WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Web Vulnerability Scanner for XSS and SQL Injection

Dr. Satish B Basapur[1] ,Padmavathi B R[2], Hiranmayi K[3], Sinchana C[4] ,Varalakshmi M[5]

[1]Assistant Professor, Dept. Of ISE, Dr. Ambedkar Institute of Technology, Karnataka, India
[2,3,4,5] Student, Dept. Of ISE, Dr. Ambedkar Institute of Technology, Karnataka, India

ABSTRACT

This project introduces a web vulnerability scanner designed to identify and mitigate two of the most prevalent security vulnerabilities in web applications: SQL Injection (SQLI) and Cross-Site Scripting (XSS). The scanner, implemented using Python, employs systematic testing techniques to analyse web pages and detect exploitable input fields. Deployed on Kali Linux, a widely recognized operating system for cybersecurity, the tool integrates seamlessly into penetration testing workflows while offering user-friendly operation and reliable performance.By focusing on SQLI and XSS vulnerabilities, the project underscores the importance of proactive vulnerability detection in safeguarding digital assets.

Keywords: Cybersecurity, Vulnerability assessment, Web Vulnerability Scanner, Cross-Site Scripting (XSS), SQL Injection (SQLI)

1. INTRODUCTION

A web vulnerability scanner is a specialized tool designed to detect and analyse security weaknesses in web applications. As cyber threats grow more sophisticated, web applications remain prime targets for attackers exploiting vulnerabilities such as SQL Injection (SQLI) and Cross-Site Scripting (XSS). Among the most prevalent and harmful vulnerabilities are SQL Injection (SQLI) and Cross-Site Scripting (XSS), which attackers exploit to gain unauthorized access, manipulate databases, or compromise sensitive user information. The consequences of these vulnerabilities range from data breaches to service disruptions, making web application security a critical area of concern for developers and organizations worldwide.The web vulnerability scanner developed in this project serves as a proactive security assessment tool, focusing specifically on identifying SQLI and XSS vulnerabilities.These vulnerabilities are widely exploited in cyberattacks and can lead to unauthorized access, data theft, or malicious code execution. Built using Python, the scanner leverages a modular and efficient approach to systematically analyse web pages, targeting input fields and application logic for potential flaws. By simulating attack scenarios with carefully crafted payloads, the scanner mimics real-world exploitation techniques to ensure comprehensive vulnerability detection. This approach not only identifies vulnerabilities but also demonstrates how they can be exploited, providing website owners with actionable insights to mitigate risks.Designed with accessibility and ease of use in mind, the scanner includes an intuitive user interface that guides users through the scanning process. This feature ensures that even individuals with minimal technical expertise can effectively identify vulnerabilities in their applications. Additionally, the tool’s adaptability allows it to test various types of web applications, making it a versatile solution for developers, system administrators, and security professionals.

1.1 Cross Site Scripting (XSS)

Cross-Site Scripting (XSS) is recognized as one of the most pervasive and critical vulnerabilities in web applications. This security flaw arises when attackers inject malicious scripts into a trusted web application, which are subsequently executed in the browser of unsuspecting users accessing the compromised site. XSS attacks typically exploit a web application’s inability to adequately sanitize user inputs or properly encode outputs, thereby enabling the injection of harmful code. The root cause of XSS vulnerabilities lies in dynamically generated web pages that fail to implement rigorous validation and sanitization mechanisms for user-provided inputs. This oversight allows attackers to embed malicious scripts, commonly in JavaScript that execute on the client side. Such scripts can lead to severe consequences, including unauthorized access to sensitive information, session hijacking, or the unintended manipulation of the web application's functionality. Common attack vectors include search bars, error messages, input forms, and web forums, where inadequate input validation is exploited to inject malicious payloads. Given the increasing sophistication of XSS attacks and their potential to compromise user data and system integrity, there is a growing need for advanced tools that can proactively detect and mitigate such vulnerabilities. Addressing this issue requires the development and deployment of robust, efficient vulnerability scanning solutions capable of identifying and neutralizing security flaws before they are exploited by malicious actors. The scanner addresses XSS vulnerabilities by identifying points where user input is reflected in the application output without proper sanitization. This process effectively pinpoints vulnerable endpoints, enabling users to mitigate XSS risks.

