RIGA TECHNICAL UNIVERSITY

TELECOMMUNICATIONS SOFTWARE

FIRST PRACTICAL EXERCISE

Nithin Anil

231AEM008

**Task 1**

* **Packet Sniffing and Wireshark Guidebook**

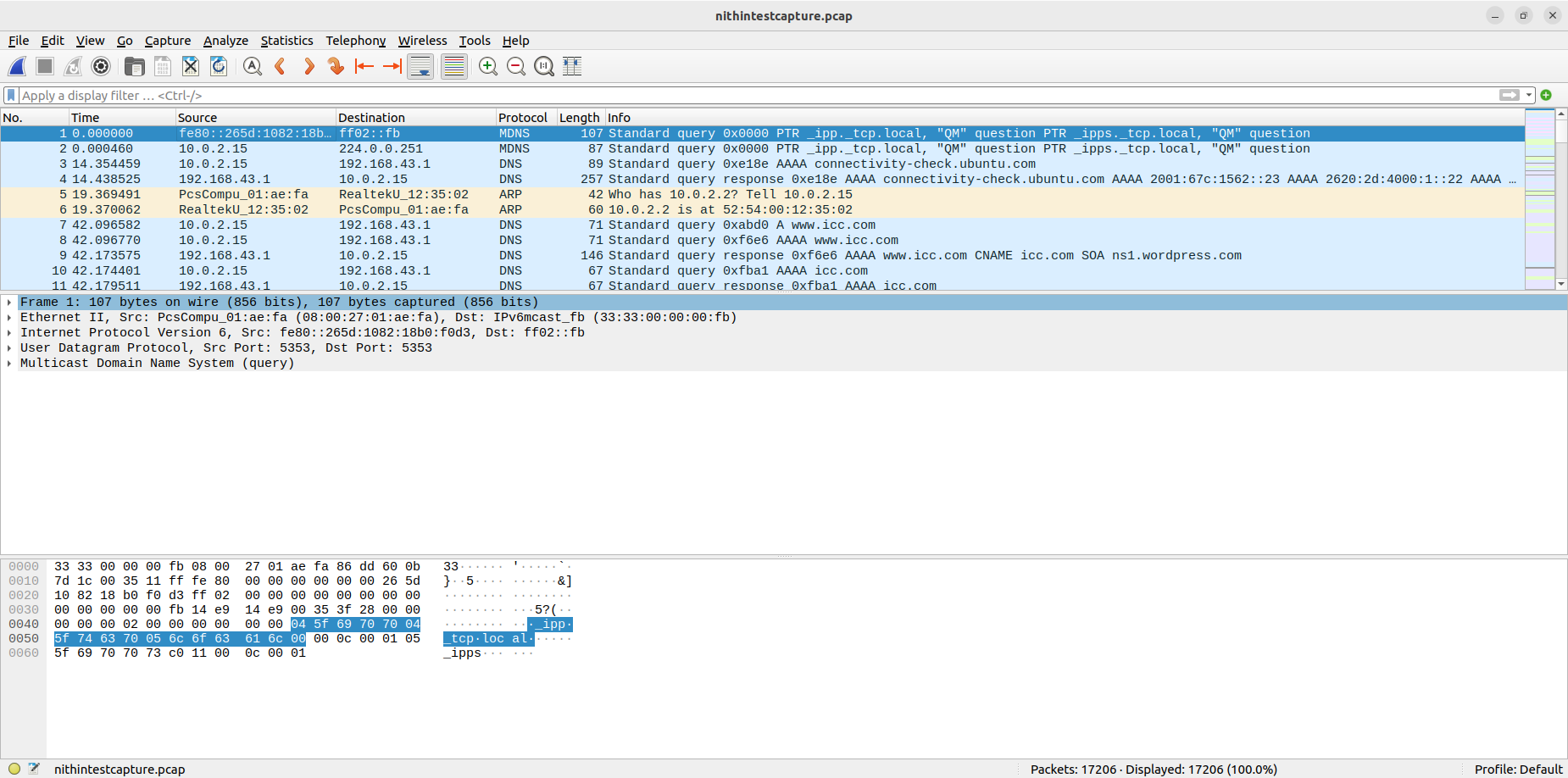
**LAB WORK REPORT**

**WIRESHARK: NETWORK TRAFFIC CAPTURE AND ANALYZE**

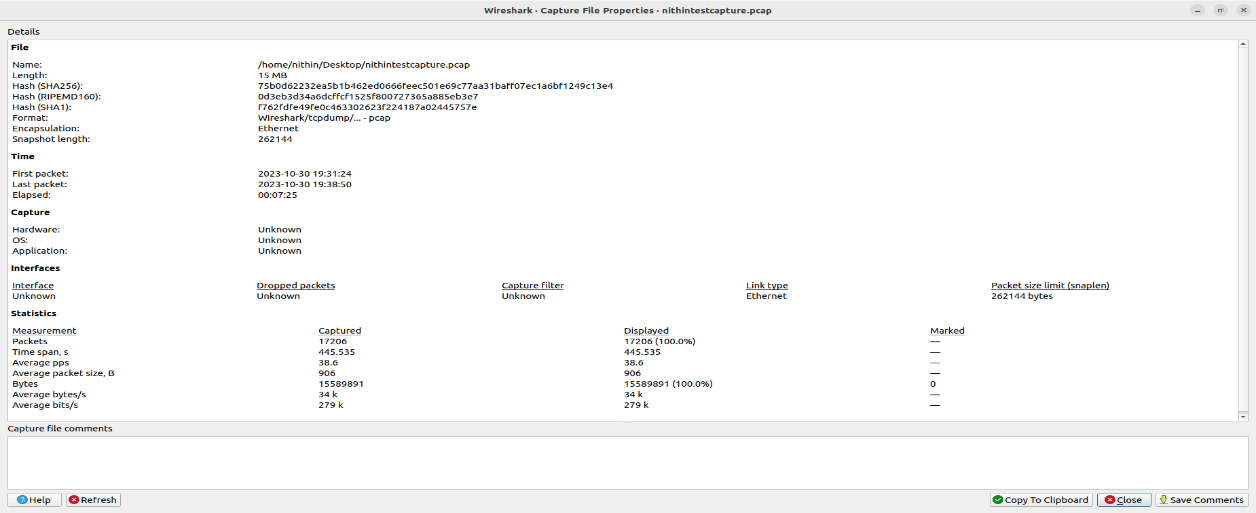
|  |  |  |
| --- | --- | --- |
| Student Name Surname: | Student ID: | Date: |
| NITHIN ANIL | 231AEM008 |  |

**My Personal PCAP file**

****

****

**1. Capture File Properties**

****

|  |  |  |
| --- | --- | --- |
| Nr | Parameter | Value |
| 1. | Time of capture, min | 00:07:25 |
| 2. | Packets | 17206 |
| 3. | Bytes, MiB | 14.88 MiB |
| 4. | Average packet size, B | 906 |
| 5. | Average packets per second, pps | 38.6 |
| 6. | Average bytes per second, B/s | 34 k |
| 7. | Relative network load L (%) | 0.2806% |

L = (Traffic [Mbits] / T [sec]) / (Bandwidth [Mbits/sec])

First, you need to convert the bytes to Mbits, given that 1 byte = 0.000008 Mbits:

Traffic [Mbits] = 15,589,891 bytes \* 0.000008 Mbits/byte = 124.719128 Mbits

Now, plug these values into the formula:

L = (124.719128 Mbits / 445.535 sec) / 100 Mbits/sec

L = (0.2806) / 100

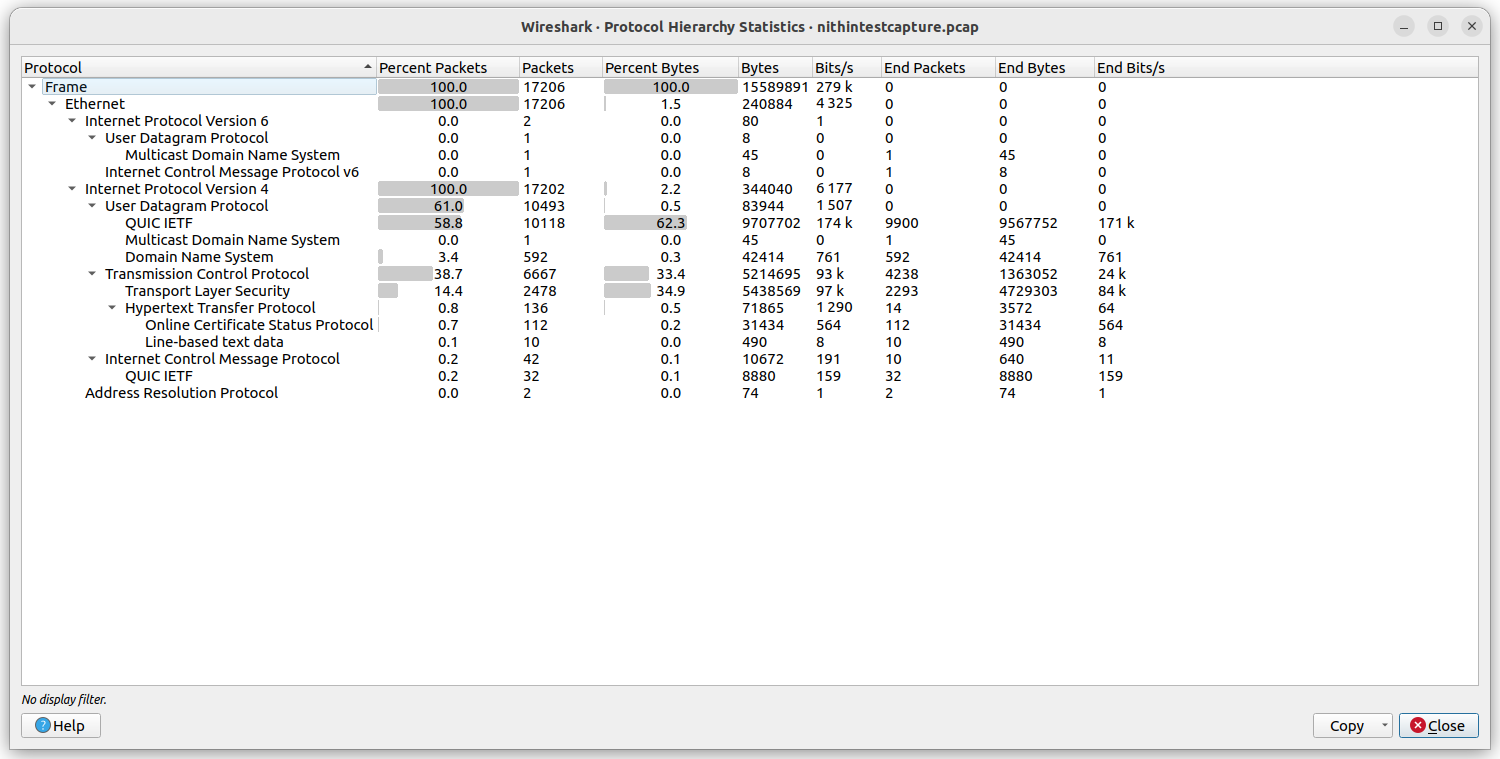
L = 0.002806

Now, to express this as a percentage, multiply by 100:

L = 0.002806 \* 100 = 0.2806%

So, in this case, the relative network load (L) is approximately 0.2806%.

