iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** III **Month of publication:** March 2021

DOI: https://doi.org/10.22214/ijraset.2021.33410

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue III Mar 2021- Available at www.ijraset.com

Web Application Security Threats and Analysis of Vulnerability Scanners

Sukriti Jaitly[1], Mohak Verma[2]
[1, 2]*Vellore Institute of Technology, Vellore*

Abstract: ***In recent years, web security has been regarded in terms of protecting the web application layer from unauthorized users' attacks. As our reliance on technology grows exponentially, security becomes increasingly important. The main objective of this paper is to provide knowledge about web application layer vulnerabilities, their prevention methods and to perform a comparison of the latest tools and mechanisms used to detect these threats and vulnerabilities.***
Keywords**:** ***Web Application Vulnerability, Cybersecurity, Hackers, Vulnerability Scanners, Cyber-Threats, SQL-Injection, Cross-Site Scripting, Broken Authentication.***

I. INTRODUCTION

The internet has had a major effect on human lives. It has not only revolutionized communication but has vastly influenced how we live our daily lives. The accessibility and use of web applications have made them indispensable. The more we rely on web applications the more crucial it becomes to keep a check on the threats and vulnerabilities these applications are susceptible to since malicious intentions have always surrounded the use of web applications. The risk of hackers getting the better of web security has made researcher's more vigilant on making web applications safer and free from malware which can cause detrimental effects. Web applications cannot survive in today's complex computer ecosystem without a well-structured and well-maintained security system as new threats and vulnerabilities are revealed every day. Hackers are getting smarter and new vulnerabilities are identified which can be attacked and benefit people with felonious intent Web applications continue to be infected with security flaws, allowing attackers access to sensitive data and allowing malware to infect legitimate websites. Hence there is an increasing need for web applications security amongst a surge in the number of cases of threats and vulnerabilities which can lead to data loss, data theft. In this paper, we will be exploring the majority of web application vulnerabilities which include SQL injection, Cross-site scraping, and others we will be reviewing how to deal with them through vulnerability scanners. We will be studying prevention techniques and basic measures which can be employed in order to keep applications safe and secure from hackers. However, precautionary measures and prevention techniques can only keep a web application safe to a certain extent, as technology gets evolved these web applications are exposed to new security challenges. In such cases, web vulnerability scanners are very essential to test web applications for security vulnerabilities. Major web application security tools are hence explored in this study. Further, we will proceed to work with these scanners on a specific web page and analyze their performance and ease of use.

II. WEB SECURITY THREATS AND VULNERABILITIES

A. SQL Injection

SQL injection is one of the most significant security flaws in Web application systems; the majority of these flaws are caused by a lack of input validation and the use of SQL parameters. [2] SQL injection is a type of web application security vulnerability in which an attacker injects malicious input into an SQL statement. Attackers use SQL injection to extract sensitive information from databases. Arguments are used in SQL statements to pass data to and from users into a secure database. Attackers use the points where the app connects to a database with a SQL argument to gain access to sensitive information and other secured areas unless the values of these user-supplied SQL arguments are secured by sanitizing statements.[1]

An attacker may make use of a SQL injection vulnerability to perform operations such as delete, add, edit, access sensitive content, or read source code from database server files. Files can be written to the database server. It all depends on the attacker's skills, exploiting a SQL injection vulnerability can even result in a complete takeover of the database and the webserver. SQL injections are to blame for a number of high-profile data breaches every year.

B. Cross Site Scripting

Cross Site Scripting Attacks also known as XSS are a type of client-side injection code attack in which malicious scripts are injected into websites that appear to be trustworthy to the user. The web page is used as a vehicle to deliver the harmful embedded script to the user's browser. Web pages such as message boards, forums, and web pages with commenting capabilities are common targets for Cross-site Scripting attacks. [4] Any web page that uses unsanitized user input in the output it generates is at risk of an XSS attack.

C. Code Injection

Code injection is a web security threat that is caused when a malicious code is injected into an unattended computer system which gives access to the server-side interpreter to impede the processing of application software.

