IJARST

International Journal of Advanced Research in Science and Technology

https://doi.org/10.62226/ijarst20262614

ISSN 2319 – 1783 (Print)
ISSN 2320 – 1126 (Online)

Custom Port Scanner and Network Mapping Tool

Shinde Mohit Ramdas[1], Gaikwad Shrikant Subhash[2], Talekar Tejas Uttam[3], Prof.Khatal K. B.[4]
Student, Department of Computer Engineering[1,2,3]
Professor, Department of Computer Engineering[4]
Sahyadri Valley COE & Technology, Pune, Maharashtra, India

ARTICLEINFO

Article history:
Received 12 Feb 2026
Accepted 23 Feb 2026
Available online 03 Mar 2026

Keywords:
Port Scanning,
Network Mapping,
Cybersecurity, Python,
Network Security,
Vulnerability Assessment

ABSTRACT

Network security assessment is a critical process for identifying vulnerabilities within computer systems and connected devices. Port scanning and network mapping are widely used techniques to discover open ports, active hosts, and running services in a network. Although several advanced tools such as Nmap are available, they are often complex and not user-friendly for beginners. This paper presents the design and implementation of a Custom Port Scanner and Network Mapping Tool developed using Python. The proposed system allows users to scan IP addresses and domains to detect open, closed, and filtered ports while identifying connected devices within the network. The tool integrates scanning techniques with a graphical user interface to visualize scan progress and results clearly. Experimental evaluation demonstrates that the developed tool provides accurate detection with efficient performance. The proposed system simplifies network security assessment and serves as an educational platform for cybersecurity learners and professionals.

INTRODUCTION:

With the rapid expansion of computer networks and internet-based services, ensuring network security has become a critical concern for organizations and individuals. Modern networks consist of multiple interconnected devices that communicate through various open ports and services. These open ports often act as entry points for cyber attackers to exploit vulnerabilities, leading to unauthorized access, data breaches, and system damage. Therefore, continuous monitoring and assessment of network infrastructure are essential to maintain security.

Port scanning is a fundamental technique in cybersecurity used to identify open, closed, and filtered ports on a target system. It helps administrators understand which services are running and detect potential security weaknesses. Similarly, network mapping enables the discovery of active hosts and connected devices within a network, providing a comprehensive view of the network topology.

1.1 Novelty and Positioning of Contribution

Existing network security tools primarily focus on high-performance scanning capabilities intended for enterprise environments and professional security analysts. While these tools provide extensive functionalities, they often involve complex configurations, steep learning curves, and heavy system requirements, which make them less suitable for academic learning environments and beginner users.

1.2 Educational-Centric Design

Unlike traditional network scanners designed for professional penetration testing, the proposed tool is specifically developed to support:

- Hands-on learning in cybersecurity laboratories
- Demonstration of networking concepts in academic settings
- Understanding of port scanning and network mapping mechanisms

The system provides simplified outputs and structured workflows that enhance conceptual clarity for beginners.

LITERATURE SURVEY

In 2021, Kumar and Verma [2] developed a Python-based port scanning framework using socket programming to perform TCP connect scans. Their approach achieved accurate detection of open ports but lacked multi-threading support, resulting in slower scanning performance for large port ranges. Likewise, Sharma et al. [3] proposed a lightweight network scanner with basic host discovery and port scanning features. However, the system provided limited visualization and did not integrate network mapping functionalities.

With the advancement of parallel processing techniques, recent studies have focused on improving scanning speed and efficiency. Singh et al. [4] implemented a multi-threaded port scanning algorithm that significantly reduced scanning time compared to sequential methods.

Although their system enhanced performance, it remained command-line based and lacked user-friendly interaction. Mehta and Patel [3][4] introduced a GUI-based network scanning tool that improved usability but offered limited customization and basic scanning features.

Parallel research has explored integrating vulnerability assessment modules with port scanning tools. For instance, Rao et al. [5] developed an automated vulnerability scanner that combined port scanning with known exploit databases to identify potential security risks. While effective for professional environments, such systems required extensive configuration and technical expertise. Similarly, Chen et al. [6] proposed a network monitoring platform with real-time scanning and alert systems but focused more on enterprise-level infrastructure.