**2. Ethernet Traffic Distribution by Protocols**

****

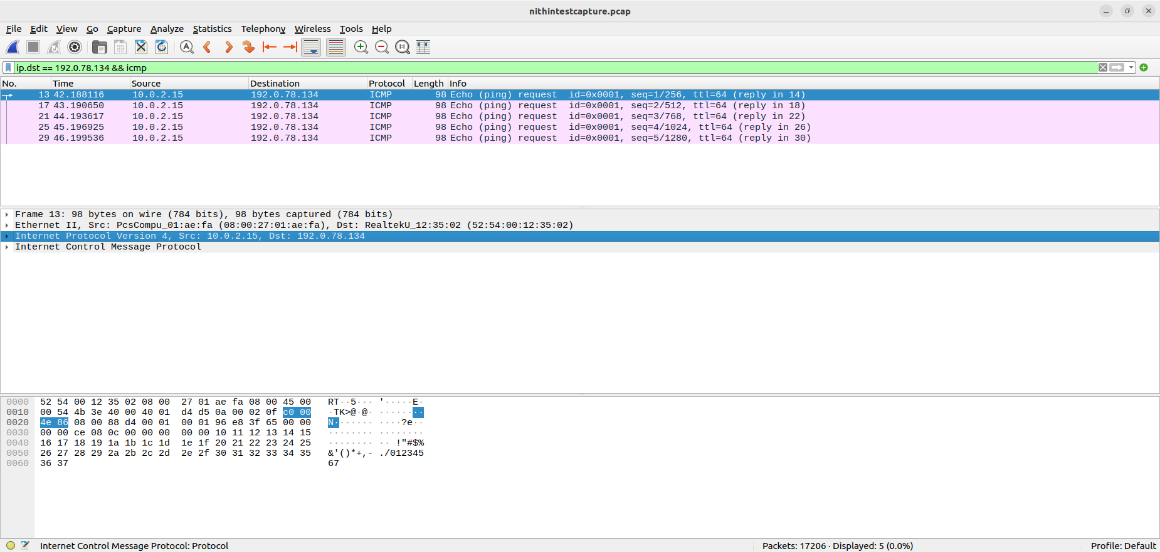
|  |  |  |  |
| --- | --- | --- | --- |
| Nr | Protocol | Traffic, МiB | Traffic, % |
| 1. | IPv6 | 0.000076 MiB | 0.0015% |
| 2. | IPv4 | 0.328 MiB | 6.39% |
| 3. | UDP | 0.080 MiB | 1.56% |
| 4. | TCP | 4.97 MiB | 96.77% |
| 5. | ICMP | 0.010 MiB | 0.20% |
| 6. | ARP | 0.000070 MiB | 0.0014% |
| 7. | 802.1X | - | - |
|  | SUM | 5.388226 MiB | 100% |

The ratio of the number of applications to the number of services = 7/6

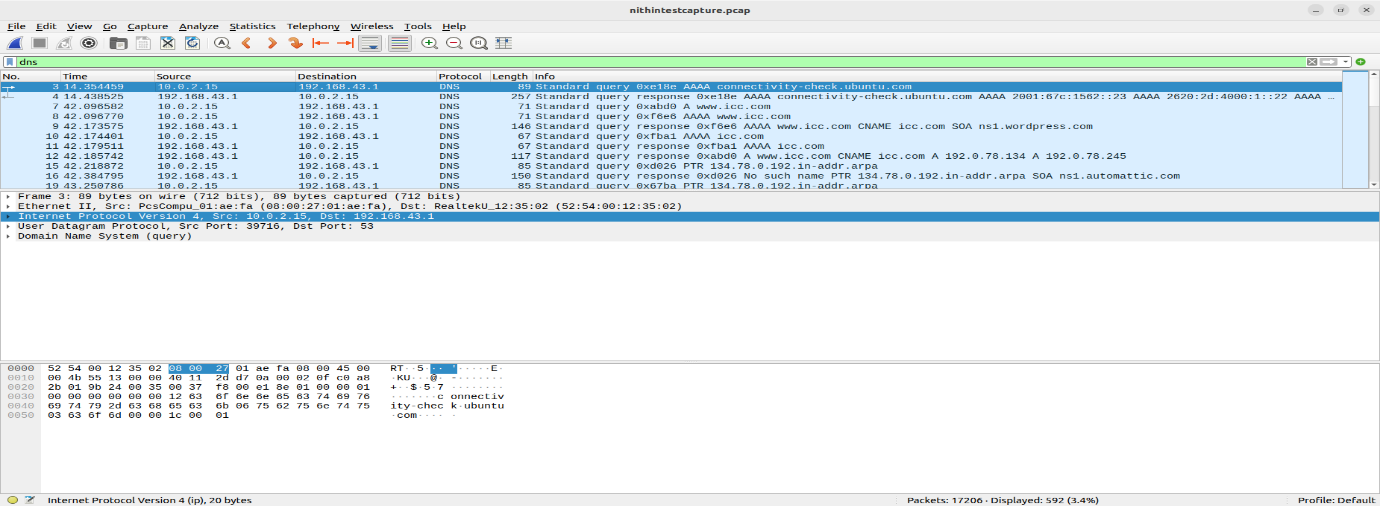
**3. Ethernet Traffic Distribution by Nodes**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nr | MAC-address | IP- address | Traffic | | | | | |
| Rx input | | Tx output | | Overall | |
| МiB | % | МiB | % | МiB | % |
| 1. |  |  |  |  |  |  |  |  |
| 2. |  |  |  |  |  |  |  |  |
| 3. |  |  |  |  |  |  |  |  |
| 4. |  |  |  |  |  |  |  |  |
| 5. |  |  |  |  |  |  |  |  |
|  | Sum | |  | 100 |  | 100 |  | 100 |