A harmful code is purposely injected with an intention to attack as the code is interpreted and executed by application leads to malevolent outcomes. The injected code is in the same language as the software application which is attacked, this malicious code hinders the proper functioning of the software and gives power to the server-side interpreter to gain access controls to change data or hinder program execution.[1]

D. Broken Authentication

Broken authentication takes place when an online portal has a very poorly implemented and constructed authentication and session management system. Authentication systems act as a safeguard to any online portal as they ensure only valid and verified users can use their own personal portal. Broken authentication is a widespread web threat that embodies several vulnerabilities which let attackers capitalize an opportunity to mimic actual users and breach through several actual online portals to cause damage.

Broken authentication attacks the weak and fragile nature of the session management system. The credential management system is also exploited vastly and IDs are hijacked by the attacker with malicious intent.[7]

III. COUNTERMEASURES TO WEB SECURITY THREATS

SQL-related web application vulnerabilities can be counteracted by using prepared statements with parameterized queries. A prepared statement sanitizes the input and guarantees that it is treated as a string literal rather than being a part of the SQL query. To put it another way, the database can distinguish between SQL data and SQL code and is no longer vulnerable to SQL injection attacks.

Another choice to prevent SQL injection Attacks would be to switch to Object Relational Mapping Tools (ORMs). [2] The Malicious effects of code injection can be prevented by validating suspicious input data, system should scan for special keys which makes authentication vulnerable towards attack.

The system must treat all data as suspicious and should thoroughly check for any alterations in credentials or possible manipulation of data. Regular Check on code structure and the system Architecture should be employed. Static analysis can help to identify major vulnerabilities which are related to unsafe sessions.[4]

For broken authentication web application security threats, the system should regulate session lengths and its duration. The web application must restrict a web session once user portal was left inactive for a long duration of time. Multi-factor authentication is one way in which hackers can be prevented from entering and hampering web portals. It should be made mandatory to set highly complex passwords with special characters and hot keys which makes it harder to break through the privacy. Improvement in session management can also keep a web portal safer as a new session is launched every-time after a successful authentication is done. Employment of brute force protection should be made compulsory as it only allows a specific amount of log in trials and once failed, it locks the IDs for a certain duration.

XSS attacks can be avoided by sanitizing user input. Validating and encoding output to prevent potentially vindictive user-provided data from causing automatic load-and-execute actions by a browser. Using a web application vulnerability scanning tool on a regular basis to detect XSS vulnerabilities in your applications adds another layer of security and prevention against XSS attacks.

IV. SECURITY VULNERABILITY SCANNERS

A. Arachni

Arachni is an open-source tool used for penetration testing environments and to detect various web application security vulnerabilities.

It is capable of performing web-based security audits and data scraping. This tool has a high level of network performance since it is based on an asynchronous HTTP request/response model, which means that in asynchronous I/O, operations can be planned in such a way that they appear to be happening at the same time, resulting in better bandwidth utilization and productivity.

B. ZAP

Zed Attack Proxy, also known as ZAP is a tool developed by the Open Web Application Security Project (OWASP). It is a Java-based tool with an easy-to-use graphical interface that allows web application security testers to perform scripting, spidering, fuzzing, and proxy attacks on web applications. This tool can be used as a scanner by entering the URL, or you can use it as an intercepting proxy to perform manual testing on particular pages.[3]

C. VEGA

It is a widely-used open-source web security scanner and tester. It provides a testing platform where one can check for web applications' security threats. Vega incorporates an automated scanner that quickly testifies vulnerabilities. JavaScript is used in VEGA threat detection modules which makes it quite handy to work in Windows Linux and OS X.

D. Burp Suite

Burp Suite is an integration of a set of penetration testing and vulnerability finder tools that are used to detect and testify the presence of various types of web application threats. The set of tools work seamlessly together to detect analyze and map the attack surface and well as search through the web application to explore new vulnerabilities.

V. ANALYSIS

To compare and analyze the working of the above-mentioned web application security tools. We ran the same web application website through all of them for the same specific amount of time which was 12 minutes.

