Security GUI

Chinmay Desai[1], Prashant Hage[2], Prajwal Vannewar[3], Prof. Swati Shinde[4]

Department of Electronics
K. J. Somaiya Mumbai, Maharashtra, India
chinmay.sd@somaiya.edu[1], prashant.hage@somaiya.edu[2]
prajwal.vannewar@somaiya.edu[3], swati.shinde@somaiya.edu[4]

Abstract

Vulnerability scanning is critical for modern cybersecurity, helping protect systems and networks from evolving threats. This paper presents an integrated vulnerability scanning platform that combines three tools: a system-level scanner, a web application scanner, and a network-level scanner, all within a unified web interface. The platform's modular architecture allows each tool to operate independently while sharing data like vulnerability signatures, scan results, and remediation recommendations. This approach streamlines security assessments, provides a comprehensive view of potential weaknesses, and improves scan efficiency. The paper discusses the system's de- sign, integration, and operational details, focusing on microservices, containerization, and automated database updates. It also evaluates the platform's performance, accuracy, and potential improvements, offering a more effective way for security teams to mitigate cyberattack risks.

Keywords: Cybersecurity, Vulnerability Scanning, Web Application Security, System Security, Network Security, Vulnerability Databases, Microservices, Containerization

I. INTRODUCTION

With the rapid expansion of digital infrastructures and the reliance on internet-connected systems, vulnerabilities have become a critical concern for organizations worldwide. Cyberattacks continue to grow in frequency, sophistication, and impact, leading to massive financial and reputational damages. The process of discovering, prioritizing, and re- mediating vulnerabilities is therefore at the heart of modern cybersecurity. While various specialized vulnerability scanners exist, organizations often struggle to manage multiple tools that operate in silos, each focusing on a specific domain such as web applications, system software, or network services. The lack of a unified view and integrated management system can introduce blind spots and inefficiencies, leaving networks exposed to potential threats.

Historically, security analysts would employ separate scanners for different tasks—one for network devices, another for web applications, and yet another for operating systems or installed software. This fragmented approach can lead to overlap, missed vulnerabilities, inconsistent reporting, and redundant efforts in remediation. Moreover, maintaining up- to-date vulnerability databases, ensuring regular scans, and generating coherent reports across disparate tools demands significant manual effort. Such complexity underscores the need for a consolidated platform that can orchestrate various scanning engines and provide a single pane of glass for cybersecurity insights.

The primary challenge in modern vulnerability management lies in the disjointed nature of existing scanning solutions. Even though each tool may be highly specialized and effective in its domain, the manual process of collating scan results, correlating data, and deriving actionable intelligence can be cumbersome. A lack of standardization in reporting formats, vulnerability scoring, and scan scheduling further complicates matters.

In addition, security teams are often constrained by limited human and financial resources, making it critical to streamline the vulnerability scanning process and focus on addressing the most critical issues first.

Moreover, the complexity of multi-cloud and hybrid environments has introduced additional layers of risk. As organizations adopt cloud services, microservices architectures, and containerization, the attack surface grows. The old paradigm of perimeter security is insufficient in this modern landscape. Continuous monitoring and scanning are essential to maintain a robust security posture, but a single platform that unifies and automates these tasks is still lacking in many enterprises.

In response to these challenges, this paper proposes an integrated vulnerability scanning platform accessible through a single website or dashboard. The overarching objectives include:

Unified Interface: Provide a centralized dashboard that hosts three distinct scanning tools:

System-Level Vulnerability Scanner: Focused on operating systems, installed software, and local configurations. Web Application Vulnerability Scanner: Specialized in identifying weaknesses in web applications, such as SQL injection, cross- site scripting (XSS), and insecure configurations. Network- Level Scanner: Capable of identifying open ports, misconfigurations, and known network service vulnerabilities.

Data Sharing and Correlation: Facilitate the sharing of vulnerability data, signatures, and results among the scanners to provide a comprehensive view of potential risks and to minimize false positives or false negatives.