* *Corresponding author.* Tel.: ; fax: +0-000-000-0000.
E-mail address:

1.2 *SQL Injection*

SQL Injection is one of the most severe and commonly exploited vulnerabilities in web applications, posing significant threats to the confidentiality, integrity, and availability of databases. This vulnerability arises when an application improperly handles user input in SQL queries, allowing attackers to inject malicious SQL code into the database query logic. These attacks exploit the failure of applications to sanitize or validate user-provided inputs, thereby enabling unauthorized manipulation of database operations. The underlying cause of SQL Injection lies in dynamic query execution without proper input sanitization. Attackers take advantage of this oversight by injecting malicious payloads through input fields, URLs, or HTTP headers, which are subsequently executed by the database. This can result in unauthorized data access, retrieval of sensitive information, alteration or deletion of records, and in some cases, complete compromise of the underlying database server. SQL Injection attacks are particularly dangerous in applications that allow direct interaction with the database through unvalidated inputs, such as login forms, search fields, or parameterized URLs. The impact of SQL Injection attacks can be devastating, ranging from unauthorized access to confidential user information to complete database breaches. As a result, there is an urgent demand for robust security measures and tools capable of detecting and mitigating SQL Injection vulnerabilities. Developing advanced scanning solutions that incorporate effective input sanitization, parameterized queries, and proactive vulnerability assessments is critical to ensuring the security and resilience of modern web applications. The scanner targets SQL Injection vulnerabilities by injecting specially crafted SQL payloads into input fields or parameters that interact with the database. It monitors the application's behaviour, such as error messages or unexpected database outputs, to detect improper query handling. This allows users to identify and address SQL Injection risks, safeguarding their application's database integrity.

2. LITERATURE SURVEY

Title: Performance comparison on sql injection and xss detection using open source vulnerability scanners

- **Author**: B. Zukran, M. M. Siraj
- **Publication**: 2021 International Conference on Data Science and Its Applications (ICoDSA). IEEE, 2021
- **Methodology**: This study investigated the efficacy of automated vulnerability scanners in identifying security weaknesses in web applications, with a particular focus on SQL Injection (SQLI) and Cross-Site Scripting (XSS) vulnerabilities, which pose significant risks to users. Their research underscored the critical importance of reliability in vulnerability scanning tools, emphasizing the need for transparency regarding the accuracy and detection capabilities of open-source solutions. The study highlighted the necessity of comparing automated vulnerability scanners to evaluate their performance, particularly concerning their precision rates and detection coverage for SQLI and XSS vulnerabilities.

Title: Testing and comparing web vulnerability scanning tools for sql injection and xss attacks

- **Author**: J. Fonseca, M. Vieira, H. Madeira
- **Publication**: 13th Pacific Rim international symposium on dependable computing (PRDC 2007). IEEE, 2007
- **Methodology**: The researchers assessed detection accuracy and coverage by injecting controlled vulnerabilities into web applications and subjecting them to scans by selected tools.

Title: Sql injection attacks and vulnerabilities

- **Author**: I. Jacob, M. Pirnau
- **Publication**: *Journal of Information Systems & Operations Management 2020*

Title: Analysis of web vulnerability using open-source scanners on different types of small entrepreneur web applications

in Malaysia

- **Author**: A. G. Buja, N. N. M. A. A. Low, A. F. Zolkeplay, N. A. Azam, F. M. Isa
- **Publication**: *Journal of Advanced Research in Applied Sciences and Engineering Technology*, vol. 40, 2024

3. METHODOLOGY

The development process of the proposed web vulnerability scanner begins with an exhaustive literature review to acquire an in-depth understanding of the current state of research, established methodologies, and best practices in the domain of web vulnerability analysis. This foundational research informs the precise definition of the scanner's requirements and functional specifications. These requirements include the classification of vulnerabilities to be identified (e.g., SQL Injection (SQLi), Cross-Site Scripting (XSS)), the scope of vulnerability assessments (e.g., target domains, specific attack vectors), and the format of the output (e.g., detailed analytical reports, alerts with severity metrics).

