# Performance Evaluation of Automated Web vulnerability scanners for cross platforms -Red Teaming

MSc Research Project
MSs cyber security

Rohan Anand Gowda
Student ID: X21178003

School of Computing
National College of Ireland

Supervisor: Mr. Nail Heffernan

***National College of Ireland***

***Project Submission Sheet – 2022/2023***

| | | | |
|---|---|---|---|
| **Student Name:** | Rohan Anand Gowda | | |
| **Student ID:** | X21178003 | | |
| **Programme:** | MSCCYB1 | **Year:** | 1 |
| **Module:** | MSc. Research project | | |
| **Lecturer:** | | | |
| **Submission Due Date:** | 18/09/2023 | | |
| **Project Title:** | Performance Evaluation of Automated Web vulnerability scanners for cross platforms -Red Teaming | | |
| **Word Count:** | 7775 including references | | |

| | |
|---|---|
| **Signature:** | Rohan Anand Gowda |
| **Date:** | 18/09/2023 |

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

# Performance Evaluation of Automated Web vulnerability scanners for cross platforms -Red Teaming

Rohan Anand Gowda
X21178003
MSc. Cybersecurity–MSCCYB1
National College of Ireland

**Abstract**

The rapid technical advancement of current times has produced favorable worldwide development. The digitization of private data in the modern day increases the likelihood that someone's information will be compromised as technology develops. The need of doing security checks is significantly increasing as cyber threats grow. The most important element in this infrastructure security procedure is the penetration testing of networks and web applications. With this growth comes a larger need for assessment, benchmarking, and evaluation of the tools and processes penetration testers utilize. We explore some of the most popular automated web penetration testing tools available today and use benchmarking techniques to evaluate them for vulnerabilities in this study. In order to have more robust and accurate results and to give more scope by specifying tools on various operating systems, the research's conclusion will be to identify several vulnerabilities utilizing multiple automated scanning tools and determine where each one's strengths lie. The research's conclusion will be to uncover several vulnerabilities using various automated scanning tools and pinpoint where each tool's strengths lie in order to achieve more thorough and accurate results.

**Keywords: Penetration testing, automated scanning tools, vulnerabilities.**

## 1. Introduction

A staggering 84% of code bases have at least one open-source weakness, according to a Forbes study on cybersecurity trends for 2023. The survey found that 73% of all code bases from businesses in the transportation, logistics, auto, aerospace, and aviation industries contained some opensource code. The paper suggests patching and penetration testing as options. (Chuck Brooks, 2023)

The subsequent FORTRA report graphic demonstrates why penetration testing is so crucial.

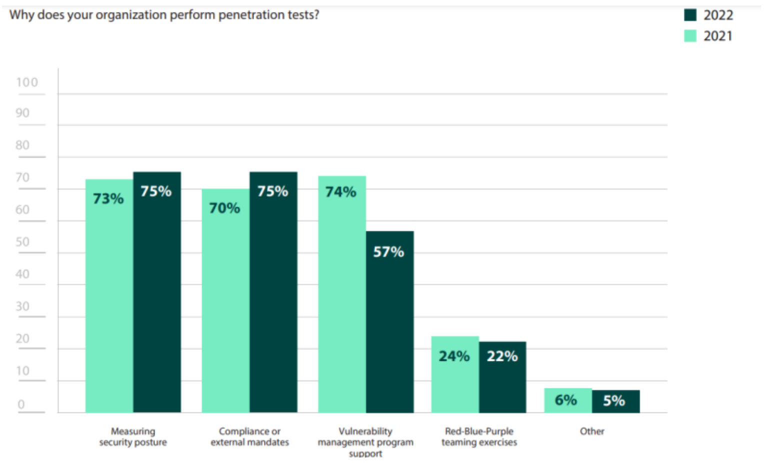


***Fig 1. Importance of a Pentest*** (Fortra, 2022)

Since a penetration test is a legal activity, it varies significantly from hacking. The customer or business starts the process by passing laws and drafting agreements that the tester must accept. The assessor then completes the task and creates a report listing every vulnerability that has to be patched in order of priority based on severity. The tester can do this test manually or with the aid of automated tools. The manual procedure is entirely carried out by the tester, takes more time, and is more effective in direct proportion to the tester's level of experience and skill. It also has the limitation that it cannot be repeated more than once due to its difficulty, whereas automated penetration testing tools have the capacity to crawl entire domains and subdomains of a web application, enumerating a large scope. Since it is a software program, the procedure can be run again with similar findings and fewer false positives.

There are many different automated pen-testing programs available, each with a unique set of attributes that are selected according on the task at hand. The most well-known ones rely on static and dynamic scanning, where login and other authentication factors are taken into account to assess the website's operation and flaws while it is active. The most important factor is the tool's effectiveness, which can be assessed using benchmarking. Many services, such Invicti, Rapid Vulnscan, OpenVas, StackHawk for DevSec Ops, and Tenable.io, take this into account.(Chad Kime, 2023) The most vital ones being Burpsuite Professional(Portswigger, 2023), Accunetix(Invicti, 2023a), Nikto and Recon-ng which are operable on Linux and windows based on their support packages. The automated tool is selected in the penetrating testing field based on its ability to recognize important vulnerabilities and interface orientation. This has motivated the hunt for an effective automated penetration testing tool that satisfies the needs of the market right now. This thesis aims to develop a framework for rating automated penetration testing tools. We demonstrate the model's efficacy and utility using examples from automated penetration testing tools. In addition to vulnerability enumeration, this study broadens the scope of evaluation and provides a ranking system that takes into account other variables for penetration testing tool settings. It also adds the factor of tools that can be operable on a non-Linux platform such as windows which still has growing scope in current times. The research will be conducted using statistical analysis of the data gathered by penetration testing equipment.

Research Question:

• To identify efficient automated penetration testing tools that would satisfy cross-platform target user demands.

Objective:

• Create a framework to assess the tools' capabilities

• To display statistical analyses of penetration testing tool outcomes.

The structure of the research thesis will be as follows: The literature and research in question are reviewed in Section 2, and the research methods and framework are discussed in Section 3. After that, parts 4 and 5 will cover the framework's needs, a tool analysis, and a case study. More information and a conclusion are provided in sections 6 and 7. These will be an expansion of the current report's description.

# 2. Related work

The subject that begins this chapter focuses on the related research being done with penetration testing and automated testing methods. It goes over some of the research' findings and explains how penetration testing works. The next sections cover several automated penetration testing tools and unique methodologies that are employed in this area of research.

