Comparative Analysis of the Automated Penetration Testing Tools

MSc Internship

Cybersecurity

Mandar Prashant Shah

Student ID: x18139469

School of Computing

National College of Ireland

Supervisor: Dr. Muhammad Iqbal

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Mandar Prashant Shah		
Student ID:	X18139469		
Programme:	MSc Cybersecurity	**Year:**	2019
Module:	Internship Thesis		
Supervisor:	Dr Muhammad Iqbal		
Submission Due Date:	08/01/2020		
Project Title:	Comparative analysis of the automated penetration testing tools		
Word Count:	**8573 Page Count 25**		

Comparative Analysis of the Automated Penetration Testing Tools

Mandar Prashant Shah

X18139469

Abstract

The requirement of performing security audit is growing day by day as the cyber threat is increasing. One of the key components in this process of securing the network is to perform penetration test of the network and web applications. With this growing need there is also a growing need of standardisation/benchmarking in the processes followed and tools used by penetration testers. In this research based thesis we look at the modern day automated web penetration testing tools and compare them with industry known OWASP Benchmark for vulnerabilities. We also address the lack of literature for framework which evaluates scanners with 360-degree view. To evaluate scanner, we performed two case studies with 4 scanners.

Our research shows that, scanners with web proxy and configured crawling perform better as compared to point and shoot scanners. It was also observed that scanners with active maintenance life cycle performed better. The conclusion drawn from this research is, to detect multiple vulnerabilities more than one automated scanning tools should be used. This gives more reliable results.

1 Introduction

According to the Internet Security Threat Report 2019 by Symantec there was 56% growth observed in web based attacks in 2018, with on an average 30 to 40 million attacks detected per month[1]. In the recent years web application exploitation have been used excessively against internet based applications. The penetration testing is a process of simulating cyber-attacks against the target system. The penetration test is a controlled process of penetrating into the network or web application environment in order to identify the vulnerabilities [1].

The difference between a hacker and a penetration tester is that the penetration test is carried out with the permission, signed contract and licences. At the end of the penetration test a report with all the observations and vulnerabilities is prepared and presented to the client. The penetration test can be performed manually in which a professional penetration tester assesses the network environment or the web application for the known vulnerabilities. The manual penetration test lacks repeatability and is largely dependent on the testers skills to identify the flaws in the application. The automated penetration test on the other hand is performed using software tools which analyses the application for vulnerabilities. The automated penetration testing tools have repeatability in their analysis, and it can produce consistent results for a target system or network. Penetration test has become an essential and critical activity not only in the information technology (IT) industry but all the industries which have online presence.

[1] Symantec Internet Security Threat Report Volume 24, February 2019: https://www.symantec.com/content/dam/symantec/docs/reports/istr-24-2019-en.pdf

The are many automated penetration testing tools available with different capabilities and qualities. The specifications which these automated penetration testing tools have vast range which is from scanning a simple single page web application to scanning a complex enterprise level multi-layered application with multiple workflows. To measure the effectiveness of the web security scanners, Benchmarking is one of the processes. There are well known benchmarks available for web vulnerability scanners, such as Web Input Vector Extractor Teaser (WIVET) [2], Web Application Vulnerability Scanner Evaluation Project (WAVSEP) benchmark [3]. In the Penetration testing industry, the automated tool is selected based on its ability to detect critical vulnerabilities and ease of usability factor. This has given the motivation to identify the efficient automated penetration testing tool which suffice the current day trends in the industry.

This research was conducted with an objective of developing a framework to compare the automated penetration testing tools. We demonstrate the frameworks efficiency and usability using automated penetration testing tool case studies. The research framework is based on the previous research done by the Mayur Turuvekere and Anala A. Pandit proposing automated penetration testing tools evaluation based on vulnerabilities identified [4]. This research widens the evaluation matrix and provides a broader scale which includes not only vulnerability detection, but also other parameters of the penetration testing tools. The research is performed using statistical investigations of the outcomes obtained from the penetration testing tools.

Research Question:

- To identify efficient automated penetration testing tool to suffice the current day industry requirement.

Objective:

- Develop Framework to compare the web applications penetration testing tools.
- To do the research based comparative analysis of automated penetration testing tools on recent trends in the industry.
- To demonstrate statistical investigations of the outcomes obtained from the penetration testing tools.

The research paper is presented as follows, Section 2 covers the literature review and related work of our research, Section 3 discusses the research methodology and framework design. Afterwards, section 4 and 5 discusses specifications of the framework, evaluation and case study of automated penetration testing tools. Further discussion and conclusion are presented in Section 6 and 7.

2 Related Work

The initial discussion in this chapter focuses on the related work in the area of penetration testing and automated testing tools. It talks about some of the research work and describes penetration testing processes. The rest of the chapter talks about different automated penetration testing tools and discussion of various approached taken in this area of research.

2.1 Penetration test

Our research started with the literature review of penetration testing methodologies. A penetration test or PEN Test is a process of identifying security vulnerabilities in the system. A vulnerability is a software or hardware loophole in a system which may allow unintended use of the system [5]. As discussed by Alireza Nowroozi, Mahin Mirjalili, Mitra Alidoosti in [6], the penetration test use for has increased significantly by large corporations to secure their information systems and services. This allows organisations to fix security issues before it is exploited. There are two ways to conduct a penetration test: 1) Manual Penetration test, 2) Automated Penetration test.

2.1.1 Manual Penetration Test:

Ideally Manual penetration test is conducted by an experienced and skilled professional who can control all the parameters during the testing. On a System under test (SuT) manual penetration tester analyses all the entry points of the application and performs tests to identify potential vulnerabilities. As discussed in [7], some vulnerabilities are difficult to find and can be found only by manual penetration test. These penetration testers consider custom logic of each application which may or may not be identified by an automated scanner. Based on the business logic of the application penetration testers develop test cases for the security testing. As suggested in [5], a penetration tester should posses' comprehensive knowledge of web application vulnerabilities and detection methods. It also says that the coverage of such a test is largely dependent on the penetration testers skills and experience. Penetration testers follow industry accepted standards to perform penetration test, such as OWASP [8]. The shortfall of this process is at times it is slow and cumbersome process. It is also a costly process as hiring an experienced penetration tester requires lot of money.

2.1.2 Automated Penetration Test:

An automated penetration testing is a technique in which a software is used to scan the application request by request [7]. The scanner application actively interacts with the target application server and analyses each request and response for known vulnerabilities. The scanner also injects various payloads in the request to detect vulnerabilities in the application. There are tools which are designed only for the identification of single vulnerability which have higher accuracy rate. But using multiple individual tools to scan complete application is not always viable option and it also requires deeper understanding of vulnerabilities, which a professional penetration tester has. Modern day web vulnerability scanners address this issue. These automated tools provide a complete framework of tools to which are required to test the application. Such tools make use of industry approved standards such as OWASP TOP 10 [8] and National Vulnerability Database by NIST[2].

The automated penetration testing tools have advantage over manual penetration test that they can perform scan using multiple connection (threads) to the web application. Automated scanners aide penetration testers to cover larger area of the application. On the other hand, automated tools suffer from large amount of false-positive detection rate [9]. This shortfall can be addressed by configuring the tool for individual web application, to obtain better results. The figure 2.1 shows the automated penetration testing process.

[2] National Institute of Standards and Technology: https://nvd.nist.gov/

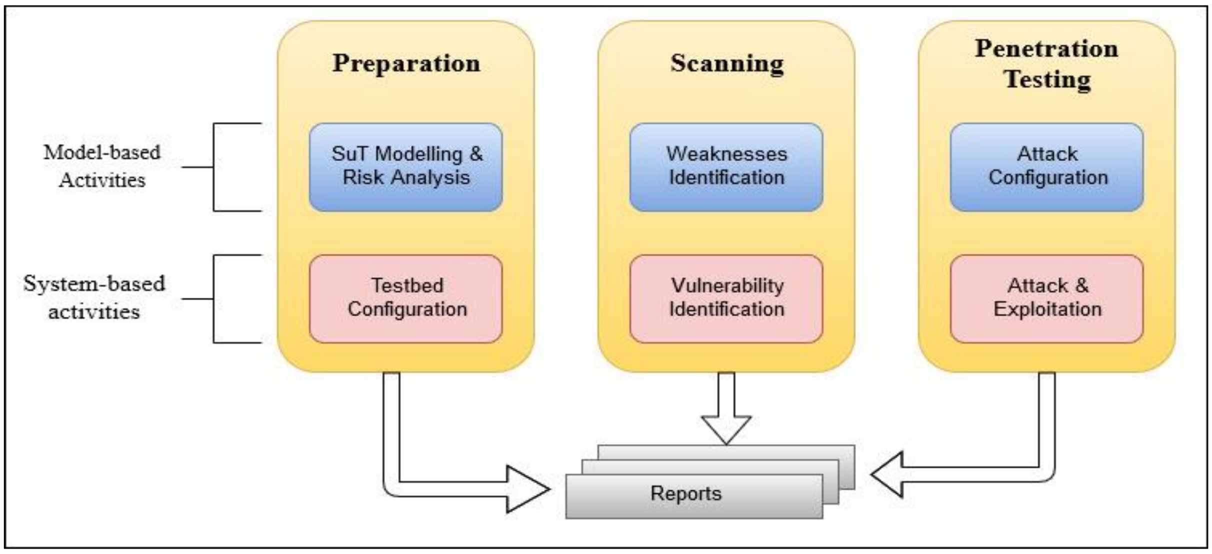


Figure 2.1: Automated Penetration test process [10]

