Cyberpulse Nexus: Monitoring Web Application Threats

1st Samarth Shinde, 2nd Sushant Singh, 3rd Raj Sovilkar, 4th Prof. Jaspreet Kaur
samshinde6352gmail.com, singhsushant.ss10@gmail.com, raj.sovilkar123@gmail.com.
Computer Engineering Department
Smt. Indira Gandhi College
of Engineering
Mumbai, India

***Abstract*—In recent times, use of web and web-based technologies have become more popular. The web applications are the most common interface for security-sensitive information and functionality available. As web applications are sources of sensitive data, they are prone to vast numbers of web-based attacks. The majority of these attacks happen because of vulnerabilities resulting from input validation problems. Although these vulnerabilities are easy to understand and mitigate, many web developers are unaware of these security aspects. Which results in more vulnerable web applications on the Internet. Among these, the most prominent vulnerabilities are SQL Injection and Cross Site Scripting (XSS). We researched a system which will scan the web application for the most frequent vulnerabilities in an automated manner. Our system detects flaws in web applications and presents a comprehensive report.**

***Index Terms*—SQL Injection, Cross Site Scripting, Web Application Testing, Security Scanner, Exploitation, Code Injection, Web Security, Machine Learning, Artificial Intelligence**

I. Introduction

As of January 2020, there have been over 1.74 billion websites on the web. Hackers attack every 39 seconds, on the average 2,244 times each day. This gives us the idea that many websites on the Internet are vulnerable to different attacks. As of the end of 2019, 42 percent of publicly facing websites are prone to SQL Injection and 19 percent Cross Site Scripting attacks. A security researcher has earned a 25,000 bug bounty after finding a DOM-based Cross-Site Scripting (XSS) vulnerability in one of the most popular social media sites ‘Facebook’. Another such attack, in August 2019, was on the famous coffee chain ‘Starbucks’ web services that created a way to access their critical database through the SQL Injection Vulnerability. From the above discussion, we can infer that Security plays an important role in developing websites. Unfortunately, web developers are not aware of these security aspects resulting in more vulnerable websites. Some of the most commonly occurring ones being SQLi, XSS, CSRF, Sensitive Data Exposure. So we are developing a system that will find these vulnerabilities in given web applications and report them to the user of the system. We are developing a system that will accept the target URL from the user.The system will then crawl the target URL in an Automated way using AI techniques and collect all the connected URLs. Then it will scan all collected URLs and it will test different payloads to exploit the vulnerabilities. Finally, a report will be generated which will contain the detected vulnerabilities and payloads used.

II. Objectives

- Develope a web-based application for seamless scanning for individuals with , focusing on industry standards.
- Create functionalities to deep scan and get desirable research.
- Implement the feature of report generation and port scanning.
- Monitoring the background and notifying the user.
- Usability and impact assessed in educational and everyday settings.

III. Scope

- To make an efficient, reliable, and user-friendly system for assisting web application scanning.
- Utilise gpt based scanning capabilities for real-time scanning to assist users.
- Achieveing unparalleled flexibility through its highly configurable scan engines, based on a YAML-based configuration. .
- Excels in subdomain discovery, pinpointing IP addresses and open ports, collecting endpoints, conducting directory and file fuzzing, capturing screenshots, and performing vulnerability scans. .
- Offers a range of pre-configured scan engines right out of the box, including Full Scan, Passive Scan, Screenshot Gathering, and the OSINT Scan Engine.

IV. Survey of Existing Systems

A. *A.G. Bacudio, X. Yuan, B.T.B.Chu and M. Jones. “An Overview of Penetration Testing”. Int. Journal of Network Security Its Applications Vol.3 (no. 6) (2011, November)*

Authors have presented a structured overview of Penetration Testing. The paper discusses the advantages and methodologies involved in conducting Penetration Testing. It further demonstrates how to conduct penetration testing using two demo sites. The findings in the paper show that penetration testing is a three-phase methodology consisting of preparation, test, and analysis phase. The test phase includes reconnaissance, scanning and vulnerability exploitation. It can be done manually or using automated tools.They have proposed a tool which helps in fingerprinting an organization. The tool presented is developed using Java which locates and saves organization specific data. The paper discusses the two types of reconnaissance and OSINT. It provides the possibility of network-based passive reconnaissance.