## 2.1 Penetration test

The overview of the literature that introduces penetration testing is the first section of the thesis. In agreement with (Xiaohong Yuan *et al.*, 2011) penetration testing is a technique used by experts in computing systems to address vulnerability assessment problems, with a focus on high-severity flaws. The article explains three principal testing strategies namely black box, white box and gray box testing. It claims that these testing strategies are applied on three types of infrastructure that are network based where the tester will attempt to identify vulnerabilities that could be exploited by an attacker to gain access to the network, application based that focuses on applications where the tester will attempt to identify vulnerabilities that could be exploited by an attacker to gain access to the application or steal data and social engineering testing that focuses on human interaction where the tester will attempt to exploit human vulnerabilities, such as a lack of security awareness, to gain access to an organization's systems or data. It stated that the four important stages of the process were recon, scanning, exploitation and reporting and concluded by stating that penetration testing is an important tool for improving the security of an organization's systems and data. It can help to identify vulnerabilities that could be exploited by attackers. However, penetration testing is not a silver bullet. It is important to follow up on the results of the penetration testing and take steps to mitigate any vulnerabilities that were found.

- Manual Penetration testing: The article by (Nagpure and Kurkure, 2017) claims that while Vulnerability assessment can be automated, but penetration testing cannot, and the subtle explanation for this is based on how the human mind behaves in each circumstance. Cascaded intelligence is needed to take action and exploit vulnerability. There is a growing body of literature on the importance of manual vulnerability scanning for web applications. A 2018 study by the SANS Institute found that 60% of web applications have at least one critical vulnerability that can be exploited by attackers. The study also found that automated tools are only able to find about 40% of these vulnerabilities. This suggests that manual scanning is essential for finding and fixing the most critical vulnerabilities in web applications. Another study, published in the Journal of Information Security and Applications in 2019, found that manual scanning can be used to identify vulnerabilities that are not found by automated tools. The study also found that manual scanning can be used to verify the design, business logic, and code of a web application. Penetration testers use the OWASP guide to carry out the process in accordance with the standard (OWASP, 2023b).

- Automated penetration testing: This method has many differences from manual testing. (Stefinko, Piskozub and Banakh, 2016) demonstrated that a group of knowledgeable security specialists and developers (tiger-team) regularly produce commercial-grade automated penetration testing solutions after conducting extensive vulnerability research, establishing secure, modern exploits, and compiling them into a straightforward, user-friendly package. A comprehensive image of an organization's security posture may be produced by an automated penetration testing system by exhaustively testing across networks, endpoints, online apps, and email users. These programs also classify vulnerabilities according to their severity using OWASP guidelines. (OWASP, 2023b)
- The ability of automated penetration testing systems to execute scans using numerous connections (threads) to the online application gives them an advantage over manual penetration testing. Automated scanners make it possible for penetration testers to look at more of the program. Automated systems, in contrast have high degree at detecting false positives (Michael Massoth, Saed Alavi and Niklas Bessler, 2018). By tweaking the application for certain websites to optimize results, this problem can be rectified.. The figure 2.1 gives a representation of the test.

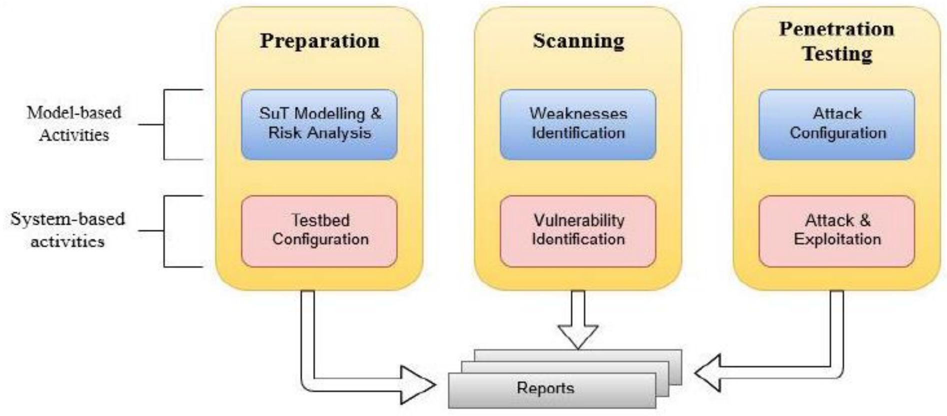
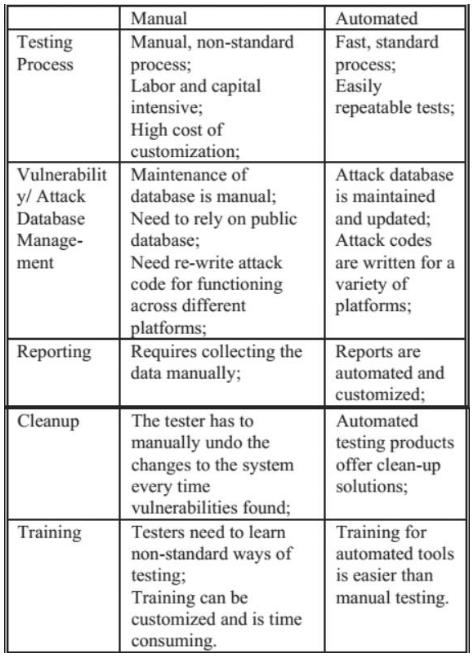


*Figure 2.1*: *Automated penetration testing*(Casola *et al.*, 2018)



| | Manual | Automated |
|---|---|---|
| Testing Process | Manual, non-standard process; Labor and capital intensive; High cost of customization; | Fast, standard process; Easily repeatable tests; |
| Vulnerability/ Attack Database Manage-ment | Maintenance of database is manual; Need to rely on public database; Need re-write attack code for functioning across different platforms; | Attack database is maintained and updated; Attack codes are written for a variety of platforms; |
| Reporting | Requires collecting the data manually; | Reports are automated and customized; |
| Cleanup | The tester has to manually undo the changes to the system every time vulnerabilities found; | Automated testing products offer clean-up solutions; |
| Training | Testers need to learn non-standard ways of testing; Training can be customized and is time consuming. | Training for automated tools is easier than manual testing. |

*Figure 2.2: Differences between automated and manual testing* (Stefinko, Piskozub and Banakh, 2016)

## 2.2 Penetration testing methodologies

Three general categories can be used to group penetration test techniques:

- Grey Box Test,
- White Box Test
- Black Box Test

### 2.2.1 Black Box test:

Black box testing treats the application as if it were a "Black Box" with no awareness of the environment it operates in, focusing exclusively on the critical parts of the system. When performing a black box test, a tester must be acquainted with the system architecture and won't have access to the source code.(Mohd. Ehmer Khan and Farmeena Khan, 2012)

### 2.2.2 Grey Box test:

The grey box testing approach will increase analysis scope through enabling us to narrow in upon every aspect of any complex system, incorporating all current white box and black box testing. The examiner must be conversant with internal data structures and algorithms for the purpose of developing test cases during grey box testing.(Mohd. Ehmer Khan and Farmeena Khan, 2012)

### 2.2.3 White Box test:

The "white box testing" approach for creating scenarios for testing draws inspiration from the procedural design's control structure. White box testing examines a piece of software's internal operations and structure to find implementation issues such inadequate key management. The system, unit, and integration layers of the application evaluation procedure may all utilize white box testing. The tester must examine the source code to determine which component of program is having trouble during white box testing.(Mohd. Ehmer Khan and Farmeena Khan, 2012)

## 2.3 Types of testing

### 2.3.1 Static application security testing:

SAST stands for Static Application Security Testing. It is a software testing technique that 8nalyses the source code of an application to identify potential security vulnerabilities. According to (Francesc Mateo Tudela *et al.*, 2020). SAST tools perform a white box security analysis, which means that they analyze the source code of an application to identify potential vulnerabilities. This can be done by looking for specific patterns or keywords in the code, or by using more sophisticated techniques such as abstract interpretation or taint analysis. The text cites a number of studies that have shown that SAST tools can be effective in detecting security vulnerabilities. For example, a study by (Díaz and Bermejo, 2013) found that SAST tools were able to detect 70% of the security vulnerabilities in a sample of web applications. SAST tools can be a valuable asset for security teams, but they are not perfect. One of the challenges with SAST tools is that they can generate a large number of false positives. Another challenge with SAST tools is that they can only analyze the source code of an application. This means that they cannot detect vulnerabilities that are introduced by the runtime environment or by configuration errors. They can be used to supplement other security analysis techniques, such as dynamic application security testing (DAST) and manual code review.

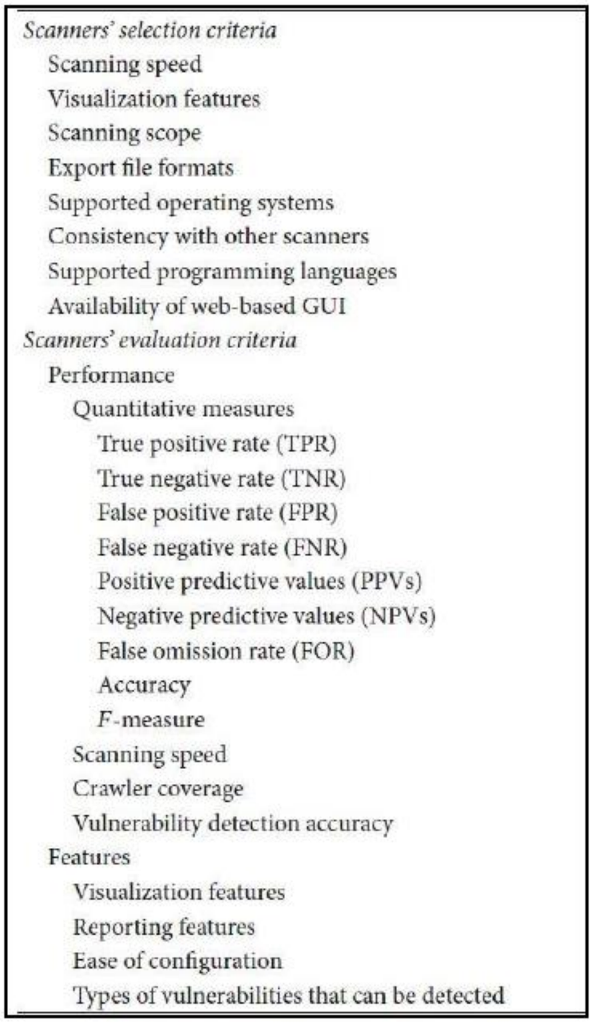
### 2.3.2 Dynamic application security testing:

A literature review on work conducted by (Francesc Mateo Tudela *et al.*, 2020)DAST tools are black-box tools that analyze a running web application for vulnerabilities. They do this by sending malicious inputs to the application and observing the response. DAST tools typically work in three phases: discovery, attack, and analysis.DAST tools have a number of advantages over other types of security testing tools, including the ability to test live web applications, the ease of use, and the ability to be automated. However, DAST tools also have some disadvantages, such as the potential for false positives, the slowness and resource-intensiveness, and the limited scope of testing. DAST tools should be used in conjunction with other types of security testing tools, such as SAST and IAST, to ensure that all vulnerabilities are found and fixed. Additional information on surveying also suggests that DAST tools are constantly evolving to incorporate new authentication methods, attack vectors, and techniques to automate the detection of vulnerabilities. DAST tools can be used to detect a wide range of vulnerabilities, including SQL injection, cross-site scripting (XSS), and file upload attacks. The results of DAST scans should be manually reviewed to identify false positives and to ensure that all vulnerabilities are properly addressed. DAST tools are a valuable tool for security testing, but they should not be used as the only tool for security testing.

## 2.4 Evaluation of vulnerability scanning tools

There are two important articles selected as the base for our analysis and are derived from the papers below. They had a concise description of analysis on tools used both on windows and linux. Web application scanners and automated tool evaluation are gaining popularity in university circles and the cyber security sector. There are many benchmark programs available for evaluating web security scanners. In 2017, (Mansour Alsaleh, Noura Alomar and Monirah Alshreef, 2017) conducted a rigorous evaluation of four open-source web application vulnerability scanners: Skipfish, Arachni, Wapiti, and IronWASP. The authors compared the scanners based on performance criteria such as scanning speed and crawling inspection. The results of the study showed that there were significant discrepancies in the results of the scanners in terms of scope. This might be due to how well each scanning tool performs. For example, Skipfish was one of the

scanners that finished scans the fastest, although it had less crawler coverage than Arachni, which took longer. In contrast to the average crawler coverage of 48%, Arachni achieved 94% scanning coverage.
The thesis also aimed to create a framework for scanner evaluation based on the latest requirements and vulnerabilities in the field of cyber security. The purpose of this framework is to help organizations evaluate the effectiveness of vulnerability scanners and to select the right scanner for their needs. The evaluation criteria as per the article suggests the following components:



*Scanners' selection criteria*
- Scanning speed
- Visualization features
- Scanning scope
- Export file formats
- Supported operating systems
- Consistency with other scanners
- Supported programming languages
- Availability of web-based GUI

*Scanners' evaluation criteria*
- Performance
  - Quantitative measures
    - True positive rate (TPR)
    - True negative rate (TNR)
    - False positive rate (FPR)
    - False negative rate (FNR)
    - Positive predictive values (PPVs)
    - Negative predictive values (NPVs)
    - False omission rate (FOR)
    - Accuracy
    - *F*-measure
  - Scanning speed
  - Crawler coverage
  - Vulnerability detection accuracy
- Features
  - Visualization features
  - Reporting features
  - Ease of configuration
  - Types of vulnerabilities that can be detected

***Figure 2.3: Evaluation matrix*** (Mansour Alsaleh, Noura Alomar and Monirah Alshreef, 2017)

The key findings of the study were that there is no single “best” vulnerability scanner. The best scanner for a particular organization will depend on its specific needs and requirements. Scanners should be evaluated based on a variety of criteria, including scanning speed, crawling inspection, detection accuracy, and crawler coverage. The results of scanner evaluations should be used to select the right scanner for a particular organization. The thesis provides a valuable contribution to the field of cyber security by providing a framework for scanner evaluation and by comparing the main characteristics and capacities of the scanners. The framework can be used by organizations to evaluate the effectiveness of vulnerability scanners and to select the right scanner for their needs.

