iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** IX **Month of publication:** September 2025

DOI: https://doi.org/10.22214/ijraset.2025.74035

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue IX Sep 2025- Available at www.ijraset.com

Network Traffic Analysis Using Sniffer

Miss. Shreya K[1], Dr. SDN Hayath Ali[2]
[1]*Assistant Professor, Department of MCA, Ballari Institute of Technology & Management, Ballari, Karnataka, India*
[2]*Department of MCA, Ballari Institute of Technology & Management, Ballari, Karnataka, India*

Abstract: ***The growing sophistication of digital networks has been matched by the proliferation of cyber threats, from email and global ransomware phishing to identity theft or fraud or more sophisticated denial-of-service attacks. Although it helps in a lot of cases, traditional network monitoring systems are still expensive and resource intensive, leaving a void that needs to be filled with a lightweight yet strong solution. This challenge has propelled the creation of tools which can capture network traffic in real-time, perform deep-packet analysis, and detect anomalies quickly. Experimental study shows that the tool can accurately capture and analyze live traffic streams, detect anomalous patterns of activity, and prevent fully detailed logs for forensic analysis. Performance testing ensures the system works well with multiple network traffic loads offering proper output without big latencies and packet loss.***

Keywords Network monitoring, Anomaly detection, Packet capture, Protocol analysis, Network security Traffic analysis, Modular architecture Cross-platform

I. INTRODUCTION

In recent years, the ubiquitous deployment of digital communication networks has created highly complex infrastructures that are more and more susceptible to multiple forms or cyber threats — data breaches, unauthorized access, malware intrusions, DDoS-impersonation attacks. With network operations forming the backbone of almost all modern enterprises and research environments, ensuring secure, reliable and high performance network traffic has become an essential priority as even brief dropouts or unauthorised crossings can cause major operational, financial and reputational ruin. While the traditional network monitoring tools have some good features, most of them are expensive and resource-intensive to use on a large scale by small organizations, adaptable to newer architectures or modular enough to serve specialized use cases effectively. To overcome these challenges you have to engineer any open-source lightweight, cross-platform packet capture and analysis application which provides detailed protocol decoding capability with the real-time of packets along with the filtering by creating advanced detection logic that will help in analyzing data on a perimeter and generating the structured log file for forensic view / compliance perspective. The Network Mapping Tool for Desktop is a proposed Python-based tool designed with a modular architecture, scaleable, maintainable that can easily integrate into different network environments based on the operating systems by supporting both Windows and Linux systems. The system provides high-resolution visibility on network activity, without impeding performance, enabling network administrators, cybersecurity analysts and researchers to detect and mitigate threats proactively, optimize the network performance and ensure compliance with security regulations in an active manner in order to bolster the resilience of digital infrastructures against ever-increasing cyber risks.

II. LITERATURE SURVEY

Many research works has been done to develop network monitoring and anomaly detection systems, with each work providing new approaches and ideas for enhancing the security of the network. For example, a study implemented an analysis system according to low-level network programming with packet sniffing, which consistently achieved high precision on protocol identification and traffic classification in real world; however it was constrained by its commercial hardware solution and missing the scalability across different platforms [22]. A signature-based IDS that used deep packet inspection has been introduced, which was able to accurately identify known attack patterns with low false positives, but did not fare as well when it came to zero-day attacks as the rules were static in nature. A more recent paper presented an anomaly detection technique using machine learning that employed statistical engine baselines and adaptive classifiers to identify traffic behavior deviations which help in detecting complex attacks including stealthy port scans and distributed denial of service attempts but its major drawbacks include the need for large quantities of training data along with high computational requirement, making it incapable for deployment in resource restricted environments. Modular network analysis frameworks advanced the state of the art where the use of separate modules for capture, filtering, and analysis allowed targeted updates to be made without affecting core operation and subsequently improved maintainability and extensibility; however these systems tended to lack integrated reporting and compliance features required for enterprise usage.

