International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 21 (2017) pp. 11068-11076
© Research India Publications. http://www.ripublication.com

Performance Evaluation of Web Application Security Scanners for Prevention and Protection against Vulnerabilities

[1] S. El Idrissi, [2] N. Berbiche,[3] F. Guerouate and [4] M. Sbihi

[1] PhD Student, LASTIMI Laboratory, Mohammadia School of Engineers, Mohammed V University in Rabat, Morocco.
[2,3,4] Research Professor, LASTIMI Laboratory and Professor in the Department of Computer Sciences, Higher School of Technology - Sale, Mohammed V University in Rabat, Morocco.

[1]Orcid ID: 0000-0001-7921-5300

Abstract

With the increasing development of the Internet, web applications have become increasingly vulnerable and exposed to malicious attacks which affect essential properties such as confidentiality, integrity or availability of information systems. To deal with these malicious threats, web application developers and IT security administrators have used the web application vulnerabilities scanners (WAVS) as scanning tools that regularly audit web applications to check for exploitable vulnerabilities. In today's market, a large number of web application scanning tools are available. Though these tools are available in market, the question is how efficient they are in addressing security concerns in web applications. The primary focus of this Article is to assess the effectiveness and performance of eleven scanners as far as vulnerability detection in web applications is concerned. This evaluation is multifunctional as it can be used to specify the degree of scanners 'efficiency, extract conclusions about their abilities to detect vulnerabilities, and prevent others by making recommendations of the use of WAVS by companies or organizations .

Keywords: Web applications, Vulnerabilities, Evaluation, Vulnerability detection, Web vulnerability Scanners.

INTRODUCTION

The advent of Web 2.0 has changed the way we present and access resources accessible via HTTP, the complexity of the technologies used today to program Web applications made it difficult to prevent the introduction of vulnerability, and limited skills and lack of security culture for developers. All these factors have increased the attack surface and rendered some web applications more vulnerable and exploitable by hackers. In addition, network security and the installation of firewalls lack to provide adequate protection against Web-based attacks as these applications are public and accessible to all [1].

Web vulnerability scanners, which are tools to regularly audit web applications, can be used to check the presence of exploitable vulnerabilities in web applications to prevent attacks. The presence of vulnerabilities by WAVS must be carried out by the different actors of the project in a coherent way. It should be noted that security must be taken proactively and non-responsively throughout the whole life cycle of the project and not added and tested at the end of the development cycle [2].

The use of WAVS most upstream possible allows to optimize the costs and the lead times. Indeed, the longer the security is taken into account in the development stages, the more the cost of correction of the faults is raised [2]. It is therefore more appropriate to schedule security checks on a Web application by performing audit tests using WAVS throughout the development cycle and even after hosting the web application. In the current market, a lot of web application analysis tools are available, both on a commercial and open source basis. Although these tools are available in the market the question is whether they are effective or ineffective in solving security problems in WEB applications.

The main objective of this article is to assess the performance of several commercial and free scanners based on well-defined indicators :

True Positive, False Positive, and False Negative.

To evaluate and compare their performance in terms of vulnerability detection in a vulnerable web application, the following metrics are used: precision, recall and F-measure. These will allow us to analyze the results and deduce conclusions for each type of scanner.

The paper is structured as follows:

The second section defines the web applications, the third lists the classifications that exist for web vulnerabilities .The fourth deals with vulnerability scanners and their architecture. The fifth treats the related works. The sixth section presents our methodology, contribution and the follow up experimental results that served to test and compare the performance of web vulnerabilities scanners. The final section summarizes this paper, and outlines future work.

WEB APPLICATION

The Web Application Security Consortium (WASC) describes a *web application* as “a software application, executed by a web server, which responds to dynamic web page requests over HTTP” [3].

A web application is constituted of a collection of scripts, which lie on a web server and interact with databases or other sources of dynamic content. Using the infrastructure of the Internet, web applications allow service providers and clients to share and manipulate information in a platform-independent manner. For a good introduction to web application from the penetration tester’s perspective.

As explained, the increasing complexity of the technologies used to develop the web applications, as well as the lack of security expertise, can largely explain the recurring vulnerabilities they present.

A web application has a distributed n-tiered architecture. Typically, there is a client (web browser), a web server, an application server (or several application servers), and a persistence (database) server. Figure 1 presents a simplified view of a web application. There may be a firewall between web client and web server.

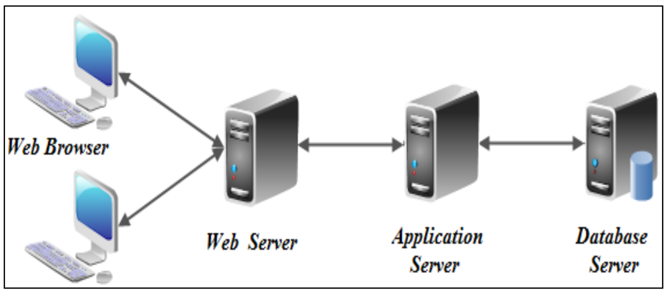

Figure 1: 3 Tier Web Application Architecture.

CLASSIFICATION OF WEB VULNERABILITIES

The increasing number of vulnerabilities on websites on the Internet have prompted many organizations to take a critical look at the quality of security of their web applications.

As a result, several communities were born with the aim of improving the security of web applications. Work in this context has also resulted in the proposal of taxonomies and classifications for the most widespread vulnerabilities and web attacks. Among these communities, we can site OWASP (Open Web Application Security Project) and WASC (Web Security Consortium application).

WASC Classification

According to (WASC Treat Classification) the Web Application Security Consortium (WASC) provides a classification of 49 threats: These threats can be divided into two categories : Weakness threats and attack threats. Besides, (WASC TC) include the source of the threats: Design, implementation or development.

The 49 threats can be grouped into six categories :

Insufficient authentication, Insufficient authorization, Client-side attacks, Command execution, Information leakage and Logical attack [4].

OWASP Classification

Unlike WASC, which describes all possible attacks, the Open Web Application Security Project (OWASP) addresses only the top 10 security risks every 3 years. It publishes the ranking of the 10 most dangerous security vulnerabilities in the document "OWASP Top 10" and allows the project team to focus on protecting the web application against the most important threats [5] .

The list of top ten risks [6] in web applications is shown in Table 1:

Table 1: Top 10 Vulnerability List

A1 Injection.
A2 Broken Authentication and Session Management.
A3 Cross-Site Scripting (XSS).
A4 Insecure Direct Object References.
A5 Security Misconfiguration.
A6 Sensitive Data Exposure.
A7 Missing Function Level Access Control.
A8 Cross-Site Request Forgery (CSRF).
A9 Using Components with Known Vulnerabilities.
A10 Unvalidated Redirects and Forwards.

WEB APPLICATION VULNERABILITY SCANNERS.

Web application vulnerability scanners are automated test tools serve to examine Web applications and detect present vulnerabilities. WAVS can also be used to search for software encoding errors such as illegal input strings and buffer overflows [7]. A good number of commercial and open source tools are available, however they have their own strengths as well as their own weaknesses.

In order to get a clearer idea of the functionality of WAVS, NIST has defined a list of requirements that all web vulnerability scanners must provide [8]:

- ✓ Identify all types of vulnerabilities described in OWASP top 10.
- ✓ Report an attack that demonstrate vulnerability.

- ✓ Indicate the attack by specifying the location of the script, inputs and context.
- ✓ Identify the vulnerability with a name semantically equivalent to those of the (OWASP) top 10.
- ✓ Be able to authenticate to the application and maintain the connected state.
- ✓ Have a sufficiently low false positive rate.

Architecture of web application vulnerabilities scanners .

Web application scanners consist of three major modules: a crawler module, an attacker module, and an analysis module.

- **The Crawling module :**

The Crawling is made by a component called Crawler. This component explores the web application to recover and identify the web pages,the associated input vectors such as fields of input of the HTML forms, request parameters GET and POST and cookies. Besides, the crawler creates an indexed list of all crawled pages. The detection of the presence of a web vulnerabilities depends essentially on the quality of the Crawler .If the crawler is mediocre then the scanners will surely miss the vulnerability [9][10].

- **The attaker (fuzzing):**

The fuzzing is made by a component called fuzzer, which analyzes pages urls and input vectors crawled by the first component, and then sends potential attack patterns to the entry points identified in the previous step. This component generates potentially vulnerable values to trigger a vulnerability for each entry and vulnerability type for which the Web application vulnerability scanner tests; for example to test the presence of XSS vulnerability the fuzzer will try to inject malicious code Javascript [9][10].

- **The analysis module:**

This module puts the results obtained during the fuzzing phase under analysis to detect the presence of the vulnerabilities and provide comments to the other modules. If the pages returned in response to the input tests for SQL injection contains a database error message, the analysis module will deduce the presence of SQL vulnerability[9][10].

RELATED WORK

Several studies evaluated WAVS .Some of them put under test scanners for a single particular vulnerability while others used a large number of vulnerability sorts in their test applications.

In [11], Bau and al. evaluated 8 commercially vulnerable scanners and tested them on well-known applications. After comparing the results,they deduced that the majority of scanners had detected SQL injection vulnerability and Reflected XSS vulnerability. Other vulnerabilities were not detected at all or were detected with a very low rate.

In [12], Ferreira and al. elaborated a vulnerable web application and tested some vulnerability scanners against it. The result was that the scanners could not detect reflected XSS and SQL injection, but could detect stored XSS and Cross-Site Request.

In [13], Fakhreldeen and al. evaluated the open source web vulnerability scanners according to OWASP Top 10-2013. Later, they compared their detection capabilities using the average metric, in order to identify the best scanners.

In [14], Suto tested 3 scanners of vulnerabilities against 3 different web application. After they measured the effectiveness of each scanner by choosing the metric code coverage to evaluate the detection capabilities of each scanner.

In [15], Alassmi and al. focused on the detection of stored XSS, based on [Beau] analysis results and confirmed the results obtained on the weaknesses and limitations of the scanners .

In [16],Yuliana M. introduces some evaluation reports from the results of running QualysGuard WAS and Acunetix WVS in opposition to a selected test bed. The identification of the most demanding vulnerabilities is present for WAVS both to spot, and compare their usefulness as penetration testing tools.

Unfortunately the authors did not compare the performance of commercial scanners to the open source scanners nor did they measure the performance of the tools on the basis metric. It was also seen that only a limited number of scanners was tested and no recommendation was given for the result. In comparison, our work, will present the largest evaluation of web application scanners in terms of the number of tools tested (eleven, both commercial and open-source), and we will use well-defined indicators to calculate the performance of each type of vulnerability scanner.

METHODOLOGY AND CONTRIBUTION.

In our evaluation, several scanners are compared to evaluate their performance for True Positives, False Positives and False Negative, in order to apply some metrics on. The tools are evaluated according to a methodology that will be defined later.

Methodology :

An appropriate methodology is required to proceed to an evaluation of WAVS chosen during our study .Our methodology process, shown in figure 1, consists of:

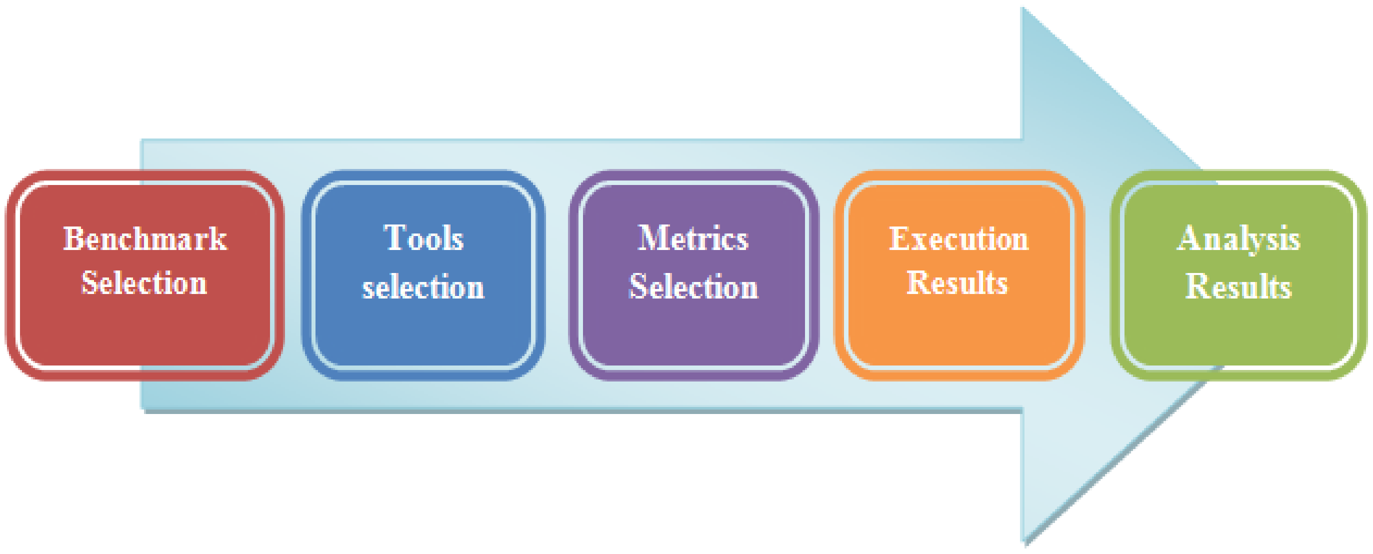

Figure 2: Methodology process.

A. Benchmark selection:

For evaluating and testing WVSs, vulnerable test applications are needed. These applications need to have exact listed vulnerabilities, so we can obtain the false negatives, true positives and false positives .

There are several vulnerable web applications like The Web Application Vulnerability Scanner Evaluation Project (WAVSEP). We are using this web application for our experiments.

WAVSEP BENCHMARK: The Web Application Vulnerability Scanner Evaluation Project (WAVSEP) is a vulnerable web application written in java JSP designed to help assessing the features, quality and accuracy of web application vulnerability scanners [17]. This evaluation platform contains a collection of unique vulnerable web pages that can be used to test the various properties of web application scanners.

WAVSEP benchmark includes the following test cases :

Table 2: The total number for each vulnerability type in WAVSEP.

Vulnerability Type.	Number of true positive cases.	Number of false positive cases.
SQL Injection.	136	10
Reflected XSS.	66	7
Remote File Inclusion (RFI)	108	6
Path traversal / Local file Inclusion (LFI).	816	8

B. Tools selection:

Tested Web Vulnerabilities Scanners .

As most of the research papers focus on commercial WAVS or open source WAVS. It has been decided to test and evaluate both open source and commercial scanners . Eleven tools have been used: Acunetix, Burp Suite, Netsparker, AppSpider, Arachni, Vega,Wapiti, Owasp ZAP, SkipFish,IronWASP and W3af .

The general characteristics of tools selected for the evaluation are the following:

The tools in pink highlight the commercial ones, while those in blue highlight the open source tools.

Table 3: Lists the WAVS used in our study and their general characteristics.

Scanners	Company/ Creator	Version	Licence	Technology
BurpSuite	PortSwiger	1.6.12	Commercial / Free (Limited Capability)	Java
Acunetix	Acunetix	10	Commercial / Free (Limited Capability)	Perl
Wapiti	InformaticaGesfor	2.3.0	Open Source	Python
SkipFish	Google	2.10	Open Source	C
Netsparker	Mavituna Security	2.3	Commercial	.Net
W3AF	W3af devel	1.2	Open Source	Python
AppSpider	Rapid 7	6.0	Commercial	Java
IronWASP	L. Kuppan	0.9.7.1	Open Source	.Net
Arachni	Tasos Laskos	2.2.1	Commercial	Ruby
ZAP	OWASP	2.3.1	Open Source	.Net
Vega	Subgraph	1.0(beta)	Open Source	Java

C. Metrics selection :

Some metrics were used to evaluate the performance degree of selected WAVS. The performance metrics are:

Precision: The ratio of correctly detected vulnerabilities to the number of all detected vulnerabilities. Also referred to as true positive

accuracy, positive predictive value or confidence. In our context it can be represented by the following formula [18]:

$$Precision = \frac{TP}{TP + FP}$$

Recall: A ratio of correctly detected vulnerabilities to the number of all known vulnerabilities. Recall is also called True Positive Rate or Sensitivity. In our context it can be represented by the following formula [18]:

$Recall = \frac{TP}{TP + FN}$

Where:

- True Positives **(TP)** is the number of true vulnerabilities detected, when a tool reports vulnerability where one is present in the code [19].
- False Positives **(FP)** is the number of vulnerabilities detected that, in fact, do not exist [19].
- False Negative **(FN)** means failure of a tool to report a vulnerability, when the vulnerability actually exists but not detected in the code [19].

