VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi-560018, Karnataka, India

**A Mini-Project Report**

on

# "TRAFFIC ANALYSER"

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

**BACHELOR OF ENGINEERING**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

***Submitted by***

**K M AKANKSHA SINGH [1VE21CS072]**

**KRITIKA TAANK [1VE21CS083]**

**MUSKAN [1VE21CS103]**

**MUSKAN PATEL [1VE21CS104]**

*Under the Guidance of*

**Dr. Karthik B U**

Assistant Professor

**Department of Computer Science and Engineering**

Sri Venkateshwara College of Engineering, Bangalore-562 157

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING SRI VENKATESHWARA COLLEGE OF ENGINEERING BANGALORE - 562 157**

**2023-2024**

# Traffic Analyser

## Abstract:

Traffic analysis plays a crucial role in understanding and managing network behavior. This paper presents an efficient traffic analyzer designed for computer networks. The proposed analyzer utilizes advanced algorithms and techniques to capture, process, and analyze network traffic data effectively. Key features include real-time monitoring, deep packet inspection, anomaly detection, and traffic classification. The analyzer employs machine learning models for predictive analysis, enabling proactive network management and security measures. Additionally, it offers scalability and flexibility to adapt to diverse network environments. Experimental results demonstrate the effectiveness and performance of the proposed traffic analyzer in enhancing network visibility, optimizing resource utilization, and improving overall network performance.

## Introduction:

In the realm of computer networks, the efficient management and analysis of network traffic are paramount for ensuring optimal performance, security, and resource utilization. With the increasing complexity and scale of modern networks, the need for advanced traffic analyzers has become more pressing than ever. This mini-project endeavors to address this need by proposing an efficient traffic analyzer tailored for computer networks.

The essence of network traffic analysis lies in understanding the patterns, characteristics, and dynamics of data packets traversing through the network infrastructure. By scrutinizing this traffic, network administrators can gain valuable insights into the health, behavior, and potential vulnerabilities of the network. Moreover, traffic analysis forms the basis for various network management tasks, including performance optimization, troubleshooting, and security enforcement.

The proposed traffic analyzer is designed to meet the evolving demands of contemporary networks. Leveraging cutting-edge algorithms and methodologies, it aims to provide comprehensive visibility into network traffic while minimizing computational overhead. Real-time monitoring capabilities enable instantaneous detection and response to network events, ensuring timely intervention in case of anomalies or security breaches.

A key aspect of the traffic analyzer is its ability to perform deep packet inspection, allowing for granular analysis of packet contents. This capability facilitates not only traffic classification based on protocol, application, or user, but also the detection of suspicious or malicious activities within the network. By employing sophisticated machine learning models, the analyzer can predict future traffic patterns and anticipate potential performance bottlenecks or security threats.

Scalability and adaptability are fundamental principles guiding the development of this traffic analyzer. Whether deployed in small-scale LANs or large-scale enterprise networks, the analyzer is designed to seamlessly integrate into diverse network environments. Moreover, it offers flexibility in terms of customization and configuration, allowing administrators to tailor its functionality to suit specific network requirements.

Through this mini-project, we aim to showcase the feasibility and efficacy of our proposed traffic analyzer in enhancing network management and security. By providing a robust framework for traffic analysis, we envision empowering network administrators with the tools they need to effectively monitor, optimize, and secure their networks in the face of evolving challenges and threats.

## How traffic analyser works:

In a mini-project focused on developing a traffic analyzer for computer networks, the system typically follows a structured process involving data collection, processing, analysis, and visualization. Below is a high-level overview of how a traffic analyzer works in such a project:

**1. Data Collection:**

The traffic analyzer begins by collecting data packets traversing the network. This data can be obtained by passively monitoring network traffic using techniques such as port mirroring, network taps, or packet sniffing.
Alternatively, the traffic analyzer may integrate with network devices (e.g., routers, switches) to extract traffic statistics or logs in real-time.