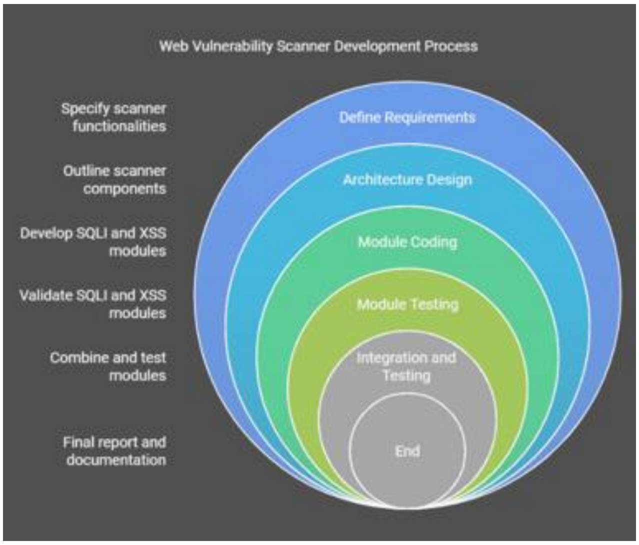

Web Vulnerability Scanner Development Process
Specify scanner functionalities
Outline scanner components
Develop SQLI and XSS modules
Validate SQLI and XSS modules
Combine and test modules
Final report and documentation
Define Requirements
Architecture Design
Module Coding
Module Testing
Integration and Testing
End

Fig. 1 -Web Vulnerability Scanner Development Process

Following the requirements definition, the architectural design phase is undertaken. This phase delineates the system's overall structure, specifying the critical components, their interdependencies, and the data flow among them. The resulting architectural framework serves as a reference blueprint, guiding subsequent phases of the development process. The implementation phase is initiated with the development of the SQL injection detection module, utilizing Python for its extensive library ecosystem and powerful capabilities in web scraping, data parsing, and network communication. This module undergoes rigorous validation through an extensive test suite encompassing both positive and negative test scenarios to ensure operational accuracy, reliability, and resilience under diverse conditions. Subsequently, the XSS detection module is developed, adhering to an iterative and systematic approach. This module, also implemented in Python, leverages advanced techniques for detecting potential XSS vulnerabilities, including the analysis of HTML and JavaScript code for indicators of malicious script injection. The iterative development cycle prioritizes robust design and incremental testing for enhanced reliability. Following the independent development of these modules, the integration phase is executed to merge the SQLi and XSS detection modules into a unified and cohesive web vulnerability scanning system. This phase involves comprehensive testing and debugging to ensure seamless interaction, efficient data flow, and strict compliance with the established system requirements. Integration testing evaluates both module interoperability and system-wide performance. The final phase encompasses a detailed evaluation of the scanner's efficacy. This includes benchmarking its performance, detection accuracy, and operational efficiency against state-of-the-art vulnerability scanning tools. The evaluation results are critically analysed to identify optimization opportunities, recommend enhancements, and propose directions for future research endeavours. Throughout the development lifecycle, meticulous documentation is maintained, encompassing all design rationales, implementation details, testing methodologies, and evaluation findings. This comprehensive record not only ensures transparency but also serves as a foundation for potential advancements in the domain of web vulnerability scanning.

