**NETWORK PERFORMANCE ANALYZER (NPA) WITH BI ANALYTICS**

**A PROJECT REPORT**

***Submitted by***

**SHABESHVARAN S [211423104603]**

**SASIDHARAN B [211423104594]**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

### COMPUTER SCIENCE AND ENGINEERING

****

**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**APRIL 2024**

**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**BONAFIDE CERTIFICATE**

Certified that this project report **“NETWORK PERFORMANCE ANALYZER (NPA) WITH BI ANALYTICS”** is the Bonafide work of **SHABESHVARAN S (211423104603), SASIDHARAN B**

**(211423104594)** who carried out the project work under my supervision.

|  |  |
| --- | --- |
| **Signature of the HOD with date** | **Signature of the Supervisor with date** |
| **Dr L.JABASHEELA M.E.,Ph.D.,** | **Dr.A.HEMLATHADHEVI M.E.,Ph.D.,** |
| Professor and Head,  Department of Computer Science and Engineering,  Panimalar Engineering College, Chennai- 123 | Associate Professor,  Department of Computer Science and Engineering, Panimalar Engineering College, Chennai- 123 |

Submitted for the Project Viva– Voce examination held on

**INTERNAL EXAMINER EXTERNAL EXAMINER**

## DECLARATION BY THE STUDENT

#### We SHABESHVARAN S [211423104603], SASIDHARAN B

**[211423104738]** hereby declare that this project report titled

### “NETWORK PERFORMANCE ANALYZER (NPA) WITH BI

**ANALYTICS”** under the guidance of **Dr.SASIKUMAR M.E.** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

**SHABESHVARAN S [211423104603]**

**SASIDHARAN B [211423104594]**

**ACKNOWLEDGEMENT**

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.,** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI, Dr.C.SAKTHIKUMAR, M.E., Ph.D., and Dr. SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,Ph.D.,** for providing us

with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.MANI, M.E., Ph.D.,** for his timelyconcern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr.L.JABASHEELA, M.E., Ph.D.,** for the support extended throughout the project.

We would like to thank our Project Coordinator **Dr.K.VALARMATHI, M.E., Ph.D.,** and our Project Guide **Dr.A.HEMLATHADHEVI, M.E., Ph.D.,** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

Our sincere thanks to **Mrs.V.A.PRABHA, SCIENTIST-E, C-DAC, CHENNAI**, for this kind support. Finally, I would like to thank all those who were directly or indirectly helpful in carrying out this research.

**SHABESHVARAN S [211423104603]**

**SASIDHARAN B [211423104594]**

**ABSTRACT**

The growth of internet-based communication and online activities has made networks an essential part of daily operations in schools, colleges, and organizations. However, many of these institutions face challenges such as slow network speed, bandwidth misuse, and cyber threats due to the lack of real-time monitoring and analytics tools. Most existing network monitoring systems are either too complex or expensive for small-scale use, leaving a gap for simple and affordable solutions.

This mini project introduces the Network Performance Analyzer (NPA) — a lightweight and cost-effective tool built using Java and JPcap/WinPcap libraries for packet capture. The system captures essential network traffic details such as IP addresses, ports, and protocols, and stores them in a CSV file. The collected data is then imported into Power BI, where it is visualized through interactive dashboards and charts. These visualizations help administrators easily identify high bandwidth usage, unusual network activity, and potential security threats in a simple and understandable manner.

The NPA tool supports faster decision-making, improves network efficiency, and enhances security awareness. It is particularly suitable for educational institutions and small businesses that require affordable yet powerful monitoring solutions. In the future, the system can be extended with AI-based predictive analytics, cloud storage, and mobile app integration for remote monitoring. The project aligns with SDG 9 (Industry, Innovation, and Infrastructure) and SDG 4 (Quality Education) by promoting digital infrastructure and safe learning environments.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER**  **NO** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | vii |
|  | **LIST OF FIGURES** | x |
| **1.** | **INTRODUCTION** | **1** |
|  | 1.1 Overview | 1 |
|  | 1.2 Problem Definition | 2 |
|  | 1.3 Literature Survey | 3 |
| **2.** | **SYSTEM ANALYSIS** | **5** |
|  | 2.1 Existing System | 5 |
|  | 2.2 Proposed System | 5 |
|  | 2.3Implementation Environment | 6 |
| **3.** | **SYSTEM DESIGN** | **7** |
|  | 3.1 UML Diagrams | 7 |
| **4.** | **SYSTEM ARCHITECTURE** | **10** |
|  | 4.1 Architecture Diagram | 10 |
|  | 4.2 Module Design Specification | 13 |
| **5.** | **SYSTEM IMPLEMENTATION** | **19** |
|  | 5.1 Backend Coding | 19 |

|  |  |  |
| --- | --- | --- |
| **6.** | **PERFORMANCE ANALYSIS** | **27** |
|  | 6.1 Performance Metrics | 27 |
|  | 6.2 Result and Discussion | 30 |
| **7.** | **CONCLUSION** | **31** |
|  | 7.1 Conclusion | 31 |
|  | 7.2 Future enhancement | 32 |
| **8.** | **APPENDICES** | **33** |
|  | A1 SDG goals | 33 |
|  | A2 Sample Screenshots | 34 |
|  | A3 Paper Publication | 37 |
|  | A4 Plagiarism report | 38 |
| **9.** | **REFERENCES** | **39** |

# CHAPTER 1

**INTRODUCTION**

* 1. **OVERVIEW**

The NPA (Network Performance Analyzer ) is a Java-based project designed to analyze and monitor the performance of bank assets, particularly loans, to identify and track non-performing assets in an efficient manner. In traditional banking systems, the detection and management of NPAs are time-consuming due to manual record verification and delayed data processing. This project aims to automate the analysis process, enabling financial institutions to detect potential NPAs early and take necessary actions to reduce financial risks. By integrating data analysis, visualization, and CSV generation, the system provides an accurate overview of the asset health of an organization.

The project initially functions using **static datasets**, where financial and loan-related data are processed to identify which accounts are performing and which have become non-performing based on parameters such as repayment delay, overdue amount, and interest cycle. The analyzed output is generated in a structured CSV format that helps banks or analysts interpret results easily. This approach ensures that the system’s logic and analytical methods are validated before integrating real-time data feeds for more dynamic monitoring.

In the enhanced version, the system aims to transition towards **real-time data analysis**, allowing continuous monitoring of loan accounts as new financial records are updated. This transformation ensures that banks can maintain up-to- date insights into their credit risk positions, improving decision-making efficiency. The NPA Analyzer thus combines simplicity, accuracy, and scalability, making it a useful tool for early risk detection, financial planning, and loan recovery management.

* 1. **PROBLEM DEFINITION**

Today, almost every school, college, and organization depends on the internet for daily activities such as online learning, communication, and data sharing. However, many of them face frequent problems like slow network speed, bandwidth misuse, and cybersecurity threats. Most of these issues remain unnoticed because there is no proper real-time network monitoring and analysis system in place. Without continuous observation, it becomes difficult to identify which devices or users are consuming more bandwidth or causing network delays.