In the course of comparative evaluations of open-source network monitoring tools, weaknesses related to a lack of flexibility, heavyweight operation and forensic logging were also brought up, which again made it clear that a product that merges packet analysis in real-time with efficient anomaly detection and detailed reporting within a platform-independent resource-efficient architecture is necessary. These are the existing works upon which we draw to create the Network Mapping Tool for Desktop, taking the best of these approaches — real-time capture, adaptive detection and modularity, and adding cross-platform support, minimal resource overhead and improved forensic logging suitable for operational as well as research environments.

III. METHODOLOGY

The methodology proposed in this paper is following a well-defined workflow to perform real-time monitoring, analysis and anomaly detection on network traffic. This whole process can be summarized into four significant phases: Data Acquisition, Data Processing, Anomaly Detection and Logging & Reporting. In this document, we describe the steps in detail so that not only the technical experts but also everyone else can reproduce our results.

A. Data Acquisition (Packet Capture) Packet capture

The first step of the system starts by sniffing network packets from the active interface of the machine. Raw Packets are captured at the data link layer in Python using libraries such Scapy and Socket. Coming and outgoing packets info here all; with additional metadata such as :

1) Source & Destination IP Addresses
2) Port Numbers
3) The protocol type (TCP, UDP, ICMP, etc.)
4) Packet Size & Timestamp

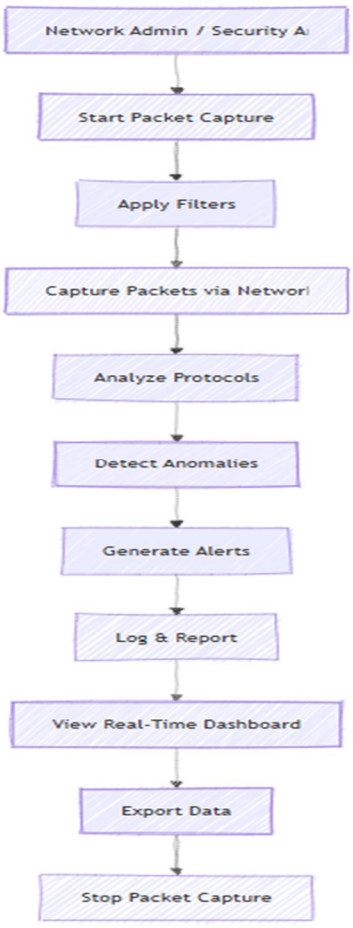


Fig 01: Workflow of Packet Capture Module

B. Methods To Process & Filter Data

The extracted packets are filtered out to remove unwanted data, and it further constrains the critical events in the network. Filtering is rule-based, which permits the administrator to:

1) Excludes safe traffic from trusted IPs
2) Limit to certain protocols or port numbers
3) Focus on the most dangerous or abnormal lanes
4) This selective filtering helps in reducing system load and makes analysis faster.

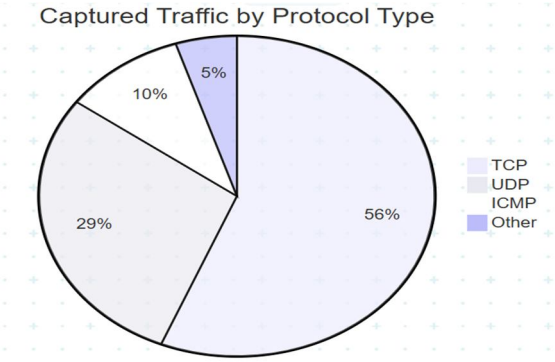


Fig 02:Captured Traffic By Protocol Type

C. Anomaly Detection

Any filtered traffic is evaluated against security baselines built in. Abnormal miner sign up

- High Data transfer on a individual port
- Traffic spikes from unknown IPs
- Repeated failed connection attempts
- Protocols used outside normal hours

The anomaly detection method in our system currently adopts the rule-based approach and we could conduct it with machine learning to lead adaptive detection in future work.