4. SYSTEM AND HARDWARE DESCRIPTION

4.1 Test Environment Kali Linux

Kali Linux serves as the foundational platform for the testing environment of the proposed web vulnerability scanner due to its established reputation and widespread adoption within the cyber security domain. As a Debian-based, open-source operating system, Kali Linux is meticulously designed to address the specific requirements of network analysts and penetration testers. It provides an extensive repository of pre-installed, industry-standard tools that are integral to ethical hacking and security assessments. The platform's widespread preference among cyber security professionals is attributed to several key factors. Its pre-configured environment significantly reduces the time and effort required for manual installations and complex configurations, thereby improving accessibility, particularly for individuals newly entering the field of ethical hacking. Additionally, Kali Linux demonstrates remarkable performance efficiency, even on systems with constrained hardware capabilities, ensuring its suitability for a broad spectrum of users with diverse system configurations. The inclusion of multilingual support further enhances its global usability, accommodating the linguistic preferences of users from various regions. As an open-source distribution, Kali Linux upholds principles of transparency and adaptability. Users have full access to its source code, enabling them to review, customize, and extend the platform to meet specific operational or research requirements. This open-source ethos fosters robust community participation and iterative enhancements, ensuring that the platform remains responsive to the dynamic and evolving challenges of cyber security. Functionally, Kali Linux offers a comprehensive suite of specialized tools essential for penetration testing and security analysis. Among its extensive toolkit are notable utilities such as Aircrack-ng, Nmap, THC Hydra, Nessus, and Wireshark. These tools equip security practitioners to conduct thorough evaluations of network defences, identify vulnerabilities with precision, and propose actionable remediation strategies. The platform's holistic approach and robust capabilities position it as an indispensable resource for advanced cybersecurity operations.

4.2 Software Used – Python

Python's succinct and expressive syntax, characterized by its conciseness, substantially minimizes the verbosity commonly associated with traditional programming languages such as C or Java. This simplicity enhances accessibility and usability, making it an ideal choice for both novice and experienced developers. The ability to achieve complex functionalities with fewer lines of code significantly optimizes the development workflow, reducing implementation overhead and accelerating project timelines. The selection of Python for this project is further validated by its exceptional capacity to efficiently handle intricate tasks, supported by a vast and active developer community. The language's widespread adoption within the developer ecosystem fosters an environment conducive to collaborative innovation. Python’s advanced memory management system abstracts away the complexities of manual memory allocation, allowing developers to focus on the core objectives of the project without being encumbered by low-level technical intricacies. The language’s intrinsic speed, coupled with its adaptability, ensures the rapid generation of scanning results, thereby addressing emerging security threats with agility and precision. Moreover, Python's robust automation capabilities are instrumental in optimizing the efficiency of the project. By automating critical tasks such as establishing network connections, extracting form data, and logging scan results, Python ensures consistent and reliable operations. This automation not only reduces the potential for human error but also enhances the accuracy and effectiveness of web vulnerability detection, making it a critical asset for addressing modern cybersecurity challenges.

5. IMPLEMENTATION

The implementation of the web vulnerability scanner follows a structured and methodical approach to ensure robust functionality and comprehensive coverage of potential security issues. The process begins with an initial decision point where the user selects the desired scanning mode—either Cross-Site Scripting (XSS) vulnerability detection, SQL Injection (SQLi) vulnerability detection, or terminates the program. This selection directs the scanner toward the appropriate operational pathway.

