AUTOMATED NETWORK ENUMERATION AND VULNERABILITY ANALYSIS TOOL

by

Sarikha S (2212012)
Sri Gomathi R (2212013)
Aruna Varshini S (2212019)

19CS67C PRODUCT DEVELOPMENT LABORATORY

Submitted to the faculty of

COMPUTER SCIENCE AND ENGINEERING

In partial fulfillment of the requirements for the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

ANNA UNIVERSITY, CHENNAI

NATIONAL ENGINEERING COLLEGE, KOVILPATTI

(An Autonomous Institution)

APRIL 2025

CERTIFICATE

Certified that this project report titled " **- AUTOMATED NETWORK ENUMERATION AND VULNERABILITY ANALYSIS TOOL**" is the bonafide work of **Sarikha S (2212012) ,Sri Gomathi R (2212013) , Aruna Varshini S (2212019)** who carried out the project under my supervision for the partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering in **COMPUTER SCIENCE AND ENGINEERING**. Certified further, that to the best of my knowledge the work reported here does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate

Place K.R.Nagar, Kovilpatti
Date

Supervisor
Dr.J.NASKATH., M.E., Ph.D.,
Associate Professor,
Computer Science and Engineering,
National Engineering College,
K.R.Nagar, Kovilpatti – 628 503

Head of the Department
Dr.V.Gomathi
Professor & Head,
Computer Science and Engineering,
National Engineering College,
K.R.Nagar, Kovilpatti – 628 503

Submitted to 19CS67C - Product Development Laboratory viva voce examination held on

____

Internal Examiner

Co / External Examiner

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	i
	ACKNOWLEGEMENT	ii
	LIST OF FIGURES	iii
	LIST OF ABBREVIATIONS	v
1	**INTRODUCTION**	1
	1.1 Background of the Existing Software/System	2
	1.2 Problem Statement	3
	1.3 Need for Innovation	4
	1.4 Objectives of the Project	5
	1.5 Expected Outcomes	6
2	**REVERSE ENGINEERING OF EXISTING SOFTWARE**	7
	2.1 Existing System Overview	8
	2.2 Source Code Analysis	10
	2.3 Functional Analysis	11
	2.4 Performance Analysis	12
	2.5 Security and Privacy Analysis	13
	2.6 Research Gaps	
3	**FORWARD ENGINEERING PROCESS**	14
	3.1 PROBLEM IDENTIFICATION AND USER ANALYZE	14
	3.1.1 Problem Understanding	14
	3.1.2 Identifying Users & Stakeholders	15
	3.1.3 Empathizing with Users	15
	3.1.4 Defining the Problem	16
	3.2 SOFTWARE ARCHITECTURE & DESIGN	17
	3.2.1 Software Requirements Specification (SRS)	17
	3.2.2 Architectural Design	17

3.2.3 Technology Stack Selection 18
3.2.4 Database Design 19
3.3 DEVELOPMENT & IMPLEMENTATION 22
3.3.1 API & Data Flow Design 22
3.3.2 Modules 24
3.3.3 Feature Implementation 28
3.3.4 UI/UX Considerations 28
3.3.5 Security & Privacy Enhancements 29
3.4 TESTING & VALIDATION 30
3.4.1 Unit Testing 30
3.4.2 Functional Testing 30
3.4.3 Performance Testing 31
3.4.4 User Testing & Feedback 32
3.4.5 Iterative Improvements 32
3.5 COMPARATIVE ANALYSIS & RESULTS 32
4 CONCLUSIONS AND FUTURE SCOPE 34
4.1 Conclusion 34
4.2 Future Enhancements 34
5 OUTCOME 36
6 REFERENCES 38

ABSTRACT

In today's rapidly changing world effective network is paramount to perform visibility and achieve safety. This work has been applied to a web-based network scanner and vulnerable detection tool using flask, NMAP and joint danger intelligence feeds. The equipment offers users to scan the defined IP address, scan live hosts, open ports and running services. It provides the ability to start both quick scans and deep scans, rapidly introducing outputs in a readable and structured manner. In addition to fundamental scanning it includes opening the vulnerability through openvas data, common weaknesses and the use of the exposure list, and the national vulnerability database (NVD) identified to correspond to correlations. A built-in interactive chatbot provides the facility of purpose by providing real-time, reference-incredible information such as service details and related hazards. The main features are real -time response, input verification, modular design and future Extensibility. Future -added features include scoring through asynchronous task handling, CVSS metrics and dynamic visualizations. On the side of students, system administrators, and cyber security professionals, provides a lean platform for active network inspection and evaluation of danger.

ACKNOWLEDGEMENT

First and foremost, we would like to thank God Almighty for showering his blessings throughout our life. He has been the tower of our strength in each step of our work. We take the privilege to express hearty thanks to our parents for their valuable support and effort to complete the project work.

We would like to express our deep sense of gratitude and respectful regards to our director **Dr. S. Shanmugavel B.Sc., D.M.I.T., Ph.D.,** for giving an opportunity to do this work.

We have great pleasure in acknowledging our Principal **Dr.K.Kalidasa Murugavel, M.E., Ph.D.,** for extending his full support to undergo this work.

We express our profound thanks to our beloved Head of the Department Dr**.V.Gomathi., M.Tech., Ph.D.,** for extending her full support and providing various facilities during the project work.

We would like to thank our project guide **Dr.J.NASKATH M.E., Ph.D.,** Associate Professor, Department of Computer Science and Engineering for her valuable guidance, technical support and suggestions helped us in doing the project work.

We express our gratitude to our project coordinator **Dr.S.Dheenathayalan,** Associate Professor, Department of Computer Science and Engineering for his valuable guidance at each and every stage of the project.

We extend our hearty thanks to our tutors and class in-charges for their valuable guidance. We are grateful to all the staff members and our dear friends for their valuable suggestions and co-operation for this work.

LIST OF FIGURES

FIGURE NO.	FIGURE DESCRIPTION	PAGE NO.
2.1	NMAPONLINE HOME PAGE	8
2.2	HTML/JAVASCRIPT ELEMENTS OF NMAPONLINE	9
2.3	FUNCTIONAL OVERVIEW OF EXISTING NMAP	10
2.4	PAGE LOAD TIME ANALYSIS	11
2.5	SITE DETAIL & SECURITY HEADER REPORT	12
3.1	EMPATHY MAP	15
3.2	STAKEHOLDER INTERACTION DIAGRAM	16
3.3	ARCHITECTURE DESIGN	18
3.4	TECHNOLOGY STACK DIAGRAM USED IN THE DEVELOPMENT OF APPLICATION	19
3.5	DATABASE ER DIAGRAM	19
3.6	FLOW DIAGRAM OF OPERATIONS	20
3.7	USERS TABLE ENTRY	20

3.8	SCAN_REPORTS TABLE ENTRY	21
3.9	VULNERABILITY_REPORT TABLE ENTRY	21
3.10	AI_CHAT_HISTORY TABLE ENTRY	22
3.11	DATA FLOW DIAGRAM	24
3.12	COMPLETE MODULE OF APPLICATIION	25
3.13	JWT AUTHENTICATION AND BCRYPT HASHING PROCESS	29
3.14	UNIT TESTING OUTPUT USING THUNDER CLIENT	30
3.15	FUNCTIONAL TESTING OUTPUT SHOWING CHATBOT STATUS	31
3.16	PERFORMANCE TESTING OUTPUT WITH RESPONSE TIME METRICS	31
3.17	COMPARATIVE ANALYSIS AND RESULTS	32
5.1	EMAIL CONFIRMATION SCREENSHOT FOR CONFERENCE PAPER SUBMISSION	37
5.2	PLAGIARISM REPORT FOR CONFERENCE PAPER	37

LIST OF ABBREVIATIONS

S.NO	ABBREVIATION	FULL FORM
1	NMAP	Network Mapper
2	IP	Internet Protocol
3	NVD	National Vulnerability Database
4	CVE	Common Vulnerabilities and Exposures
5	CVSS	Common Vulnerability Scoring System
6	CLI	Command-Line Interface
7	OS	Operating System
8	PDF	Portable Document Format
9	JSON	JavaScript Object Notation
10	OPENVAS	Open Vulnerability Assessment System
11	ICMP	Internet Control Message Protocol
12	TCP	Transmission Control Protocol
13	UI/UX	User Interface/User Experience

14	API	Application Programming Interface
15	CSV	Comma-Separated Values
16	HTTP	HyperText Transfer Protocol
17	AJAX	Asynchronous JavaScript and XML

CHAPTER 1

INTRODUCTION

As time progresses, the world is becoming increasingly connected, thanks to the internet and advancements in networking technologies. However, the open nature of the internet has brought network security into sharp focus. With organizations shifting many of their business functions to public networks, massive volumes of personal, commercial, and corporate data are now accessible through global network infrastructures. This growing exposure demands robust security measures to ensure that sensitive data remains protected and inaccessible to unauthorized individuals. Unauthorized network access—whether by external hackers or disgruntled internal users—can lead to the compromise, damage, or destruction of critical information. Such breaches not only threaten the organization's reputation but also impact its ability to remain competitive.

As a result, network security has become increasingly important, especially considering the ease with which intellectual property can be accessed online with minimal effort. To counter these risks, organizations employ a variety of security strategies—including scanning, vulnerability assessment, and penetration testing.Network scanning plays a crucial role in gathering real-time information about the current state of systems and networks. It helps in identifying active hosts and is often a starting point for security assessments. Vulnerability assessment, on the other hand, provides a structured evaluation of an information system's security posture[3].Together, these techniques form a comprehensive approach to auditing, penetration testing, reporting, and patch management—ensuring that an organization's network remains secure against ever-evolving threats.

1.1 BACKGROUND OF THE EXISTING SOFTWARE/SYSTEM

The current system used in most institutions and security forums, including NmapOnline available through platforms like https://hackertarget.com/nmap-online-port-scanner/, is a web-based version of the resourceful Nmap (Network Mapper) tool. This is a simplified web interface for conducting basic network reconnaissance functions such as host discovery and port scanning. It is mainly intended for rapid access to the features of Nmap without the need for local installation or knowledge of command-line protocols. Though the system allows users to discover open ports, services, and overall network exposure, it runs on a restrictive model. Users are allowed to start scans using predetermined sets of commands and restricted scan depth. Custom scans, ranges of IP, or use of the Nmap Scripting Engine (NSE) are usually disabled. Consequently, users cannot have any say over the complexity of the scan other than what is offered by the website. Additionally, output tends to be unprocessed and does not include contextual information, like levels of risk, specific service vulnerabilities, or mitigation recommendations.While this approach ensures ease of use and prevents misuse by limiting advanced features, it brings depth, flexibility, and actionable intelligence limitations[11].For instance, CVE/NVD mapping-based vulnerability assessments or exploit detections are missing or are tackled externally. Users who require detailed results are required to download raw scan output and process it independently using alternate tools or databases.

In addition, NmapOnline does not offer features such as interactive result analysis, threat level analysis, or chatbot support for handling questions. There is little feedback or educational assistance provided to non-experts analyzing scan results. Since there are no features such as authentication, project history, or secure reporting, professionals have to use other websites or desktop software for reliable reporting and ongoing monitoring.The limitations of existing browser-based Nmap solutions underscore the importance of a more converged, dynamic, and user-friendly solution. Platforms that converge scanning, vulnerability scanning, and user guidance into a single secure and user-friendly platform represent a strong way forward in making network auditing more achievable and useful.

1.2 PROBLEM STATEMENT

Although **Nmap** is recognized as one of the most powerful and essential tools for network scanning and security auditing, its effectiveness is often hindered by its complexity and reliance on a command-line interface. For users unfamiliar with command-line operations, the process of executing scans and interpreting results can be overwhelming. The steep learning curve associated with understanding various scanning parameters, flags, and output formats discourages many potential users—especially those without a technical background—from leveraging the full potential of Nmap. As a result, this limits the tool's accessibility and usefulness in broader contexts where cybersecurity awareness and practices are increasingly important. Another significant limitation is the lack of a **centralized and user-friendly web platform** that integrates Nmap's core functionalities. Currently, users must download and install the tool locally, operate it through the command line, and manually handle scan data. This fragmented approach creates inefficiencies and raises the risk of errors, especially in scenarios that require frequent, routine scanning.Furthermore, there are limited options for managing scan outputs in a structured and secure way. Users are often left without the means to store scan results, analyze trends over time, or produce formalized reports for audits or team reviews.

In today's digital landscape, there is a growing need for a **flexible, web-based solution** that bridges the gap between Nmap's technical power and the demand for intuitive usability. Such a solution should offer different access modes tailored to user expertise. For example, it could provide quick, no-login scans for general users who simply want to check the exposure of their IP or domain with minimal effort. At the same time, it should support advanced, authenticated scans for experienced users like IT professionals and network administrators[1].These users would benefit from full access to all of Nmap's scanning modes—such as OS detection, version scanning, script-based scans, and traceroutes—through a user-friendly interface. Moreover, this platform should include features that **automate and simplify scan management**, such as secure cloud storage for scan results, real-time data visualization through dashboards and charts, and downloadable reports in formats like PDF or JSON. These enhancements would allow users to maintain a history of scan activities, track vulnerabilities over time, and prepare audit-ready documentation without relying on external tools. Ultimately, such a web application would extend the utility of Nmap beyond just technical users, democratizing access to network scanning and empowering organizations and individuals alike to take a more proactive approach to cybersecurity.

1.2 NEED FOR INNOVATION

As cyber threats continue to evolve in complexity and frequency, traditional approaches to network vulnerability assessment are no longer sufficient. Most existing scanning tools, while powerful, are either too complex for non-technical users or lack the flexibility needed for modern, dynamic network environments. Tools like Nmap and OpenVAS provide deep scanning capabilities, but their command-line interfaces and fragmented outputs create barriers for efficient, routine usage—especially among small organizations, educational institutions, or individuals without advanced cybersecurity expertise. There is a pressing need for innovation in how network scanning and vulnerability assessment are conducted. Integrating the strengths of established tools into a unified, browser-based platform can dramatically improve accessibility, usability, and scalability.

A web-based system removes the need for local setup and simplifies the user experience through graphical interfaces, guided workflows, and automated reporting. By making vulnerability assessment visual, interactive, and easier to interpret, even those with minimal cybersecurity knowledge can identify threats and take corrective actions.Furthermore, automating the generation and storage of scan reports, visualizing data trends, and offering different levels of scan customization—ranging from basic scans to in-depth authenticated assessments—will redefine how security operations are managed. This innovation not only reduces administrative overhead but also promotes real-time, informed decision-making, empowering users to maintain secure networks proactively rather than reactively. In essence, this project addresses a critical gap: the need for a smart, accessible, and adaptive vulnerability assessment tool that can evolve alongside the threats it is designed to counter. The innovation lies not in reinventing scanning technologies, but in reimagining how they are delivered and experienced.

1.4 OBJECTIVES OF THE PROJECT

The core objective of this project is to design and develop a web-based network vulnerability scanning tool that integrates the robust capabilities of Nmap while offering an accessible, user-friendly interface for users of varying technical expertise. This tool is intended to simplify the process of performing real-time network scans and enhance the overall vulnerability assessment experience through automation, visualization, and ease of use.One of the primary goals is to create an intuitive front-end interface using HTML, CSS, and JavaScript, ensuring a responsive and visually engaging experience for the user. The interface will guide users through different scanning modes, display scan progress, and visualize the results in an easy-to-understand format. On the server side, the application will utilize Python with the Flask framework to implement the core backend logic. Flask will serve as the controller that connects the front-end interface with the Nmap scanning engine. Through the use of Python's subprocess module, the backend will securely execute Nmap commands and fetch results in real time.The tool will support two main scanning modes: Quick Scan and Deep Scan. Quick Scan mode, available without login, will allow users to perform basic checks such as ICMP ping sweeps and TCP port scans[2]. In contrast, Deep Scan mode will be available to authenticated users and will provide advanced features like version detection, OS fingerprinting, and script-based scanning for identifying detailed vulnerabilities and misconfigurations.

Another objective is to ensure that scan results are persistently stored either in a local database (such as SQLite) or a cloud-based storage system, depending on user preferences and deployment environments[10]. This persistent storage will enable historical analysis, result comparison, and easier audit tracking. For professional use and compliance purposes, the tool will include functionality to generate PDF reports of Deep Scan results. These reports will summarize findings, highlight critical vulnerabilities, and present data in a well-organized layout suitable for sharing with IT teams or management. This guide will offer step-by-step instructions on how to use each scanning mode, interpret results, and understand key network security concepts, making the platform educational as well as functional. Through these objectives, the project aims to bridge the gap between the technical power of tools like Nmap and the usability demands of a modern, secure, and scalable web application.

1.5 EXPECTED OUTCOMES

The target system, called , is envisioned as a comprehensive and user-centered web-based platform intended to deliver robust network vulnerability scanning and facilitate increased cybersecurity consciousness within academic, administrative, and professional settings. With a Python Flask backend architecture and Nmap for its foundational scanning features, the system allows surface-level as well as deep reconnaissance activities. The frontend, coded with HTML, CSS, and JavaScript, is optimized for responsiveness and usability for hassle-free access from laptops, desktops, and smartphones. The application has two different scanning modes. Quick Scan, accessible by anyone without login authentication, allows one to carry out basic activities like open port enumeration and host discovery—perfect for quick information and leisure learning. Deep Scan, an exclusive feature available for authenticated users, enables specialized capabilities such as operating system fingerprinting, detection of service version, and executing specially designed targeted Nmap scripting engine (NSE) scripts to conduct more profound probing. An innovative feature of the Deep Scan module is how it incorporates up-to-date, real-time feeds from the National Vulnerability Database (NVD) and Common Vulnerabilities and Exposures (CVE) repository. This brings together not only the capability to detect open services but also the ability to correlate them with known vulnerabilities that currently exist, thus enabling accurate threat modeling and risk scoring.