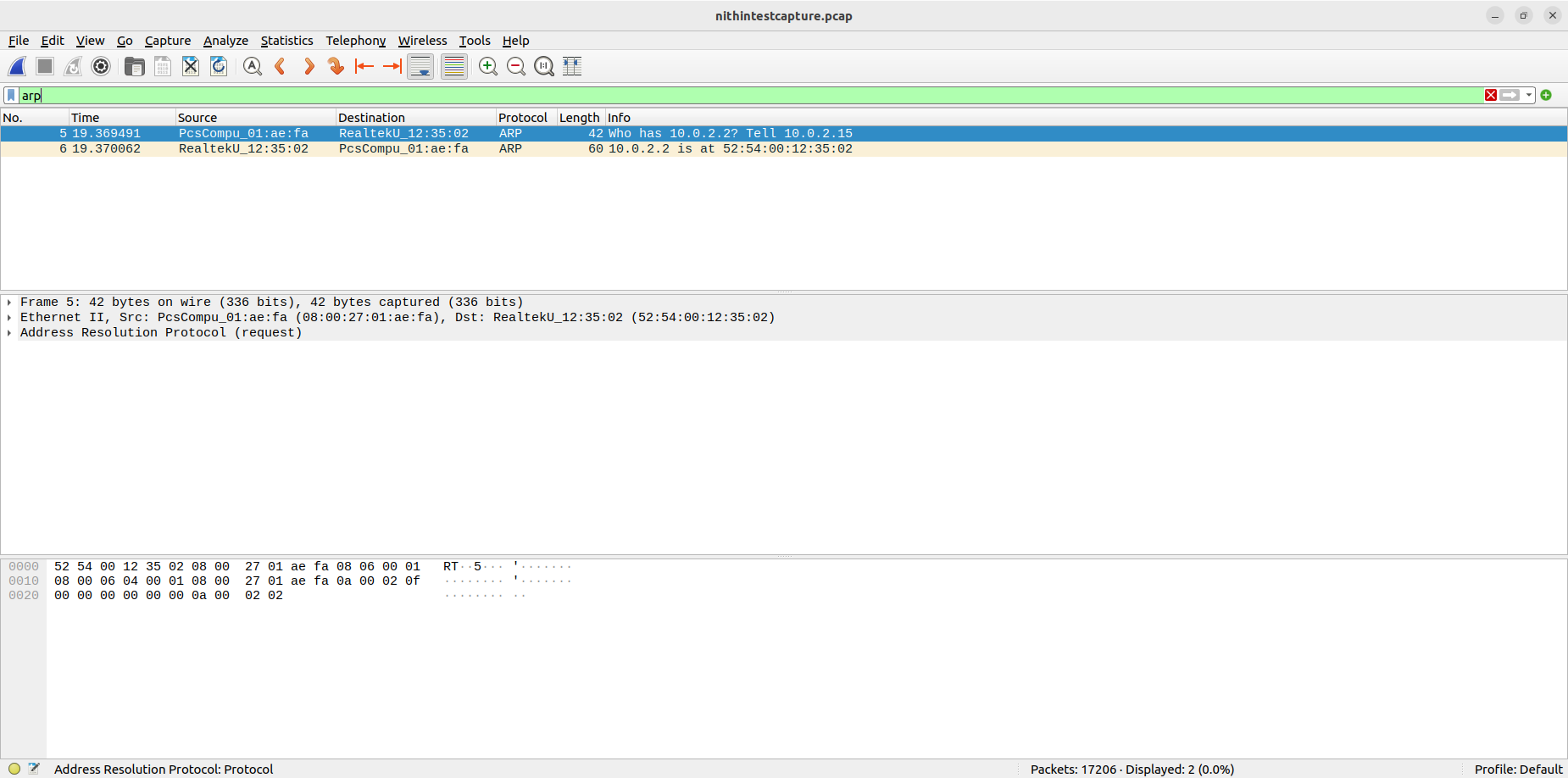
**4. Display Filters**

****

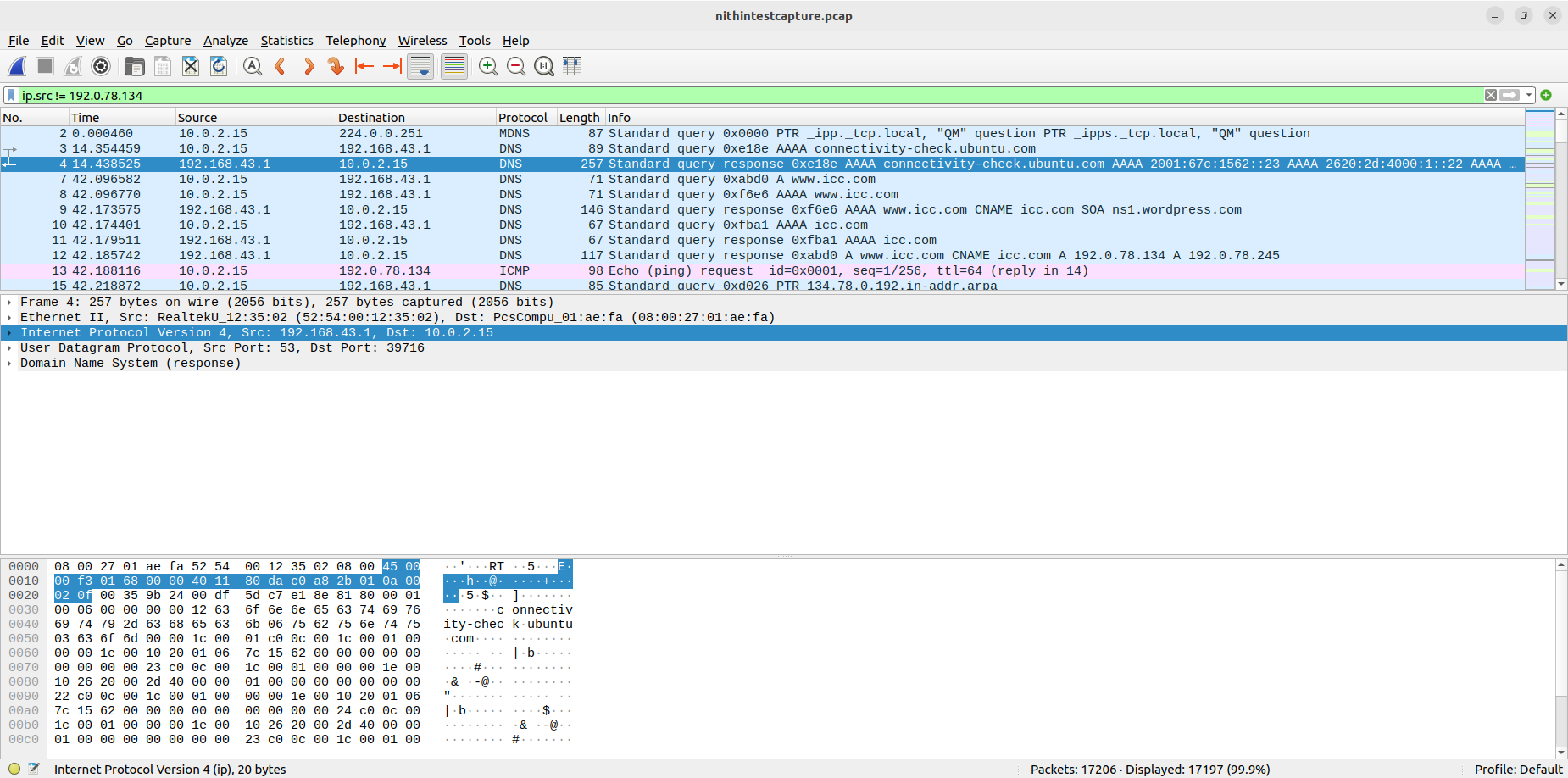
**ip.dst == 192.0.78.134 && ICMP**

****

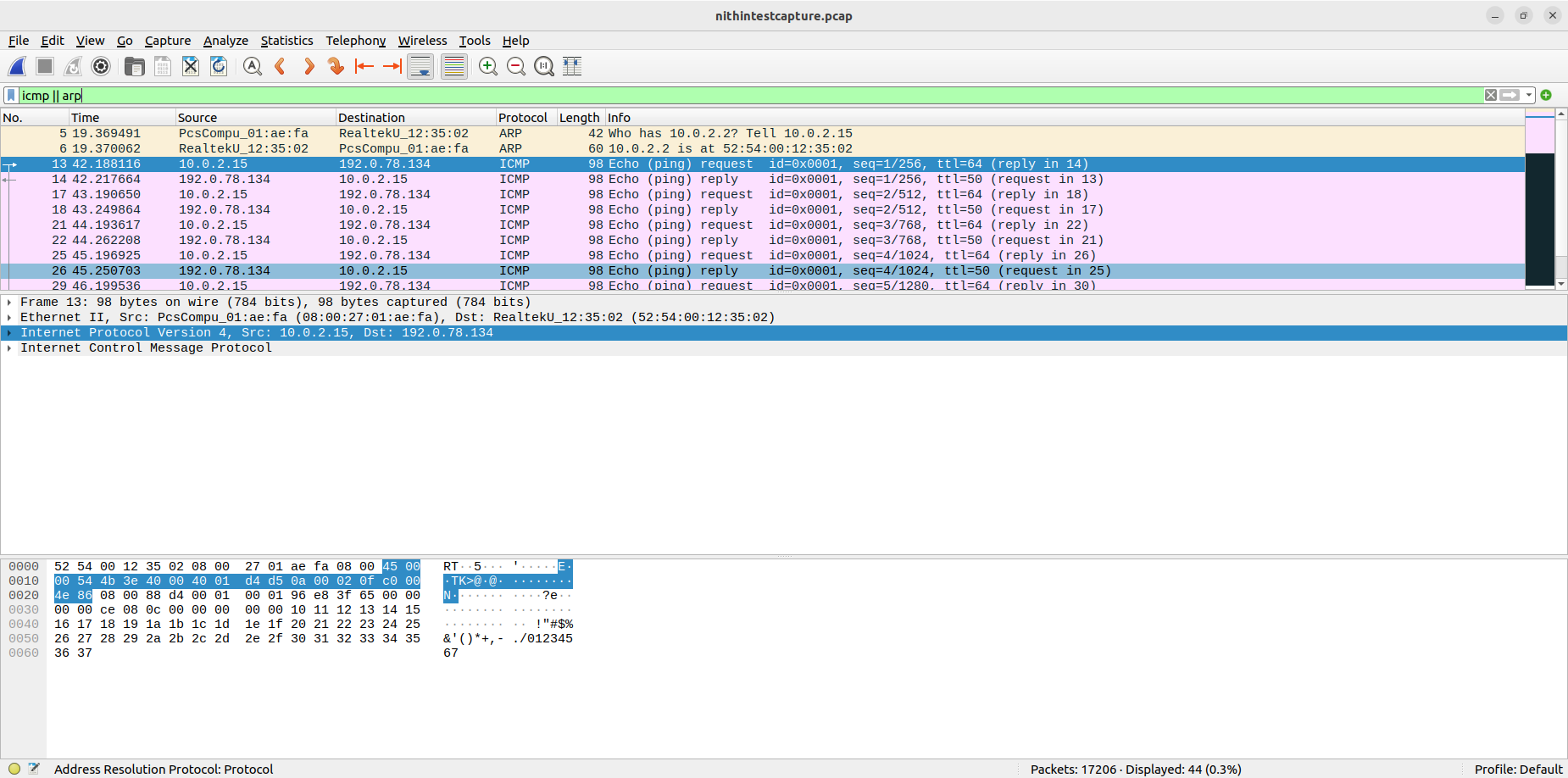
**DNS**

****

**ARP**

****

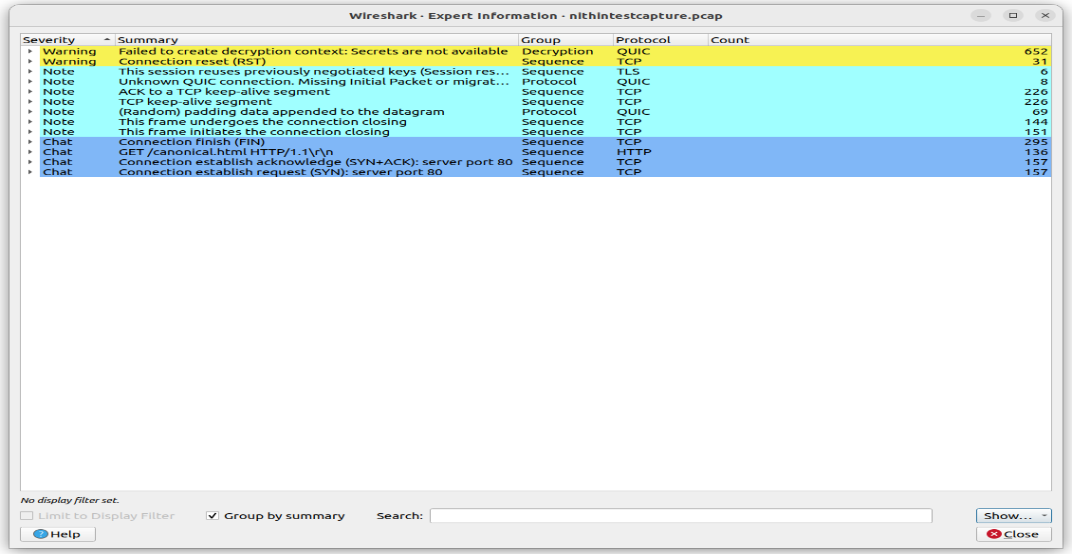
**ip.src != 192.0.78.134**

****

**ICMP || ARP**

|  |  |  |
| --- | --- | --- |
| Nr | Display Filter | Description |
| 1. | ip.dst == 192.0.78.134 && ICMP | This filter will show ICMP packets (ping requests or replies) sent to the specified IP address. |
| 2. | DNS | This filter will show all DNS packets in the capture. |
| 3. | ARP | This filter will show both ARP request and response packets. |
| 4. | ip.src != 192.0.78.134 | This filter will exclude packets originating from the specified IP address. |
| 5. | ICMP || ARP | This filter shows either ICMP packets or ARP packets |

**5. Network Problem Analyse**

****

|  |  |  |  |
| --- | --- | --- | --- |
| Nr | Expert Information | Severity | Your Short Description (Problem Analysis) |
| 1. | Connection reset (RST) | Warning | It indicates that the connection has been forcibly closed by one of the parties, typically because of a problem or a request to terminate the connection. |
| 2. | TCP keep-alive segment | Note | It is a mechanism used to maintain an open connection by periodically sending small packets to check if the communication channel is still active. These keep-alive segments help prevent idle connections from being prematurely closed due to inactivity, ensuring a stable and persistent network connection. |
| 3. | This frame undergoes connection closing | Note | Frame 430 in the TCP capture represents the final stage of a connection, as it signifies the process of connection closure. This frame captures the moment when the TCP connection is undergoing closure, indicating the orderly termination of communication between two devices. |
| 4. | Connection finish (FIN) | Chat | It is an acknowledgment that one party has completed its data transmission and is ready to close the connection, allowing both parties to conclude their communication session. |
| 5. | Failed to create decryption context: Secrets are not available | Warning | indicates a problem in the QUIC (Quick UDP Internet Connections) protocol, suggesting that the necessary cryptographic secrets for secure communication are missing or unavailable. This error typically results in the failure to establish a secure QUIC connection, which may be caused by misconfigurations or key exchange issues. |