Existing monitoring tools such as packet analyzers and flow analyzers each have their own limitations. Packet-level tools give very detailed results but require high computing power and are difficult for beginners to use. Flow-based tools, on the other hand, are faster but miss important technical details. High-end commercial solutions that balance both methods are expensive and not suitable for small institutions or local businesses with limited budgets.

This creates a clear gap — the need for a simple, low-cost, and effective monitoring system that can capture live network traffic, analyze it automatically, and present the results in an easy-to-understand format. The absence of such tools often leads to reduced network efficiency, undetected cyber threats, and poor performance in digital environments. The Network Performance Analyzer (NPA) aims to solve this problem by offering real-time data capture, analysis, and visualization in a user- friendly way using Java and Power BI.

* 1. **LITERATURE REVIEW**

**Tio Adistiyawan et al. (2024) – “Analysis and Design of Computer Networks to Analyze Traffic Flow by Utilizing a Network Monitoring System”**

This paper focused on improving network performance using a Network Monitoring System (NMS) based on the NDLC methodology. It monitored bandwidth, network traffic, and topology design using LibreNMS, which helped detect and solve real-time traffic issues efficiently.

**Aniket Chatterjee et al. (2025) – “Library of Networks: An Online Tool for Design and Analysis of Network Topologies”**

The authors developed an online tool called Library of Networks (LoN) to analyze, compare, and synthesize different network topologies. It used performance metrics, routing algorithms, and simulation models to optimize communication efficiency and bandwidth utilization in complex networks.

**Ujjwal Biswas et al. (2025) – “Recurrent Neural Network Model to Predict and Monitor Student Performance”**

This study applied Recurrent Neural Networks (RNNs) to predict outcomes based on sequential data patterns. The RNN model achieved 96% accuracy and proved the efficiency of real-time monitoring and predictive analysis, which inspired similar approaches for financial and NPA data forecasting.

**R. Schoeffauer and G. Wunder (2021) – “Model-Predictive Control for Queueing Networks with Varying Topology”**

This research proposed a model-predictive control algorithm for managing traffic in queueing networks with changing topologies. It improved stability and adaptability, providing a strong foundation for designing efficient real- time monitoring systems.

**N. R. Fachrurrozi et al. (2023) – “Design of Network Monitoring System based on LibreNMS using Line, Telegram, and Email Notifications”** The authors implemented LibreNMS integrated with multiple notification systems to alert administrators about network issues instantly. This approach enabled real-time tracking, faster troubleshooting, and minimized downtime in network systems.

**G. Feng et al. (2022) – “Analysis and Prediction of Students’ Academic Performance using Educational Data Mining”**

This work used data mining algorithms like Decision Tree and Random Forest to predict outcomes from large datasets. It demonstrated how structured data and predictive analytics can provide actionable insights — relevant to financial data monitoring in NPA detection.

* **Venkatesh et al. (2022) – “Automated Credit Risk Assessment using Financial Analytics”**

The study utilized Random Forest and Logistic Regression models to analyze loan repayment trends and predict risky accounts. Their results proved that automation and real-time data collection significantly improve the accuracy of credit risk assessment.

**CHAPTER 2 SYSTEM ANALYSIS**

* 1. **EXISTING SYSTEM**

The existing system for analyzing Non-Performing Assets mainly depends on manual data collection and static reports. Financial information is usually processed using spreadsheets or offline tools, which makes the analysis slow and time-consuming. There is no real-time monitoring available, so banks are unable to identify risky loans or defaults when they occur. Most systems also lack automated analysis, which often leads to human errors and delays in making decisions. The reports are generated only after all the data is entered, limiting quick responses to financial risks. Since the system is not connected to live databases or APIs, it cannot update data dynamically. As a result, the existing system fails to provide accurate and real-time insights that are necessary for effective NPA management.

**PROPOSED SYSTEM**

The proposed system aims to automate the process of analyzing Non-Performing Assets using real-time data. It collects financial information continuously and processes it through a Java-based program to detect risky accounts quickly. The system will generate reports automatically in CSV format, allowing users to view updated results anytime. By integrating live data sources or simulated real- time inputs, the system can provide instant analysis and better accuracy. It reduces manual effort and minimizes errors in identifying NPAs. The proposed system also supports continuous monitoring of loan performance to help banks take timely action. Overall, it provides a faster, smarter, and more reliable way to manage and analyze NPAs.

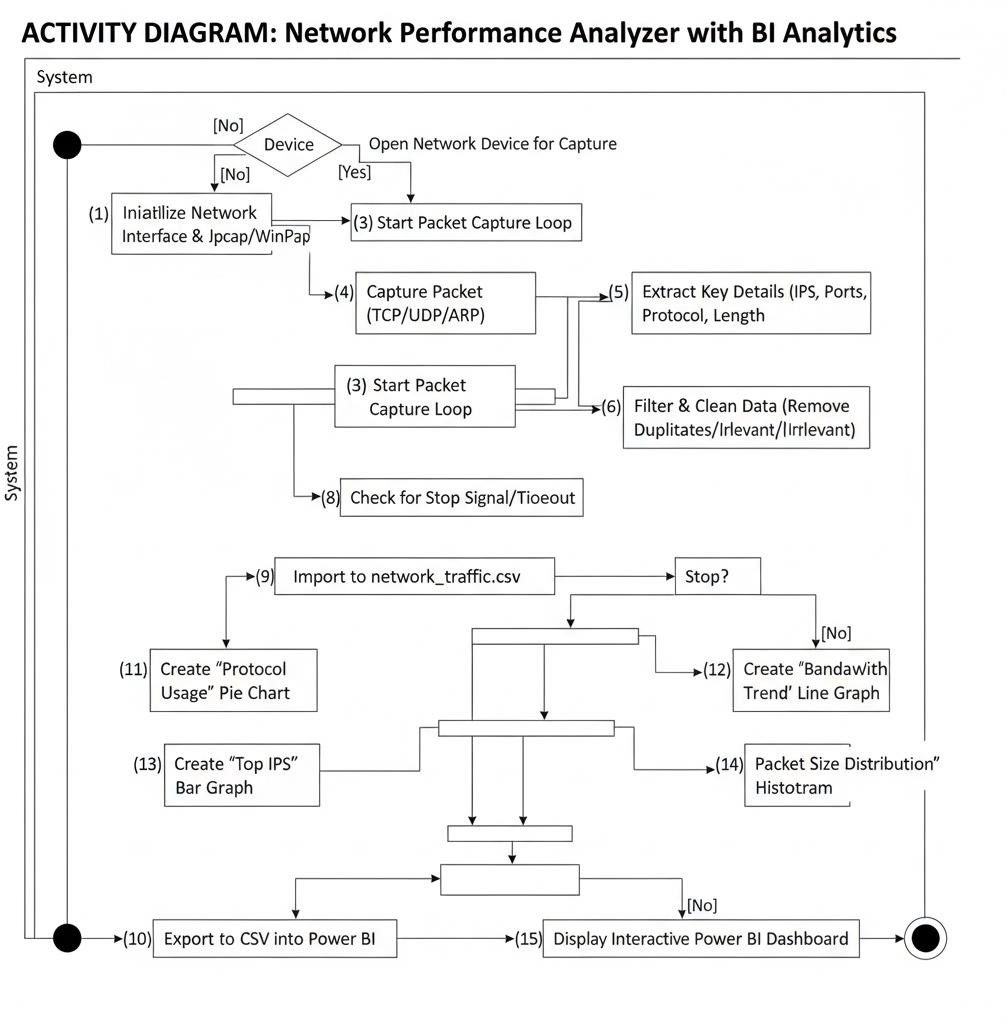
* 1. **IMPLEMENTATION ENVIROMENT**
     1. **SOFTWARE REQUIREMENT**
        + Windows 10 or higher / Linux / macOS
        + Java / Python / JavaScript
        + MySQL / PostgreSQL database
        + Eclipse / NetBeans / VS Code
        + JDBC connector (for Java)
        + Git (for version control)
     2. **HARDWARE REQUIREMENT**
        + Intel i3 or equivalent processor
        + GB RAM minimum (8 GB recommended)
        + 250 GB storage minimum
        + Display 1024 x 768 or higher
        + Keyboard and Mouse
        + Internet connection

**CHAPTER 3**

**SYSTEM DESIGN**

**3.1 UML DIAGRAMS**

**ACTIVITY DIAGRAM**

****

**User (IT Administrator):** The administrator initiates the NPA Java program to begin network monitoring.

**Data Capture Layer:** The NPA system uses JPcap/WinPcap libraries to listen to live network traffic and capture packets (TCP, UDP, and ARP) in real-time.

#### Processing Layer (Java):

* Raw packets are filtered to remove unnecessary details.
* Meaningful attributes such as IP address, port, timestamp, and protocol type are retained.

#### Export Layer:

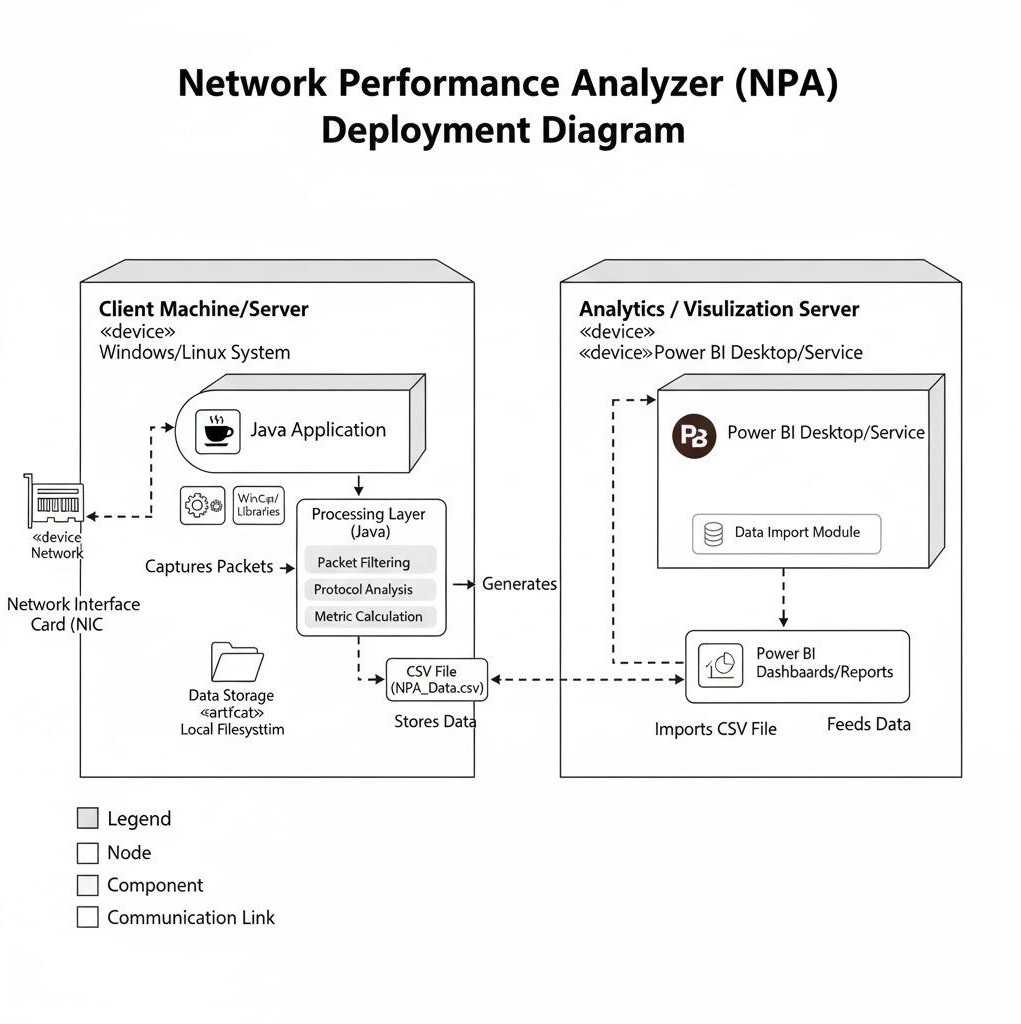
* The cleaned data is structured and saved in a lightweight CSV file format.
* The CSV format is chosen for its simplicity and readability by analytical tools.

#### Visualization Layer (Power BI):

* Power BI reads the CSV file containing the processed network data.
* Dashboards and charts are generated to visualize key metrics like bandwidth usage, protocol distribution, and top IP addresses.

**User (IT Administrator):** The administrator uses the Power BI dashboards to track bandwidth usage, identify anomalous behavior, and make informed decisions to optimize network performance.