**2. Pre-processing:**

Raw packet data collected from the network is pre-processed to extract relevant information. This may involve parsing packet headers to identify protocol types, source/destination IP addresses, port numbers, packet size, and timestamps.
Pre-processing may also include filtering out irrelevant traffic or noise to focus on meaningful data for analysis.

**3. Traffic Analysis:**

Once pre-processed, the traffic data undergoes various analysis techniques to extract insights and identify patterns.

Traffic analysis may involve:
Protocol Identification: Classifying traffic based on the communication protocols used (e.g., HTTP, FTP, SSH).
Traffic Volume Analysis: Examining the volume of traffic over time to identify peak usage periods, trends, and anomalies.
Traffic Flow Analysis: Analyzing the flow of packets between network nodes to understand communication patterns and identify potential bottlenecks.
Anomaly Detection: Employing statistical methods or machine learning algorithms to detect unusual or suspicious behavior in network traffic, such as DDoS attacks or malware infections.
Quality of Service (QoS) Analysis: Assessing network performance metrics such as latency, packet loss, and jitter to ensure adherence to service level agreements (SLAs).

**4. Visualization and Reporting:**

The results of traffic analysis are presented to the user through visualizations such as graphs, charts, and dashboards.
Visualization techniques help convey complex network data in a comprehensible manner, enabling network administrators to quickly identify trends, anomalies, and areas for improvement.
Additionally, the traffic analyzer may generate reports summarizing key findings, trends, and recommendations for network optimization and security enhancement.

**5. Real-time Monitoring and Alerts:**

The traffic analyzer may provide real-time monitoring capabilities, allowing administrators to observe network traffic as it occurs.
In conjunction with real-time monitoring, the system can generate alerts or notifications based on predefined thresholds or detected anomalies, enabling prompt response to network incidents.

Overall, the traffic analyzer in a computer network mini-project functions as a critical tool for network monitoring, analysis, and management, empowering administrators to optimize network performance, ensure security, and maintain reliability.

## Advantages:

- Improved Network Visibility.
- Proactive Monitoring and Detection.
- Enhanced Security.
- Optimization of Network Performance.

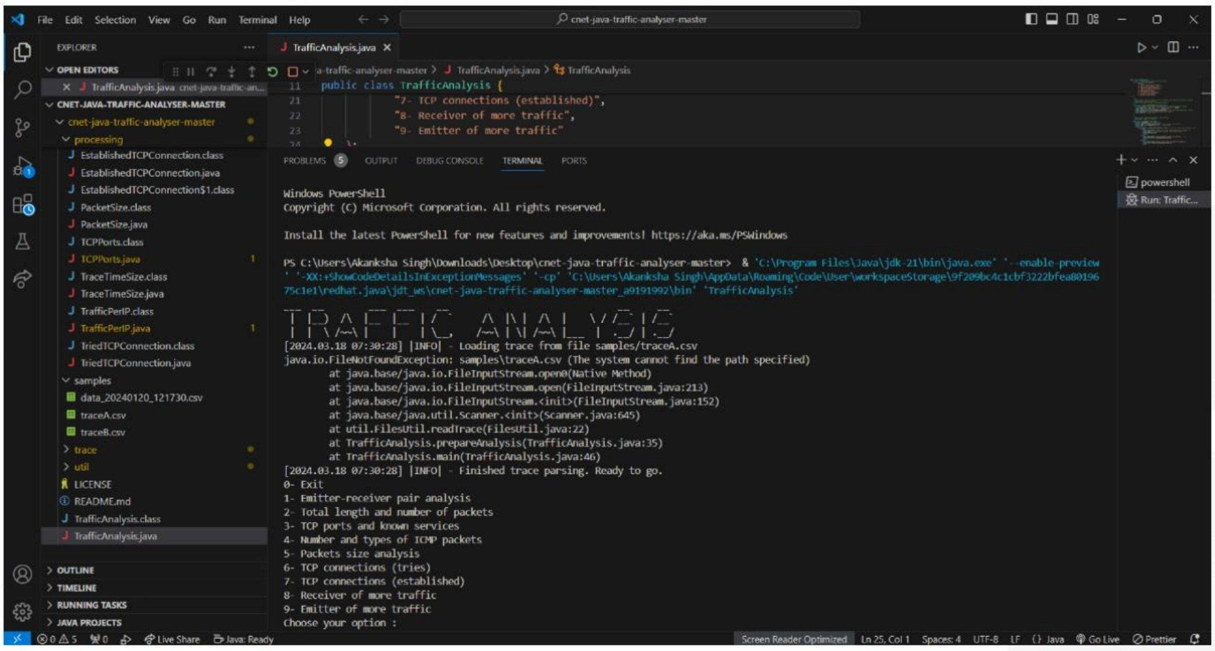
## Disadvantages:

