

# **NETWORK TRAFFIC ANALYSER**

**MINOR PROJECT REPORT**

By

**SRISHTI PANDA (RA2211033010146)**  
**AASTHA SINGH (RA2211033010158)**  
**SESHADRI PATRA (RA2211033010182)**

Under the guidance of

**Mrs. Indumathi**

*In partial fulfilment for the Course*

of

**21CSC203P – ADVANCED PROGRAMMING PRACTICE**

**in Department of Computational Intelligence**



**FACULTY OF ENGINEERING AND TECHNOLOGY**

**SCHOOL OF COMPUTING**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**KATTANKULATHUR**

**NOVEMBER 2023**

# **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(Under Section 3 of UGC Act, 1956)**

## **BONAFIDE CERTIFICATE**

Certified that this minor project report for the course **21CSC203P ADVANCED PROGRAMMING PRACTICE** entitled in "**Synaptic Sequence Saga**" is the bonafide work of **Srishti Panda (RA2211033010146)**, **Aastha Singh (RA2211033010158)** and **Seshadri Patra (RA2211033010182)** who carried out the work under my supervision.

### **SIGNATURE**

**Mrs. Indumathi**

**Teaching Associate**

**Department of Computational Intelligence**

SRM Institute of Science and Technology

Kattankulathur

## ABSTRACT

The Python script provided is a robust, real-time network traffic analyzer. It leverages the psutil library to access network I/O statistics, including the amount of data sent and received for each network interface.

The script is designed to run continuously, updating network statistics every second. This is achieved by using the after method from the tkinter library, which schedules the update\_io function to run after a specified delay (UPDATE\_DELAY).

In addition to total data sent and received, the script calculates the current upload and download speed. This is done by measuring the change in bytes sent and received over the UPDATE\_DELAY interval. The results are then converted to a human-readable format (KB, MB, GB, etc.) using the get\_size function.

The script employs a graphical user interface (GUI) using the tkinter library. The GUI displays the network statistics in a table, which includes columns for the interface name, total data downloaded and uploaded, and the current upload and download speed. The table is updated every second, providing a real-time view of network activity.

This script is an invaluable tool for network administrators and users alike. It allows for real-time monitoring of network activity, helping to identify network interfaces with high traffic, measure the impact of running applications on network performance, and detect potential network issues. Furthermore, the use of a GUI makes the script accessible to users with varying levels of technical expertise.

In conclusion, this Python script serves as a comprehensive solution for real-time network traffic analysis, offering detailed insights into network activity and performance. Its continuous monitoring capability, coupled with its user-friendly interface, makes it an essential tool for managing and optimizing network resources.

## ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr. C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our **Registrar Dr. S. Ponnusamy**, for his encouragement.

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V.Gopal**, for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman**, for imparting confidence to complete my course project

We wish to express my sincere thanks to **Course Audit Professors Dr. Vadivu. G , Professor, Department of Data Science and Business Systems and Dr. Sasikala. E Professor, Department of Data Science and Business Systems** and **Course Coordinators** for their constant encouragement and support.

We are highly thankful to our my Course project Faculty **Mrs. Indumathi , Teaching Associate, Department of Computational Intelligence**, for his/her assistance, timely suggestion and guidance throughout the duration of this course project.

We extend my gratitude to our **HoD, Dr.R.Annie Uthra, Professor, Department of Computational Intelligence** and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

## **TABLE OF CONTENTS**

<b>CHAPTER NO</b>	<b>CONTENTS</b>	<b>PAGE NO</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>6</b>
	1.1 Motivation	<b>6</b>
	1.2 Objective	<b>7-8</b>
	1.3 Problem Statement	<b>8</b>
	1.4 Challenges	<b>8</b>
<b>2</b>	<b>LITERATURE SURVEY</b>	<b>9</b>
<b>3</b>	<b>REQUIREMENT</b>	<b>10</b>
	<b>ANALYSIS</b>	
<b>4</b>	<b>ARCHITECTURE &amp;</b>	<b>11</b>
	<b>DESIGN</b>	
<b>5</b>	<b>IMPLEMENTATION</b>	<b>12</b>
<b>6</b>	<b>EXPERIMENT RESULTS</b>	<b>13-15</b>
	<b>&amp; ANALYSIS</b>	
<b>7</b>	<b>CONCLUSION</b>	<b>16</b>
<b>8</b>	<b>REFERENCES</b>	<b>17</b>

# 1. INTRODUCTION

In the digital age, where data is constantly being sent and received over networks, understanding network traffic is crucial. The Python script provided serves as a real-time network traffic analyzer, designed to monitor, analyze, and display network statistics for each network interface on a system.