One of the most significant advancements in is the implementation of an AI-powered interactive chatbot. This chatbot functions as a smart assistant, which can read scan results, break down technical terminologies into simple language, and provide mitigation advice based on identified vulnerabilities. This aspect greatly enhances the accessibility of the platform to non-tech users while also being an educational aid for students and cybersecurity trainees. The system will also enable automatic scan report generation and downloading in PDF format. The reports will be nicely formatted to contain metadata like scan timestamp, IP range, scan type, detected hosts, open ports, possible CVEs, and mitigation recommendations. In addition, a back-end relational database (e.g., MySQL or SQLite) will store detailed records of user scans and related metadata, allowing historical scan comparison, vulnerability trend monitoring, and compliance auditing. To enhance the usability of the tool, will feature a built-in, animated user guide that walks users through each step of the scan configuration, execution, and result interpretation process[4]. This interactive help system ensures that even first-time users can confidently use the tool without external guidance. Finally, the app will be tested thoroughly on all the major web browsers (Chrome, Firefox, Microsoft Edge) and operating systems (Windows, macOS, Linux) to ensure wide compatibility and provide a similar experience to every user. Overall, should provide an advanced, smart, and educational platform for real-time scanning of networks and vulnerability assessment to bridge the divide between professional-quality tools and user-friendly cybersecurity tools.

CHAPTER 2

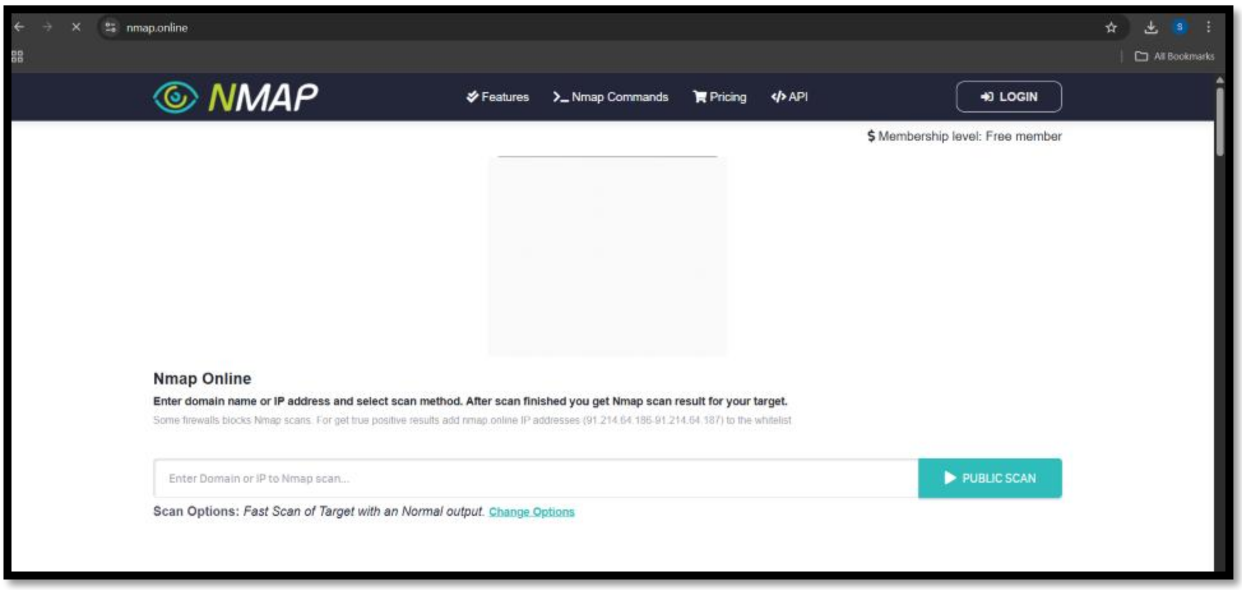
REVERSE ENGINEERING OF EXISTING SOFTWARE

2.1 EXISTING SYSTEM OVERVIEW

Network Scanning Tool NmapOnline, available through websites like https://hackertarget.com/nmap-online-port-scanner, provides an easy yet efficient web-based interface to perform basic port scans and host discovery operations from within a browser. These websites are commonly used by system administrators, cybersecurity students, and ethical hackers to easily obtain network information without having to install or set up command-line tools. Entering a domain name or IP address enables users to make scans that output live hosts, open ports, and basic banners of services. Like many other light Internet scanning tools, though, NmapOnline does not function under a dynamic, interactive model. Users are restricted to choosing between a predefined variety of scan modes with no capacity for varying the intensity of scanning, the time of execution, or incorporating features of Nmap scripting. Moreover, the results of the scan are usually delivered in unformatted plain text without interpretive context or visual analysis. This can be a challenge for new users who might not be able to comprehend the security implications of discovered services and open ports.

As illustrated in Figure 1.2: Nmap Online Interface, the tool is only a simple one-way interface—users initiate a scan and passively accept output—without the capability for dynamic vulnerability mapping, CVE searching, or enrichment via outside threat intelligence feeds like the National Vulnerability Database (NVD). The lack of context analysis compels users to manually match technical information against security advisories, enhancing the chances of error or misinterpretation. Additionally, no integrated chatbot or live help exists to support users in the interpretation of results, learning technical terms, or accessing remediation measures. Facilities such as session logging, user history, reportable results, or comparison of scan results are absent, which lowers reusability and efficiency for long-term security testing.

Although NmapOnline is a handy utility for rapid and remote scanning, its inflexible architecture and absence of intelligent, interactive features restrict its use in contemporary, educational, and professional settings where context, automation, and actionable information are crucial. To fill these voids, more sophisticated systems need to include vulnerability detection, user interaction through conversational interfaces, and real-time data correlation to improve usability and effectiveness.



Nmap Online

Enter domain name or IP address and select scan method. After scan finished you get Nmap scan result for your target.

Some firewalls blocks Nmap scans. For get true positive results add nmap.online IP addresses (91.214.64.186-91.214.64.187) to the whitelist

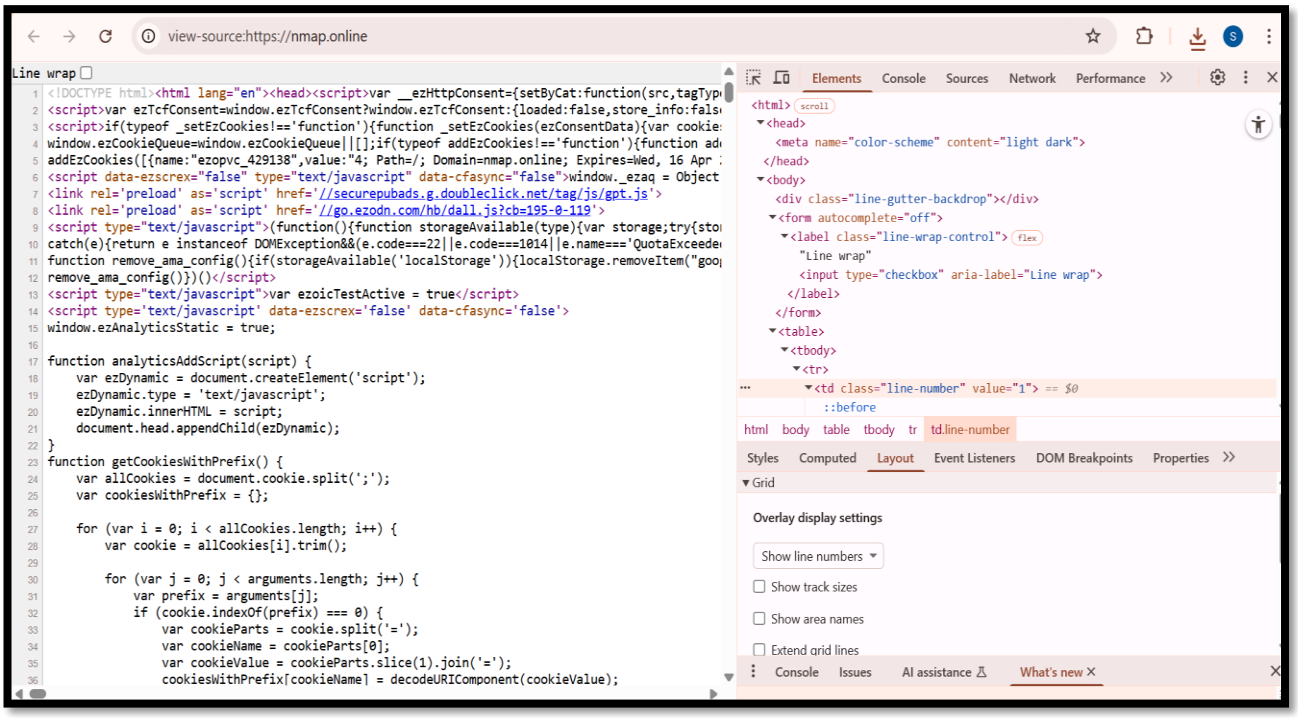
Scan Options: *Fast Scan of Target with an Normal output.*

Figure 2.1: NmapOnline Home Page

2.2 SOURCE CODE ANALYSIS

Since the source code for the NmapOnline system is proprietary and therefore not accessible publicly, a comprehensive code-level reverse engineering analysis is limited. From the examination of the frontend with browser developer tools and network tracing, the system seems to have been developed on conventional server-side rendering technologies perhaps in Python utilizing Flask along with a relational database backend like MySQL or PostgreSQL. JavaScript is fairly used in the frontend for making AJAX requests to allow asynchronous communication with the back-end. The system does not use a contemporary single-page application (SPA) framework, but rather depends on simple form submissions and partial page reloads for some actions.

The UI and layout are static, with no responsive design features, no mobile-first approach, and this might lead to display problems on smaller screens. The frontend employs basic HTML and CSS, with no high-level dynamic client-side interactivity. All of the dynamic data, for example, the scan results, are retrieved from the server for every request, and the page structure is normally reloaded upon a scan's completion. Session management is controlled through cookies, which are used to track users and their scan history. The network scan business logic, user authentication, and result processing are all tightly bound to the backend logic, not providing much room for flexibility or scalability for enhancements in the future. This model implies that the architecture is neither modular nor component-based, reflecting more classic, server-oriented design patterns. The HTML and JavaScript framework, as seen in Figure 2.2, demonstrates little client-side interaction and basic DOM frameworks. The lack of a component- or module-based frontend framework shows that the system employs older, legacy systems. This general architecture indicates that the NmapOnline system adheres to conventional server-side rendering principles but can be limited by scalability and flexibility because of the use of tightly coupled backend logic and minimal utilization of contemporary web development methodologies.



```
<!DOCTYPE html><html lang="en"><head><script>var __ezHttpConsent={setByCat:function(src,tagTyp
<script>var ezTcfConsent=window.ezTcfConsent?window.ezTcfConsent:{loaded:false,store_info:fals
<script>if(typeof _setEzCookies!=='function'){function _setEzCookies(ezConsentData){var cookie
window.ezCookieQueue=window.ezCookieQueue||[];if(typeof addEzCookies!=='function'){function ad
addEzCookies([{name:"ezopvc_429138",value:"4; Path=/; Domain=nmap.online; Expires=Wed, 16 Apr
<script data-ezscrex="false" type="text/javascript" data-cfasync="false">window._ezaq = Object
<link rel='preload' as='script' href='//securepubads.g.doubleclick.net/tag/js/gpt.js'>
<link rel='preload' as='script' href='//go.ezodn.com/hb/dall.js?cb=195-0-119'>
<script type="text/javascript">(function(){function storageAvailable(type){var storage;try{sto
catch(e){return e instanceof DOMException&&(e.code===22||e.code===1014||e.name==='QuotaExceede
function remove_ama_config(){if(storageAvailable('localStorage')){localStorage.removeItem("goo
remove_ama_config()})()</script>
<script type="text/javascript">var ezoicTestActive = true</script>
<script type='text/javascript' data-ezscrex='false' data-cfasync='false'>
window.ezAnalyticsStatic = true;

function analyticsAddScript(script) {
    var ezDynamic = document.createElement('script');
    ezDynamic.type = 'text/javascript';
    ezDynamic.innerHTML = script;
    document.head.appendChild(ezDynamic);
}
function getCookiesWithPrefix() {
    var allCookies = document.cookie.split(';');
    var cookiesWithPrefix = {};

    for (var i = 0; i < allCookies.length; i++) {
        var cookie = allCookies[i].trim();

        for (var j = 0; j < arguments.length; j++) {
            var prefix = arguments[j];
            if (cookie.indexOf(prefix) === 0) {
                var cookieParts = cookie.split('=');
                var cookieName = cookieParts[0];
                var cookieValue = cookieParts.slice(1).join('=');
                cookiesWithPrefix[cookieName] = decodeURIComponent(cookieValue);
```

```
<html> scroll
▾<head>
   <meta name="color-scheme" content="light dark">
 </head>
▾<body>
   <div class="line-gutter-backdrop"></div>
  ▾<form autocomplete="off">
    ▾<label class="line-wrap-control"> flex
       "Line wrap"
       <input type="checkbox" aria-label="Line wrap">
     </label>
   </form>
  ▾<table>
    ▾<tbody>
      ▾<tr>
        ▾<td class="line-number" value="1"> == $0
           ::before
```

html body table tbody tr td.line-number

Styles Computed Layout Event Listeners DOM Breakpoints Properties

▾ Grid

Overlay display settings

Show line numbers

☐ Show track sizes

☐ Show area names

☐ Extend grid lines

Console Issues AI assistance What's new

Figure 2.2: HTML/JavaScript Elements of NmapOnline

2.3 FUNCTIONAL ANALYSIS

Functionally, the Nmap Online platform provides a full range of features optimized for network scanning and vulnerability discovery, suitable for both novice and expert users. The main services provided are port scanning (fast scan, normal scan, and full scan), service version detection, and others like traceroute, firewall detection, and banner grabbing[7]. Users are able to track targets and review results via the interactive report page of the platform, which is designed to give detailed information about open ports, operating services, and possible vulnerabilities. Scanning history and access to professional-strength tools, can only be obtained via the Pro edition. The addition of an API also makes the platform more usable, enabling users to integrate Nmap's scanning functionality into their own systems or automate processes. To integrate Nmap scans into broader security processes or monitoring systems.

Yet, no user interaction or workflow automation other than scanning itself, like ticketing or alerting, is built-in. Users can initiate scans and access reports but cannot have any real-time updating during scan runs or automatic notification of possible threats. Neither are there collaboration tools nor per-user access to historical scan data unless on the Pro plan.This capability is reflected in Figure 2.3, which offers a clear mapping of the features available by user role, from the restrictions on free accounts through to the extended capabilities provided with the Pro subscription. The figure emphasizes the one-way data flow from scan creation to report creation, along with the lack of advanced collaborative or automation features.

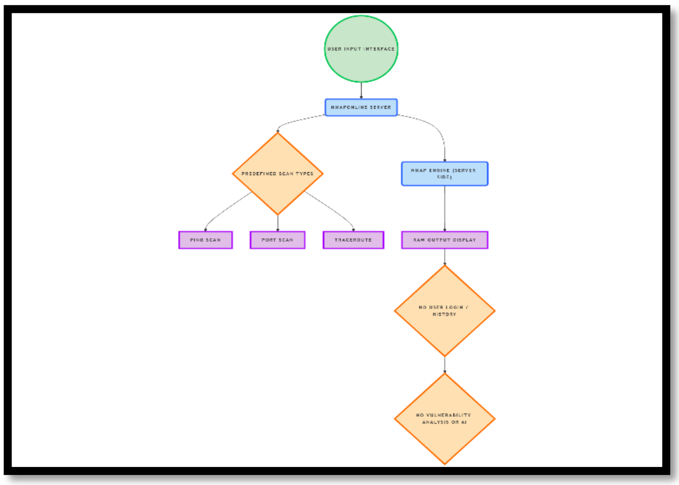
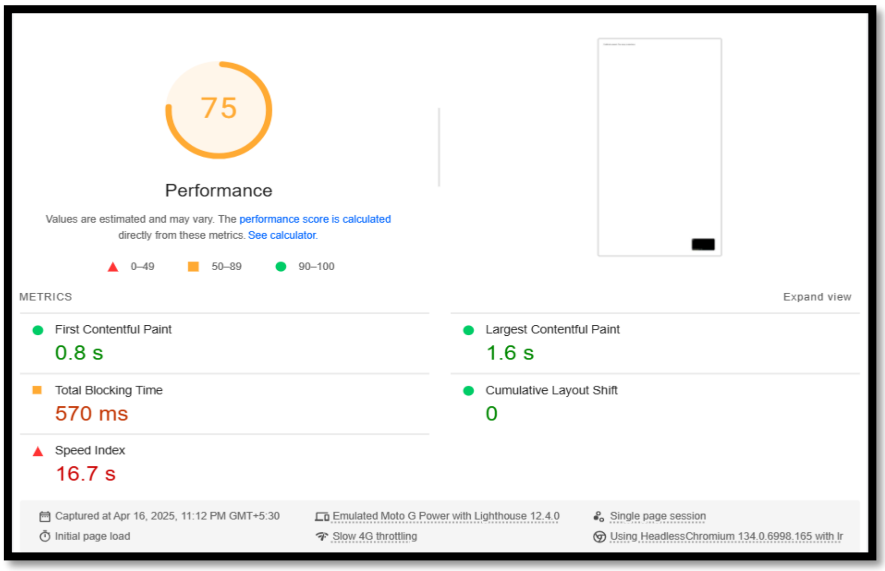


Figure 2.3: Functional Overview of Existing NmapOnline

2.4 PERFORMANCE ANALYSIS

Performance analysis was conducted to understand how well Nmap Online handles different types of scans and what issues might arise during use. The following performance characteristics were noted. Quick Scans: Basic Ping Scans were executed swiftly, often completing in under 5 seconds. This is typical for Ping scans, as they are lightweight operations intended to identify live hosts in the network.Advanced Scans: More resource-intensive scans such as OS Detection and Service Version Scanning took much longer to complete, with scan times reaching up to 30 seconds. These scans require more processing power and network resources, as they attempt to determine the target's operating system and service versions. No Asynchronous Processing: A critical issue with Nmap Online is that it does not support asynchronous processing. When a scan is running, the browser becomes locked, and users cannot interact with the interface until the scan completes. This creates a poor user experience, especially during longer scans, as users are forced to wait without any indication of scan progress. Scalability: Performance issues arise when multiple scans are initiated simultaneously. The tool struggles with handling multiple requests, leading to noticeable delays in executing additional scans or refreshing the page. This highlights a scalability problem that could hinder its usability in environments where many scans are needed in parallel (e.g., during network-wide vulnerability assessments). These performance bottlenecks point to the necessity of adopting background processing in . Utilizing asynchronous handling or job queues could significantly improve user experience and performance scalability, allowing users to initiate multiple scans without delays.