Personally, its understood that the study highlights the importance of evaluating vulnerability scanners before using them. Not all scanners are created equal, and some may be more effective than others at detecting specific types of vulnerabilities. The study also highlights the importance of using multiple scanners. By using multiple scanners, organizations can increase their chances of detecting all of the vulnerabilities in their web applications. The study is a valuable resource for organizations that are looking to improve the security of their web applications. The framework for scanner evaluation can be used to select the right scanner for a particular organization, and the comparison of scanner characteristics and capacities can help organizations to understand the strengths and weaknesses of different scanners.

The paper by (Chanchala Joshi and Umesh Kumar Singh) focuses on the growing security concerns associated with the proliferation of internet usage, specifically in relation to web applications. The authors carry out an in-depth study to understand how effective web application vulnerability scanners are in addressing security concerns. In the introduction, the authors articulate the broad objectives of the paper: 1) to evaluate the effectiveness of web application vulnerability scanners, 2) to identify some common types of vulnerabilities in web environments, and 3) to propose defence measures to significantly secure the application. In the "Experimental Details" section, the authors discuss their plans for carrying out the research, which includes the following five steps: designing a web application with vulnerabilities, selecting vulnerability scanners, scanning the application to identify potential vulnerabilities, manually verifying the identified vulnerabilities, and analyzing the results. They outline some existing web applications that demonstrate common vulnerabilities, and explain why they have decided to design their own ("shopatujjain") for testing purposes. The authors describe the main functionalities of their application, which replicates the process of a user interacting with a regular, dynamic e-commerce-type web page.

This paper includes a methodology section, focusing on the technical details of the "shopatujjain" Web Application, such as its use of MySQL, PHP, HTML, CSS, JavaScript, and AJAX technologies.
In the "Observations" section, the authors ask pertinent questions about the false-positive rate of the scanners and the most common types of vulnerabilities observed. The results of the paper suggest that the vulnerability scanners – Acunetix and Netsparker – performed quite well in detecting some vulnerabilities, particularly Cross-Site Scripting (XSS) but had variable performance in detecting others such as SQL Injection and Security Misconfiguration. The authors discuss how these results can guide future research to enhance the scanner's detection rates and effectiveness.
In conclusion, the paper delivers on its objective of evaluating the efficiency of web application vulnerability scanners and contributing to the body of knowledge needed to enhance the security measures of web-based applications.

## 2.4 Benchmarking

The paper by (Mburano and Si, 2018)discusses the effectiveness of two open-source web vulnerability scanner tools, Arachni and ZAP. These tools are evaluated using the widely respected Open Web Application Security Project (OWASP) benchmark and their findings are then compared with existing results from the Web Application Vulnerability Security Evaluation Project (WAVSEP) benchmark to highlight the need for using multiple benchmarks for a comprehensive evaluation. The researchers highlight the importance of benchmarking web vulnerability scanners, especially due to their differing performance levels and widespread usage. They have identified a literature gap where different benchmark results have not been compared and fill this gap by comparing OWASP and WAVSEP. Three benchmark tools, WIVET, WAVSEP, and OWASP are explored in the related work section. For the study, they've chosen Arachni and ZAP due to their frequent updates, number of contributors, and popularity. They also demonstrated that Arachni and ZAP have specific advantages that make them ideal for such a study. The authors established a methodology to select and evaluate the chosen scanners by creating a local network for the analysis, wherein

OWASP Benchmark, OWASP ZAP, and Arachni were installed to carry out the assessment, and discuss their process. They evaluated the scanners based on true positives, false positives, true negatives, and false negatives. The experiment resulted in Arachni performing better for certain vulnerability category detections while ZAP excelled in others. Upon comparing these results with a previous study that used the WAVSEP benchmark, the authors noted discrepancies suggesting the need for multiple benchmarks to comprehensively evaluate web vulnerability scanners' efficacy. The paper is first to compare these two reputable benchmarks in the literature, hence providing valuable recommendations for the practice of benchmarking web scanners.

### 2.5 Motivation for conducting the research

The thesis researched upon is an addition to performance and comparative analysis of penetration testing tools that have been carried out previously. The papers surveyed in the section of the related work mostly focus on comparing tools which are majorly used on kali Linux. A majority of beginners in the field of penetration testing are not familiar with the environment of Linux based systems whereas windows is one of the most used operating systems out in the market and is easily operable. Moreover, windows is very suitable to host GUI based applications which is easier to use and interact with. Another factor is that most of the tools are open source and hence enterprise tools are not generally studied upon which are crucial for real time analysis. The aim of this research is to analyse the performance of such tools that are easily available and accessible on windows as GUI tools and tools with considerably non-complex commands that do not require advanced experience to operate on a CLI based environment. The research also aims to explain how modern tools make it easier for testers in all experience levels by tools that can be operated on different platforms increasing the landscape for deploying penetration testing tools. This study was carried out to establish, present, and show the effectiveness of the tools under investigation. This will clarify which tool to use in which circumstances and show how using both platforms together can increase the amount of vulnerability identification.

## 3. Research Method & Specification

In this section, we go over the research process for the framework used to analyse web application security scanners and penetration test results. Figure below depicts the approach:

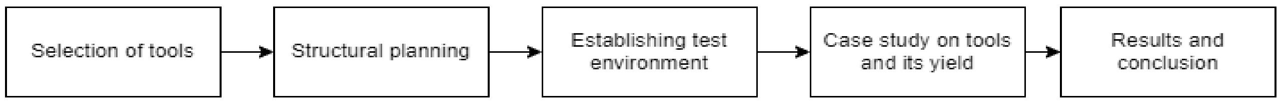

Selection of tools
Structural planning
Establishing test environment
Case study on tools and its yield
Results and conclusion

*Figure 3: Methodology*

### 3.1 Selection of Tools:

The choice of research tools will be the first step in the investigation. The intention is to choose and assess widely used tools in the field of cyber security. We take into consideration well-known solutions for application security testing such as Burpsuite professional, Netsparker, HCL Appscan, Nikto where the scope of scanners is narrowed down to two commonly used open-source and two commercial tools. The tools will be separated into two group namely tools for proxies and scanners. The proxy tools include scanning tools built right into them and can also be used as web proxies. The target request-response interaction can be controlled more precisely with this kind of tool. Additionally, it makes it very simple for penetration testers to scan post-login requests. The proxy component enables manual penetration testing by testers. In Point and Shoot (PaS) arrangement, the scanner tools—standalone scanners—are utilized to carry out automatic scans.

| Tool used | Version Info | License | Tool type | Price (per annum) |
|---|---|---|---|---|
| Netsparker | 6.3.0 | INVICTI | Scanner | USD 1499 |
| HCL appscan | 10.2.1 | HCL technologies Ltd | Scanner | USD 1199 |
| Burpsuite Professional | 2023.8.7 | Portswigger | Proxy | USD 449 |
| Nikto | 2.5.0 | N/A | Scanner | N/A |

*Table 3.1: Selected tools*

**3.1.1 Netsparker:** The Netsparker tool is easy to use, it has a great user interface, and it does a respectable job at identifying the most critical vulnerabilities(Invicti, 2023c). It's got good reporting features that are logically constructed and simple to read. Additionally, it provides the capability to confirm discovered vulnerabilities. This functionality may spare the tester a lot of time because they don't have to verify the weaknesses that were previously reported by Netsparker.