B. *Nagendran K, Adithyan A, Chethana R, Camillus P, Bala Sri Varshini K B. "Web Application Penetration Testing". International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-10 (2019, August 10)*

Authors discussed the approach to perform manual penetration testing in web applications and the research paper is suitable to act as a guide for testing OWASP Top 10 vulnerabilities. The paper discusses all the five phases of penetration testing. The objective of the paper is to provide knowledge about all the phases of penetration testing. the author has discussed an approach that lines up the web application security testing practices with the basic principles of security, namely, confidentiality, integrity and availability; which are collectively known as The CIA Triad. The approach proposed first signifies the requirements of each component of the CIA, mainly focusing on confidentiality. The paper depicts the most vulnerable processes in an application while highlighting the test-intensive areas. It then derives an acceptance criteria and a thought process to develop a test strategy covering both static and dynamic code analysis. It also describes the know-how to apply the DREAD model to categorize vulnerabilities spanning from critical to low intensity vulnerabilities.A Distributed Vulnerability Scanning on Machine Learning in the year 2019 by Xiaopeng TIAN, Di TANG, establishing standardized and quantified data sets for different industries and different businesses is of great help to improve the quality of testing. Commix: automating evaluation and exploitation of command injection vulnerabilities in Web applications published within the year 2019 by Anastasios Stasinopoulos, Christoforos Ntantogian and Christos Xenakis. It supports a plethora of functionalities that attempt to cover various exploitation scenarios such as different authentication mechanisms, custom headers, tornet working, attack vectors produced by programming languages, system user enumeration. Dimitris E. Simos, Jovan Zivanovic, Manuel Leithner proposed Automated Combinatorial Testing for Detecting SQL Vulnerabilities in Web Applications in the year 2019. It demonstrates that our approach can successfully evade faulty filtering mechanisms.

C. *Mujahid Shah, Sheeraz Ahmed, Khalid Saeed, Muhammad Junaid, Hamayun Khan, Ata-ur-rehman. "Penetration Testing Active Reconnaissance Phase – Optimized Port Scanning With Nmap Tool". 2019 International Conference on Computing, Mathematics and Engineering Technologies – iCoMET 2019. (2019, March)*

Authors have proposed different strategies to deal with large number of hosts and reduce time for a specific task. This paper deals with traffic accountability and time taken to complete a task during active reconnaissance using Nmap tool. The result section of this paper presents graphs and figures to depict variations is timings when the type of scan and number of ports vary. The results show that if scan is performed without any strategy then it affects the bandwidth and time to complete scan.Machine Learning for Web Vulnerability Detection The Case of Cross-Site Request Forgery published within the year 2020 by Stefano Calzavara, Mauro Conti, Riccardo Focardi, Alvise Rabitti, Gabriele Tolomei. It has the key advantage of offering a language-agnostic vulnerability-detection approach, which abstracts from the complexity of scripting languages and offers a consistent interface to the widest possible range of web applications. An efficient algorithm and tool for detecting dangerous website vulnerabilities in the year 2020 and written by Hoang Viet Long, Tong Anh Tuan, David Taniar, Nguyen Van Can, Hoang Minh Hue. The proposed new technique has the advantage of detecting attacks in nested SQL queries and giving a good performance. An Automated Composite Scanning Tool with Multiple Vulnerabilities within the year 2019 published by Xun Zhang, Jinxiong Zhao, Fan Yang, Qin Zhang, Zhiru Li, Bo Gong, Yong Zhi, Xuejun Zhang. It enables the automatic detection tool to implement automatic vulnerability scanning.

V. Web Security Vulnerabilities

A. ***Sql Injection:*** SQL injection attacks are one among the topmost threats in database-centric web applications and SQL injection vulnerabilities are the foremost serious Vulnerability types. SQL Injection allows the attacker to gain control over the database of an application. Every other website needs input from the user for a variety of reasons and if they are not validated properly, they might lead to some critical issues. Consider a login function where the user has to provide a username and password. These credentials are then validated at the backend through SQL query statements and if they are correct, then the user is successfully logged in.Now let's consider a situation where it can be abused. If the user provides some malformed inputs and the application accepts it as it is, then the attacker can leverage this to perform a database attack. We can see that the attacker Alice is providing username as 'admin" ;– ' and some arbitrary password.This results in breaking of the structure of the SQL query used at the backend. So the effective query will be 'SELECT * FROM Users WHERE Username="admin";– " AND Password="random";'. This will

