Evaluation of Web Vulnerability Scanners Based on OWASP Benchmark

Balume Mburano [1], Weisheng Si [1]

[1] *School of Computing, Engineering and Mathematics, Western Sydney University, Australia*

b. mburano, w.si@westernsydney.edu.au

Abstract— **The widespread adoption of web vulnerability scanners and their differences in effectiveness make it necessary to benchmark these scanners. Moreover, the literature lacks the comparison of the results of scanners effectiveness from different benchmarks. In this paper, we first compare the performances of some open source web vulnerability scanners of our careful choice by running them against the OWASP benchmark, which is developed by the Open Web Application Security Project (OWASP), a well-known non-profit web security organization. Furthermore, we compare our results from the OWASP benchmark with the existing results from the Web Application Vulnerability Security Evaluation Project (WAVSEP) benchmark, another popular benchmark used to evaluate scanner effectiveness. We are the first to make a comparison between these two benchmarks in literature. Our evaluation results allow us to make some valuable recommendations for the practice of benchmarking web scanners.**

Keywords — security measures, penetration testing, web vulnerability scanner, benchmarking.

I. Introduction

Web applications have become an integral part of everyday life, but many of these applications are deployed with critical vulnerabilities that can be fatally exploited. As the technology used to develop these applications become sophisticated, so are the attacker's techniques. Therefore, Web vulnerability scanners have been heavily used to assess the security of web applications [1].

With the popularity of web vulnerability scanners, there is a need to measure their effectiveness. Benchmarking is one of the techniques used to do so [2]. Several benchmarks have been used to evaluate the effectiveness of web vulnerability scanners. These include Web Input Vector Extractor Teaser (WIVET) [3], IBM Application Security Insider [4], Web Application Vulnerability Scanner Evaluation Project (WAVSEP) benchmark [5] and Open Web Application Security Project benchmark (OWASP) benchmark [6]. Although these benchmarks are considered effective in evaluating scanners, there still exist two research gaps: (1) not all popular scanners have been evaluated by these benchmarks; (2) the results from these benchmarks have never been compared before.

To fill these research gaps, we first use OWASP benchmark to evaluate and compare two popular web scanners, Arachni and OWASP ZAP of their latest versions (v1.5.1 and v2.7) respectively. Before our work, any version of Arachni has not been evaluated by OWASP benchmark, nor has ZAP V2.7. We note here that ZAP V2.6 has been evaluated before by OWASP benchmark, and in our evaluation section, we will point out the performance differences between V2.6 and V2.7. Further, we compare the performance results of these two scanners from OWASP benchmark with their existing results from the WAVSEP benchmark, showing the different capabilities of these two benchmarks.

The remainder of this paper is structured as follows. In Section II, we review benchmarking and vulnerability scanners literature. In Section III, we describe the experimental environment, the selection of benchmark and scanners. In Section IV, we detail our experimental results. Then in Section V, we give conclusions drawn from the experiments and make recommendations.

II. Related Work

To form the basis of this study, we reviewed benchmarks and vulnerability scanners literature to make an informed selection of appropriate benchmark and web vulnerability scanners for evaluation. We focused on scanners' vulnerability detection performance and benchmarks capability to reflect this.

A. Benchmarking Literature

Benchmarks are used to evaluate the performance of web vulnerability scanners. We examined three such tools including Web Input Vector Extractor Teaser (WIVET) [3], Web Application Vulnerability Scanner Evaluation Project (WAVSEP) benchmark, and Open Web Application Security Project (OWASP) benchmark [6]. Of these, WAVSEP benchmark has been the most used to evaluate a wide variety of commercial and open source web vulnerability scanners [7-9]. WIVET, on the other hand, has been used to assess vulnerability scanners' crawling coverage [10]. However, OWASP benchmark has not been used to evaluate many popular web vulnerability scanners although it is developed by a well-known organization and is actively maintained. Moreover, no study has compared results from these benchmarks before. Therefore, we use OWASP benchmark to evaluate Arachni and ZAP and then compare the results with existing WAVSEP benchmark results to show these benchmarks capabilities in reflecting the effectiveness of web vulnerability scanners.