Following table compares manual penetration test and automated penetration test in brief,

	Manual	Automated
Testing Procedure	• Work-Intensive, inconsistent and prone to error, without any specific quality standards. • Requires a lot of different tools. • The results may vary considerably from test to test. • Generally, the running and interpretation of tests requires highly paid, skilled security people.	• Quick, simple and secure. • Eliminates errors and tedious manual tasks. • Centralized and standardized to produce reliable and repeatable findings. • Easy to operate and offers clear, actionable reports.
Network Modification	Tester might unintentionally modify the system.	Systems remain the same.
Exploit Expansion and Management	• It takes time to develop and manage an exploit database and requires substantial expertise. • Public exploits are suspicious and can be risky to run. • For cross-platform functionality, re-writing and porting code is required.	• All exploits are developed and maintained by the software vendor. • Exploits for maximum effectiveness are continuously updated. • Professional exploits are developed, thoroughly tested and safe to run. • Exploits for a variety of platforms and attack vectors are written and optimized.
Clean up	• All changes must be remembered and undone by the tester.	• Leading scanners deliver thorough cleaning and never install backdoors.

	• It is possible to leave backdoors behind.	
Reporting	• Requires a great deal of time, manually tracking and gathering all data. • All reports can be customized as per client's requirement.	• Comprehensive history and documentation of results are generated automatically. • Customizable files.
Logging/Reviewing	• Tester has to enable logging for each test and tool that he performs.	• Reports all events in detail automatically.
Training	• Testers need to know non-standardized forms of ad-hoc testing.	• Users can know as small as one day training and install it.

Table 1.1: Comparison Manual and Automated Penetration test [10]

2.2 Penetration test Methodology

The penetration test approaches can be categorised broadly in three types:

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

2.2.1 Black Box test:

This type of test is also known as 'Zero Knowledge' test as the tester has no knowledge of the application and underlying infrastructure [11]. It depends only on the information obtained from the web application. Most of the automated scanners are designed to perform black box test.

2.2.2 Grey Box test:

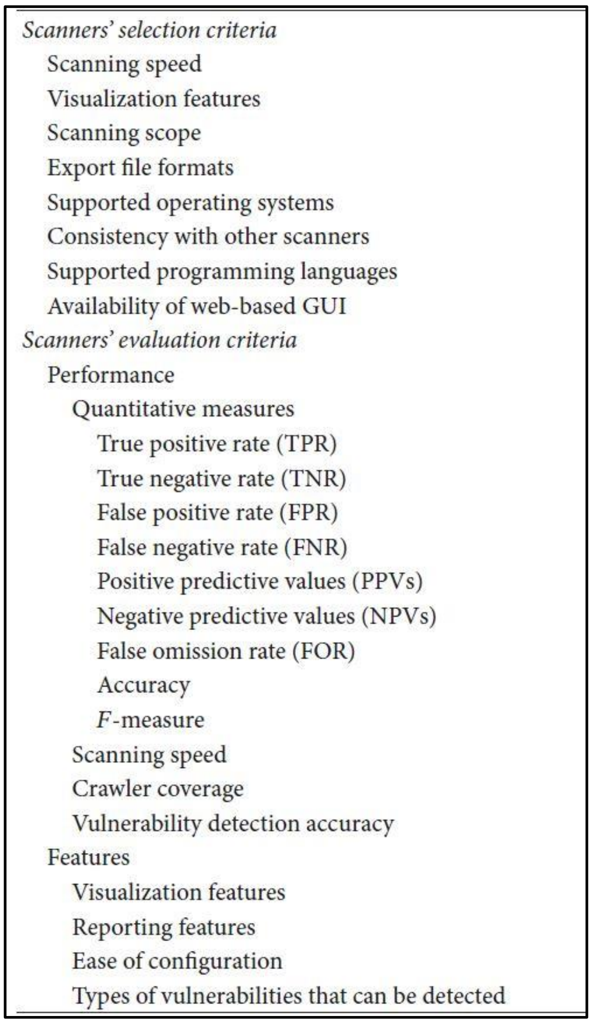
It is a mix of white and grey box test. Here the penetration tester has knowledge of the infrastructure and the target web application. Tester also has user credentials for the application, using which business logic and internal vulnerabilities can be tested and identified [11]. Automated tools if configured can perform grey box tests. It does not come naturally to automated tool to perform a grey box test, but manual intervention and configuration is required time to time.

2.2.3 White Box test:

In white box test, all the information about the web application such as, application code, infrastructure diagram, user privilege details and server configuration details are provided to the penetration tester. Static and dynamic code analysers are used for this test. It is called as full information test [11].

2.3 Web Vulnerability Scanner Evaluation

In the cyber security industry and academia there is growing interest in web application scanners and automated tools evaluation. There are multiple benchmark applications available for web security scanner evaluation. In 2017 Mansour Alsaleh et al. [12] has critically assessed four penetration testing automated tools, namely, Skipfish[3], Arachni[4], Wapiti[5], IronWASP[6]. The authors have compared open source web application vulnerability scanners based on the performance parameters such as scanning speed and crawling coverage. The case study was conducted on following evaluation criteria as shown in figure 2.2.



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 2.2: Vulnerability evaluation matrix [12]

The evaluation of these tools was specific to the version considered in the test case. Scanners speed were evaluated using three vulnerability test websites: (1) Web Scanner Test Site[7] and (2) Acunetix site[8]. The paper compares scanner behaviour, speed, detection accuracy and crawler coverage. The research observed that Skipfish was one of the fastest scanners to complete the scan but had lesser crawler coverage as compared to Arachni, which took more time to complete the scan. Arachni obtained 94% scanning coverage, whereas average crawler coverage observed was 48%. The paper concludes that there were considerable

[3] Skipfish: https://code.google.com/archive/p/skipfish/
[4] Arachni: https://www.arachni-scanner.com/
[5] Wapiti: http://wapiti.sourceforge.net/
[6] IronWASP: http://blog.ironwasp.org/
[7] Web Scanner Test Site, 2015: http://webscantest.com/
[8] Test Website for Acunetix Web Vulnerability Scanner: http://testaspnet.vulnweb.com/

inconsistencies between outputs of the scanners in scope. This might be because of performance factors of each scanning tool.

In [13], the author has proposed a combinatorial testing. The case study was performed to provide a framework for Cross-site scripting (XSS) attack and SQL injection attack, using manual as well as automated testing. The research considers Burp Suite[9] and OWASP ZAP[10] tool for the evaluation. The test suggests that out of two test applications DVWA and Mutillidae the Burp suite was able to detect 66.6% XSS instances. On the other hand, OWASP ZAP was able to detect 80% of the XSS attacks. The research also states that out of all the test scans made, OWASP ZAP performed 25% better as compared to Burp Suite. We have also used these tools in our research and broadened the scope of comparison to all the features of the tool. In a research conducted by Jose Fonseca and Marco Vieira and Henrique Maderia [14] also the researcher has compared web application scanners on serious vulnerability detection such as XSS and SQL injection attack. In this research three scanners were tested, which resulted in higher percentage of false positive rate.

In an attempt to provide comparative analysis of web application vulnerability assessment (VA) and Penetration testing (PT) in [7], the author has studied and compared three web vulnerability scanners. Acunetix[11], Burp Suite, OWASP ZAP are the web vulnerability scanners included in the research. Although two of the tools in this research are commercial and have better post sales support the OWASP ZAP also has a strong open source community to support and add features. The research compares results obtained from the manual penetration test with the results obtained from the web vulnerability scanners. The research concluded that in though the detection rate of web application scanners is not as par with manual test, but this approach gives advantage when money and time factors are considered.

The web application scanner also performs scanning of web services. In this approach Nuno Autunes and Marco Vieira [15], evaluates automated tool from web services point of view. Author has discussed HP Webinspect[12], IBM Rational Appscan[13] and Acunetix web vulnerability scanners results after scanning web service included in 'Benchmark for SQL injection vulnerability detection tools' [16]. The paper compares false positive vulnerabilities reported and confirmed vulnerability reported. The evaluation performed on more than 20 web services shows that signature-based approach to identify vulnerabilities obtain better results. The false positive rate of reporting was observed to be reduced by a large margin. A coverage rate of ~70% was observed.

The research done by Mayur Turuvekere and Anala A. Pandit [4] on comparative study of pen testing tools studies 5 well known penetration testing tools. The tool set includes 3 commercial tools Acunetix, Burp suite, Netsparker and 2 freeware tools Wapiti, Arachni. Objective of research is to evaluate the various pen testing tools. The research has utilised WIVET (Web Input Vector Extractor Teaser) for measuring crawling coverage. Acunetix was able to get highest coverage of crawling 94% in test application. The pen testing tools were compared on 21 vulnerability detection.

[9] Burp Suite: https://portswigger.net/burp
[10] OWASP ZAP: https://www.zaproxy.org/
[11] Acunetix: https://www.acunetix.com/
[12] Web inspect: https://www.microfocus.com/en-us/products/webinspect-dynamic-analysis-dast/overview
[13] IBM Appscan: https://www.ibm.com/cz-en/security/application-security/appscan

OWASP scanner benchmark has been used to evaluate the web application vulnerability scanners [17]. The author also compares OWASP benchmark and WAVSEP benchmark as many research papers have discussed WAVSEP previously. The research found that after combining the performance of two scanners in both the benchmarks the version of Arachni performed better in SQLI, XSS and CMDI (Command Injection) attack detection.
The author Zoran Duric in [18] has developed a web application penetration testing tool (WAPTT) by combining static (SAST) and dynamic security (DAST) analysis approaches. The objective was to develop a reliable black box vulnerability scanner. Further author has compared performance of WAPTT with six other well-known tools. The test targets were JSP, JSF and PHP applications. Although the authors own tool performed well in detecting vulnerabilities, but Acunetix had performed best amongst those sever tools with least false positive rate. The author says that as compared to other commercial tools his tool has advantage of modularity. This will allow end-user to easily extend functionalities in the scanner tool.

Since the major research done till now is related to open source scanners performance and their evaluation based on limited benchmarks, we intend to provide a framework for scanner evaluation based on the latest demands and vulnerabilities in the cyber security domain. The objective of this study is to provide a framework for scanner evaluation and test the framework against industry accepted scanners. The second objective of this thesis is to provide a comparative study of key features and capabilities of the scanners.

3 Research Methodology

In this chapter we discuss research methodology for web application penetration test scanner comparison framework and evaluate selected web application security scanners. Following figure shows steps undertaken in methodology.

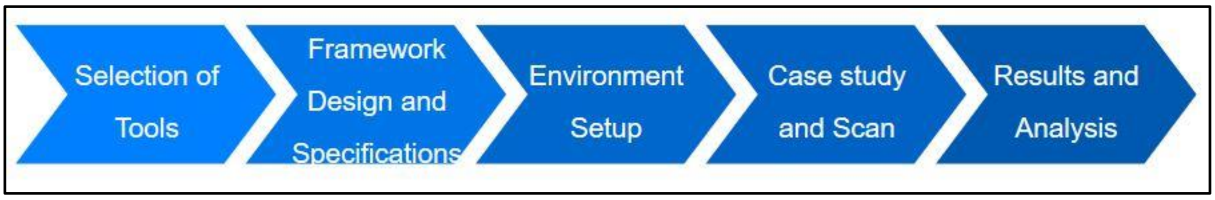
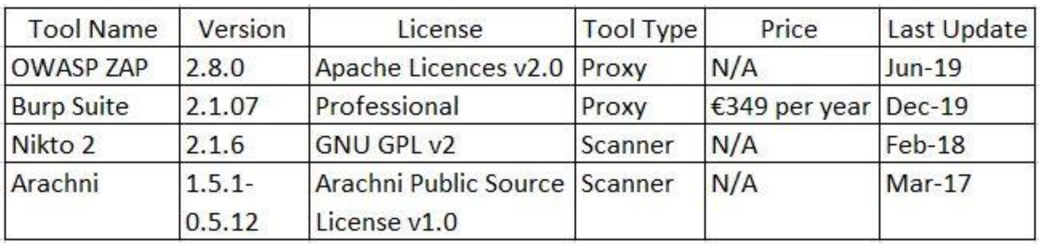


Figure 3.1: Methodology

3.1 Selection of Tools:

Our research started from selection of tools for our research. The objective was to select and evaluate well known tools in cyber security industry. We had considered well known tools based on Gartner Magic Quadrant for application security testing [19] such as Netsparker, Acunetix, Micro Focus Webinspect and IBM appscan. But due to the unavailability of licenced version of the application it was not possible to proceed with their research. Taking learnings from the work done during the internship period, we shortlisted three widely used open source tools and one licensed tool.

The tools were divided in two categories 1) Proxy tools and 2) Scanner tools. The Proxy tools are used as web proxy as well as they have scanner tools embedded in them. This type of tools provides more fine control over the target request-response interaction. It also enables penetration tester to scan post login requests with much ease. The proxy component allows testers to perform manual penetration test. The scanner tools are standalone scanners which are used in Point and Shoot (PaS) configuration to perform automated scan. Table 3.1 lists all the tools evaluated in this thesis.



Tool Name	Version	License	Tool Type	Price	Last Update
OWASP ZAP	2.8.0	Apache Licences v2.0	Proxy	N/A	Jun-19
Burp Suite	2.1.07	Professional	Proxy	€349 per year	Dec-19
Nikto 2	2.1.6	GNU GPL v2	Scanner	N/A	Feb-18
Arachni	1.5.1-0.5.12	Arachni Public Source License v1.0	Scanner	N/A	Mar-17

Table 3.1: Evaluated Scanners

3.1.1 OWASP ZAP:

OWASP ZAP proxy is an open source web application penetration testing tool. The tool is actively maintained by OWASP community and volunteers [20]. ZAP proxy tool provides many functionalities such as Man-in-the-middle proxy, Spider tool, Active scanner, Passive Scanner, In app browser for manual test, Web socket scanner, SSL scanner and code review vulnerability detection [20]. It also has mature module for web services and API scanning. OWASP ZAP also has a marketplace option which allows users to add optional add-ons to the tool.

3.1.2 Burp Suite:

Burp Suite is most widely used penetration testing tool by security professionals. It is published by PortSwigger Ltd[14]. It is a graphic user interface (GUI) based tool. It provides community version of the tool with limited functionalities. The tool features include web proxy, automated scanner, intruder, spider module, repeater, inbuilt decoder, comparer and sequencer modules. Similar to OWASP ZAP, Burp Suite also provides options to add extensions in the tool [21].

3.1.3 Nikto 2:

Nikto is a command line based freeware tool for penetration test and automated scanning [22]. The tool is a pearl based security scanner. Nikto makes use of various switches for configuration and to use various modules. Switch such as '-evasion' will use various techniques to evade firewall or detection system on the target system. It has various plugins to detect vulnerabilities as per CVE[15] details. It is a lightweight web application scanner which can run on a system with lower configuration.

3.1.4 Arachni:

Arachni is another widely used web application penetration testing tool. It is similar to Nikto. It is a command line based tool. Arachni work on all the major operating systems. As per its developer's recommendation, we have used this tool on Linux based environment. It provides output results in 7 formats which is the maximum number of formats encountered till now. Arachni has been popular in academic community and been discussed in [9], [12] and [7].

The next section will discuss about the framework used to evaluate these tools and our contribution to the process in depth.

[14] PortSwigger Ltd: https://portswigger.net/contact

[15] Common Vulnerabilities and Exposure: https://cve.mitre.org/

3.2 Framework Design and Specifications

This section describes our framework which is a comparative matrix. We will discuss parameters selected in our matrix in depth. This section addresses the first objective of our research.

After studying existing web application scanner evaluation frameworks such as Web Input Vector Extractor Teaser (WIVET) [2], Web Application Vulnerability Scanner Evaluation Project (WAVSEP) [3] and OWASP Benchmarking project [23] which focuses towards specific areas of the automated scanners, we are proposing a framework which covers larger ground as compared to existing frameworks.