**6. Exporting File from Traffic Stream**

1. Your variant Nr: 8

2. Starting packet number: 274

3. Source IP: 65.54.95.142

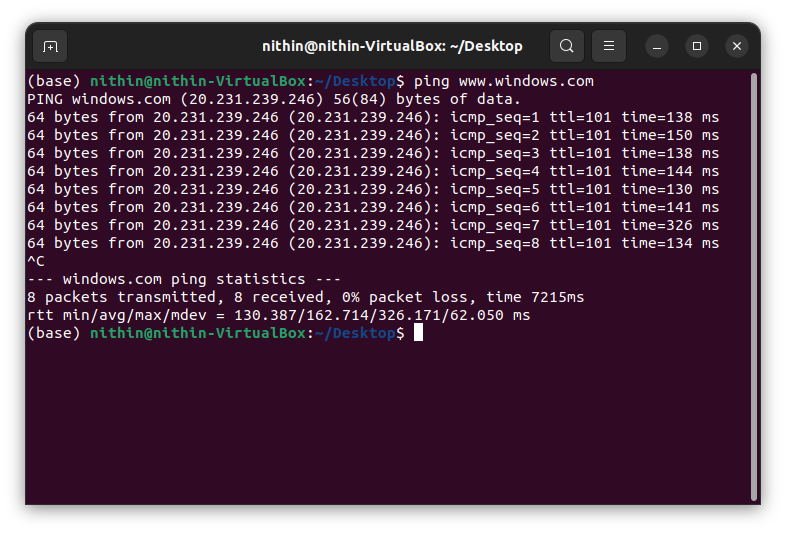
4. Destination IP:192.168.3.131

5. Jpeg file size: 32 KB

6. Picture****

* **Check the network operation tools manual and options, and try to do some terminal exercises**.
* **ping** tool: The 'ping' command sends ICMP echo requests and receives replies. When you ping a host, you'll see responses that indicate the round-trip time and packet loss.

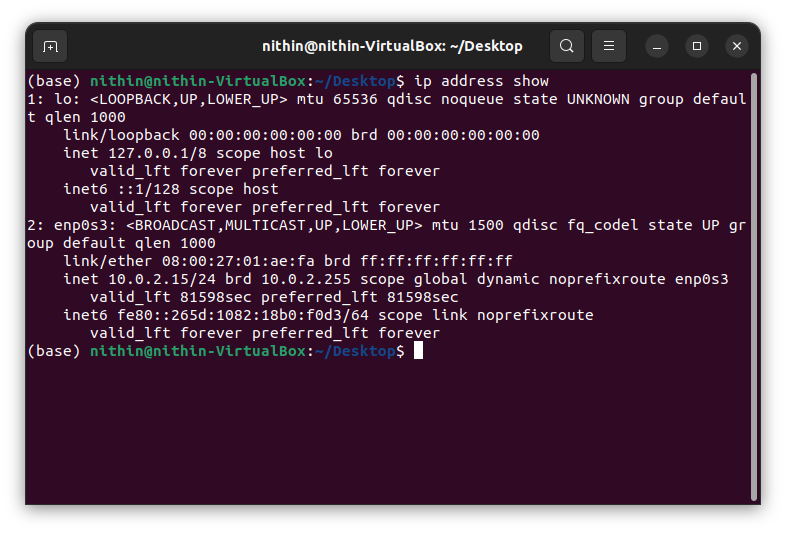
Example:



It shows that 8 packets were transmitted and 8 were received with 0% loss. The time elapsed is 7215ms

* **ip** tool: The 'ip' command provides detailed information about network interfaces. For example, running **ip address show** will display information about your network interfaces, including their IP addresses, MAC addresses, and more.

Example:

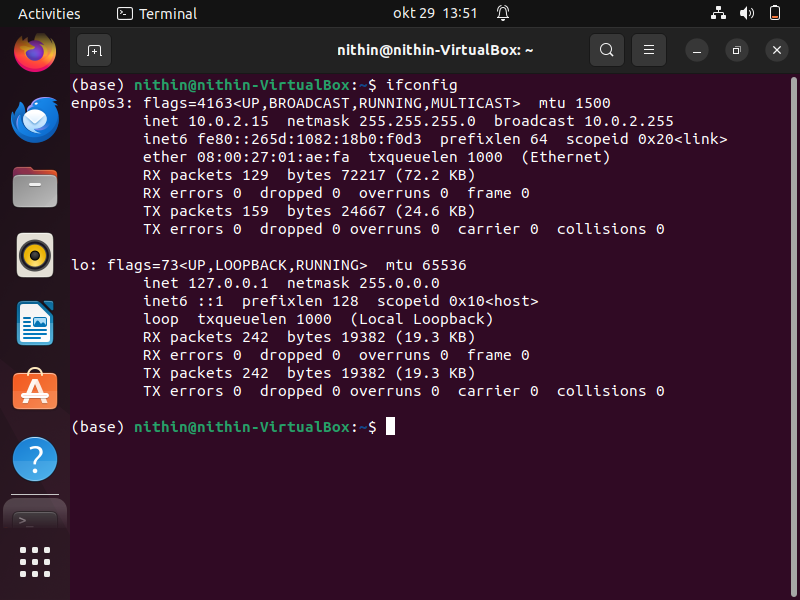


**lo**: This is the name of the network interface. It stands for "loopback," which is a special interface used for internal communication within your computer.

**enp0s3**: This is the name of another network interface. Unlike the loopback interface, this is a physical network interface, often named based on its device or slot in the system. It's not a loopback but a real network interface.

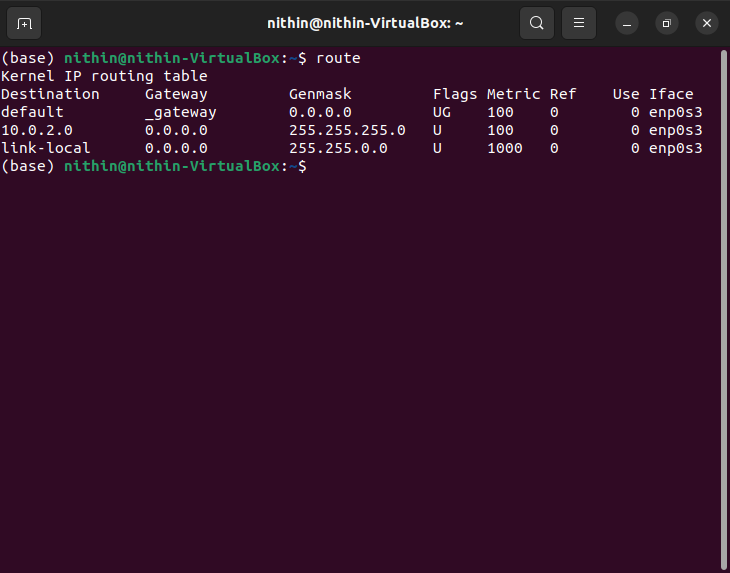
* **Ifconfig:** The 'ifconfig' command displays network interface information. When you run ifconfig, you'll see details about each network interface, such as its IP address, MAC address, and more.

Example



* **route:** The route command in Linux is used to view and manipulate the IP routing table, which defines how network packets are routed to their destination. It can display the current routing table, add or delete routes, and more

Example

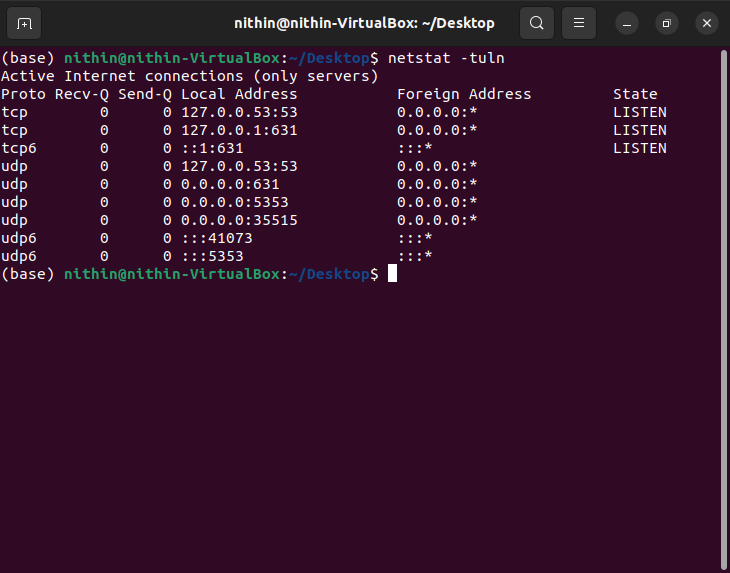


This is a representation of the kernel's IP routing table, which defines how network packets should be routed based on their destination

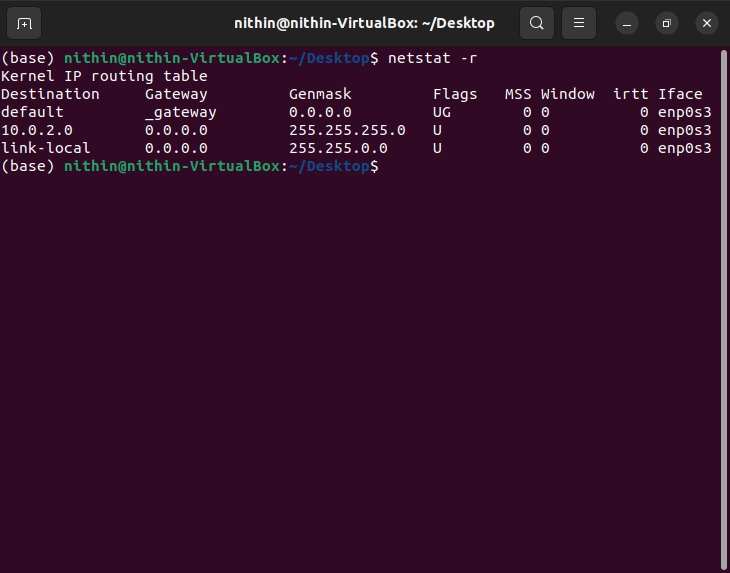
* **netstat:** The netstat command is a network utility tool available in many Unix-like operating systems, including Linux. It provides information about various network-related statistics and activities. You can use netstat to display information about network connections, routing tables, network interfaces, and more.

