HASIL CEK_60010313_(1)

by Sunardi 60010313

Submission date: 10-Dec-2020 08:29AM (UTC+0700)
Submission ID: 1470391211
File name: CEK1_60010313.pdf (669.74K)
Word count: 4589
Character count: 25261

Vulnerability Analysis of E-voting Application using Open Web Application Security Project (OWASP) Framework

Sunardi[1]
Department of Electrical Engineering
Universitas Ahmad Dahlan
Yogyakarta
Indonesia

Imam Riadi[2]
Department of Information System
Universitas Ahmad Dahlan
Yogyakarta
Indonesia

Pradana Ananda Raharja[3]
Master Program of Informatics
Universitas Ahmad Dahlan
Yogyakarta
Indonesia

***Abstract*—This paper reports on security concerns in the E-voting used for the election of village heads. Analysis of the system and server uses two different tools to determine the accuracy of scanning vulnerabilities based on the OWASP Framework. We reported that the results of the scanning using the ZAP tool got vulnerability information with the following risk level, one high level, three medium levels, and eleven low levels. The Arachni tool got vulnerability information with the following risk level, one high level, three medium levels, and two low levels. ZAP has a more complex vulnerability view than Arachni. Fatal findings on E-voting in this E-voting system is XSS, which impacts clients, which can be exploited by attackers to bypass security. Directory Traversal allows attackers to access directories and can execute commands outside of the web server's base directory. Cyber Hiscox Readiness report in 2018 in several European countries such as The United States, Britain, Germany, Spain, and the Netherlands, that the Attackers target through the most vulnerable security holes such as injection, Broken Authentication, Sensitive Data Exposure, XXE, Merged, Security Misconfiguration, XSS, Insecure Deserialization, Using Components with Known Vulnerabilities, Insufficient Logging, and Monitoring. The purpose of cyberattacks alone can threaten the stability of the country and disturb other factors. E-voting, as part of an electronic government system, needs to be audited in terms of security, which can cause the system to disrupt.**

Keywords*—Vulnerability; e-voting; open web application security project framework; attacker

I. Introduction

The development of information technology is used to replace previous habits that have not been computerizing. E-voting is a web-based technology that can be utilized by the government in carrying out election activities [1] [2] [3]. The implementation of E-voting[1] in government can have a severe impact if it is not ready to respond to cyber-attacks [4] [5] [6]. Fig. 1 shows cyberspace statistics that, on average, there are millions of data breaches, and that dangerous attacks continue to increase. On average, 73% of businesses are not ready to respond to cyber-attacks is by the 2018 Cyber Hiscox Readiness Report[2].

The United States, Britain, Germany, Spain, and the Netherlands found that most organizations were not prepared and would be severely affected by cyber-attacks [7].

OWASP TOP 10 2017[3] explains ten application security risks, as in Fig. 2. Zed Attack Proxy[4] (ZAP) from OWASP is one of the most popular free security scanning tools in the world and is actively managed by hundreds of international volunteers. ZAP can automatically scan for security vulnerabilities in web applications when they are developed and tested. ZAP is a reliable tool for experienced penetration testers to be used as automatic safety testing tools [8][9].

In addition to explaining application security risks, OWASP Top 10 is a Guide for developers and security teams to control weaknesses in web applications that are vulnerable to attack and to anticipate. These various vulnerabilities make it easy for intruders to embed malware, search for data, or completely take over the site [10].

Even though the web server is physically protected, web applications that run in the environment are not protected from attacks through computer networks. The attacks referred to according to OWASP Top 10-2017 among other things, Injection Weaknesses such as SQL injection[5], NoSQL[6], OS[7], and LDAP[8] are caused when fake data is sent to the server as part of the order [11].

Broken Authentication is an application function related to authentication and sessions misapplied so that an attacker can ignore passwords, tokens, and exploit weaknesses as other implementations to use the identity of other users [12].

[1] https://github.com/pradavanraharja/ta_dup

[2] https://www.hiscox.com/sites/default/files/content/2018-Hiscox-Cyber-Readiness-Report.pdf