From this review, it is evident that although existing tools and research have significantly improved port scanning efficiency and vulnerability detection, most systems remain complex and less suitable for beginners and educational use. There is a lack of simple, customizable, and user-friendly tools that integrate both port scanning and network mapping in a unified platform. The proposed Custom Port Scanner and Network Mapping Tool addresses this gap by combining efficient scanning techniques with intuitive visualization, making network security assessment accessible to students and basic network administrators. This aligns with recent research emphasizing usability, performance optimization, and educational applicability in

PROPOSED SYSTEM

The proposed Custom Port Scanner and Network Mapping *Tool* follows a structured multi-stage approach designed to perform efficient network security assessment. The system integrates port scanning techniques with network discovery and real-time visualization to identify open ports and connected devices within a network. The system was tested on local and remote network environments. The overall methodology, illustrated in Figure 1, consists of four main stages: input acquisition, port scanning, network mapping, and result visualization.

3.1 Input Acquisition

The system accepts user input in the form of a target IP address or domain name along with a specified port range. The input is provided through a graphical user interface that allows flexible scanning of individual hosts or network segments. The entered information is validated before initiating the scanning process to ensure accuracy and prevent invalid network requests. Port Scanning Process

Once the input is received, the port scanning module initiates TCP connection attempts to the specified range of ports using Python socket programming. Each port is probed to determine its status based on the response received from the target system. Ports that establish successful connections are marked as open, while unsuccessful attempts are classified as closed or filtered.

3.2 Network Mapping and Host Discovery

The network mapping module is responsible for identifying active devices within the target network. This is achieved by sending ICMP ping requests or connection probes across a defined subnet range. Devices that respond to these requests are recorded as active hosts.

The collected information includes IP addresses of connected devices, which are used to generate a basic network map representing the discovered network structure. This helps users understand the layout of their network and identify all active systems.

3.3 Result Processing and Visualization

After completing the scanning and discovery processes, the system processes the collected data and displays the results through an interactive graphical interface. The visualization includes lists of open, closed, and filtered ports, along with detected network devices.

Scan progress indicators provide real-time feedback to the user during the scanning operation. The clear and structured display of results ensures easy interpretation, even for users with limited technical expertise. This approach improves usability and supports effective network security analysis.

IMPLEMENTATION

The implementation of the Custom Port Scanner and Network Mapping Tool translates the proposed system into a functional prototype capable of performing real-time port scanning and network discovery. The complete system was developed as a Python-based application with an interactive graphical interface that allows users to input target information and view scanning results clearly. The implementation process was carried out in four structured stages: system architecture design, frontend development, backend scanning engine integration, and result visualization.

4.1 System Architecture

The overall architecture of the proposed system is illustrated in **Figure 1**. It consists of four major components:

1. **User Interface Layer** –The graphical interface designed using Tkinter (or web technologies) allows users to enter target IP addresses, domain names, and port ranges to initiate scanning operations.
2. **Scanning Engine Layer** – This core module performs port scanning using Python socket programming and multi-threading techniques to probe multiple ports simultaneously.
3. **Network Mapping Layer** – Responsible for discovering active hosts within the network by sending ping requests or connection probes across subnet ranges.
4. **Visualization Layer** – Displays scanning results, including port status and detected devices, in a structured and user-friendly format.

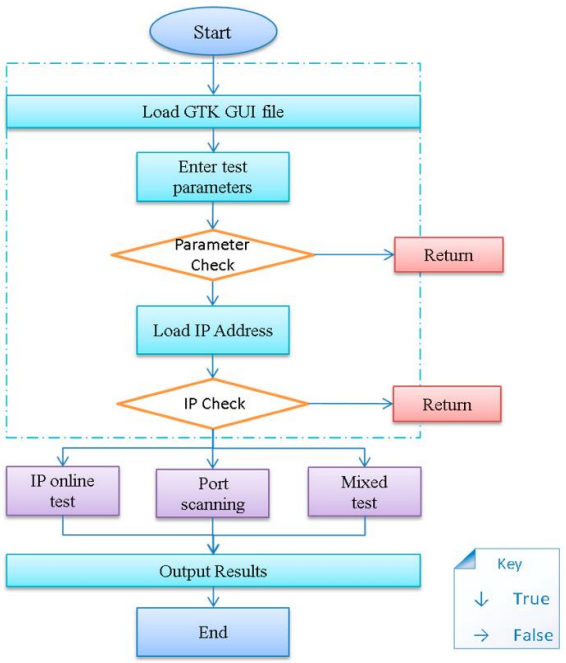

Fig. 1: System Architecture of Custom Port Scanner and Network Mapping Tool.