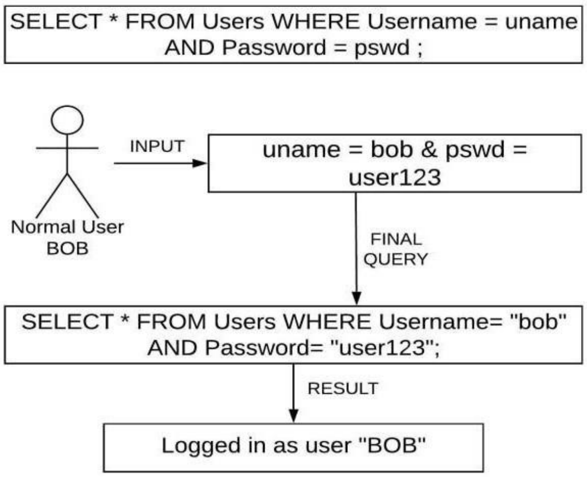
SELECT * FROM Users WHERE Username = uname AND Password = pswd ;
Normal User BOB
INPUT
uname = bob & pswd = user123
FINAL QUERY
SELECT * FROM Users WHERE Username= "bob" AND Password= "user123";
RESULT
Logged in as user "BOB"

Fig. 1. SQL Injection User Perspective

to lead to a change in the application logic, as the double-quote entered in the username will match the starting double quote of the query and as the ’–’ is considered as an identifier for comment in most of the relational databases, it simply comments out the succeeding part of the query. So the new query will be ’SELECT * FROM Users WHERE Username=”admin”;’. The attacker can now log in to an account without knowing the password.

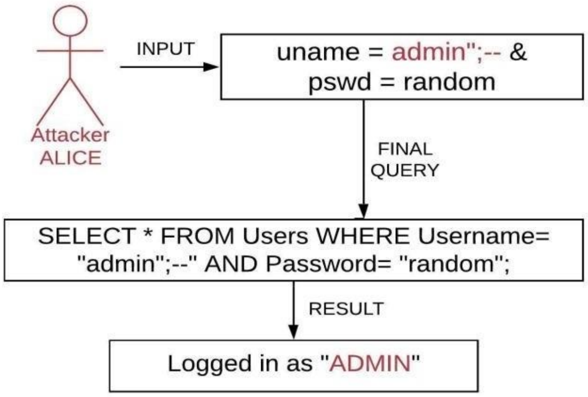
Attacker ALICE
INPUT
uname = admin";-- & pswd = random
FINAL QUERY
SELECT * FROM Users WHERE Username= "admin";--" AND Password= "random";
RESULT
Logged in as "ADMIN"

Fig. 2. SQL Injection Attackers Perspective

B. ***Cross Site Scripting:*** The Cross Site Scripting attack is a critical vulnerability that affects web application’s security. XSS attack is an injection of malicious script code into the web application by the attacker in the client-side within user’s browser or in the serverside within the database, this malicious script is written in JavaScript code and injected within untrusted input data on the web application [8]. Many applications provide the facility to search for specific content. Whenever the user searches for the required content, the relevant results are displayed on the webpage along with a search keyword entered by the user. Now let us consider the attacker’s perspective. The malicious user makes use of this search functionality as any of the normal user and checks whether the searched keyword gets reflected on the resultant page returned by the application. If it succeeds, then the attacker comes to know that there is a possibility of XSS to take place at that particular location.