[3] https://www.owasp.org/index.php/Main_Page

[4] https://www.owasp.org/index.php/OWASP_Zed_Attack _Proxy_Project

[5] https://en.wikipedia.org/wiki/SQL_injection

[6] https://en.wikipedia.org/wiki/NoSQL

[7] https://en.wikipedia.org/wiki/Operating_system

[8] https://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol

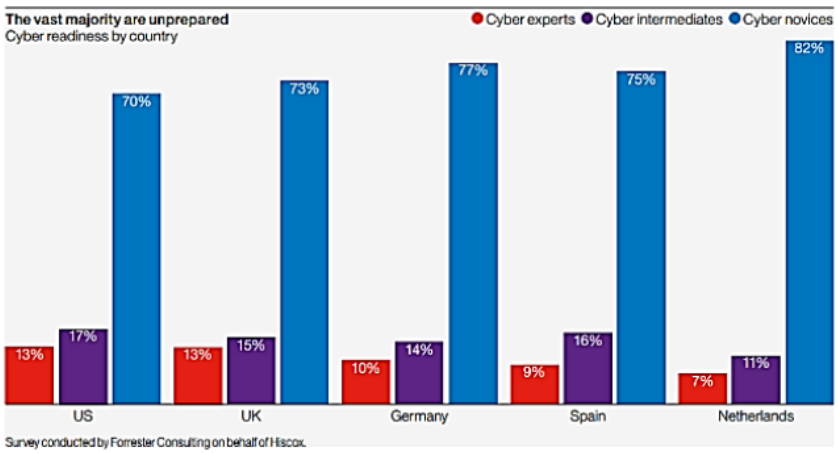

Fig. 1. The Organization is not Ready to Face Cyber-Attacks in 2018.

Sensitive Data Exposure is a matter related to web applications and the Application Programming Interface (API)[9]. It does not protect sensitive data accurately, such as financial, health, and personally identifiable information (PII). Rescue can be done, or change data is protected, but some are protected, so it is necessary to remove from credit cards, identity protection, and other crimes [13]. This sensitive data is not well protected because it is not encrypted when there is an exchange of data with the browser [14]. XML External Entities (XEE) are XML[10] documents that are outdated or not properly configured to be an attacker reference. Attackers can uncover internal files using URI Handler documents, internal file shares, internal port scanning, remote code execution and Denial of Service (DoS) attack.

OWASP Top 10 2017
A1:2017 – Injection
A2:2017 – Broken Authentication and Session Management
A3:2013 – Sensitive Data Exposure
A4:2017 – XML External Entity (XXE) [NEW]
A5:2017 – Broken Access Control [Merged]
A6:2017 – Security Misconfiguration
A7:2017 – Cross-Site Scripting (XSS)
A8:2017 – Insecure Deserialization [NEW, Community]
A9:2017 – Using Components with Known Vulnerabilities
A10:2017 – Insufficient Logging & Monitoring [NEW, Comm.]

Fig. 2. Top Ten Attacks by OWASP Top 10-2017.

Broken Access Control is a limitation of the actions that can be performed by authenticated users that are often misused. Attackers can exploit this weakness to access unauthorized functionality and data, such as accessing other users, viewing sensitive documents, modifying other users' data, and changing access rights [15]. Security Misconfiguration is the most commonly seen problem. Generally, it is the result of default configuration that is not safe, incomplete or ad hoc configuration, open cloud storage, configuration errors in the HTTP header, and error messages that contain sensitive information [16].

Cross-Site Scripting (XSS) can occur when an application is inserted fake data on a web page without validation or can update web pages with data entered through a browser, resulting in resulting HTML and Javascript. XSS is used by attackers to execute scripts in victim browsers that can hijack when accessed, delete the web, or redirect users to malicious sites [17]. Insecure Deserialization can cause remote code execution, even if insecure deserialization can be used to carry out attacks, including replay attacks, injection attacks, and privilege escalation attacks [18].

Using components with known vulnerabilities includes components, libraries, frameworks, and other software modules. If a component is vulnerable to exploitation, attacks like that have an impact on data loss or server takeover. The application and API use components with known vulnerabilities, thus damaging the application's defence and activating various attacks [19]. Insufficient logging and monitoring that is incomplete and ineffective with the incident response so that the attacker can carry out further attacks for maintaining position, expand attacks on the system, damage, extract, and destroy data. Research reports that approved for more than 200 days are usually known or detected from the private sector or monitoring [20].

ZAP is a tool that implements the OWASP Top 10 method with the main features of Intercepting Proxy, Automated Scanner, Passive Scanner, Brute Force Scanner, Spider, Fuzzer, Port Scanner, Dynamic SSL Certificate, API,

[9] https://en.wikipedia.org/wiki/Application_programming_interface
[10] https://en.wikipedia.org/wiki/XML

Beanshell Testing. As if an attacker from outside to break into the system to get data or make DoS Attack [21][22].

This research focuses on preventing attacks targeting web applications, servers, and artefacts related to vulnerability analysis in the E-voting tested concerning source code, libraries, folders, encryption, and web interfaces. The aim is to find out the possibility of an attack [23].

The graph in Fig. 3 shows that "exploitation of security misconfiguration" is 11%, and "Exploitation of software vulnerability" is 5%, including eight primary enemy targets [24]. The following can be a source of problems in security devices and is a challenge for practitioners to provide appropriate solutions and assistance to find out the source of problems, treatment, prevention, and repair, OWASP Top 10 is the best choice.