B. Web vulnerability scanners Literature

Although web vulnerability scanners have been established to be effective in unveiling flaws in web applications [11], previous studies have also shown that there are variations between the results reported by different scanners [8, 9, 11, 12]. Specifically, these scanners performance vary in different vulnerability categories and crawler coverage [11]. This increases the necessity for better evaluation of these scanners' effectiveness. Although there are an abundance of open source web vulnerability scanners, we reviewed six such tools including Wapiti, Watabo, W3af, Arachni and OWASP ZAP.

Of these, Arachni and ZAP were the most popular and well-maintained (considering previous studies results [7-9, 11]).

1) Arachni. A high-performance free Open Source ruby-based framework that is aimed to help administrators and penetration testers evaluate the security of web applications. Arachni supports multiple platforms including Windows, Linux, and Mac OSX and can be instantly deployed using its portable packages [13]. Its deployment options include: *Command Line Interface*(CLI) for quick scans, *Web User Interface* (WebUI) for multi-user, multi-scan and multi-dispatcher management and distributed system with remote agents [13]

2) *OWASP Zed Attack Proxy (ZAP),* an easy to use open source scanner for finding vulnerabilities in web applications. It is one of the OWASP flagship projects that is recommended by OWASP for web applications vulnerability testing. ZAP is widely used by people ranging from security professionals, developers, and functional testers for automated security tests that can be incorporated into the continuous development environment. Additionally, ZAP is a free Open Source cross-platform scanner that is becoming a framework for advanced web application vulnerability testing [14].

III. EVALUATION APPROACH

An appropriate methodology was required to evaluate the chosen web vulnerability scanners for this study. The following subsections explain the decision and steps taken throughout the study to select and evaluate the scanners.

A. Benchmark Selection

The first phase of our study involved conducting an initial survey of the existing popular open source benchmarks. While OWASP Benchmark is a free open source program, it remains state-of-the-art as it has a significant number of contributors and it is regularly updated. Therefore, OWASP Benchmark is considered one of the benchmark choices for measuring the effectiveness of vulnerability scanners [6, 15]. Besides, OWASP benchmark has not been used to evaluate Arachni and the latest version of ZAP. It gives the score of a tested scanner based on True Positive Rate(TPR), False Positive Rate(FPR), True Negative Rate(TNR) and False Negative Rate(FNR) [6, 16]. This is particularly important because time and ability needed to discover true and false metrics of a scanner make them incredibly important and a clear understanding of these is required for the choice of a web vulnerability scanner. We, therefore, used OWASP benchmark for this study.

B. Metric Selection.

The used benchmark metrics for scanners evaluation are presented as follow:

- *True Positives (TP)*, False *Positives (FP*), *True Negatives (TN*), *False Negatives (FN*),
- *True Positive Rate (TPR)* [17, 18]

$$TPR = \frac{TP(number\ of\ true\ positives)}{P(total\ number\ of\ positive\ tests)}$$

- *False Positive Rate (FPR)* [17].

$$FPR = \frac{FP(number\ of\ false\ positives)}{N(total\ number\ of\ negative\ tests)}$$

- *True Negative Rate(TNR)* [17, 18].

$$TNR = \frac{TN\ (Number\ of\ true\ negative)}{N\ (total\ number\ of\ negatuve\ tests)}$$

- *False Negative Rate(FNR)* [17, 18].

$$FNR = \frac{FN(Number\ of false\ negatives)}{P(Total\ number\ of\ positive\ tests)}$$

- *Accuracy:* is the proximity of measurement results to the true value[19].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- *Youden Index*:

The score produced by OWASP Benchmark is the ***Youden index*** which is a standard method that summarises test set accuracy [6]. OWASP benchmark accuracy score is the normalized distance from the random guess line which is the difference between a scanner's TPR and FPR**.** Since WAVSEP benchmark uses accuracy and OWASP benchmark uses Yuden index, we therefore converted the benchmarks results to **score** for comparison.

$$Score = \text{TPR} - \text{FPR}$$

C. Scanners Selection

Our review of open source web vulnerability scanners revealed that Arachi and ZAP were best suited for this study considering their regular update, number of contributors and their popularity as confirmed in [20]. Although not considered as better as commercial scanners, some open source scanner features such as Crystal Report (RPT) interface have the potential of making these (particularly Arachni) a must-have scanner in Software as a Service (SAAS) multi-product environment [20]. Additionally, regardless of the size of the application scanned, number of threads and even against an easy target, Arachni appears to produce consistent results" [20]. ZAP, on the other hand, has also proven to be ideal for developers and functional testers who are both experienced and new to penetration testing [21].

D. Testing Environment Design.

We created a testing environment consisting of a local area network of two computers with one acting as a target and the other as the attacking computer. All applications necessary to perform the benchmarking which include OWASP Benchmark, OWASP ZAP, and Arachni, were installed.

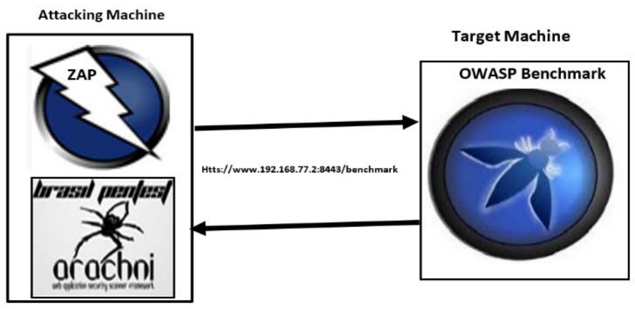


Figure 1: Experiment environment and scanners evaluation process

We obtained the benchmarking results by first executing the scanners against OWASP Benchmark. The scanners results were then used to generate an XML file that OWASP benchmark scorecard generator used to create scorecards. The generated scorecards were examined to draw conclusions on the performance of the scanners (see table 1). The benchmarking results of each scanner were discussed and compared to each other. Then, both scanners results were compared to results from a previous study that have used the WAVSEP benchmark to evaluate these scanners.

The benchmarking process comprised of three significant steps:

Step1. Setting the chosen scanner (Arachni and ZAP) to attack OWASP Benchmark. This was an essential step as it subjected the scanner to the Vulnerability test cases within benchmark and generated reports that we used to measure the scanners' performance using true positive and false positive, true negative and false negative metrics.

For ZAP, first, we have run a spider on the target (OWASP Benchmark) to discover all the resources (URLs) that are available in the target before launching the attack, then launched the attack using the 'Active Scan' command.

For Arachni, using the command line interface, we navigated to the bin folder then executed run Arachni command while specifying the target URL, the checks to be executed and the report name.

*Step 2***.** For Arachni, the command line interface generated a. Afr (Arachni Framework Report) report. This report was then used to produce other reports in different formats including HTML and XML. For this study, the XML report was the most needed to generate benchmark scorecards. On the other hand, when Arachni or ZAP Web interface was used, at the end of a successful scan, the scanners automatically generated reports in different formats that could be downloaded from provided links.

Step 3. The XML report was then copied back into results folder in OWASP Benchmark, then the command ***createScoreCards.bat*** (for Windows) or ***createscorecards.sh*** (for Linux) was executed to generate benchmark results known as Scorecards.

For the accuracy of the results, multiple scans were run with one scan targeting all categories and then each category separately. This method was applied more to obtain the Arachni results.

Five vulnerability categories were considered for evaluating the effectiveness of the scanners. These include Command Injection, LDAP Injection, SQL Injection, Cross Site Scripting and Path Traversal [22-25]. These categories were chosen in consideration of their criticality.

On each of the categories mentioned above and each scanner, OWASP benchmark applied some metrics including TP, FN, TN, FP, TPR, and FNR [17, 26] to obtain the most appropriate measures to score each scanner, to promote a reasonable interpretation of results and draw sound conclusions. We, therefore, executed the command ***createScoreCards*** to instruct OWASP benchmark to produce scorecards that highlight the overall performance of each scanner in the selected categories.