75

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100

METRICS Expand view

● First Contentful Paint
0.8 s

● Largest Contentful Paint
1.6 s

■ Total Blocking Time
570 ms

● Cumulative Layout Shift
0

▲ Speed Index
16.7 s

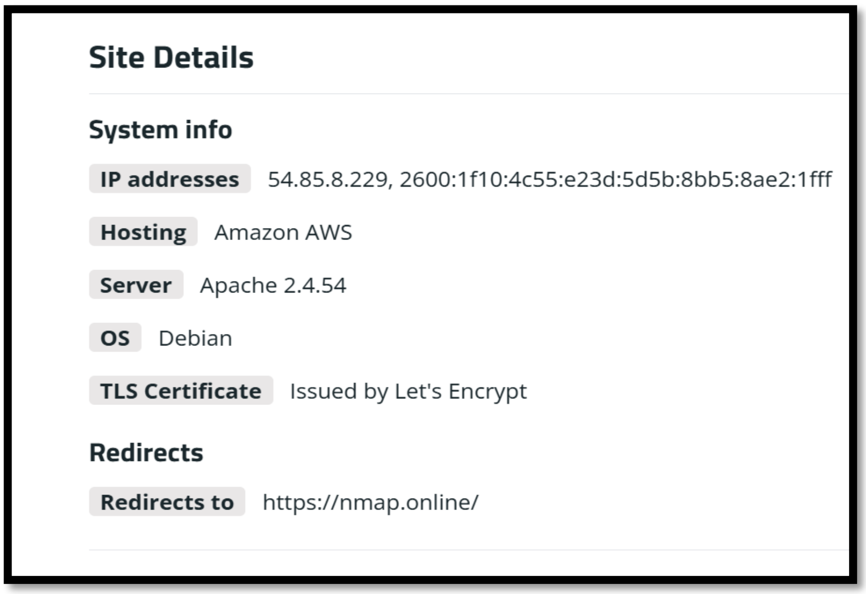
Captured at Apr 16, 2025, 11:12 PM GMT+5:30
Initial page load
Emulated Moto G Power with Lighthouse 12.4.0
Slow 4G throttling
Single page session
Using HeadlessChromium 134.0.6998.165 with lr

Figure 2.4: Page Load Time Analysis

2.5 SECURITY AND PRIVACY ANALYSIS

From a security and privacy perspective, NmapOnline shows essential compliance with broad web application security practices but displays limitations in protection of user information and confidentiality of scans—particularly for non-authenticated (anonymous) users. The site lacks the enforcement of user login to facilitate basic scanning, which allows anonymous users to carry out port scans, causing possible misuse or overuse without apparent accountability. The absence of transport layer security indicators in scans (such as HTTPS enforcement on all sub-routes) is worrying with respect to target data interception. While inputs and outputs for scans seem to be processed across secure channels, there is no detailed explanation of how scan data is stored, processed, and wiped out—particularly for free users who have no access to scan history and session management. For Pro users, API access supports automated scan integration, but there is no definition in the documentation of authentication mechanisms (e.g., OAuth tokens, API key rotation, or IP whitelisting). Without fine-grained API access control or usage quotas, there is a risk of credential leakage or unauthorized access. Furthermore, the platform does not have any discernible vulnerability disclosure program or periodic security audit notices, making its underlying infrastructure security posture uncertain.From a privacy standpoint, NmapOnline lacks a clear privacy policy for how user-submitted target information is processed or stored. Lack of consent dialogues or cookie tracking disclosures also indicates low compliance with data protection laws like GDPR or CCPA, especially for global users.

Figure 2.5. shows the scan result page of NmapOnline. It displays target IP, open ports, services, and OS details. This information is accessible without login, which raises privacy and misuse concerns. Figure highlights missing security headers like Content-Security-Policy and X-Frame-Options. These gaps make the site more vulnerable to attacks like clickjacking and content injection.

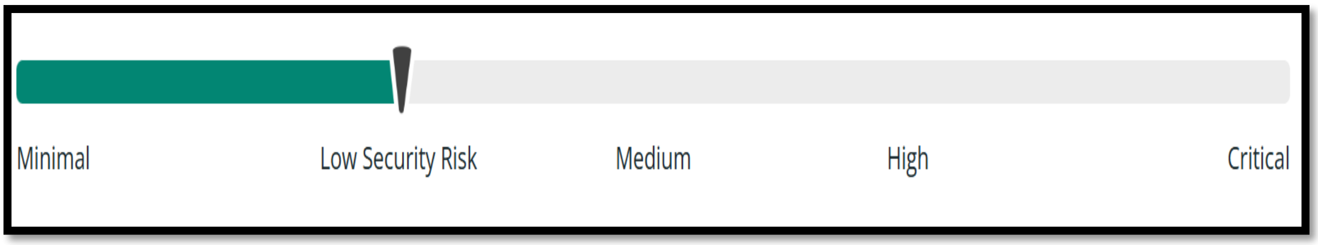

Site Details

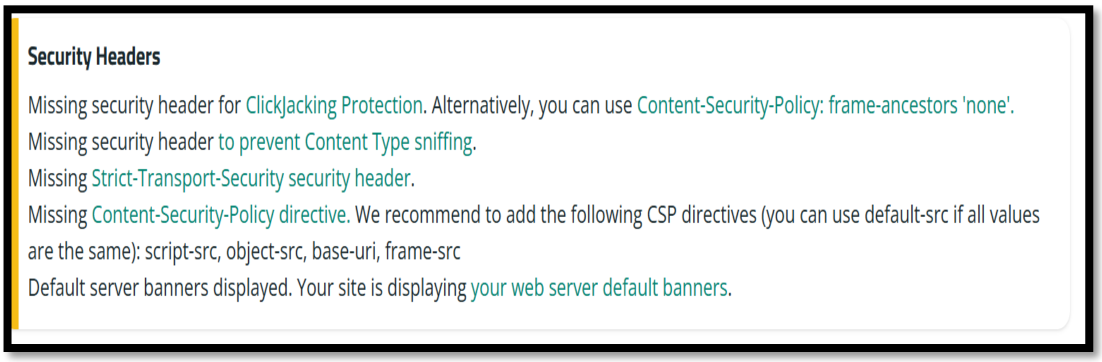
System info

IP addresses	54.85.8.229, 2600:1f10:4c55:e23d:5d5b:8bb5:8ae2:1fff
Hosting	Amazon AWS
Server	Apache 2.4.54
OS	Debian
TLS Certificate	Issued by Let's Encrypt

Redirects

Redirects to	https://nmap.online/




Security Headers

Missing security header for ClickJacking Protection. Alternatively, you can use Content-Security-Policy: frame-ancestors 'none'.
Missing security header to prevent Content Type sniffing.
Missing Strict-Transport-Security security header.
Missing Content-Security-Policy directive. We recommend to add the following CSP directives (you can use default-src if all values are the same): script-src, object-src, base-uri, frame-src
Default server banners displayed. Your site is displaying your web server default banners.

Figure 2.5: Site Detail & Security Headers Report

2.6 RESEARCH GAPS

The NmapOnline service, although providing basic port scanning and network discovery features, has a number of limitations. User authentication and history logging for non-subscribers are not supported, limiting customized use and scan log storage. API access and integration features are reserved for Pro subscribers alone, which limits its usability for automated or large-scale business use in the free service [9]. Moreover, the site does not provide support for scheduling or custom scan configurations, which limits flexibility in performing focused scans. Report creation is minimal with static downloadable output only and no sophisticated visualization or filtering capabilities. In addition, the site is not mobile-friendly, leading to a poor user experience on small screens.

From a security perspective, the lack of robust security headers is indicative of potential concerns regarding data privacy and the web security architecture's overall strength as a platform.

CHAPTER 3

FORWARD ENGINEERING PROCESS

3.1 PROBLEM INDENTIFICATION AND USER ANALYZE

3.1.1 PROBLEM UNDERSTANDING

In the modern digital era, **network security** has become a critical concern due to the growing number of cyberattacks and the increasing complexity of networks. Organizations and individuals need to safeguard their networks, which involves identifying vulnerabilities in their infrastructure, such as unsecured devices, open ports, and weak services. This process is known as **network enumeration**, and it’s a key aspect of vulnerability assessments. One of the most widely used tools for network enumeration is **Nmap (Network Mapper)**, which scans networks and identifies devices and services. Nmap provides a comprehensive view of open ports, services, and even security flaws. However, it has a significant limitation: its **command-line interface** can be intimidating, especially for **beginners** or those not familiar with network security. The command syntax, flags, and options can be difficult to understand and remember, which hampers its adoption among less technical users.

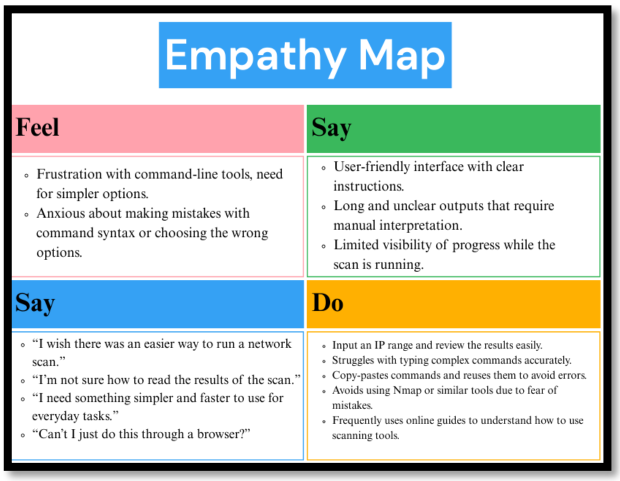
Additionally, while Nmap is a powerful tool, it lacks a **user-friendly graphical interface**. Users need to install and configure it on their machines, and the results are typically presented in a raw format, which is not always easy to interpret. The **problem** here is that despite its capabilities, **Nmap** remains underutilized by a significant portion of potential users because its complexity limits accessibility. For instance, beginners or small-scale users (such as students or non-technical staff in organizations) may not know how to use it effectively, hindering its widespread adoption.

3.1.2 IDENTIFYING USERS & STAKEHOLDERS

The primary users of the NmapOnline service are cybersecurity newbies, IT admins, ethical hackers, and students. Newbies require an easy and accessible interface to learn simple scanning methods[8].IT admins enjoy fast and simple scans with minimum setup. Ethical hackers and security researchers use features such as scan history and reporting. Students and educators teaching cybersecurity classes appreciate using the tool for hands-on training without having to comprehend command-line tools.Primary stakeholders include educational institutions, project mentors, and the wider cybersecurity community. The tool is used by colleges and universities to facilitate practical learning. Mentors and assessors steer the project to ensure that it addresses user requirements.

3.1.3 EMPATHIZING WITH USERS

Empathy mapping and user-centered design methodologies were taken up in the first phase of developing a minimalist NmapOnline interface including engaging with students, newbie cybersecurity professionals, and IT staff through feedback, casual interviews, and usability tests of conventional Nmap CLI applications. These meetings revealed a range of pain points like fear of command-line syntax mistakes, crushing scan outputs, insufficient guidance, and ambiguous scan progress. These findings emphasized the necessity for a browser-based application that provides clarity, simplicity, and automation without compromising functionality. The gathered information was organized visually into an empathy map that, as indicated in Figure 3.1.



Empathy Map

Feel

- Frustration with command-line tools, need for simpler options.
- Anxious about making mistakes with command syntax or choosing the wrong options.

Say

- User-friendly interface with clear instructions.
- Long and unclear outputs that require manual interpretation.
- Limited visibility of progress while the scan is running.

Say

- “I wish there was an easier way to run a network scan.”
- “I’m not sure how to read the results of the scan.”
- “I need something simpler and faster to use for everyday tasks.”
- “Can’t I just do this through a browser?”

Do

- Input an IP range and review the results easily.
- Struggles with typing complex commands accurately.
- Copy-pastes commands and reuses them to avoid errors.
- Avoids using Nmap or similar tools due to fear of mistakes.
- Frequently uses online guides to understand how to use scanning tools.

Figure 3.1: Empathy Map

3.1.4 DEFINING THE PROBLEM

The primary problem is the lack of accessibility of network enumeration tools like Nmap for non-technical users, as current solutions require command-line knowledge or installation. This project aims to address this by creating a web-based tool using Flask and Nmap, offering two modes: **Quick Scan** (for beginners, no login required) and **Deep Scan** (for advanced users, with login and more features). The tool will feature a structured user interface, animated guides, scan history tracking, and PDF report generation. The goal is to make network security more accessible to a broader audience, including beginners, students, and IT professionals.

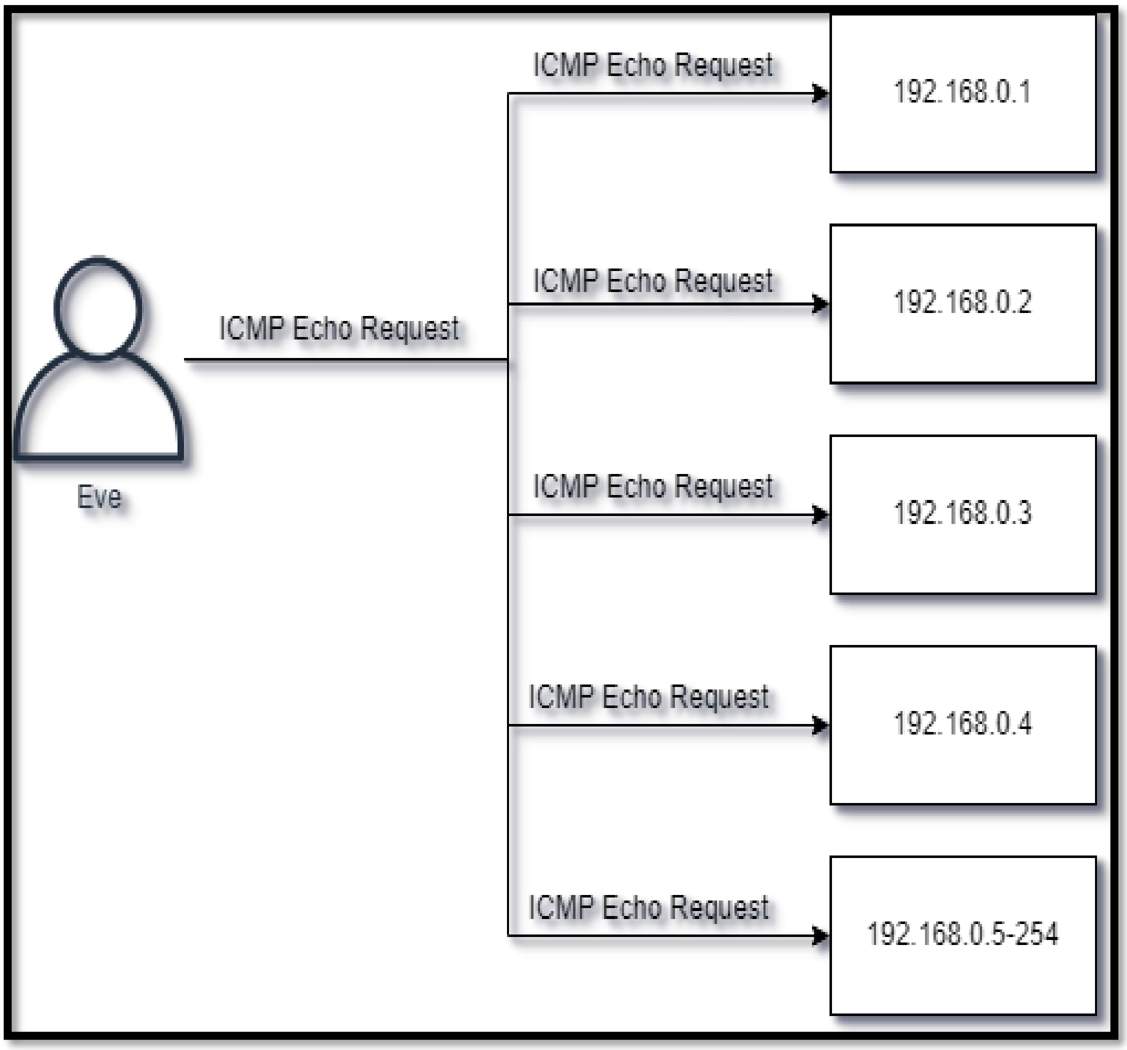


Figure 3.2: Stakeholder Interaction Diagram

3.2 SOFTWARE ARCHITECTURE AND DESIGN

3.2.1 SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

The Software Requirements Specification outlines both functional and non-functional aspects of the system. Functionally, the tool must allow users to input an IP address or a range and support two scanning modes: Quick Scan, which requires no login, and Deep Scan, which is accessible only after authentication. It should execute Nmap commands via Python and display results on the frontend. Input validation is essential to avoid command injection and ensure scan accuracy. Deep Scan results must be stored in the database, and the system should allow exporting scan outcomes as PDF reports[5]. Non-functional requirements include responsiveness across modern browsers, graceful error handling, smooth user experience through real-time feedback, and robust security to prevent unauthorized access or misuse of the scanning engine.