Example

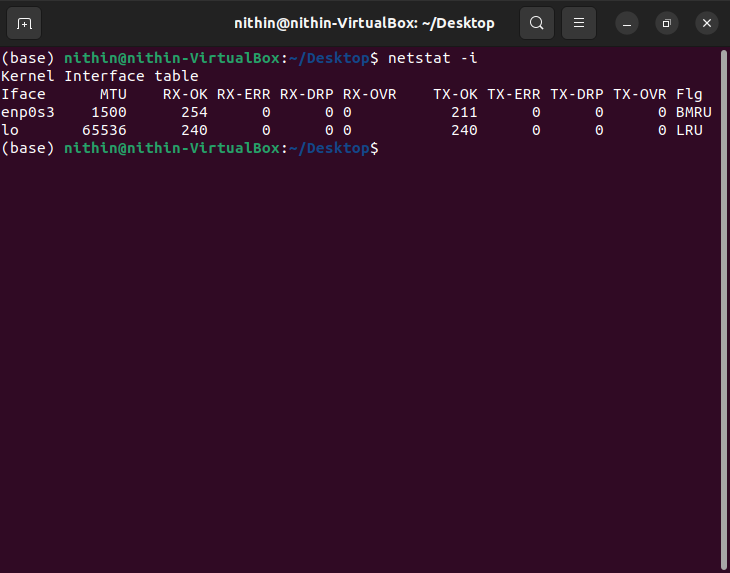
You can use **netstat** to display a list of all current network connections, including TCP, UDP, and UNIX domain socket connections. To do this, you typically use the **-t** and **-u** options for TCP and UDP connections, respectively.



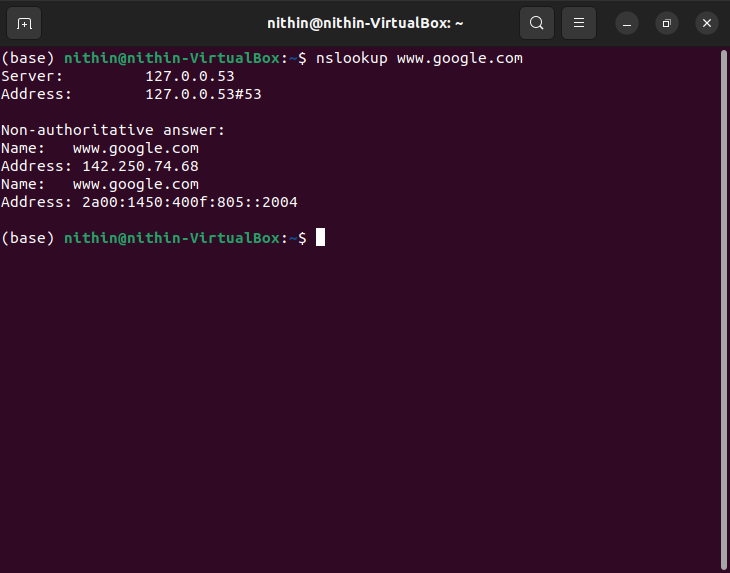
You can use **netstat** to show the kernel's IP routing table. The **-r** option is used for this purpose



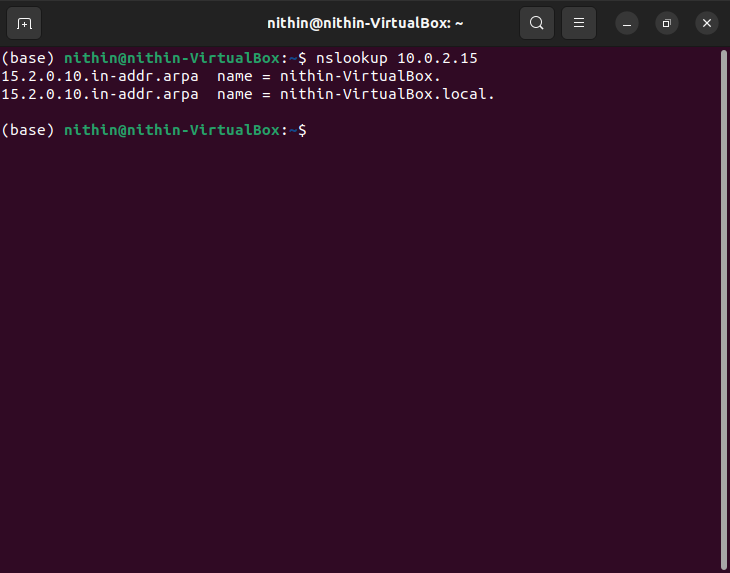
To view a list of all network interfaces on your system, you can use the **-i** option. This can be helpful for identifying network interface names and their current status.



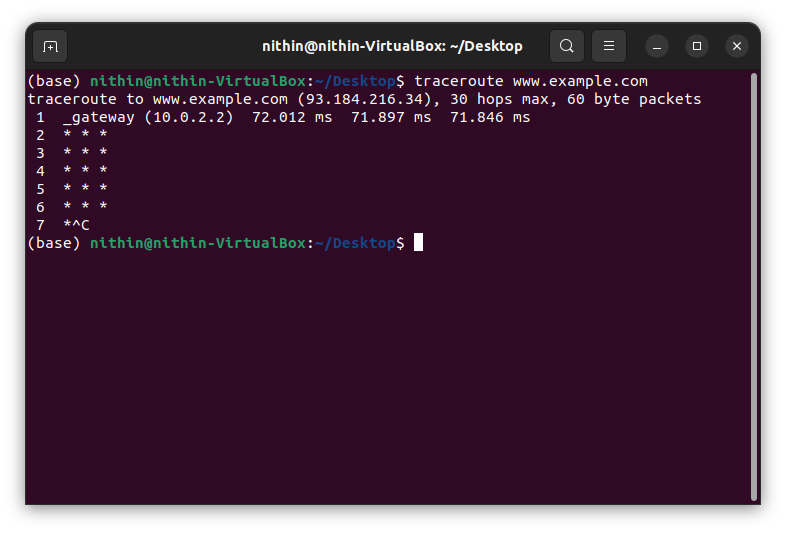
* **nslookup:** 'nslookup' is used to query DNS. When you run nslookup www.example.com, you'll get the IP address of www.example.com, along with the DNS server that provided the information.



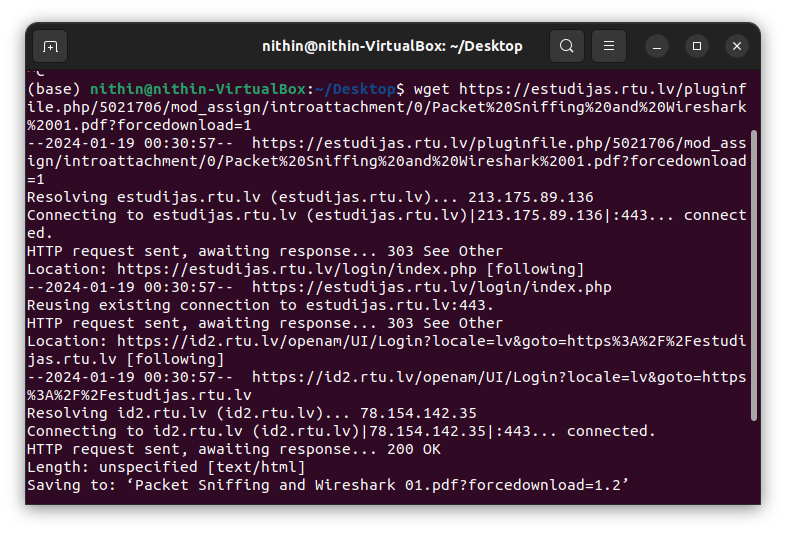
You can also perform a reverse DNS lookup to find the domain associated with an IP address.



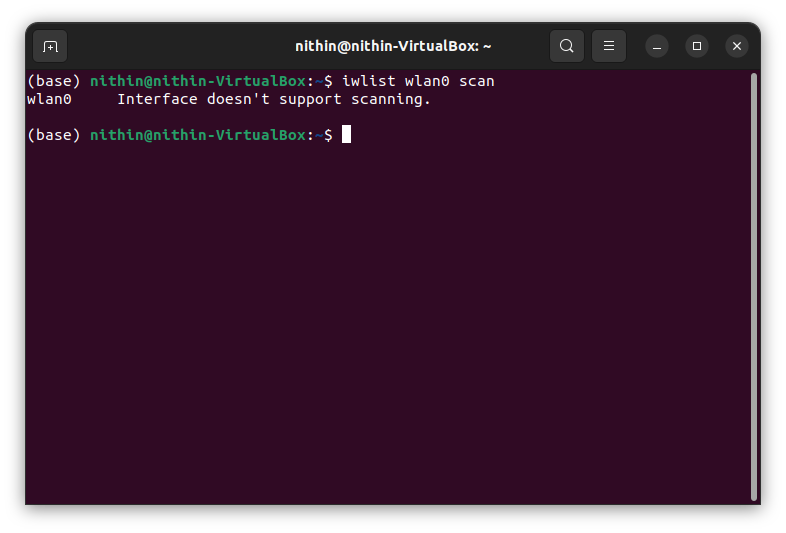
* **traceroute:** Traceroute is a command-line tool used in Linux to trace the route that packets take from your computer to a destination IP address or domain. It helps you identify the routers or hops between your computer and the destination.



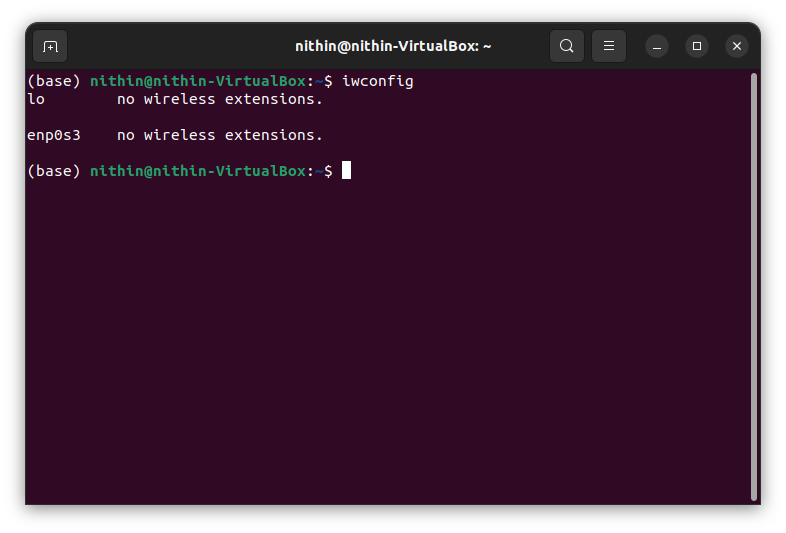
* **wget:** wget is a command-line utility for non-interactive downloading of files from the web. It supports HTTP, HTTPS, and FTP protocols and allows for the retrieval of entire websites or specific files.