VI. EXPERIMENTAL RESULTS

We have organized our results into two Subsections. In Subsection A, we compare the OWASP benchmark results of the two scanners in chosen vulnerability categories. Then, in Subsection B, these (OWASP benchmark results) are then compared with the existing WAVSEP results.

A. *Scanners Comparative Evaluation Results*

After using Arachni and ZAP to scan the OWASP benchmark test cases, we observed the scanning behavior and their detection accuracy. We then compared these scanners accordingly. In this section, we present and discuss the scanners' benchmarking results comparatively.

1. Speed. After scanning the targeted OWASP benchmark test cases and recorded the elapsed scanning time, we observed that Arachni took around 10 hours and 30 minutes to scan each category whereas ZAP took around six hours and a half per category.

	Command Injection		LDAP Injection		SQL Injection		Cross Site Scripting		Path Traversal	
	OWASP ZAP	ARACHNI	OWASP ZAP	ARACHNI	OWASP ZAP	ARACHNI	OWASP ZAP	ARACHNI	OWASP ZAP	ARACHNI
TP	41	39	8	20	158	136	158	136	0	0
FN	85	87	19	19	114	136	114	136	133	133
TN	125	125	32	32	224	227	224	227	135	135
FP	0	0	0	0	8	5	8	5	0	0
TPR%	32.54	30.96	29.63	74.07	58.09	20	58.09	20	0	0
FPR%	0	0	0	0	3.45	2.16	3.45	2.16	0	0
Score %	33	31	30	74	55	48	76	64	0	0

Table 1:Scanners Detection score results for CMDI, LDAPI, SQLI, XSS and Path Traversal

2. Quantitative Measures. Based on OWASP benchmark test cases, six metrics were calculated to evaluate the effectiveness of the two scanners in detecting Command Injection, LDAP Injection, XSS, SQL Injection, and Path Traversal attacks. Table 1 shows the detailed summary of the scanners results in the calculated metrics.

The obtained experimental results (in table 1) has allowed us to get an overview of the performances of Arachni and ZAP related to Command Injection, LDAP Injection, SQL injection, Cross-Site Scripting and Path Traversal. We then gave a close comparison of the two scanners performance in the five chosen categories (See figure 2)

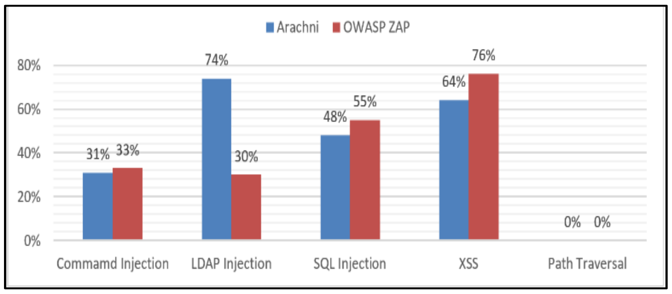


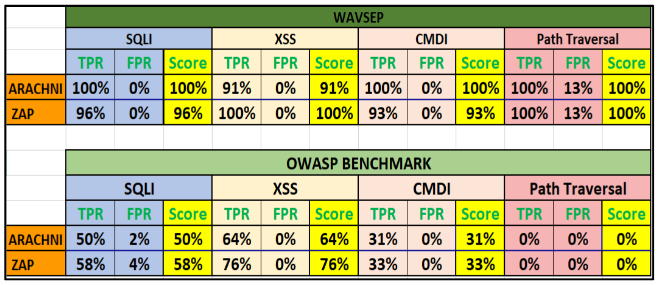
Figure 2: Side by side Comparison of OWASP benchmark Scores of Arachni and ZAP in CMDI, LDAPI, SQLI, XSS and Path Traversal

After producing the OWASP benchmark scorecard for each scanner in the five categories, we compared their performance results. Based on the OWASP benchmark reported scores, Arachni had the highest score of 74% in LDAP injection whereas ZAP scored 30%. However, in Command Injection, SQL Injection and XSS categories ZAP outperformed Arachni with the score of 33%, 55%, and 76% respectively. Although each scanner outperformed the other in some categories, we considered the difference in the scores for a better evaluation of their performance in each of the categories. This difference, therefore, shows that although there is a need for both scanners to uplift their performance in their losing categories, it is evident that more work is needed in raising both scanners performance in Path Traversal and ZAP in LDAP Injection.