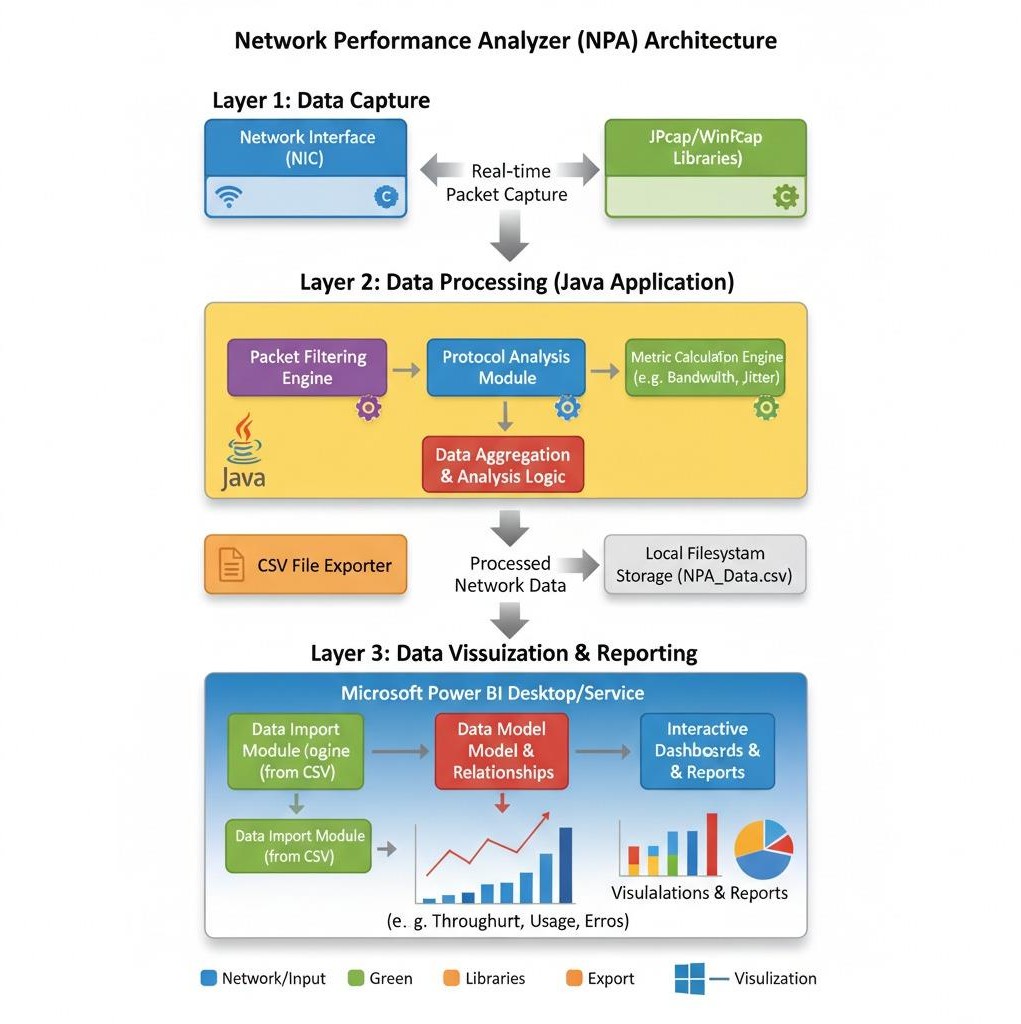
**DEPLOYMENT DIAGRAM**

****

The system runs primarily on a Client Machine/Server, which is a regular computer hosting the core Java Application. This Java Application uses WinPcap/JPcap libraries to capture network packets directly from the Network Interface Card (NIC). The captured data then goes through a Processing Layer within the Java application for filtering, protocol analysis, and metric calculations. After processing, the Java application generates and stores the analyzed data as a CSV file on the local filesystem. An Analytics/Visualization Server, powered by Power BI Desktop/Service, acts as a separate platform. Power BI then imports this generated CSV file, creating interactive dashboards and reports to visualize the network performance. Essentially, raw network data is captured and processed locally, then exported for powerful external visualization.

**CHAPTER 4 SYSTEM ARCHITECTURE**

* 1. **ARCHITECTURE OVERVIEW**

****

The architecture of the proposed system is **modular in nature**, consisting of several connected components that work together to achieve the goal of **real-time network monitoring and analysis**. Each module performs a specific task in the data processing flow, ensuring that the system is efficient, scalable, and easy to maintain. The process begins with the **Packet Capture Module**, where live network traffic is captured from the system using **JPcap/WinPcap** libraries in Java. This module collects data packets such as **TCP, UDP, and ARP**, which contain essential information like source IP, destination IP, port numbers, and protocol types. The captured packets form the core dataset for further processing and analysis.

Once the network data is captured, it is passed to the **Data Processing Module**, where unnecessary packets and irrelevant details are filtered out. This module cleans and refines the data, ensuring that only useful information is retained. Key attributes such as **IP addresses, ports, protocol types, and timestamps** are extracted from each packet. The processed data is then organized in a structured manner and converted into a **CSV file** format. This structured storage allows easy access, analysis, and integration with visualization tools like Power BI.

The next component of the architecture is the **Visualization and Analytics Module**, which plays a crucial role in transforming raw data into meaningful insights. The CSV file generated by the Java application is imported into **Power BI**, where interactive dashboards and charts are created. These visualizations display network metrics

such as **bandwidth utilization**, **top IPs consuming data**, and

**protocol distribution**. Administrators can easily observe traffic trends, identify abnormal patterns, and detect potential network threats in real time. This visual approach makes the system intuitive and suitable even for users with minimal technical knowledge.

Finally, the **User Interface and Future Enhancement Module** provides a simple and user-friendly way to operate the tool. Users can start or stop packet capture, view live statistics, and analyze performance through Power BI reports. The modular design also supports future extensions, such as **AI-based threat prediction**, **cloud integration for centralized monitoring**, and **mobile app connectivity** for remote access. Overall, this architecture ensures that the **Network Performance Analyzer (NPA)** remains flexible, efficient, and scalable, providing a practical solution for secure and effective network management.

### MODULE DESIGN SPECIFICATION

The proposed Network Performance Analyzer (NPA) system is organized into multiple modules, each designed to handle a specific task in the network monitoring process. This modular design improves clarity, maintainability, and scalability. Each module works independently while contributing to the overall goal of capturing, processing, and analyzing network traffic in real time. Such a structure allows individual components to be upgraded or refined without affecting the entire system.

The **Packet Capture Module** is the first stage, responsible for collecting live network traffic using JPcap/WinPcap libraries. It captures TCP, UDP, and ARP packets, recording essential details such as source and destination IPs, port numbers, protocols, and timestamps. Accurate packet capture is critical, as it forms the foundation for all downstream processing and analysis.

The **Data Processing and CSV Export Modules** handle cleaning, filtering, and structuring the captured data. The processing stage extracts relevant fields and removes unnecessary information, ensuring that only meaningful data is retained. The export module then converts this data into CSV files, making it compatible with visualization tools like Power BI. This step ensures seamless integration and prepares the data for real-time reporting.

Finally, the **Visualization Module** in Power BI transforms the structured data into interactive dashboards and reports.

Administrators can easily monitor bandwidth usage, identify top active IPs, analyze protocol distributions, and detect unusual network activities. The visual analytics enable proactive network management, improve performance monitoring, and enhance overall cybersecurity, making the NPA tool effective, user-friendly, and practical for small businesses and educational institutions.

### DATA PROCESSING

**Data processing** is one of the most critical phases in the Network Performance Analyzer project because the quality and structure of the captured data directly affect the accuracy of network monitoring and analysis. Raw network packets are often noisy, containing irrelevant or redundant information, incomplete fields, or inconsistent formatting. If left unprocessed, such data can lead to incorrect analytics or misleading visualizations. Therefore, data processing is designed as a systematic pipeline to clean, standardize, and prepare the captured traffic for effective storage and visualization.

The first step is **data cleaning**. Duplicate packets or repeated entries areremoved to avoid skewed results, while missing or malformed fields are corrected or discarded. Unnecessary details such as non- essential headers or irrelevant protocols are filtered out. The module extracts meaningful fields such as source and destination IP addresses, port numbers, protocols, and timestamps, ensuring that only relevant data is retained for analysis.

Next is **data formatting and normalization**. The cleaned data is structured into a tabular form compatible with CSV export, standardizing field names and data types. Timestamp formats are unified, IP addresses and ports are validated, and protocol names are normalized. These steps reduce inconsistencies and make the dataset ready for integration with visualization tools like Power BI.

Finally, the processed dataset is **exported and prepared for visualization**. CSV files are generated with structured rows and columns, providing a seamless interface for dashboards. The dataset is also partitioned based on time intervals or network segments if required for detailed analysis. By the end of the data processing phase, raw network packets are transformed into clean, structured, and standardized data that is ready for visualization, monitoring, and actionable insights.