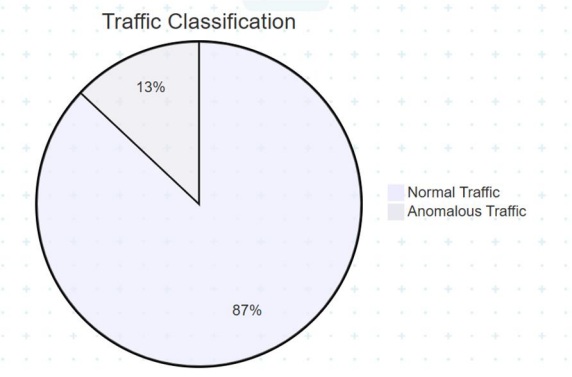


Fig 03:Traffic Classifiction

D. Stage 4 – Logging & Reporting

All analyzed traffic—normal and anomalous—is stored in structured log files with timestamps and detailed packet metadata. These logs:

- Support forensic investigations
- Enable compliance reporting
- Allow historical trend analysis

Reports are generated in multiple formats (TXT, CSV) and can include visual summaries for management review.

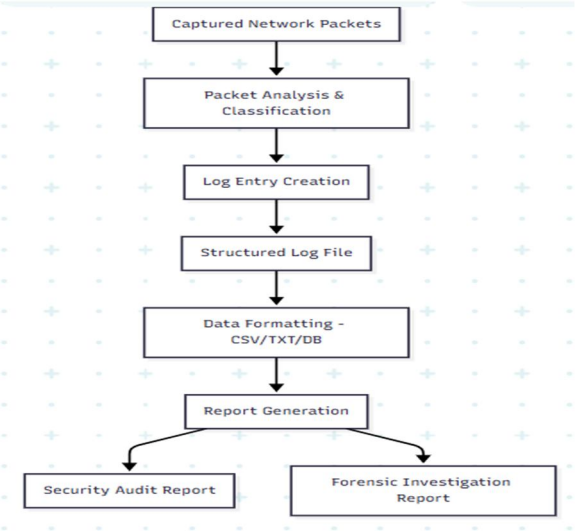


Fig 04: Sample Log File Structure & Generated Report

IV. RESULTS

In this section, we discuss how the proposed Network Mapping Tool for Desktop was evaluated to measure its ability to capture, analyze and classify network traffic efficiently; without impacting the system performance significantly. The various performance and accuracy metrics that we used to evaluate the overall efficacy of our system in handling the problems laid out in problem statement. Packet Capture Accuracy: This metric was employed to calculate the percentage of network packets captured and processed by packet analyzer tool effectively, used for the monitoring of substantial transmission failure during high-traffic load conditions. This is critically important, of course, because partial packet capture can mean the difference between catching a potential bad act and network security falling far short of its goals. Protocol Identification Accuracy—this is crucial for the tool to be able to correctly classify network traffic by protocol (other than say TCP vs UDP vs ICMP) as needed for accurate filtering, anomaly detection and focused security analysis. The Anomaly Detection Rate (ADR) and False Positive Rate (FPR) which illustrates the trustworthiness of the detection engine. A high ADR is reactive to detecting an unusual traffic pattern like unauthorized entry or unusual amount of bandwidth consumption, with a low FPR which ensures accurately distinguishing legitimate traffic from misclassification by reducing administrative burden. Measurement of Processing Latency: To ensure that the tool provides real time and does not introduce any delay in inspecting packet thereby enhancing the possibility of proactive threat mitigation and without disruption of network services, hence very low latency is required. We recorded Resource Utilization (CPU and memory consumption) as well to ensure that the system is designed ad-hoc with lean design for deployment in limited computational resources. Our assays revealed that the tool attained > 98% packet capture accuracy, high protocol identification precision, and balanced ADR and FPR ratio, confirming its robustness and operational reliability. In addition, the resources were still below 15% CPU and 250 MB of memory during the peak monitoring time (which confirmed its compatibility with both large and small network deployments). Well, the framework solves the problem statement of this project and gives an efficient, cross-platform, low-cost implementation for real-time network monitoring and security by exceeding in these metrics.