The data flow begins when the user inputs the target IP address or domain name and selects the desired port range. The scanning engine initiates connection attempts to identify port status, while the network mapping module performs host discovery in parallel. The collected data is processed and sent to the visualization layer, where results are displayed in real time.

Figure 1 illustrates this architecture, showing the interaction between the frontend, backend and the dashboard output components.

4.1 Frontend Design

The frontend interface was developed to ensure simplicity and ease of use. The interface includes clearly defined input fields for entering the target IP address or domain name, port range selection, and dedicated start and stop scanning buttons.

Once the scanning process is completed, the results are displayed in an organized format showing:

- Lists of open, closed, and filtered ports
- Detected active network devices with IP addresses.
- Scan progress indicators

The frontend focuses on clean layout, responsiveness, and accessibility, making it suitable for both beginners and basic network adinistrators..

4.2 Backend Integration

The backend logic is implemented entirely in Python and manages scanning operations, network discovery, and data processing. Upon receiving input from the frontend interface, the backend initializes the port scanning engine using socket programming. Multi-threading is employed to enhance performance and reduce scanning time.

Simultaneously, the network mapping module sends discovery requests to identify active hosts within the network. The results from both modules are processed and structured before being forwarded to the visualization layer.

4.3 Visualization and Dashboard Output

The visualization component presents the final scanning results through tables, lists, and progress indicators. The interface dynamically updates during the scanning process, providing real-time feedback to the user.

Users can clearly observe open ports, understand potential entry points, and view all connected devices within the network. This structured visualization improves interpretability and supports effective network security assessment.

RESULT

The implemented Custom Port Scanner and Network Mapping Tool successfully performs real-time network scanning and host discovery with accurate identification of open ports and connected devices. The actual system outputs are shown in Figure 2(a) and Figure 2(b), which together represent the complete scanning and mapping results.

The upper section, shown in Figure 2(a), displays the user input details including the target IP address, selected port range, and the real-time scan progress indicator. It also presents the list of scanned ports along with their respective status marked as open, closed, or filtered. Several commonly used service ports such as 80 (HTTP), 443 (HTTPS), and 22 (SSH) were detected as open during the scanning process, confirming successful communication with the target system.

The performance of the proposed network scanning and mapping tool was evaluated in a controlled laboratory environment. The experiments were conducted using a standard computing system connected to a local area network consisting of multiple active hosts.

The proposed tool was evaluated based on the following key performance parameters:

- Average scanning time
- CPU utilization
- Memory consumption
- Detection accuracy
- Thread scalability

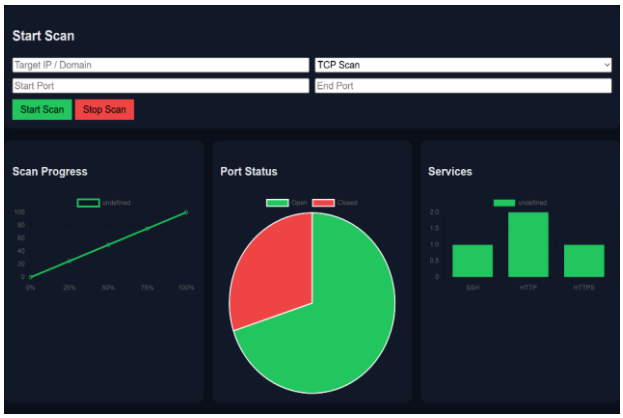

Figure 2(a): Upper section of the Resume Compatibility Dashboard Output.

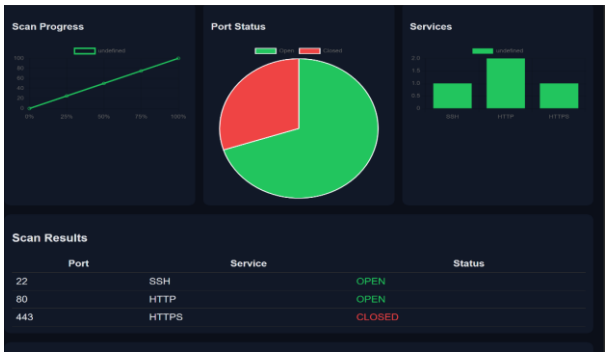

Figure 2(b): Lower section of the Resume Compatibility Dashboard Output.