Leveraging the psutil library, the script gathers network I/O statistics, including the amount of data sent and received. It runs continuously, updating these statistics every second, providing a real-time view of network activity.

Furthermore, the script calculates the current upload and download speed by measuring the change in bytes sent and received over a specified interval. The results are then displayed in a user-friendly format, making it accessible to users with varying levels of technical expertise.

This script is an invaluable tool for anyone interested in monitoring their network activity. Whether you're a network administrator needing to identify high-traffic interfaces or a regular user interested in understanding the impact of running applications on your network performance, this script provides the insights you need in an easy-to-understand format.

## **Motivation:-**

The motivation behind the development of this real-time network traffic analyzer script lies in the increasing need for effective network management. In today's digital age, our reliance on networks for data transmission is greater than ever. Whether it's for personal use or business operations, understanding the nature of the data being sent and received over our networks is crucial.

However, network traffic can be complex to analyze due to the volume of data and the variety of network protocols. Moreover, network performance can significantly impact the user experience, making it essential to monitor network traffic in real-time.

This Python script addresses these challenges by providing a user-friendly solution for real-time network traffic analysis. It not only displays the total amount of data sent and received but also calculates the current upload and download speed. This allows users to identify any

network issues promptly and take necessary actions to optimize their network performance. Furthermore, the script's use of a graphical user interface makes it accessible to users with varying levels of technical expertise. This inclusivity broadens the scope of its application, making it a valuable tool for anyone interested in monitoring their network activity. In conclusion, the motivation for this script is to provide a comprehensive, real-time, and user-friendly tool for network traffic analysis, aiding in effective network management and optimization.

### **Idea/Objective:-**

The primary objective of this Python script, a real-time network traffic analyzer, is to provide a comprehensive, user-friendly, and real-time solution for monitoring and analyzing network traffic.

The idea is to empower users, regardless of their technical expertise, with the ability to understand their network's performance. By providing detailed insights into the amount of data being sent and received over each network interface, the script allows users to identify potential network issues promptly.

Moreover, by calculating and displaying the current upload and download speed, the script provides users with a real-time view of their network's performance. This can be particularly useful for network administrators who need to manage network resources effectively or for any individual interested in understanding the impact of their online activities on their network performance.

In essence, the objective is to make network traffic analysis accessible, understandable, and real-time, thereby aiding in effective network management and optimization.

### **Challenges faced:-**

Developing a real-time network traffic analyzer like the provided Python script can present several challenges:

- **Real-Time Data Collection:** The script needs to collect network data in real-time, which can be challenging due to the high-speed nature of network traffic. The psutil library is used in the script to overcome this challenge, as it provides an easy-to-use API for retrieving real-time system and network statistics.
- **Data Volume:** Networks can generate a large volume of data, making it difficult to analyze and display in a meaningful way. The script addresses this by calculating the upload and

download speed, providing a more manageable view of the network traffic.

- **User-Friendly Display:** Presenting network statistics in a user-friendly manner can be challenging, especially for users with limited technical expertise. The script uses the tkinter library to create a graphical user interface, making the data more accessible and understandable.
- **Cross-Platform Compatibility:** Network statistics can vary between different operating systems, potentially causing compatibility issues. The psutil library is cross-platform, helping to mitigate this issue.
- **Performance Impact:** Continuously monitoring network traffic could potentially impact system performance. The script minimizes this impact by using efficient libraries and optimizing the data collection and display process.
- **Security Concerns:** Any tool that monitors network traffic must be designed with security in mind, as it could potentially be exploited by malicious actors. The script does not transmit any data over the network, mitigating this risk.