3.2.2 ARCHITECTURAL DESIGN

The system adheres to the traditional three-tier architecture, with the Presentation Layer, Application Layer, and Data Layer. A modular approach ensures improved separation of concerns, maintenance simplicity, and future scalability. The user interface is built with HTML5, CSS3, and JavaScript, following a terminal-inspired layout that will be familiar to command-line tool users. Users can specify an IP address or range, select between Quick or Deep scans, and see real-time updates of scan activity. Real-time interactivity like animations, status messages, and error messages are handled through JavaScript. Optionally, Bootstrap can be incorporated for added responsiveness and visual appearance. Developed using Python 3.2.2 and the Flask web framework, the backend has the application's core functionality. It regulates routing, session management, form validation, and securely builds Nmap commands from user input. Python's subprocess module is utilized to run Nmap scans, while Flask provides authenticated access to enhanced scan modes. This layer applies business logic like user role verification, input sanitizing, and result formatting.

The system uses SQLite as its database solution to handle user accounts, scan history, and report metadata. This lightweight relational database is used to store information like login credentials (with permission levels based on roles), scan settings, and URLs for downloadable reports. It is an important component in supporting the authenticated Deep Scan capability and provides users with the facility to track and revisit past scan results. Figure 3.2.2 depicts the system's three-tier architecture, demonstrating how each layer works together to offer a secure, user-friendly, and effective network scanning platform.

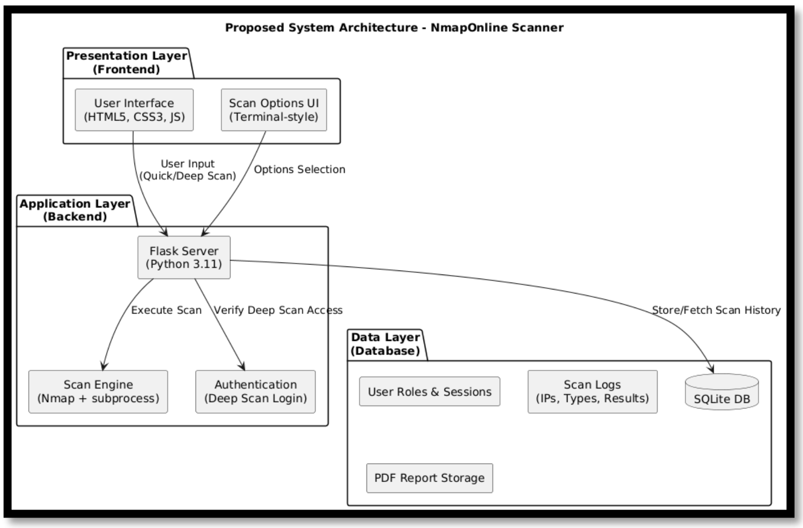

Proposed System Architecture - NmapOnline Scanner
Presentation Layer (Frontend)
User Interface (HTML5, CSS3, JS)
Scan Options UI (Terminal-style)
User Input (Quick/Deep Scan)
Options Selection
Application Layer (Backend)
Flask Server (Python 3.11)
Execute Scan
Verify Deep Scan Access
Store/Fetch Scan History
Scan Engine (Nmap + subprocess)
Authentication (Deep Scan Login)
Data Layer (Database)
User Roles & Sessions
Scan Logs (IPs, Types, Results)
SQLite DB
PDF Report Storage

Figure 3.3: Architecture Design

3.2.3 TECHNOLOGY STACK SELECTION

The system to be suggested utilizes a judiciously chosen technology stack that affords simplicity, dependability, and scalability to users from novice to expert levels. At the frontend, HTML5 organizes content, and CSS3 and terminal-like theme offer a clear and elegant user interface. JavaScript is used to handle dynamic content, real-time updates of results, and responsiveness without complete page reloads. Bootstrap is used optionally to enable responsive design on devices. The backend is constructed in Python 3.11 for readability and extensive library support. Flask, being a lightweight web framework, takes care of routing, form posting, and session management, thus suitable for rapid development and modularity. The Python subprocess module executes Nmap commands securely by sanitizing the input and capturing output for processing. SQLite acts as the back-end database, providing a light and server-less solution for storing scan history, user roles, and report metadata. Nmap acts as the scanning engine core, offering reliable network enumeration and vulnerability testing. These technologies combine to provide a solid, easy-to-use, and informative network scanning platform.

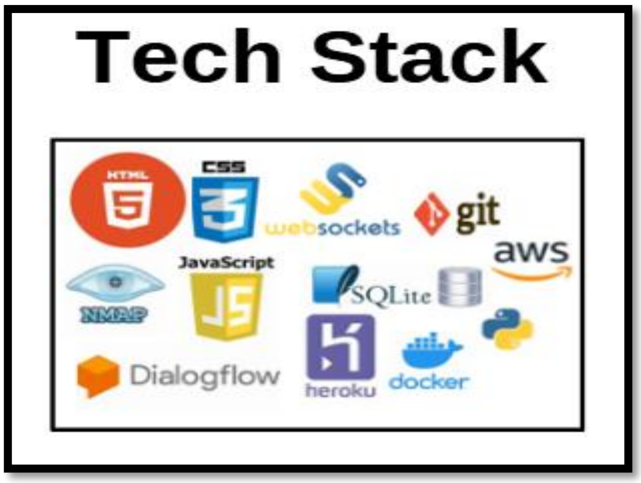


Figure 3.4: Technology Stack Diagram used in the development

3.2.4 DATABASE DESIGN

The database design focuses on supporting key functionalities of the application such as user authentication, scan history tracking, and PDF report generation. A relational database model using SQLite is implemented due to its lightweight nature and ease of integration with Flask. The schema is kept minimal yet efficient, ensuring that both guest and authenticated users can interact with the application seamlessly.

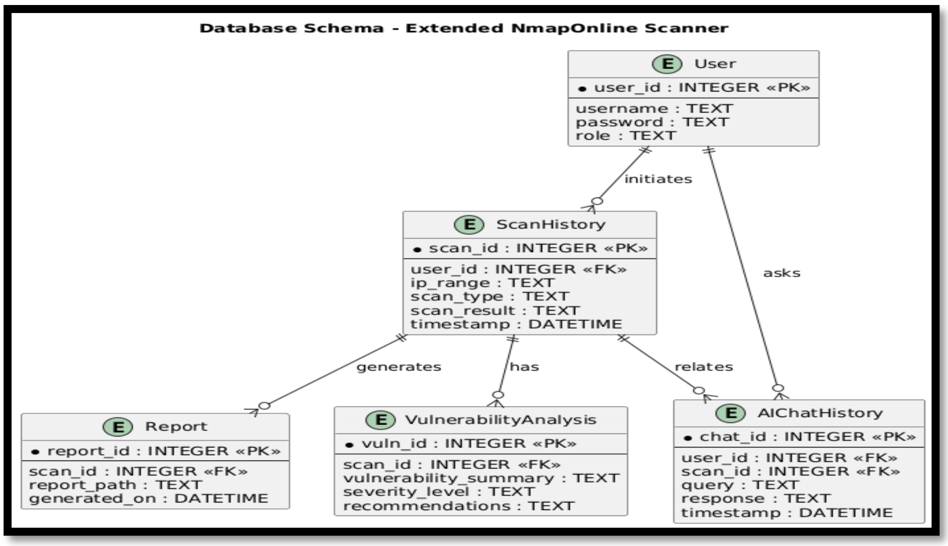


Figure 3.5: Database ER Diagram

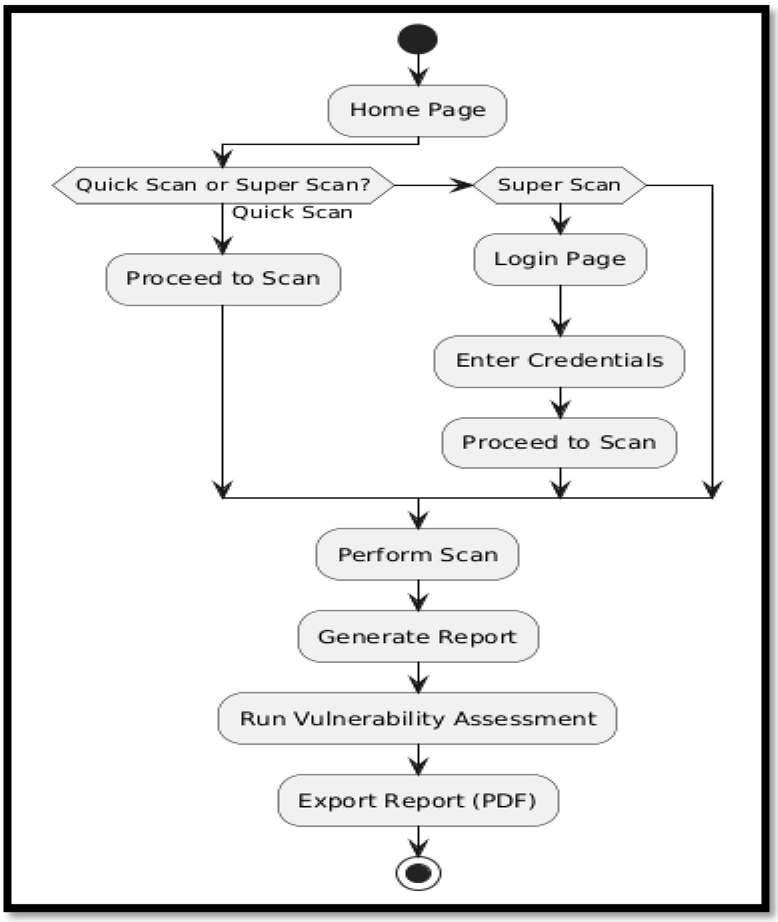
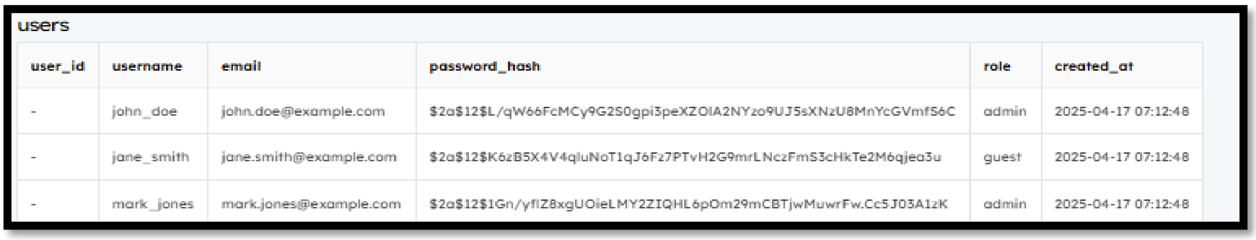


Figure 3.6: Flow Diagram of Operations

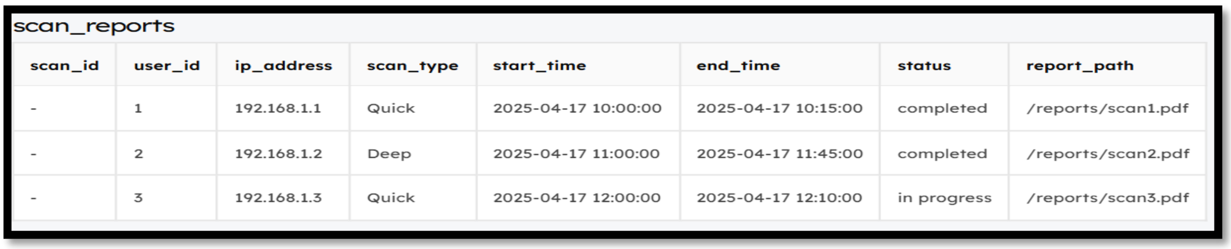
Figure 3.7 shows the structure of the **Users Table**, which stores authentication-related details for each user. It includes a unique ID, username, email, hashed password, role (admin or guest), and the date of account creation. Unique constraints on the username and email prevent duplicates, while the role field controls access permissions within the application.


users

user_id	username	email	password_hash	role	created_at
-	john_doe	john.doe@example.com	$2a$12$L/qW66FcMCy9G2S0gpi3peXZOIA2NYzo9UJ5sXNzU8MnYcGVmfS6C	admin	2025-04-17 07:12:48
-	jane_smith	jane.smith@example.com	$2a$12$K6zB5X4V4qluNoT1qJ6Fz7PTvH2G9mrLNczFmS3cHkTe2M6qjea3u	guest	2025-04-17 07:12:48
-	mark_jones	mark.jones@example.com	$2a$12$1Gn/yflZ8xgUOieLMY2ZIQHL6pOm29mCBTjwMuwrFw.Cc5J03A1zK	admin	2025-04-17 07:12:48

Figure 3.7: users Table entry

Figure 3.8 represents the **Scan Reports Table**, which logs details about every network scan performed by users. Each record includes the scan ID, associated user ID, target IP address, scan type (Quick or Deep), timestamps for the start and end of the scan, the scan's status, and the file path where the report is stored. It is linked to the Users Table via a foreign key.

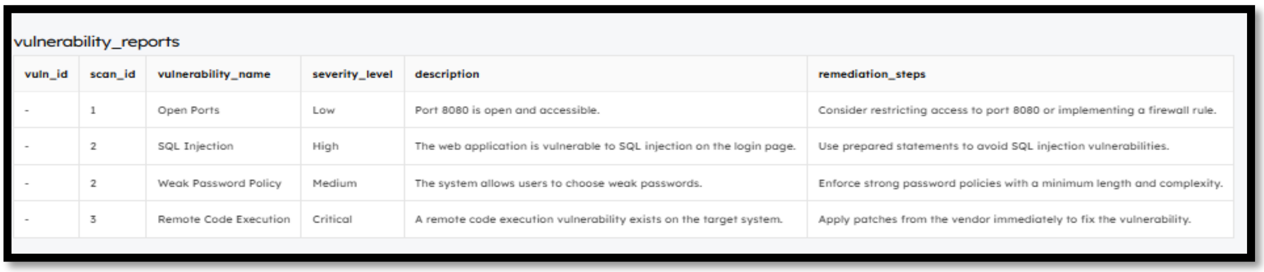


scan_reports

scan_id	user_id	ip_address	scan_type	start_time	end_time	status	report_path
-	1	192.168.1.1	Quick	2025-04-17 10:00:00	2025-04-17 10:15:00	completed	/reports/scan1.pdf
-	2	192.168.1.2	Deep	2025-04-17 11:00:00	2025-04-17 11:45:00	completed	/reports/scan2.pdf
-	3	192.168.1.3	Quick	2025-04-17 12:00:00	2025-04-17 12:10:00	in progress	/reports/scan3.pdf

Figure 3.8: scan_reports Table Entry

Figure 3.9 shows the **Vulnerability Reports Table**, which stores information about vulnerabilities found during scans. Each entry includes a unique vulnerability ID, the associated scan ID, name of the vulnerability, severity level, a description, and remediation steps. It is linked to the Scan Reports Table using a foreign key to ensure traceability.

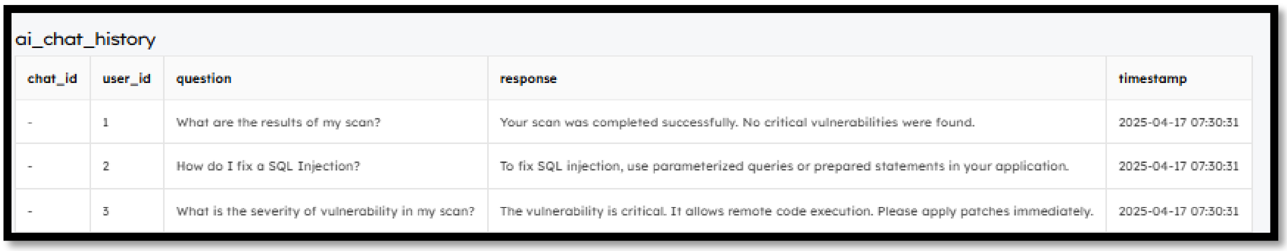


vulnerability_reports

vuln_id	scan_id	vulnerability_name	severity_level	description	remediation_steps
-	1	Open Ports	Low	Port 8080 is open and accessible.	Consider restricting access to port 8080 or implementing a firewall rule.
-	2	SQL Injection	High	The web application is vulnerable to SQL injection on the login page.	Use prepared statements to avoid SQL injection vulnerabilities.
-	2	Weak Password Policy	Medium	The system allows users to choose weak passwords.	Enforce strong password policies with a minimum length and complexity.
-	3	Remote Code Execution	Critical	A remote code execution vulnerability exists on the target system.	Apply patches from the vendor immediately to fix the vulnerability.