B. Comparison of OWASP benchmark with WAVSEP benchmark

Arachni and ZAP have been evaluated before, and the WAVSEP benchmark was used in for benchmarking in these studies. In contrast, our study has evaluated these scanners based on OWASP benchmark. To highlight the importance of using a variety of benchmarks to get an overall conclusion in the evaluation of the effectiveness of web application vulnerability scanners, we have compared the obtained OWASP Benchmark results of Arachni and ZAP to a previous study that have evaluated these canners based on WAVSEP benchmark. We, therefore, chose the latest study by Shay Chen [8] for this purpose. Our choice of Shay Chen's study was based on the accuracy of his results, and his reputation as the author of WAVSEP benchmark. Additionally, his benchmarking results have never been contrasted with results based on other benchmarks.

The categories including SQLI, XSS, CMDI and Path Traversal were considered for benchmarks results' comparison purposes. Although LDAP category was examined in our experiments, it was not included in the comparison because it was not examined in Chen's study.



	WAVSEP											
	SQLI			XSS			CMDI			Path Traversal		
	TPR	FPR	Score	TPR	FPR	Score	TPR	FPR	Score	TPR	FPR	Score
ARACHNI	100%	0%	100%	91%	0%	91%	100%	0%	100%	100%	13%	100%
ZAP	96%	0%	96%	100%	0%	100%	93%	0%	93%	100%	13%	100%
	OWASP BENCHMARK											
	SQLI			XSS			CMDI			Path Traversal		
	TPR	FPR	Score	TPR	FPR	Score	TPR	FPR	Score	TPR	FPR	Score
ARACHNI	50%	2%	50%	64%	0%	64%	31%	0%	31%	0%	0%	0%
ZAP	58%	4%	58%	76%	0%	76%	33%	0%	33%	0%	0%	0%

Table 2: Arachni and ZAP OWASP benchmark and WAVSEP benchmark comparison

Our examination of the results in Table 2 demonstrated that there was some similarity in the performance pattern of the scanners in some categories such as XSS and Path Traversal. However, there were significant variation in detection rate and dissimilarities of scanners performance in some other categories such as SQLI. This variation was verifiable by examining WAVSEP benchmark results of XSS category which showed that ZAP had a 100% accuracy score and Arachni 91% whereas OWASP benchmark results indicated that ZAP scored 76% and Arachni 64% in the same category. While both Arachni and ZAP had a similar score of 100% detection rate in Path Traversal WAVSEP benchmark results, they scored 0% in the same category in OWASP benchmark results. In SQLI on the other hand, OWASP benchmark results indicated that ZAP had performed better than Arachni with 58% and 50% respectively whereas WAVSEP benchmark results showed the opposite (ZAP 96% and Arachni 100%). This differences in results, however, was explained by the fact that OWASP benchmark results were obtained from the latest version of ZAP (2.7) while Chen's study examined the previous version of ZAP (2.6).

Moreover, our results (OWASP benchmark) have demonstrated that the current version of ZAP has improved compared to its predecessor in some categories (e.g., in XSS and LDAP, V2.7 scored 76%, and 30% whereas V2.6 scored 29%, and 0% respectively). Nevertheless, there was still much difference in the SQLI score in OWASP benchmark results and WAVSEP benchmark results with a score of 100% for Arachni and 96 % for ZAP in WAVSEP results, while in OWASP benchmark results ZAP scored 58% and Arachni scored 50%. The interesting part, however, was that the differences in the scanners performance scores in OWASP benchmark results and WAVSEP results were both averaging 3.5%.

In other categories, ZAP outperformed Arachni with a 100% and 76% score as compared to 91% and 64% in XSS in WAVSEP benchmark results and OWASP benchmark results respectively. Additionally, in SQLI there was 8% and 4 % performance difference in OWASP benchmark and WAVSEP benchmark results respectively.

After a general discussion of the scanners results from OWASP benchmark and WAVSEP benchmark, we compared these in each of the chosen categories side-by-side.

1) Command Injection

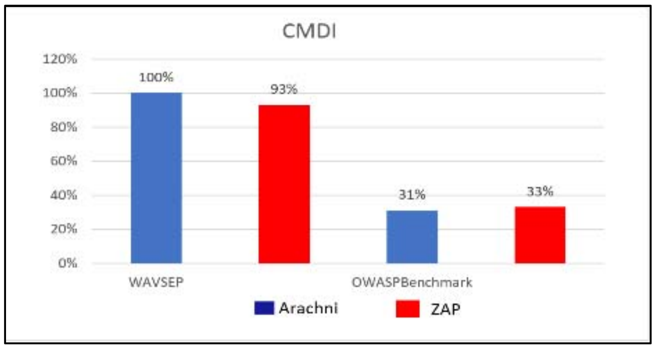


Figure 3: comparison of Arachni and ZAP CMDI score results in OWASP benchmark and WAVSEP benchmark

Based on the comparison results in figure 3, Arachni outperformed ZAP in WAVSEP benchmark results with 100% and 93% detection rates respectively, whereas the opposite occurred in OWASP benchmark results with ZAP scoring 33% and Arachni 31%. Although the scanners performance differences are not significant for both WAVSEP and OWASP benchmarks (with a difference of 7% and 2% respectively), WAVSEP benchmark detection rate for both scanners is three times higher than OWASP benchmark with an average of 96.5% and 32 % respectively.

2) SQL Injection

The comparison results of SQLI category indicated a contrast in the detection rate between OWASP benchmark and WAVSEP benchmark results.

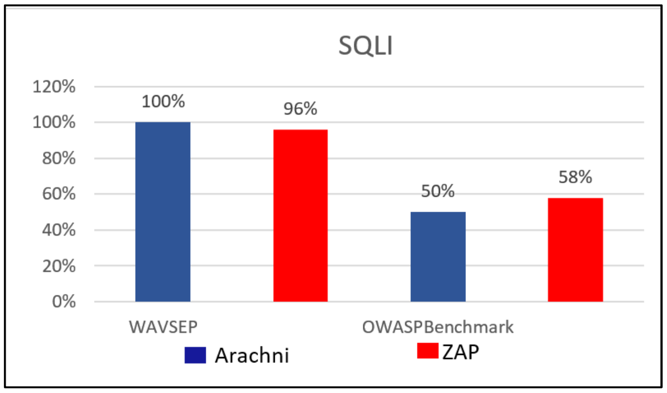


Figure 4: comparison of Arachni and ZAP SQLI score results in OWASP benchmark and WAVSEP benchmark

Arachni outperformed ZAP by 4% in WAVSEP benchmark results whereas OWASP benchmark results indicated that ZAP outperformed Arachni by 8% in this category. Although it was clear that Arachni outperformed ZAP in the existing WAVSEP benchmark results with a score of 100% and 96% respectively, we considered OWASP benchmark results. This was because OWASP benchmark examined the latest version of ZAP whereas existing WAVSEP benchmark study examined an older version of ZAP. Furthermore, our discussion of OWASP benchmark results (see Section IV.B) has confirmed that there has been a significant improvement in the examined version of ZAP as compared to previous versions.

