Programming in C – Project Report

University of Petroleum and Energy Studies

Instructor: Dr. Tanu Singh
Student: Vaibhav Singh
SAP ID: 590025376
Project Title: Network Speed Monitoring System
Date: December 2025

Abstract

This project presents a command-line network speed monitoring system written in C for Linux. It measures real-time internet upload and download speed by reading network interface statistics from the `/proc/net/dev` file. The program samples byte transfer data every second for ten seconds, calculates network throughput in Mbps, and displays a visual ASCII graph representing the network speed over time. This project demonstrates concepts of system programming, Linux file system interaction, data computation, loops, arrays, and formatted terminal visualization.

Problem Definition

Most commonly used graphical network monitoring tools show averaged or delayed data and do not allow lightweight monitoring from the terminal. Developers, network engineers, researchers, and students working on performance analysis often need real-time command-line based network monitoring without using external GUI applications.
This project provides a simple, resource-light solution that displays real-time bandwidth usage numerically and visually through a dynamic ASCII bar graph.

System Design

Working Principle

- The program reads received (`rx`) and transmitted (`tx`) network bytes of a selected network interface from `/proc/net/dev`.
- It takes an initial sample, waits one second, takes another sample, and calculates the difference.
- The difference is converted into bits-per-second and then displayed in Mbps.
- A bar graph visual representation is printed for each second.

- The program records speeds for ten seconds and calculates the average.

Flowchart

```
┌──────────────┐
│    START     │
└──────┬───────┘
       │
┌──────▼───────┐
│ Enter iface  │
└──────┬───────┘
       │
┌──────▼───────────┐
│ Read initial RX TX│
└──────┬───────────┘
       │
┌──────▼───────┐
│ Loop 10 sec  │
└──────┬───────┘
       │
┌──────▼─────────────────────────────┐
│ Sleep 1s, Read RX TX, Calc Mbps    │
│ Display ASCII speed bar            │
└──────┬─────────────────────────────┘
       │
┌──────▼───────┐
│ Compute Avg  │
└──────┬───────┘
       │
```

```
┌──────▼──────┐
│   Display   │
└──────┬──────┘
       │
    ┌──▼─────────┐
    │    END     │
    └────────────┘
```

Algorithm

1. Start
2. Ask the user to input interface name (e.g., `eth0`, `wlan0`, `enp3s0`)
3. Read initial RX and TX values from `/proc/net/dev`
4. For `i = 1 to 10`:
    - Wait 1 second
    - Read RX and TX again
    - Compute difference between current and previous
    - Convert to Mbps
    - Display time, Mbps, and ASCII graph bar
5. Compute the average Mbps
6. Display the average and exit
7. Stop

Implementation Details

- Programming language: C
- Operating system: Linux
- IDE / Editor: VS Code
- Input source: `/proc/net/dev`

- Key programming concepts used:
    - File parsing
    - Loops and conditional statements
    - Numeric calculations and arrays
    - Terminal ASCII visualization
- Output format: Console bar graph

Testing & Results

- Tested on `wlp3s0` Wi-Fi and `enp2s0` Ethernet interfaces
- Verified speed changes dynamically when downloading files or streaming video
- Bar graph successfully visualizes real-time throughput variations
- Average calculated correctly from all samples

Sample output format:

```
t =  1s |  12.53 Mbps | ############
t =  2s |   8.21 Mbps | ########
t =  3s |  17.02 Mbps | #################
Average speed over 10 seconds: 11.42 Mbps
```

Conclusion

The Network Speed Monitoring System successfully displays real-time bandwidth usage using C and Linux system interfaces. It offers a lightweight and educational alternative to GUI-based tools and demonstrates practical system programming concepts.
Future enhancements may include:

- Continuous monitoring until pressing `q`
- ncurses-based real-time UI display
- Separate upload/download graphing
- User-configurable sample duration

References

- Linux `/proc` filesystem documentation
- GNU GCC compiler documentation
- System programming tutorials for C