The are 14 parameters which we will be considering while evaluating the web application scanners. We propose a score based comparative analysis of tools. Each key parameter has a score system of up to 5 points.

3.2.1 Parameters

1) Type of tool:

There are two types of tools: 1) Graphic user interface (GUI) based, 2) Command line interface (CLI) tools. This parameter plays a key role in ease of access factor. While using the scanner many penetration testers prefer tools having GUI interface. On the other hand, CLI based tools are lighter and puts less efforts for graphical processing hence are faster. We will also discuss whether it is a web proxy tool, which runs by intercepting a browser request or a scanner tool which directly interacts with the target for scanning.

2) Type of Penetration test:

Initially penetration test automated tools were used for Black box test only as they lack the capability to handle the session cookies required for authenticated scan. Modern scanning tool however have ability to handle the application session and can detect irregularities in application code as well.

Score for type of penetration test:

- 1 - Black box test
- 2 – Black box and Grey box test
- 3 – Black, Grey and White box test

3) Crawling types:

Crawling is a type of scan which involves mapping the application by navigating request by request. The crawler catalogues the links found during the process, later these links are utilised for scanning purpose. There are two types of crawling 1) Active crawling and 2) passive crawling. Active crawler interacts with the application and sends requests to the application server to get the active links. Passive crawler works silently with out actively engaging the application. The passive crawler works while manually navigating through the application. Hence it can cover more links and paths in the application, proving a greater coverage.

Score for crawling function:

- 1- only Active crawler
- 2- Active and Passive Crawler

4) Crawler coverage:

Web application crawling is a part of information gathering stage in the penetration testing process. In this stage a penetration tester would like to gather as much information as possible about the web application. Therefore, it is crucial that automated scanner covers as much application it can without manual intervention. The crawler coverage can be measured by number of URL crawled by scanner.

Score for Crawling coverage:

- 1- Less than 25% coverage
- 2- 25% to 50% coverage
- 3- 50% to 70% coverage
- 4- 70% to 90% coverage
- 5- More than 90% coverage

5) Scanning Speed:

The automated tool is utilised by the penetration tester to cover the greater area in a large application. Therefore, time taken by the application to completely scan the application is essential to understand the efficiency. In our evaluation we will scan a benchmark application and record the time taken and number of requests made by each scanner.

Score for Scanning Speed:

- 1- More than 6 Hours
- 2- More than 3 Hours
- 3- More than 2 Hours
- 4- More than 45 minutes
- 5- Less than 30 minutes

6) Scan type:

There are two scan types when it comes to application scanning, Passive scan and Active scan. The passive scan analyses request and responses gathered from manual crawling or interaction by the penetration tester and does not send additional data to the application server. The passive scanning includes identifying issues such as missing response security headers, SSL certificate misconfigurations. It also includes client side Java Script analysis. Active scan phase can detect presence of vulnerabilities by interacting with the web application server. The scanner sends crafted requests to the server and tries to exploit the vulnerabilities. Scanner will try to find common vulnerabilities such as listed in OWASP Top 10.

Score for Scan Type:

- 1- Only active scan
- 2- Active and Passive scan
- 3- Active, Passive and static script vulnerabilities detection

7) Vulnerability Detection Rate:

The vulnerability detection rate would be the number of vulnerabilities detected in a test against a benchmark application. The detection rate would tell us how many vulnerabilities the application was able to detect successfully. It will also show the ability of application to detect various vulnerabilities.

i) Formula for vulnerability detection:

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

Equation number: 3.1

ii) Score for Vulnerability detection rate:

- 1- 10% vulnerabilities detected
- 2- Up to 25% vulnerabilities detected

- 3- Up to 50% vulnerabilities detected
- 4- Up to 75% vulnerabilities detected
- 5- 100% vulnerabilities detected

8) False Positive reported:

False positives are the vulnerabilities which were reported by the scanner as positives but not present in the application [23]. The scanner which provides lesser percentage of false positive results is preferred.

i) Formula for False positive rate:

FP rate = $\frac{False\ Positive}{False\ Positive+True\ Negative} * 100$

Equation number:3.2

ii) Score for False positive reported:
- 1- Greater than 50%
- 2- Grater than 30%
- 3- Lesser than 30 %

9) True Positive reported:

A True positive is a vulnerability which is accurately detected by the scanner and reported as well [23]. The true positive rate is always desired as high as possible. It is also known as precision rate [24]. The figure 3.2 shows the OWASP benchmark results interpretation guide. According to the guide the ideal value of the results should come in the marked region.

i) Formula for True Positive:

TP rate = $\frac{True\ Potive}{True\ Positive+False\ Negative} * 100$

Equation number:3.3

ii) Score for True Positive reported:
- 1- Less than 10%
- 2- Up to 25%
- 3- Up to 50%
- 4- 50% and greater

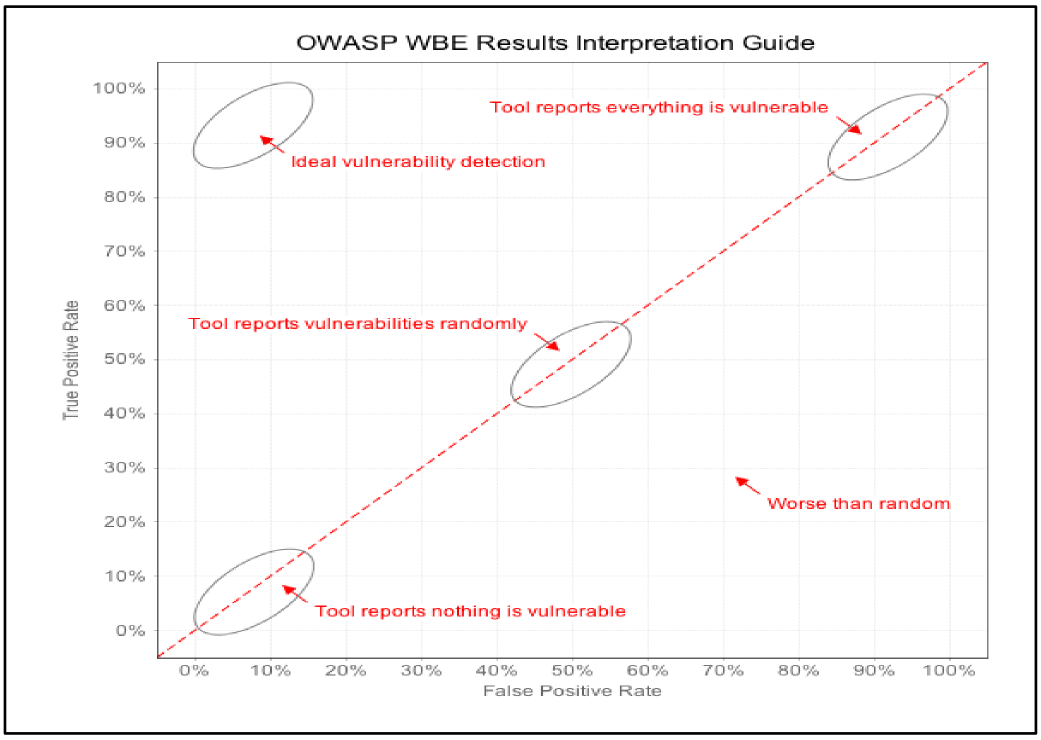


Figure 3.2: OWASP interpretation guide [23]

10) Reporting Features:

There are various formats in which automated tool generate reports, but few common points such as executive summary, list of vulnerabilities detected, list of request response and details of target scanned are expected in the report. Common formats such as PDF, HTML and XML are also commonly provided by scanners. On top of that compliance reports with various standards such as PCI-DSS[16], HIPAA[17] and OWASP top 10 are also provided by the scanner which allows user to quickly analyse if they have achieved desired standard.

11) Addons and Extension features:

The automated tools have functionality extension feature, which allows user to add new features to the scanner. Extension feature enables user to add tools to the scanner which can be available from time to time.

12) Ease of configuration:

The ease of configuration defines how easy it is to use the scanner and what are its dependencies. We define three levels of configuration, 1) Easy (Plug and Play) out of the box ready to use application, 2) Hard: having some dependencies such as Java, php installation, 3) Difficult (Expert level): Having specific requirements such as server and database configuration.

13) Scan logs:

In penetration testing process logging of each request and response is essential. As these logs serve as proof of the actions performed on the customers application environment. Automated tool makes number of requests to the application server while scanning. These

[16] PCI-DSS: https://www.pcisecuritystandards.org/pci_security/

[17] HIPAA: https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html

requests also contain crafted payloads for various vulnerability detection. This should be recorded in a log file and can be reproduced if required. Automated tools provide options to store logs in .csv, html or xml formats.