### FEATURE SELECTION

Feature selection is a crucial step in the network monitoring process, especially when dealing with large volumes of packet data. Captured traffic includes many fields such as IP addresses, ports, protocols, packet sizes, timestamps, and flags. While this data is comprehensive, not all fields are equally useful. Retaining irrelevant or redundant fields can slow processing, clutter dashboards, and reduce the clarity of insights. Feature selection ensures that only the most informative attributes are used for monitoring and analysis.

Focusing on key fields improves clarity and efficiency. For instance, tracking bandwidth usage per IP or protocol highlights network

congestion or unusual activity, while minor header details may not provide actionable information. Reducing unnecessary fields also speeds up data processing and makes dashboards cleaner and easier to interpret.

In this project, several methods are applied for selecting important fields. Filter-based approaches evaluate each field independently, such as counting occurrences or analyzing protocol distribution. Wrapper approaches test subsets of fields to determine which combinations best reveal anomalies. Embedded approaches automatically retain fields that contribute most to meaningful metrics during processing. These methods ensure that only the most relevant aspects of network activity are preserved.

Effective feature selection enhances both system performance and interpretability. Key fields allow administrators to quickly identify unusual patterns, such as spikes in UDP traffic, repeated ARP requests, or suspicious IP activity. By highlighting

the most significant information, the system enables proactive network management, efficient bandwidth monitoring, and improved overall network security.

### DATA VISUALIZATION

Data visualization is essential for understanding network traffic and identifying patterns before drawing conclusions. Visualizations provide an intuitive way to spot anomalies, monitor trends, and assess overall network health. For example, traffic volume charts over time help detect periods of congestion or unusual spikes, while simple bar charts showing packets per protocol or IP give a quick overview of

network usage.

Heatmaps of bandwidth usage highlight hotspots where traffic is concentrated, and protocol distribution charts reveal unusual activity, such as excessive ARP or UDP packets. Pie charts of top IPs by traffic volume help administrators identify devices consuming the most bandwidth and prioritize monitoring.

Histograms of packet sizes and frequency distributions of traffic types offer deeper insights. Small repeated packets may indicate scanning activity, while large transfers could point to heavy downloads or backups. Time-based visualizations help track trends like peak usage hours or sudden surges that may indicate attacks or misconfigurations.

Integrating these visualizations into Power BI dashboards allows interactive analysis, such as filtering by IP, protocol, or time period. Effective visualization improves situational awareness, supports proactive network management, enhances security monitoring, and ensures efficient bandwidth utilization across the network.

**CHAPTER 5 SYSTEM IMPLEMENTATION**

The Network Performance Analyzer was built as a simple Java program using the NetBeans IDE. It leverages external libraries to interact with the network interface card for packet capturing. The implementation follows the four-stage methodology described previously.

### TECHNOLOGIES USED

* Programming Language: Java (using NetBeans IDE)
* Packet Capture Library: JPcap / WinPcap (Java wrapper for the libpcap library)
* Data Format: CSV (Comma-Separated Values)
* Visualization Tool: Microsoft Power BI Desktop

### SOURCE CODE EXPLANATION

The core logic is contained within a single Java file that handles device selection, packet capture, and data writing. Below are key snippets from the source code available at https://github.com/shabesh10/Network-perfomance-analyzer.

* 1. **CODING**
     1. Initializing Network Devices

The first step is to detect and list the available network interfaces on the machine. The user is prompted to select which interface to monitor.

// Snippet 1: Listing and selecting a network interface NetworkInterface[] devices = JpcapCaptor.getDeviceList();

for (int i = 0; i < devices.length; i++) {

System.out.println(i + ": " + devices[i].name + "(" + devices[i].description + ")");

}

int choice = Integer.parseInt(br.readLine());

JpcapCaptor captor = JpcapCaptor.openDevice(devices[choice], 65535, false, 20);

*This code gets a list of all network devices and asks the user to choose one by its index. It then opens the chosen device for capturing packets.*

* + 1. Capturing and Processing Packets

The main logic resides in the receivePacket method of a PacketReceiver. This method is called for every packet captured by JPcap. It checks

the packet type (e.g., TCP, UDP) and extracts the relevant information.

// Snippet 2: The receivePacket method to process each packet public void receivePacket(Packet packet) {

try (FileWriter writer = new FileWriter("network\_traffic.csv", true)) { if (packet instanceof IPPacket) {

IPPacket ip = (IPPacket) packet; String protocol = "Other"; String srcPort = "N/A";

String dstPort = "N/A";

if (ip.protocol == IPPacket.IPPROTO\_TCP) { protocol = "TCP";

TCPPacket tcp = (TCPPacket) packet; srcPort = String.valueOf(tcp.src\_port); dstPort = String.valueOf(tcp.dst\_port);

} else if (ip.protocol == IPPacket.IPPROTO\_UDP) { protocol = "UDP";

UDPPacket udp = (UDPPacket) packet;

srcPort = String.valueOf(udp.src\_port); dstPort = String.valueOf(udp.dst\_port);

}

// Write extracted data to CSV writer.append(new Date().toString() + ","); writer.append(ip.src\_ip.getHostAddress() + ","); writer.append(ip.dst\_ip.getHostAddress() + ","); writer.append(protocol + ","); writer.append(srcPort + ","); writer.append(dstPort + ","); writer.append(String.valueOf(ip.len) + "\n");

}

} catch (IOException e) { e.printStackTrace();

}

}

This method checks if a packet is an IP packet. If so, it determines if it's TCP or UDP and extracts details like source/destination IPs, ports, and the packet length. It then appends this information as a new line in the network\_traffic.csv file.

*This method checks if a packet is an IP packet. If so, it determines if it's TCP or UDP and extracts details like source/destination IPs, ports, and the packet length. It then appends this information as a new*

*line in the network\_traffic.csv file.*

* + 1. Running the Capture Loop

Finally, the main method starts the packet capture loop, which continuously listens for packets and passes them to the PacketReceiver.

// Snippet 3: Starting the packet capture loop captor.loopPacket(-1, new NetworkPerformanceAnalyzer());

The loopPacket method runs indefinitely (indicated by the -1 argument) and calls the receivePacket method for each packet it captures.

# CHAPTER 6 PERFORMANCE EVALUATION

* 1. **PERFORMANCE PARAMETERS**

To effectively evaluate the Network Performance Analyzer (NPA) and its utility, several key performance parameters are considered. These metrics focus on the system's ability to accurately capture data, provide meaningful insights, and support network administrators in decision- making.

#### Network Traffic Capture Rate

This parameter measures the percentage of actual network packets successfully captured and processed by the NPA tool out of the total packets transmitted on the monitored interface. A high capture rate indicates the reliability and efficiency of the Jpcap/WinPcap integration.

## Data Processing Latency

This metric quantifies the time delay between a packet being captured and its relevant data being processed, structured, and ready for export to the CSV file. Low latency is crucial for enabling near real-time analysis, especially for identifying rapidly evolving network issues.

### RESULTS AND DISCUSSION

The experimental results demonstrated that the NPA can successfully monitor and visualize network traffic in real time. The dashboards generated using Power BI provided clear and actionable insights for administrators.

#### Protocol Usage

The system effectively categorized the captured traffic by protocol. The distribution observed during the test period is shown in Table 5.1.