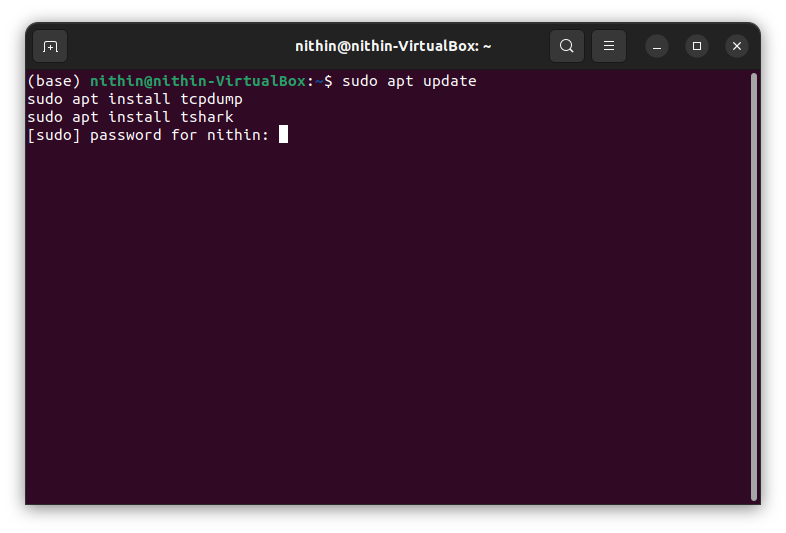
* **iwlist**: iwlist is a command-line tool for scanning wireless networks. It provides detailed information about available wireless networks, including their encryption methods, signal strengths, and other parameters.



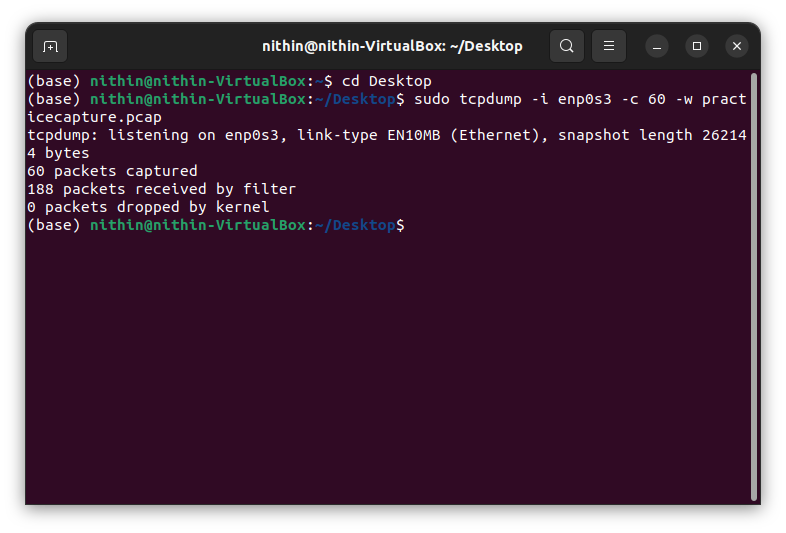
* **Iwconfig**: iwconfig is a command-line tool for configuring wireless network interfaces. It allows you to view and modify parameters such as the wireless network name (SSID), encryption settings, and power management options.



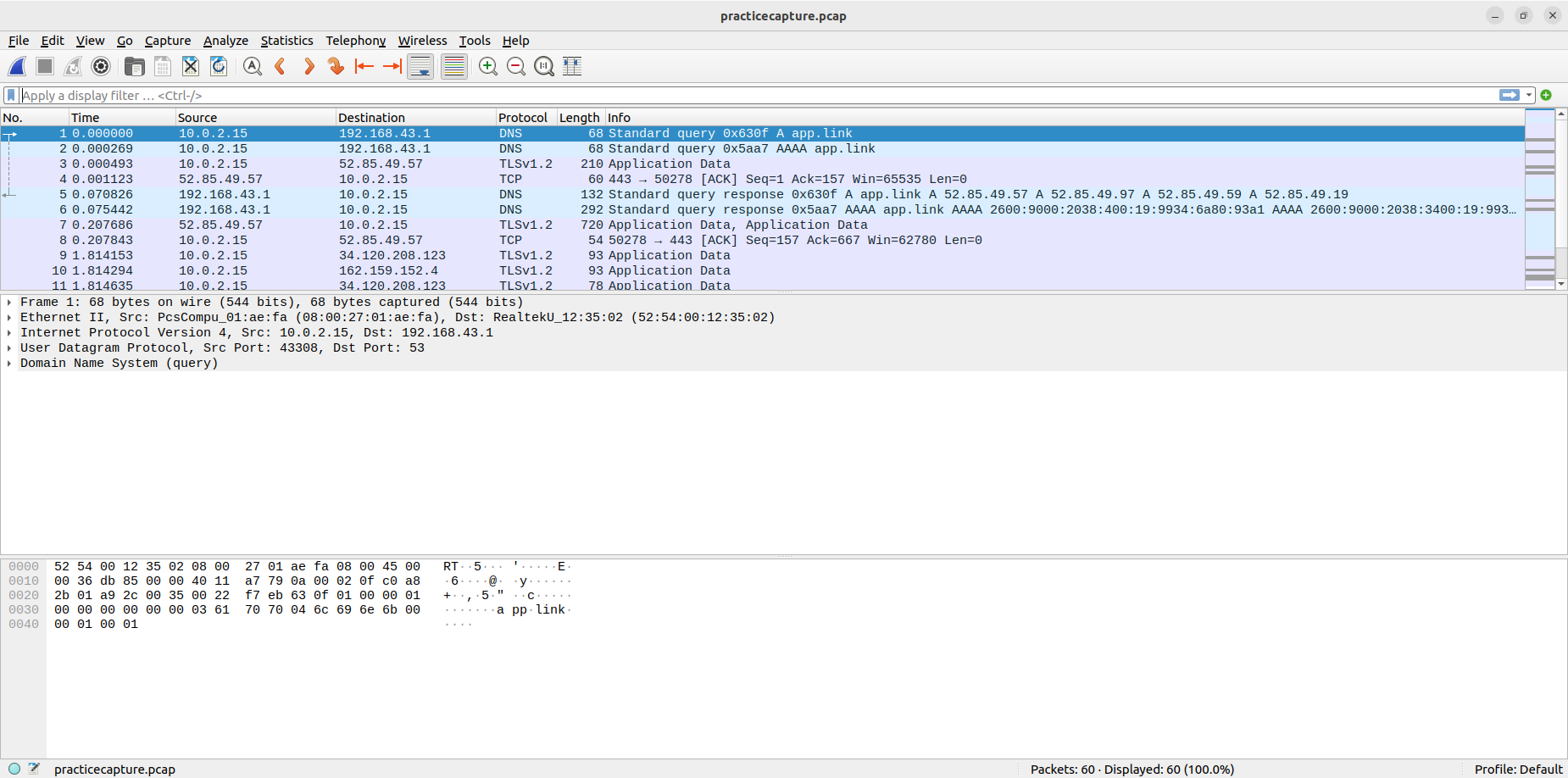
* **Tcpdump and Tshark terminals practical exercises**

****

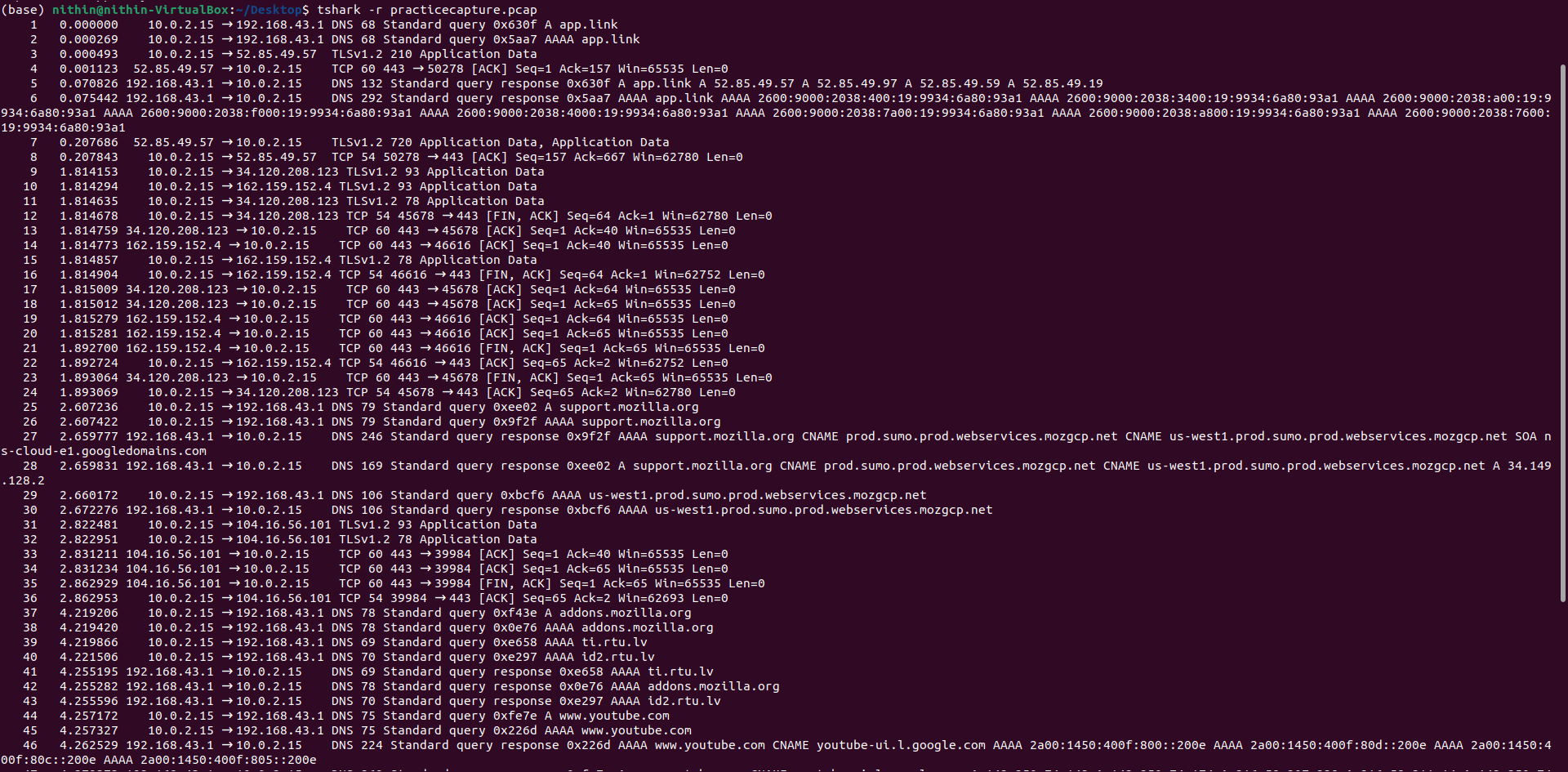
* These commands update the package list and install Tcpdump and Tshark.