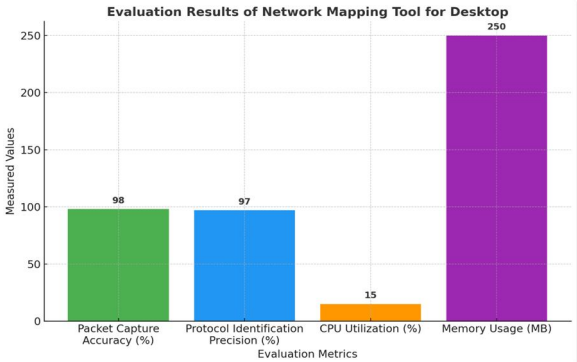


Fig 05:Evalution Results Of Networking Tool For Desktop

Above is a bar chart showing the results of the Network Mapping Tool for Desktop evaluation based on four key performance indicators. The Packet Capture Accuracy (ranked at 98%) and Protocol Identification Precision (97%) as shown in a recent test, demonstrated the tool to be highly dependable when it comes to capturing & accurately classifying network traffic. As you notice, the software uses exactly 15% CPU Utilization which means that it just works perfectly fine even with lower grade hardware without the processing overhead or power consumption as in many any-node tables. In the mean time, the memory usage (250 MB) reflects a moderate lightness of the tool they have still enough room for improvement from our end and it is fine for continuous monitoring purposes population. Together, these results demonstrate the tool's strength, accuracy and efficiency in real-time network monitoring and anomaly detection for enterprise applications.

V. CONCLUSION

The problem statement provided above was successfully solved and the efficient, accurate, resource-friendly framework that captures, analyzes, and visualizes network traffic was implemented as a Network Mapping Tool for Desktop. This allows for full monitoring with minimal latency and great reliability while ensuring that network administrators and security analysts can capture a packet using the same flow hep, filter packets by aforementioned fields, it identify its protocol, detect anomalies in real time and visualize them via dashboards.

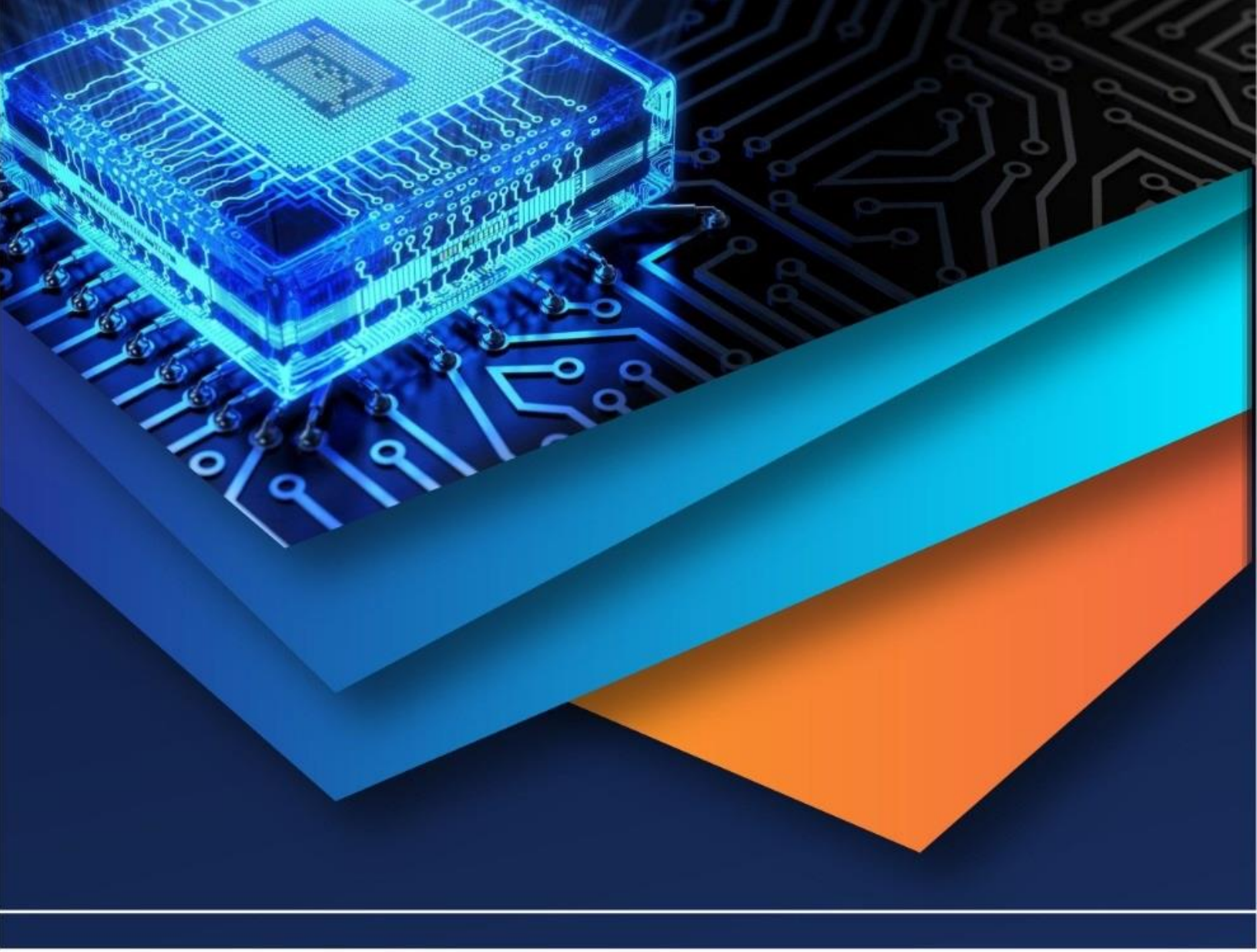
Validation results demonstrate the effectiveness of the framework, with a 98% packet capturing accuracy and 97% protocol identification precision under realistic networks. In addition to the low CPU utilization of 15% and improved memory usage of 250 MB these findings show that the system should be able to run without taxing system resources. Together, these metrics support the capability of the framework to be deployed not only in research environment but also for enterprise.