ZAP, as a testing tool designed explicitly for installing web applications, ZAP is known as a main-in-middle proxy. ZAP works between the browser tester and the web application so that it can intercept and read messages sent between the browser and the web application, manage the content needed, and can install packages to the destination we can see in Fig. 4 [25].

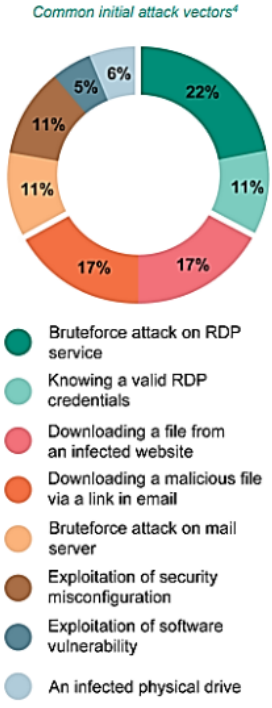

Fig. 3. Adversary Attack Vectors in 2018.

Fig. 4. The Concept of main-in-the-Middle Proxy on ZAP.

The functionality provided by ZAP for various levels ranging from Developers and Security Testing Specialists [26]. The results received by the tester are in the form of reports from ZAP in the form of HTML and XML files. In ZAP Scanning Report consists of Summary of Alerts, which are grouping in certain levels along with the amount.

II. Literature Review

This paper consists of two parts: Risk rating study based on OWASP on E-voting and comparison of reporting results from ZAP and Arachni.

A. Risk Rating

Currently, finding a vulnerability to be necessary, but being able to estimate risks associated with the business is equally important. One can identify security problems in architecture or design using threat modelling. Someone has the possibility of finding security problems using source code or penetration testing, and problems not yet found until the application is in production and entirely compromised.

This method might be used to estimate the severity of all risks to the web application and make decisions based on information about what to do with those risks. Reviewing the ranking of risks to the system will save time and eliminate polemics about priorities. The results of research on web applications are not affected by small risks, and more significant risks not known with certainty.

As shown in Table I, explained that in this step, the likelihood estimation and impact estimation are combined to calculate the overall severity of this risk. It is finding out if the likelihood is low, medium, or high and then doing the same for impact. Scale 0 to 9 into three parts. Ideally, there will be a universal risk assessment system that will accurately estimate all risks for all organizations. However, vulnerabilities that are very important for one organization may not be too significant for another. So the basic framework presented here is for specific organizations [27].

B. OWASP Top 10-2017

OWASP Top 10-2017 is an update of OWASP as an open-source community dedicated as an organization that develops, funding, maintains applications and APIs. The benefits of OWASP (1) Tools and application security standards, (2) Complete books on application security testing, (3) code of development, and safe code review (4) Presentations and videos, (5) Cheat sheets on many general topics , (6) Control and standard security libraries, (7) local chapters around the world, (8) Leading research, (9) Extensive conferences around the world, (10) Lists of correspondence [28].

C. Architecture of Networks

The network architecture applied in this research consists of a web server acting as an E-voting server and a client acting as a Vulnerability Scanner, and there is a ZAP Application. Both the server and client use the Windows 10 operating system. The network architecture in Fig. 5 is a network architecture that simulates vulnerability scanning activities on the e-voting server to obtain important information related to vulnerabilities on the server and web application.

TABLE. I. RISK LEVEL

Likelihood and Impact Levels	
0 to <3	Low
3 to <6	Medium
6 to 9	High

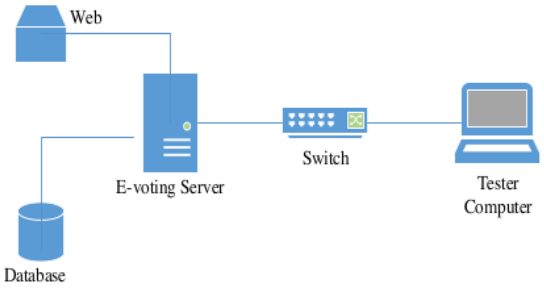

Fig. 5. NIST Methodology.

D. Methodology

The method used in this research is the forensic method based on the National Institute of Standards and Technology (NIST) with the forensic stages of acquisition, inspection, utilization, and review, as described in Fig. 6 [29][30][31].

NIST is the organization responsible for developing minimum standards, guidelines, and requirements to provide adequate information security for all assets and parties with digital forensic competencies [32].

1) Acquisition: The first step in the research process is to identify the source of the data, the phase of data acquisition related to a particular event that will be identified, collected, and protected. Table II shows the equipment and material requirements needed.

2) Examination: The data obtained in the next phase is to examine the data, identify, collect, and organize relevant information from the data obtained. This phase can also involve bypassing or mitigating operating system or application features that obscure data and code, such as data compression, encryption, and access control mechanisms. It is a testing phase of appropriate tools and techniques for the types of data collected during the first phase to identify and analyse relevant information from the data obtained.