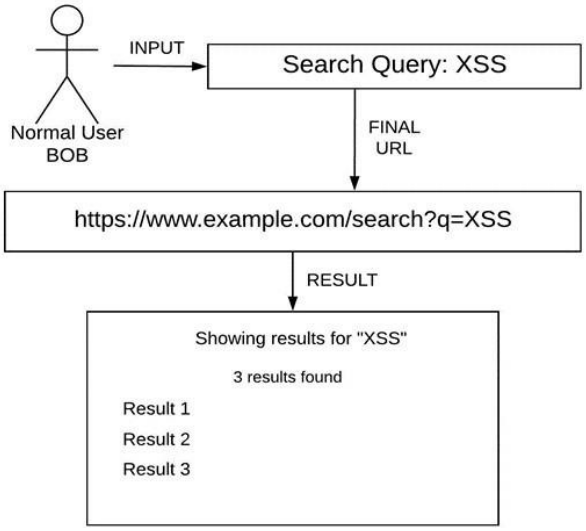

Fig. 3. XSS-User Perspective

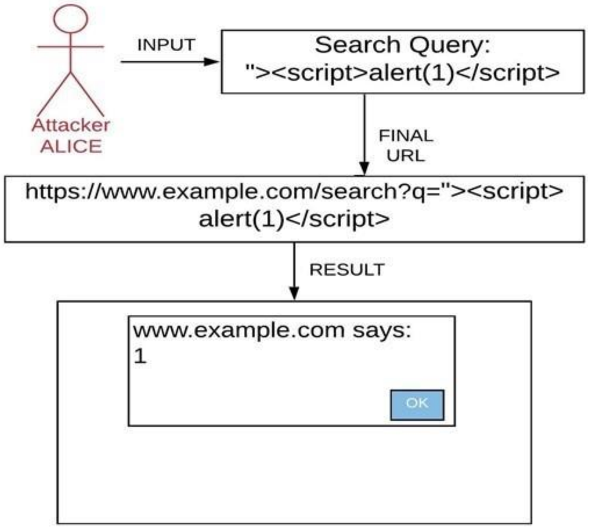
Attacker ALICE
INPUT
Search Query: "><script>alert(1)</script>
FINAL URL
https://www.example.com/search?q="><script>alert(1)</script>
RESULT
www.example.com says:
1
OK

Fig. 4. XSS - Attackers Perspective

If the application does not perform either encoding or filtering on the search query given by the user, then it might be possible for an attacker to break out of the previous HTML tag and insert a new one. From the figure given below, we can see that the attacker is able to insert a new script tag that can be used for malicious purposes. Being able to insert a new script tag can have several consequences, including but not limited to, stealing session cookies, bypassing CSRF protections, theft of user's personal and sensitive data. In some extreme scenarios, it might be possible to exploit the user's browser by leveraging the XSS.

VI. Proposed System

A. System Architecture

Users: Store information about users of the CyberPulse Nexus system, including usernames, passwords (hashed), roles, contact information, and authentication tokens.

Web Applications: Maintain details about the web applications being monitored, such as URLs, names, descriptions, owner organizations, and status (active/inactive).

Vulnerabilities: Store information about vulnerabilities discovered during monitoring, including CVE IDs, descriptions, severity levels, affected web applications, and remediation steps.

Scans: Track details of scans performed on web applications, including scan timestamps, scan types (e.g., full scan, quick scan), and associated user IDs.

Scan Results: Store results of each scan, including findings such as open ports, detected vulnerabilities, and system configurations. Include references to the scan ID and related web application.

Events and Alerts: Record security events and alerts generated by the system, including timestamps, event types (e.g., intrusion attempt, suspicious activity), affected web applications, and severity levels.

Logs: Maintain logs of system activities and user interactions, including timestamps, actions performed, user IDs, and related entities (e.g., scanned web applications, viewed vulnerabilities).

B. Project Scope

• If the web application is not having the robots.txt file then the user has to explicitly specify the restricted URLs. • The system will scan the target application and check if the web application is having any of these vulnerabilities: – SQL Injection – Cross Site Scripting • The report will be generated consisting of endpoint affected, payload used, and generalized remediation.

C. User Classes and characteristics

• WebSpider: – Robots.txt Parsing Checks for the presence of the robots.txt file and if present, collect allowed and disallowed URLs. – URL Parsing: All URLs specified within the anchor tag from the current page are saved in a List. Multiple threads will be created to crawl different hyperlinks simultaneously. Relative URLs (like /admin or footer) are converted into Absolute URL (like https://example.com/admin or https://example.comfooter) URLs which are not in the scope of target application are removed from the list (for example twitter.com or instagram.com) Hyperlinks with 'mailto:' or 'javascript:' and those pointing to static file types like images, pdfs, fonts, etc. are also removed. • WebScanner: – Search for form elements in crawled URLs. From the listed form elements, find out input fields. – Pass the appropriate payloads to the input field and save the response received from the server. • SQL Injection: – For an Error-Based SQL Injection attack, we try to break the syntax of the SQL query being used by the server by passing SQL-special characters (E.g.',", etc.) through user input. – For Union SQL injections, a dataset consisting of SQL queries of specific types will be created. – In the user input, these payloads will be passed to the server to check if SQL query is well formed after inserting the given payload. – The system requires the target URL to be entered by the user. – If the response from the server for the payload is similar to the usual response, then we can infer that SQL injection is possible. • Cross Site Scripting: – For Cross Site Scripting, a dataset will be created consisting of different XSS payloads. – These payloads will then be tested against all input fields present on that web page. – The responses are checked for the presence of a particular payload and to check if XSS is successful. – If successful, that part of the web page is considered vulnerable and a report will be generated giving a detailed knowledge about the vulnerability detection.