F-Measure: Represents the harmonic mean of precision and recall. Equivalent to the F1 Score and PS+ [18].

The formula for the **F-measure** is:

$$F - Measured = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

It should be reminded that the greater the Precision is, the smaller the number of false positive gets. As a result, the tool becomes more accurate in detecting the vulnerability involved.

It should also be reminded that the larger the recall is, the smaller the false negative number becomes. Consequently the tool detects the vulnerability better.

The three measures can be used to establish a ranking of several tools depending on the purposes of the benchmark user. Note that these are proven metrics that are typically used to portray the effectiveness of many computer systems. Thus, these measures are easier to be understood by most of the users [19].

D. Execution Results:

The tools are executed against WAVSEP test suite. Table 4 shows the benchmark detection results . For each vulnerability scanner and vulnerability type, the number of TP, FN, FP and detection rate was calculated. The table below summarizes the results detected . The underlined values in bold type indicate the total number of vulnerabilities in the WAVSEP mentioned in Table 3.Those in yellow indicate the detection rate, while the others are the values of the TP, FN, FP found.

Table 4: WAVSEP Benchmark true positive, false negatives and false positives results

Scanners	**WAVSEP VULNERABILITES RESULTS**											
	SQLI			**XSS**			**RFI**			**LFI**		
	TP	**FN**	**FP**	**TP**	**FN**	**FP**	**TP**	**FN**	**FP**	**TP**	**FN**	**FP**
	136		**10**	**66**		**7**	**108**		**6**	**816**		**8**
BurpSuite	136	0	3	62	4	0	80	28	0	496	320	1
	100%		30%	93,93%		0%	74,07%		0%	60,78%		12,5%
Wapiti	136	0	2	44	22	3	64	44	0	414	402	1
	100%		20%	66,66%		42,85%	59,25%		0%	50,73%		12,5%
Acunetix	136	0	0	66	0	0	87	21	0	292	524	0
	100%		0%	100%		0%	80,55%		0%	35,78%		0%
SkipFish	102	34	0	65	1	0	39	69	1	312	504	2
	75%		0%	98,48%		0%	36,11%		16,66%	38,23%		25%
Netsparker	136	0	3	64	2	0	57	51	0	467	349	0
	100%		30%	96,96%		0%	52,77%		0%	57,23%		0%
W3AF	81	55	3	19	47	3	13	95	1	461	355	1
	59,55%		30%	28,78%		42,85%	12,03%		16,66%	56,49%		12,5%
AppSpider	132	4	0	66	0	0	80	28	0	660	156	1
	97,05%		0%	100%		0%	74,07%		0%	80,88%		12,5%
IronWASP	136	0	5	52	14	0	106	2	0	288	528	1
	100%		50%	78,78%		0%	98,14%		0%	35,29%		12,5%
Arachni	136	0	5	63	3	0	46	62	0	162	654	0
	100%		50%	95,45%		0%	42,59%		0%	19,85%		0%
ZAP	136	0	0	63	3	0	108	0	1	590	226	0
	100%		0%	95,45%		0%	100%		16,66%	72,30%		0%
Vega	136	0	2	66	0	0	108	0	0	519	297	5
	100%		20%	100%		0%	100%		0%	63,60%		62,5%

The assessment of execution results against the benchmark is accomplished applying the following metrics .

- Recall
- Precision
- F-measure

The metrics selected are applied to obtain the most appropriate measures to promote good interpretation of results and draw good conclusions.

The following table summarizes the findings of the precision, recall, and f-measures for each vulnerability scanners against each vulnerability type.