- Resource Intensiveness.
- Complexity of Implementation.
- Privacy Concerns.
- Integration Challenges.

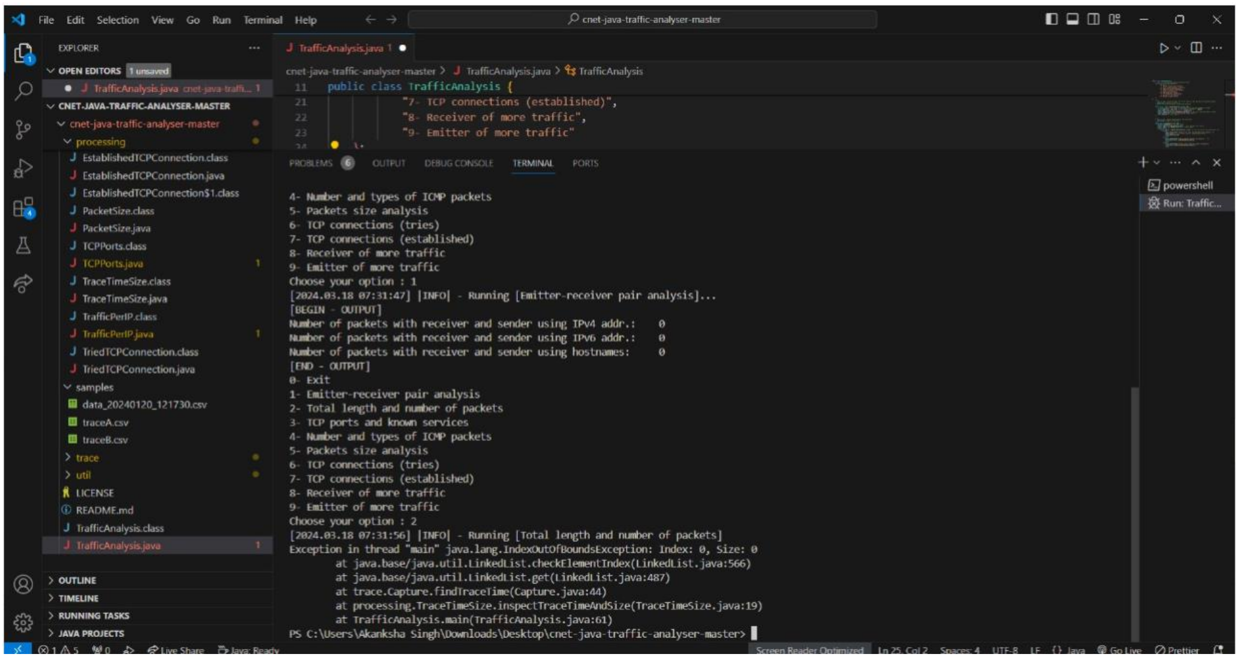
**Code:**

```
import util.FilesUtil;
import util.MenuUtils;
import util.Logger;
import trace.Capture;
import processing.*;
import java.util.Scanner;


/**
 * this class provides methods and functions to process interactively a capture and get information
by applying multiple modules
 */
public class TrafficAnalysis {
    // array of module execution options available to user
    private static String[] options = {
            "0- Exit",
            "1- Emitter-receiver pair analysis",
            "2- Total length and number of packets",
            "3- TCP ports and known services",
            "4- Number and types of ICMP packets",
            "5- Packets size analysis",
            "6- TCP connections (tries)",
            "7- TCP connections (established)",
            "8- Receiver of more traffic",
            "9- Emitter of more traffic"
    };

     /**
     * Given an args, verify the target file of the analysis, read and return a instance of Capture
     * @param args  Arguments of execution
     * @return a instance of Capture, parsed from targetFile
     */
    private static Capture prepareAnalysis(String[] args) {
        // verify if there's any target file defined by args and setup targetFile variable
        String targetFile = args.length == 0 ? "samples/traceA.csv" : args[0];
        Logger.info("Loading trace from file " + targetFile);
        Capture capture = FilesUtil.readTrace(targetFile); // read trace and build Capture instance
        Logger.info("Finished trace parsing. Ready to go.");
        return capture; // return capture
    }

     /**
     * Given an args, execute TrafficAnalisys tool life-cycle
     * @param args  Arguments of execution
     */
 public static void main(String[] args) {
        MenuUtils.showBanner(); // display banner
        Capture capture = prepareAnalysis(args); // parse args and build capture
        Scanner stdin = new Scanner(System.in); // setup scanner
        while (true) {
           switch (MenuUtils.getUserOption(options, stdin)) { // get user option and evaluate cases

              case 0:
                 Logger.info("Thank you for using TrafficAnalysis tool! Exiting...");
                 stdin.close(); // close scanner and exit program with status 0
                 System.exit(0);
                 break;

             case 1:
                 Logger.info("Running [Emitter-receiver pair analysis]... ");
                 EmitterReceiverPair.inspectAddresses(capture);
                 break;

              case 2:
                 Logger.info("Running [Total length and number of packets]");
                 TraceTimeSize.inspectTraceTimeAndSize(capture);
                 break;

            case 3:
                 Logger.info("Running [TCP ports and known services]");
                 TCPPorts.inspectPortsAndServices(capture);
                 break;

              case 4:
                 Logger.info("Running [Number and types of ICMP packets]...");
                 ControlMessagePacket.inspectICMP(capture);
                 break;

              case 5:
                 Logger.info("Running [Packets size analysis]...");
                 PacketSize.inspectPacketsSize(capture);
                 Logger.info("Exported additional data to ./samples/data_{DATETIME}.csv file.");
                 Logger.info("In order to generate a barplot from the exported data, execute the
python script inside 'plugin' folder.");
                 break;

              case 6:
                 Logger.info("Running [TCP connections (tries)]");
                 TriedTCPConnection.inspectTriedTcpConnections(capture);
                 break;
               case 7:
                 Logger.info("Running [TCP connections (established)");
                 EstablishedTCPConnection.inspectEstablishedTcpConnections(capture);
                 break;


               case 8:
                 Logger.info("Running [Receiver of more traffic]");
                 TrafficPerIP.inspectReceivers(capture);
                 break;


              case 9:
                 Logger.info("Running [Emitter of more traffic]");
                 TrafficPerIP.inspectEmitters(capture);
                 break;
              default:
                 Logger.error("The selected option is not implemented yet.");
                 break;
           }
        }
```

**Output:**



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Akanksha Singh\Downloads\Desktop\cnet-java-traffic-analyser-master>  & 'C:\Program Files\Java\jdk-21\bin\java.exe' '--enable-preview
' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Akanksha Singh\AppData\Roaming\Code\User\workspaceStorage\9f209bc4c1cbf3222bfea80196
75c1e1\redhat.java\jdt_ws\cnet-java-traffic-analyser-master_a9191992\bin' 'TrafficAnalysis'

TRAFFIC ANALYSIS
[2024.03.18 07:30:28] |INFO| - Loading trace from file samples/traceA.csv
java.io.FileNotFoundException: samples\traceA.csv (The system cannot find the path specified)
        at java.base/java.io.FileInputStream.open0(Native Method)
        at java.base/java.io.FileInputStream.open(FileInputStream.java:213)
        at java.base/java.io.FileInputStream.<init>(FileInputStream.java:152)
        at java.base/java.util.Scanner.<init>(Scanner.java:645)
        at util.FilesUtil.readTrace(FilesUtil.java:22)
        at TrafficAnalysis.prepareAnalysis(TrafficAnalysis.java:35)
        at TrafficAnalysis.main(TrafficAnalysis.java:46)
[2024.03.18 07:30:28] |INFO| - Finished trace parsing. Ready to go.
0- Exit
1- Emitter-receiver pair analysis
2- Total length and number of packets
3- TCP ports and known services
4- Number and types of ICMP packets
5- Packets size analysis
6- TCP connections (tries)
7- TCP connections (established)
8- Receiver of more traffic
9- Emitter of more traffic
Choose your option :
```



```
public class TrafficAnalysis {
                "7- TCP connections (established)",
                "8- Receiver of more traffic",
                "9- Emitter of more traffic"
```

```
4- Number and types of ICMP packets
5- Packets size analysis
6- TCP connections (tries)
7- TCP connections (established)
8- Receiver of more traffic
9- Emitter of more traffic
Choose your option : 1
[2024.03.18 07:31:47] |INFO| - Running [Emitter-receiver pair analysis]...
[BEGIN - OUTPUT]
Number of packets with receiver and sender using IPv4 addr.:    0
Number of packets with receiver and sender using IPv6 addr.:    0
Number of packets with receiver and sender using hostnames:     0
[END - OUTPUT]
0- Exit
1- Emitter-receiver pair analysis
2- Total length and number of packets
3- TCP ports and known services
4- Number and types of ICMP packets
5- Packets size analysis
6- TCP connections (tries)
7- TCP connections (established)
8- Receiver of more traffic
9- Emitter of more traffic
Choose your option : 2
[2024.03.18 07:31:56] |INFO| - Running [Total length and number of packets]
Exception in thread "main" java.lang.IndexOutOfBoundsException: Index: 0, Size: 0
        at java.base/java.util.LinkedList.checkElementIndex(LinkedList.java:566)
        at java.base/java.util.LinkedList.get(LinkedList.java:487)
        at trace.Capture.findTraceTime(Capture.java:44)
        at processing.TraceTimeSize.inspectTraceTimeAndSize(TraceTimeSize.java:19)
        at TrafficAnalysis.main(TrafficAnalysis.java:61)
PS C:\Users\Akanksha Singh\Downloads\Desktop\cnet-java-traffic-analyser-master>
```

## Conclusion:

In conclusion, the development of a traffic analyzer for computer networks represents a significant advancement in network management, security, and performance optimization. Through the comprehensive analysis of network traffic data, the analyzer offers valuable insights into network behavior, facilitates proactive monitoring and detection of anomalies, and enables informed decision-making by network administrators.

While the traffic analyzer presents numerous advantages, including improved network visibility, enhanced security, and optimized performance, it also poses certain challenges and considerations. These include resource intensiveness, complexity of implementation, privacy concerns, potential for false positives/negatives, network overhead, integration challenges, cost considerations, and legal/regulatory compliance requirements.

Despite these challenges, the benefits of deploying a traffic analyzer are substantial, providing organizations with the tools and capabilities needed to effectively manage and secure their networks in today's dynamic and evolving digital landscape. By addressing the challenges through careful planning, implementation, and management, organizations can harness the full potential of the traffic analyzer to optimize network operations, enhance security posture, and ensure compliance with regulatory requirements.

In essence, the traffic analyzer serves as a critical component of modern network infrastructure, empowering organizations to proactively identify and address network issues, mitigate security threats, and optimize resource utilization, thereby enabling them to achieve their business objectives in a connected and digitized world. Through ongoing innovation, refinement, and adaptation, the traffic analyzer continues to evolve to meet the ever-changing demands and challenges of today's networks, ensuring the continued efficiency, reliability, and security of network communications.

## References:

1. PingPlotter: Continually test and network and plot latency, packet loss, and jitter on an infinite timeline.

   Website: PingPlotter

2. Wireshark: It is simple-to-use interface provides an overview of your capture traffic in the list pane and specific information about each packet in the details pane.

   Website: Wireshark

3. GitHub