**3.1.2 HCL Appscan standard:** For developers, DevOps, security teams, and CISOs, HCL AppScan is a comprehensive suite of application security solutions with on-premises, in the cloud, and hybrid deployment choices (HCL Technologies, 2023). It provides a range of functions, such as the following, to assist enterprises in locating and addressing application vulnerabilities: Application vulnerabilities including SQL injection, cross-site scripting, and buffer overflows are scanned for during dynamic application security testing (DAST). In order to find vulnerabilities that DAST might have missed, interactive application security testing (IAST) keeps track of programs as they are created and tested. Analyzes application code for flaws such buffer overflows, format string flaws, and unsafe coding techniques. Static application security testing (SAST). Open-source security scanning: Checks open source dependencies for flaws to help organizations reduce the risk posed by known flaws in outside programs.

**3.1.3 Burpsuite Pro:** Burp Suite Professional is the most advanced version of Burp Suite, the world's leading web application security testing toolkit (Portswigger, 2023). It includes all of the features of Burp

Suite Community Edition, plus a number of additional features that make it ideal for professional penetration testers and security analysts.

### 3.1.4 Nikto:

Nikto is a free and open-source web server scanner written in Perl. It is designed to scan web servers for vulnerabilities.('Nikto 2', 2023) Nikto can detect over 6700 potentially dangerous files/CGIs, checks for outdated versions of over 1250 servers, and version specific problems on over 270 servers. It also checks for server configuration items such as the presence of multiple index files and HTTP server options, and will attempt to identify installed web servers and software. Scan items and plugins are frequently updated and can be automatically updated.

## 3.2 Structural planning

The criteria used to evaluate the tool based on a variety of aspects are covered in this section. These elements involve variables that are essential for defining the tool's effectiveness. The evaluation criteria are based on the following variables, which provide some insight:

1.) Type of tool: CLI or GUI: the two main types of tools used in penetration testing: GUI-based tools and CLI-based tools. GUI-based tools have a graphical user interface, which makes them easier to use for beginners. However, they can be slower than CLI-based tools because they have to render the graphical elements. CLI-based tools have a command-line interface, which requires users to type commands. This can be more difficult for beginners, but CLI-based tools are often faster and more powerful than GUI-based tools. The markings for this parameter will be taken as 1 and 2 points for CLI and GUI respectively.

| **Feature** | **GUI-based tools** | **CLI-based tools** |
|---|---|---|
| **Ease of use** | Easier for beginners | More difficult for beginners |
| **Speed** | Slower | Faster |
| **Power** | Less powerful | More powerful |
| **Cost** | Often more expensive | Often less expensive |

*Table 3.2.1: Features of CLI and GUI tools*

2.) Type of testing: Black box testing is a type of penetration test where the tester has no prior knowledge of the target application. This means that the tester cannot login to the application or use any of its features that require authentication. As a result, early penetration test automated tools were only able to identify vulnerabilities that could be exploited without authentication. Modern penetration test automated tools are more advanced and can handle session cookies. Session cookies are used to authenticate users and track their sessions in a web application. This means that modern penetration test automated tools can be used to perform authenticated scans. Authenticated scans can identify vulnerabilities that require authentication, such as SQL injection and cross-site scripting vulnerabilities. In addition to handling session cookies, modern penetration test automated tools can also detect irregularities in application code. This means that they can identify vulnerabilities that are not directly related to authentication, such as buffer overflows and denial-of-service vulnerabilities. The points awarded for this parameter is 1,2 and 3 for black box, black box and grey box and black, grey and white box testing.

3.) Speed of scan: automated tools developed by penetration testers can scan a large web application much faster than humans can. This is because automated tools can perform repetitive tasks quickly and without getting tired. As a result, automated tools can cover a greater area of a web application in less time. The time taken to scan a web application is important for scanner evolution because it determines how many vulnerabilities can be found. If a scanner takes too long to scan a web application, then it may not be able to find all of the vulnerabilities. This is why scanner developers are constantly working to improve the speed of their scanners. Grade points for this parameter are 1,2,3,4,5 for 6 hrs, 3 hrs, 2 hrs, 1hr and 30 mins respectively.

4.) Crawling features: Crawling is the first step in most web application security tests. It involves mapping the application by navigating through it request by request. The crawler catalogues the links found during the process, and these links are later used for scanning purposes. There are two types of crawling: active crawling and passive crawling. Active crawling interacts with the application and sends requests to the application server to get the active links. Passive crawling works silently without actively engaging the application. The passive crawler works while manually navigating through the application. Passive crawling can cover more links and paths in the application, proving a greater coverage. This is because the passive crawler is not limited by the application's API. It can navigate to any link that is visible to the user, even if it is not explicitly defined in the application's API. Grading for this parameter is 1 point for active crawler support and 2 points for active and passive crawler support.

5.) Crawling coverage: The goal of a penetration tester is to learn as much as they can about a web program, including its design, functionality, and security flaws. One way to do this is to use an automated scanner to crawl the web application. This scanner will visit all of the URLs on the application, and it will also try to follow all of the links. This will help the penetration tester to map out the application and to identify any potential vulnerabilities. It is important that the automated scanner covers as much of the application as possible without manual intervention. This is because manual crawling can be time-consuming and error-prone. If the scanner only crawls a small part of the application, then the penetration tester may miss important information. The crawler coverage can be measured by the number of URLs that the scanner has crawled. A high crawler coverage means that the scanner has visited a large number of URLs on the application. This is a good indication that the scanner has gathered a lot of information about the application. Grading for this parameter would be from 1-5 for 10%, 25%, 50%, 75% and more than 75%.

