Vulnerability Analysis and Effectiveness of OWASP ZAP and Arachni on Web Security Systems

Yudhi Arta, Anggi Hanafiah, Nesi Syafitri, Panji Rachmat Setiawan, Yudhistira Hadi Gustianda

Department of Informatic Engineering, Engineering Faculty, Universitas Islam Riau
{yudhiarta, anggihanafiah, nesisyafitri, panji.r.setiawan, yudhistirahadi}eng.uir.ac.id

Abstract. The importance of using a security scanner to find web application weaknesses before they are released is very beneficial for the continuity of an organization. In this study, we analyze web security using Awasp and Arachni by testing it on web applications that reside in organizations. This research was conducted based on several studies regarding web security, but we see that some of these studies have not been able to measure the effectiveness of scanners in a measurable way. the results of OWASP ZAP got a score of 75.61 and Arachni with a score of 62.20%. By using these benchmarks, the results of scanner analysis become more measurable and can be assessed statistically so that the organization can take steps to overcome these security holes.

Keywords: Owasp ZAP, Arachni, Security, Benchmark, Scanner

1 Introduction

Evaluation of Information Technology infrastructure security by exploiting weaknesses in a system is called Penetration Testing. Penetration Testing was first implemented in the 1970s by the United States Department of Defense with the aim of uncovering security issues in computer systems to protect systems from digital crimes so that system security issues can be corrected before crimes occur. With the increasing popularity of computers and the ability to exchange information on the internet, the challenge of protecting information on the internet is also increasing[1], [2]. Because of this, in early 1965 experts in the field of computer security issued a warning about the possibility of attempted breaches of information security on the internet. About 15000 government, business analysts and experts in the field of computer security discussed this issue and came to the conclusion of implementing penetration testing. A collection of experts from NASA, CIA, cyber security experts and academics was formed. The team demonstrates the utility of penetration testing as a tool for evaluating computer system security. Currently, hacker techniques are increasingly sophisticated, especially with the increasing complexity of the technology used to develop web applications, penetration testing is becoming even more important as a method for evaluating computer security using a vulnerability scanner[3]–[5].

However, with the availability of many scanners circulating on the internet, it becomes very difficult to evaluate which scanner is the most effective and efficient in evaluating a web application. What's more, each scanner has its own advantages and disadvantages so that sometimes it is not enough with just one scanner to evaluate a web application[6]. Therefore we conducted this research to analyze the two most popular scanners namely OWASP ZAP and Arachni, which of these two scanners is more accurate in the evaluation process, and analyze the advantages and disadvantages of each scanner. We hope that with this research, web developers or penetration testers can easily choose which scanner to choose and under what conditions these two scanners are most effective and efficient to use[7], [8].

2 Research Methodology

The right method is needed to carry out the analysis and evaluation of the scanners studied in this study. The following are the stages of the method used:

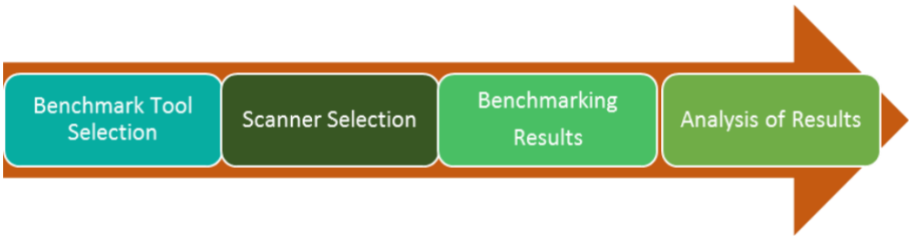

Fig. 1. Methodology

The importance of using a vulnerability scanner to find weaknesses in web applications before they are released has been realized by many organizations. This has been demonstrated in a study entitled A Case Study on Web Application Vulnerability Scanning Tools[9]. With the increasing number of cybercrimes, this research has tested several scanners that can be used to detect weaknesses in web applications that can easily be missed if you only use manual testing. This study explains that while using a scanner for testing is essential, these scanners vary in performance and provide different results depending on the configuration settings and how often the scanner is updated. This study also adds that the efficiency of a scanner can be judged by how many vulnerabilities it can detect and this also depends on how many plugins are available for the scanner. In choosing a scanner, this study also suggests using paid scanners because they are updated more often than open source scanners. However, a study entitled Price and Feature Comparison of Web Application Scanners said that several open source scanners such as Arachni have started to compete with paid scanners in terms of effectiveness[10], [11]. In addition, this study also suggests that to get better scanning results, scan using different scanners, use different settings and scan at other times to take advantage of the updates to these scanners. However, researchers realize that more research is needed to evaluate the effectiveness of existing scanners. Lately there have been many researchers who are interested in conducting research related to scanner evaluation.

For example, in a study entitled Using Web Security Scanners to Detect Vulnerabilities in Web Services they evaluated the vulnerability detection capabilities of four commercial scanners (Webinspect, Appscan, WSDigger, and Wsfuzzer)[12], [13]. They conducted an experiment using 300 popular web applications, and found that the four scanners used generated false positives between 35% and 40% during the scanning process.

Then in another study entitled Studying Open Source Vulnerability Scanners For Vulnerabilities in Web Applications they evaluated the vulnerability detection capabilities of three open source scanners (w3af, Skipfish, and OWASP ZAP) on the Damn Vulnerable Web Application (DVWA)). In this study, they concluded that OWASP ZAP has better performance than other scanners[14], [15]. Recognizing the difficulties faced by companies in selecting scanners, a study entitled Web Application Security Tools Analysis has looked for ways that can be used to deal with this problem. This research is continued by identifying the factors that cause vulnerabilities in web applications and the reasons why it is so difficult to eradicate existing vulnerabilities in web applications[16], [17]. For each existing vulnerability, this study suggests which scanners are suitable for each vulnerability. For vulnerabilities like XSS and SQL Injection, the research above suggests:

1. Netcraft as a scanner that can be used to detect the above vulnerabilities because this scanner is able to collect important footprint information related to the destination domain.
2. XSSer is a framework that can be used to detect vulnerabilities related to CSS.
3. OWASP Xenotix XSS supports manual and automatic detection of XSS vulnerabilities.
4. And several scanners for SQL Injection: SQL Inject Me, SQLninja, and Havij.

Although this research has tried several ways to deal with the above problems by suggesting using more than one scanner for several vulnerabilities, this research has concluded that determining the effectiveness of a scanner is still a challenge for companies or web developers. So from some of the problems above, research is needed to find the method needed to determine the effectiveness of a web vulnerability scanner. And that's why we decided to conduct this research to analyze between the two popular scanners used to evaluate web applications, namely OWASP ZAP and Arachni, which scanner is more effective for certain vulnerabilities.

SAST is a code-based web application test that can be done manually or by using a tool for code analysis to find vulnerabilities in the application's source code, or it can also be called White Box Testing. However, it is difficult to do to find all the vulnerabilities that exist by analyzing the source code, especially if the application is very complex[18]. In addition, knowing the internal structure, design and implementation of the application can be a barrier for testers to find vulnerabilities.

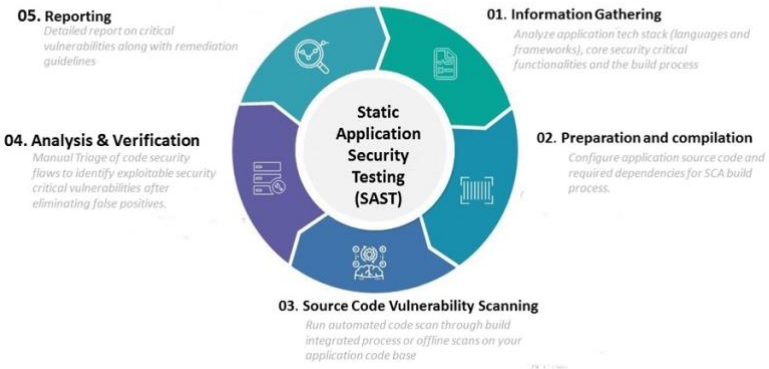

Fig. 2. Static Application Security Testing (SAST)

DAST is a process for finding vulnerabilities in web applications without knowing the internal structure, design and implementation of the application. This method can also be called Black Box Testing or Penetration Testing. Fuzzing, scraping, and crawling are some of the techniques used in this method to find vulnerabilities in web applications. (OWASP ZAP and Arachni use DAST as their method[18], [19].