Table 5: Precision, Recall, and F-measure metrics applied to WAVSEP test results

Scanners	WAVSEP Results											
	SQLI			XSS			RFI			LFI		
	Precision %	Recall %	F-measure %	Precision %	Recall %	F-measure %	Precision %	Recall %	F-measure %	Precision %	Recall %	F-measure %
BurpSuite	97,84	100	98,90	100	93,93	96,87	100	74,07	85,10	99,79	60,78	75,54
Wapiti	98,55	100	99,26	93,61	66,66	77,86	100	59,25	74,41	99,75	50,73	67,25
Acunetix	100	100	100	100	100	100	100	80,55	89,22	100	35,78	52,70
SkipFish	100	100	100	100	100	100	100	80,55	89,22	100	35,78	52,70
Netsparker	97,84	100	98,90	100	96,96	98,45	100	52,77	69,08	100	57,23	72,79
W3AF	96,42	59,55	73,62	86,36	28,78	43,17	92,85	12,03	21,30	99,78	56,49	72,13
AppSpider	100	97,05	98,50	100	100	100	100	74,07	85,10	99,84	80,88	89,36
IronWASP	96,45	100	98,19	100	78,78	88,13	100	98,14	99,06	99,65	35,29	52,12
Arachni	96,45	100	98,19	100	95,45	97,67	100	42,59	59,73	100	19,85	33,12
ZAP	100	100	100	100	95,45	97,67	99,08	100	99,53	100	72,30	83,92
Vega	98,55	100	99,26	100	100	100	100	100	100	99,04	63,60	77,45

E. Results Analysis:

The analysis of these results allows getting the performances of the tools related to precision, Recall and F-Measure of the different scanners for each vulnerability cited above.

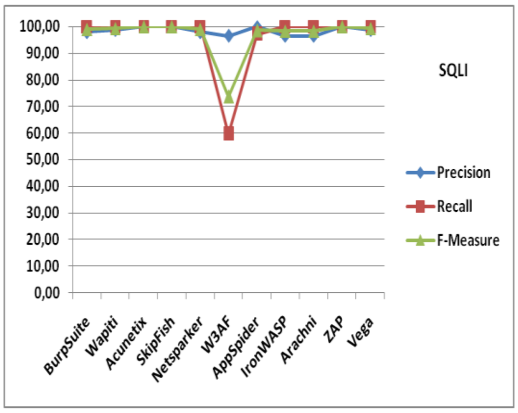

Figure 3: Precision, Recall, and F-measure metrics of SQLI vulnerability in WAVSEP

As it can be seen shown, Acunetix, SkipFish, ZAP Acunetix, SkipFish, ZAP were the tools with the highest F-Measure (100%), closely followed by BurpSuite, Netsparker, AppSpider, IronWASP and Arachni with the same (98%) . W3af had the lowest F-Measure (73, 62%).

Apart from the W3AF scanner, it can be deduced that both types of open source and commercial scanners perform well a as far as the detection of the SQL vulnerability is concerned.

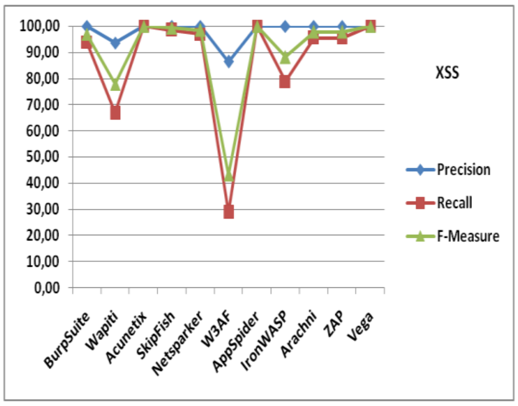

Figure 4: Precision, Recall, and F-measure metrics of XSS vulnerability in WAVSEP...

The F-Measure of Acunetix, Skipfish,Appspider and Vega presented a higher percentage than the others (100% for each). Next to them was Netsparker, Arachni, Zap, Burpsuite and Iron Wasp with a F-Measure around 90%, whereas W3AF had the lowest value (43, 17%) proceeded by Wapiti with (77, 86 %).

Apart from Arachni the commercial scanners had very good results as far as the detection of the XSS Vulnerability is concerned. Apart from Wapiti and WA3F, the open source scanners also detected this Vulnerability which had low results compared to other scanners of the same type.