Over the course of 12 minutes, Arachni was able to discover 91 vulnerabilities and threats. The 91 vulnerabilities were divided into 19 categories. The software generated a well-documented report that was simple to read and understand. The other software's were able to discover more vulnerabilities, but Arachni's information was rather straightforward. The scan was also simple to complete. As a result, this can be recommended to those who are beginners and aren't well versed. In the same 12 minutes, ZAP discovered nearly 300 vulnerabilities, threats, and data. The report, on the other hand, was cluttered and difficult to read and understand. Despite the disparity in the number of vulnerabilities discovered, ZAP only divided them into four categories. Furthermore, the number of high and medium alerts was about the same. As a result, we can deduce that there might be a discrepancy in how they classify threats/vulnerabilities. Many of the alerts were simply just extra information that had nothing to do with the threats. As a result, it is suited for those who are slightly more advanced. In the same timeframe, Burp suite discovered around 100 vulnerabilities. Its additional information was not especially valuable in comparison to all the others. It was, nonetheless, a great application for scanning for vulnerabilities alone, as well as providing more functionality for testing for these flaws.

Vega was able to discover around 300 vulnerabilities/threats/information in the given timeframe. It was also delivered in a well-documented and easy-to-read manner. It was broken down into 12 different categories. We can conclude that VEGA was the foremost software we tested for scanning web applications for vulnerabilities and threats. It is suitable for the majority of people.

VI. CONCLUSIONS

Through the course of this study, we looked at various threats that a web application can be subjected to by an attacker outlining the causes and the ill affects these threats have on a web application. While studying various threats and ways to counteract these vulnerabilities we outlined the precautionary measures and prevention techniques which can be employed in order to protect one web application from malicious intent. In this paper, we compared 4 vulnerability scanners and found VEGA to come up at the top for scanning for web app vulnerabilities. It not only scans and detects a wide range of critical vulnerabilities, but it also provides a wealth of information about each one, including the type, level of threat, possible design flaws that lead to the vulnerabilities, and suggested remediation. As technology advances at a rapid rate, the need for protection against various cyber-threats will only grow in the coming years. Cybersecurity will become a necessity in our daily lives and will play a larger role in industries.

REFERENCES

[1] S. Kumar, R. Mahajan, N. Kumar and S. K. Khatri, "A study on web application security and detecting security vulnerabilities," 2017 6th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2017, pp. 451-455, doi: 10.1109/ICRITO.2017.8342469.

[2] Li Qian, Zhenyuan Zhu, Jun Hu and Shuying Liu, "Research of SQL injection attack and prevention technology," 2015 International Conference on Estimation, Detection and Information Fusion (ICEDIF), Harbin, China, 2015, pp. 303-306, doi: 10.1109/ICEDIF.2015.7280212.