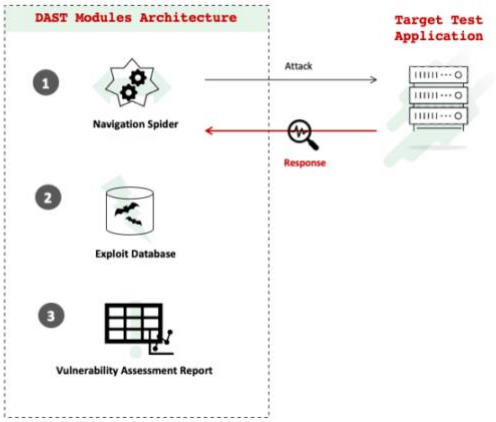

Fig. 3. Dynamic Application Security Testing (DAST)

IAST is a combination of SAST and DAST. Designed to combine the two methods, IAST takes advantage of the advantages of each method and therefore helps in minimizing the weaknesses of each method. This method can also minimize errors in detecting vulnerabilities in web applications when evaluating using the two methods above (SAST and DAST) by confirming with each method. IAST does this by placing

agents into web applications which will be evaluated for monitoring and analysis in real time[19], [20].

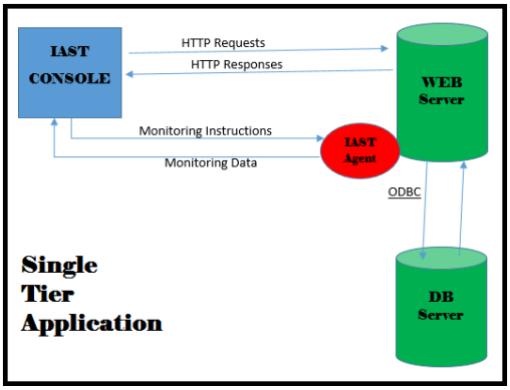

Fig. 4. Interactive Application Security Testing (IAST)

3 Result and Discussion

To evaluate scanners, applications that have vulnerabilities (benchmarks) needed to perform testing of scanners are needed. The right method for choosing a benchmark is to look at previous studies related to this research with the aim of understanding the procedure for benchmarking and also to find out what benchmarks are available. What's more, by looking at previous studies related to this research, we can choose the right scanner to study and the right benchmark as a benchmark for testing. There are several benchmarks available, including the OWASP Benchmark and the Web Application Vulnerability Scanner Evaluation Project (WAVSEP). In this study, we used the OWASP Benchmark as a benchmark. Scanner Selection Although many previous studies have examined paid and open source scanners, in this study we focused on two open source based scanners namely OWASP ZAP and Arachni. All required applications will be installed on one computer and the testing process against OWASP ZAP and Arachni will be carried out on one computer and OWASP Benchmark will be installed as localhost as a benchmark. Benchmarking Results The results of the benchmarking are obtained by executing the selected scanner against the OWASP Benchmark. The results of the scanning will then be used to produce a file with an XML extension which will then be inputted into OWASP Benchmark to produce the results of the scanning process which will then draw conclusions from the performance of the selected scanner.

