“Vulnerability Scanner”

A

Project Report

submitted in partial fulfillment of the

requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

by

Name	Roll No.
Devansh Puniani	500110797
Aum vipul	500111825
Deepanshu Chowdhury	500107436

under the guidance of

Dr. Raviteja Panduranga

School of Computer Science

University of Petroleum & Energy Studies

Bidholi, Via Prem Nagar, Dehradun, Uttarakhand

Nov – 2024

CANDIDATE'S DECLARATION

I/We hereby certify that the project work entitled **"Vulnerability Scanner"** in partial fulfilment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in Cyber Security & Digital Forensics and submitted to the Department of Systemics, School of Computer Science, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my/ our work carried out during a period from September, **2024** to November**, 2024** under the supervision of **Dr.Raviteja Panduranga,**
Assistant Professor, SOCS.

The matter presented in this project has not been submitted by me/ us for the award of any other degree of this or any other University.

(Devansh Puniani)
Roll No.500110797

(Aum Vipul)
Roll No.500111825

(Deepanshu Chowdhury)
Roll No.500107436

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: 21 November 2024

Dr Raviteja Panduranga
Project Guide

ACKNOWLEDGEMENT

We wish to express our deep gratitude to our guide **Name**, for all advice, encouragement and constant support he/she has given us throughout our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thanks to our respected **Name of HoD, Head Department of Dr. Ajay Kumar,** for his great support in doing our project in Vulnerability Scanner.

We are also grateful to Dean SoCS UPES for giving us the necessary facilities to carry out our project work successfully. We also thanks to our Course Coordinator, Dr. Ram Kumar for providing timely support and information during the completion of this project.

We would like to thank all our **friends** for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our **parents** who have shown us this world and for every support they have given us.

Name	**Devansh Puniani**	**Aum vipul**	**Deepanshu Chowdhury**
Roll No.	**500110797**	**500111825**	**500107436**

ABSTRACT

The Vulnerability Scanner Tool is a project aimed at identifying and assessing vulnerabilities in web applications. This tool focuses on detecting common vulnerabilities such as SQL Injection, Cross-Site Scripting (XSS), Open Ports, and Insecure Headers. The tool's goal is to offer an automated, easy-to-use solution that helps both cybersecurity professionals and beginners understand and address security flaws in websites. By using scanning algorithms, the tool will be able to analyze web environments and provide detailed reports on security risks, thereby helping users improve their website's security posture.

The tool is designed to:

- Provide customizable scanning options for different vulnerability types.
- Serve as an educational resource for students and professionals in cybersecurity.
- Enable small businesses and educational institutions to secure their web environments with an affordable and easy-to-use tool.

By integrating features such as real-time scanning, vulnerability prioritization, and automatic report generation, this tool aims to contribute to proactive security practices in both personal and organizational use.

TABLE OF CONTENTS

INTRODUCTION

1.1 Overview of Web Application Security

Web applications are a critical part of modern digital infrastructures, from e-commerce sites to social media platforms. Web application security focuses on protecting websites and web applications from attacks (like SQL injection, XSS, open ports, and vulnerability in headers) ensuring that they are safe from unauthorized access, data breaches, and malicious exploitation.

1.2 History of Web Vulnerabilities

Over the past few decades, web applications have become prime targets for cyberattacks. Common vulnerabilities include SQL injection, Cross-Site Scripting (XSS), and broken authentication mechanisms. The evolution of these threats has led to the development of tools to help developers identify and mitigate them, including vulnerability scanners like OWASP ZAP, Burp Suite, and others.

1.3 Importance of Vulnerability Scanning in Cybersecurity

Vulnerability scanning is an essential component of proactive cybersecurity measures. By identifying weaknesses before attackers can exploit them, these tools help secure applications from common vulnerabilities like SQL injections and cross-site scripting (XSS). Regular scanning minimizes risk, reduces attack surface, and ensures that any newly discovered vulnerabilities are quickly mitigated.

1.4 Importance of Vulnerability Scanner in Cybersecurity.

Vulnerability scanning is an essential component of proactive cybersecurity measures. By identifying weaknesses before attackers can exploit them, these tools help secure applications from common vulnerabilities like SQL injections and cross-site scripting (XSS). Regular scanning minimizes risk, reduces attack surface, and ensures that any newly discovered vulnerabilities are quickly mitigated.

1.5 Requirement Analysis

The vulnerability scanner tool must be able to detect a variety of vulnerabilities in web applications. The primary requirements include:

- Detection of SQL injection, XSS, open ports, insecure headers, etc.
- Ease of use, even for users with minimal technical knowledge.
- Customizable scanning options to fit different use cases.
- Automated report generation for vulnerabilities present in it.