Automation and Scheduling: Implement an automated scheduling mechanism for scans, allowing administrators to configure periodic scans for different layers (system, web, network) from a single interface.

Reporting and Remediation: Generate consolidated reports that highlight critical vulnerabilities, suggest remediation steps, and track resolution progress.

Scalability and Maintainability: Employ modern software development paradigms (microservices, containerization) to ensure the platform can scale horizontally and remain easy to maintain and update.

By addressing these objectives, we aim to reduce complexity, lower the total cost of ownership, and improve the overall security posture for organizations of varying sizes and sectors.

II. RELATED WORK

Past work in web security scanning has emphasized various strategies: 1] Signature-Based Scanning – Identifying vulnerabilities based on known attack patterns. 2] Behavioral Analysis – Employing heuristics for identifying suspicious behavior. Our work is distinct from existing research as it combines a friendly interface with real-time security scanning. Contrary to scanners based on AI, our rule-based system yields improved results with minimal computational costs. By consolidating multiple security tools under a single platform and including a chatbot for vulnerability searches, our project provides an open-source, distinctive alternative to proprietary security scanners.

III. LITERATURE SURVEY

Web security has been an important research area because of the rise in cyber threats, data breaches, and security vulnerabilities in contemporary web applications. Several methods have been developed throughout the years to detect, analyze, and respond to security vulnerabilities. Web security research has mainly involved automated vulnerability scanning, penetration testing techniques, and the implementation of artificial intelligence for security analysis.

Numerous researches have investigated various web vulnerability detection approaches, varying from static code analysis to dynamic analysis. Static code analysis tools examine the source code of the application for detecting possible vulnerabilities prior to deployment. Such tools are applicable to developers at the initial development phase but fail to detect runtime vulnerabilities, which need dynamic analysis. Conversely, dynamic analysis tools like OWASP ZAP and Burp Suite conduct runtime security testing by engaging with

the application in real-time, revealing vulnerabilities that would not be apparent in the source code.

One of the most important areas of research has been the creation of web vulnerability scanners. Several open-source tools like Nikto, Skipfish, and W3AF have been created to automatically scan web applications for vulnerabilities. These tools assist in identifying common security vulnerabilities, such as SQL injection, cross-site scripting (XSS), and insecure server configurations. . Although these tools are useful, most of them need technical skills to run and interpret the results, which makes them less convenient for novices or non-security experts.

An additional pivotal aspect of web security research is security misconfiguration study. Most web applications have improper security configurations that result in weaknesses like exposed server banners, lack of security headers, and insecure authentication mechanisms.

In addition, there has been recent research on the application of artificial intelligence to vulnerability detection. Machine learning algorithms based on massive collections of security vulnerabilities have been employed to identify potential security weaknesses in web applications. Although AI-powered scanners have been promising, they are based on heavy training data and computational capabilities.

By leveraging the research that has been conducted on web security, vulnerability scanning, and integrating tools for security, our project is a real-world, open-source implementation of automatic security auditing. Adding a chatbot component that does CVE lookups increases the usability because users do not need to manually search for vulnerabilities.

IV. METHODOLOGY

Our research approach adopts a systematic approach to guarantee the effectiveness and usability of our web security scanner. We separated our approach into four significant phases: Requirement Analysis, Development, Integration, and Testing.

1) Stage 1: Requirement Analysis

Prior to crafting the system, we carried out thorough research with the aim of determining the main features required for a contemporary web security test tool. We carried out an investigation of available web vulnerability scanners and discovered that most tools are numerous but fall within two categories:

a) Commercial tools (e.g., Burp Suite, Acunetix) – These deliver full security reviews but come with a cost-requiring paid license.
b) Open-source tools (e.g., OWASP ZAP, Nikto) – They are available for free but can be tricky for novices to use.

From these insights, we set our system's main goals:

i. Offer a user-friendly web vulnerability scanner.
ii. Bring together current security tools into one platform.
iii. Make it possible for users to receive real-time vulnerability information through a chatbot.
iv. Keep the system open-source and accessible to all.

2) Stage 2: Development

We developed the system using Python's Flask framework, which allows lightweight, scalable web applications. The front end was built using HTML, CSS, and JavaScript, ensuring a responsive design.

Key features developed during this phase include:

i. Web Vulnerability Scanner – A scanner that checks for common vulnerabilities such as SQL injection, XSS, security header misconfigurations, and exposed server information.
ii. Chatbot Integration – A chatbot that retrieves CVE information from public APIs and displays vulnerability information to users.

iii. Security Tools Interface – A web-based UI through which users can execute tools like Nmap, Gobuster, and ExifTool.

3) **Stage 3: Integration**

The development phase was then followed by the integration of various security tools. We provided smooth execution of external security tools via shell scripting and Flask's subprocess module. The chatbot was integrated with RESTful API calls to dynamically fetch CVE data.

4) **Stage 4: Testing and Validation**

After the development, we carried out extensive testing using live websites to verify the effectiveness of our scanner. We tested the following scenarios:

i. Identifying missing security headers.
ii. Finding SQL injection vulnerabilities.
iii. Verifying insecure external links.

5) **Stage 4: System Workflow**

Figure 1 illustrates the integrated workflow of our vulnerability scanning platform. The process begins with user authentication via a login page. Upon successful login, the user accesses the unified dashboard, offering two primary paths: security tools selection or direct web vulnerability scanning. The selected tool executes, returning real-time scan results, including system-level, network-level, and web application vulnerabilities. Consolidated reports are generated, mapping vulnerabilities to remediation recommendations. Users can then log out or return to the dashboard for further scans. This flowchart visually simplifies the scanning process, streamlines operations, and enhances user interaction with complex cybersecurity tools, ensuring robust security efficiency.