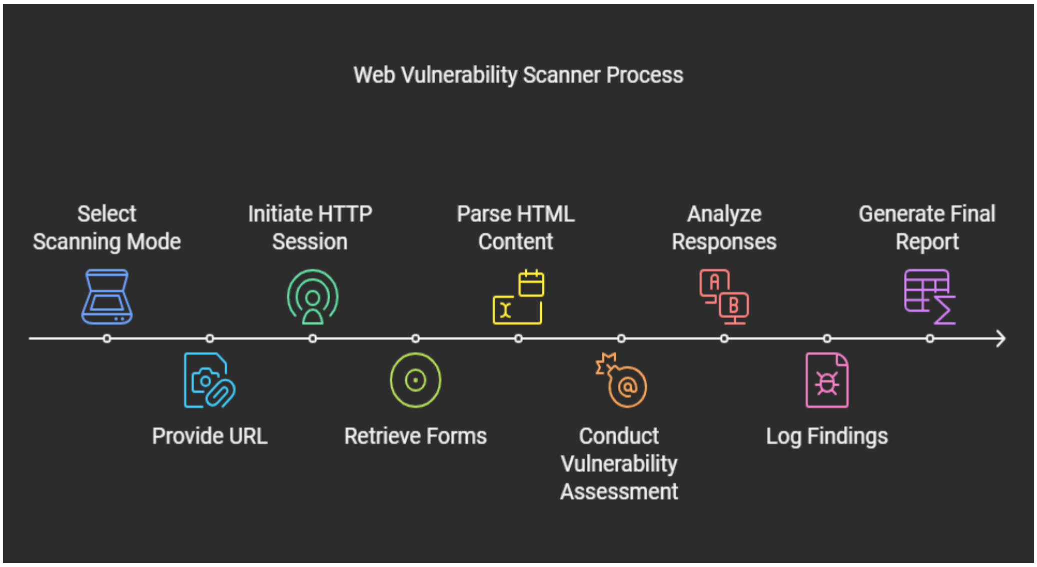


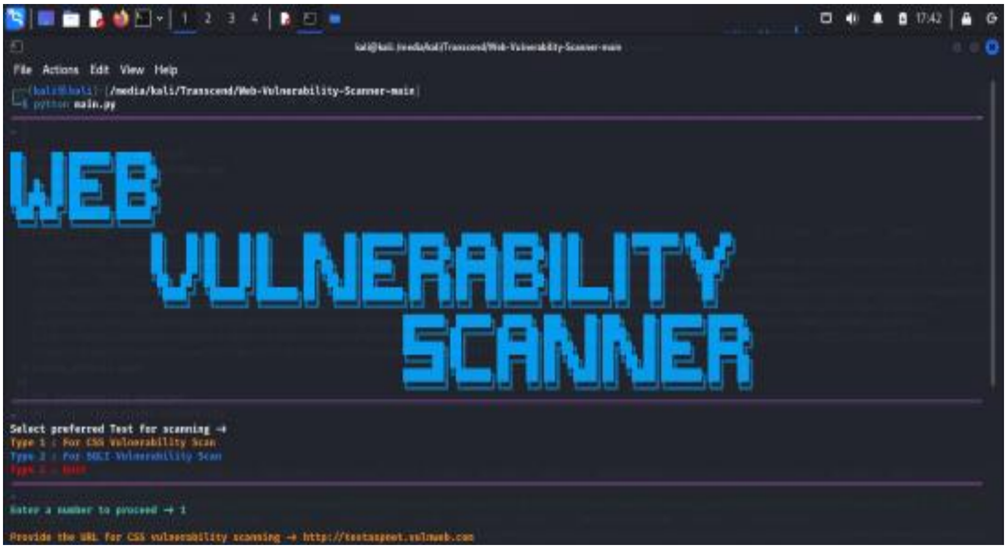
Fig. 2 –Implementation of web vulnerability scanner

Upon mode selection, the user is prompted to provide the Uniform Resource Locator (URL) of the target web application. This URL serves as the entry point for initiating an HTTP session, establishing a communication channel with the target server. Through this session, the scanner systematically retrieves all forms available on the target web page, enabling an exhaustive assessment of input vectors. Subsequent to form retrieval, the scanner parses the HyperText Markup Language (HTML) content to extract detailed information about the forms, including input fields, parameters, and associated metadata. This extracted data forms the foundation for subsequent vulnerability analysis. The scanner employs sophisticated parsing mechanisms to ensure the accurate identification and handling of diverse web page structures. In the vulnerability assessment phase, the scanner employs a predefined library of payloads tailored for the specific vulnerabilities under investigation. These payloads are injected into the identified input fields to simulate potential attack scenarios. The responses from the target application are rigorously analysed to determine whether any vulnerabilities are present. If vulnerability is detected, the affected component is flagged as compromised, and a detailed record of the finding is logged for reporting purposes. In cases where no vulnerabilities are identified, the scanner continues to evaluate other components of the application, ensuring comprehensive coverage of the target's attack surface. The scanning process concludes with a final report summarizing the detected vulnerabilities, if any, or confirming the absence of identified security issues. This implementation process emphasizes modularity, enabling seamless integration and interaction between various components. The adoption of standardized testing methodologies, robust parsing mechanisms, and systematic input testing ensures high accuracy and reliability in vulnerability detection. Furthermore, the design is inherently extensible, allowing for future enhancements to accommodate emerging threat vectors and evolving cybersecurity standards.

