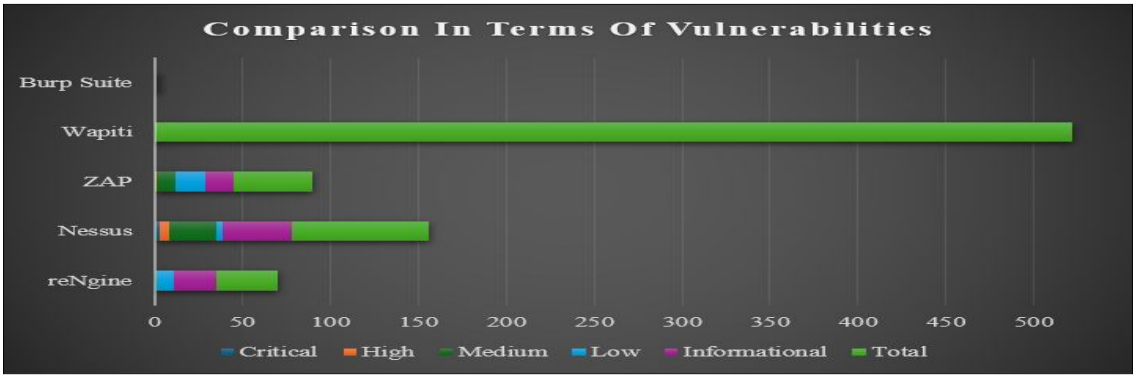

Figure 2 : Types of Vulnerabilities Detected By Each Tool

6.1.2 Ease of Use and Integration

Second, evaluation criteria is ease of use. reNgine (2.2.0) is an advanced and modern tool with web-based interface. It is easy to install with couple of dependencies. It is specifically designed for reconnaissance and vulnerability management. Its dashboard is very user friendly. You can find the vulnerabilities details with URLs and targets on the main dashboard with brief details. It has marvellous integration capabilities with tools like, Nmap, Metasploit, Nikto etc. With the help of these integration capabilities, it is also efficient in port scanning and gathering information and present details about the technologies used in the scanned web applications. This distinct ability differentiates reNgine to ither tools. It stores all the information regarding target web-application on premises which ensure confidentiality. There are several tools for reconnaissance but reNgine provides data correlation because mostly available options are commercial. It is best in terms of continuous monitoring as well; you can configure it for regular scanning and generating alerts in case of threat detection. There are multiple scanner options available in reNgine, you can select as per your need.

Nessus (10.8.3) provides web-based scanning interface. It also has a simplified GUI and is easy to use. You can add multiple at one time. It provides limited options in essential version i.e. basic network scan, advanced scan, web application tests etc. It also provides support for API integration. You cannot ensure confidentiality because data is stored on Nessus server which is not on premises and mostly companies do not prefer it.

ZAP (2.15.0) also provides GUI based interface which is very simple and advanced. It provides both manual testing and automatic testing options as per your need. By default, it performs passive scanning using proxy and analyzes the web applications by scanning the messages between the browser and application. You can just add your desired URL, and traditional spider will scan the application and vulnerabilities can be examined through alert tab in the scanner. It also provides active scanning options, but you must know manual testing expertise, but it does not provide reconnaissance and asst discovery features means you must perform separate scans with other tools and analyze manually for data correlation.

Wapiti (3.0.4) is a command line interface and not user friendly. It creates many hurdles while working with latest Python versions. Mostly it works well with an old version of Python and some other dependencies. So, in this research Wapiti was not working on Kali Linux due to its upgradation and latest version of Python-3.12. So, I installed windows subsystem for Linux (WSL) on host system to perform testing then I installed Wapiti in WSL and performed testing. Wapiti took half hour to scan WordPress. It took two days to perform scanning of OWASP-BWA web application but noy succeeded. Then I quit the scanning process and generated the report. After that instead of main URL I tried to scan couple of applications in OWASP-BWA project, then Wapiti provided some results, and I saved the reports. So, Wapiti can be used but with limited scope and applications. It also does not provide deep integration options.

Burp Suite (2024.8.5) is an advanced user interface and moderate friendly with users and has integration feature as well. It provides exceptional features for manual testing which requires some expertise in manual testing as well. Add target IP or URL of web application is not as straight forward like other scanners. Unfortunately, community edition does not

provide automatic scanning features. So, no scanning is performed with Burp Suite because manual testing is time consuming at this stage of research.