* Tcpdump :Capture packet
* -i enp0s3: Specifies the network interface.
* -c 60: Captures only 60 packets.
* -w practicecapture.pcap: Writes the capture to a file named practicecapture.pcap.



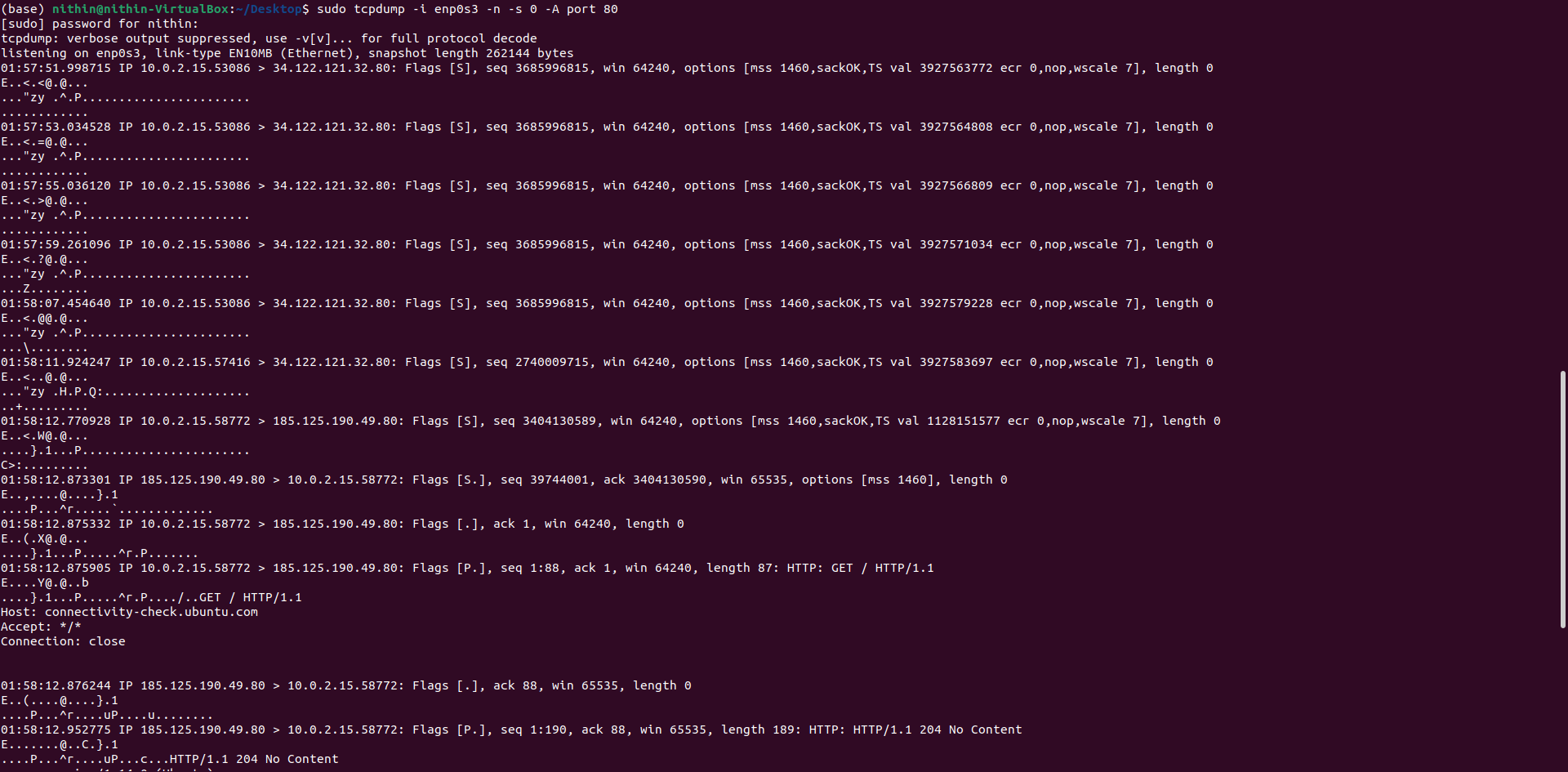
* Captured file is saved in the desktop and can be viewed in the wireshark software.
* Tshark: Display Packet Details: tshark -r practicecapture.pcap



r practicecapture.pcap: Reads packets from the specified capture file.

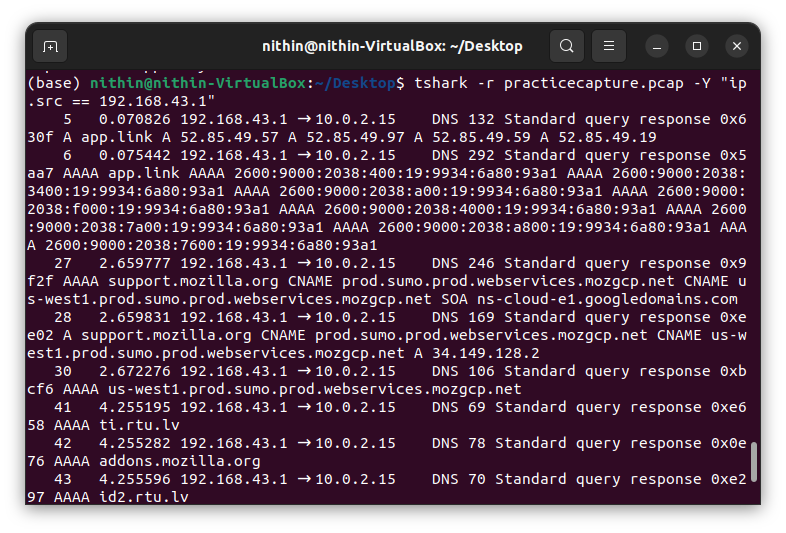
* Tcpdump: sudo tcpdump -i enp0s3 -n -s 0 -A port 80

Captures and displays HTTP traffic on the **enp0s3** interface without resolving hostnames, showing the entire packet in ASCII.



* Tshark : tshark -r practicecapture.pcap -Y "ip.src == 192.168.43.1"

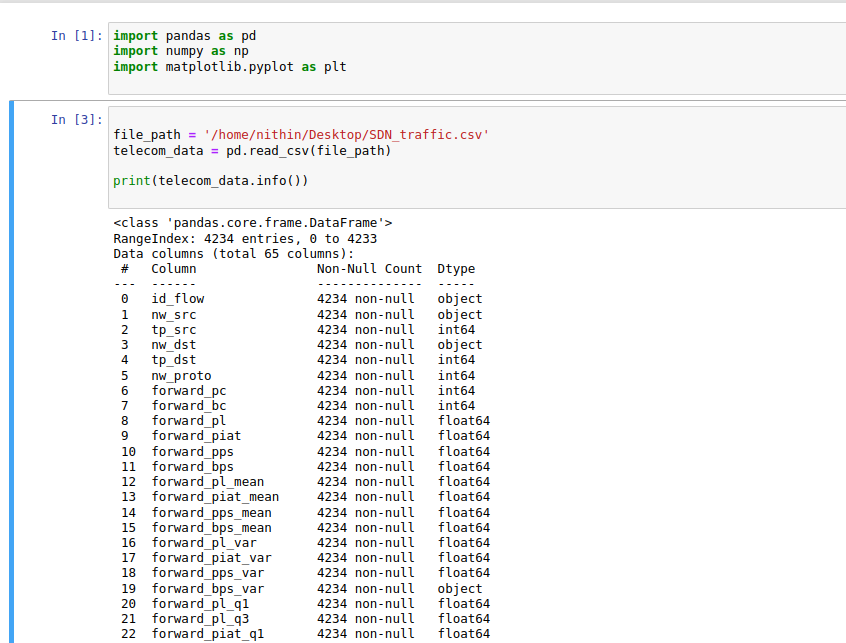
Reads packets from practicecapture.pcap and displays only those with a source IP of **192.168.43.1**.



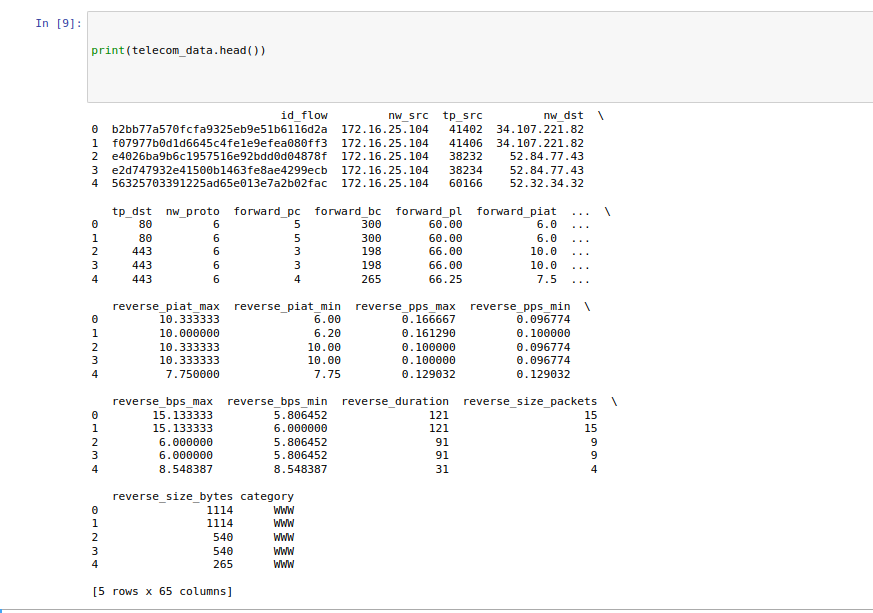
**Task 2**

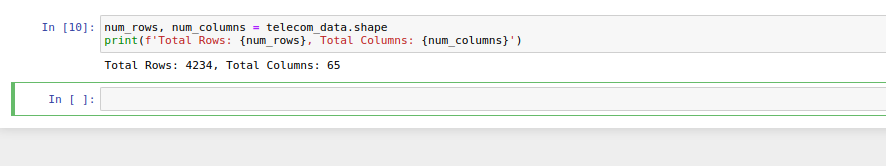
 Python Numpy, Pandas, and Matplotlib library-related exercises