3) Cross Site Scripting

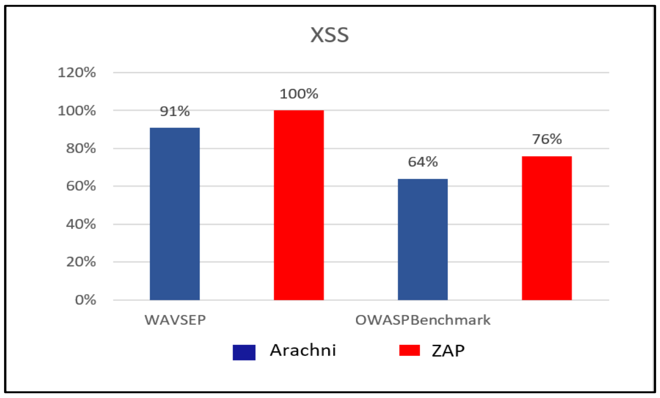


Figure 5: comparison of Arachni and ZAP XSS score results in OWASP benchmark and WAVSEP benchmark

In this category, ZAP performed better than Arachni both in WAVSEP benchmark results and OWASP benchmark results. However, the differences in detection rates were apparent in both vulnerability scanners in both benchmarks results with ZAP scoring 100% and 76% in WAVSEP benchmark and OWASP benchmark respectively. The same occurred with Arachni results which were 91% in WAVSEP benchmark results and 64% in OWASP benchmark results.

4) Path Traversal

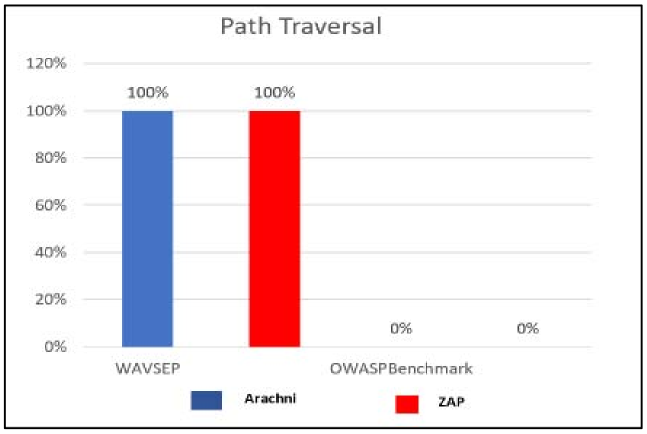


Figure 6: comparison of Arachni and ZAP Path Traversal score results in OWASP benchmark and WAVSEP benchmark

The strictness of benchmarks test cases was apparent in the comparison results of this category. As it can be seen, although both scanners scored 100% detection rate in WAVSEP benchmark results, the opposite occurred in the OWASP benchmark results with both scanners scoring 0% detection rate in the same category as shown in figure 6 above.

V. CONCLUSIONS AND RECOMMENDATIONS

The results of our comparative evaluation of the scanners confirmed again that scanners perform differently in different categories. Therefore, no scanner can be considered an all-rounder in scanning web vulnerabilities. However, combining the performances of these two scanners in both benchmarks, we concluded that ZAP performed better than Arachni in SQLI, XSS and CMDI categories. Arachni, on the other hand, performed much better in LDAP category.

There were considerable variations in the performances of these two scanners between the OWASP benchmark and the WAVSEP benchmark. Specifically, our results of benchmarks comparison revealed that for both scanners and all the four vulnerability categories compared, the scores under the WAVSEP benchmark were much higher than those under the OWASP benchmark. This reflects that the OWASP benchmark is more challenging than the WAVSEP benchmark in these four vulnerability categories. Therefore, we recommend that, if a scanner is to be evaluated on these four vulnerability categories, the OWASP benchmark should be chosen as the main target, while the WAVSEP benchmark can be used as a secondary target to complement the evaluation results.

REFERENCES

[1] CoreSecurity. *What is Penetration Testing?* Available: https://www.coresecurity.com/content/penetration-testing, 2018.

[2] K.Reintjes, "A benchmark approach to analyse the security of web frameworks," Master, Computer Science, Radboud University Nijmegen, Nijmegen, Netherlands, 2014.

[3] E. Tatlı and B. Urgun, *WIVET—Benchmarking Coverage Qualities of Web Crawlers* vol. 60, 2016.