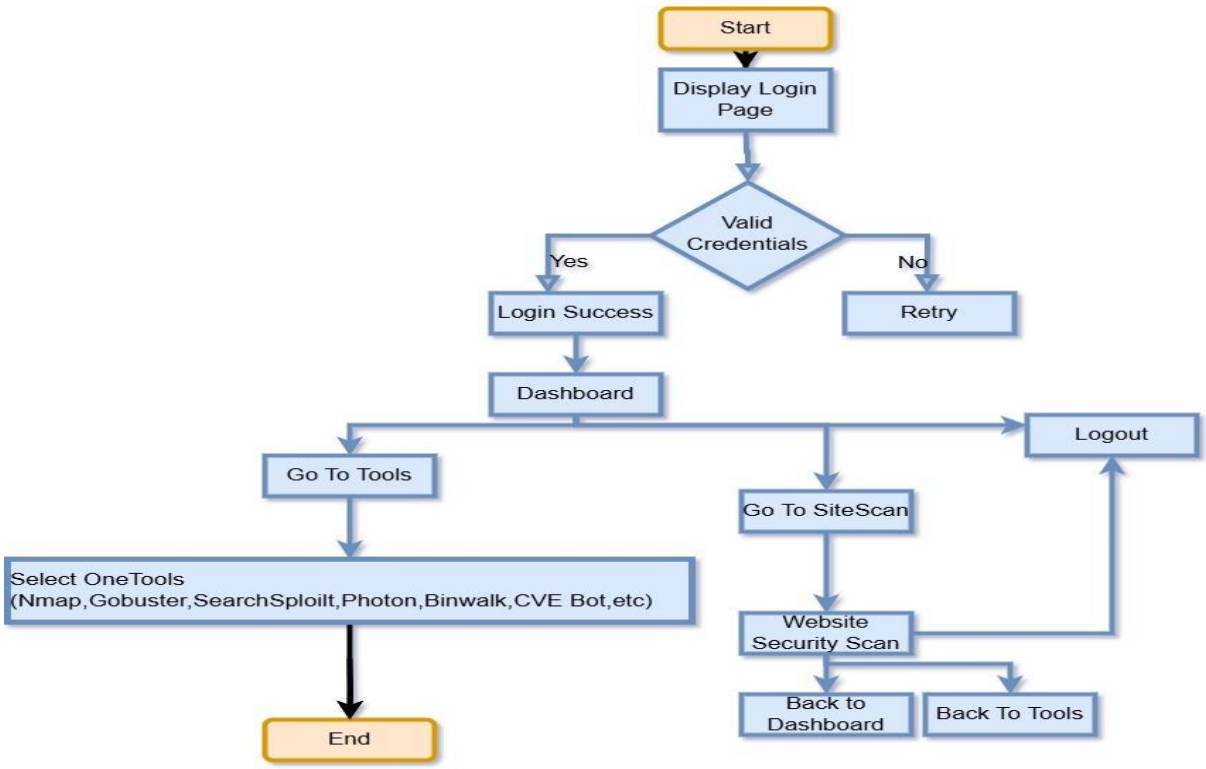

Fig 1: Flowchart

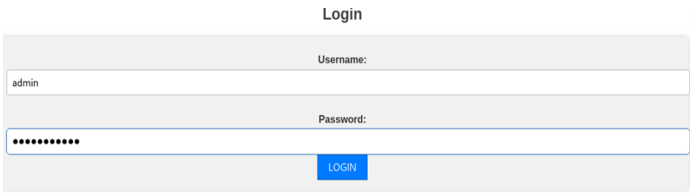

Fig 2: Login page.

Fig 3: Welcome page.

V. REQUIREMENTS

Functional Requirements

i. The system must enable users to enter a website URL and trigger a security scan. It must present a comprehensive report dynamically, mapping the severity to the identified vulnerabilities.
ii. The UI must be able to support various security tools, such as Nmap, Gobuster, ExifTool, Searchsploit, Binwalk, Foremost, theHarvester, Photon.
iii. The chatbot must retrieve CVE data dynamically.

Non-Functional Requirements

i. Scalability – The application must be able to process multiple scans effectively.
ii. Security – The scanner must have access control mechanisms to avoid unauthorized use.
iii. Performance – The scanner must provide fast results without utilizing too many resources.

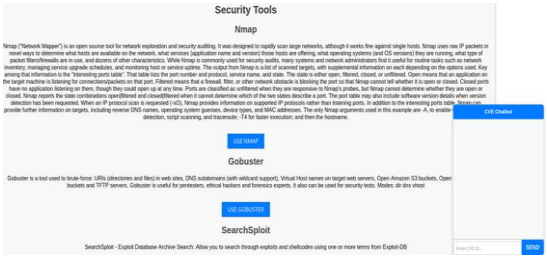

Fig 4: Security Tools.

1. Nmap (Network Mapper): Nmap is a robust open- source network scanner utilized for security auditing and reconnaissance. It assists in the detection of live hosts, open ports, services on a system, and vulnerabilities. Nmap is able to identify operating systems, firewall settings, and network topology and is therefore an important tool for penetration testers. In our project, Nmap is incorporated so that users are able to execute network scans with options that can be customized and generate detailed network security reports.

2. Gobuster: Gobuster is a brute-force directory enumeration tool that helps discover hidden directories, files, DNS subdomains, and virtual hosts. It is popularly applied in web penetration testing to find misconfigurations and exposed directories.

Gobuster can perform dictionary-based attacks with predefined or custom wordlists. Gobuster is incorporated in our project to assist users in directory and subdomain enumeration, enabling them to discover possible attack vectors on web applications.

Fig 5: Gobuster.

3. SearchSploit: SearchSploit is a command-line tool that can be used to search the Exploit Database (Exploit-DB) for an application's, software's, or service's vulnerabilities. It gives immediate access to public exploits without internet connectivity. SearchSploit incorporated in our project helps users locate possible exploits of vulnerable services and aids penetration testers in performing vulnerability scans effectively.
4. Binwalk: Binwalk is an analysis tool to extract embedded files and examine binary firmware images for forensic analysis. It is majorly employed to reverse engineer firmware to detect covert malicious code, backdoors, or vulnerabilities. Our project utilizes Binwalk in order to permit users to scan binary files and extract significant metadata to assist forensic investigators and security researchers.
5. ExifTool: ExifTool is an image, document, and multimedia extraction tool used in extracting metadata from images, documents, and multimedia files. In digital forensics, the tool is employed to parse out metadata such as timestamps, geographic location, and camera information that can play significant roles in computer investigations. Within our project, ExifTool offers a way for users to parse out metadata effectively.
7. Foremost: Foremost is a recovery tool for files that utilizes headers, footers, and data structures to recover lost or deleted files from storage devices. It is regularly utilized in forensic analysis to retrieve evidence from disk images, hard drives, and USBs. With the integration of Foremost, our project offers a mechanism for recovering deleted or concealed files, helping forensic specialists recover valuable information.
8. theHarvester: theHarvester is an OSINT tool utilized for extracting information regarding a target domain, such as email addresses, subdomains, employee names, and public records. It searches search engines, APIs, and data sources to extract publicly available information. In our project, theHarvester is incorporated to enable security professionals to perform reconnaissance on potential targets.
9. Photon: Photon is a web spider that can be used to scrape sensitive data like URLs, subdomains, JavaScript files, and API keys from a target site. It is commonly employed in OSINT research and bug bounty hunting. In our project, Photon enables users to carry out deep web crawling to reveal concealed links and possible security threats.

Fig 6: Photon.

10. Website Security Scanner (In-House Developed): Our project has an in-house Website Security Scanner that scans web applications for security without using proprietary software from Kali Linux. The scanner scans for weaknesses like SQL Injection, Cross-Site Scripting (XSS), missing security headers, open directories, insecure cookies, and external links via HTTP rather than HTTPS. Contrary to other scanners, it does not rely on such tools as Nikto or WPScan but processes web pages directly with a mix of active security checks and pattern matching. It returns in a standard report, prioritizing vulnerabilities by severity and marking secure security checks. This tool improves web security through insights on possible vulnerabilities, hence becomes a good addition for penetration testers and security researchers.

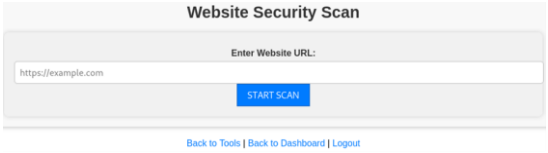

Fig 7: Website Security Scanner.

VI. RESULT

Fig 8: Report

VII. CONCLUSION

This project creates a comprehensive cybersecurity web application integrating various security and forensic tools into one easy-to-use platform. It combines tools like Nmap, Gobuster, SearchSploit, Binwalk, ExifTool, and others commonly used in cybersecurity audits. A key innovation is a custom Python-based website vulnerability scanner, offering real-time security analysis. Additionally, a chatbot enables dynamic querying of CVE vulnerabilities, providing instant threat intelligence. The platform emphasizes user experience with a uniform dark-themed UI, clear presentation of results, and intuitive navigation. By

removing the need for command- line proficiency, the platform is accessible to a broader au- dience, including students, researchers, and IT professionals with minimal cybersecurity knowledge. Future developments include cloud deployment, machine learning for automated threat detection, and real-time monitoring. More OSINT and digital forensics tools could further enhance the platform. This project bridges the gap between powerful cybersecurity tools and user-friendliness, optimizing vulnerability assessments for security experts.

REFERENCES

[1] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 DARPA off-line intrusion detection evaluation," Computer Networks, vol. 34, no. 4, pp. 579–595, 2000.

[2] A. Moen, A. Maehre, and J. M. Bjørndalen, "A survey of system-level vulnerability scanners," International Journal of Information Security, vol. 17, no. 2, pp. 187–206, 2018.

[3] S. Christey and B. Martin, "Vulnerability type distributions in CVE," Mitre Corporation, May 2013. [Online]. Available: https://cve.mitre.org/