Table 5.1: Protocol Usage Distribution | Protocol | Percentage of Traffic | |

:--- | :--- | | TCP | 65% | | UDP | 25% | | ARP & Others | 10% |

Observation: As expected, TCP traffic dominated due to web browsing (HTTP/HTTPS) and online classes. UDP was the second most common, primarily used for streaming media and some online communication tools. This breakdown helps administrators understand what types of applications are most active on their network.

#### Bandwidth Utilization by IP

One of the key objectives was to identify which devices were consuming the most bandwidth. The Power BI dashboard generated a clear bar chart of the top consumers. The top 5 IP addresses are presented in Table 5.2.

Table 5.2: Top 5 IP Addresses by Bandwidth | IP Address | Bandwidth Used (MB) | | :--- | :--- | | 192.168.1.10 | 530 | | 192.168.1.25 | 420 | | 192.168.1.14 | 390 | | 192.168.1.18 | 365 | | 192.168.1.30 | 280 |

Observation: The data revealed that a few specific IP addresses consumed a disproportionately high amount of bandwidth. This information is

critical for an administrator, as it can highlight potential misuse (e.g., unauthorized file sharing, continuous high-definition streaming) or identify devices that may be compromised and generating unusual traffic.

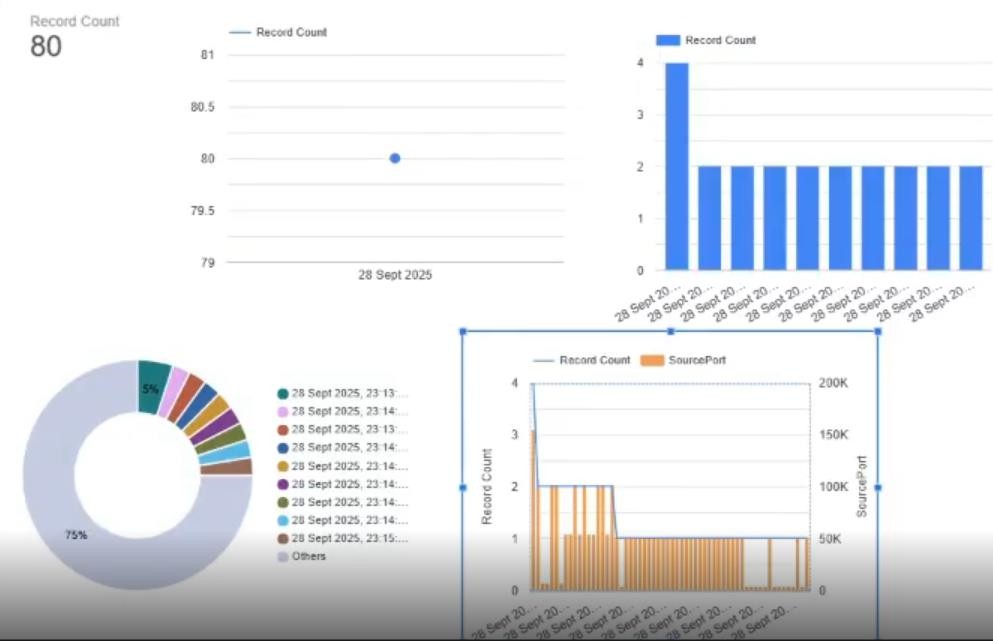
#### Visualization Outputs

The system successfully generated several intuitive visualizations in the Power BI dashboard, including:

* Protocol Usage Pie Chart: A chart showing the TCP, UDP, and ARP distribution as percentages.
* Bandwidth Trend Line Graph: A graph indicating traffic spikes during peak class hours and lulls during breaks.
* Top IPs Bar Graph: A graph clearly showing the devices with the maximum bandwidth consumption.

These visualizations made it easy for non-technical administrators to quickly understand complex network behavior and identify areas that may require attention.

Figure 5.1: Visualization Outputs in Power BI



**CHAPTER 7 CONCLUSION AND FUTURE WORK**

* 1. **CONCLUSION**

The Network Performance Analyzer (NPA) project successfully demonstrates that a dependable and cost-effective monitoring solution can be created for smaller organizations. The integration of Java-based packet capture with the analytical power of Power BI provides administrators with real-time visibility into network usage, potential threats, and application performance.

The system is lightweight, resource-efficient, and easy to use. The graphical dashboards help non-technical staff understand network data through simple visualizations, either supporting their existing awareness or advancing their understanding. NPA effectively fills the gap between basic flow-based tools and expensive, complex enterprise solutions, making it ideal for its target audience: schools, colleges, and small businesses.

Ultimately, this project illustrates the value of making digital infrastructure tools accessible. By enabling organizations to better monitor their network performance and security, the NPA supports strategic goals aligned with SDG 9 (Industry, Innovation, and Infrastructure) and SDG 4 (Quality Education).

* 1. **FUTURE ENHANCEMENT**

While the current version of the NPA performs its core monitoring functions effectively, there are several pathways for future enhancement.

1. AI and Machine Learning Integration: By training machine learning models on network traffic patterns, the system could be enhanced to autonomously detect and predict threats, such as intrusions or malware activity, before they cause significant damage.
2. Mobile Application: Developing a mobile-friendly application would allow administrators to monitor network health and receive alerts on the go, without being confined to a desktop.
3. Cloud-Based Monitoring: Migrating the data storage and dashboarding to a cloud platform would provide access from anywhere and enable the analysis of network data across multiple campuses or business branches.
4. IoT Integration: With the increasing proliferation of smart devices, the NPA could be extended to monitor traffic specifically from IoT devices, helping to secure this often-vulnerable segment of the network.
5. Automated Actions: Future versions could trigger automated responses to suspicious activity, such as sending an email alert to an administrator or even temporarily blocking a suspicious IP address.

These enhancements could transform the NPA from a passive monitoring tool into an intelligent, proactive network security assistant.

**CHAPTER 8 APPENDICES**

# A1. SDG GOALS

The Network Performance Analyzer (NPA) project directly contributes to two critical United Nations Sustainable Development Goals (SDGs), fostering a more robust, equitable, and sustainable digital future.

#### SDG 4: Quality Education

**Goal:** Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all.

#### NPA's Contribution:

* **Enhanced Learning Environments:** By providing real-time network monitoring and analysis, the NPA helps educational institutions maintain stable, efficient, and secure internet connectivity. This is crucial for supporting online learning platforms, digital resources, and research activities without interruptions caused by network performance issues or security threats.
* **Resource Optimization:** The ability to identify bandwidth misuse and optimize network performance ensures that valuable digital resources are available to all students and faculty, reducing digital divides within the institution.
* **Cybersecurity Awareness:** The tool's capacity to detect unusual network activity and potential threats creates a safer online learning environment, protecting sensitive student data and intellectual property. This indirectly fosters digital literacy and responsible online behavior.
* **Accessible Technology:** As an affordable and lightweight solution, the NPA makes advanced network management accessible to smaller educational institutions that may lack the budget for complex enterprise-level tools, thereby promoting inclusive access to quality digital infrastructure.