6. RESULTS

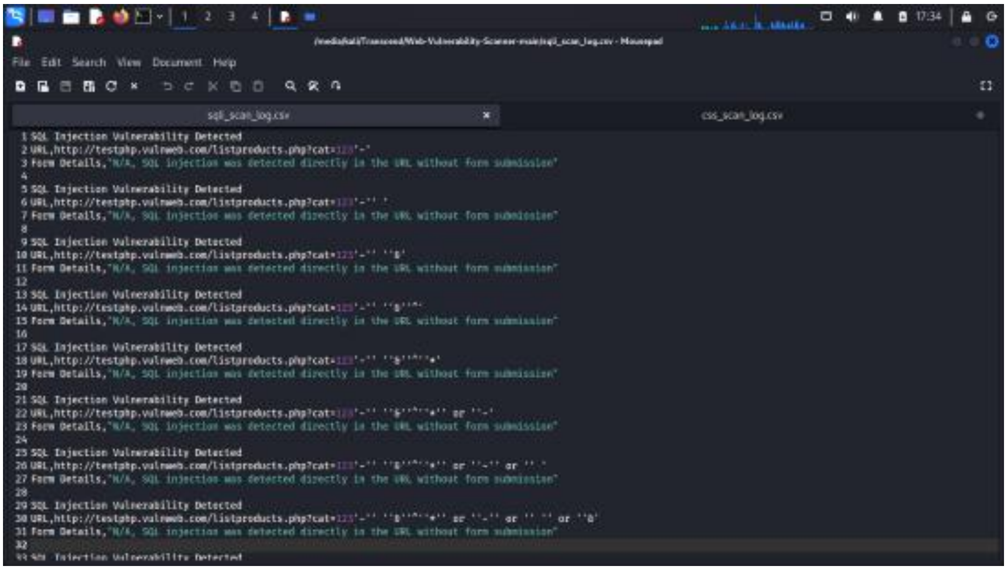
This project effectively illustrates the creation and deployment of a web vulnerability scanner aimed at identifying critical security flaws, such as SQL Injection (SQLI) and Cross-Site Scripting (XSS), in web applications.In examining the results of the web vulnerability scanner project, it is evident that the final product offers a good solution for detecting SQL Injection and Cross Site Scripting vulnerabilities in web applications.

Secondly, the scanner reduces the entry barriers for vulnerability scanning by requiring minimal technical knowledge to use. Its intuitive interfaces and clear documentation ensure a seamless scanning process, making it accessible to users without specialized cybersecurity expertise. In comparison, manual scanning requires advanced skills, expertise, and experience, restricting its accessibility and efficiency.


kali@kali: /media/kali/Transcend/Web-Vulnerability-Scanner-main
File Actions Edit View Help
17:42

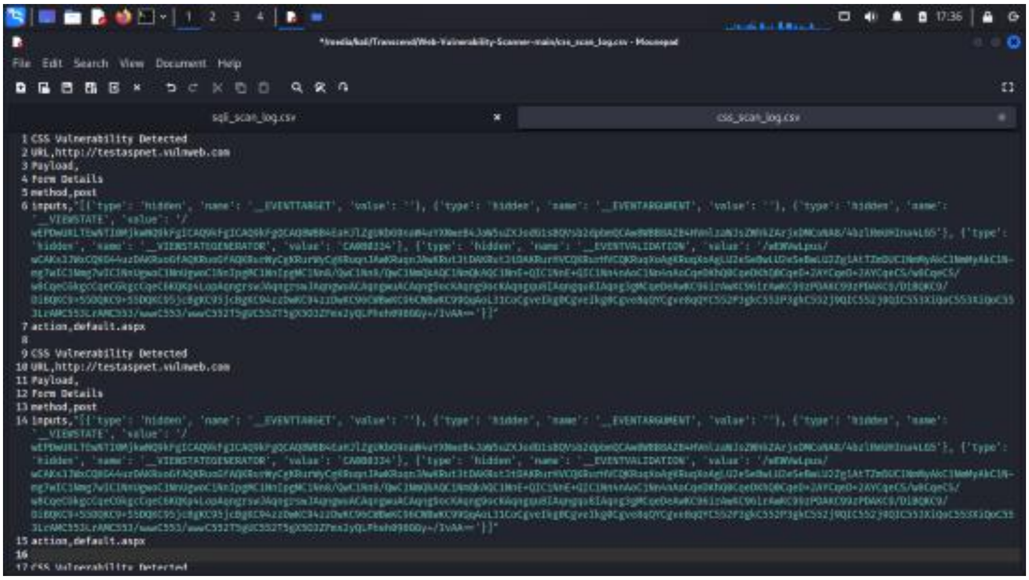
```
┌──(kali㉿kali)-[/media/kali/Transcend/Web-Vulnerability-Scanner-main]
└─$ python main.py

WEB
    VULNERABILITY
                  SCANNER

Select preferred Test for scanning →
Type 1 : For CSS Vulnerability Scan
Type 2 : For SQLI-Vulnerability Scan
Type 3 : Exit

Enter a number to proceed → 1

Provide the URL for CSS vulnerability scanning → http://testaspnet.vulnweb.com
```

Fig. 3 –User Terminal Interface


/media/kali/Transcend/Web-Vulnerability-Scanner-main/sqli_scan_log.csv - Mousepad
File Edit Search View Document Help
sqli_scan_log.csv
css_scan_log.csv
17:34

```
SQL Injection Vulnerability Detected
URL,http://testphp.vulnweb.com/listproducts.php?cat=123'-'
Form Details,"N/A, SQL injection was detected directly in the URL without form submission"

SQL Injection Vulnerability Detected
URL,http://testphp.vulnweb.com/listproducts.php?cat=123'-'' '
Form Details,"N/A, SQL injection was detected directly in the URL without form submission"

SQL Injection Vulnerability Detected
URL,http://testphp.vulnweb.com/listproducts.php?cat=123'-'' ''&'
Form Details,"N/A, SQL injection was detected directly in the URL without form submission"

SQL Injection Vulnerability Detected
URL,http://testphp.vulnweb.com/listproducts.php?cat=123'-'' ''&''^'
Form Details,"N/A, SQL injection was detected directly in the URL without form submission"

SQL Injection Vulnerability Detected
URL,http://testphp.vulnweb.com/listproducts.php?cat=123'-'' ''&''^''*'
Form Details,"N/A, SQL injection was detected directly in the URL without form submission"

SQL Injection Vulnerability Detected
URL,http://testphp.vulnweb.com/listproducts.php?cat=123'-'' ''&''^''*'' or ''-'
Form Details,"N/A, SQL injection was detected directly in the URL without form submission"

SQL Injection Vulnerability Detected
URL,http://testphp.vulnweb.com/listproducts.php?cat=123'-'' ''&''^''*'' or ''-'' or '' '
Form Details,"N/A, SQL injection was detected directly in the URL without form submission"

SQL Injection Vulnerability Detected
URL,http://testphp.vulnweb.com/listproducts.php?cat=123'-'' ''&''^''*'' or ''-'' or '' '' or ''&'
Form Details,"N/A, SQL injection was detected directly in the URL without form submission"

SQL Injection Vulnerability Detected
```

Fig. 4 –SQL Injection Detection Documentation


*/media/kali/Transcend/Web-Vulnerability-Scanner-main/css_scan_log.csv - Mousepad
File Edit Search View Document Help
sql_scan_log.csv
css_scan_log.csv
17:36

```
CSS Vulnerability Detected
URL,http://testaspnet.vulnweb.com
Payload,
Form Details
method,post
inputs,"[{'type': 'hidden', 'name': '__EVENTTARGET', 'value': ''}, {'type': 'hidden', 'name': '__EVENTARGUMENT', 'value': ''}, {'type': 'hidden', 'name': '__VIEWSTATE', 'value': '/[illegible]'}, {'type': 'hidden', 'name': '__VIEWSTATEGENERATOR', 'value': 'CA0B0334'}, {'type': 'hidden', 'name': '__EVENTVALIDATION', 'value': '/wEWAwLpus/[illegible]/IvAA=='}]"
action,default.aspx

CSS Vulnerability Detected
URL,http://testaspnet.vulnweb.com
Payload,
Form Details
method,post
inputs,"[{'type': 'hidden', 'name': '__EVENTTARGET', 'value': ''}, {'type': 'hidden', 'name': '__EVENTARGUMENT', 'value': ''}, {'type': 'hidden', 'name': '__VIEWSTATE', 'value': '/[illegible]'}, {'type': 'hidden', 'name': '__VIEWSTATEGENERATOR', 'value': 'CA0B0334'}, {'type': 'hidden', 'name': '__EVENTVALIDATION', 'value': '/wEWAwLpus/[illegible]/IvAA=='}]"
action,default.aspx

CSS Vulnerability Detected
```

Fig. 5 –Cross Site Scripting Detection Documentation

7. CONCLUSION

This project effectively demonstrates the design and implementation of a web vulnerability scanner designed to detect critical security vulnerabilities, such as SQL Injection (SQLI) and Cross-Site Scripting (XSS), within web applications. By integrating modular Python scripts and optimized scanning methodologies, the tool provides a robust solution for identifying security weaknesses that could jeopardize user data and the integrity of web applications. The use of an automated approach enables website administrators and developers to proactively protect their digital assets from prevalent cyber threats.

In terms of result documentation, the automated scanner generates detailed, standardized reports on the vulnerabilities identified during the scan. These reports streamline the analysis and communication of findings but may lack the contextual insights that manual review could offer.Moreover, the scanner reduces the technical expertise required for conducting web vulnerability assessments. Its user-friendly interface and comprehensive documentation simplify the scanning process, making it accessible to individuals without specialized knowledge in cybersecurity. In contrast, manual scanning typically demands advanced technical skills, a deep understanding of web application security, and significant practical experience, limiting its accessibility to a narrower pool of professionals. This project highlights the critical role of automated vulnerability detection in ensuring web application security, providing an effective tool for identifying and mitigating SQLI and XSS vulnerabilities in real-time.

REFERENCES

[1] Zukran, B., & Siraj, M. M. (2021, October). Performance comparison on sql injection and xss detection using open source vulnerability scanners. In *2021 International Conference on Data Science and Its Applications (ICoDSA)* (pp. 61-65). IEEE.

[2] Fonseca, J., Vieira, M., & Madeira, H. (2007, December). Testing and comparing web vulnerability scanning tools for SQL injection and XSS attacks. In *13th Pacific Rim international symposium on dependable computing (PRDC 2007)* (pp. 365-372). IEEE.

[3]Jacob, I., &Pirnau, M. (2020). SQL INJECTION ATTACKS AND VULNERABILITIES. *Journal of Information Systems & Operations Management*, 68-81.

[4] Buja, A. G., Low, N. N. M. A. A., Zolkeplay, A. F., Azam, N. A., & Isa, F. M. (2024). Analysis of Web Vulnerability Using Open-Source Scanners on Different Types of Small Entrepreneur Web Applications in Malaysia. *Journal of Advanced Research in Applied Sciences and Engineering Technology*, *40*(1), 174-188.

[5] Makino, Y., &Klyuev, V. (2015, September). Evaluation of web vulnerability scanners. In *2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)* (Vol. 1, pp. 399-402). IEEE.

[6] Sharma, C., & Jain, S. C. (2014, August). Analysis and classification of SQL injection vulnerabilities and attacks on web applications. In *2014 International Conference on Advances in Engineering & Technology Research (ICAETR-2014)* (pp. 1-6). IEEE.

[7]Antunes, N., & Vieira, M. (2009, September). Detecting SQL injection vulnerabilities in web services. In *2009 Fourth Latin-American Symposium on Dependable Computing* (pp. 17-24). IEEE.