3) Utilization: The utilization of data is the process of preparing and presenting information generated from the inspection stage. Many factors influence data utilization, including data reduction, alternative explanations, audience considerations, and actionable information. The final phase, which involves the reporting process and practice in the context of current events to identify policy deficiencies, then procedural errors, and other problems that need to be corrected.

To identify each computer in the network, in this research we provide 192.168.130.0 networks on two computers as shown in Table III.

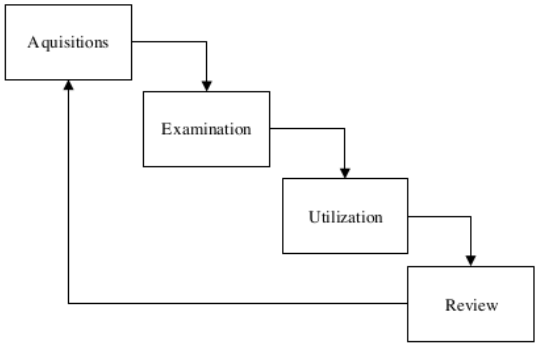

Fig. 6. NIST Methodology.

TABLE. II. EQUIPMENT FOR SUPPORT VULNERABILITY SCANNING PROCESS

No	Equipment	Description
1	E-voting Computer Server	Intel Pentium CPU, 4GB RAM, SSD 128GB
2	Computer Tester	Intel i5 CPU, 12 GB
3	Apache Web Server	Version 2.2
4	MySQL database	Version 10.1.16
5	Switch	Tp-link TL-SG1005D
6	ZAP	Version 2.8.0
7	Arachni Framework	Version 1.5.1
8	Arachni WebUI	Version 0.5.12

TABLE. III. SERVER, TESTER'S COMPUTER AND IP ADDRESS

No	Host	IP Address
1	E-voting Server (Windows 10)	192.168.130.248 (*static*)
2	Computer Tester (Windows 10)	192.168.130.90 (*dynamic*)

Fig. 7 explains the reporting process on vulnerability scanner tools [33]:

a) The tester runs penetration testing tools.

b) Design API report on Penetration testing tools.

c) APIs Report inserted in Penetration testing tools.

d) Penetration testing tools produce XML documents.

e) XML documents and Report APIs are combined to produce formatted reports.

4) Review: The analysis is repeated to improve the process and practice in the context of the current task to help with policy problems, procedural problems, and other problems that need fixing. Regular updating of skills through courses, workplace experience, and academic resources helps ensure that the person performing the data analysis follows technology developments and rapidly changing job responsibilities. Regular reviews of policies and procedures also help to ensure that organizations stay abreast of the latest technology and change laws or rules.

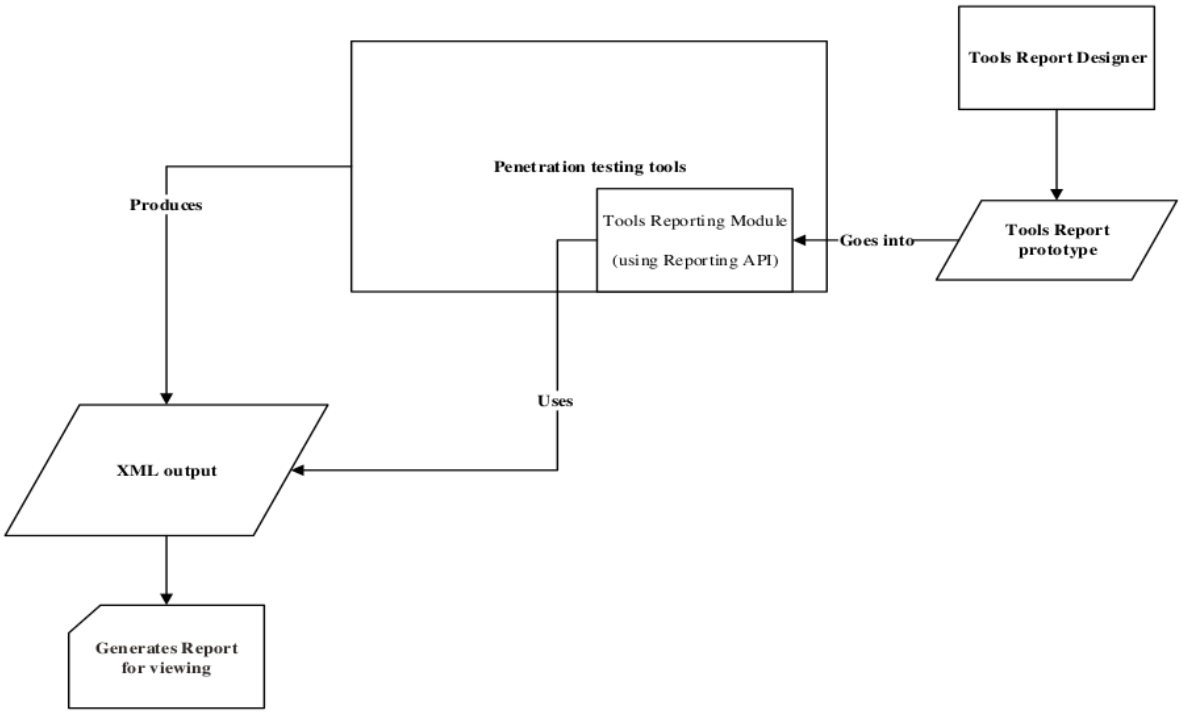

Fig. 7. Reporting Process on Vulnerability Scanning Tools.

