Network Port Monitoring Tool

Jashraj Gandhi | Rekula Prudhvi Raj | Jay Patel | Madhu Talluri

*Abstract*— The proliferation of network-based systems has heightened the importance of robust network monitoring and port management tools. This paper introduces a comprehensive application designed to monitor network traffic, analyze network data, visualize network topology, manage ports, and generate detailed reports. Built using Python, PyQt5 for the graphical interface, and matplotlib for visualization, this tool integrates multiple functionalities into a unified interface. It offers dynamic visualizations, real-time port state management, trace route analysis, and automated report generation.

# **Introduction**.

Effective network management is critical in maintaining secure and efficient computing environments. However, traditional network management tools often lack integrated functionalities, making network administration cumbersome. This paper details the development of a sophisticated yet user-friendly Network Monitoring and Port Scanning Tool. The primary objective is to consolidate critical network management capabilities into a single intuitive graphical user interface (GUI) to enhance usability and effectiveness for network administrators.

# **Related Work**

Network management tools traditionally employ command-line utilities such as Wireshark, Nmap, and netstat for traffic monitoring and port scanning (Combs, 2021; Lyon, 2021). However, graphical tools such as SolarWinds Network Performance Monitor offer richer visual analytics but often at significant cost and complexity. Recently, open-source Python libraries like psutil and matplotlib have enabled the creation of custom network monitoring solutions (Rodola, 2020). Tools like QT-based visual network scanners (Jahic, 2022) have demonstrated the viability of Python and Qt frameworks for network monitoring applications.

Our project builds upon these precedents by integrating multiple network management tasks into one cohesive tool. It specifically leverages PyQt5's robust GUI capabilities (Summerfield, 2016) combined with matplotlib's advanced visualization features (Hunter, 2007).

# **Description**

Frontend

Built with PyQt5, the frontend provides an intuitive user interface segmented into various functional pages:

* Home: Displays real-time network traffic and port statuses.
* Data Analysis: Visualizes network data, including bandwidth usage, top talkers, protocol distribution, packet loss, and connection latency.
* Reports: Generates reports such as daily traffic summaries, bandwidth usage, protocol analysis, security alerts, and peak usage hours.
* Network Scanner: Provides ARP table-based network scans and graphical topology visualization.
* Port Blocker: Manages port blocking and unblocking via firewall rules.
* Trace Visualizer: Visualizes trace routes on a world map using geolocation data.
* Settings: Customizable dark mode interface and adjustable monitoring refresh rate.

Backend

The backend employs Python's psutil for real-time data collection on network connections, port states, and processes. Port activity and state changes are logged and visualized dynamically.

Data Analysis

Leveraging matplotlib, various network metrics are presented in an accessible format, enhancing the visibility of network conditions and anomalies.

Reporting

Automated report generation provides structured HTML reports on various network metrics, improving decision-making and administrative efficiency.

# **Uses**

This Network Monitoring and Port Scanning Tool has multiple practical applications, including:

* Network Security: Identifying and managing vulnerabilities through real-time port monitoring and automated security alerts.
* Performance Optimization: Analyzing bandwidth usage and latency to optimize network performance and resource allocation.
* Troubleshooting: Facilitating rapid diagnosis and resolution of network issues through visual analytics and comprehensive reporting.
* Infrastructure Management: Simplifying the management of network infrastructure by providing an intuitive interface for monitoring and maintenance tasks.
* Educational Purposes: Serving as a learning tool for network administrators and cybersecurity students to understand network operations and threats.
* Compliance Auditing: Assisting in regulatory compliance by generating detailed network usage and security reports.

# **Results**

The application was successfully implemented and tested in real-world scenarios. Key outcomes include:

* Improved visibility and real-time management of network resources.
* Efficient identification and response to network threats and anomalies.
* Simplified administrative tasks through automated visualizations and reporting.
* Enhanced overall network security posture and reduced response time to incidents.
* Increased accuracy and ease in compliance reporting.
* User feedback indicated the tool significantly enhanced administrative productivity and network security management.

# **Future Development**

Future improvements for this tool could include:

* Enhanced scalability to manage larger network infrastructures.
* Integration with cloud-based monitoring services.
* Advanced machine learning models for predictive analytics and automated anomaly detection.
* Support for mobile platforms, allowing remote monitoring and management.
* Integration with existing network management systems for seamless interoperability.

# **Conclusion**

The Network Monitoring and Port Scanning Tool effectively integrates various essential network management tasks into a single intuitive interface. Utilizing Python, PyQt5, and matplotlib, it provides comprehensive network analysis, real-time monitoring, dynamic visualization, and robust reporting capabilities. This tool represents a significant advancement in network management, simplifying administrative tasks and enhancing operational security.

# **References**

* Combs, G. (2021). Wireshark Network Analyzer. Retrieved from <https://www.wireshark.org>
* Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering, 9(3), 90-95.
* Jahic, H. (2022). QT-Based Visual Network Scanning. Journal of Cybersecurity Tools, 3(1), 55-61.
* Lyon, G. F. (2021). Nmap: Network Exploration Tool and Security Scanner. Retrieved from <https://nmap.org>
* Rodola, G. (2020). Psutil: Process and System Utilities. Retrieved from <https://psutil.readthedocs.io>
* Summerfield, M. (2016). Rapid GUI Programming with Python and Qt. Pearson Education.
* <https://www.kentik.com/kentipedia/how-to-evaluate-network-monitoring-tools/>
* M. G. Nagaraja R. R. Chittal K. Kumar "Study of Network Performance Monitoring Tools-SNMP" IJCSNS vol. 7 no. 7 pp. 310 2007.
* IT Management Software & Monitoring Tool| SolarWinds. online Available: <https://www.solarwinds.com/>
* "Network Automation Software| Network Automation Tool| NetBrain. online Available: <https://www.netbraintech.com/>
* Nagios - The Industry Standard In IT Infrastructure Monitoring. online Available: <https://www.nagios.org/>
* PRTG Network Monitor » All-In-One Network Monitoring Software. online Available: <https://www.paessler.com/prtg>