14) Cost of the tool:

The cost factor plays major role in the selection of a tool. Many tools have similar functionalities and features, but the cost difference due to their brand is major. Some freeware tools also have better performance and features than commercial tools because of its strong development community.

We have considered above all the parameters for automated tool evaluation. Our framework proposes a scoring scale of 30 points. We can put any automated penetration testing tool against it to identify which tool is better.

4 Implementation

In this section we will discuss the artefact implementation. We have taken two approached to evaluate the scanning, crawling and vulnerability detection capabilities of the tool. We have utilized OWASP Benchmark test [23] tool to evaluate the vulnerability detection and crawling coverage of the tool.

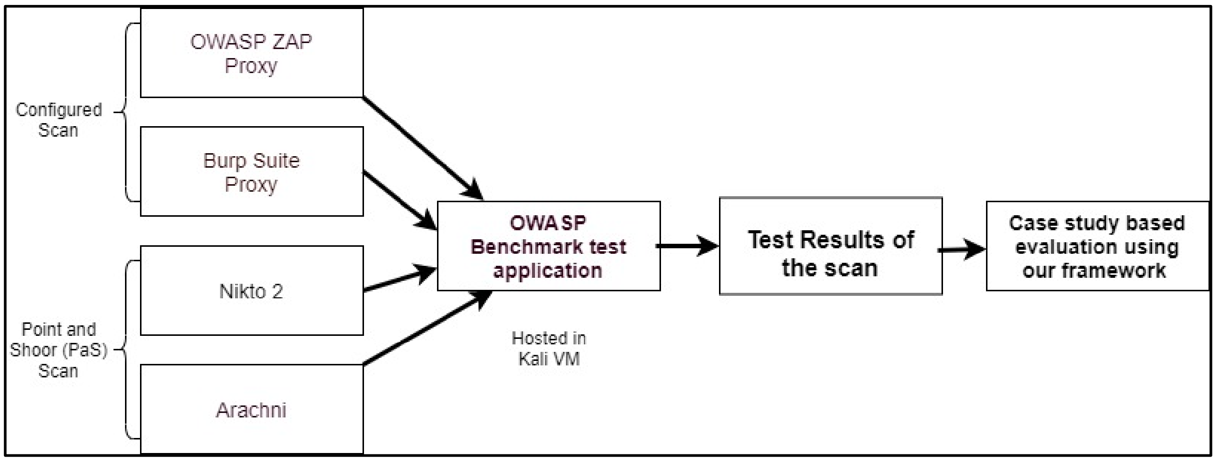
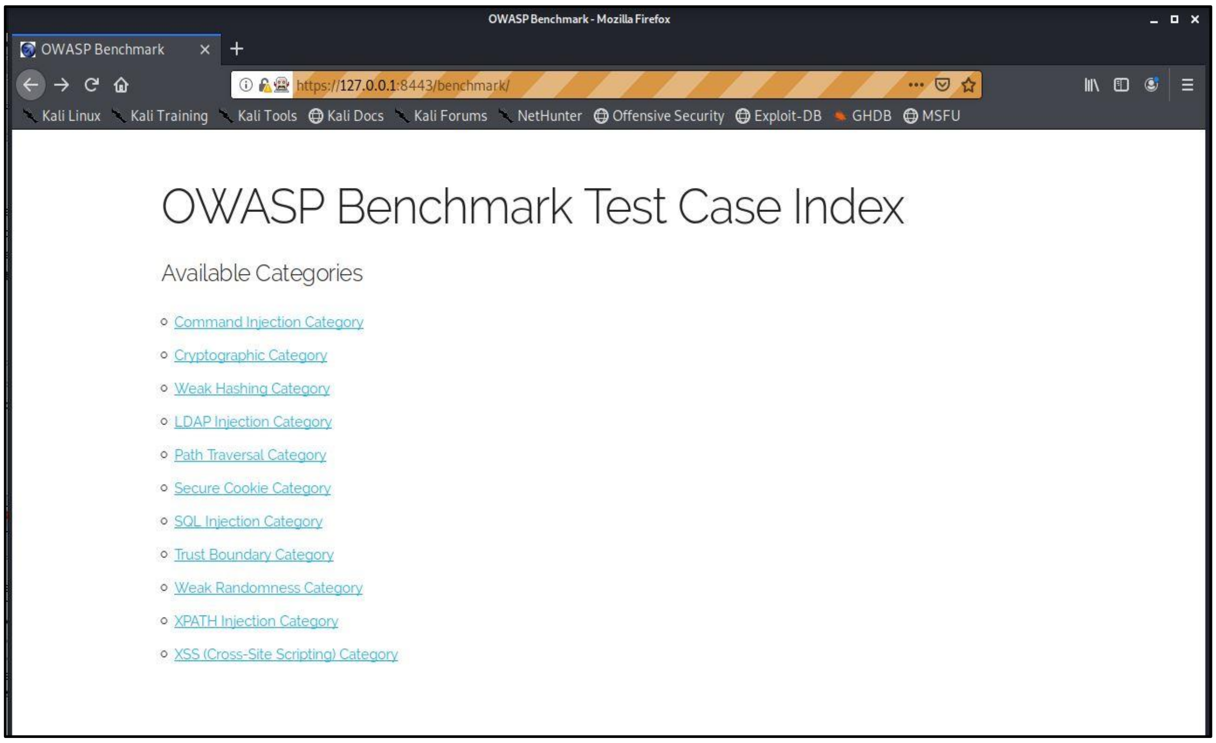

Configured Scan
OWASP ZAP Proxy
Burp Suite Proxy
Point and Shoor (PaS) Scan
Nikto 2
Arachni
OWASP Benchmark test application
Hosted in Kali VM
Test Results of the scan
Case study based evaluation using our framework

Figure 4.1: Scanner evaluation process

4.1 OWASP Benchmark test:

The OWASP benchmark test is an open source JAVA based vulnerability test case suite designed to evaluate accuracy of vulnerability detection, speed and crawler coverage of automated scanners. It is state of the art benchmarking tool which in regularly updated. It has over 2740 instances of vulnerabilities ranging across OWASP top 10. It has not been used to evaluate the current versions of OWASP ZAP, Burp suite, Nikto 2 and Arachni till now. It runs on Tomcat 8.x server and default port is 8443.
The application can be accessed from the browser at URL: https://127.0.0.1:8443/benchmark/.


OWASP Benchmark - Mozilla Firefox

OWASP Benchmark

https://127.0.0.1:8443/benchmark/

Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU

OWASP Benchmark Test Case Index

Available Categories

- Command Injection Category
- Cryptographic Category
- Weak Hashing Category
- LDAP Injection Category
- Path Traversal Category
- Secure Cookie Category
- SQL Injection Category
- Trust Boundary Category
- Weak Randomness Category
- XPATH Injection Category
- XSS (Cross-Site Scripting) Category

Figure 4.2: OWASP Benchmark application

4.2 Test Approach:

We evaluated scanners based on following two testing approaches:

1) Point and Shoot (PaS) scan

In this approach we only give the URL of the target system to the application and allow it to run on a default profile. We have not done any changes to the setting of the tool and no extra switched are added to the scanner command. This approach is usually taken by first time users and beginners. It allowed us to analyse the natural capability of an automated tool to crawl the application and scan the same for vulnerabilities. The penetration tester relies on scanner ability to detect vulnerabilities with little human interaction.

2) Configured Scan

In configured scan, we tweak the settings as per the application requirements. We performed manual crawling to train the scanner as per the application. This involved clicking on each URL in the application and visiting each page, visiting each page with proxy mode enabled so that scanner can gather list of all the URLs.

4.3 Environment Setup:

The evaluation process contains following setup. The setup is configured in Kali Linux VM and Windows host machine:

Step 1: Setting up the environment includes installing the scanner (Burp suite, OWASP ZAP, Arachni and Nikto) to attack the OWASP benchmark application. It also included installation of dependency applications and services. For Burp suite and OWASP ZAP we first setup the proxy to the browser and intercept the application in web proxy. Then we crawled the application manually, request by request. We then performed active crawling on the application from both the tools. This discovered further URLs in the application and listed then under target scope. We added target URL to the scope section of the tool. By doing this the tool will only intercept and scan the target URL. Then we perform the scan on the benchmark target system by clicking ‘Active scan’ button in ZAP and ‘Scan’ button in Burp Suite’.