Figure 3.9: vulnerability_report Table entry

Figure 3.10 illustrates the **AI Chat History Table**, which keeps a record of user interactions with the AI assistant [6]. Each entry includes a chat ID, the user ID who asked the question, the question itself, the AI's response, and the timestamp of the conversation. This helps in maintaining user-specific AI history and feedback.



ai_chat_history

chat_id	user_id	question	response	timestamp
-	1	What are the results of my scan?	Your scan was completed successfully. No critical vulnerabilities were found.	2025-04-17 07:30:31
-	2	How do I fix a SQL Injection?	To fix SQL injection, use parameterized queries or prepared statements in your application.	2025-04-17 07:30:31
-	3	What is the severity of vulnerability in my scan?	The vulnerability is critical. It allows remote code execution. Please apply patches immediately.	2025-04-17 07:30:31

Figure 3.10: ai_chat_history Table Entry

3.3 DEVELOPMENT AND IMPLEMENTATION

3.3.1 API AND DATA FLOW DESIGN

In the proposed web-based Nmap scanning tool, API integration serves as the communication bridge between the frontend (user interface) and the backend (server logic). The backend is developed using Python's Flask framework, which exposes a set of RESTful API endpoints. These endpoints handle essential functions such as user authentication (login/signup), executing Nmap scans (quick and deep), retrieving scan history, and generating PDF reports. Each API is designed to respond to specific HTTP methods like POST or GET, allowing the frontend to send user inputs and receive responses dynamically without reloading the page. For example, when a user initiates a quick scan from the browser interface, the frontend sends a POST request to the /api/quickscan endpoint with the target IP address. The Flask server processes this request, runs the Nmap command using the subprocess module, and returns the scan result to be displayed immediately in the browser.

The deep scan functionality works in a more advanced manner. When a logged-in user selects custom scan parameters (like specific ports or scan types), these options are sent to the /api/deepscan endpoint. The backend then authenticates the user session, validates the inputs to prevent malicious commands, and securely executes the scan. Once completed, the results are not only shown to the user but also saved into the database for future reference. Similarly, the /api/scan-history endpoint allows authenticated users to view a list of their previous scans, while the /api/report/<scan_id> endpoint is responsible for generating downloadable PDF reports from stored scan results. This modular API structure makes the system both secure and scalable, providing a clean separation of concerns between different components.

The data flow design of the system outlines how data moves from user actions on the frontend to processing on the backend and finally to storage or output. When a user interacts with the interface—such as entering an IP address and choosing a scan type—the data is first captured by the frontend, built using HTML, CSS, and JavaScript. These inputs are then sent to the appropriate backend API endpoint using AJAX or the Fetch API. On the backend, Flask receives the request and processes it accordingly. For scans, it validates the input data to ensure it's in the correct format and free from any harmful content. It then constructs a secure Nmap command and executes it using Python's subprocess module. For quick scans, the result is returned immediately to the frontend and displayed on the screen. These results are not stored in the database, as the user is not authenticated.

In contrast, deep scans (available only to logged-in users) have additional layers. After execution, the scan result is saved in the SQLite database along with details such as the user ID, scan time, IP range, and scan type. If the user chooses to generate a report, the backend fetches the relevant data and uses tools like xhtml2pdf or FPDF to create a professionally formatted PDF file. This report can then be downloaded directly from the frontend. The flow of data—starting from user input, moving through validation, execution, storage, and finally back to output—ensures that the tool remains user-friendly, secure, and efficient, meeting both beginner and expert needs in the network security domain.

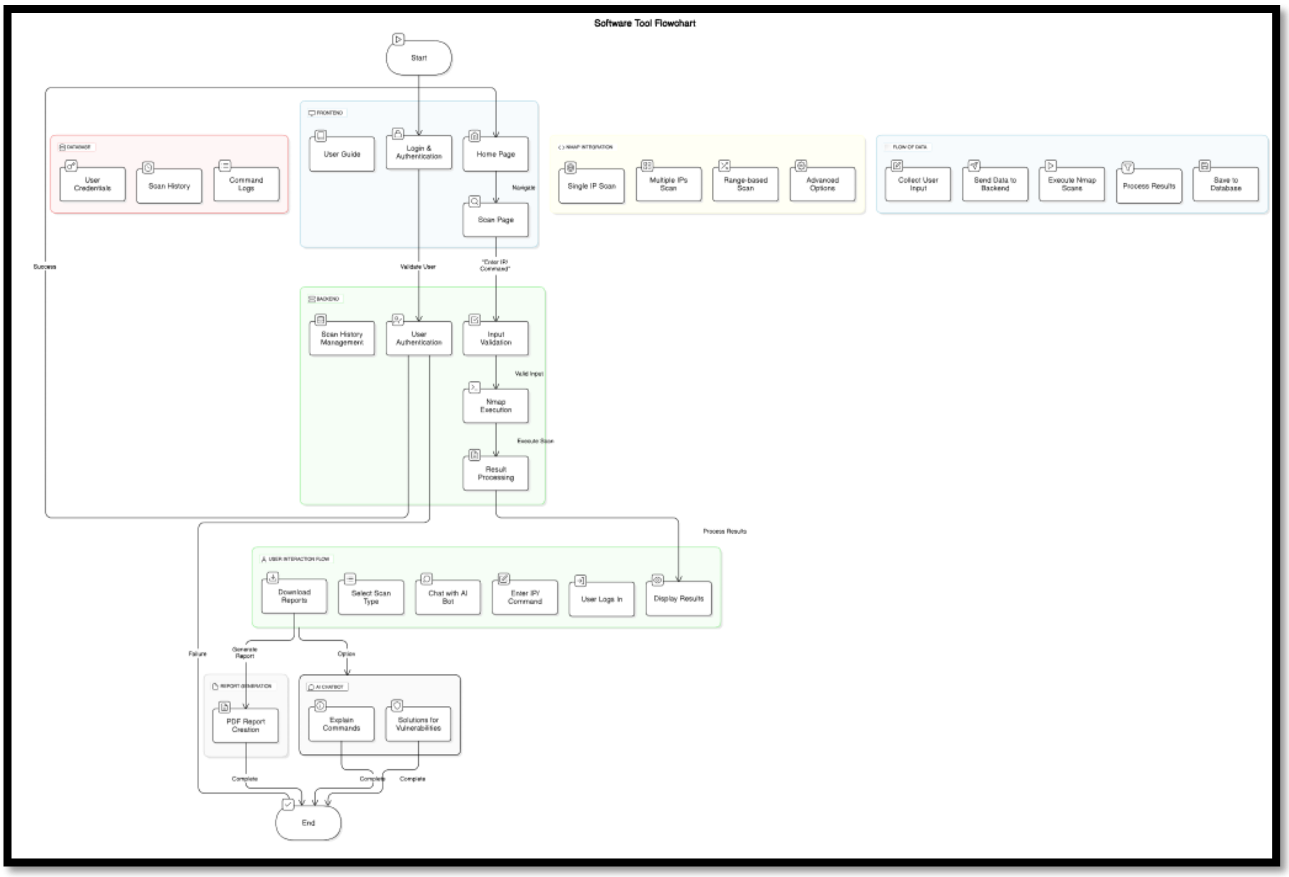


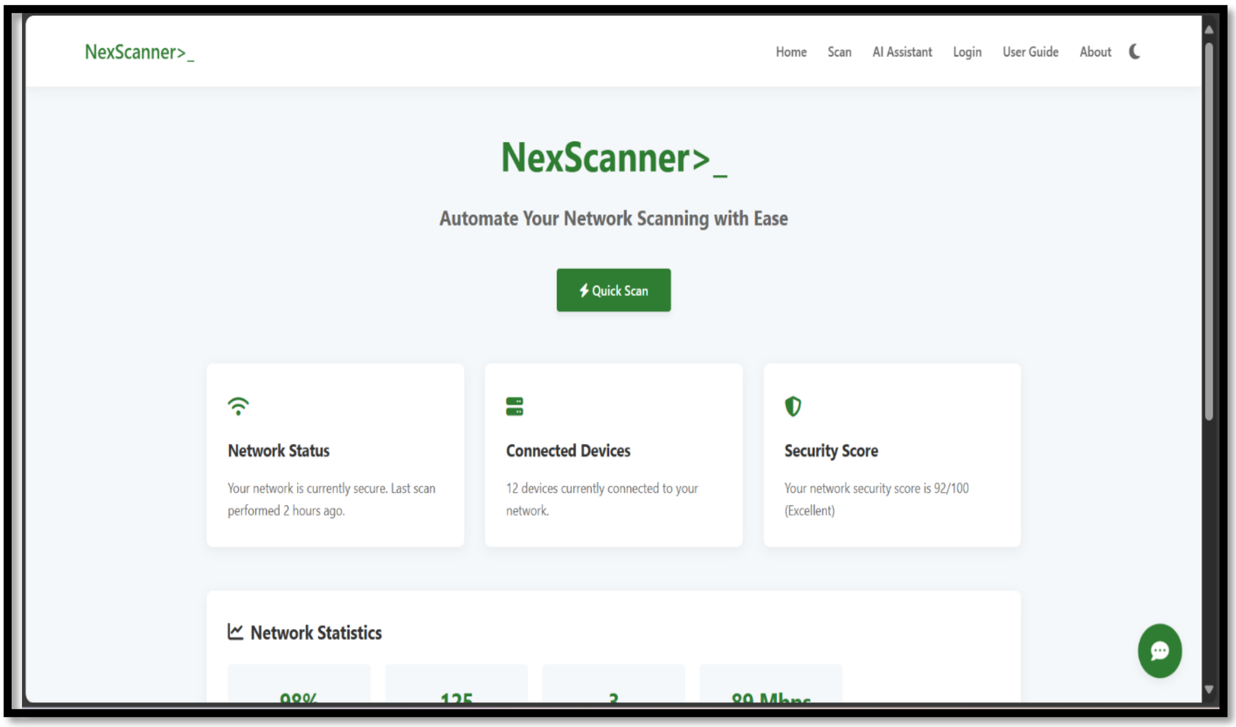
Figure 3.11: Data Flow Diagram

3.3.2 MODULES

The web-based Nmap scanning system is divided into several key modules, each responsible for a specific set of functions, ensuring modularity, ease of maintenance, and scalability. These modules work together to deliver a seamless user experience for both beginners and advanced users engaging in network enumeration and vulnerability scanning. The User Authentication Module manages the registration and login functionality. It securely handles user credentials by hashing passwords before storing them in the database. This module also distinguishes between guest users (for quick scans) and authenticated users (for deep scans with history tracking and report generation). Session management is also handled within this module to ensure secure access to user-specific data.

The Scanning Module is at the core of the system. It is responsible for accepting input parameters such as IP address or range, validating them, and constructing the appropriate Nmap command. It uses Python's subprocess module to execute the scan and capture the output. Depending on whether it's a quick scan or deep scan, the output is either displayed directly or stored in the database for future reference. The Scan History and Report Module is available only to authenticated users. This module stores the output of deep scans in a structured format and allows users to view a list of previous scans along with their details. It also includes functionality for generating downloadable PDF reports using tools like xhtml2pdf or FPDF, making it useful for documentation, academic reports, or professional record-keeping. The Frontend Interface Module provides a clean, responsive, and interactive user experience. Built using HTML, CSS, and JavaScript (optionally enhanced with Bootstrap), it allows users to enter scan details, trigger scans, view results in real time, and download reports. This module ensures that even users with no prior command-line experience can use the tool effectively.

Finally, the API Communication Module handles data exchange between the frontend and backend. It ensures that user inputs are properly sent to the server and that responses—such as scan results or login status—are efficiently returned. This module is vital for real-time interactivity, smooth transitions between user actions, and secure data handling.



NexScanner>_

Automate Your Network Scanning with Ease

Network Status

Your network is currently secure. Last scan performed 2 hours ago.

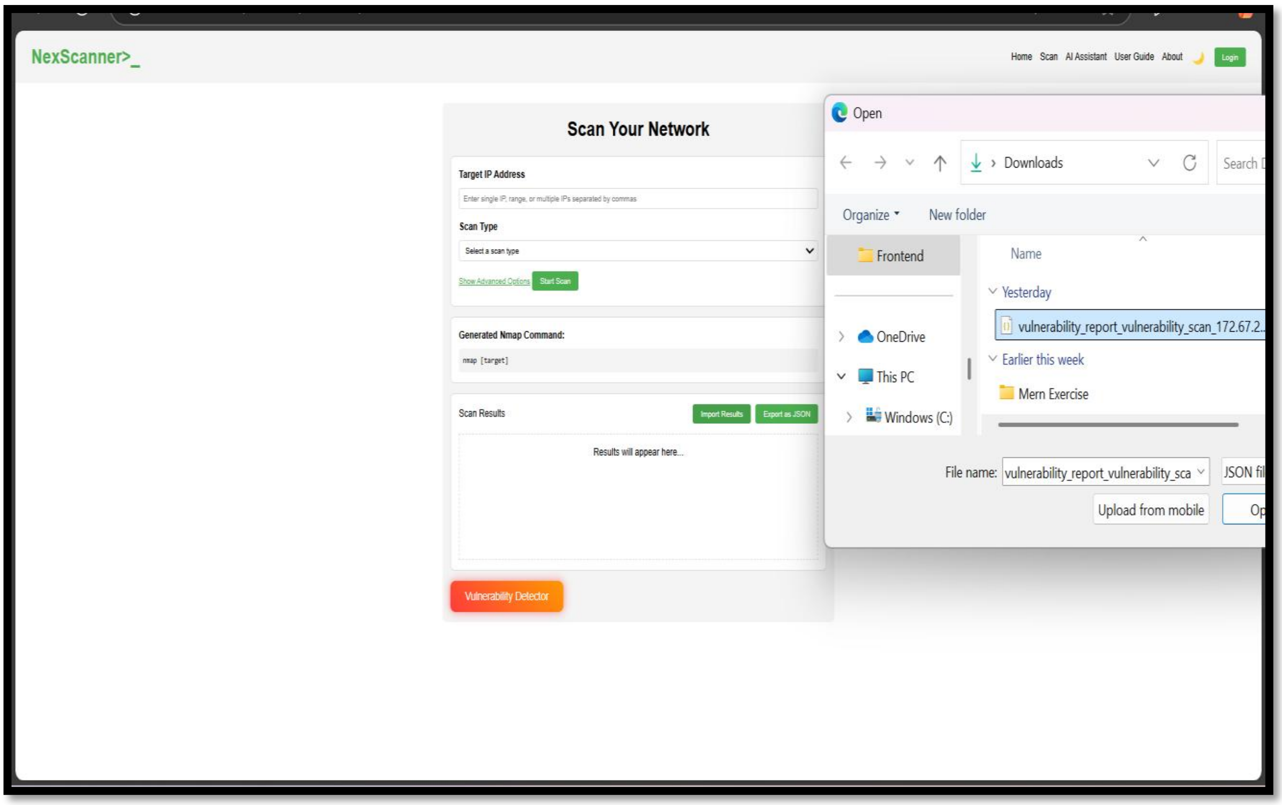
Connected Devices

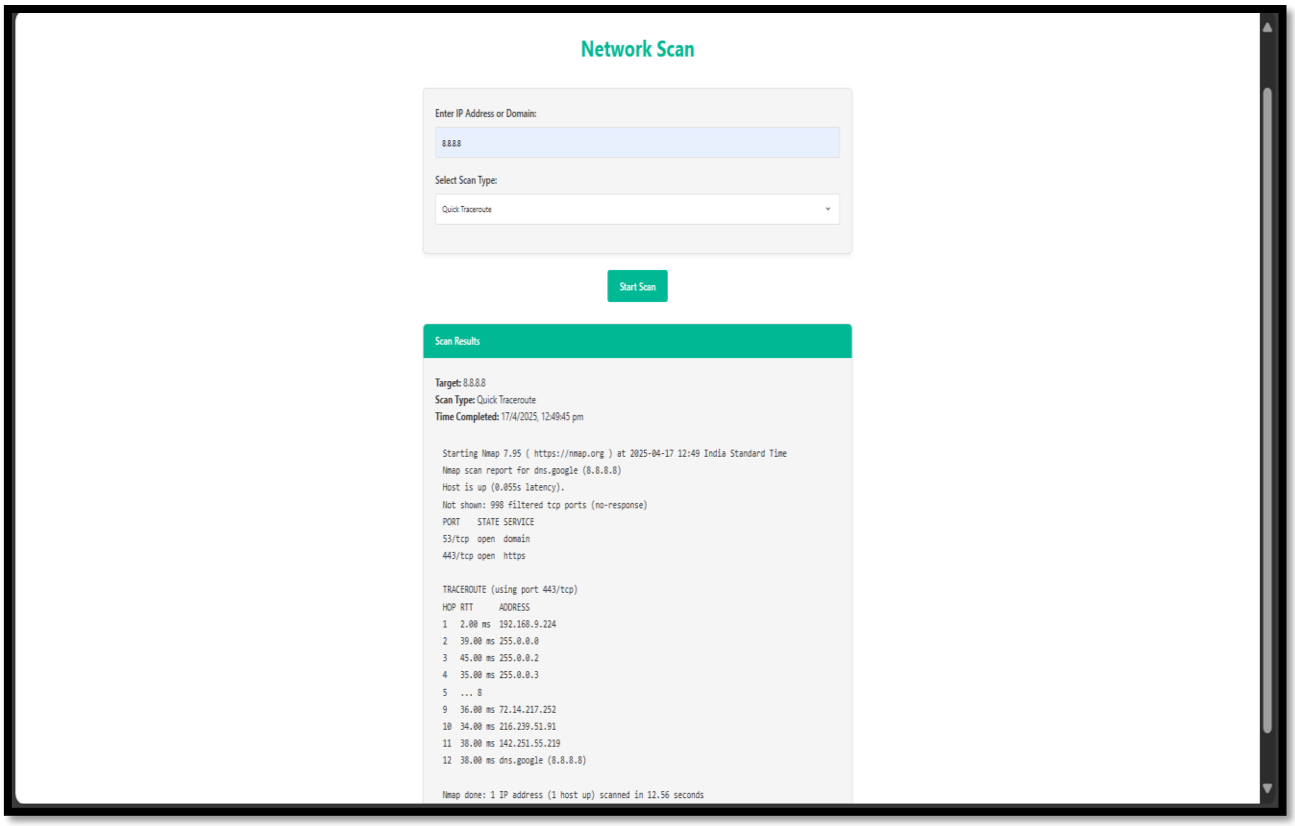
12 devices currently connected to your network.