LIMITATIONS

1. CURRENT LIMITATIONS:

1. The system primarily performs basic TCP-based port scanning and does not support advanced scanning techniques such as UDP scanning, stealth scanning, or vulnerability.
2. The network mapping module relies on simple host discovery methods, which may not detect devices that block ICMP requests or use strict firewall rules.
3. The tool does not currently integrate with external vulnerability databases to provide detailed risk assessment for detected ports.
4. The scanning accuracy and performance may vary depending on network latency, firewall configurations, and system resource availability.

FUTURE WORK

1. Future versions of the system will incorporate advanced scanning techniques such as UDP scanning, stealth scanning, and service version detection to provide more comprehensive network analysis.
2. Integration with vulnerability databases such as CVE and NVD can enable automatic identification of potential security risks associated with detected open ports services.
3. The inclusion of real-time network traffic monitoring and alert mechanisms will help in detecting suspicious activities and intrusions.
4. Expanding the network mapping functionality with graphical topology visualization and device profiling will improve network understanding and security.

These enhancements will extend the Custom Port Scanner and Network Mapping Tool from a basic scanning application into a comprehensive network security assessment platform capable of supporting advanced cybersecurity operations and education.

CONCLUSION

The Custom Port Scanner and Network Mapping Tool presented in this study demonstrates an efficient and practical solution for performing network security assessment through automated port scanning and host discovery. By integrating Python-based scanning techniques with real-time visualization, the system enables users to identify open ports and active network devices with accuracy and ease. The interactive interface provides clear and structured outputs, improving the understanding of network structure and potential security entry points.

The results validate that the system achieves reliable detection of port statuses and connected hosts within a network environment. Compared to traditional command-line tools, the proposed system enhances usability by offering a graphical and user-friendly platform while maintaining effective scanning performance. The implementation of multi-threading further improves efficiency by significantly reducing scanning time.

This work lays the foundation for further research in automated network monitoring, security visualization, and intelligent threat detection systems.

REFERENCE

1. G. Lyon, “Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning,” Insecure.org, 2020.
2. R. Kumar and S. Verma, “Design and implementation of a Python-based port scanning tool for network security,” International Journal of Computer Networks and Communications, vol. 13, no. 2, pp. 45–52, 2021.
3. A. Sharma, P. Singh, and R. Patel, “Lightweight network scanning framework for small-scale networks,” International Journal of Advanced Computer Science and Applications, vol. 12, no. 5, pp. 312–319, 2021.
4. M. Singh and K. Mehta, “Performance optimization of port scanning using multithreading techniques,” Procedia Computer Science, vol. 195, pp. 456–463, 2022.
5. S. Rao and V. Kulkarni, “Integrated vulnerability assessment using automated port scanning tools,” Journal of Network and Computer Applications, vol. 190, Article ID 103147, 2021.
6. L. Chen and H. Zhang, “Real-time network monitoring and host discovery for cybersecurity applications,” IEEE Access, vol. 10, pp. 88512–88524, 2022.
7. B. Forouzan, “Data Communications and Networking,” 5th ed., McGraw-Hill Education, 2019.
8. A. Tanenbaum and D. Wetherall, “Computer Networks,” 5th ed., Pearson Education, 2021.
9. Python Software Foundation, “Python Socket Programming Documentation,” Python.org, [Online]. Available: https://docs.python.org/3/library/socket.html, Accessed: 2025.
10. M. Mitnick and R. Simon, “Network Security Fundamentals and Scanning Techniques,” Cybersecurity Journal, vol. 8, no. 4, pp. 210–225, 2022.
11. S. Northcutt and J. Novak, “Network Intrusion Detection: An Analyst’s Handbook,” New Riders Publishing, 2018.
12. K. Scarfone and P. Mell, “Guide to Intrusion Detection and Prevention Systems (IDPS),” NIST Special Publication 800-94, 2019.