1.6 Project Objectives

- To create a vulnerability scanning tool to detection of common vulnerabilities in web applications.
- To provide detailed reports of vulnerability present in it.
- To create an intuitive user interface for both professionals and beginners in cybersecurity.

1. System Analysis

2.1 Overview of Existing Vulnerability Scanners

There are several widely used vulnerability scanners available today, **including OWASP ZAP, Burp Suite, and Acunetix**. These scanners vary in terms of functionality, usability, and cost. This section will discuss the features and shortcomings of these tools, which will help identify the gaps that the proposed vulnerability scanner seeks to fill.

2.2 Motivation for the Project

Existing vulnerability scanners like Nmap and SQLmap can be complex and time-consuming for beginners or small businesses. The proposed tool simplifies the process by integrating these well-known tools, offering faster and more efficient results. It is designed to be user-friendly, even for those with limited cybersecurity knowledge. Additionally, the tool includes **educational resources**, such as videos, courses, and ***safety tips***, to help users build basic cybersecurity knowledge, stay protected from common cyber threats, and pursue a career in the field.

2.3 Comparative Analysis of Tools

This section evaluates popular tools like OWASP ZAP and Burp Suite, highlighting their strengths and limitations. While these tools are powerful, they often have steep learning curves and require technical expertise. The proposed tool addresses these challenges by offering a simpler interface, focusing on common vulnerabilities, and ensuring quick and efficient scans, making it accessible to beginners and small organizations.

2.4 Proposed Vulnerability Scanner

This tool will aim to provide an easy-to-use solution to help detect vulnerabilities in web applications. It will:

- Scan for vulnerabilities such as SQL Injection, XSS, Open Ports, Insecure Headers, etc.
- Offer detailed, actionable reports of risks.
- Allow users to customize scans based on specific needs.

1.5 System Modules Overview

1.5.1 Vulnerability Detection Technique

The core functionality of the tool will be its ability to detect various vulnerabilities. Each module will have a different detection algorithm:

- **SQL Injection**: Uses various techniques like error-based injection, blind injection, and time-based analysis.
- **XSS**: Detects both reflected and stored XSS by injecting malicious scripts and analyzing responses.
- **Open Ports**: Uses port scanning techniques to identify open ports and associated risks.

1.5.2 Scanning and Detection Algorithms

These tools are designed to scan websites for common vulnerabilities. By simply entering a URL, the tool checks for issues like SQL injections and XSS vulnerabilities using predefined

checks/payloads. The process is simple and requires no complex setup, giving users a clear report on any detected vulnerabilities.

1.5.3 Reporting and Output Generation

After a scan is completed, the tool will generate a detailed report for the user. This report will:

- List the vulnerabilities found.

2. DESIGN

3.1 Architecture of the Vulnerability Scanner

The architecture of the vulnerability scanner consists of:

- **Frontend (UI)**: A simple and intuitive user interface that allows users to start a scan, view reports, and configure settings.
- **Backend**: The backend processes the scanning, running algorithms to detect vulnerabilities and storing search history + result + details of users in a database.
- **Database**: The system will store scan results and user preferences, allowing users to track past scans and outcomes.

3.2 Database Design

A simple database schema will be used to store scan data, including vulnerability, test results, and timestamps. This will allow users to review past scans, track progress, and analyze trends in vulnerability detection over time.

3. IMPLEMENTATION

4.1 SQL Injection Test

SQL Injection tests will be implemented by injecting payloads into input fields and monitoring responses for SQL error messages or changes in the database state. The tool will check for common weaknesses like improper sanitization and unfiltered user inputs.

4.1.1 SQL Injection Detection Algorithms

The tool will simulate SQL injection attempts using various techniques, such as error-based, union-based, and blind SQL injection. It will analyze HTTP responses to identify any possible vulnerabilities.

4.1.2 Sample Test Cases and Results

Sample test cases will include:

- A login page with vulnerable SQL query handling.
- An input form where user data is not properly sanitized. The results will indicate the presence of SQL injection and its severity.

➢

4.2 XSS (Cross-Site Scripting) Test

XSS vulnerabilities are tested by injecting malicious JavaScript payloads into input fields and observing if the script is executed when the page loads. The tool checks for reflected and stored XSS vulnerabilities.

4.2.1 XSS Detection Algorithms

The tool uses payloads like <script>alert('XSS')</script> to simulate a Cross-Site Scripting attack. It evaluates whether these scripts are executed in the response.

4.2.2 Sample Test Cases and Results

Sample test cases:

- An input field that displays unsanitized user input.
- A comment section where users can submit text.

4.3 Open Port Scan

Open Port scanning checks a website or server for open ports that could be vulnerable to exploitation. This is done by scanning for common open ports such as 80 (HTTP), 443 (HTTPS), and others.

4.3.1 Open Port Scan Algorithms

The tool uses network scanning techniques, such as Nmap, to check for open ports. It reports any open ports and their potential security risks.

4.3.2 Sample Test Cases and Results

Sample test cases:

- A website or server with an open port exposed.
- A server with restricted ports that shouldn’t be open.

4.4 Security Header Scan

The security headers scan evaluates whether a website is using important security headers like X-Content-Type-Options, Strict-Transport-Security, and Content-Security-Policy.

4.4.1 Security Header Detection Algorithms

The tool checks the presence and correctness of these headers in the HTTP response. If any required security headers are missing or misconfigured, they are flagged as vulnerabilities.

4.4.2 Sample Test Cases and Results

Sample test cases:

- A website missing security headers.
- A website with improperly configured security headers.

4. RESULTS

5.1 Sample Reports

Sample reports will be generated after scanning a test website. These reports will include:
- Detected vulnerabilities details.
- A summary of vulnerabilities (e.g., SQL Injection detected, payload use, response, etc).
- Recommendations for fixing the issues.

5.2 Test Cases and Outcomes

Various web applications will be tested to demonstrate the scanner's accuracy. Test cases will cover all minor vulnerability types, ensuring the tool provides comprehensive detection.

5. CONCLUSION AND FUTURE WORK

6.1 Summary of the Tool's Capabilities and Features

This section will summarize the features of the vulnerability scanner, including its ability to detect multiple types of vulnerabilities, generate detailed reports, and response of these payloads.

6.2 Future Enhancements and Upgrades

Future updates may include:
- Support for more complex vulnerabilities (e.g., server misconfigurations, broken authentication).
- Integration with CI tools for automated scans.
- A more advanced UI with real-time monitoring and additional features.

6.3 Potential Challenges in Scaling and Deployment

Challenges in scaling the tool may include:
- Handling higher traffic volumes.
- Managing large datasets.
- Ensuring accuracy with an increasing number of scans.
- Adding multiple processing for parallel task execution and using multiple tools for better vulnerability detection.

Appendix A: User Manual for Vulnerability Scanner

1. Accessing the Tool

1. **Visit the Website:**
    - Open your preferred web browser and navigate to the official website of the Vulnerability Scanner Tool.

- Example: http://127.0.0.1:5000

2. **Home Page:**

Once the website is loaded, you will be greeted with a simple user interface.

The home page will feature:

- An input field where you can enter the URL of the website you wish to scan.
- A **Start Scan** button to initiate the scanning process.
- **Login** and **Sign Up** options for users to log into their accounts or create a new one.

2. User Authentication

1. **Login Option:**

- If you already have an account, click on the **Login** button located on the top right of the homepage.
- Enter your **Username** and **Password**, then click **Submit** to access your personalized dashboard where you can track your scan history and view detailed results.

2. Sign Up Option:

- If you do not have an account, click the **Sign Up** button to create a new account.
- You will need to provide a **Username**, **Email Address(not know for future)**, and **Password**. After entering the required information, click **Submit** to create your account.
- Once your account is created, you can log in using your new credentials.

3. Forgot Password (future):

If you've forgotten your password, click on the **Forgot Password** link on the login page and follow the instructions to reset it.

3. How to Use the Vulnerability Scanner

1. **Enter the Target URL:**

- After logging in, you will be directed to the dashboard.

- In the input field provided, enter the URL of the website you want to scan for vulnerabilities.
- Example: https://example.com
- Ensure the URL is properly formatted (i.e., it begins with http:// or https://).

2. **Choose the Type of Scan:**

- After entering the URL, select the type of scan you want to perform from the available options:
    - **SQL Injection**
    - **Cross-Site Scripting (XSS)**
    - **Open Ports**
    - **Security Headers**

3. **Start the Scan:**

- After entering the URL and selecting the scan type, click on the **Start Scan** button.
- The scanner will begin testing the website for the selected vulnerabilities.
- A loading indicator will appear while the scan is in progress. Please wait until the scan completes.

4. **View Scan Results:**
    - Once the scan is complete, the results will be displayed on the screen.
        - If vulnerabilities are detected, you will see detailed information about each vulnerability
        - If no vulnerabilities are found, a message will indicate that the website is secure from low attack .
        - Detailed information, such as payloads used during the scan will be displayed.
        -

5. ## Scanning for XSS Injection

1. **Enter the Website URL:**
    - In the input field, enter the website URL you want to scan, such as https://example.com
2. **Select XSS Injection Scan:**

References

- OWASP Top 10, OWASP Foundation, 2023
- Cryptography and Network Security: Principles and Practice, William Stallings, 2017
- OWASP Top 10 Documentation, Version 2023, https://owasp.org/www-project-top-ten/