6.) Type of scan: When it comes to application scanning, there are two distinct scan categories such as passive scan and active scan. The passive scan does not transmit more information to the application server; instead, it 15nalyses requests and responses acquired by manual crawling or interaction by the penetration tester. Passive scanning includes locating problems like incorrectly configured SSL certificates and missing response security headers. Additionally, client side Java Script analysis is included. By connecting with the web application server, the active scan stage might identify the presence of vulnerabilities. The scanner attempts to exploit the vulnerabilities by sending tailored requests to the server. The security tool will look for widespread vulnerabilities like those in the OWASP Top 10 list. Grading: Active scan-1, active and passive scan-2

7.) Rate of vulnerability detection: Ratio of the detected vulnerabilities by the scanner to the actual vulnerabilities present in the benchmark multiplied by hundred.

$$\text{Rate of detection} = \frac{\text{Number of vulnerabilities detected by the scanner}}{\text{Number of vulnerabilities present in a benchmark application}} * 100.$$

Grading for this parameter would be from 1-5 for 10%-25%, 25-50%, 50-75% and more than 75%.

8.) Usability: Usability is the factor of convenience and ease of use towards users of the tool. The easier and better the interface the more is its usability. Grading: Easy configuration- 2 Complex configuration – 1.

9.) False positives and true positive rate: False positives are vulnerabilities that the scanner flagged as present but were not in the application. It is preferred to use a scanner that produces fewer false positive results(Antunes and Vieira, 2015). A true positive is a vulnerability that the scanner correctly identifies and also reports. It is always preferred to have a high true positive rate. Precision rate is another name for it (Josephine S Akosa, 2017). Grading: TPR-> 1-4 for 10%, 25%, 50% and above 50% and vice versa for FPR ie 4-1 for 0-10%, 10-25%, 25-50% and above 50%.

| True Positive rate | False Positive rate |
|---|---|
| $\frac{True\ Potive}{True\ Positive+False\ Negative} * 100$ | $\frac{False\ Positive}{False\ Positive+True\ Negative} * 100$ |

These values are found by the benchmarking tool as given below:

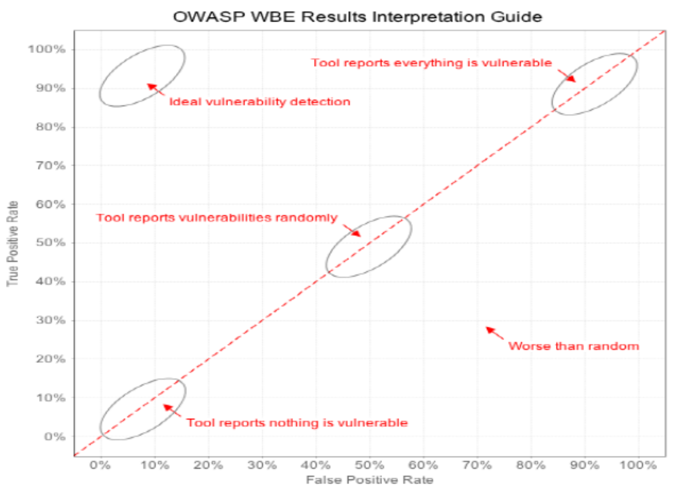


*Figure 3.2: Owasp interpretation guide*(OWASP, 2023a)

10.) In-app features: These are factors that are offered additionally by the tool adding to an increase in its functionality. The more a tool can offer the better the convenience for the tester. Grading: Additional 1 point for unique features.

11.) Reporting features: Regardless of the report's structure, automated programs that generate reports for security scanning frequently contain a few standard components. These components include: An executive summary that offers a high-level overview of the scan results, including the number of vulnerabilities found, their severity, and their effects. A list of vulnerabilities found, often including the name, impact, and description of each vulnerability. A list of requests and responses that demonstrates the HTTP requests and answers that were sent throughout the scan. Debugging vulnerabilities may be aided by this. Information about the scanned target, including the operating system, hostname, and IP address. Grade: 1 point for only xml or one type of reporting and 2 points for multiple reporting formats.

12.) Cost: cost of a tool is an important factor to consider when selecting it. There are many tools available that have similar functionalities and features, but the cost can vary significantly depending on the brand. Some commercial tools are very expensive, while there are also many freeware tools that offer great performance and features. It is also important to consider the long-term cost of a tool. Some freeware tools may require more maintenance or have fewer features than commercial tools. Grading: Open-source – 3, 500USD and below- 2, above 500usd – 1.

# 4. Implementation

## 4.1 Establishing test environment

This section deals with the requirements needed to carry out our analysis. Since we are performing an analysis on cross platforms namely Windows 11 and Kali Linux, we require them as operating sysems installed on our systems and hence use Windows 11 as our base system and kali Linux as a virtual machine hosted by VirtualBox. The reason VirtualBox is chosen over VMWare is because of convenient snapshot features and the fact that Vmware is a commercial tool whereas Vbox is free to use (InterviewBit, 2023). The tools for analysis are then installed on their respective systems i.e. HCL appscan and Netsparker on Windows and burpsuite professional and nikto on kali Linux. The results are then extracted from the report generated that might be in formats like .xml or .scan depending on the formats supported by the tools. The values are then analysed statistically and the results are discussed based on the performance yield of the tools. For benchmarking we use two distinct targets namely OWASP juice shop: a web application designed by owasp and made vulnerable on purpose for testers to learn how vulnerabilities could be found on a website and as for our linux tools we use the OWASP benchmark which is a benchmarking tool that gives us the scoring for each tool.

## 4.2 Testing Approach

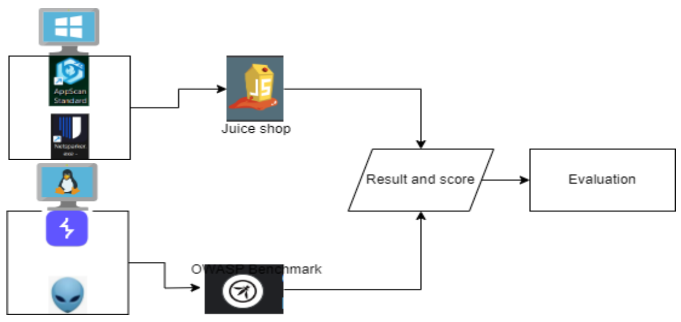


Figure 4.1: Evaluation process

The tools once installed onto the respective systems are tested by using their benchmarks as shown above. The juice shop application by owasp is used for the Windows GUI tools since it is one of the most modern vulnerable web applications designed and is also challenging to find the weaknesses as opposed to older ones like altoro mutual and dvwa where the vulnerabilities were easily obtained. A point to add to that is that it poses as a real-world challenge to the commercial tools which can help us identify the functionality in a precise manner. The Linux tools are tested on the owasp benchmark which has similar vulnerabilities to juice shop and also because the benchmark creates the scorecard based on which the tool can be evaluated in a more convenient manner. The tools are then graded based on the criteria of marking as defined in the structural planning phase. The total a tool can be graded for is 40.

# 5. Evaluation

This section deals with the output of the tools on the benchmark namely OWASP juice shop and benchmark for Windows and Linux espectively. The outputs help us analyze the efficiency of the tools.