III. Result and Analysis

Based on the results of the research and analysis carried out, there are criteria scanning analysis as a parameter to explain the expected results related to the most critical web application security risks in Table IV.

Identifying and analysing the vulnerability scanning process from this research can following steps will be taken to obtain results in the form of digital evidence as reports from ZAP and Arachni vulnerability scanning tools.

A. Acquisition

The acquisition of this research is to run a vulnerability scanner on the client to obtain vulnerabilities on the E-voting web server. In Fig. 8, we can see that the E-voting scan on the web server is ready to work.

TABLE. IV. Severity Level and Vulnerability Parameters

No	Severity Level	Vulnerability Parameters	Result
1	High	Cross-Site Scripting XSS	Yes
2	Medium	Source code disclosure / Application Error Disclosure	Yes
3	Medium	Common Directory / Directory Browsing	Yes
4	Low	Missing 'X-Frame-Options' header / X-Frame Options Header Not Set	Yes

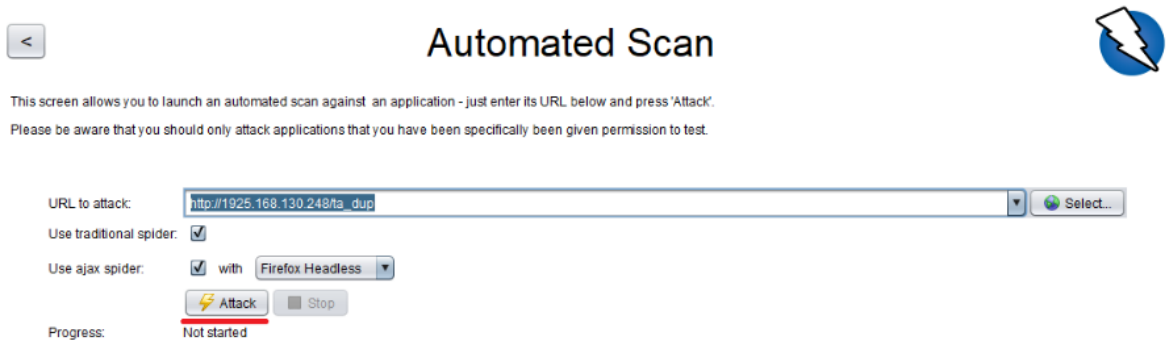

Fig. 8. Automated Scan Facilities.

Requirements must be worked out in preparation for this research.

1) All major components (Apache[11], MySQL[12], and web application[13]) in XAMPP[14] are ready to be scanned on the Server Computer.

2) All components of XAMPP is working on the server.

3) Vulnerability scanner installed on a computer.

a) ZAP and Arachni must load the Web Application URL as a target before scanning.

b) Then carry out scanning execution automatically by pressing the Attack button, so this application scans the target.

4) Finally, get a report from the vulnerability scanner.

The acquisition uses a vulnerability scanner as proof that it is scanning the target for the IP address by showing hexadecimal and the text in Fig. 9.

As shown in Fig. 10, when the scan is in progress, it can be seen in the Active Scan section with valuable information in the form of Request Timestamp, Response Timestamp, methods, and URL.

Depending on the device and server computer, this scanning process takes 15 minutes using ZAP, then 1 hour and 45 minutes using Arachni, because other factors influence this process. After finished scanning the server, the next step is to collect evidence and reports from Arachni as a vulnerability scanning tool, then the entire acquisition process using ZAP and Arachni is finished, we then proceed to the inspection process.

B. Examination

Previously we have made an acquisition, then an examination of the results of the acquisition will be carried out on both the vulnerability scanner and the application.

1) Examination of the results of the acquisition by scanning using ZAP has obtained reports. In this research, ZAP has conducted scanning for vulnerability to E-voting without constraints. The inspection process using ZAP can be measured because we can do the same thing in using other vulnerability scanning tools. ZAP scanning measures the level of vulnerability that exists in E-voting and application.

Fig. 11 explains ZAP has obtained and collected the results as evidence of response messages that come from scanning the Application, so the scanning process using ZAP will explain to us about valuable information relating to vulnerabilities.

