Internship Project Report

Web Application Vulnerability Scanner

Name: TUSHAR KANTI MAJUMDER
Internship Position: Cybersecurity Intern
Organization: Elevate Labs
Duration: [August 2025- PRESENT]
Date of Submission: August 19, 2025

Introduction

As part of my internship at **Elevate Labs**, I undertook the development of a lightweight **Web Application Vulnerability Scanner**. The goal was to identify common vulnerabilities found in real-world web applications using ethical methods. This project simulates basic attacks to test for security flaws like **Cross-Site Scripting (XSS)**, **SQL Injection (SQLi)**, and **Cross-Site Request Forgery (CSRF)**. The tool is designed with educational and internal testing use in mind, providing foundational insights into secure web development and penetration testing.

Abstract

The vulnerability scanner is a Python-based application that provides a basic yet effective mechanism to analyze input fields in a website, inject test payloads, and detect potential vulnerabilities based on responses. It uses libraries such as requests and BeautifulSoup for crawling and form handling, and Flask to offer a user-friendly web interface.

Once a URL is submitted through the interface, the scanner performs automated tests and displays the findings. It also logs each scan with timestamps, payloads used, and severity of any discovered vulnerabilities. This tool simulates the behavior of common web vulnerability scanners like OWASP ZAP or Burp Suite, but in a simplified and transparent format for learning and internal assessments.

Tools and Technologies Used

Tool / Library	Purpose
Python 3	Core scripting language
Flask	Web application backend

Tool / Library	Purpose
Requests	Sending HTTP GET/POST requests
BeautifulSoup4	Parsing HTML and extracting forms
Validators	Validating input URLs
HTML (index.html)	Frontend for interacting with scanner
Kali Linux	Development and testing environment

Steps Involved in Building the Project

1. **Setup Environment**
   - Initialized a virtual Python environment on Kali Linux
   - Installed required libraries (flask, requests, beautifulsoup4, validators)
   - Created a clean project structure with folders for logs and templates
2. **Scanner Logic**
   - Developed custom functions to detect:
     - **XSS** via <script> injection and response checking
     - **SQL Injection** by injecting ' OR '1'='1 and matching SQL error patterns
     - **CSRF** by checking the presence of CSRF tokens in forms
   - Implemented form submission automation and response analysis
3. **Web Interface (Flask + HTML)**
   - Created a single-page web app (index.html) to input URLs and display results
   - Used Flask to link backend logic with the frontend
   - Kept the UI clean and simple without advanced styling or color coding
4. **Result Logging**
   - Stored all scan results in logs/scan_results.txt
   - Each log includes date, time, target URL, and detailed results with severity levels
   - Also provided an option to download logs from the UI

Conclusion

This project served as a hands-on introduction to core web security principles and vulnerability assessment methodologies. It enhanced my understanding of how real-world scanners work and how attackers target insecure forms and input handling.

By simulating attack payloads and observing system responses, I learned the importance of secure input validation, token usage (CSRF protection), and output encoding. Though simplified, this scanner lays the groundwork for more advanced security tooling and shows how vulnerabilities can be detected programmatically.

The experience also strengthened my proficiency in Python, Flask, HTML, and common web exploitation techniques—skills that are directly relevant to my cybersecurity journey at **Elevate Labs**.