Security Score

Your network security score is 92/100 (Excellent)

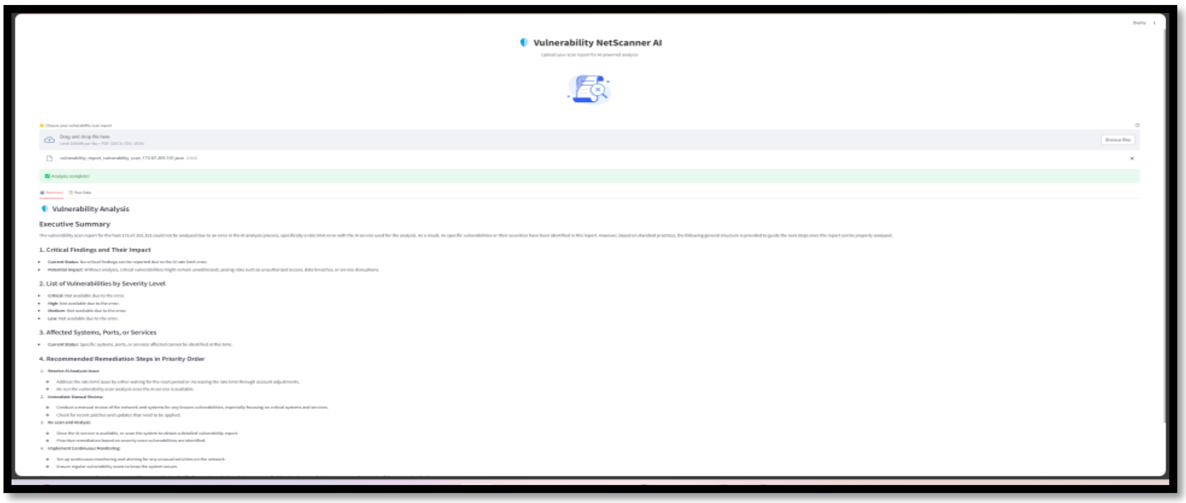
Network Statistics





Target: 8.8.8.8
Scan Type: Quick Traceroute
Time Completed: 17/4/2025, 12:49:45 pm

```
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-17 12:49 India Standard Time
Nmap scan report for dns.google (8.8.8.8)
Host is up (0.055s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
53/tcp  open  domain
443/tcp open  https

TRACEROUTE (using port 443/tcp)
HOP RTT      ADDRESS
1   2.00 ms  192.168.9.224
2   39.00 ms 255.0.0.0
3   45.00 ms 255.0.0.2
4   35.00 ms 255.0.0.3
5   ... 8
9   36.00 ms 72.14.217.252
10  34.00 ms 216.239.51.91
11  38.00 ms 142.251.55.219
12  38.00 ms dns.google (8.8.8.8)

Nmap done: 1 IP address (1 host up) scanned in 12.56 seconds
```



Vulnerability NetScanner AI

Upload your scan report for AI powered analysis

Choose your vulnerability scan report

Drag and drop file here

Limit 200MB per file • PDF, DOCX, CSV, JSON

vulnerability_report_vulnerability_scan_172.67.205.101.json

Analysis complete!

Vulnerability Analysis

Executive Summary

The vulnerability scan report for the host 172.67.205.101 could not be analyzed due to an error in the AI analysis process, specifically a rate limit error with the AI service used for the analysis. As a result, no specific vulnerabilities or their severities have been identified in this report. However, based on standard practices, the following general structure is provided to guide the next steps once the report can be properly analyzed.

1. Critical Findings and Their Impact

- **Current Status:** No critical findings can be reported due to the AI rate limit error.
- **Potential Impact:** Without analysis, critical vulnerabilities might remain unaddressed, posing risks such as unauthorized access, data breaches, or service disruptions.

2. List of Vulnerabilities by Severity Level

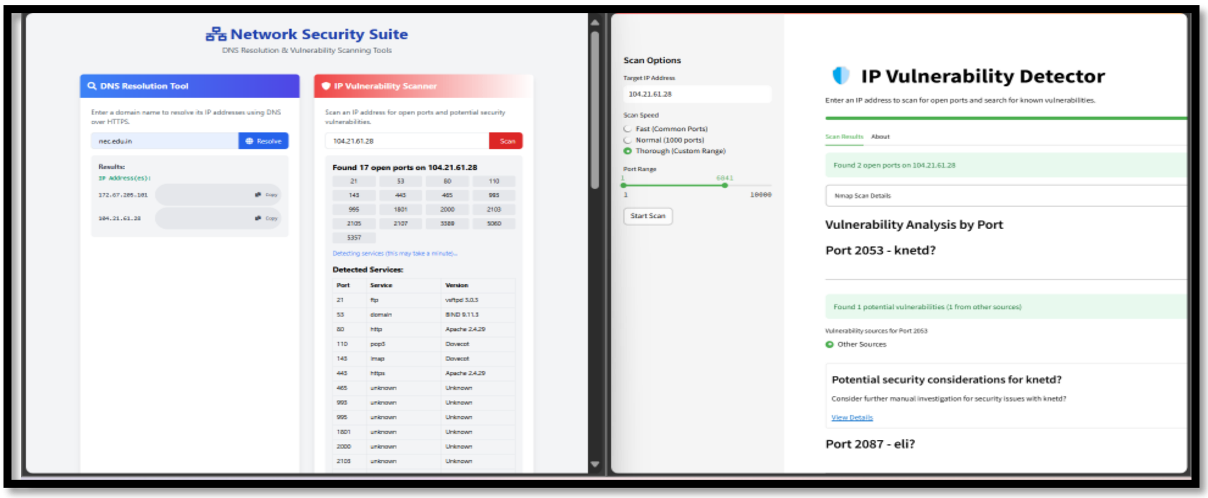
- **Critical:** Not available due to the error.
- **High:** Not available due to the error.
- **Medium:** Not available due to the error.
- **Low:** Not available due to the error.

3. Affected Systems, Ports, or Services

- **Current Status:** Specific systems, ports, or services affected cannot be identified at this time.

4. Recommended Remediation Steps in Priority Order

1. **Resolve AI Analysis Issue:**
   - Address the rate limit issue by either waiting for the reset period or increasing the rate limit through account adjustments.
   - Re-run the vulnerability scan analysis once the AI service is available.
2. **Immediate Manual Review:**
   - Conduct a manual review of the network and systems for any known vulnerabilities, especially focusing on critical systems and services.
   - Check for recent patches and updates that need to be applied.
3. **Re-scan and Analyze:**
   - Once the AI service is available, re-scan the system to obtain a detailed vulnerability report.
   - Prioritize remediation based on severity once vulnerabilities are identified.
4. **Implement Continuous Monitoring:**
   - Set up continuous monitoring and alerting for any unusual activities on the network.
   - Ensure regular vulnerability scans to keep the system secure.



Network Security Suite

DNS Resolution & Vulnerability Scanning Tools

DNS Resolution Tool

Enter a domain name to resolve its IP addresses using DNS over HTTPS.

nec.edu.in

Results:

IP Address(es):

172.67.205.101

104.21.61.28

IP Vulnerability Scanner

Scan an IP address for open ports and potential security vulnerabilities.

104.21.61.28

Found 17 open ports on 104.21.61.28

21	53	80	110
143	443	465	993
995	1801	2000	2103
2105	2107	3389	5060
5357			

Detecting services (this may take a minute)...

Detected Services:

Port	Service	Version
21	ftp	vsftpd 3.0.3
53	domain	BIND 9.11.3
80	http	Apache 2.4.29
110	pop3	Dovecot
143	imap	Dovecot
443	https	Apache 2.4.29
465	unknown	Unknown
993	unknown	Unknown
995	unknown	Unknown
1801	unknown	Unknown
2000	unknown	Unknown
2103	unknown	Unknown

Scan Options

Target IP Address: 104.21.61.28

Scan Speed

- Fast (Common Ports)
- Normal (1000 ports)
- [✓ Thorough (Custom Range)]

Port Range: 1 – 6841 (1 to 10000)

IP Vulnerability Detector

Enter an IP address to scan for open ports and search for known vulnerabilities.

Found 2 open ports on 104.21.61.28

Vulnerability Analysis by Port

Port 2053 - knetd?

Found 1 potential vulnerabilities (1 from other sources)

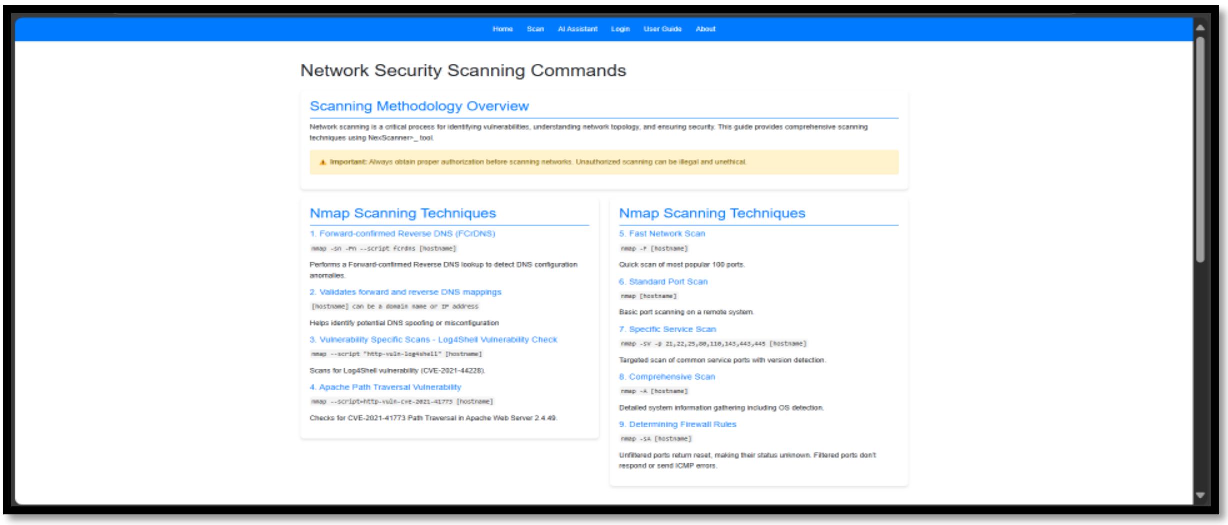
Vulnerability sources for Port 2053

[✓ Other Sources]

Potential security considerations for knetd?

Consider further manual investigation for security issues with knetd?

Port 2087 - eli?



Network Security Scanning Commands

Scanning Methodology Overview

Network scanning is a critical process for identifying vulnerabilities, understanding network topology, and ensuring security. This guide provides comprehensive scanning techniques using NexScanner>_ tool.

⚠ **Important:** Always obtain proper authorization before scanning networks. Unauthorized scanning can be illegal and unethical.

Nmap Scanning Techniques

1. Forward-confirmed Reverse DNS (FCrDNS)

```
nmap -sn -Pn --script fcrdns [hostname]
```

Performs a Forward-confirmed Reverse DNS lookup to detect DNS configuration anomalies.

2. Validates forward and reverse DNS mappings

```
[hostname] can be a domain name or IP address
```

Helps identify potential DNS spoofing or misconfiguration

3. Vulnerability Specific Scans - Log4Shell Vulnerability Check

```
nmap --script "http-vuln-log4shell" [hostname]
```

Scans for Log4Shell vulnerability (CVE-2021-44228).

4. Apache Path Traversal Vulnerability

```
nmap --script=http-vuln-cve-2021-41773 [hostname]
```

Checks for CVE-2021-41773 Path Traversal in Apache Web Server 2.4.49.

Nmap Scanning Techniques

5. Fast Network Scan

```
nmap -F [hostname]
```

Quick scan of most popular 100 ports.

6. Standard Port Scan

```
nmap [hostname]
```

Basic port scanning on a remote system.

7. Specific Service Scan

```
nmap -sV -p 21,22,25,80,110,143,443,445 [hostname]
```

Targeted scan of common service ports with version detection.

8. Comprehensive Scan

```
nmap -A [hostname]
```

Detailed system information gathering including OS detection.

9. Determining Firewall Rules

```
nmap -sA [hostname]
```

Unfiltered ports return reset, making their status unknown. Filtered ports don't respond or send ICMP errors.

Figure 3.12: Complete Module of .Application

3.3.3 FEATURE IMPLEMENTATION

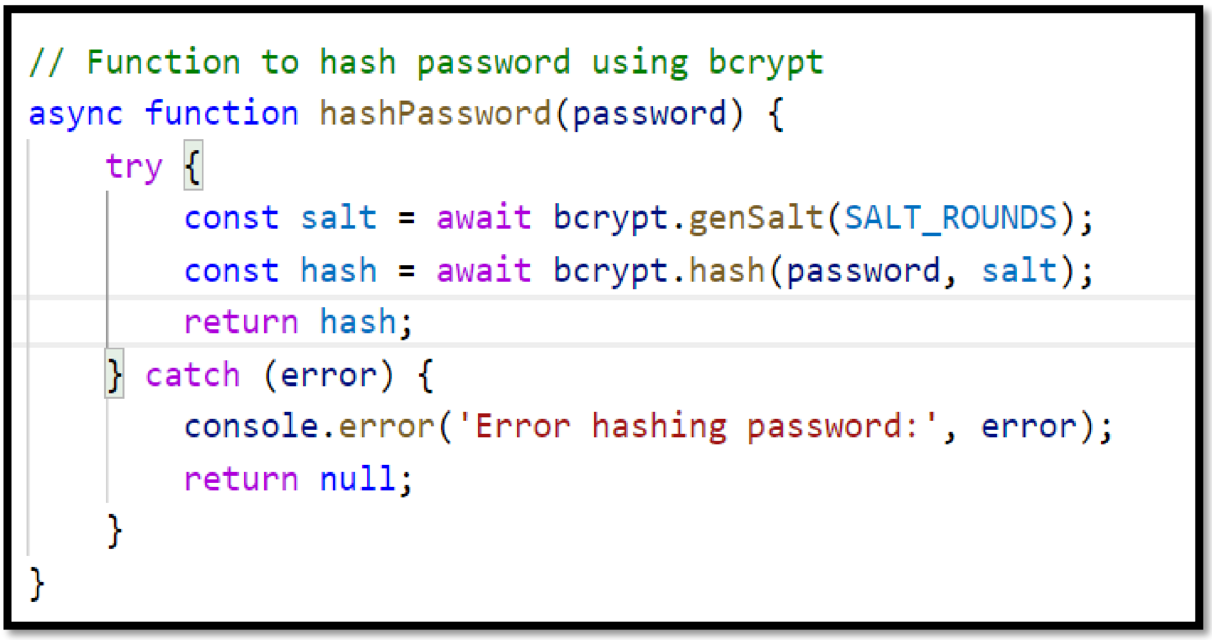
The feature implementation of the web-based Nmap scanning tool focuses on providing a user-friendly, secure, and efficient experience. It includes a user authentication system for login and session management, allowing users to access both quick scans (for guests) and deep scans (for authenticated users). The scanning process is handled by Python and Flask, which construct and execute Nmap commands securely. Users can view past scans and generate downloadable PDF reports for deep scan results. The frontend, built with HTML, CSS, and JavaScript, provides an intuitive, terminal-style interface, dynamically updating as scans progress. Security measures include input validation, session management, and role-based access control to protect user data. The system also ensures real-time feedback during scans and formats the results for easy interpretation. These features combine to create a powerful yet accessible tool for both beginners and advanced users in the cybersecurity field.

3.3.4 UI/UX CONSIDERATIONS

The UI/UX design for the web-based Nmap scanning tool integrates HTML, CSS, JavaScript, and Python Flask to provide a seamless, intuitive user experience. HTML forms the structure of the web page, organizing the content into sections for input (such as IP addresses), scan initiation buttons, and the display of results. The layout uses semantic HTML elements to ensure accessibility and proper organization. CSS enhances the interface with a terminal-like theme, familiar to cybersecurity professionals, and ensures the design is responsive across various screen sizes, using media queries. Interactive elements such as buttons and input fields are styled with hover and focus effects to guide users smoothly through the tool. JavaScript is responsible for dynamic interactions, such as real-time scan progress updates, input validation, and updating the interface with scan results without requiring a page reload. It also ensures that user inputs are correctly formatted before executing scans. On the backend, Python Flask manages the application's business logic, handling user authentication, constructing Nmap commands, and executing them using Python’s subprocess module. Flask also manages the scan results, sending them back to the frontend and enabling the generation of downloadable PDF reports for deep scans. Together, these technologies ensure the tool is user-friendly, interactive, and secure, making it accessible to both beginners and advanced users in the cybersecurity field.