Appendix – Source Code

```
// netspeed_graph.c
// Live ASCII graph of network speed on Linux using /proc/net/dev
// Samples every 1 second for 10 seconds and prints a bar graph.
//
// Compile: g++ netspeed_graph.c -o netspeed_graph
// Run:     ./netspeed_graph

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

#define TOTAL_TIME 10       // total duration in seconds
#define SAMPLE_INTERVAL 1   // sample every 1 second
#define MAX_BAR_WIDTH 50    // characters in graph bar

// Read network bytes for a given interface
int read_bytes(const char *iface, unsigned long long *rx, unsigned long long *tx) {
   FILE *fp = fopen("/proc/net/dev", "r");
   if (!fp) {
        perror("Error opening /proc/net/dev");
        return -1;
    }

    char line[256];
    *rx = *tx = 0;

    while (fgets(line, sizeof(line), fp)) {
        // Look for the interface name followed by ":" to avoid partial matches
        char *pos = strstr(line, iface);
        if (pos && pos == line + strspn(line, " ")) { // interface at start (ignoring spaces)
            // Format in /proc/net/dev:
            // Inter-|   Receive                                    |  Transmit
            //  face |bytes packets errs drop fifo frame compressed multicast|bytes packets errs
drop fifo colls carrier compressed
            char iface_name[32];
            unsigned long long rbytes, rpackets, rerrs, rdrop, rfifo, rframe, rcompressed,
rmulticast;
            unsigned long long tbytes, tpackets, terrs, tdrop, tfifo, tcolls, tcarrier, tcompressed;

            int n = sscanf(line,
                           " %[^:]: %llu %llu %llu %llu %llu %llu %llu %llu %llu %llu %llu %llu %llu
%llu %llu %llu",
                           iface_name,
                           &rbytes, &rpackets, &rerrs, &rdrop, &rfifo, &rframe, &rcompressed,
&rmulticast,
                           &tbytes, &tpackets, &terrs, &tdrop, &tfifo, &tcolls, &tcarrier, &tcompressed);
            if (n >= 10) { // we at least got rbytes and tbytes
                *rx = rbytes;
                *tx = tbytes;
                fclose(fp);
                return 0;
            }
        }
    }

    fclose(fp);
    return -1;
}

// Draw a horizontal bar for a given Mbps value
void draw_bar(double mbps) {
    // Simple scaling: 1 Mbps -> 1 char (you can tweak)
    int width = (int)mbps;
    if (width > MAX_BAR_WIDTH) width = MAX_BAR_WIDTH;
    if (width < 0) width = 0;

    for (int i = 0; i < width; i++) {
        putchar('#');
    }
    putchar('\n');
}
```

```
int main() {
  char iface[32];
  printf("Enter network interface (e.g. eth0, wlan0, enp3s0): ");
  if (scanf("%31s", iface) != 1) {
    printf("Invalid input.\n");
    return 1;
  }

  unsigned long long prev_rx, prev_tx, cur_rx, cur_tx;

  if (read_bytes(iface, &prev_rx, &prev_tx) != 0) {
    printf("Interface '%s' not found or error reading /proc/net/dev.\n", iface);
    return 1;
  }

  int samples = TOTAL_TIME / SAMPLE_INTERVAL;
  double speeds[samples];
  double sum_mbps = 0.0;

  printf("\nMeasuring for %d seconds...\n", TOTAL_TIME);
  printf("Each '#' is ~1 Mbps (capped at %d chars)\n\n", MAX_BAR_WIDTH);

  for (int i = 0; i < samples; i++) {
    sleep(SAMPLE_INTERVAL);

    if (read_bytes(iface, &cur_rx, &cur_tx) != 0) {
      printf("Error: failed to read /proc/net/dev during sampling.\n");
      return 1;
    }

    unsigned long long rxdiff = (cur_rx >= prev_rx) ? (cur_rx - prev_rx) : 0;
    unsigned long long txdiff = (cur_tx >= prev_tx) ? (cur_tx - prev_tx) : 0;

    prev_rx = cur_rx;
    prev_tx = cur_tx;

    double interval = (double)SAMPLE_INTERVAL;
    double total_bits = (double)(rxdiff + txdiff) * 8.0;
    double mbps = total_bits / (interval * 1000000.0);

    speeds[i] = mbps;
    sum_mbps += mbps;

    printf("t = %2ds | %7.2f Mbps | ", (i + 1) * SAMPLE_INTERVAL, mbps);
    draw_bar(mbps);
  }

  double avg_mbps = sum_mbps / samples;
  printf("\nAverage speed over %d seconds: %.2f Mbps\n", TOTAL_TIME, avg_mbps);

  return 0;
}
```