2) Arachni reports as a whole that recorded on the results of the webserver scan with a report explaining starting from the base URL to the web application directory in Fig. 12.

The scan produced by Arachni is known to be the response message that comes from scanning the entire web server and web application. So it requires a re-examination of the scan results. However, the results reported by Arachni are relatively the same as ZAP.

C. Utilization

ZAP and Arachni scanners utilize response messages from the application in the form of reports that are easy to use. So in the process of utilization, we also get the convenience of the tools provided by exploring the Desktop Application interface on ZAP and WebUI on Arachni by merely clicking on the menu button available. This step will also provide appropriate information from both vulnerability scanning tools.

1) Utilization of ZAP: ZAP used to scan has many categories of results of the scan that have a level of risk in the results of the scan, one of its features is the ZAP Scanning Report, where ZAP can obtain classified evidence so that practitioners can utilize the information provided by ZAP Scanning Report. In this *research*, we can obtain vulnerability information that has classified and use it through this facility.

As shown in Fig. 13 shows the reporting results of the investigation results from ZAP. However, as we can see, the source comes from web applications and server computers with file extensions, and IP addresses http://192.168.130.248/ta_dup/in more detail.

2) Utilization of Arachni: Using Arachni can report the results of the scan, the scan results extracted, and risk levels have arranged in Fig. 14. Similar to ZAP, Arachni also explained the scan results in the form of IP address and direction.

D. Review

We have conducted investigations including acquisition, examination, utilization, and then the final step is to conduct a review. ZAP and Arachni have successfully obtained digital evidence using scanning through a computer network. Evidence in the form of reports obtained from a vulnerability scanner and then carried out an analysis, the details of the report contain descriptions, URLs, methods, parameters, information, and evidence.

[11] https://en.wikipedia.org/wiki/Apache_HTTP_Server
[12] https://en.wikipedia.org/wiki/MySQL
[13] https://en.wikipedia.org/wiki/Web_application
[14] https://en.wikipedia.org/wiki/XAMPP

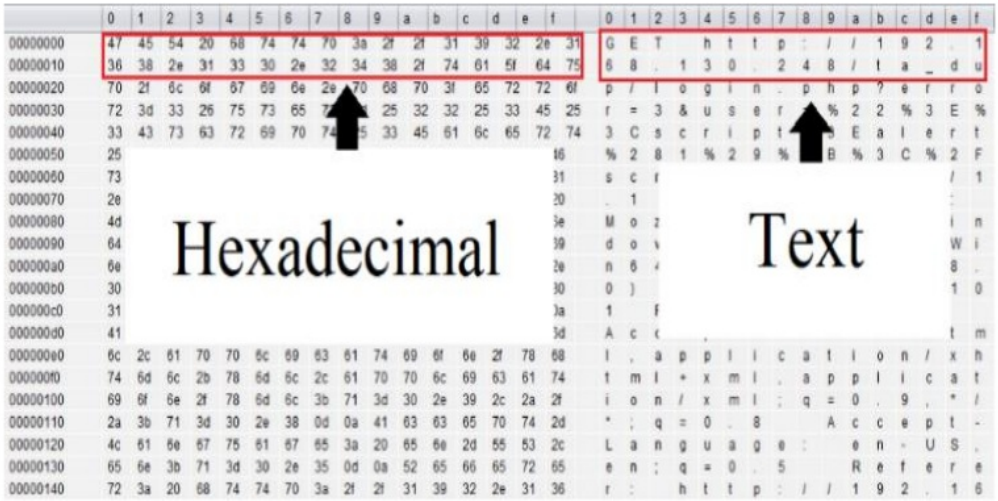

Fig. 9. Request Scan in Hexadecimal and Text Form.

Id	Req. Timestamp	Resp. Timestamp	Method	URL
363	8/29/19, 7:04:14 PM	8/29/19, 7:04:14 PM	GET	http://192.168.130.248/ta_dup/asset
364	8/29/19, 7:04:14 PM	8/29/19, 7:04:14 PM	GET	http://192.168.130.248/ta_dup/asset/bootstrap
365	8/29/19, 7:04:14 PM	8/29/19, 7:04:14 PM	GET	http://192.168.130.248/ta_dup/asset/bootstrap/css
366	8/29/19, 7:04:14 PM	8/29/19, 7:04:14 PM	GET	http://192.168.130.248/ta_dup/asset/css
367	8/29/19, 7:04:14 PM	8/29/19, 7:04:14 PM	GET	http://192.168.130.248/ta_dup/asset/Images
368	8/29/19, 7:04:15 PM	8/29/19, 7:04:15 PM	GET	http://192.168.130.248/ta_dup/login.php?error=c...
369	8/29/19, 7:04:15 PM	8/29/19, 7:04:15 PM	GET	http://192.168.130.248/ta_dup/login.php?error=.....

Fig. 10. Progress of Scanning.

