Web Vulnerability Scanner Project Report

Maya Rozenberg , ID:313381600  
Tal Vazana , ID: 313454472

# Introduction

The Web Vulnerability Scanner is a cybersecurity tool designed to identify and analyze common vulnerabilities in web applications, such as SQL Injection and Cross-Site Scripting (XSS). These vulnerabilities pose significant risks to web security, as they can be exploited by attackers to gain unauthorized access, manipulate databases, or execute malicious scripts. This project aims to address these vulnerabilities through a simple yet effective scanner that can be used for educational purposes and as a foundation for more advanced security tools.

# Background

In the rapidly evolving field of cybersecurity, web application vulnerabilities remain a major concern. SQL Injection and XSS attacks are among the most common and dangerous threats, as they can lead to data breaches, loss of user trust, and financial damage. Existing solutions in the field range from commercial web application firewalls to open-source scanners. However, many of these tools are either too complex for beginners or too limited in scope. This project aims to fill this gap by providing a basic yet functional scanner that is easy to understand and modify.

# Project Design

The project is designed to scan web applications for SQL Injection and XSS vulnerabilities. The primary objective is to create a tool that can identify these vulnerabilities by sending crafted requests to a target URL and analyzing the responses. The methodology involves parsing the target URL, injecting payloads that mimic potential attacks, and checking for indicators of vulnerability in the response. The project is structured to be modular, with separate components handling different types of vulnerabilities.

# Implementation

The implementation process involves several key steps:  
1. Setup and Initialization: A new Python project is initialized, and necessary libraries like `requests` and `beautifulsoup4` are installed.  
2. Input Handling: The scanner accepts a URL as input, ensuring that it is correctly formatted and adding the HTTP schema if missing.  
3. Vulnerability Detection Modules: Separate modules are created for detecting SQL Injection and XSS vulnerabilities. These modules send crafted requests to the target URL and analyze the responses.  
4. Scanning Logic: The core scanning engine orchestrates the process, running each detection module against the provided URL and collecting the results.  
5. Report Generation: A detailed report of the findings is generated, summarizing the identified vulnerabilities.  
6. User Interface: A simple command-line interface (CLI) is provided for users to interact with the scanner.  
7. Security Measures: The scanner includes basic input validation and rate limiting to prevent overloading the target server.  
Challenges faced during implementation included handling different URL formats and ensuring the scanner was robust enough to handle various response types from target servers.

# Results and Analysis

The scanner was tested on several web applications to evaluate its effectiveness. The results showed that the tool was able to successfully detect basic SQL Injection and XSS vulnerabilities. However, the scanner's ability to identify more sophisticated or obfuscated vulnerabilities was limited, reflecting its design as an educational tool rather than a comprehensive security solution. The performance metrics indicated that the tool operated efficiently with minimal resource consumption, making it suitable for small-scale scanning tasks.

# Improvement Suggestions

To enhance the scanner's capabilities, several improvements are suggested:  
1. Advanced Detection Techniques: Implementing more sophisticated detection algorithms to identify a wider range of vulnerabilities.  
2. User Interface Enhancements: Developing a graphical user interface (GUI) to improve usability.  
3. Extensibility: Adding support for additional vulnerability types, such as CSRF and file inclusion attacks.  
4. Integration with Other Tools: Enabling integration with existing cybersecurity tools and platforms for a more comprehensive security analysis.  
These improvements would help address the current limitations of the project and expand its applicability in the field of cybersecurity.

# Conclusion

The Web Vulnerability Scanner project demonstrates the fundamental principles of web security assessment. While the tool is basic, it serves as an educational resource and a foundation for more advanced developments. The proposed improvements have the potential to significantly enhance the scanner's effectiveness and usability, contributing to better security practices in web development.

# Critical Evaluation

Students are encouraged to critically engage with the content of this project, examining the methodologies used, the effectiveness of the implementation, and the broader implications of the work. Evaluating the significance of the project's contribution to cybersecurity is essential, particularly in understanding the limitations and potential for future expansion. Critical thinking should be applied to assess the balance between the simplicity of the tool and its practical applicability in real-world scenarios.

# Clarity and Organization

The report is structured to provide a logical flow of ideas, ensuring that each section transitions smoothly to the next. Clear and concise language is used throughout, avoiding unnecessary jargon or overly technical terms. This approach ensures that the report is accessible to readers with varying levels of technical expertise.

# References and Citations

Proper citations have been included for any sources used in the project. These citations follow a consistent and appropriate academic style, ensuring that all references are clearly documented.

# Declaration of Language Model Use

This report was partially assisted by a Language Model (LLM). Prompts used included queries about the project structure, code implementation, and writing of report sections. The role of the LLM was primarily to provide guidance and suggestions, with all final decisions and content creation being carried out by the project author.

# Deliverables

1. Source Code: The complete source code of the implemented project is included.  
2. Written Report: A detailed written report analyzing the selected topic and implementation is provided.  
3. Presentation: A presentation demonstrating the project and its results is included.