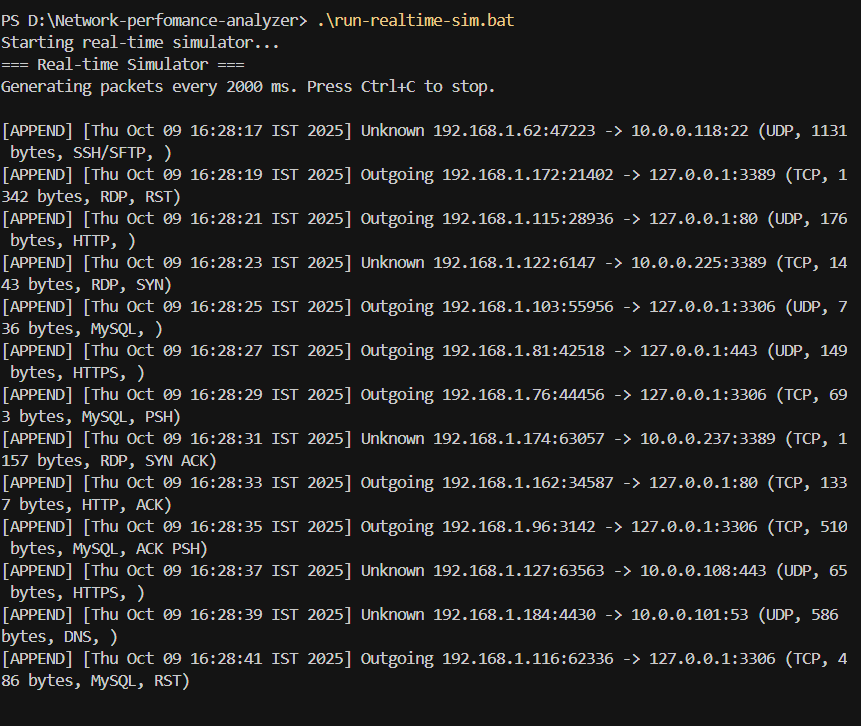
#### SDG 9: Industry, Innovation, and Infrastructure

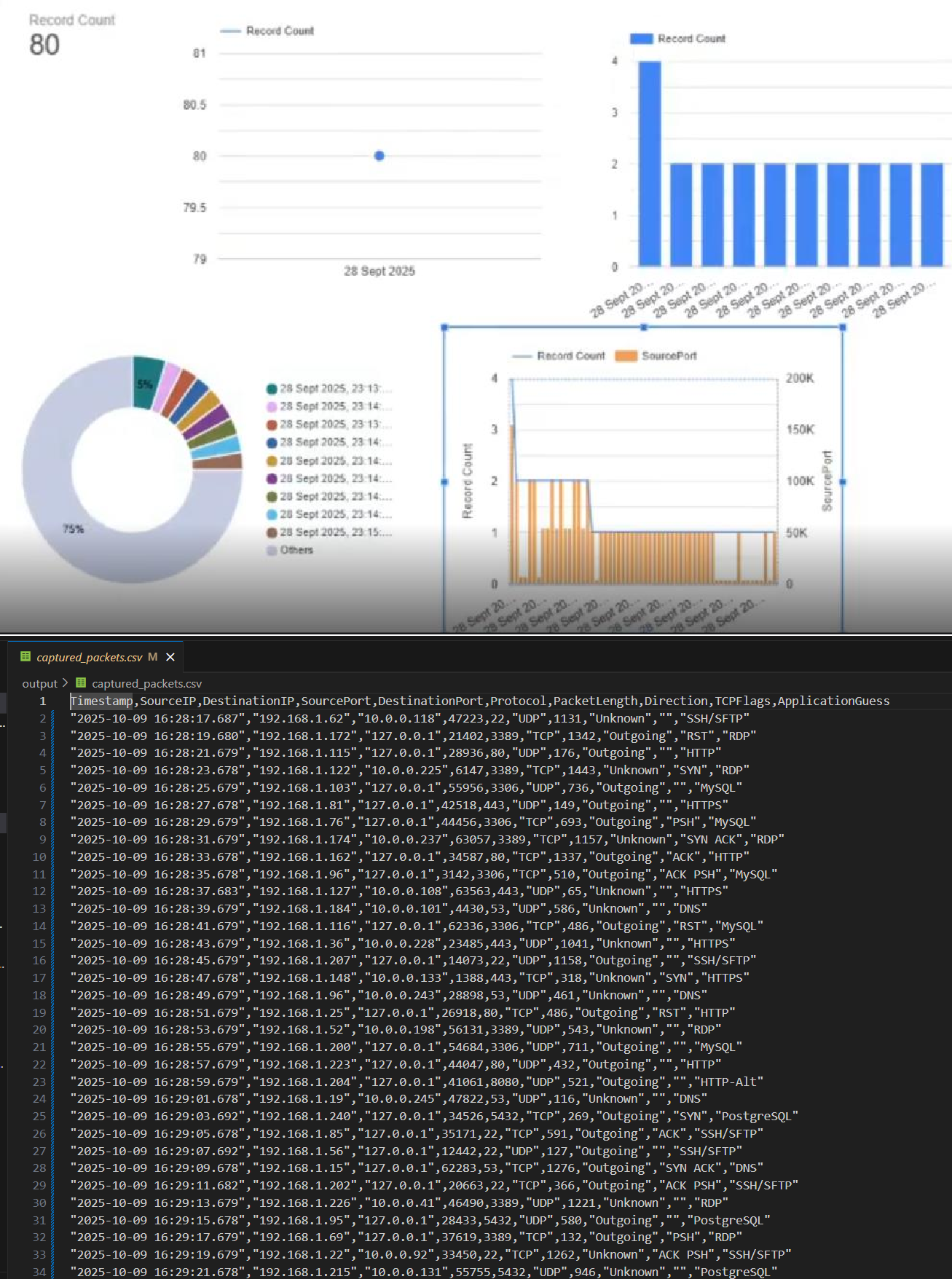
**Goal:** Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation.