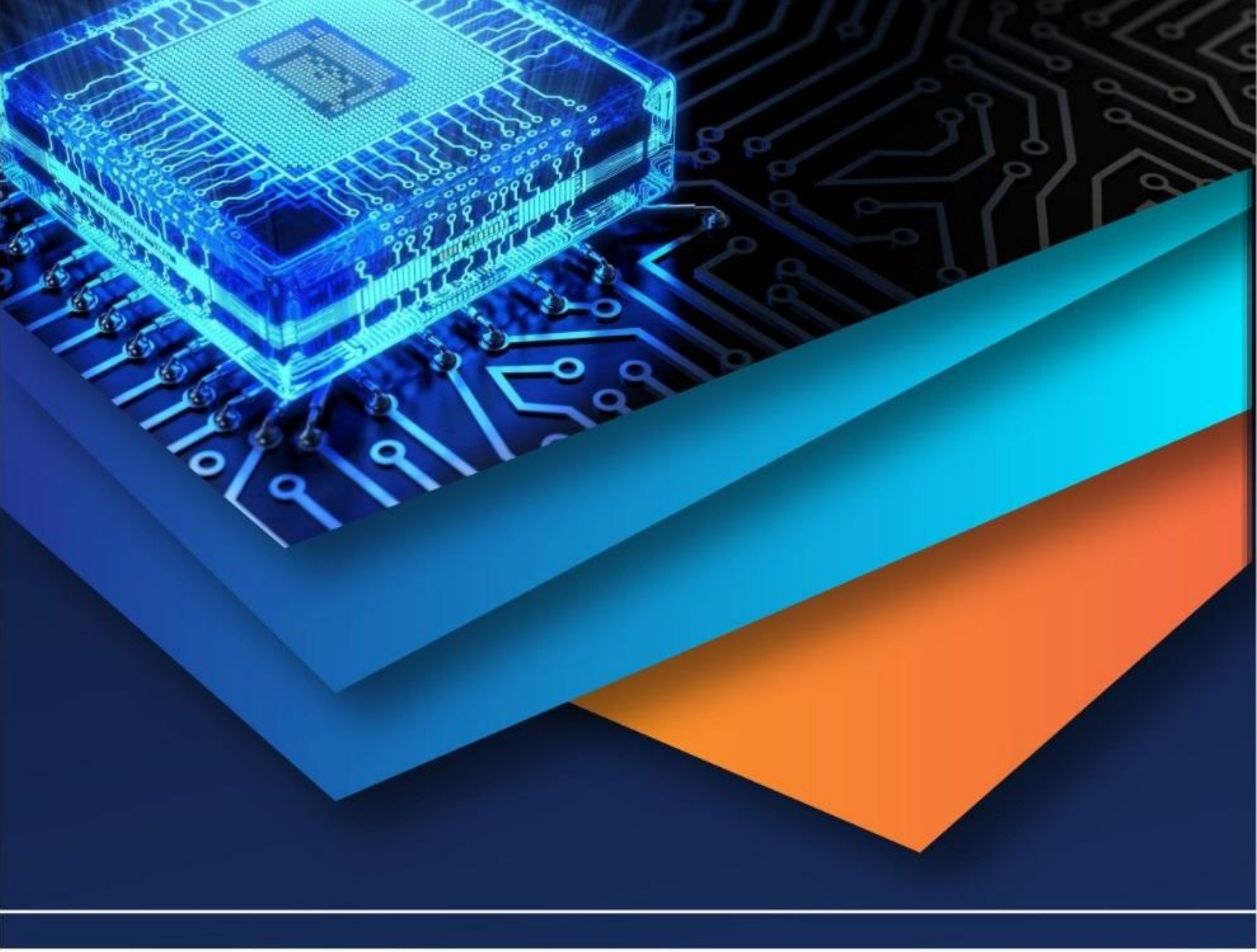
[3] Holm, Hannes & Sommestad, Teodor & Almroth, Jonas & Persson, Mats. (2011). A quantitative evaluation of vulnerability scanning. Inf. Manag. Comput. Security. 19. 10.1108/09685221111173058.

[4] Elkhodr M., Patel J.K., Mahdavi M., Gide E. (2020) Prevention of Cross-Site Scripting Attacks in Web Applications. In: Barolli L., Amato F., Moscato F., Enokido T., Takizawa M. (eds) Web, Artificial Intelligence and Network Applications. WAINA 2020. Advances in Intelligent Systems and Computing, vol 1150. Springer, Cham. https://doi.org/10.1007/978-3-030-44038-1_100.

[5] Erturk, Emre & Rajan, Angel. (2017). Web Vulnerability Scanners: A Case Study.

[6] W. Qianqian and L. Xiangjun, "Research and design on Web application vulnerability scanning service," 2014 IEEE 5th International Conference on Software Engineering and Service Science, Beijing, China, 2014, pp. 671-674, doi: 10.1109/ICSESS.2014.6933657.

[7] Noman, Muhammad & Iqbal, Muhammad & Manzoor, Engr. Dr. Amir. (2020). A Survey on Detection and Prevention of Web Vulnerabilities. International Journal of Advanced Computer Science and Applications. 11. 521-540. 10.14569/IJACSA.2020.0110665.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY