World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:11, No:5, 2017

Benchmarking of Pentesting Tools

Esteban Alejandro Armas Vega, Ana Lucila Sandoval Orozco, Luis Javier García Villalba

***Abstract*—**The benchmarking of tools for dynamic analysis of vulnerabilities in web applications is something that is done periodically, because these tools from time to time update their knowledge base and search algorithms, in order to improve their accuracy. Unfortunately, the vast majority of these evaluations are made by software enthusiasts who publish their results on blogs or on non-academic websites and always with the same evaluation methodology. Similarly, academics who have carried out this type of analysis from a scientific approach, the majority, make their analysis within the same methodology as well the empirical authors. This paper is based on the interest of finding answers to questions that many users of this type of tools have been asking over the years, such as, to know if the tool truly test and evaluate every vulnerability that it ensures do, or if the tool, really, deliver a real report of all the vulnerabilities tested and exploited. This kind of questions have also motivated previous work but without real answers. The aim of this paper is to show results that truly answer, at least on the tested tools, all those unanswered questions. All the results have been obtained by changing the common model of benchmarking used for all those previous works.

***Keywords*—**Cybersecurity, IDS, security, web scanners, web vulnerabilities.

I. Introduction

IN 2015 there were 5334 attacks to web applications, of these cases a total of 908 confirmed exposure of sensitive data that managed the web application [1]. Web applications are present in almost every aspect of our daily lives. The use of web applications to access financial services, purchase products, government services or even interact with other people, are just a small sample of all the web applications which are used daily. These web applications manage, receive, send and store much of information which, in most cases, is personal and confidential, therefore, security within these web applications should be a number one priority.

Security verification must be present in all stages of the development cycle and especially in the final stages which are critical to the implementation and launch of the application. In these stages of development, it is fundamental to perform tests that verify failures and possible presence of vulnerabilities, in order to reduce the opportunity of a successful attack [2], preventing possible data loss and also economic losses.

There are many options in terms of automatic pentesting tools, which allow the review and detection of potential vulnerabilities in web applications, therefore, choosing the right tool is a essential task. The developer has full confidence in the results given from the used tool. Hence, those results influence on vulnerability that can reach the application once it is in production.

The use of penetration and dynamic analysis tools to improve the security in an web application reduces time and effort in the development cycle and allows to focus greater efforts on more complex safety tasks [3] and other stages of development cycle. Such tools are not trivial to set up, correctly, for those who are not familiar with this kind of software [4]. And because one of its biggest weaknesses is the presence of false positive results [3]. Therefore it is important to have accurate knowledge of the real capabilities of these tools.

The aim of this work is to evaluate most of the generated data over the interaction between the black-box tools and the analysed web applications. This is achieved by gathering execution time data, use of network resources, carried out attacks and also the alerts and vulnerabilities showed in the report. All this obtained information was contrasted and compared between the result of each of the tools used in this paper.

II. Previous Works

Previous studies have conducted several comparatives about the accuracy of dynamic analysis tools for vulnerabilities on web applications. In this section its reviewed the most relevant works related to this paper.

In [4] the authors compare OWASP ZAP [5] and Skipfish [6] tools, both free and widely used. These tools were run in order to evaluate vulnerabilities in two vulnerable web applications: Damn Vulnerable Web Application (DVWA) [7] and Web Application Vulnerability Scanner Evaluation Project (WAVSEP) [8]. Evaluation criteria includes the analysis of reports and a comparison of characteristics that each tool provides for the execution of the analysis. Results showed a favorable value in the accuracy for detecting vulnerabilities from the OWASP ZAP tool and a low value of false positives.

In [9] a group of 32 free analysis tools were compared by applying the Web Application Security Scanner Evaluation Criteria (WASSEC) [10] from Web Application Security Consortium. WASSEC establishes eight main criteria to be taken into account when assessing a dynamic analysis tool. Out of the eitght criteria, Command and Control and Reporting criteria were avoided in this study. From this analysis the best tool in four of the six evaluated criteria was W3AF 1.2-rev509 [11].

In [12] the performance of three dynamic analysis tools (Nessus, Acunetix and OWASP ZAP) were analyzed on two test applications developed by the researchers. The tests were developed in two stages. In the first step attacks on both applications are performed using the three tools with the same

Esteban Alejandro Armas Vega, Ana Lucila Sandoval Orozco and Luis Javier García Villalba are with the Group of Analysis, Security and Systems (GASS), Department of Software Engineering and Artificial Intelligence (DISIA), Faculty of Computer Science and Engineering, Office 431, Universidad Complutense de Madrid (UCM), Calle Profesor José García Santesmases, 9, Ciudad Universitaria, 28040 Madrid, Spain (e-mail: esarmas@ucm.es, asandoval@fdi.ucm.es, javiergv@fdi.ucm.es).

configuration. In the second stage attacks are performed with only the Nessus tool with different settings. From the results obtained, the time and accuracy of each tool is analyzed as well as every tool configuration. Besides the accuracy and traffic that each tool generates, the time it takes to run the analysis is also an important factor to consider.

Some previous studies have used the SNORT - Network Intrusion Detection and Prevention System (SNORT) [13] to detect SQL attacks on web applications. In [14] SNORT was used to detect SQL Injection attacks on applications such as DVWA. The purpose was to analyze the efficiency in detecting SQL Injection attacks by a group of new rules proposed.

The set of rules developed in [15] for SNORT allow to detect three different types of attacks: SQL Injection, XSS and Command Execution. The experiment consists in perform attacks on the DVWA application, which will be prevented by SNORT and the new set of rules developed.

In this paper the authors intend to apply a different approach than previous studies by introducing into the typical structure an intruders detection system IDS, that allows to obtain the attack requests of each analysis tool and contrast these requests with the corresponding report. Therefore, it would be possible to get information that will determine the efficiency of the tools at the time of analysis.

III. Testing Model

The system was run on two computers, one attacker and another as a server. Both are connected by a physical network and a network router, as shown in Fig. 1.

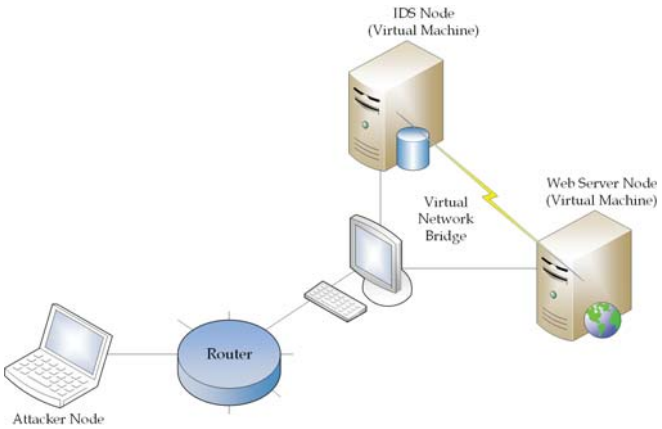

Fig. 1 Test Environment

A. System Tools

The tools used in this paper were OWASP ZAP, Acunetix WVS, HP WebInspect [16] and the Arachni Scanner [17]. These tools were chosen because all of them are well known and used in the vulnerability analysis of web applications. These tools also have been used in previous studies [4], [18], where those tools highlighted over the rest of tools.

All tools used in this work were configured with its standard scan profiles that each bring default, without setting any further particular feature.

B. IDS

To the evaluation of pentest tools the SNORT network intrusion detection and prevention system is used in this paper. SNORT has been chosen because it is free and open source.

SNORT has been configured with more than seventy-three thousand rules exclusively for HTTP-HTTPS traffic. All of this rules try to cover all possibilities of attack that each pentest tool can make during analysis.

C. Vulnerable Web Applications

In this work one vulnerable web application was used. Its documented and has supply detailed information on the location of each vulnerability. All this information allow to compare the obtained results more easily.

WackoPicko is a vulnerable web application developed by Adam Doupe and used in [19]. Its design was intended to simulate the behaviour of an regular not intentionally vulnerable web application. It acts as an application for sharing and selling images, contains different sections such as authentication page for entry to the site, an viewer section to see and comment any selected image, an upload section where the user can upload images into the site to share it. All the vulnerabilities present in WackoPicko are showed in Table I [20].