6.1.3 Performance and Speed

Table 3 : Performance and Speed Analysis

Feature	reNgine	Nessus	OWASP ZAP	Wapiti	Burp Suite
Performance	Moderate	Comprehensive	Good	Basic	NA
Resources	Low	Moderate	High	Low	NA
Speed of Scan	2 Hours	2 Hours	5 Minutes	3 Days	NA

reNgine is mainly focused on reconnaissance and asset discovery including web application vulnerability detection and management and its performance depends on tools use for enumeration subdomains, open ports and services i.e. Nmap, Nikto etc. and you will face moderate speed while scanning web applications and it also depends on the complexity of web application. It took more than 2 hours to complete the scan of both applications as shown in Table 3 above. It consumes less system resources.

While on the hand, Nessus is highly optimized for dep scanning and it used skimming techniques to scan unchanged assets which saves time. It has many plugins support over 50,000 for this reason it consumes extra system resources. Its speed is slow while scanning the complex environments. Nessus took 2 hours to complete the scan of both applications.

OWASP ZAP is efficient in Dynamic Application Security Testing (DAST). It provides fast automatic scanning of web applications. It speed is faster than reNgine and Nessus, but it consumes more system resources. It took only 5 minutes to scan both applications.

In case Wapiti, no doubt it is a lightweight web application scanner, but it provides limited coverage due to command line interface. It only performs well with older versions of Python and does not support latest versions. For this reason, I could not use it in Kali Linux, and I installed (WSL) on host machine to test the applications. It is extremely slow and poor while testing the main URL of OWASP-BWA. It took half an hour to test WordPress and almost three days to test the OWASP-BWA and still was not finished so I quit. But it is efficient to test small or a specific web application.

Burp Suite is exceptional in manual testing which is not performed due to limited time and automatic testing is not enabled in community edition. So, no further details can explain about performance and speed.

6.1.4 Reporting and Documentation

reNgine and Nessus both provide good visually reports in the format of PDF. Both provide the name and details about the identified vulnerabilities and severity level as well. They also provide the CVSS scores and remediation steps against the vulnerabilities. reNgine also

provides information about subdomains and Open Ports while Nessus requires another type of scan for these services. For both tools detailed documentation is available.

OWASP ZAP provides basic level of reporting focusing on web vulnerabilities based on risk level. Report format is HTML. It also provides details about alerts tags, detail descriptions, related information and solutions. It also provides references for the report about the known vulnerabilities. ZAP also gives information about the CWE ID and WASC ID of detected vulnerabilities. Documentation and tutorials are available for the OWASP ZAP.

While Wapiti provides documentation about the vulnerabilities in HTML format. It has minimal reporting capabilities. Report tells us about the category of vulnerabilities and number of found vulnerabilities. It provides little information about hosts and cookies as well. It has limited official documentation. Finally, Burp Suite provides reports in the format of PDF and HTML including CWE references and remediation steps too. Because it has not been tested on, no further details are available.

6.1.5 Community and Support

Table 4 : Community Support and Official Support

Feature	reNgine	Nessus	OWASP ZAP	Wapiti	Burp Suite
Official Support	No	Licensed Users	No	No	Licensed Users

reNgine is an open-source project and does not have official support while all discussions and supports are available through GitHub community support. Contributions and bugs are welcomed and fixed by the developers. On Contrary, Nessus has professional support available for the licensed version. It also has tenable community support to provide quick resolution of common issues. There are regular updates available for plugins. Furthermore, OWASP ZAP is also an open-source project and provides no official support, but it has large GitHub community for support to common issues and bugs. In addition to this, there are annual events organized by OWAP team where they directly interact with the users. Moreover, Wapiti does not provide any support but only a limited GitHub support is available with minimal official documents. Finally, Burp Suite provides official support for the licensed version only.

6.1.6 Compliance and Security Standards

Table 5 : Compliance and Security Standards

Feature	reNgine	Nessus	OWASP ZAP	Wapiti	Burp Suite
NIST Standards	Minimal	Extensive	Limited	No	Minimal
OWASP Top 10	Integrations	Built-in Plugins	Core Feature	Basic	Advanced
ISO 27001 Support	Limited	Pre-defined	Partial	Limited	Limited

reNgine is not completely following the National Institute of Standards and Technology (NIST) or framework while there is a minimal alignment and has no pre-defined compliance templates for reporting. It supports and follows OWASP-Top 10 indirectly by aggregating the data from integrating tools. It has a limited support for information security Management standard (ISO-27001).

While Nessus provides extensive support for NIST and OWASP-Top 10 security standards. It has pre-defined polices to find vulnerabilities as per NIST standards and OWASP Top 10 security standards. It also provides pre-defined ISO templates for compliance.

OWASP ZAP is not directly mapping to NIST controls while it has built-in features to detect and mitigate vulnerabilities related to OWASP-Top 10 and can provide detail reporting against them. It does not provide direct support for ISO standards and requires addition tools to cover the ISO 27001 standards.