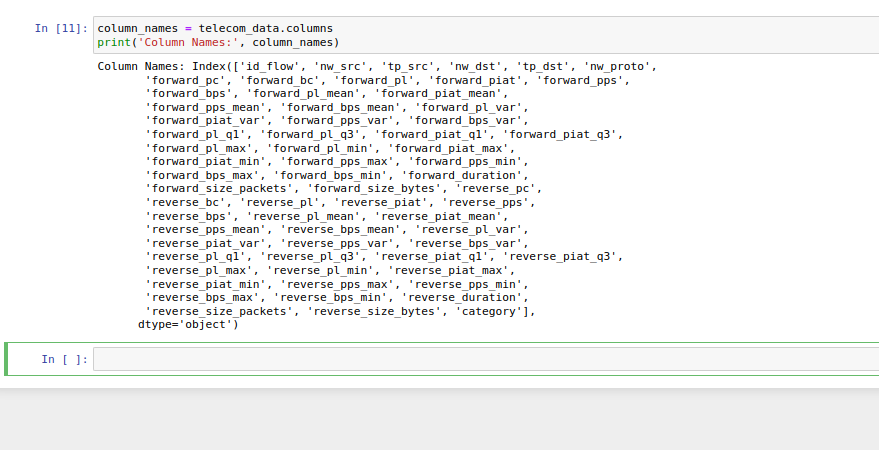
* Load the data and show the column names, non-null counts, and data types for each column.



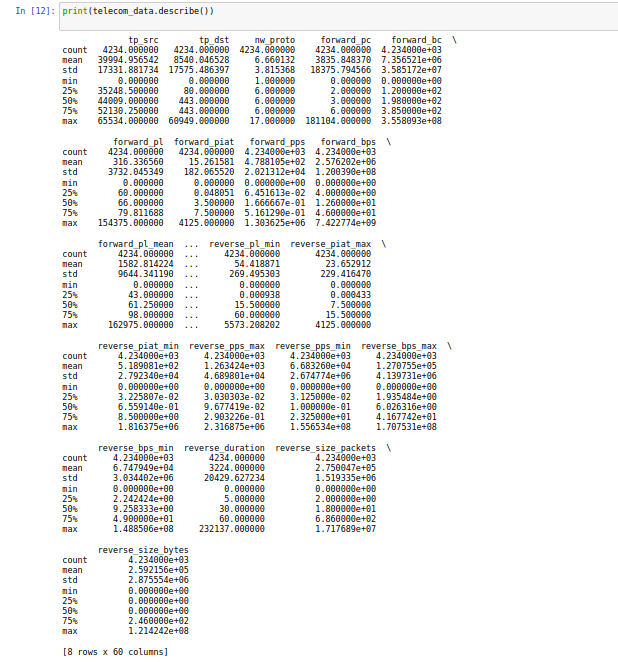
* Now we need to display the total number of columns and rows. Also displays few rows and names of columns.



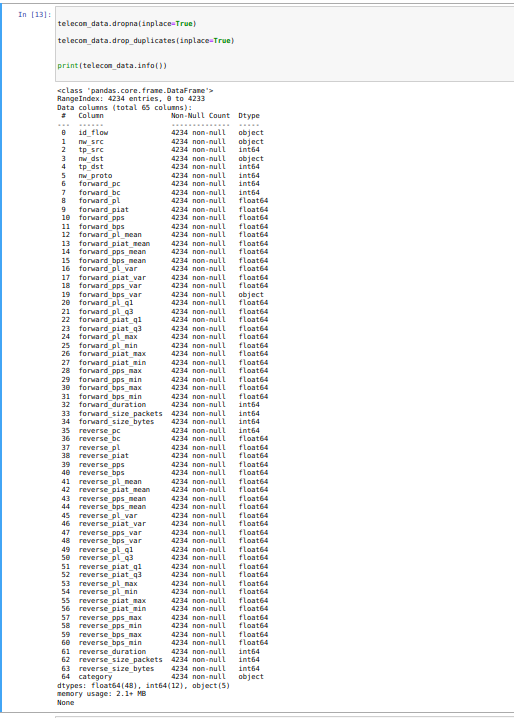


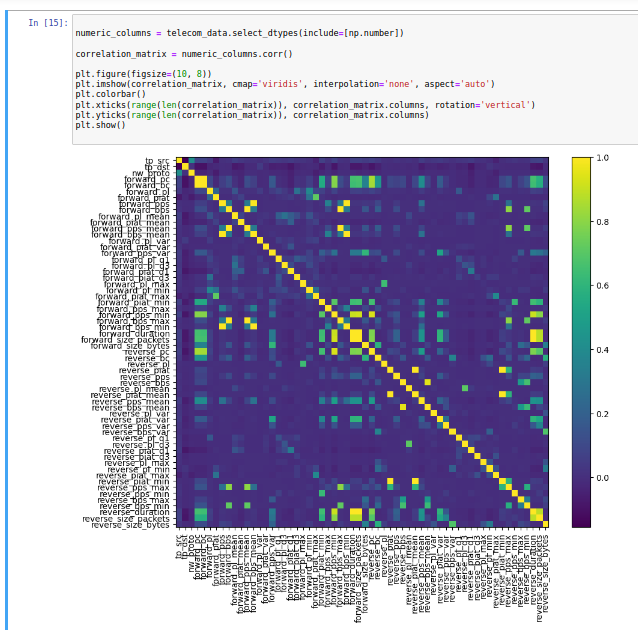
****

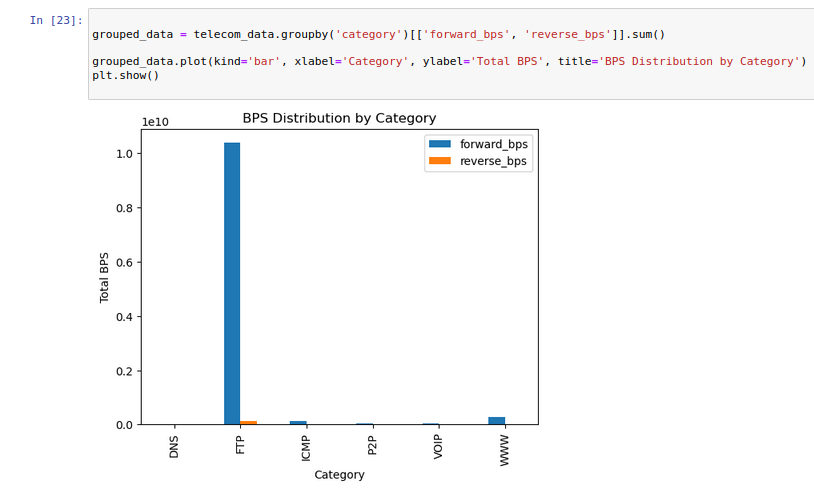
* Now we need to display descriptive statistics of the numerical columns



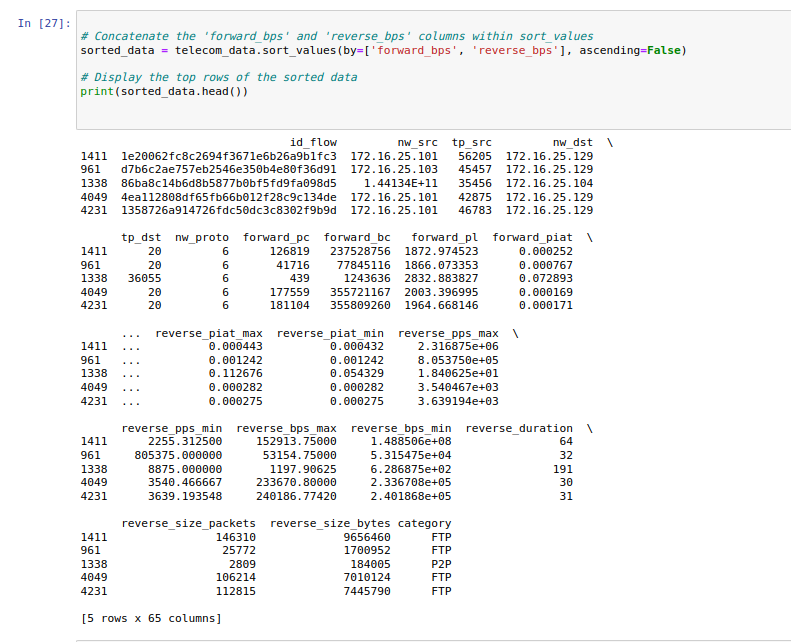
* Next is cleaning the values in the data by checking for any mistakes or duplicate file and again displaying the data.



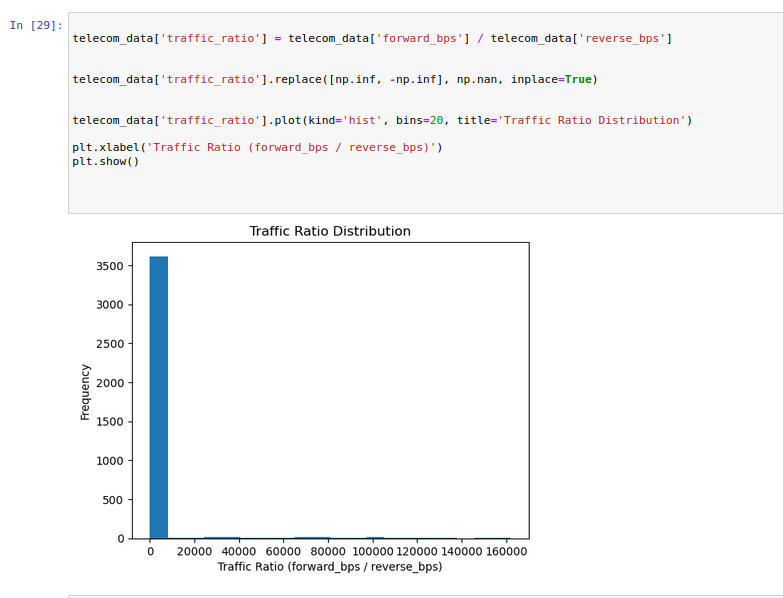
* Data analysis and visualization
* Next we can group the data by the 'category' column and sum the 'forward\_bps' and 'reverse\_bps' for each category. It will then create a bar plot to visualize the total BPS distribution by category.



* Now sorting the data by total traffic in descending order



* Now plotting the traffic data distribution.



* Finally displaying the statistical summary of the data.