For Arachni and Nikto, we perform the scan using command line interface. We install Arachni and Nikto as discussed in configuration manual. We execute the point and shoot scan using command ‘arachni’ and ‘nikto’ command followed by application root URL. We also add switches to the command to save the scan results to our project folder.

Step 2: To get the benchmark evaluation we need ‘.xml’ output from each tool. For Burp suite, OWASP ZAP and Nikto we take scan output in ‘.xml’ format by default. For Arachni the default output format is ‘.afr’ (Arachni Framework Report) the report was then converted in ‘.xml’ format using ‘arachni_reporter’ tool.

Step 3: The OWASP benchmark analyses the tool results and provides compliance output. The XML result files from all the scanners were copied to ‘results’ folder in ‘benchmark’ root folder. Run command ‘createScorecard.sh’ to create the results from benchmark.

We have primarily considered five vulnerabilities given in the OWASP benchmark for evaluation. These ten vulnerabilities include command injection, LDAP injection, XSS, SQL injection, and Secure cookie flags [23]. Apart from these ten vulnerabilities we also consider other vulnerabilities reported by the scanners for overall evaluation of the scanning tool.

We have discussed results obtained from the benchmark test and other parameters of the framework in the following section.

5 Evaluation

In this chapter we present our findings from running experimental scan on OWASP benchmark test application. We also compared the scanners based on our framework and calculate test score for each of the scanner and rank them. This chapter addresses second and third objective of our research.

5.1 Case Study 1: OWASP ZAP and Burp suite

The tools considered in this case study are GUI based. We have performed this case study with manual crawling of the application which would enable the tool to cover maximum area of the application and selecting the configuration such as audit speed to ‘Thorough’ in Burp suite and making concurrent scanning per host to ‘2’ in OWASP ZAP. We will compare OWASP ZAP and Burp Suite on points mentioned in framework.

1) Type of tool:

The OWASP ZAP and Burp Suite both are GUI based tools and also are web proxy tools. GUI based tools are easy to use for any level of user. During our experiment we found ZAP to be more user friendly, since all the modules were clearly specified and easily accessible. OWASP ZAP provides an option to launch a browser from within the tool. This browser is pre-configured with ZAP proxy so that user does not have to manually configure it. On the other hand, the Burp suite has to be manually configured with the browser by changing the proxy configurations under the ‘settings’ or ‘options’ tab.

2) Type of Penetration test:

Primarily both the scanners can perform Black box test without any manual interaction. For grey box type of test, both the scanners have add-ons which can handle login request and credentials. In Burp suite this can be performed using ‘macro’ function and in OWASP ZAP this can be handled using ‘session properties’ settings. These add-ons can handle login logout requests which can keep session active throughout the scanning process. Both the tools perform Dynamic Security Application Test (DAST) type of test and perform above two types of tests.

3) Crawling types:

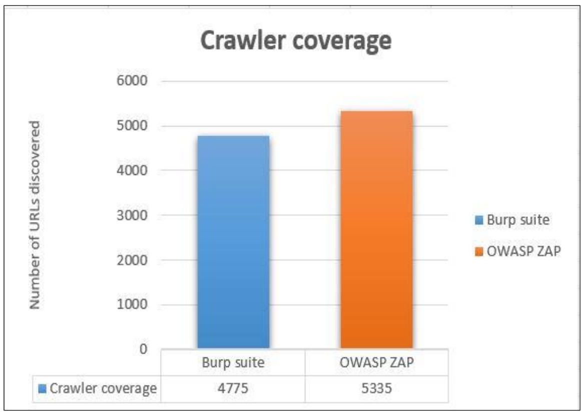
Both the tools have active and passive scanning feature which allows application crawl logging while browsing the application.

4) Crawler coverage:

The OWASP benchmark has nearly 5500 URLs including the pages and actions (Login, submit etc), in the OWASP benchmark project ZAP was able to discover 5335 URLs which is 97% of the URLs and Burp Suite was able to discover 4775 URLs which is 87% of the URLs. Shown in figure 5.1.1.

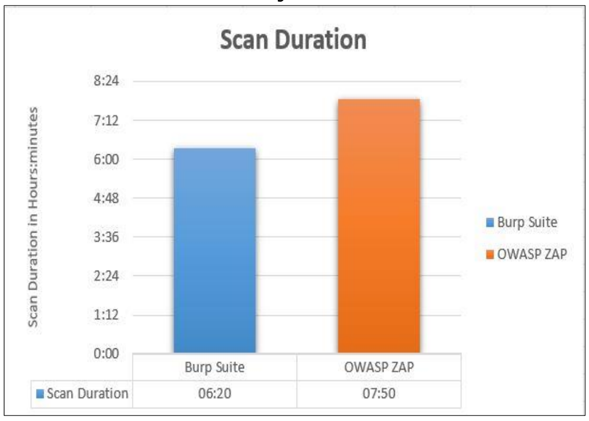
5) Scanning Speed:

As shown in the figure 5.1.2, during the experiment the OWASP ZAP performed scan in 7 hour and 50 minutes and Burp suite performed in 6 hours and 20 minutes. Both the scanners were unable to score high points in this section. This might be due to the benchmark being a large completely vulnerable application. But this is not the case always.



	Burp suite	OWASP ZAP
Crawler coverage	4775	5335

Figure: 5.1.1 Crawler Coverage



	Burp Suite	OWASP ZAP
Scan Duration	06:20	07:50

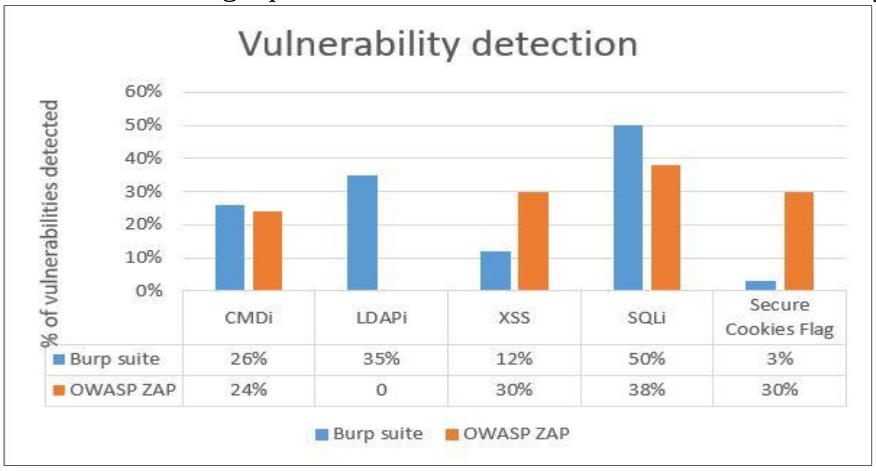
Figure 5.1.2 Scan duration

6) Scan type:

The scanners were able to perform active as well as passive scan successfully on the benchmark tool. But none of the CSS or java script based vulnerabilities were detected.

7) Vulnerability Detection Rate:

The total number of vulnerabilities detected by ZAP is 18 and by the Burp suite is 26. This includes High, Medium, Low level and informational findings as well. Considering the OWASP Benchmark vulnerabilities figure. 5.1.3 shows the percentage of the selected vulnerabilities detection. Using equation 3.1, we calculate the rate of vulnerability detection.

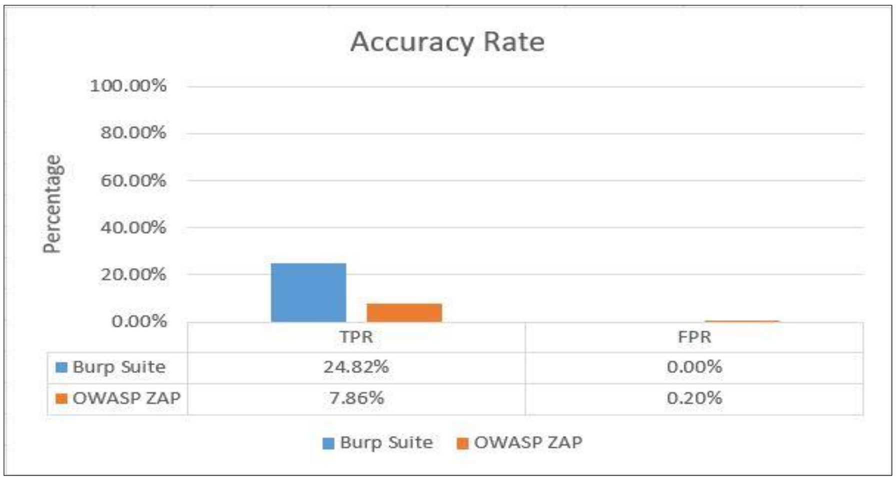


	CMDi	LDAPi	XSS	SQLi	Secure Cookies Flag
Burp suite	26%	35%	12%	50%	3%
OWASP ZAP	24%	0	30%	38%	30%

Figure 5.1.3 Rate of vulnerability detection

8) True Positive and False Positive reported:

The figure 5.1.4 shows true positive and false positive detection rate by both the tool. The Burp suite was able to detect 24.82% of vulnerabilities which were actually present in the application whereas OWASP ZAP was able to detect 7.86% of the vulnerabilities in the application. Note that these are all the vulnerabilities which were present in the application. ZAP also reported few vulnerabilities which actually were not present in the application. Its percentage was 0.20%.