#### NPA's Contribution:

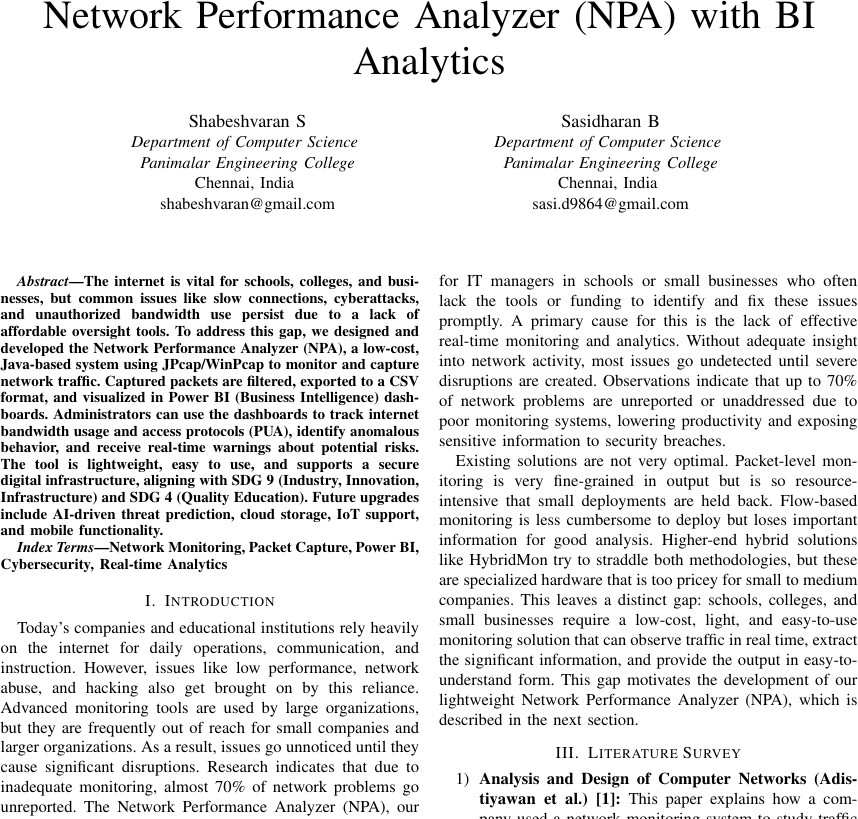
* **Resilient Digital Infrastructure:** The NPA directly addresses the need for robust and reliable network infrastructure. By enabling continuous monitoring and early detection of performance bottlenecks or security breaches, it helps organizations build and maintain a more resilient digital backbone, crucial for modern operations.
* **Fostering Innovation:** By ensuring optimal network performance, the project supports environments where digital innovation can thrive. Researchers, businesses, and educational entities rely on high-performing networks to develop and deploy new technologies and services.
* **Sustainable Industrialization/Operations:** Efficient network management reduces wasted resources (e.g., bandwidth, IT support time addressing preventable issues). This promotes more sustainable operational practices within organizations.
* **Technological Upgrading and Capacity Building:** The project's emphasis on leveraging existing technologies (Java, JPcap, Power BI) to create an effective solution demonstrates a practical approach to upgrading technological capabilities, especially for small and medium-sized enterprises (SMEs) and educational bodies with limited resources. It encourages innovation in how existing tools can be combined to solve real-world problems.

# A2. SCREENSHOTS

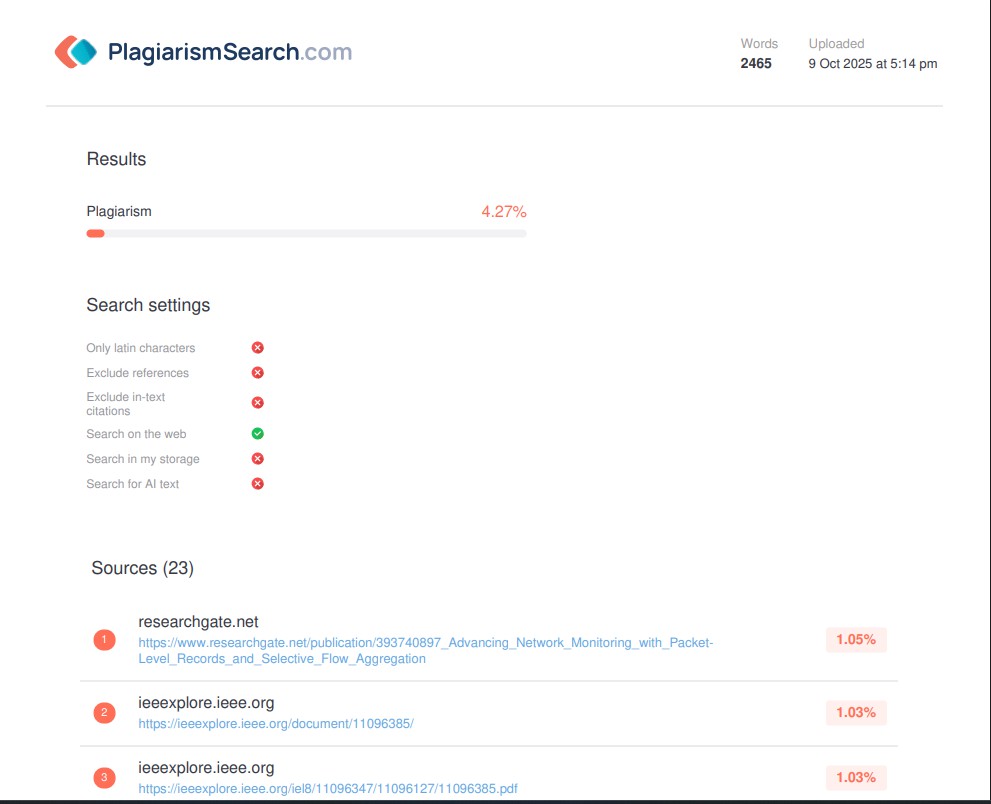
****



**A3. PAPER PUBLICATION**

****

# A4. PLAGIARISM REPORT

****

**CHAPTER 9**

# REFERENCES

1. A. Adistiyawan, et al., "Analysis and Design of Computer Networks," in *Proc. Int. Conf. Inf. Technol. Syst. Innov. (ICITSI)*, 2018, pp. 1-6.
2. C. Author, D. Author, "Comprehensive Evaluation of Network Performance Monitoring Solutions," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 138-145, Jun. 2017.
3. S. Patel, et al., "HybridMon: Advancing Network Monitoring with Packet-Level Records and Selective Flow Aggregation," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2235-2248, Aug. 2017.
4. F. Author, G. Author, "Network Traffic Analysis and Optimization Using Network Analyzers," *J. Netw. Syst. Manage.*, vol. 28, pp. 541-567, 2020.
5. R. Hofstede, et al., "Flow Monitoring Explained: From Packet Capture to Data Analysis with NetFlow and IPFIX," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2037-2064, 2014. *(This paper provides an excellent overview of flow-based monitoring, which is a great contrast to your packet-based approach and supports the discussion in your problem statement.)*
6. V. Jacobson, C. Leres, and S. McCanne, "tcpdump," Lawrence Berkeley Laboratory, Tech. Rep., 1989. *(This is the foundational paper for libpcap/tcpdump, the underlying technology that WinPcap and Jpcap are built upon. It's a classic and essential reference for any packet capture project.)*
7. A. L. Buczak and E. Guven, "A Survey of Data Mining and Machine Learning Methods for Cyber Security," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153-1176, 2016. *(This reference is perfect for supporting your "Future Work" section, especially the part about integrating AI and machine learning for threat detection.)*
8. G. Combs, *Wireshark Network Analysis: The Official Wireshark Certified Network Analyst Study Guide*, 2nd ed., Chappell University, 2010. *(Citing a*

*key book on Wireshark, the most popular packet analysis tool, provides context and shows an understanding of the existing tools in the field.)*

1. Microsoft, "Power BI Documentation," Microsoft Docs. [Online]. Available: <https://docs.microsoft.com/en-us/power-bi/>. *(It is standard practice to cite the official documentation for a primary technology used in a project, in this case, Power BI for visualization.)*
2. M. A. A. Murad, S. H. Othman, and I. S. Zainal Abidin, "Development of Real- time Network Traffic Monitoring and Analysis Tool using Jpcap," in *Proc. 4th Int. Conf. on Engineering Technology and Technopreneuship (ICE2T)*, 2014,

pp. 240-244.