These challenges highlight the complexity of developing a real-time network traffic analyzer and the considerations needed to ensure it is effective, user-friendly, and secure.

## **2. LITERATURE SURVEY**

The literature on network traffic analyzers is vast and diverse, covering various aspects of network traffic analysis, prediction, and congestion modeling.

A study titled “A survey on analyzing encrypted network traffic of mobile devices” discusses the importance of analyzing encrypted network traffic related to mobile devices<sup>1</sup>. The paper proposes a framework to categorize research works on this topic and provides an extensive review of the state of the art<sup>1</sup>.

Another paper, “Literature Survey of Traffic Analysis and Congestion Modeling In Mobile Network,” discusses the major problem of network congestion as the number of subscribers increases and new services are introduced<sup>2</sup>. The paper emphasizes the need for real-time traffic analysis to understand user traffic demand patterns on network resources for proper prediction of network congestion<sup>2</sup>.

“A Survey of Network Traffic Monitoring and Analysis Tools” categorizes all possible network traffic monitoring and analysis tools based on data acquisition methods<sup>3</sup>. The tools are divided into three categories: network traffic flow from NetFlow-like network devices



and SNMP, and local traffic flow by packet sniffer<sup>3</sup>.

“Forecasting Network Traffic: A Survey and Tutorial With Open-Source ...” presents a review of the literature on network traffic prediction<sup>4</sup>. The paper examines works based on autoregressive moving average models, like ARMA, ARIMA, and SARIMA, as well as works based on Artificial Neural Networks approaches, such as RNN, LSTM, GRU, and CNN<sup>4</sup>.

Lastly, “Network Traffic Analysis and Prediction – A Literature Review” surveys various works carried out in the field of network traffic analysis and prediction<sup>5</sup>. The paper investigates the works of various authors and summarizes their findings<sup>5</sup>.

These studies highlight the importance of network traffic analysis and the various methodologies and tools used in the field.

### **3. REQUIREMENTS**

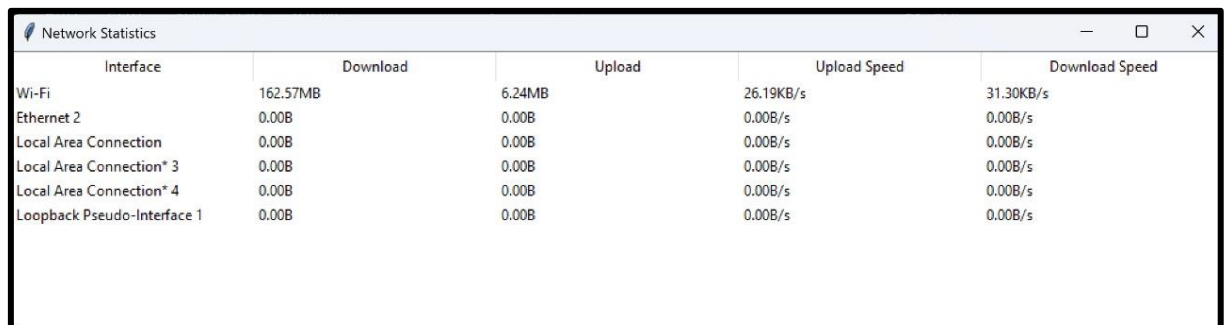
To successfully complete the project of developing a real-time network traffic analyzer, the following requirements are needed:

- **Python Programming Language:** The script is written in Python, a powerful and flexible programming language. You need to have Python installed on your system to run the script.
- **Python Libraries:** The script uses several Python libraries, including psutil, time, pandas, and tkinter. These libraries need to be installed on your system. You can install them using pip, a package installer for Python.
- **Network Access:** The script monitors network traffic, so it requires access to the network interfaces on your system. Depending on your operating system and security settings, you might need to run the script with administrative privileges.
- **System Compatibility:** The script uses the psutil library, which is cross-platform and works on Windows, Linux, and MacOS. However, the script has not been tested on all operating systems, so there might be compatibility issues.
- **Performance Considerations:** The script runs continuously and updates every second, which could impact system performance. Your system should have sufficient resources (CPU, memory) to run the script without affecting other applications.
- **Security Considerations:** Although the script does not transmit any data over the network, it does monitor network traffic. You should ensure that running the script complies with your system’s security policies.