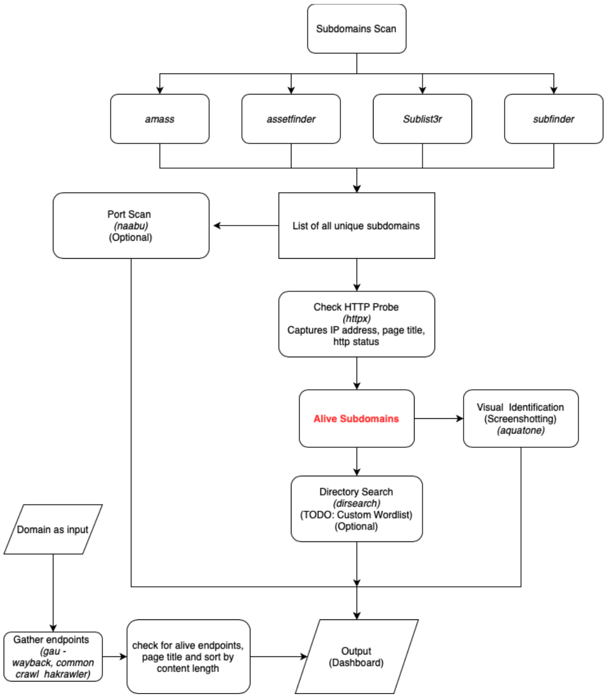

Fig. 5. System Architecture

VII. Other Specifications

A. Advantages

It Supports automated and reliable crawling. It Optimized use of the number of threads to control the load on the target application and gives detailed vulnerability analysis report and User-friendly GUI.

B. Limitations

This Model can currently handle non-CAPTCHA registrations and logins. Possible to detect first-order SQL Injection and XSS vulnerabilities through the way of automated scanning. Current focus is on small to medium-sized web applications.

C. Applications

Sys Admin: Sys Admin is a super user that has permission to modify system and scan related configurations, scan engines, create new users, add new tools etc. Super user can initiate scans and subscans effortlessly.

Penetration Tester: Penetration Tester will be allowed to modify and initiate scans and subscans, add or update targets, etc. A penetration tester will not be allowed to modify system configurations.

Auditor: Auditor can only view and download the report. An auditor can not change any system or scan related configurations nor can initiate any scans or subscans.

GPT Vulnerability Report Generation: Get ready for the future of penetration testing reports with reNgine's groundbreaking feature: ”GPT-Powered Report Generation”! With the power of OpenAI's GPT, reNgine provides with detailed vulnerability descriptions, remediation strategies, and impact assessments that read like they were written by a human security expert! But that's not all! Our GPT-driven reports go the extra mile by scouring the web for related news articles, blogs, and references.

GPT-Powered Attack Surface Generation: , reNgine seamlessly integrates with GPT to identify the attacks that you can likely perform on a subdomain. By making use of reconnaissance data such as page title, open ports, subdomain name etc, reNgine can advice you the attacks you could perform on a target. reNgine will also provide you the rationale on why the specific attack is likely to be successful.

Continuous monitoring: Continuous monitoring is at the core of reNgine's mission, and it's robust continuous monitoring feature ensures that their targets are under constant scrutiny. With the flexibility to schedule scans at regular intervals, penetration testers can effortlessly stay informed about their targets.