3.1 Analysis of Results

The results of benchmarking from each scanner will be analyzed and the comparison will be seen from each scanner. Then it will be compared in which vulnerabilities

each scanner excels in carrying out the scanning process and which scanner finds more vulnerabilities for each vulnerability scanned. The analysis process will be calculated using True Positive, False Positive, True Negative, False Negative calculations. Here's the definition of each metric:

- True Positive (TP): True Positive is the number of cases that are positive and detected as positive.
- False Positive (FP): False Positive is the number of cases which were negative but detected as positive.
- True Negative (TN): True Negative is the number of cases that are negative and detected as negative.
- False Negative (FN): False Negative is the number of positive cases but detected as negative.
- True Positive Rate (TPR): is the value at which the scanner correctly identifies and detects the correct vulnerability (positive cases). This value is obtained by dividing the number of True Positives by the number of positive cases.
- False Positive Rate (FPR): is the value at which the scanner reports non-existent cases as positive. This value is obtained by dividing the number of Fasle Positives by the number of negative cases.
- True Negative Rate (TNR): is the value at which the scanner correctly ignores negative cases. This value is obtained by dividing the number of True Negatives by the number of negative cases.
- False Negative Rate (FNR): is the value at which the scanner fails to correctly identify and detect vulnerabilities (positive cases). This value is obtained by dividing the number of False Negatives by the number of positive cases.
- Accuracy: is a value to measure the percentage accuracy of the scanning results from a scanner. This value is obtained by dividing the number of True Positives and True Negatives by the number of positive and negative cases.

Based on the methodology that we have previously described, in this study we have prepared a case example from one of the methodological processes described above, namely the results of the benchmarking process. In this case, we use scanners that comply with the specifications above (OWASP ZAP and Arachni) and use benchmarks that comply with the specifications above (OWASP Benchmark). For this example, we will only scan for cross-site scripting (XSS) vulnerabilities. The following is the result of the benchmarking process that we have done:

Tool	Type	TP	FN	TN	FP	Total	TPR	FPR	Score
Arachni v1.5.1	DAST	153	93	209	0	455	62.20%	0.00%	62.20%
FBwFindSecBugs v1.4.0	SAST	3	243	209	0	455	1.22%	0.00%	1.22%
FBwFindSecBugs v1.4.3	SAST	1	245	209	0	455	0.41%	0.00%	0.41%
FBwFindSecBugs v1.4.4	SAST	3	243	209	0	455	1.22%	0.00%	1.22%
FBwFindSecBugs v1.4.5	SAST	246	0	78	131	455	100.00%	62.68%	37.32%
FBwFindSecBugs v1.4.6	SAST	246	0	78	131	455	100.00%	62.68%	37.32%
FindBugs v3.0.1	SAST	1	245	209	0	455	0.41%	0.00%	0.41%
OWASP ZAP v2.11.0	DAST	186	60	209	0	455	75.61%	0.00%	75.61%
OWASP ZAP vD-2015-08-24	DAST	71	175	209	0	455	28.86%	0.00%	28.86%
OWASP ZAP vD-2016-09-05	DAST	71	175	209	0	455	28.86%	0.00%	28.86%
PMD v5.2.3	SAST	0	246	209	0	455	0.00%	0.00%	0.00%
SonarQube Java Plugin v3.14	SAST	0	246	209	0	455	0.00%	0.00%	0.00%
VisualCodeGrepper v2.2.0	SAST	0	246	209	0	455	0.00%	0.00%	0.00%

Fig. 5. Benchmark Results

From the results above, the scanners we use are Arachni v1.5.1 and OWASP ZAP V2.11.0. It can be seen that OWASP ZAP has a score that is superior to Arachni, namely OWASP ZAP with a score of 75.61% and Arachni with a score of 62.20%, which means OWASP ZAP managed to find more vulnerabilities (True Positive) than Arachni, namely OWASP ZAP with 186 True Positives and Arachni with 153 True Positives. Both scanners have a True Negative of 209 and a False Positive of 0, in other words the two scanners manage to ignore all the fake vulnerabilities or traps that have been prepared by OWASP Benchmark, so that both scanners are perfect in the True Negative and False Positive categories.

In this discussion, we will compare the results of the Arachni and OWASP ZAP scans. In the comparison between Arachni and OWASP ZAP, several metrics for each scanner have been used as benchmarks and scores have been calculated, namely True Positive, False Negative, True Negative, False Positive, True Positive Rate, and False Negative Rate. Comparison of the results for each scanner can be seen in the table below.