[4] IBM. *The Most Comprehensive Web Application Security Scanner Comparison Available Marks AppScan Standard as the Leader (Again)*. Available: http://blog.watchfire.com/wfblog/2012/08/the-most-comprehensive-web-application-security-scanner-comparison-available-marks-appscan-standard-as-the-leader.html, 2012.

[5] Darknet. *wavsep-web-application-vulnerability-scanner-evaluation-project*. Available: https://www.darknet.org.uk/2011/09/wavsep-web-application-vulnerability-scanner-evaluation-project/, 2017.

[6] OWASP. *OWASP Benchmark*. Available: https://www.owasp.org/index.php/Benchmark, 2017.

[7] M. El, E. McMahon, S. Samtani, M. Patton, and H. Chen, "Benchmarking vulnerability scanners: An experiment on SCADA devices and scientific instruments," in *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2017, pp. 83-88.

[8] S. Chen. *Evaluation of Web Application Vulnerability Scanners in Modern Pentest/SSDLC Usage Scenarios*. Available: http://sectooladdict.blogspot.com/2017/11/wavsep-2017-evaluating-dast-against.html, 2017.

[9] S. E. Idrissi, N. Berbiche, F. G. and, and M. Sbihi, "Performance Evaluation of Web Application Security Scanners for Prevention and Protection against Vulnerabilities.pdf," *International Journal of Applied Engineering Research,* vol. 12, pp. 11068-11076, 2017.

[10] Y.Smeets, "Improving the adoption of dynamic web security vulnerability scanners," Computer Science, In Dei Nomine Feliciter, 2015.

[11] M. Alsaleh, N. Alomar, M. Alshreef, A. Alarifi, and A. Al-Salman, "Performance-Based Comparative Assessment of Open Source Web Vulnerability Scanners," *Security and Communication Networks,* vol. 2017, pp. 1-14, 2017.

[12] Y. Makino and V. Klyuev, "Evaluation of web vulnerability scanners," in *2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, 2015, pp. 399-402.

[13] Sarosys.LLC. *Arachni Web Application Security Scanner Framework*. Available: http://www.arachni-scanner.com/Sarosys LLC, 2017.

[14] OWASP. *OWASP Zed Attack Proxy Project*. Available: https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project, 2018.

[15] P. J. Fleming and J. J. Wallace, "How not to lie with statistics: the correct way to summarize benchmark results," *Communications of the ACM,* vol. 29, pp. 218-221, 1986.

[16] A. Baratloo, M. Hosseini, A. Negida, and G. El Ashal, "Part 1: Simple Definition and Calculation of Accuracy, Sensitivity and Specificity," *Emergency,* vol. 3, pp. 48-49, 2015.

[17] N. Antunes and M. Vieira, "On the Metrics for Benchmarking Vulnerability Detection Tools," in *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2015, pp. 505-516.

[18] J. S. Akosa, "Predictive Accuracy: A Misleading Performance Measure for Highly Imbalanced Data," 2017.

[19] Exsilio.Solutions. *Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures*. Available: https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/, 2018.

[20] S. Chen. *Price and Feature Comparison of Web Application Scanners*. Available: http://sectoolmarket.com/price-and-feature-comparison-of-web-application-scanners-unified-list.html, 2017.

[21] ToolsWatch. *2016 Top Security Tools as Voted by ToolsWatch.org Readers*. Available: http://www.toolswatch.org/2018/01/black-hat-arsenal-top-10-security-tools/.

[22] OWASP. *Cross Site Scripting*. Available: https://www.owasp.org/index.php/Cross-site_Scripting_(XSS), 2016.

[23] Microsoft. *Establishing an LDAP Session*. Available: https://msdn.microsoft.com/en-us/library/aa366102(v=vs.85).aspx, 2018.

[24] OWASP. *SQL Injection*. Available: https://www.owasp.org/index.php/SQL_Injection, 2016.

[25] PortSwigger_Ltd. *SQL injection*. Available: https://portswigger.net/kb/issues/00100200_sql-injection, 2018, 2018.

[26] GitHub.Inc. *OWASP Benchmark*. Available: https://github.com/OWASP/Benchmark/compare/1.2beta...master, 2018, February.