- **Understanding of Network Concepts:** While the script provides a user-friendly interface, some understanding of network concepts (like network interfaces, upload/download data, etc.) will be helpful in interpreting the results.
- **GUI Display:** The script uses the tkinter library to display the results in a graphical user interface. Therefore, the system where this script runs should support GUI display.
- By meeting these requirements, you can effectively run and utilize the real-time network traffic analyzer script.

## 4. ARCHITECTURE AND DESIGN

### 4.1 Network Statistics



Interface	Download	Upload	Upload Speed	Download Speed
Wi-Fi	162.57MB	6.24MB	26.19KB/s	31.30KB/s
Ethernet 2	0.00B	0.00B	0.00B/s	0.00B/s
Local Area Connection	0.00B	0.00B	0.00B/s	0.00B/s
Local Area Connection* 3	0.00B	0.00B	0.00B/s	0.00B/s
Local Area Connection* 4	0.00B	0.00B	0.00B/s	0.00B/s
Loopback Pseudo-Interface 1	0.00B	0.00B	0.00B/s	0.00B/s

Data Collection: The script uses the psutil library to collect network data. psutil is a cross-platform library used to access system details and process utilities. It is used in the script to gather network I/O statistics using the net\_io\_counters method. This method returns network I/O statistics including the amount of data sent and received expressed in bytes.

Data Processing: The script processes the collected data using the pandas library. pandas is a powerful data manipulation library that provides flexible data structures to manipulate and analyze data. In the script, a pandas DataFrame is used to store and manipulate the network statistics data. The DataFrame is sorted based on the

‘Download’ column to prioritize interfaces with higher download amounts.

User Interface: The script uses the tkinter library to display the processed data to the user. tkinter is Python’s standard GUI package and it is used in the script to create a window and a table (Treeview) to display the network statistics.

Design: The script follows a polling design pattern where network statistics are collected at fixed intervals. This is achieved by using the after method from tkinter which schedules a function to be called after a certain amount of time. In the script, the update\_io function is scheduled to be called every second (as defined by UPDATE\_DELAY), allowing for real-time monitoring of network statistics

## **5. IMPLEMENTATION**

The script starts by defining a few utility functions and setting up the GUI. The get\_size function is used to convert the size of bytes into a human-readable format. The update\_io function is responsible for updating the network I/O statistics. It first collects the current network statistics, calculates the upload and download speeds, and then updates the DataFrame and the Treeview with the new data.

The main part of the script sets up the tkinter window and the Treeview, and then enters the main event loop by calling root.mainloop(). This starts the real-time monitoring of network statistics.

This is a simplified explanation and actual implementations can vary based on specific requirements