3.3.5 SECURITY AND PRIVACY ENHANCEMENTS

Security was a key aspect in the development and design of the platform. To provide secure user authentication, bcrypt.js was utilized for password hashing with 10 rounds of salt, as well as a password validation mechanism. AES-256 encryption with CryptoJS was used for securely storing user credentials if the "Remember Me" feature is activated, with encryption keys being generated via a secure process. To improve user experience while enhancing security[12], a password strength meter was included that gave instant feedback in the form of a color-coded indicator of password strength. Cross-Site Request Forgery (CSRF) protection was implemented through token generation and verification upon form submission. Other security features consist of the implementation of correct autocomplete attributes, secure session management, and cryptographically secure random value generation. Although client-side functionality enhances user experience, production deployment prioritizes server-side security practices such as secure storage of hashed credentials, HTTPS-only transport, and usage of HTTP-only secure cookies for authentication. Such layered security features together add robustness and reliability to the system.



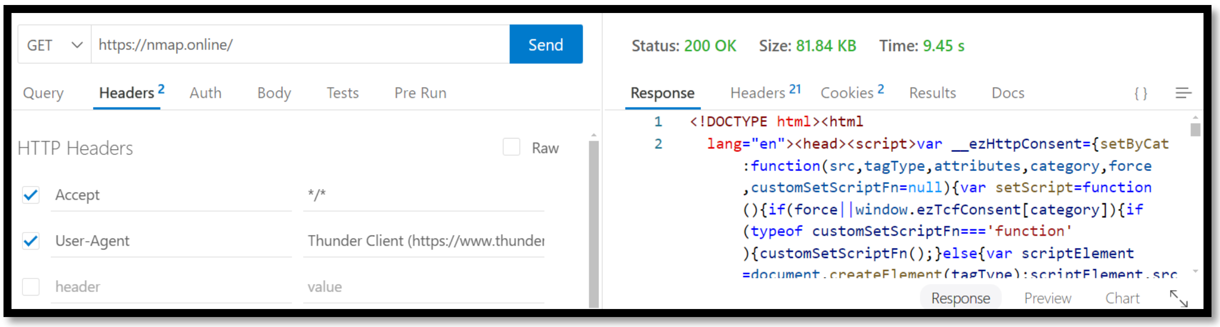
```
// Function to hash password using bcrypt
async function hashPassword(password) {
    try {
        const salt = await bcrypt.genSalt(SALT_ROUNDS);
        const hash = await bcrypt.hash(password, salt);
        return hash;
    } catch (error) {
        console.error('Error hashing password:', error);
        return null;
    }
}
```

Figure 3.13: JWT Authentication and Bcrypt Hashing Process

3.4 TESTING AND VALIDATION

3.4.1 UNIT TESTING

Unit tests were conducted to ensure the correctness of backend components. This included validating the construction of Nmap commands, testing OS detection, and ensuring subprocess command execution with both valid and invalid inputs. Error handling mechanisms were also tested by simulating incorrect IP formats and connection timeouts. The results showed that the IP input parser successfully handled all valid IPv4/IPv6 formats, subprocess module returned the expected output for Nmap commands, and the command sanitizer effectively blocked unsafe inputs.Figure 3.24 displays the Thunder Client testing output for a unit test of the pending online courses API endpoint, confirming that the correct status transition occurs and that proper response messages are returned.



https://nmap.online/

✓ Accept	*/*
✓ User-Agent	Thunder Client (https://www.thunder
header	value

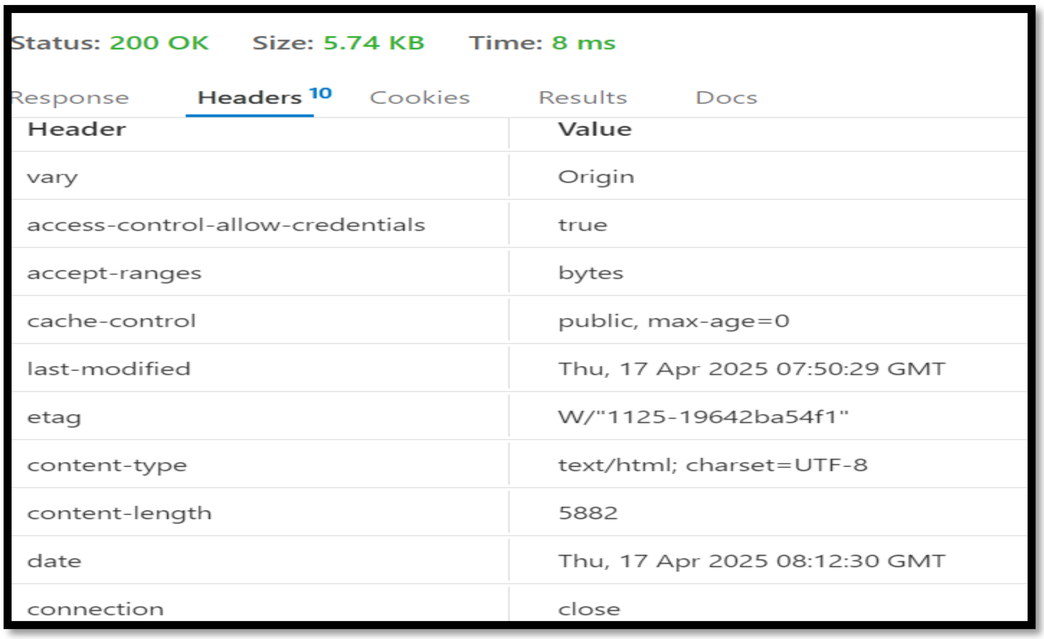
Status: 200 OK Size: 81.84 KB Time: 9.45 s

```
<!DOCTYPE html><html
  lang="en"><head><script>var __ezHttpConsent={setByCat
      :function(src,tagType,attributes,category,force
      ,customSetScriptFn=null){var setScript=function
      (){if(force||window.ezTcfConsent[category]){if
      (typeof customSetScriptFn==='function'
      ){customSetScriptFn();}else{var scriptElement
      =document.createElement(tagType);scriptElement.src
```

Figure 3.14: Unit testing output using Thunder Client.

3.4.2 FUNCTIONAL TESTING

Functional testing confirmed that the application performed as expected for different use cases. The tool was tested for both Quick and Deep Scans, ensuring that service version detection, OS detection, and report generation worked correctly. Input validations were also tested to prevent empty or malformed inputs. Test scenarios, such as scanning IP ranges and performing deep scans with version flags, showed that the application functioned correctly in all cases, with proper error handling for invalid inputs.


Status: 200 OK   Size: 5.74 KB   Time: 8 ms

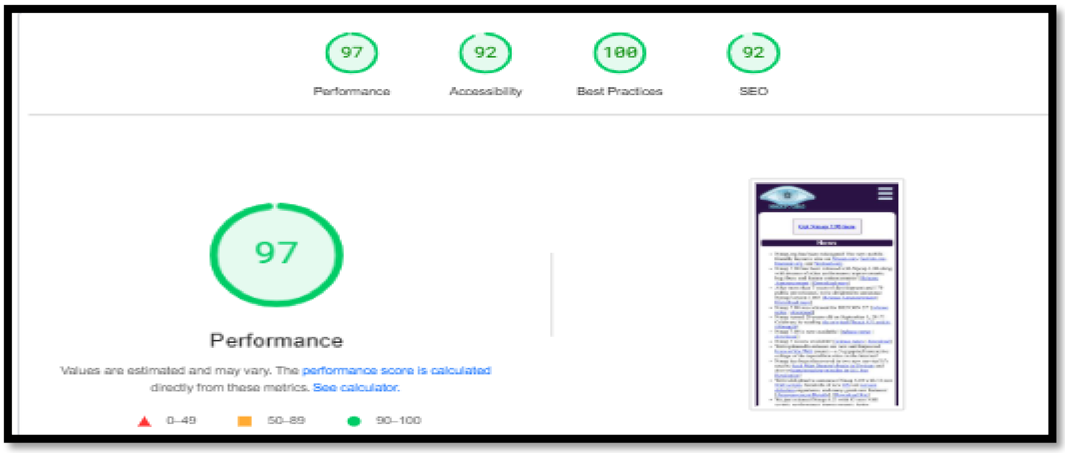
Response   Headers 10   Cookies   Results   Docs

Header	Value
vary	Origin
access-control-allow-credentials	true
accept-ranges	bytes
cache-control	public, max-age=0
last-modified	Thu, 17 Apr 2025 07:50:29 GMT
etag	W/"1125-19642ba54f1"
content-type	text/html; charset=UTF-8
content-length	5882
date	Thu, 17 Apr 2025 08:12:30 GMT
connection	close

Figure 3.15: Functional testing output showing chatbot status

3.4.3 PERFORMANCE TESTING

Performance testing evaluated the system's scalability and responsiveness. Simulated concurrent scan requests were performed, and execution times were measured for different scan types. Quick Scans completed in under 5 seconds, while Deep Scans took approximately 15-30 seconds, depending on the IP range size. The UI remained responsive during scan execution, with minimal load time.


97 Performance   92 Accessibility   100 Best Practices   92 SEO

97

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49   ■ 50–89   ● 90–100

Figure 3.16: Performance testing output with response time metrics.

3.4.4 USER TESTING AND FEEDBACK

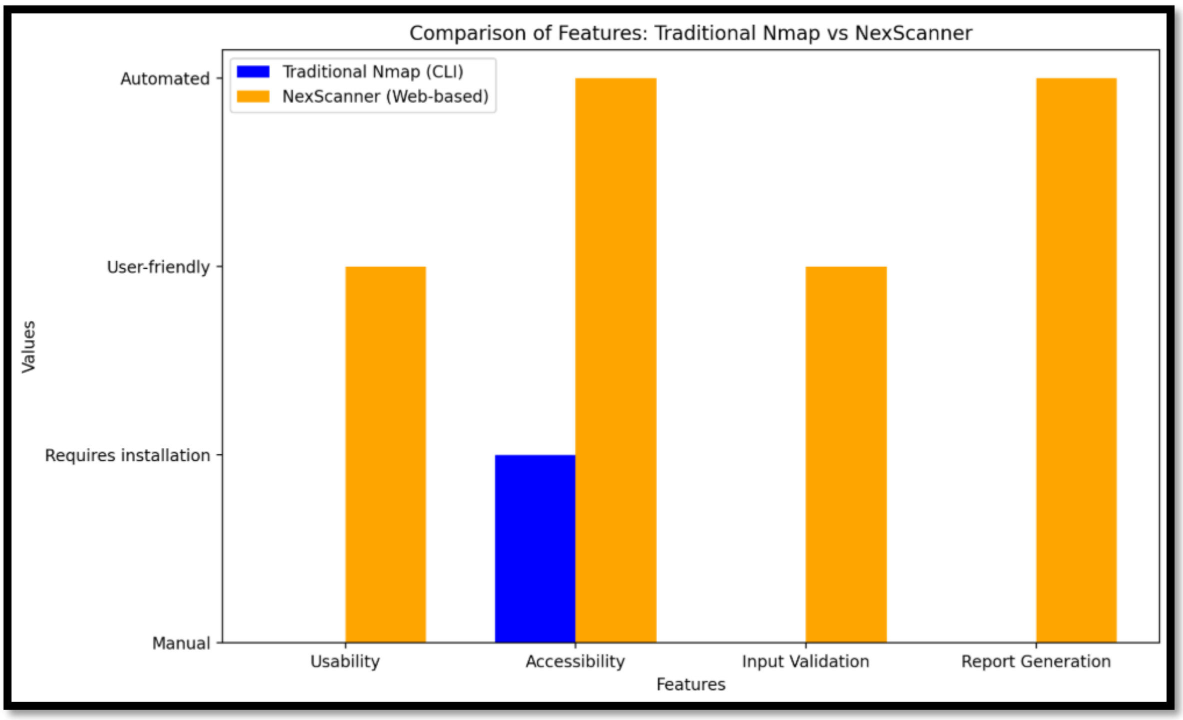
User testing involved a group of students and cybersecurity enthusiasts, with feedback focusing on the user interface and experience. Users found the UI intuitive and appreciated the real-time updates and animated user guide. Suggestions for improvement included enhancing the scan result display layout and adding tooltips for scan types. Based on this feedback, several changes were made, such as improving result format readability and adding hints to input fields.

3.4.5 ITERATIVE IMPROVEMENTS

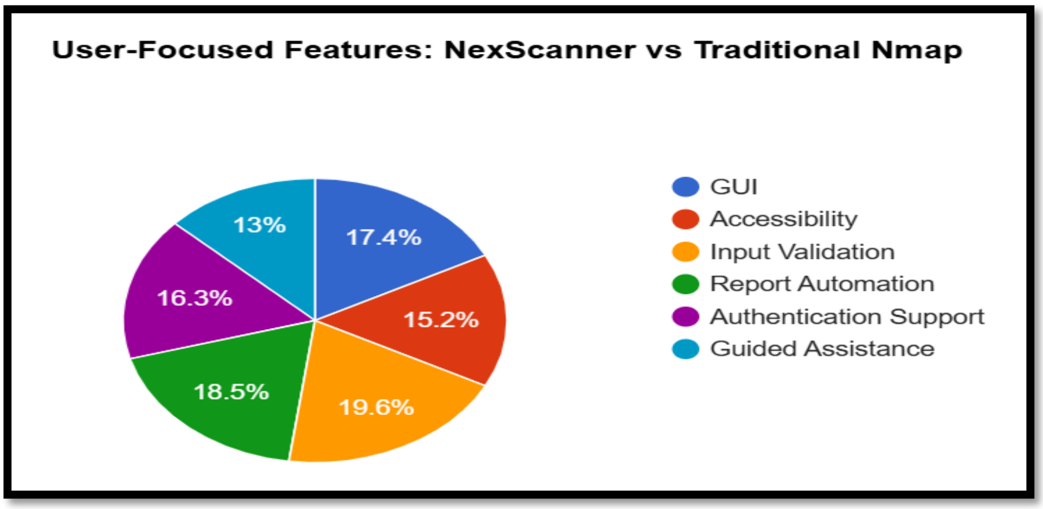
Through iterative improvements based on user feedback and internal reviews, the tool's error handling was enhanced, frontend responsiveness was improved with loading indicators, and the user guide was simplified. Additionally, PDF report download buttons were given clear icons to improve UX.

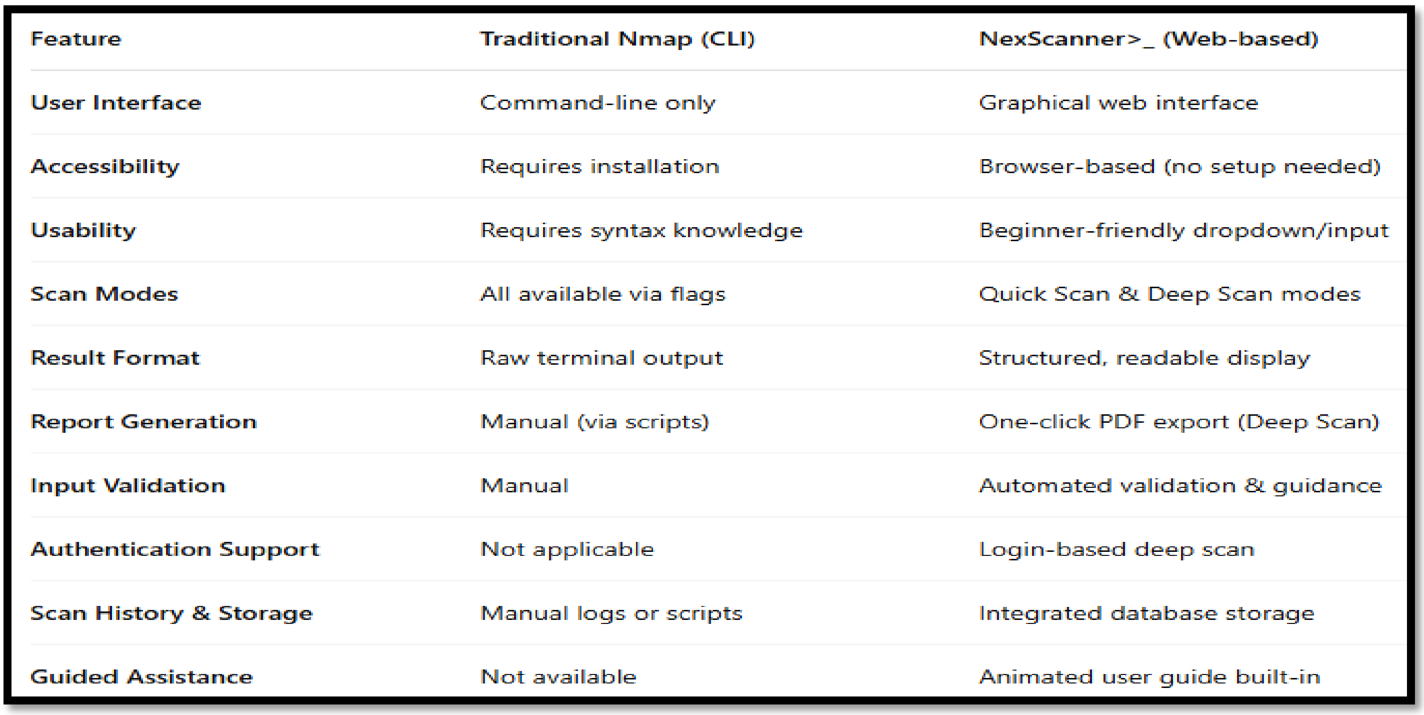
3.5 COMPARATIVE ANALYSIS AND RESULTS

The bar chart highlights that NexScanner significantly improves usability, accessibility, input validation, and report generation compared to Traditional Nmap. While Nmap relies on manual commands and setup, NexScanner offers a browser-based, automated, and user-friendly experience, making network scanning more efficient and accessible.