Table 1. Comparison of Results of Arachni and OWASP ZAP

Scanner	TP	FN	TN	FP	TPR	FPR
Arachni	157	89	209	0	63.82%	0.00%
OWASP ZAP	186	60	209	0	75.61%	0.00%

In each of the categories in the table above, OWASP Benchmark applies the metrics that we have discussed above to obtain the most appropriate measurement values to assess each scanner and obtain appropriate analysis results and conclusions. That's why OWASP Benchmark also produces an assessment in the form of a scorecard that shows the results of the performance of each scanner in each category. The final value

of the OWASP Benchmark assessment (Score) is the percentage distance between the True Positive Rate and the False Positive Rate (Score = TPR - FPR). The performance results of each scanner in the cross-site scripting (XSS) vulnerability category can be seen in the diagram below.

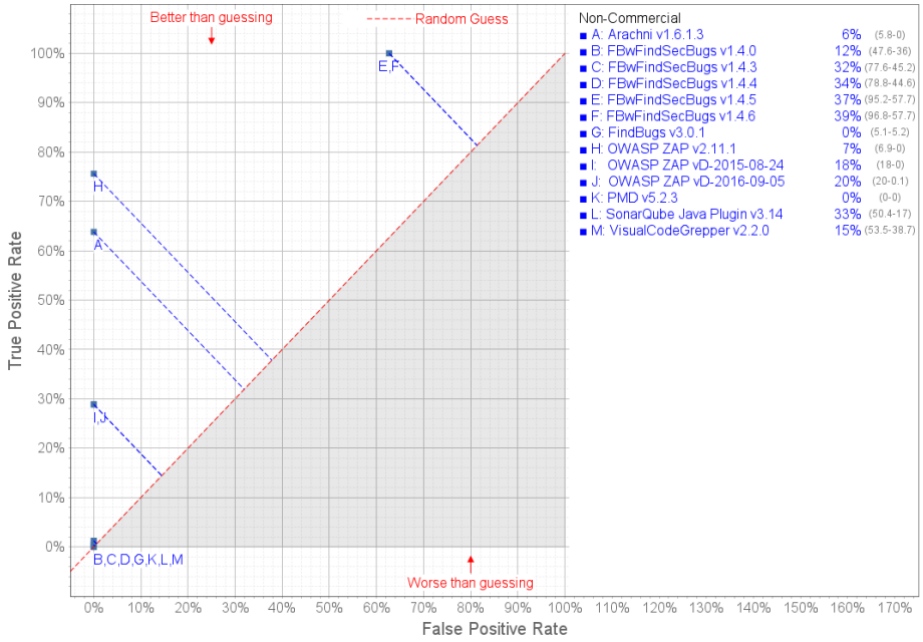

Fig 5. Arachni and OWASP ZAP Performance Charts

4 Conclusion

Based on the analysis in this study, it can be concluded that OWASP ZAP and Arachni have different scanning procedures and processes for each scanner. To perform a performance test on a scanner, accurate benchmarks are needed to perform analysis such as the OWASP Benchmark. OWASP ZAP has better performance than Arachni in the XSS vulnerability category. We can see that in the cross-site scriptimg (XSS) vulnerability category OWASP ZAP has better performance than Arachni with a detection accuracy value of 76% compared to Arachni's accuracy value of 64%. And we can also see the difference in percentage distance between the detection accuracy values of OWASP ZAP and Arachni that OWASP ZAP is 12% superior to Arachni.

References

[1] L. Gashi, A. Luma, and A. Aliu, "A comprehensive review of cybersecurity perspective for Wireless Sensor Networks," in *2022 International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, 2022, pp. 392–395.

[2] M. G. Cains, L. Flora, D. Taber, Z. King, and D. S. Henshel, "Defining cyber security and cyber security risk within a multidisciplinary context using expert elicitation," *Risk Anal.*, vol. 42, no. 8, pp. 1643–1669, 2022.

[3] E. O. Yeboah-Boateng, *Cyber-security challenges with smes in developing economies: Issues of confidentiality, integrity & availability (CIA)*. Institut for Elektroniske Systemer, Aalborg Universitet, 2013.