## 5.1 Case Study 1: Analysing Windows based tools:

The two tools namely HCL Appscan and netsparker were used to scan the vulnerable application and were analysed as follows:

| Parameter | HCL Appscan score | Netsparker score |
|---|---|---|
| Type of tool | 2 | 2 |
| Type of testing | 3 | 3 |
| Speed of scan | 3 | 3 |
| Crawling features | 2 | 2 |
| Crawling Coverage | 4 | 4 |
| Type of scan | 2 | 2 |
| Rate of detection | 3 | 3 |
| Usability | 2 | 2 |
| True positive rate | 2 | 1 |
| False positive rate | 4 | 4 |
| In-app features | 1 | 1 |
| Reporting features | 2 | 2 |
| Cost | 1 | 1 |
| Total score | 31/40 | 30/40 |

The two tools came very close to almost producing the same results. Both tools scored full points for being a convenient GUI tool. The speed of scan could be varied for each tool by optimizing the scans but both tools comfortably finished the scans in 39 and 40 mins for netsparker and hcl appscan respectively. Both the tools support both active and passive crawling and score full points. The crawling coverage on both are good with the tools being able to scan more than 50% but miss out as the application has a login page which had to be authenticated. Both tools were capable of performing active and passive scans. Both applications have inn app features such as scan optimization and login configuration. Both tools had a good vulnerability rate detection with appscan being (69/101)*100= 68% and netsparker with (52/101)*100=52%. Netsparker also has a feature where the entire website could be opened inside the tool giving it a higher functionality. The false positive rate for both the scanners are low with netsparker having only 0.1% FPR detection rate (Invicti, 2023b)whereas hcl having a 10% FPR detection rate. The true positive rate for HCL was found to be

4/(4+10)*100 w.r.t owasp top 10 was around 28% while netsparker was found to be 1/(1+10)*100 and was around 9%. Appscan had the upper hand as it detected vulnerabilities related to SQLi, Security misconfiguration and insecure deserialization. Both tools have multiple reporting formats and could be customised according to enterprise or private standards that help for compliance and pentest audits. The cost on both tools is on the pricey side where they score just 1 point.

## 5.1 Case Study 2: Analyzing Linux based tools:

The scoring for the tools are as follows:

| Parameter | Burpsuite score | Nikto score |
|---|---|---|
| Type of tool | 2 | 1 |
| Type of testing | 3 | 1 |
| Speed of scan | 4 | 4 |
| Crawling features | 2 | 1 |
| Crawling Coverage | 3 | 1 |
| Type of scan | 1 | 1 |
| Rate of detection | 1 | 1 |
| Usability | 2 | 1 |
| True positive rate | 1 | N/A |
| False positive rate | 4 | N/A |
| In-app features | 1 | - |
| Reporting features | 2 | 2 |
| Cost | 2 | 1 |
| Total score | 28/40 | 14/40 |

The tools once configured had problems relating to connecting to the benchmark. The error generated was because burpsuite pro is a proxy tool meant for realtime web applications and might not work the same with benchmarks. Therefore, we use (Albahar, Alansari and Jurcut, 2022) where a similar analysis was done. H estates that 5% command injection, 8% cross-site scripting, 3% unsecure cookies, 4% LDAP injection, 3% path traversal, 8% SQL injection, and 7% XPath injection were all covered by Burp Suite Professional. 70% of the OWASP Top 10 are covered by the scanner and says the TP value for Burp Suite Professional is 56, 56/3235 100 = 1.73%.. Burpsuite is a GUI proxy scanner and nikto is a CLI based scanner therefore scoring 2 and 1 respectively. Burpsuite comes with an efficient pricing whereas Nikto is opensource and both tools score well for reporting features as they support csv, xml, etc. The rate of detection on both tools were low and less that 10%. This is because burpsuite is very efficient when it’s used to intercept the traffic while manually crawling and helps the tester to not miss important details. Burpsuite also has many inbuilt features and has a store for adding extensions that are very useful. Nikto on the other hand is less resource consuming and one of the oldest tools with good credibility. Burpsuite helps in active and passive scanning whereas nikto has only active scan options and the crawling coverage depends on the application as to how complex it is.

# 6. Discussion

The research conducted gives an idea as to how much more efficient commercial tools are compared to open-source tools. HCL appscan and netsparker came very close and differed just by 1 point. On the other hand burpsuite scored decently well to the windows tools. While nikto couldn’t connect to the target. The study clearly proves that commercial tools though consume more system resources produces a considerable efficient result with lesser true positives. Another factor to observe in the netsparker tool was that the time of scan could be varied by varying he requests sent hence increasing the crawler coverage if there were to be a larger sub domain to be scanned. Both Appscan and netsparker had login configuration options for white and grey box testing support. This factor helps in authenticating login pages before scans are configured therefore helping to increase the coverage as well. HCL was he only scanner to detect a top 10 vulnerability i.e. SQLi based weakness and hence the scanner could be reputed to have he best performance in the analysis. Netsparker then takes over with second place for having very helpful features with burpsuite following he list. Burpsuite also has a bunch of helpful features and the tools efficiency is raised when the applications are crawled manually Burpsuite could also be run on windows adding to its usability. Nikto comes in last since the scans couldn’t be configured but is the best option considered for information gathering rather than recon. The evaluation matrix designed helps find out each aspect of the tool in detail and hence leaves a smaller room for errors. The related works usually focus on open-source tools and vulnerabilities detected whereas the thesis covers factors like reporting features, costing and crawling study adding to the accuracy for analysis.

## 7. Conclusion and future work.

The objective of the research was to find the efficiency and performance of tools that could be used on windows and Linux. The research aims to have solved the question by proposing a detailed evaluation matrix which showed the strengths and weaknesses of each tool in a real-time setup due to the use of a deployed web application. It shows how well commercial tools fare against open-source tools and explains how important it is for using a scanner based on circumstance could increase the output and detection rate. Hence it is suggested to use multiple scanners to increase the detection rate. The research also added the factor of using multiple benchmarks as suggested by previous works and hence increased the scope of analysis,. Future scope includes: 1) Evaluation of newer commercial tools, 2) Evaluation based on benchmarks other than owasp benchmark, 3) An evaluation on a more modern real-time application compared to OWASP Juice shop.

## References

Albahar, M., Alansari, D. and Jurcut, A. (2022) 'An Empirical Comparison of Pen-Testing Tools for Detecting Web App Vulnerabilities', *Electronics*, 11(19), p. 2991. Available at: https://doi.org/10.3390/electronics11192991.

Antunes, N. and Vieira, M. (2015) 'On the Metrics for Benchmarking Vulnerability Detection Tools', in *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, pp. 505–516. Available at: https://doi.org/10.1109/DSN.2015.30.

Casola, V. *et al.* (2018) 'Towards Automated Penetration Testing for Cloud Applications', in *2018 IEEE 27th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*. IEEE, pp. 24–29. Available at: https://doi.org/10.1109/WETICE.2018.00012.