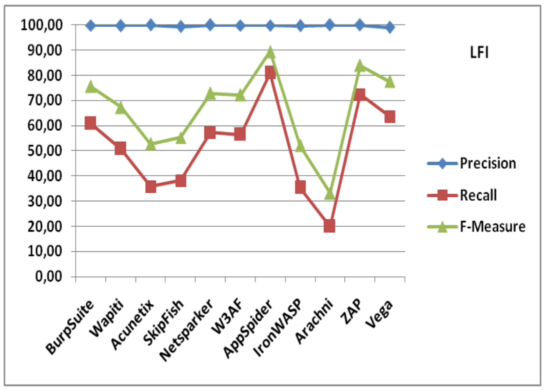

Figure 5: Precision, Recall, and F-measure metrics of LFI vulnerability in WAVSEP

As it can be seen, AppSpider and ZAP are the tools with the highest F-Measure, with a respective percentage of (89, 36 %), (83, 92 %). Next to them, are Vega, Burp suite, Netsparker, W3AF and Wapiti with f-measure very close to (70%). The lowest f-measure was for Arachni, proceeded by Acunetix, Skipfish and Iron wasp which have the same F-Measure (52%).

It has been noted that the open source Zap and Vega scanners had very good results as far as the detection of LFI is concerned. Unlike other commercial scanners like Arachni and Accunetix which have detected these vulnerabilities with low rates.

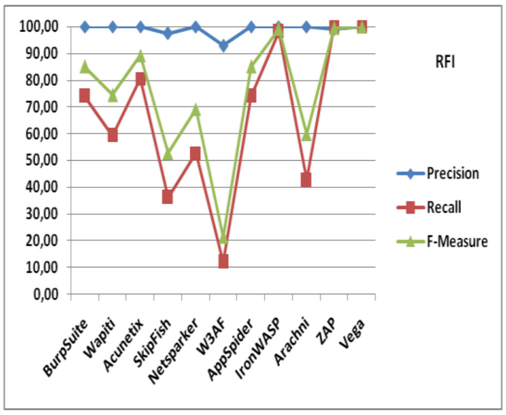

Figure 6: Precision, Recall, and F-measure metrics of XSS vulnerability in WAVSEP

According to the figure above it is evident that the percentage of F-Measure of Vega, Zap and Iron wasp have higher efficiency than the others with (100%), (99, 53 %) and (99, 06 %).

Acunetix, Skipfish, Burpsuite and Appspider had a very close F-Measure about 80%. Next to them was Wapiti and Netsparker with (74, 41%),(69, 08%).

Arachni has (59, 73%) followed by W3AF which had the lowest F-Measure among all the scanners with (21, 30%).

It has been deduced that the open source Vega, Zap and Iron wasp scanners had the highest place as far as the RFI detection followed by commercial scanners is concerned. The lowest value is occupied by the open source W3AF scanner.

CONCLUSION AND FUTURE WORK.

It can be concluded from the results above that the performance of each scanner is different for each type of vulnerability. It can also be seen from the results that the performance of all scanners in SQLI and XSS is higher than the performance in LFI and RFI.

Commercial and open source scanners have been powerful in terms of the detection of SQL and XSS vulnerabilities, but as far as the detection of the LFI and RFI vulnerabilities is concerned, it has been observed that some open source Vega and ZAP scanners have obtained better results unlike some commercial scanners.

Besides, during the scan, it was concluded that commercial tools had more features for web 2.0 analyses as AJAX, HTML5, JAVASCRIPT and WEB SERVICES.

What is more, is that the analysis performed against WAVSEP Benchmark indicates that the crawling capacity to find the complete structure and links of an application is higher for commercial tools than for open source tools, expect for Owasp zap and Vega scanners .

Furthermore, we found that there was no correlation between the cost and quality of free and commercial scanners.

As a result of this study, the appropriate scanner should be chosen for every vulnerability depending on previous assessment. We recommend that developers and web security managers should use the two open source vulnerability scanners Owasp zap and Vega, as they obtain better results in terms of detection of all types of vulnerabilities present in the WAVSEP unlike other scanners.