[4] G. P. Moynihan, "An executive information system: Planning for post-implementation at NASA," *J. Syst. Manag.*, vol. 44, no. 7, p. 8, 1993.

[5] Y. Arta, M. Ilhan, and A. Hanafiah, "Analisis Kebutuhan Keamanan Informasi Menggunakan Metode SQUARE Pada Aplikasi HRIS Studi Kasus: PT. XYZ," *CogITo Smart J.*, vol. 7, no. 1, pp. 61–73, 2021.

[6] V. Clincy and H. Shahriar, "Web application firewall: Network security models and configuration," in *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, 2018, vol. 1, pp. 835–836.

[7] M. Alfarizi and I. F. Ashari, "Vulnerability Analysis and Proven On The neonime. co Website Using OWASP ZAP 4 and XSpear," *JTKSI (Jurnal Teknol. Komput. dan Sist. Informasi)*, vol. 5, no. 2, pp. 75–81, 2022.

[8] A. Lathifah, F. B. Amri, and A. Rosidah, "Security Vulnerability Analysis of the Sharia Crowdfunding Website Using OWASP-ZAP," in *2022 10th International Conference on Cyber and IT Service Management (CITSM)*, 2022, pp. 1–5.

[9] N. I. Daud, K. A. A. Bakar, and M. S. M. Hasan, "A case study on web application vulnerability scanning tools," in *2014 Science and Information Conference*, 2014, pp. 595–600.

[10] S. Chen, "Price and feature comparison of web application scanners." Online, 2017.

[11] A. Siswanto, Y. Arta, E. A. Kadir, and Bimantara, "Text File Protection Using Least Significant Bit (LSB) Steganography and Rijndael Algorithm," in *Proceedings of International Conference on Smart Computing and Cyber Security: Strategic Foresight, Security Challenges and Innovation (SMARTCYBER 2020)*, 2021, pp. 205–213.

[12] M. Vieira, N. Antunes, and H. Madeira, "Using web security scanners to detect vulnerabilities in web services," in *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, 2009, pp. 566–571.

[13] A. A. Almutairi, S. Mishra, and M. AlShehri, "Web Security: Emerging Threats and Defense.," *Comput. Syst. Sci. Eng.*, vol. 40, no. 3, 2022.

[14] D. Sagar, S. Kukreja, J. Brahma, S. Tyagi, and P. Jain, "Studying open source vulnerability scanners for vulnerabilities in web applications," *IIOAB J.*, vol. 9, no. 2, pp. 43–49, 2018.

[15] 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, vol. 1, pp. 399–402.

[16] A. Alzahrani, A. Alqazzaz, Y. Zhu, H. Fu, and N. Almashfi, "Web application security tools analysis," in *2017 ieee 3rd international conference on big data security on cloud (bigdatasecurity), ieee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security (ids)*, 2017, pp. 237–242.

[17] S. M. Srinivasan and R. S. Sangwan, "Web app security: A comparison and

categorization of testing frameworks," *IEEE Softw.*, vol. 34, no. 1, pp. 99–102, 2017.

[18] J. Li, "Vulnerabilities mapping based on OWASP-SANS: a survey for static application security testing (SAST)," *arXiv Prepr. arXiv2004.03216*, 2020.

[19] F. Mateo Tudela, J.-R. Bermejo Higuera, J. Bermejo Higuera, J.-A. Sicilia Montalvo, and M. I. Argyros, "On Combining Static, Dynamic and Interactive Analysis Security Testing Tools to Improve OWASP Top Ten Security Vulnerability Detection in Web Applications," *Appl. Sci.*, vol. 10, no. 24, p. 9119, 2020.

[20] A. Seth, S. Bhattacharya, S. Elder, N. Zahan, and L. Williams, "Comparing Effectiveness and Efficiency of Interactive Application Security Testing (Iast) and Runtime Application Self-Protection (Rasp) Tools in A Large Java-Based System," *Available SSRN 4306114*, 2022.