Chad Kime (2023) *The 8 Best Vulnerability Scanner Tools for 2023.* Available at: https://www.esecurityplanet.com/networks/vulnerability-scanning-tools/#invicti (Accessed: 16 April 2023).

Chanchala Joshi and Umesh Kumar Singh (no date) 'Performance Evaluation of Web Application Security Scanners for More Effective Defense', *International Journal of Scientific and Research* [Preprint], (2016). Available at: https://d1wqtxts1xzle7.cloudfront.net/47370836/ijsrp-p5490-libre.pdf?1468998289=&response-content-disposition=inline%3B+filename%3DPerformance_Evaluation_of_Web_Applicatio.pdf&Expires=1691166593&Signature=OtVU~bjZpaPhoLCaLnJ3UFbAAdjJa2qcAHw3RSQdRQFn6gWwhx2ffxnQOVeXwYNe1H2WnUJBlcFzalTLs2qD2Hn3Odx7ooNeKGo9cqbUZqRjMeEtC-LyE2rlcHa~0zzmhQYjOBmIgtolfTyIERI15thTuTtffTt5zn5r~OfzongRrm25pFYKOxlicwfJZAWRVmw4Genm5mxBz~MZwwM5VcyfpLIE2v7Wnsnhu2K-QI2Fsnoledv3t7ZM0S4-T32kfMG8Vjc~84qVMZcHRYUdsGFwBJ5J5cALPa81Os9zVidzuzKqOMxf1MCXlOWuuWFaaVTL-QR8BuTGJzxm6SAjVg__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA (Accessed: 4 August 2023).

Chuck Brooks (2023) *Cybersecurity Trends & Statistics For 2023; What You Need To Know.* Available at: https://www.forbes.com/sites/forbesdigitalcovers/2018/07/19/the-inside-story-of-papa-johns-toxic-culture/? (Accessed: 16 April 2023).

Díaz, G. and Bermejo, J.R. (2013) 'Static analysis of source code security: Assessment of tools against SAMATE tests', *Information and Software Technology*, 55(8), pp. 1462–1476. Available at: https://doi.org/10.1016/j.infsof.2013.02.005.

Fortra (2022) '2022 Penetration Testing Report', pp. 4–4. Available at: https://static.fortra.com/core-security/pdfs/guides/cs-2022-pen-testing-report.pdf (Accessed: 16 April 2023).

Francesc Mateo Tudela *et al.* (2020) 'On Combining Static, Dynamic and Interactive Analysis Security Testing Tools to Improve OWASP Top Ten Security Vulnerability Detection in Web Applications', *MDPI* [Preprint]. Available at: https://www.mdpi.com/2076-3417/10/24/9119 (Accessed: 9 August 2023).

HCL Technologies (2023) 'HCL Appscan'. Available at: https://www.hcltech.com/brochures/software/hcl-appscan-standard#:~:text=HCLTech%20AppScan%20Standard%20is%20a,web%20applications%20and%20web%20services. (Accessed: 11 August 2023).

InterviewBit (2023) *VMware vs VirtualBox: What's The Difference?* Available at: https://www.interviewbit.com/blog/vmware-vs-virtualbox/ (Accessed: 11 August 2023).

Invicti (2023a) *Accunetix, 2023*. Available at: https://www.acunetix.com/ (Accessed: 16 April 2023).

Invicti (2023b) *False Positive rate detection - Netsparker*. Available at: https://www.invicti.com/blog/news/comparison-web-vulnerability-scanners-netsparker-2013-

2014/#:~:text=False%20Positives%20and%20Web%20Security%20Scans%20Time%20Consumption&text=Funnily%20enough%20Netsparker%2C%20the%20only,false%20positive%20SQL%20Injection%20vulnerabilities. (Accessed: 14 August 2023).

Invicti (2023c) ‘Netsparker’. Available at: https://www.invicti.com/web-vulnerability-scanner/ (Accessed: 10 August 2023).

Josephine S Akosa (2017) ‘Predictive Accuracy: A Misleading Performance Measure for Highly Imbalanced Data ’. Available at: https://support.sas.com/resources/papers/proceedings17/0942-2017.pdf (Accessed: 12 August 2023).

Mansour Alsaleh, Noura Alomar and Monirah Alshreef (2017) ‘Performance-Based Comparative Assessment of Open Source Web Vulnerability Scanners’. Available at: https://www.hindawi.com/journals/scn/2017/6158107/ (Accessed: 16 April 2023).

Mburano, B. and Si, W. (2018) ‘Evaluation of Web Vulnerability Scanners Based on OWASP Benchmark’, in *2018 26th International Conference on Systems Engineering (ICSEng)*. IEEE, pp. 1–6. Available at: https://doi.org/10.1109/ICSENG.2018.8638176.

Michael Massoth, Saed Alavi and Niklas Bessler (2018) ‘A Comparative Evaluation of Automated Vulnerability Scans versus Manual Penetration Tests on False-negative Errors’. Available at: http://personales.upv.es/thinkmind/dl/conferences/cyber/cyber_2018/cyber_2018_1_10_80034.pdf (Accessed: 16 April 2023).

Mohd. Ehmer Khan and Farmeena Khan (2012) ‘A Comparative Study of White Box, Black Box and Grey Box Testing Techniques’, *(IJACSA) International Journal of Advanced Computer Science and Applications*, 3. Available at: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=ab0ed151c010e03e2d34284ae5f89756372e7690#page=22 (Accessed: 16 April 2023).

Nagpure, S. and Kurkure, S. (2017) ‘Vulnerability Assessment and Penetration Testing of Web Application’, in *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*. IEEE, pp. 1–6. Available at: https://doi.org/10.1109/ICCUBEA.2017.8463920.

‘Nikto 2’ (2023). Available at: https://cirt.net/Nikto2 (Accessed: 11 August 2023).

OWASP (2023a) *Owasp interpretation guide*. Available at: https://owasp.org/www-project-benchmark/ (Accessed: 12 August 2023).

OWASP (2023b) *OWASP Web Security Testing Guide*. Available at: https://owasp.org/www-project-web-security-testing-guide/ (Accessed: 16 April 2023).

Portswigger (2023) *Burpsuite*. Available at: https://portswigger.net/burp (Accessed: 16 April 2023).

Stefinko, Y., Piskozub, A. and Banakh, R. (2016) ‘Manual and automated penetration testing. Benefits and drawbacks. Modern tendency’, in *2016 13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET)*. IEEE, pp. 488–491. Available at: https://doi.org/10.1109/TCSET.2016.7452095.

Xiaohong Yuan *et al.* (2011) ‘An Overview of Penetration Testing’. Available at: https://www.researchgate.net/publication/274174058_An_Overview_of_Penetration_Testing (Accessed: 16 April 2023).