URL	http://192.168.130.248/ta_dup/login.php?error=4&user=%22%3E%3Cscript%3Ealert%281%29%3B%3C%2Fscript%3E
Method	GET
Parameter	user
Attack	"><script>alert(1);</script>
Evidence	"><script>alert(1);</script>

Fig. 11. ZAP Scan Report.

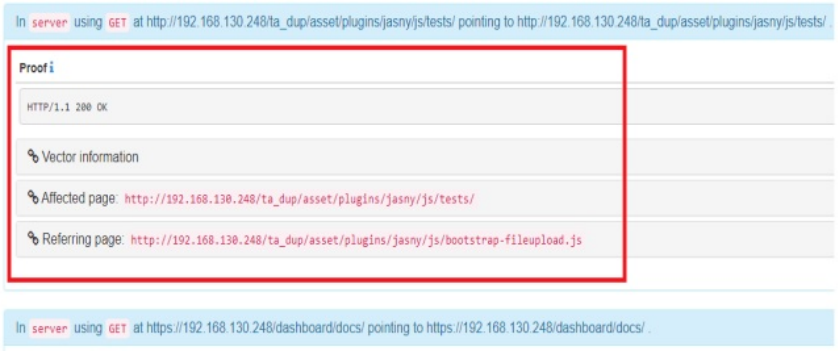

Fig. 12. Arachni Scan Report.

ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	1
Medium	4
Low	11
Informational	0

Fig. 13. A Report of the Results by ZAP.

Fig. 14. A Report of the Results by Arachni.

IV. Conclusion

Based on the results of the research that has reviewed, ZAP and Arachni completed the scanning and analysis of reports that were measured and run through the client's computer. ZAP and Arachni managed to get evidence in the form of descriptions, URLs, methods, parameters, information, and evidence: (1) 1 high level in ZAP & 1 high level in Arachni, (2) 4 medium levels in ZAP & 3 medium levels on Arachni, (3) 11 low levels in ZAP & 2 low levels in Arachni. We hope that ZAP and Arachni can be developed to identify digital evidence of vulnerabilities in mobile applications.

References

[1] R. Krimmer, "The Art of Structuring," in The Art of Structuring, Springer International Publishing, 2019, pp. 421–426.

[2] B. Rexha and I. Murturi, "Applying Efficient Crowdsourcing Techniques for Increasing Quality and Transparency of Election Processes," Electron. Gov., vol. 15, no. 1, pp. 107–128, 2019.

[3] S. Djanali, D. P. Nugraha, H. Studiawan, and B. Adi Pratomo, "Vote identification and integrity of ballot in paper-based e-voting system," Electron. Gov., vol. 14, no. 3, pp. 240–254, 2018.

[4] T. D. Wagner, E. Palomar, K. Mahbub, and A. E. Abdallah, "A Novel Trust Taxonomy for Shared Cyber Threat Intelligence," Secur. Commun. Networks, vol. 2018, 2018.

[5] A. Fadlil, I. Riadi, and S. Aji, "Review of detection DDOS attack detection using naive bayes classifier for network forensics," Bull. Electr. Eng. Informatics, vol. 6, no. 2, pp. 140–148, 2017.

[6] A. R. Caesarano and I. Riadi, "Network Forensics for Detecting SQL Injection Attacks using NIST Method," Int. J. Cyber-Security Digit. Forensics, vol. 7, no. 4, pp. 436–443, 2018.

[7] R. Mardisalu, "14 Most Alarming Cyber Security Statistics in 2019," 2019. [Online]. Available: https://thebestvpn.com/cyber-security-statistics-2019/.

[8] The OWASP Foundation, "ZAP Proxy."

[9] I. Riadi, R. Umar, and W. Sukarno, "Vulnerability of Injection Attacks Against The Application Security of Framework Based Bebsites Open Web Access Security Project (OWASP)," J. Inform., vol. 12, no. 2, pp. 53–57, 2018.

[10] D. Pandya and N. Patel, "Owasp Top 10 Vulnerability Analyses in Government," Int. J. Enterp. Comput. Bus. Syst., vol. 6, no. 1, 2016.

[11] M. Volkova, P. Chmelar, and L. Sobotka, "Machine Learning Blunts the Needle of Advanced SQL Injections," Mendel, vol. 25, no. 1, pp. 23–30, 2019.

[12] A. Yassine, E. H. Ettifouri, J. Berrich, and B. Toumi, "Modeling The OWASP Most Critical WEB Attacks," in International Conference Europe Middle East & North Africa Information Systems and Technologies to Support Learning, 2019, pp. 442–450.

[13] J. M. Dharmalingam and M. Eswaran, "An agent based intelligent dynamic vulnerability analysis framework for critical SQLIA attacks: Intelligent SQLIA vulnerability analyzer agent," Int. J. Intell. Inf. Technol., vol. 14, no. 3, pp. 56–82, 2018.

[14] J. Blue and E. Furey, "A Novel Approach for Protecting Legacy Authentication Databases in Consideration of GDPR," 2018 Int. Symp. Networks, Comput. Commun. ISNCC 2018, pp. 1–6, 2018.

[15] H. Sohoel, M. G. Jaatun, and C. Boyd, "OWASP Top 10 - Do Startups Care?," 2018 Int. Conf. Cyber Secur. Prot. Digit. Serv. Cyber Secur. 2018, no. 0102, 2018.

[16] M. Srokosz, D. Rusinek, and B. Ksiezopolski, "A new WAF-based architecture for protecting web applications against CSRF attacks in malicious environment," Proc. 2018 Fed. Conf. Comput. Sci. Inf. Syst. FedCSIS 2018, vol. 15, pp. 391–395, 2018.

[17] R. Wang, G. Xu, X. Zeng, X. Li, and Z. Feng, "TT-XSS: A novel taint tracking based dynamic detection framework for DOM Cross-Site Scripting," J. Parallel Distrib. Comput., vol. 118, pp. 100–106, 2018.

[18] G. S. Leite and A. B. Albuquerque, "An Approach for Reduce Vulnerabilities in Web Information Systems," in Proceedings of the Computational Methods in Systems and Software, 2019, pp. 86–99.

[19] C. Gil, L. Baquero, and M. Hernández, "A Conceptual Exploration for the Safe Development of Mobile Devices Software Based on OWASP," Int. J. Appl. Eng. Res., vol. 13, no. 18, pp. 13603–13609, 2018.

[20] The OWASP Foundation, "Top 10 2017 A10-Insufficient Logging & Monitoring," 2017. [Online]. Available: https://www.owasp.org/index.php/Top_10-2017_A10-Insufficient_Logging%26Monitoring.

[21] The OWASP Foundation, "OWASP: The Ten Most Critical Web Application Security R isks," 2017. [Online]. Available: https://www.owasp.org/index.php/Top_10-2017_Top_10.

[22] A. Kurniawan, I. Riadi, and A. Luthfi, "Forensic Analysis and Prevent of Cross Site Scripting in Single Victim Attack using Open Web Application Security Project (OWASP) Framework," J. Theor. Appl. Inf. Technol., vol. 95, no. 6, pp. 1363–1372, 2017.

[23] M. Alzahrani and L. Georgieva, "Structural modeling and verification of web applications," in Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics, 2018.

[24] Kaspersky, "Adversary Attack Vectors 4," 2018.

[25] D. Saputra and I. Riadi, "Network Forensics Analysis of Man in the Middle Attack Using Live Forensics Network Forensics Analysis of Man in the Middle Attack Using Live Forensics Method," Int. J. Cyber-Security Digit. Forensics, vol. 8, no. 1, pp. 66–73, 2019.

[26] K. Pandey, "A Bug Tracking Tool for Efficient Penetration Testing," Int. J. Educ. Manag. Eng., vol. 8, no. 3, pp. 14–20, 2018.

[27] The OWASP Foundation, "OWASP Risk Rating Methodology," 2019. [Online]. Available: https://www.owasp.org/index.php/Threat_Risk_Modeling.

[28] The OWASP Foundation, OWASP Top 10 – 2017 The Ten Most Critical Web Application Security Risks. 2017.

[29] I. Riadi, R. Umar, and A. Firdonsyah, "Forensic tools performance analysis on android-based blackberry messenger using NIST measurements," Int. J. Electr. Comput. Eng., vol. 8, no. 5, pp. 3991–4003, 2018.

[30] I. Riadi, R. Umar, and A. Firdonsyah, "Identification Of Digital Evidence On Android's Blackberry Messenger Using NIST Mobile Forensic Method," Int. J. Comput. Sci. Inf. Secur., vol. 15, no. 5, pp. 155–160, 2017.

[31] A. Yudhana, I. Riadi, and I. Anshori, "Identification of Digital Evidence Facebook Messenger on Mobile Phone With National Institute of Standards Technology (Nist) Method," Kursor, vol. 9, no. 3, pp. 111–118, 2019.

[32] Sunardi, I. Riadi, and A. Sugandi, "Forensic analysis of Docker Swarm cluster using GRR Rapid Response framework," Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 2, pp. 459–466, 2019.

[33] The OWASP Foundation, "GSoC2013 Ideas/OWASP ZAP Exploring Advanced reporting using BIRT," 2013. [Online]. Available: https://www.owasp.org/index.php/GSoC2013_Ideas/OWASP_ZAP_Exploring_Advanced_reporting_using_BIRT.

HASIL CEK_60010313_(1)

ORIGINALITY REPORT

4%	4%	0%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	archive.org Internet Source	4%

Exclude quotes	On	Exclude matches	< 2%
Exclude bibliography	Off		