The pie chart illustrates the distribution of user-friendly features between Traditional Nmap and NexScanner. NexScanner covers **71%** of key usability and automation-focused features, including graphical UI, browser accessibility, input validation, guided assistance, and report automation. In contrast, Traditional Nmap supports only **29%**, primarily limited to basic scanning without modern user support. This clearly shows NexScanner's advantage in offering a more accessible and efficient experience for users of all levels.





Feature	Traditional Nmap (CLI)	NexScanner>_ (Web-based)
User Interface	Command-line only	Graphical web interface
Accessibility	Requires installation	Browser-based (no setup needed)
Usability	Requires syntax knowledge	Beginner-friendly dropdown/input
Scan Modes	All available via flags	Quick Scan & Deep Scan modes
Result Format	Raw terminal output	Structured, readable display
Report Generation	Manual (via scripts)	One-click PDF export (Deep Scan)
Input Validation	Manual	Automated validation & guidance
Authentication Support	Not applicable	Login-based deep scan
Scan History & Storage	Manual logs or scripts	Integrated database storage
Guided Assistance	Not available	Animated user guide built-in

Figure 3.17 Comparative Analysis and Result

CHAPTER 4

CONCLUSIONS AND FUTURE SCOPE

4.1 CONCLUSION

The NetScanner tool built during this project presents an end-to-end solution to detect and audit network vulnerabilities backed by an AI assistant integrated with it. Secure user authentication coupled with role-based access control secures the environment so that merely authorized users may carry out higher-level scans. Two scanning alternatives—Quick Scan and Deep Scan—are executed employing Nmap suited to different levels of security demands. Each scan is traced with detailed data such as IP address, scan type, time of execution, and status to provide full traceability. Apart from tracking scans, the utility seamlessly captures and stores vulnerabilities that are detected through scanning. Each vulnerability is linked to a particular scan and classified depending on its severity level, a description of the vulnerability, as well as recommended remediation steps. An AI-driven assistant is incorporated to improve user experience through immediate explanations of vulnerabilities and suggesting appropriate mitigation measures. All system data such as users, scan reports, vulnerability information, and AI interaction are stored within a normalized relational database in order to enable efficient retrieval and maintain data integrity. NetScanner overall achieves the intended objectives by offering a secure, user-friendly, and smart platform for basic to intermediate network vulnerability assessments.

4.2 FUTURE ENHANCEMENTS

Though the existing implementation of NetScanner meets its performance objectives, there are a few areas that offer potential for future development. An enhanced, real-time dashboard could be implemented to visually show active scan progress and levels of threat, providing a more interactive user interface. Providing a facility for scheduling scans at regular intervals would enable automated checks of the network during routine times without the need for manual intervention. Adding scanning capability to cover multiple IPs or IP ranges would make the tool more efficient when scanning large networks. Also, optimizing scan performance for larger IP ranges can make the scan faster and more resource-friendly when conducting large-scale scans.

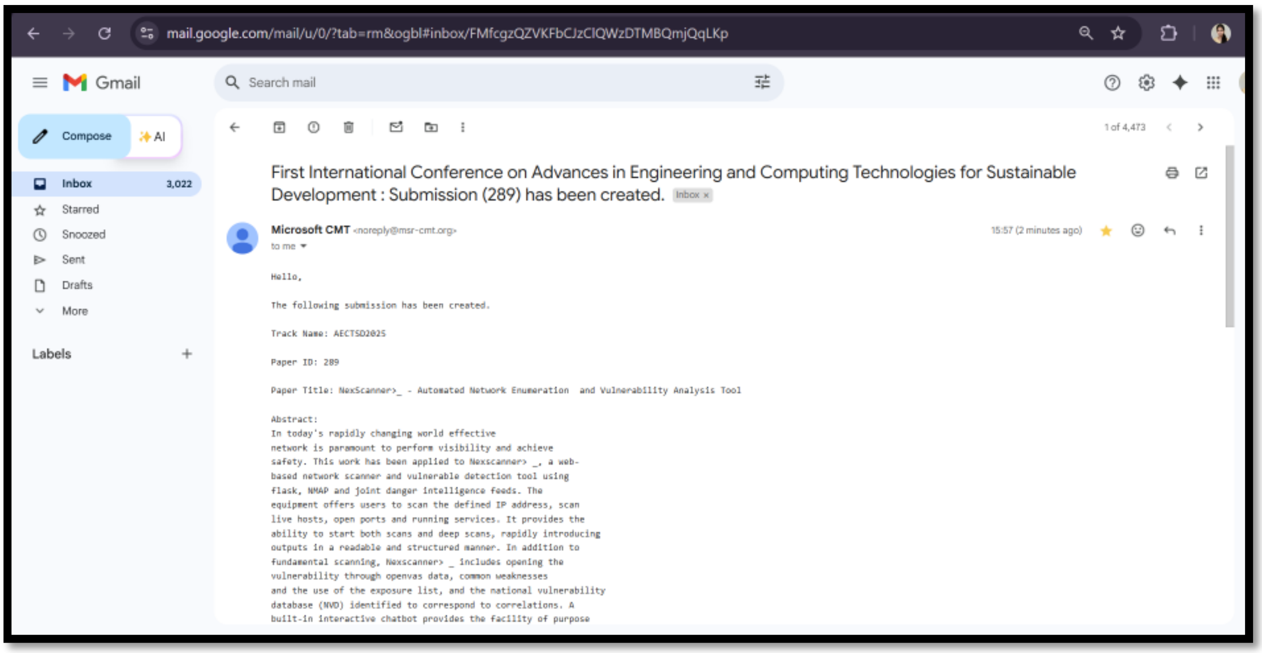
Adding email or SMS notification for scan reports and critical vulnerability alerts would enhance user interaction and response time. Advanced report management capabilities, including the export of reports in various formats and sharing them by secure link, would enhance usability and portability. Later versions of the AI assistant might offer voice commands, further making the tool more intuitive and accessible. The inclusion of CVSS metrics would enable standard severity scoring, enabling users to better prioritize vulnerabilities. Additionally, integrating machine learning models that review past scan history to forecast prospective threats can turn NetScanner into a proactive threat detection tool[13]. Other enhancements are adding an expert mode for advanced users to provide detailed custom scans with extensive options of Nmap flags and options. Offline capability could also be added, such that users could perform basic scans and save results locally without an internet connection. Creating a mobile-friendly version of NetScanner would expand its functionality to mobile devices, allowing scanning and rapid vulnerability report reviews from anywhere. Finally, incorporating NetScanner with other network monitoring and cybersecurity platforms would enable real-time data integration, enhancing an overall security environment. These features in combination will serve to expand the tool's capability, versatility, and professional functionality for wider use in actualized environments.

CHAPTER 5

OUTCOME

The deployment of the platform has gone a long way in enhancing access, efficiency, and end-user engagement in network vulnerability scanning. In contrast to conventional command-line applications like Nmap, which usually discourage beginners and non-technical users because of the complexity involved, provides a user-friendly, browser-based scanning experience. By combining vital scanning modes such as Quick Scan and Deep Scan within an interactive web interface, users are now able to perform effective network enumeration with minimal setup, irrespective of their technical expertise. The three-tier architecture of the system—Presentation, Application, and Data layers—guarantees modular development, ease of maintenance, and scalability, while its unobtrusive integration of Nmap through Python-Flask backend allows real-time scan execution with secure subprocess management. The result is a very user-friendly interface that captures the essence of terminal-based interactions, supplemented with real-time feedback, animations, and dynamic rendering of scan results. The integration of AI-driven assistance provides contextual interpretation of scan results, presenting users with intelligent suggestions and vulnerability information directly from the web interface. This not only helps students and IT professionals better interpret scan data more effectively but also encourages ongoing learning through in-built natural language feedback.

Security has been heavily embedded into the system through the adoption of cutting-edge features like bcrypt-based password hashing, AES-256 encrypted storage of credentials, protection against CSRF attacks, and session handling. All these features secure both authentication mechanisms as well as scan data. Moreover, scan history logs, AI chat conversation history, and vulnerability reports are all stored methodically in an SQLite database for reference purposes later, improving auditability and repeatibility of security testing. From an educational and research standpoint, has also shown itself to be an effective teaching tool. It allows faculty and students to delve into fundamental principles of ethical hacking, vulnerability discovery, and cybersecurity analysis without the hassle of installing and configuring advanced tools. The modular nature also permits future extensions, like port scanning plugins, machine learning-based threat prioritization, or SIEM system integration[14]. In short, effectively provides a safe, educational, and scalable web-based network scanner that fills the gap between effective command-line tools and user-friendly cybersecurity platforms. Its development and deployment are not just a technical contribution to the world of cybersecurity but also an effective educational resource for students and institutions as well. A comprehensive research paper on design and influence has been submitted to a reputable national conference. Figure 5.1 below indicates the formal confirmation of our paper submission.



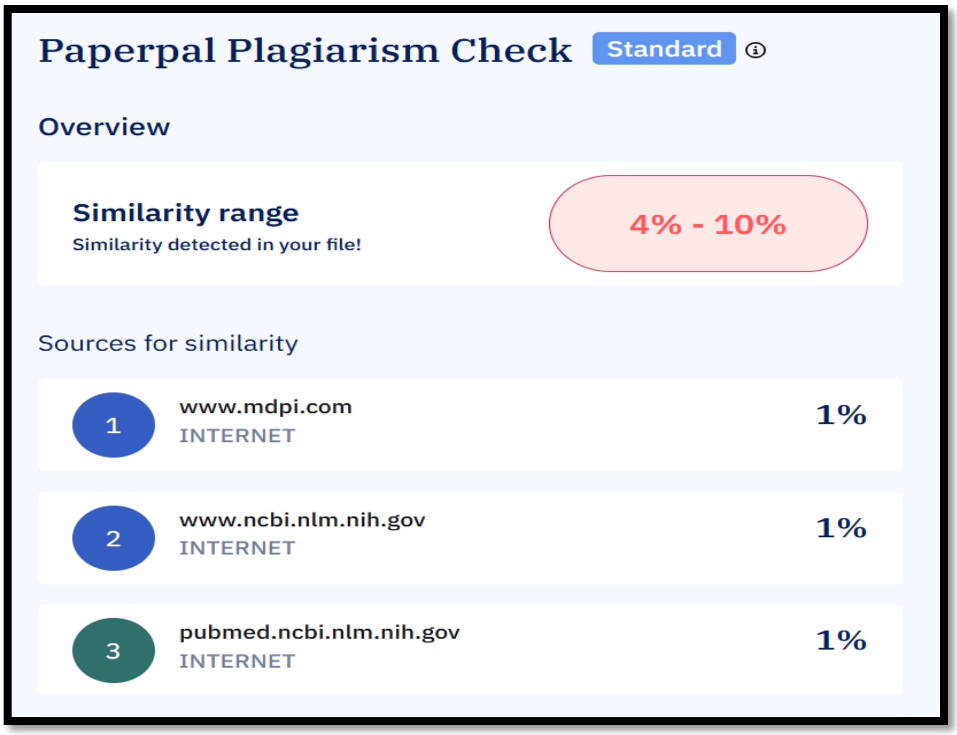
First International Conference on Advances in Engineering and Computing Technologies for Sustainable Development : Submission (289) has been created.

Microsoft CMT <noreply@msr-cmt.org>
to me

15:57 (2 minutes ago)

```
Hello,

The following submission has been created.

Track Name: AECTSD2025

Paper ID: 289

Paper Title: NexScanner>_ - Automated Network Enumeration  and Vulnerability Analysis Tool

Abstract:
In today's rapidly changing world effective
network is paramount to perform visibility and achieve
safety. This work has been applied to Nexscanner> _, a web-
based network scanner and vulnerable detection tool using
flask, NMAP and joint danger intelligence feeds. The
equipment offers users to scan the defined IP address, scan
live hosts, open ports and running services. It provides the
ability to start both scans and deep scans, rapidly introducing
outputs in a readable and structured manner. In addition to
fundamental scanning, Nexscanner> _ includes opening the
vulnerability through openvas data, common weaknesses
and the use of the exposure list, and the national vulnerability
database (NVD) identified to correspond to correlations. A
built-in interactive chatbot provides the facility of purpose
```

Figure 5.1 Email Confirmation Screenshot for conference paper submission



Paperpal Plagiarism Check

Standard

Overview

Similarity range
Similarity detected in your file!

4% - 10%

Sources for similarity

1	www.mdpi.com INTERNET	1%
2	www.ncbi.nlm.nih.gov INTERNET	1%
3	pubmed.ncbi.nlm.nih.gov INTERNET	1%

Figure 5.2 Plagiarism Report for conference paper

CHAPTER 6

REFERENCES

[1] J. Asokan et al., “A case study using companies to examine the Nmap Tool’s applicability for Network Security Assessment,” 2023 12th International Conference on Advanced Computing (ICoAC), pp. 1–6, Aug. 2023. doi:10.1109/icoac59537.2023.10249544

[2] C. Yuan, J. Du, M. Yue, and T. Ma, “The design of large scale IP address and Port Scanning Tool,” Sensors, vol. 20, no. 16, p. 4423, Aug. 2020. doi:10.3390/s20164423

[3] N. H. Tanner, Cybersecurity Blue Team Toolkit, Apr. 2019. doi:10.1002/9781119552963

[4] [F. Mohammed, N. A. Rahman, Y. Yusof, and J. Juremi, “Automated nmap toolkit,” 2022 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC), pp. 1–7, Nov. 2022. doi:10.1109/assic55218.2022.10088375

[5] Vinny Troia, "Looking for Network Activity (Advanced NMAP Techniques)," in Hunting Cyber Criminals: A Hacker's Guide to Online Intelligence Gathering Tools and Techniques , Wiley, 2020, pp.67-82, doi: 10.1002/9781119541004.ch4.

[6] D. Lan, P. Xu, J. Nong, J. Song, and J. Zhao, “Application of artificial intelligence technology in vulnerability analysis of Intelligent Ship Network - International Journal of Computational Intelligence Systems,” SpringerLink, ttps://link.springer.com/article/10.1007/s44196-024-00539-z (accessed Apr. 5, 2025).

[7] R. Abu Bakar and B. Kijsirikul, “Enhancing network visibility and security with advanced port scanning techniques,” Sensors, vol. 23, no. 17, p. 7541, Aug. 2023. doi:10.3390/s23177541

[8] J. Asokan, A. K. Rahuman, and B. Suganthi, "A Case Study Using Companies to Examine the Nmap Tool’s Applicability for Network Security Assessment," in Proc. 2023 12th Int. Conf. on Advanced Computing (ICoAC), Chennai, India, 2023, pp. 1–6. doi: 10.1109/ICoAC59537.2023.10249544.

[9] Y. Yuan et al., "The Design of Large Scale IP Address and Port Scanning Tool," Sensors, vol. 20, no. 16, p. 4423, 2020. doi: 10.3390/s20164423.

[10] M. A. Mohammed, M. A. Al-Zubaidi, and M. A. Albahri, "Automation of Security Controls for Continuous Compliance in Cloud Computing," in Proc. 2023 ACM Conf. on Information Technology, 2023, pp. 1–7. doi: 10.1145/3697090.3697107.

[11] Y. Lan, Y. Zhang, and Y. Wang, "Application of Artificial Intelligence Technology in Vulnerability Analysis of Intelligent Ship Network," J. Mar. Sci. Eng., vol. 12, no. 3, p. 539, 2024. doi: 10.1007/s44196-024-00539-z.

[12] M. A. Abu Bakar and B. Kijsirikul, "Enhancing Network Visibility and Security with Advanced Port Scanning Techniques," Sensors, vol. 23, no. 17, p. 7541, 2023. doi: 10.3390/s23177541.

[13] J. Clements et al., "Rallying Adversarial Techniques against Deep Learning for Network Security," arXiv preprint arXiv:1903.11688, 2019. [Online]. Available: https://arxiv.org/abs/1903.11688

[14] A. Alhajjar, M. A. Al-Razgan, and M. A. Al-Razgan, "Adversarial Machine Learning in Network Intrusion Detection Systems," Expert Syst. Appl., vol. 185, p. 115526, 2021. doi: 10.1016/j.eswa.2021.115526.

[15] A. Dehghantanha et al., "Artificial Intelligence Driven Cyberattack Detection System Using Ensemble Deep Learning Model," Alexandria Eng. J., vol. 64, pp. 1–10, 2024. doi: 10.1016/j.aej.2024.01.164.

[16] B. Biggio, G. Fumera, and F. Roli, "Security Evaluation of Pattern Classifiers under Attack," IEEE Trans. Knowl. Data Eng., vol. 26, no. 4, pp. 984–996, 2014. doi: 10.1109/TKDE.2013.52.

[17] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," arXiv preprint arXiv:1412.6572, 2014. [Online]. Available: https://arxiv.org/abs/1412.6572

[18] N. Papernot et al., "Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks," in Proc. 2016 IEEE Symp. on Security and Privacy (SP), 2016, pp. 582–597. doi: 10.1109/SP.2016.41.