TABLE I
Vulnerabilities Inside of WackoPicko

Vulnerability	Quantity
Reflected Cross Site Scripting	3
Stored Cross Site Scripting	2
Stored SQL Injection	1
Reflected SQL Injection	1
Path Traversal	1
Command-line Injection	1
Remote File Inclusion	1
Parameter Manipulation	1
Logic Flaw	1
Weak username/password	1
Forceful Browsing	1
SessionID vulnerability	1

IV. Analysis and Results

Each tool has been use once and separately. In each single tests the reports generated by SNORT were compared. All of the vulnerabilities of the evaluated web application and the report generated by each of the pentesting tool have been taken into account. Fig. 2 shows the method that has been followed to check the accuracy of the scanners in this paper. A summary of the results of WackoPicko analysis can be observed in Table II.

A. WackoPicko Results

Acunetix WVS identify less than 50% of vulnerabilities presents in this web application. Furthermore, was not reported two critical vulnerabilities: Directory Traversal and Command-line Injection, presents in the web application and

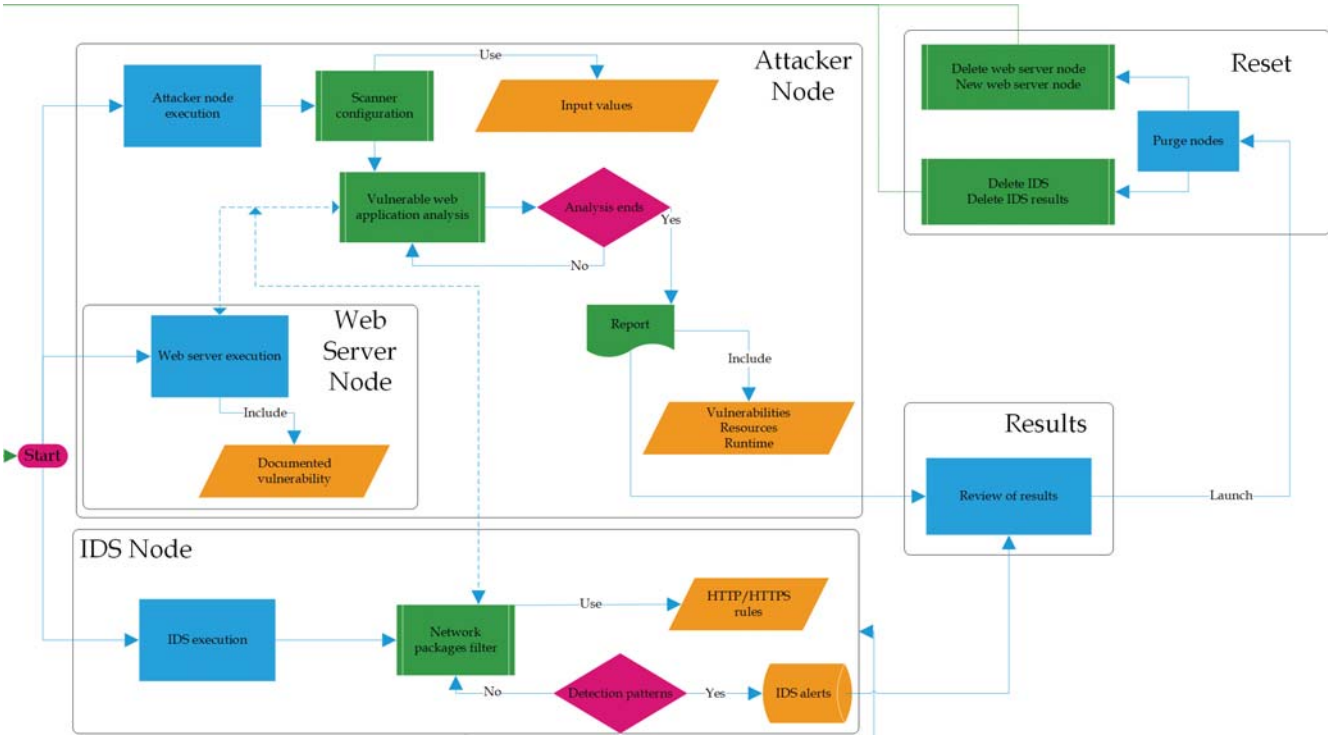

Fig. 2 Method followed to check the scanners accuracy

TABLE II
RESULTS OF ANALYSE WACKOPICKO

Vulnerability	WackoPicko	IDS	Acunetix WVS	IDS	OWASP ZAP	IDS	HP WebInspect	IDS	Arachni
Reflected Cross Site Scripting	3	X	5	X	6	X	5	X	7
Persistent Cross Site Scripting	2		1		2		-		6
Stored SQL Injection	1	X	1	X	2	X	1	X	1
Refected SQL Injection	1								
Path traversal	1	X	-	X	-	X	1	-	-
Command-line Injection	1	X	-	-	-	X	-	X	1
Remote File Inclusion	1	-	2	X	1	X	-	-	1
Parameter Manipulation	1	-	-	-	-	-	-	-	-
Logic Flaw	1	-	-	-	-	-	-	-	-
Weak username/password	1	-	1	-	-	-	-	-	-
Forceful Browsing	1	-	-	-	-	-	-	-	-
SessionID vulnerability	1	-	-			-	-	-	-

also vulnerabilities whitch had been attacked by Acunetix according to SNORT's alerts report. Acunetix could not determine the category of SQL Injection attacks. Thus, in the Table II all kind of SQL injections attacks are shown together.

OWASP ZAP reported fewer successes in vulnerabilities founds compared with the vulnerabilities that Acunetix WVS found. OWAS ZAP does not include in its report the presence of Directory Trave rsal vulnerability, despite of the fact that it was exploited according to the SNORT's alerts report. SNORT shows an attack to exploit that vulnerability during the analysis with OWAS ZAP.

HP WebInspect does not showed more accurancy than the other evaluated tools. SNORT shows that there were File Inclusion and Command-line injection attacks into the WackoPicko application, but HP WebInspect did not report the presence of such vulnerabilities, although WackoPicko has these two. Like the other tools HP WebInspect did not perform a search for at least all the vulnerabilities in the OWASP Top 10 2013 [21].

In the report presented by Arachni all vulnerabilities found were attacked, as reported by SNORT. Also there were no alerts in Arachni not reported by SNORT.

V. CONCLUSION

The approach proposed in this paper allowed to obtain details on the results that in previous work were not considered. The evaluated tools showed deficiencies to identifying vulnerabilities in the tested web applications.

Most of the tools that were used in this paper demonstrated that attacks were made and confirmed by SNORT, but in the final report of the tool was not considered as vulnerable, despite its existence in the application. In addition becomes clear that there was not carried out attacks in search of at least OWASP TOP 10 2013. This can be considered as

a disadvantage in these tools, mainly because its intention is to cover at least the vulnerabilities more important and documented and that according to reports obtained in this work was not done.

Although SNORT had thousands of rules for a wide range of possible attacks, in his report could not disaggregate fine grain type of attack, similar to each tool presents, in order to refine the results and be more accurate in comparison. It is important to develop new rules specifically allow SNORT to determine the type of attack not to generalize results. Each tool report contain basic information and helps the developer to identify faults in the application. The runtime of any tool black box is much less than the time it would take to analyze the application manually. To improve the analysis of these tools, it is important to conduct a study on the false positives obtained in each report, that this work was not done. Also it is essential to include rules in SNORT that allow specifically determine the type of attack and that no it generalizes when comparing results. Because of the results obtained, it is considered important to conduct more analysis using different tools and applications containing vulnerable closer to a real web application, as the case of WackoPicko features

ACKNOWLEDGMENT

This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 700326.

REFERENCES

[1] Verizon Enterprise. 2016 Data Breach Investigations Report. Report, Verizon Enterprise, July 2016.
[2] A. Sagala and E. Manurung. Testing and Comparing Result Scanning Using Web Vulnerability Scanner. *Advanced Science Letters*, 21(11):3458–3462, November 2015.
[3] P. Baral. Web Application Scanners: A Review of Related Articles. *IEEE Potentials*, 30(2):10–14, March 2011.
[4] Y. Makino and V. Klyuev. Evaluation of Web Vulnerability Scanners. In *Proceedings of the IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, volume 1, pages 399–402, Warsaw, PL, September 2015.
[5] The Open Web Application Security Project OWASP. OWASP Zed Attack Proxy Project. https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project, April 2016.
[6] Google. Google Code - Skipfish. https://code.google.com/archive/p/skipfish/, March 2016.
[7] RandomStorm. Damn Vulnerable Web Application (DVWA). http://www.dvwa.co.uk, March 2016.
[8] Google. Google Code - WAVSEP. https://code.google.com/archive/p/wavsep/, March 2016.
[9] F. A. Saeed. Using WASSEC to Analysis and Evaluate Open Source Web Application Security Scanners. *International Journal of Computer Science and Network*, 3(2):43–49, April 2014.
[10] Web Application Security Consortium. Web Application Security Scanner Evaluation Criteria WASSEC. http://goo.gl/aePtyC, April 2016.
[11] W3af. W3af - Open Source Web Application Security Scanner. http://w3af.org, Abril 2016.
[12] N. I. Daud, K. A. A. Bakar, and M. S. Md. Hasan. A Case Study on Web Application Vulnerability Scanning Tools. In *Proceedings of the Conference of Science and Information (SAI)*, pages 595–600, 2014.
[13] Snort - Network Intrusion Detection and Prevention System. https://www.snort.org/, Abril 2016.
[14] H. Alnabulsi, Md. R. Islam, and Q. Mamun. Detecting SQL Injection attacks using SNORT IDS. In *Proceedings of the 2014 Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE)*, pages 1–7. IEEE, Nov 2014.
[15] M. Dabbour, I. Alsmadi, and E. Alsukhni. Efficient Assessment and Evaluation for Websites Vulnerabilities using SNORT. *International Journal of Security and its Applications*, 7(1), 2013.
[16] HP. HP WebInsPect. Product Manual, HP, March 2015.
[17] Arachni. ARACHNI Web Application Security Scanner Framework. http://www.arachni-scanner.com, March 2016.
[18] F. A. Saeed. Using WASSEC to Evaluate Commercial Web Application Security Scanners. *International Journal of Soft Computing and Engineering (IJSCE)*, 4(1):177–181, March 2014.
[19] A. Doupé, M. Cova, and G. Vigna. Detection of Intrusions and Malware, and Vulnerability Assessment. In Christian Kreibich and Marko Jahnke, editors, *Proceedings of the 7th International Conference (DIMVA 2010)*, pages 111–131, Bonn, Germany, July 2010.
[20] A. Doupé. WackoPicko Vulnerable Website. https://github.com/adamdoupe/WackoPicko, March 2016.
[21] The Open Web Application Security Project OWASP. OWASP Top 10 - 2013 The Ten Most Critical Web Application Security Risks. Release, The Open Web Application Security Project OWASP, June 2013.

Esteban Alejandro Armas Vega He received a Informatics Engineering degree from the Polytechnic Institute "José Antonio Echeverria" of Havana (Cuba) in 2009. He holds a M.Sc. in Informatics Engineering from the University Complutense of Madrid in 2016. He is currently a Ph.D. student at the University Complutense of Madrid and a member of the research group GASS (http://gass.ucm.es), in the Department of Software Engineering and Artificial Intelligence (DISIA) of the Faculty of Computer Science and Engineering. His main research interests are Multimedia Forensic and Information Security.

Ana Lucila Sandoval Orozco She received a Computer Science Engineering degree from the Universidad Autnoma del Caribe (Colombia) in 2001. She holds a Specialization Course in Computer Networks (2006) from the Universidad del Norte (Colombia), and holds a M.Sc. in Research in Computer Science (2009) and a Ph.D. in Computer Science (2014), both from the Universidad Complutense de Madrid (Spain). She is currently a Research Assistant at Complutense Research Group GASS. Her main research interests are Computer Networks and Computer Security.

Luis Javier García Villalba He received a Telecommunication Engineering degree from the Universidad de Mlaga (Spain) in 1993 and holds a M.Sc. in Computer Networks (1996) and a Ph.D. in Computer Science (1999), both from the Universidad Politcnica de Madrid (Spain). Visiting Scholar at COSIC (Computer Security and Industrial Cryptography, Department of Electrical Engineering, Faculty of Engineering, Katholieke Universiteit Leuven, Belgium) in 2000 and Visiting Scientist at IBM Research Division (IBM Almaden Research Center, San Jose, CA, USA) in 2001 and 2002, he is currently Associate Professor of the Department of Software Engineering and Artificial Intelligence at the Universidad Complutense de Madrid (UCM) and Head of Complutense Research Group GASS (Group of Analysis, Security and Systems) which is located in the Faculty of Information Technology and Computer Science at the UCM Campus. His professional experience includes research projects with Hitachi, IBM, Nokia and Safelayer Secure Communications. His main research interests are Computer Networks and Computer Security.