Finally, Wapiti and Burp Suite do not support NIST security standards. On the other hand, Wapiti provides minimal, and Burp Suite provides exceptional coverage related to OWASP-Top 10 vulnerabilities. Wapiti does not have any pre-defined templates for auditing while Burp Suite provides limited support for compliance.

6.1.7 Cost and Licensing

Table 6 : Cost and Licensing

Feature	**reNgine**	**Nessus**	**OWASP ZAP**	**Wapiti**	**Burp Suite**
Cost / Year	Free	5,191.73 Euros / Year[32]	Free	Free	$449 (Pro), $3,999 (Ent)[33]
Licensing	General Public License [34]	Proprietary	Contributor License Agreement [35]	General Public License	Proprietary

reNgine, OWASP ZAP and Wapiti are fully free and available for both personal and commercial use. reNgine and Wapiti are under General Public License (GPL) allows free usage and modifications while OWASP ZAP is licensed under contributor License Agreement. Nessus essentials is free but limited usage and scan maximum up to 16 IP addresses and professional is cost up to 5,191.73 Euros per year while Burp Suite community

[32] Tenable Multiproduct
[33] Burp Suite Professional - PortSwigger
[34] reNgine 2.0 Redefining the future of reconnaissance!! - reNgine
[35] ZAP – ZAP Contributor License Agreement

edition only provides manual evaluation and the cost for professional and enterprise edition is from $449 to $3,999 per year.

7 Recommended Tool

Based on the results and evaluations, it is strongly recommended reNgine, an open-source reconnaissance and asset discovery framework which is freely available and easy to use, having moderate speed for scanning the applications, providing data correlation, offering consistent reports and minimal compliance standards, having community support for fixing the bugs and issues, finding vulnerabilities in web applications and WordPress websites.

8 Discussion and Conclusion

Five open-source vulnerability scanners i.e. reNgine, Nessus, OWASP ZAP, Wapiti and Burp Suite were tested to find the vulnerabilities on OWASP-BWA and WordPress website. reNgine and OWASP ZAP found 35 and 45 total number of vulnerabilities while Nessus performed comparatively well with 78 found vulnerabilities. Furthermore, Wapiti results were unexpected with 522 vulnerabilities which may cause some technical issues or may be considered as it has lack of severity categorization when compared to other tools. It looks like all issues are listed without prioritization of risk level. All tools evaluated on basis of found vulnerabilities, ease of use and integration, speed, documentation, support and cost. I found reNgine is best with some features like data correlation in addition to some advanced features i.e. asset discovery, continuous monitoring of assets, smart alerts, reconnaissance and customizable audit report generation. There were some limitations with AWS-EC2 instance in terms of scanning. We could not perform vulnerability testing to cloud environment because it needs approval from cloud service provider. In addition to this, some scanners were not tested thoroughly due to limited versions and features. There were some configuration and technical issues with Wapiti and reNgine while viewing the results which could not be sorted due to time constraints.

In conclusion, the objective was to evaluate the performance of open-source vulnerability scanners for web applications and WordPress websites and proposed the best available free tool with advanced features. Tools were tested and proposed the reNgine as best one based on objectives including free, easy to use, address the OWASP-Top 10 vulnerabilities, provide detail documentation, having smart alerts and continuous asset monitoring ability and data correlation.

For future research work, it is recommended to evaluate the scanners with full versions including capabilities to find cloud related vulnerabilities and prior approval must be got to perform testing on the cloud. In addition, it is also recommended not to use the main URL of a project with multiple applications i.e. OWASP-BWA, in fact only specific applications should be tested so that results should be confined and proper. Furthermore, try to test the real live web applications with proper prior approval.

9 References

Abdullah, H.S. (2020) ‘Evaluation of Open Source Web Application Vulnerability Scanners’, *Academic Journal of Nawroz University*, 9(1), p. 47. Available at: https://doi.org/10.25007/ajnu.v9n1a532.

Alsaleh, M. *et al.* (2017) ‘Performance-Based Comparative Assessment of Open Source Web Vulnerability Scanners’, *Security and Communication Networks*, 2017, pp. 1–14. Available at: https://doi.org/10.1155/2017/6158107.

Alya Geogiana Buja *et al.* (2024) ‘Analysis of Web Vulnerability Using Open-Source Scanners on Different Types of Small Entrepreneur Web Applications in Malaysia’, *Journal of Advanced Research in Applied Sciences and Engineering Technology*, 40(1), pp. 174–188. Available at: https://doi.org/10.37934/araset.40.1.174188.

Chaturvedi, A. *et al.* (2024) ‘A Comprehensive Vulnerability Tools Analysis for Security and Control in IT Environment and Organizations’, in *2024 5th International Conference on Electronics and Sustainable Communication Systems (ICESC)*. *2024 5th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pp. 612–618. Available at: https://doi.org/10.1109/ICESC60852.2024.10689860.