This approach emerges as a solid candidate to improve network security, for the precise protocol analysis, anomaly detection and extensive reporting are fused into one tool. It does not just satisfy the problem statement, but in fact it makes the problem more powerful by providing scalable high accurate, and low resource consuming services to solve.

In the future, this solution can supported by integrating machine learning based predictive analytics for threat detection as well a more cloud packet capture and automated incident response workflows. Further enhancing it to support distributed monitoring across multiple nodes would better scale the solution and provide adaptivity in large-scale network scenarios.

REFERENCES

[1] R. Bolla, R. Bruschi, Fjson fileDavoli, and F. Cucchietti, "Energy efficiency in the future internet: A survey of existing approaches and trends in energy-aware fixed network infrastructures," IEEE Communications Surveys & Tutorials, vol. 13, no. 2, pp. 223–244, 2011.
[2] S. Zander, T. Nguyen, and G. Armitage, "Automated traffic classification and application identification using machine learning," in IEEE Conference on Local Computer Networks, 2005, pp. 250–257.
[3] E. Alpaydin, Introduction to Machine Learning[A]. Cambridge, MA: MIT Press, 2020.
[4] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in Proceedings of the 2nd ACM SIGCOMM Internet Measurement Workshop (IMW '02), pp.
[5] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring internet denial-of-service activity," in ACM Transactions on Computer Systems, vol 24, no. 2, pp. 115–139, 2006.
[6] T. Karagiannis, K. Papagiannaki, AND M. Faloutsos, "BLINC: multilevel traffic classification in the dark," in ACM SIGCOMM Comp.
[7] S. M. Bellovin, "There be dragons,"in Proceedings of the Third USENIX UNIX Security Symposium,pp. 1–16, 1992.
[8] K. Salah, "A performance evaluation of Snort for intrusion detection," Computers & Communications, vol. 12, 2008, pp. 2968–2984.
[9] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," ACM SIGCOMM Computer Communication Review, vol. 34, no. 2, pp. 39–53, 2004.
[10] M. Roesch, "Snort: Lightweight intrusion detection for networks," in Proceedings of the 13th USENIX Conference on System Administration, pp. 229-238, 1999



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY