# LINUX NETWORK PACKET STATISTICS DISPLAY PROJECT REQUIREMENT DOCUMENT

Table of Contents

[**1.** **INTRODUCTION** 3](#_Toc168389856)

[**2.** **OBJECTIVES** 3](#_Toc168389857)

[**3.** **REQUIREMENTS** 4](#_Toc168389858)

[***a)*** ***Functional Requirements:*** 4](#_Toc168389859)

[*a.* *Packet Capture Functionality:* 4](#_Toc168389860)

[*b.* *Protocol Identification:* 4](#_Toc168389861)

[*c.* *Header Parsing:* 4](#_Toc168389862)

[*d.* *Logging Functionality:* 4](#_Toc168389863)

[*e.* *User Interaction:* 4](#_Toc168389864)

[***b)*** ***Non-Functional Requirements:*** 4](#_Toc168389865)

[*a.* *Performance:* 4](#_Toc168389866)

[*b.* *Reliability:* 5](#_Toc168389867)

[*c.* *Security:* 5](#_Toc168389868)

[4. **SYSTEM** **DESIGN** 5](#_Toc168389869)

[**5.** **IMPLEMENTATION** 5](#_Toc168389870)

[**6.** **DEPLOYMENT** 6](#_Toc168389871)

[*1.System Requirements:* 6](#_Toc168389872)

[*2. Compilation and Build:* 6](#_Toc168389873)

[*3. Configuration:* 6](#_Toc168389874)

[*4. Permissions and Execution:* 6](#_Toc168389875)

[*5. Monitoring and Management:* 6](#_Toc168389876)

[*6. Logging and Analysis:* 6](#_Toc168389877)

# **INTRODUCTION**

This C program is designed to capture network packets and analyze them based on their protocol type. It uses raw socket programming to intercept packets at a low level, allowing for detailed analysis. The program can differentiate between TCP and UDP packets, extract relevant information from their headers, and log this information into separate files for further analysis.

# **OBJECTIVES**

The main objective of the program is to provide a tool for network administrators and security professionals to monitor and analyze network traffic. By capturing packets and logging information such as source IP, destination IP, packet size, and protocol type, the program enables users to identify patterns, anomalies, and potential security threats on their networks.

# **REQUIREMENTS**

# ***Functional Requirements:***

# *Packet Capture Functionality:*

* The program should be able to capture raw network packets from the underlying network interface.
* It should use raw socket programming to intercept packets at a low level, bypassing the TCP/IP stack.

## *Protocol Identification:*

* The program should differentiate between TCP and UDP packets based on their protocol numbers in the IP header.
* It should process each packet and determine its protocol type for further analysis.

## *Header Parsing:*

* Once a packet's protocol is identified, the program should parse relevant information from its header.
* For TCP packets, it should extract source IP, destination IP, packet size, source port, and destination port.
* For UDP packets, similar information should be extracted from the UDP header.

## *Logging Functionality:*

* The program should write the parsed information into separate log files for TCP and UDP packets.
* Each log entry should include source IP, destination IP, packet size, protocol type, source port, and destination port.

## *User Interaction:*

* Provide an interactive shell for user interaction, allowing users to select protocols and choose how to print captured data.
* The shell should accept user input to select protocols (TCP, UDP) and print data in different formats (histogram, table).

# ***Non-Functional Requirements:***

## *Performance:*

* The program should efficiently capture and process network packets without significant performance overhead.
* It should handle high traffic volumes and process packets in real-time to provide timely insights.

## *Reliability:*

* The program should be robust and resilient to handle unexpected inputs, errors, or interruptions gracefully.
* It should not crash or malfunction under normal operating conditions.

## *Security:*

* Since the program deals with raw network packets, proper error handling and input validation mechanisms should be implemented to prevent buffer overflows or other security risks.

# **SYSTEM** **DESIGN**

1. ***Packet Capture Module****:* Utilizes raw socket programming to intercept network packets.
2. ***Packet Processing Module:*** Identifies packet protocols (TCP, UDP) and parses header information.
3. ***Logging Module****:* Writes parsed information to separate log files (**tcp.txt** and **udp.txt**).
4. ***User Interface Module****:* Offers an interactive shell for user interaction, allowing protocol selection and data printing options.
5. ***Printing Module****:* Provides options to print captured data as a text-based histogram or a tabular format.

# **IMPLEMENTATION**

* The program is implemented in C and utilizes standard libraries for socket programming *(****<sys/socket.h>****,* ***<netinet/in.h>***), file handling (***<stdio.h>***), string manipulation (***<string.h>***), and threading (***<pthread.h>***).
* It uses raw sockets to capture network packets and processes them based on the Internet Protocol (IP) header and Transport Layer protocol (TCP/UDP) headers.
* The user interface is implemented using a combination of command-line input and output.

# **DEPLOYMENT**

## *1.System Requirements:*

* Ensure the deployment environment meets the minimum system requirements, including operating system compatibility and sufficient resources (CPU, memory, disk space).
* Supported operating systems typically include Linux distributions such as Ubuntu, CentOS, or Debian.

## *2. Compilation and Build:*

* 1. *Compile the Source Code:*
* Clone or download the source code from the repository to the deployment server.
* Compile the program using a C compiler such as GCC (**gcc -o packet\_sniffer packet\_sniffer.c -pthread**).

## *3. Configuration:*

* 1. *Set Command-Line Arguments:*
* Determine which protocols to capture by providing appropriate command-line arguments (e.g., **1** for TCP, **2** for UDP).
* Optionally, specify other parameters such as file paths or logging options.

## *4. Permissions and Execution:*

* 1. *Set Permissions:*
* Ensure that the user running the program has necessary permissions to access network interfaces and write to log files.
* Grant elevated privileges using **sudo** if required (**sudo ./packet\_sniffer <arguments>**).

## *5. Monitoring and Management:*

* 1. *Monitor Terminal Output:*
* Start the program and monitor the terminal output for real-time packet capture and analysis.
* Verify that packets are being captured and processed according to the specified protocols.

## *6. Logging and Analysis:*

* 1. *Review Log Files:*
* Check the generated log files (**tcp.txt** and **udp.txt**) to inspect captured packets and protocol-specific details.
* Analyze the logged data using text editors or command-line tools for further examination and insights.