FUTURE WORK

In perspective, the efficiency and performance of open source web vulnerability scanners can be improved, while improving each component of the WAVS architecture:

- Improving crawling capabilities :

Some scanners need a better sophisticated crawling mechanism to guarantee that all the contents of a web application are scanned without omitting any one .

- Improving fuzzing component :

The fuzzing logic employed by some scanners need to be improved better to raise the detection accuracy .

REFRENCES

[1] R.Akrout, 2012 "Vulnerability analysis and evaluation of intrusion detection systems for Web applications.,"

[2] S. Gioria, 2009 " Web application security," Clusif, pp. 1–20.

[3] Web Application Security Consortium Glossary,http://www.webappsec.org/projects/ .

[4] "The Web Application Security Consortium / Threat Classification.", http://projects.webappsec.org/ Classification .

[5] R. Jnena, 2013 "Modern Approach for WEB Applications Vulnerability Analysis,".

[6] "Benchmark - OWASP Ten Most Critical Web Application Security Vulnerabilities,", https://www.owasp.org/index.php/Benchmark.

[7] A. Rajan and E. Erturk, 2016 "Web Vulnerability Scanners: A Case Study,".

[8] Black, P. E., Fong, E., Okun, V., & Gaucher, R. National Institute of Standards and Technology (NIST). "Software Assurance Tools: Web Application Security Scanner Functional Specification,".

[9] F. Kagorora, J. Li, D. Hanyurwimfura, and L. Camara, 2015 "Effectiveness of Web Application Security Scanners at Detecting Vulnerabilities behind AJAX / JSON,".

[10] A. Doupé, M. Cova, and G. Vigna, 2010 "Why Johnny Can’t Pentest: An Analysis of Black- Web Vulnerability Scanners," Proc. 7th Int. Conf. on Detection of Intrusions and Malware, and Vulnerability Assessment, pp. 111-131.

[11] J. Bau,, E. Bursztein, D. Gupta, and J. Mitchell, 2010 "State of the Art: Automated Black-Box Web Vulnerability Testing," in Proc. 2010 IEEE Symposium on Security and Privacy, pp. 32-345.

[12] A. M. Ferreira, and H. Kleppe, "Effectiveness of Automated Application Penetration Testing Tools," Master dissertation., Master Education SNE/OS3, University of Amsterdam, Netherlands.

[13] Fakhreldeen A. and Eltyeb E., 2014 "Assessment of Open Source Web Application Security Scanners," College of Computer Science and Information Technology, KAU, Khulais, Saudi Arabia.

[14] L. Suto, 2010 "Analyzing the Accuracy and Time Costs of Web Application Security Scanners," BeyondTrust,.http://www.beyondtrust.com/Content/ whitepapers/Analyzing-the-Accuracy-and-Time-Costs-of-Web-Application-Security-Scanners.pdf

[15] S. Alassmi, P. Zavarsky, D. Lindskog, R. Ruhl, A. Alasiri, and M. Alzaidi,,2012 "An analysis of the Effectiveness of Black-box Web Application Scanners in Detection of Stored XSSI Vulnerabilities," International Journal of Information Technology and Computer Science, vol. 4, No. 1.

[16] Yuliana M., 2012 "Security Evaluation of Web Application Vulnerability Scanners’ Strengths and Limitations Using Custom Web Application," California State University.

[17] Y. Makino and V. Klyuev, 2015 "Evaluation of web vulnerability scanners," Proc. 2015 IEEE 8th Int. Conf. Intell. Data Acquis. Adv. Comput. Syst. Technol. Appl. IDAACS 2015, vol. 1, no. September, pp. 399–402.

[18] D. M. W. POWERS, 2011 "Evaluation: From Precision, Recall and F-Measure To Roc, Informedness, Markedness & Correlation," J. Mach. Learn. Technol., vol. 2, no. 1, pp. 37–63.

[19] N. Antunes and M. Vieira, 2010 "Benchmarking vulnerability detection tools for web services," ICWS 2010 - 2010 IEEE 8th Int. Conf. Web Serv., pp. 203–210.

[20] I. Lafram, N. Berbiche, and J. El Alami, “A Random Forest Estimator Combined With N-Artificial Neural Network Classifiers to Optimize Network Intrusion Detection,” vol. 12, no.16,pp.5835–5843,2017.