	TPR	FPR
🟦 Burp Suite	24.82%	0.00%
🟧 OWASP ZAP	7.86%	0.20%

Figure 5.1.4 False positive and True positive ratio

9) Reporting Features:

The OWASP ZAP has advantage over the Burp suite when comparing reporting feature. The Burp suite can generate report in 2 formats HTML and XML, whereas the OWASP ZAP can generate report in 4 formats HTML, XML, JSPN and markdown report format.

10) Addons and Extension features:

We found that in OWASP ZAP we can make use of ‘Market place’ tab to add extensions. During the analysis there were 66 extensions available in zap marketplace. The Burp suite has 200+ extensions whereas its community version has 150+ extensions

11) Ease of configuration:

We found that both the tools can be categorised as easy to configure and use. There were very less dependencies such as JAVA for Burp suite.

12) Scan logs:

Burp suite and OWASP ZAP logs all the requests under history tab. For OWASP ZAP maximum number of requests to be stored is set to 1000 by default.

13) Cost of the tool:

OWASP ZAP is opensource tool will all the functionalities. Burp suite has three versions Enterprise, Professional and Community version. The Community version is free and has limited number of functionalities. Enterprise and Professional versions costs €3499 and €349 respectively per year.

Sr. No	Parameters	Burp Suite	OWASP ZAP
1	Type of tool	GUI Proxy	GUI Proxy
2	Type of Penetration test	2	2
3	Crawling types	2	2
4	Crawler coverage	5	4
5	Scanning Speed	1	1
6	Scan type	2	2
7	Vulnerability Detection Rate	3	2
8	False Positive reported	3	3
9	True Positive reported	2	1
	Total Score:	20/30	17/30

Table: 5.1

5.2 Case Study 2: Arachni and Nikto2

This case study has been performed with tool in Point and shoot (PaS) configuration. The command line interface was used without any extra switches applied to the application.

1) Type of tool:

Arachni and nikto are both command line based scanners. Arachni also provides additional web interface for ease of use.

2) Type of Penetration test:

Both the scanners can perform only black box test. Arachni does have credential handling option is its manual but it is not always effective. But we will consider that in our analysis.

3) Crawling types:

As the scanners do not have a web proxy interface, hence user cannot perform passive crawling. The Arachni and Nikto perform active crawling only.

4) Crawler coverage:

In the scanning report of both scanners there were no significant crawl results obtained. The Crawler performed dictionary based directory brute forcing in both the scanners, but it turned unfruitful.

5) Scanning Speed:

As there were few URLs to be scanned the scan completed in 30 minutes and15 minutes for Arachni and Nikto 2 respectively. Multiple scans were performed to check if the application is being scanned properly. This is represented in figure 5.2.1.

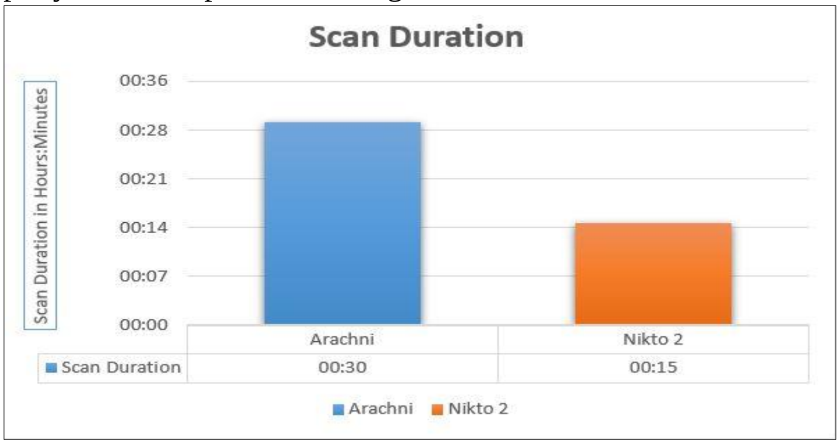


Figure: 5.2.1 Scan duration

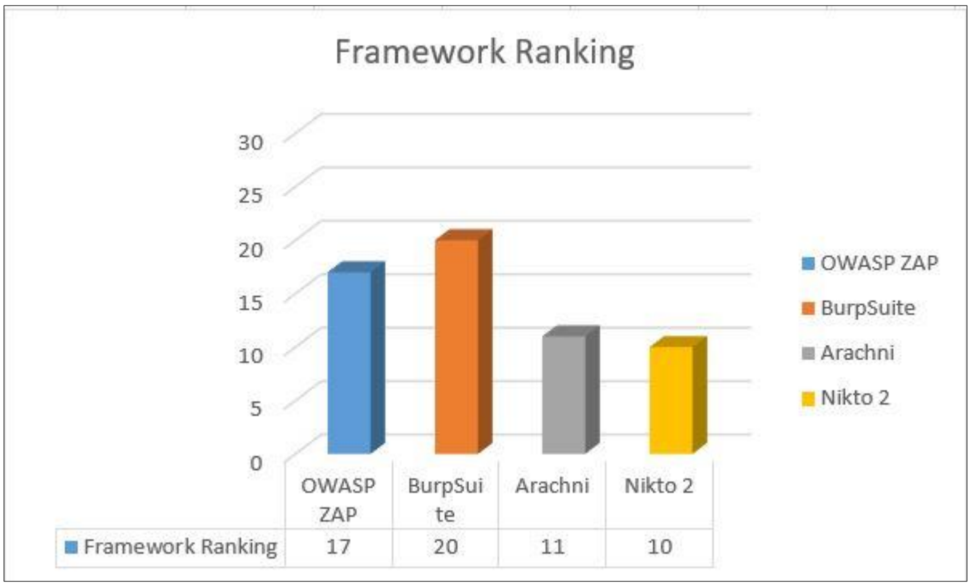
6) Scan type:
Both scanners performed Active scanning only, depending on the crawler coverage.
7) Vulnerability Detection Rate:
The scanners did not identify any significant vulnerability in the Benchmark application. The detection was limited to missing request-response headers and TLS suite identification.
8) Reporting Features:
The reporting feature of Arachni is strongest from what we have studied till now. It can produce report in 8 formats which are stdout, XML, YAML, HTML, Marshal, json, ap, txt. Nikto also has 5 formats in its reporting structure, which are text, CSV, HTML, XML, MSF.
9) Addons and Extension features:
Arachni and Nikto does not have active support for add-ons but do have pre-configured switches for scan improvement. Arachni do have one additional feature which is web interface. This can be used using command 'arachni-web' in 'usr/share/arachni/' directory.
10) Ease of configuration:
Both the tools were used on Kali linux. If the user is aware of basic Linux commands, then the installation and use would be fairly simple. Each action can be performed using single commands.
11) Scan logs:
Both the tools generate scan logs by using command line switch '--reroute-to-logfile' in Arachni and '-Format' in Nikto in form of report.
12) Cost of the tool:
Both the tools are opensource tools. Arachni does provide enterprise support in a paid version.

Sr. No	Parameters	Arachni	Nikto 2
1	Type of tool	CLI Scanner	CLI Scanner
2	Type of Penetration test	1	1
3	Crawling types	2	1
4	Crawler coverage	1	1
5	Scanning Speed	5	5
6	Scan type	1	1
7	Vulnerability Detection Rate	1	1
8	False Positive reported	0	0
9	True Positive reported	0	0
	Total Score:	11/30	10/30

Table: 5.2

6 Discussion

This chapter discusses the results of case studies and four scanners. The analysis will allow us to discuss the objective of this research. We will discuss outcomes of all the scanners with a high-level overview.