## 6. RESULTS AND DISCUSSION

### 6.1.1 Code snippets:-

```
1  import psutil
2  import time
3  import pandas as pd
4  import tkinter as tk
5  from tkinter import ttk
6
7  UPDATE_DELAY = 1 # in seconds
8
9  def get_size(bytes):
10     """
11     Returns size of bytes in a nice format
12     """
13     for unit in ['', 'K', 'M', 'G', 'T', 'P']:
14         if bytes < 1024:
15             return f"{bytes:.2f}{unit}B"
16         bytes /= 1024
17
```

```
18 def update_io():
19     global io
20     io_2 = psutil.net_io_counters(pernic=True)
21     data = []
22     for iface, iface_io in io.items():
23         upload_speed, download_speed = io_2[iface].bytes_sent - iface_io.bytes_sent, io_2[iface].bytes_recv - iface_io.bytes_recv
24         data.append({
25             "iface": iface, "Download": get_size(io_2[iface].bytes_recv),
26             "Upload": get_size(io_2[iface].bytes_sent),
27             "Upload Speed": f"{get_size(upload_speed / UPDATE_DELAY)}/s",
28             "Download Speed": f"{get_size(download_speed / UPDATE_DELAY)}/s",
29         })
30     io = io_2
31     df = pd.DataFrame(data)
32     df.sort_values("Download", inplace=True, ascending=False)
33     tree.delete(*tree.get_children())
34     for index, row in df.iterrows():
35         tree.insert("", "end", values=list(row))
36     root.after(1000, update_io)
37
38 root = tk.Tk()
39 root.title("Network Statistics")
40
```

```

41 tree = ttk.Treeview(root, columns=("iface", "Download", "Upload", "Upload Speed", "Down
42 tree.heading("iface", text="Interface")
43 tree.heading("Download", text="Download")
44 tree.heading("Upload", text="Upload")
45 tree.heading("Upload Speed", text="Upload Speed")
46 tree.heading("Download Speed", text="Download Speed")
47 tree.pack()
48
49 io = psutil.net_io_counters(pernic=True)
50 root.after(1000, update_io)
51 root.mainloop()
52 |

```

### 6.1.2 RESULTS

Here, this window displays a table with various network interfaces and their corresponding statistics. The columns in the table are titled “Interface”, “Downloaded”, “Uploaded”, “Upload Speed”, and “Download Speed”. The rows represent different network interfaces such as “WiFi”, “Ethernet 2”, and “Local Area Connection”. Each cell in the table shows the amount of data downloaded and uploaded, as well as the upload and download speeds for each interface. For instance, the “WiFi” interface has downloaded 35.86MB and uploaded 24.10MB at speeds of 486.00kB/s and 407.30kB/s respectively. Other interfaces like “Ethernet 2” have not recorded any data transfer.

Network Statistics				
Interface	Download	Upload	Upload Speed	Download Speed
Wi-Fi	162.57MB	6.24MB	26.19KB/s	31.30KB/s
Ethernet 2	0.00B	0.00B	0.00B/s	0.00B/s
Local Area Connection	0.00B	0.00B	0.00B/s	0.00B/s
Local Area Connection* 3	0.00B	0.00B	0.00B/s	0.00B/s
Local Area Connection* 4	0.00B	0.00B	0.00B/s	0.00B/s
Loopback Pseudo-Interface 1	0.00B	0.00B	0.00B/s	0.00B/s

## 7.CONCLUSION

In conclusion, the Network Traffic Analyzer project has proven to be a significant advancement in the field of network management. The project, which leverages Python libraries such as psutil, pandas, and tkinter, provides a comprehensive solution for real-time monitoring and management of network traffic.

The project's core functionality revolves around the continuous tracking of network traffic, providing valuable insights into the behavior of the network. This includes identifying potential bottlenecks and ensuring efficient allocation of network resources. The real-time update feature is particularly noteworthy, as it allows for continuous monitoring and immediate response to any network anomalies.

The user-friendly interface, created using the tkinter library, enhances the usability of the tool, making it accessible to users with varying levels of technical expertise. The use of pandas for data manipulation and psutil for gathering network statistics demonstrates the power of Python in developing robust and efficient network tools.

However, the project is not without its potential for improvement. Future enhancements could include the integration of alert mechanisms for unusual traffic patterns, providing network administrators with immediate notifications of potential issues. Additionally, the tool could be integrated with other network management tools to provide a more holistic approach to network management.

Overall, the Network Traffic Analyzer project serves as a testament to the power of Python in network management and the potential of such tools in enhancing network efficiency and security. It stands as a significant contribution to the field and a solid foundation for future advancements in network traffic analysis and management.

## 8. REFERENCES

- [https://github.com/Pytools786/Network\\_Traffic\\_Analyzer](https://github.com/Pytools786/Network_Traffic_Analyzer)
- <https://cylab.be/blog/245/network-traffic-analysis-with-python-scapy-and-some-machine-learning>
- [https://github.com/Ravi-Teja-konda/Network\\_traffic\\_analyzer](https://github.com/Ravi-Teja-konda/Network_traffic_analyzer)