Cruz, D.B., Almeida, J.R. and Oliveira, J.L. (2023) ‘Open Source Solutions for Vulnerability Assessment: A Comparative Analysis’, *IEEE Access*, 11, pp. 100234–100255. Available at: https://doi.org/10.1109/ACCESS.2023.3315595.

Elshheibia, T.A., Mohamed, M.I. and Almahdi, A.M. (2024) ‘WEB Applications Vulnerability Analysis and prevention’, *مجلة جامعة بني وليد للعلوم الإنسانية والتطبيقية*, pp. 519–531. Available at: https://doi.org/10.58916/jhas.v9i1.219.

Fadlil, A., Riadi, I. and 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), pp. 635–645. Available at: https://doi.org/10.5829/IJE.2024.37.04A.06.

Gajula, M.R. and Vassilakis, V.G. (2024) ‘Evaluating the Performance Open-Source Vulnerability Scanners’, in *2024 14th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*. *2024 14th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*, Rome, Italy: IEEE, pp. 377–382. Available at: https://doi.org/10.1109/CSNDSP60683.2024.10636578.

Hossain Hadi, M. and Hashim Al-Saedi, K. (2024) ‘Adaptive Hybrid Learning for Websites Vulnerability prediction’, *Journal of Al-Qadisiyah for Computer Science and Mathematics*, 16(1). Available at: https://doi.org/10.29304/jqcsm.2024.16.11433.

Jimmy, F. (no date) ‘Cyber security Vulnerabilities and Remediation Through Cloud Security Tools’.

Kyer, M.A. (no date) ‘J-WAVE: A Java Web Application for Vulnerability Education’.

Makino, Y. and Klyuev, V. (2015) ‘Evaluation of web vulnerability scanners’, in *2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*. *2015 IEEE 8th International Conference*

on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Warsaw, Poland: IEEE, pp. 399–402. Available at: https://doi.org/10.1109/IDAACS.2015.7340766.

Matti, E. (no date) 'Evaluation of open source web vulnerability scanners and their techniques used to find SQL in- jection and cross-site scripting vulnerabilities'.

Onukrane, A. *et al.* (2023) 'Navigating Web Application Security: A Survey of Vulnerabilities and Detection Solutions', in *2023 IEEE 64th International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*. *2023 IEEE 64th International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*, Riga, Latvia: IEEE, pp. 1–6. Available at: https://doi.org/10.1109/ITMS59786.2023.10317708.

Pradhana, B.S. (2024) 'Website Security Analysis Using the OWASP10 Method (Case Study: almumtazparfumebatam.store)', 8(1).

Premchand, P. *et al.* (no date) 'Vulnerability Scanner Build a Tool That Scans a System forPotential Vulnerability'.

Sagar, D. *et al*. (2018) 'STUDYING OPEN SOURCE VULNERABILITY SCANNERS FOR VULNERABILITIES IN WEB APPLICATIONS', *COMPUTER SCIENCE*, 9.

Samgir, A.B. *et al.* (2024) 'Automated Penetration Testing Architecture Using Metasploit and OWASP ZAP for Web Applications', in *2024 2nd International Conference on Sustainable Computing and Smart Systems (ICSCSS)*. *2024 2nd International Conference on Sustainable Computing and Smart Systems (ICSCSS)*, pp. 649–657. Available at: https://doi.org/10.1109/ICSCSS60660.2024.10625033.

Shaikh, M.I. and Lokhande, P.S. (2024) 'Tackling Threats: A Study of Vulnerability Testing and Mitigation in Web Applications', *SSRN Electronic Journal* [Preprint]. Available at: https://doi.org/10.2139/ssrn.4823623.

Shivananjappa, N. and Creutzburg, R. (2024) 'Vulnerability Management Using Open-Source Tools', *Electronic Imaging*, 36(3), pp. 326-1-326–8. Available at: https://doi.org/10.2352/EI.2024.36.3.MOBMU-326.

Sllame, A.M., Tomia, T.E. and Rahuma, R.M. (2024) '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 *2024 IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA)*. *2024 IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA)*, pp. 68–75. Available at: https://doi.org/10.1109/MI-STA61267.2024.10599685.

Vaish, A. *et al.* (2024) 'Development of Cyber Security Platform for Experiential Learning', *Journal of Cybersecurity Education, Research and Practice*, 2024(1). Available at: https://doi.org/10.62915/2472-2707.1184.

Vemula, S.R. (2024) 'Automating Security Testing: Strategies for Vulnerability Scanning, Penetration Testing, and Compliance Checks', 11(07).