	OWASP ZAP	BurpSuite	Arachni	Nikto 2
■ Framework Ranking	17	20	11	10

Figure 6.1 Framework Score ranking

The figure 6.1 shows the points achieved by each tool on our 30 points frameworks. It was observed that tools with web proxy perform better than command line tools. The OWASP Benchmark has its own set of limitations when it comes to testing a DAST tool. It only challenges a tool to identify specific vulnerabilities. Also, it was observed that the Benchmark tool does not identify NIKTO results. As a result, the scan output was analyzed manually. Arachni and Nikto were unable to identify the vulnerabilities which we considered for the test. This was surprising as Arachni has performed well in previous research done in [12], [25]. This might be due to the version upgrade and mismatch with Benchmarks present format. The configured scan produced more and confident results as compared to point and shoot scan. This is because of the visibility of the application and its internal URL which passive crawling provides is difficult to get in active scanner.

Crawl coverage of the ZAP was maximum with 5335 URLs, but this also resulted in longer scan duration. ZAP was able to determine highest number of command line injection attacks, XSS attacks and SQL injection attacks. Some crawling and scanning difficulties were observed by Arachni and Nikto, which resulted into lesser vulnerabilities reported. The Burp Suite had maximum number of true positive findings with maximum successful detection SQL injection attacks. Burp and ZAP can incorporate reporting tool such as the one in Arachni. Despite being an opensource tool OWASP ZAP performed better in the test scan. It has detected almost all the vulnerabilities which were detected by commercial Burp suite.

Apart from Benchmark expected vulnerabilities scanners detected many other vulnerabilities such as potential buffer overflow, anti-clickjacking header not present, session cookies related issues.

Automated tools comparison framework which are at present published focuses more on the vulnerabilities detected. But as understood from the internship wok the scanner tool, its features, ease of usability and the services which it can offer are equally important. Hence, our comparative framework covers these points to analyze and compare automated tool with a 360-degree view.

7 Conclusion and Future Work

The objective of this research was to find popular web application scanner which will cover present industry needs. For this we proposed and evaluated a framework which will work as a guideline for future tools evaluation. The evaluation was conducted on 4 well known tools in the industry. Overall the tools with web proxy were found to be more efficient against the benchmark application. ZAP and Burp suite performed better over each other in different categories. Our comparison framework evaluation showed Burp Suite has scored maximum points in the experiments. The test showed that for various vulnerabilities no single scanner can be used. Detection rate of each scanner is different. Therefore, appropriate scanner should be used to detect particular vulnerability. The test also shows that, open source tools also perform better than commercial tools in many cases. For Dynamic web application penetration test its better to have hold on more than one scanner. Not all scanners report all the vulnerabilities.

In this research we evaluated four tools based on OWASP Benchmark application. The vulnerabilities found or missed were limited to this application and its build. However, there are many other benchmarking applications available on which this research can be performed and further evaluated.

Future work of this research could be, 1) Improvement in scanners based on the benchmark results, 2) Evaluation of new open-source scanners, 3) Comparison based on multiple benchmarks. A better future work would be evaluation based on real world application, which would give a more realistic picture on the scanner abilities.

8 Acknowledgement

I would like to thank my supervisor Dr. Muhammad Iqbal for encouraging and guiding me throughout the research process. I would also like to thank entire team of BSI CSIR, Dublin for providing me an opportunity to do internship and their support in research. Lastly, I express my gratitude towards my family and friends who have always been my support and backbone.

9 References

[1] D. D. Bertoglio and A. F. Zorzo, "Overview and open issues on penetration test," *Journal of the Brazilian Computer Society,* pp. DOI 10.1186/s13173-017-0051-1, 2017.

[2] E. T. İslam and B. Urgun, "WIVET—Benchmarking Coverage Qualities of Web Crawlers," *The Computer Journal,* vol. 60, no. 4, p. 555–572, March 2017.

[3] S. Chen, "WAVSEP – Web Application Vulnerability Scanner Evaluation Project," 10 November 2017. [Online]. Available: http://sectooladdict.blogspot.com/. [Accessed 12 November 2019].

[4] M. Turuvekere and A. A. Pandit, "A Comparative Study of Pen Testing Tools," *International Journal of Computer Applications,* vol. 179, no. 50, pp. 26-30, 2018.

[5] P. Xiong and P. Liam, "A Model-Driven Penetration Test Framework for Web applications," in *Eighth Annual International Conference on Privacy, Security and Trust*, Ottawa, ON, Canada, 2010.

[6] M. Mirjalili, A. Nowroozi and M. Alidoosti, "A survey on web penetration test," *Advances in Computer Science: an International Journal (ACSIJ),* vol. 3, no. 6, pp. 107-121, 2014.

[7] S. Nagpure and S. Kurkure, "Vulnerability Assessment and Penetration Testing of Web Application," in *Third International Conference on Computing, Communication, Contril and Automation (ICCUBEA)*, Pune, India, 2017.

[8] O. Communicaty, "OWASP," 2017. [Online]. Available: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf. [Accessed 15 November 2019].

[9] S. Alavi, N. Bessler and M. Massoth, "A Comparative Evaluation of Automated Vulnerability Scans versus Manual Penetration Tests on False-negative Errors," in *CYBER 2018 : The Third International Conference on Cyber-Technologies and Cyber-Systems*, Germany, 2018.

[10] V. Casola, A. D. Benedictis, M. Rak and U. Villano, "Towards Automated Penetration Testing for Cloud Applications," in *IEEE 27th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Italy, 2018.

[11] K. Shaukat, A. Faisal, R. Masood, A. Usman and U. Shaukat, "Security quality assurance through penetration testing," in *19th International Multi-Topic Conference (INMIC)*, Islamabad, Pakistan, 2016.

[12] M. Alsaleh, N. Alomar, M. Alshreef, A. Alarifi and A. Al-Salman, "Performance-Based Comparative Assessment of Open Source Web Vulnerability Scanners," *Hindawi Security and Communication Networks,* vol. 2017, no. Article ID: 6158107, p. 14, 2017.

[13] B. Garn, I. Kapsalis, D. E. Simos and S. Winkler, "On the applicability of combinatorial testing to web application security testing: a case study," in *JAMAICA 2014: Proceedings of the 2014 Workshop on Joining Academia and Industry Contributions to Test Automation and Model-Based Testing*, San Jose, USA, 2014.

[14] J. Fonseca, M. Vieira and H. Madeira, "Testing and comparing web vulnerability scanning tools for SQL injection and XSS attacks," in *13th IEEE International Symposium on Pacific Rim Dependable Computing*, Portugal, 2007.

[15] N. Auntunes and M. Vieira, "Evaluation and Improving Penetration Testing in Web Services," in *IEEE 23rd International Symposium on Software Reliability Engineering*, Coimbra, Portugal, 2012.

[16] N. Antunes and M. Vieira, "Benchmarking Vulnerability Detection Tools for Web Services," in *IEEE International Conference on Web Services*, Miami, FL, USA, 2010.

[17] B. Mburano and W. Si, "Evaluation of Web Vulnerability Scanners Based on OWASP Benchmark," in *26th International Conference on Systems Engineering (ICSEng)*, Sydney, Australia, Australia, 2018.

[18] Z. ĐURIĆ, "WAPTT - Web Application Penetration Testing Tool," *Advances in Electrical and Computer Engineering,* vol. 14, no. 1, pp. 93-102, 2014.

[19] A. Tirosh, M. Horvath and D. Zumerle, "Magic Quadrant for Application Security Testing," Gartner, 18 April 2019. [Online]. Available: https://b2bsalescafe.files.wordpress.com/2019/09/gartner-magic-quadrant-for-application-security-testing-april-2019.pdf. [Accessed 25 November 2019].

[20] S. Bennetts, R. Pereira and R. Mitchell, "OWASP Zed Attack Proxy Project," OWASP, 2019. [Online]. Available: https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project. [Accessed 15 September 2019].

[21] P. Swigger, "The basics of using Burp," Port Swigger, 2019. [Online]. Available: https://portswigger.net/burp/documentation/desktop/penetration-testing. [Accessed 20 November 2019].

[22] C. Sullo, "Nikto v2.1.6," 2014. [Online]. Available: https://cirt.net/nikto2-docs/. [Accessed 15 November 2019].

[23] D. Wichers, "OWASP Benchmark," OWASP, 05 June 2016. [Online]. Available: https://www.owasp.org/index.php/Benchmark#tab=Main. [Accessed 10 November 2019].

[24] D. M. W. Powers, "Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation," School of Informatics and Engineering, Flinders University, Adelaide, Australia, 2007.

[25] E. I. S, B. N, G. F and G. M, "Performance Evaluation of Web Application Security Scanners for Prevention and Protection against Vulnerabilities," *International Journal of Applied Engineering Research,* vol. 12, no. 21, pp. 11068-11076, 2017.