VIII. Conclusion

In summary, We have tried to find some of the common vulnerabilities on the web, such as SQL Injection and Cross Site Scripting. We have proposed an algorithm and further enhancements in the system to improve the efficiency of the vulnerability detection in the web application. We proposed a system that will crawl the entire web application, scan different types of vulnerabilities, and generate a report specifying an overview of the detected vulnerabilities.In this article, the focus is on the important process known as reconnaissance which is a vital step in the website hacking process. Different types of reconnaissance processes are involved to gather the essential information. After extensive research, it has been noticed that the COVID-19 pandemic has highlighted the seriousness of data security breaches that could be potentially dangerous in the online world. There are many free online tools with guides on how to use is available even to amateurs who are looking to extract information for their benefit. This has become a serious threat to the personal information of individuals as well as the company. It is important to allocate a budget and hire experts who will be able to protect the information that might be vulnerable to threats. Therefore, companies can implement some security measures such as SSL certificates, closing of ports when not in use, and having website services updated regularly to reduce the threats present. These are some of the measures that could help protect the websites. There is a lot of competition present in the world, so everyone tries to take the easy step. It can be concluded that reconnaissance is an important step taken in the process of website hacking. Through the research, it was established that multiple free tools can be used to perform the reconnaissance. It is important to always keep a watch on attackers and make sure that the websites are implemented with the necessary security measures at regular periods.

IX. Future Scope

The ”Cyberpulse Nexus: Monitoring Web Applications Threats” project offers a promising future scope with several avenues for advancement and growth. These include:

Enhanced Reporting and Visualization can use improving the reporting and visualization capabilities of the tool can provide users with more insightful and actionable insights. Incorporating data visualization techniques can make it easier for security professionals to interpret assessment results. Integration with Additional Security Tools expanding the tool's integration capabilities to collaborate seamlessly with a broader range of security tools, frameworks, and platforms can create a unified ecosystem for comprehensive security assessment and monitoring. Threat Intelligence Integration incorporating threat intelligence feeds and databases can empower the tool to stay updated with the latest threats and vulnerabilities. Real-time threat data can enhance the accuracy of assessments and proactive threat prevention.

X. Acknowledgement

It is a pleasant task to express gratitude to all those who contributed in many ways to this project and made it an unforgettable experience for us. First of all we would like to thank our internal guide Prof. Jaspreet Kaur. This work would not have been possible without her guidance, support and encouragement. We are highly indebted to Dr. Sunil Chavan, Prin- cipal, Smt. Indira Gandhi College of Engineering, Dr. KT Patil, Head of the Department, Computer Department and also all the faculty members, We are thankful to lord for keeping us energized that every end seems to be beginning. [2] [7] [5] [9] [1] [8] [4] [3] [10] [6]

References

[1] Abdulrahman Alzahrani, Ali Alqazzaz, Ye Zhu, Huirong Fu, and Nabil Almashfi. Web application security tools analysis. In *2017 ieee 3rd international conference on big data security on cloud (bigdatasecurity), ieee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security (ids)*, pages 237–242. IEEE, 2017.

[2] Mark Curphey and Rudolph Arawo. Web application security assessment tools. *IEEE Security & Privacy*, 4(4):32–41, 2006.

[3] Ulfar Erlingsson, V Benjamin Livshits, and Yinglian Xie. End-to-end web application security. In *HotOS*, 2007.

[4] Andrew Hoffman. *Web application security*. " O'Reilly Media, Inc.", 2024.

[5] Yao-Wen Huang, Shih-Kun Huang, Tsung-Po Lin, and Chung-Hung Tsai. Web application security assessment by fault injection and behavior monitoring. In *Proceedings of the 12th international conference on World Wide Web*, pages 148–159, 2003.

[6] Sandeep Kumar, Renuka Mahajan, Naresh Kumar, and Sunil Kumar Khatri. A study on web application security and detecting security vulnerabilities. In *2017 6th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*, pages 451–455. IEEE, 2017.

[7] Xiaowei Li and Yuan Xue. A survey on web application security. *Nashville, TN USA*, 25(5):1–14, 2011.

[8] JD Meier. Web application security engineering. *IEEE Security & Privacy*, 4(4):16–24, 2006.

[9] Bala Musa Shuaibu, Norita Md Norwawi, Mohd Hasan Selamat, and Abdulkareem Al-Alwani. Systematic review of web application security development model. *Artificial Intelligence Review*, 43:259–276, 2015.

[10] Jahanzeb Shahid, Muhammad Khurram Hameed, Ibrahim Tariq Javed, Kashif Naseer Qureshi, Moazam Ali, and Noel Crespi. A comparative study of web application security parameters: Current trends and future directions. *Applied Sciences*, 12(8):4077, 2022.