LAB Manual
PART A
(PART A : TO BE REFFERED BY STUDENTS)

Experiment No. 5

A.1 Aim:
To install network sniffer (Wireshark) , analyze the working of packet sniffing and study other features of Wireshark

A.2 Prerequisite:
Role of packet sniffers in network security

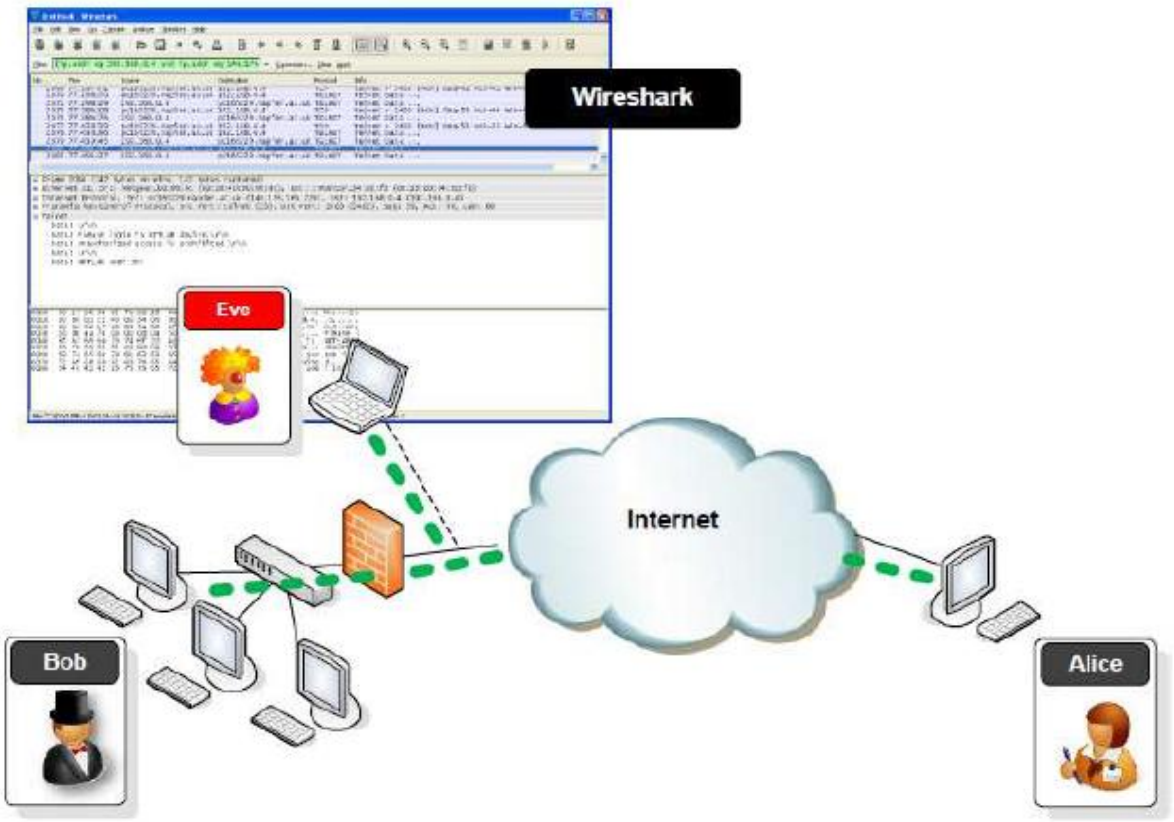
A.3 Outcome:
After successful completion of this experiment students will be able to
1. Appreciate wireshark as a tool to analyze the packets travelling in a network.
2. Know how this tool can be used by malicious intruders to capture and analyze network traffic.

A.4 Theory:

A packet sniffer, sometimes referred to as a network monitor or network analyzer, can be used by a network or system administrator to monitor and troubleshoot network traffic. Using the information captured by the packet sniffer an administrator can identify erroneous packets and use the data to pinpoint bottlenecks and help maintain efficient network data transmission. In its simple form a packet sniffer simply captures all of the packets of data that pass through a given network interface. By placing a packet sniffer on a network in promiscuous mode, a malicious intruder can capture and analyze all of the network traffic.

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible.

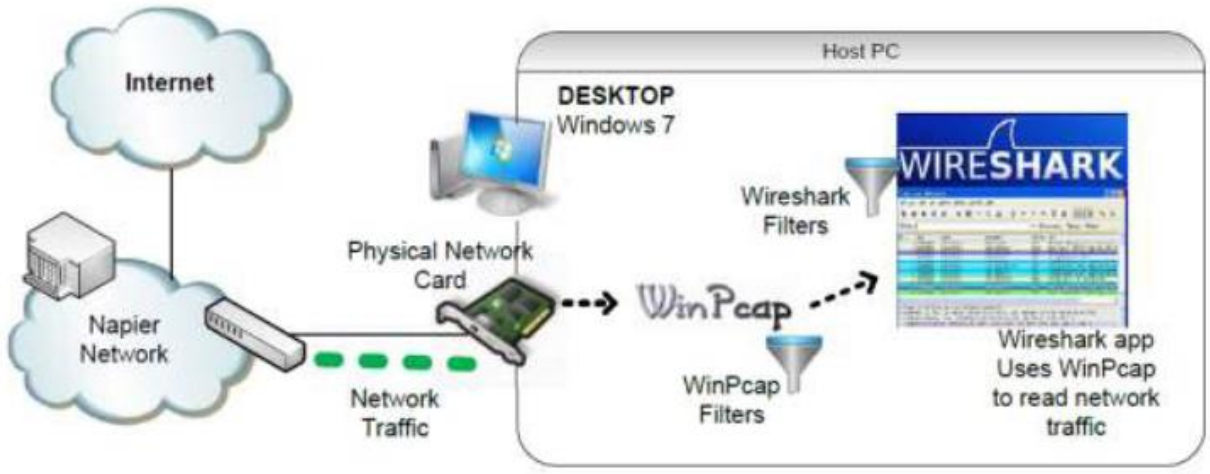
Packet Capture (Packet Sniffing)
A **packet sniffer** is an application which can capture and analyse network traffic which is passing through a system's Network Interface Card (NIC). The sniffer sets the card to **promiscuous mode** which means all traffic is read, whether it is addressed to that machine or not. The figure below shows an attacker sniffing packets from the network, and the **Wireshark** packet sniffer/analyser (formerly known as ethereal).



Packet Analysis
Wireshark is an open source cross-platform packet capture and analysis tool, with versions for Windows and Linux. The GUI window gives a detailed breakdown of the network protocol stack for each packet, colorising packet details based on protocol, as well as having functionality to filter and search the traffic, and pick out TCP streams. Wireshark can also save packet data to files for offline analysis and export/import packet captures to/from other tools. Statistics can also be generated for packet capture files.
Wireshark can be used for **network troubleshooting**, to **investigate security issues**, and to **analyse and understand network protocols**. The packet sniffer can exploit information passed in plaintext, i.e. not encrypted. Examples of **protocols** which pass information in plaintext are **Telnet, FTP, SNMP, POP, and HTTP**.

Wireshark is a GUI based network capture tool. There is a command line based version of the packet capture utility, called **TShark**. TShark provides many of the same features as it's big brother, but is console-based. It can be a good alternative if only command line access is available, and also uses less resources as it has no GUI to generate.

Using Wireshark to Capture Traffic



Select a Network Interface to Capture Packets through.

Start the Wireshark application. When Wireshark is first run, a default, or blank window is shown. To list the available network interfaces, select the **Capture->Interfaces** menu option.

Wireshark should display a popup window such as the one shown in Figure 2. To capture network traffic click the **Start** button for the network interface you want to capture traffic on. Windows can have a long list of virtual interfaces, before the Ethernet Network Interface Card (NIC).

Figure 2 - Wireshark Interfaces Window

Generate some network traffic with a Web Browser, such as Internet Explorer or Chrome. Your Wireshark window should show the packets, and now look something like.

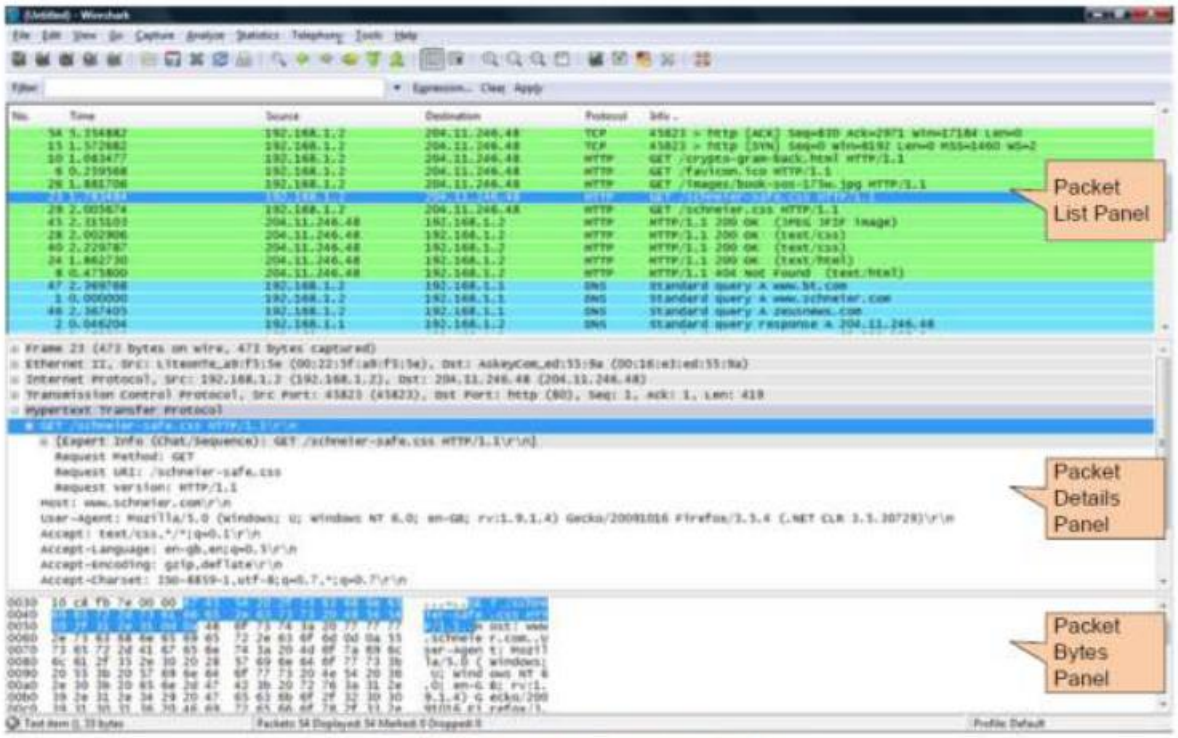


Figure 3 - Wireshark capuring traffic

To stop the capture, select the **Capture->Stop** menu option, Ctrl+E, or the Stop toolbar button. What you have created is a Packet Capture or ***„pcap'***, which you can now view and analyse using
the Wireshark interface, or save to disk to analyse later.
The capture is split into 3 parts:
1. **Packet List Panel –** this is a list of packets in the current capture. It colours the packets based on the protocol type. When a packet is selected, the details are shown in the two panels below.
2. **Packet Details Panel –** this shows the details of the selected packet. It shows the different protocols making up the layers of data for this packet. Layers include Frame, Ethernet, IP, TCP/UDP/ICMP, and application protocols such as HTTP.
3. **Packet Bytes Panel –** shows the packet bytes in Hex and ASCII encodings.
To select more detailed options when starting a capture, select the **Capture->Options** menu option, or **Ctrl+K**, or the Capture Options button on the toolbar (the wrench). This should show a window such as shown in Figure 4.
Some of the more interesting options are:
 ***Capture Options > Interface* -** Again the important thing is to select the correct Network Interface to capture traffic through.
 ***Capture Options > Capture File* –** useful to save a file of the packet capture in real time, in case of a system crash.
 ***Display Options > Update list of packets in real time* –** A display option, which should be checked if you want to view the capture as it happens (typically switched off to capture straight to a file, for later analysis).
 ***Name Resolution > MAC name resolution* –** resolves the first 3 bytes of the MAC Address, the Organisation Unique Identifier (OUI), which represents the Manufacturer of the Card.

 Name Resolution > Network name resolution – does a DNS lookup for the IP Addresses captured, to display the network name. Set to off by default, so covert scans do not generate this DNS traffic, and tip off who's packets you are sniffing.
Make sure the **MAC name resolution** is selected. Start the capture, and generate some Web traffic again, then stop the capture.

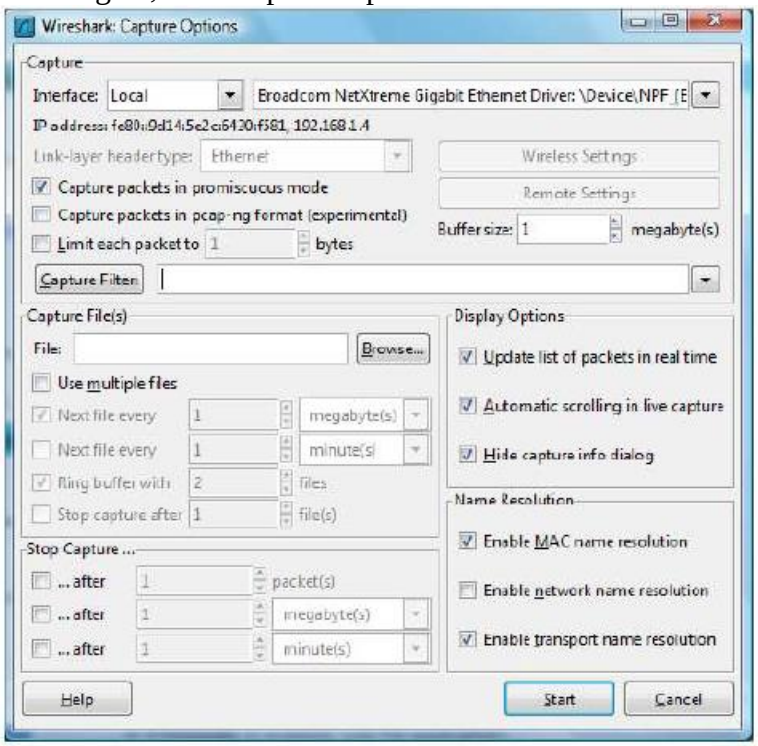


Figure 4 - Wireshark Capture Options

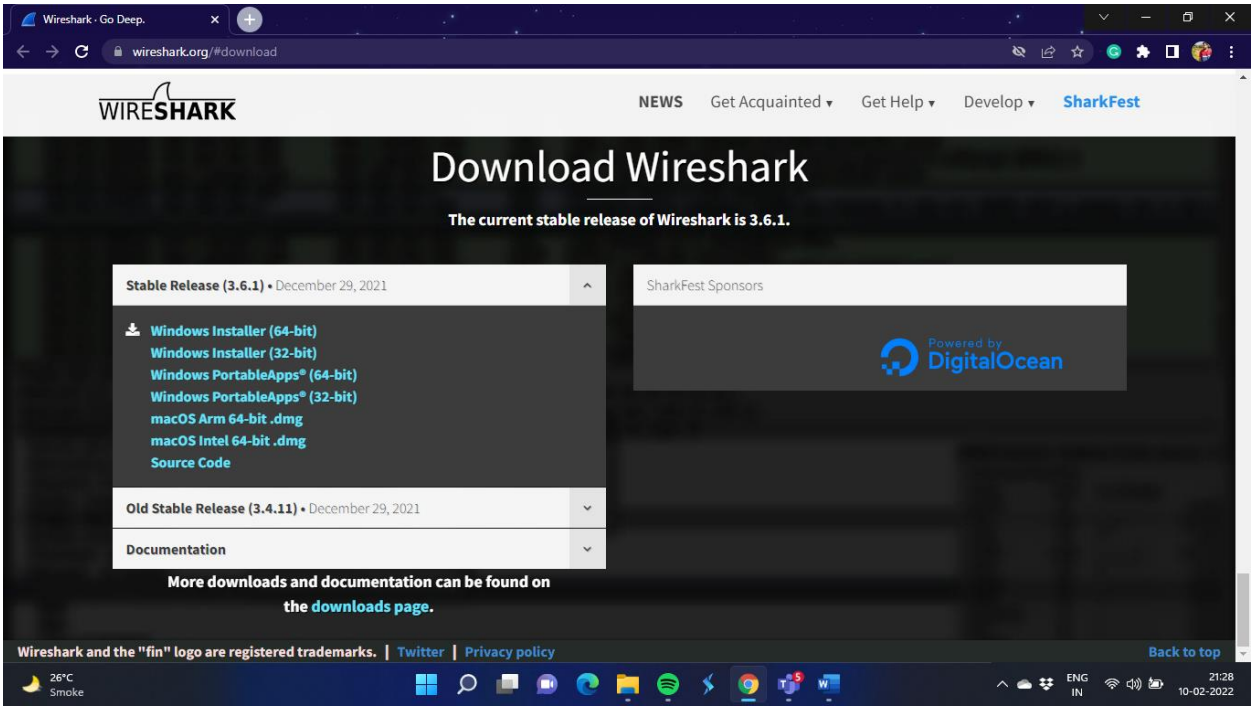
PART B

(PART B: TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)

Roll. No. A16	Name: Varun Khadayate
Class B.Tech CsBs	Batch: 1
Date of Experiment: 04-02-2022	Date of Submission:10-02-2022
Grade:	

B.1 Software installation issues faced:



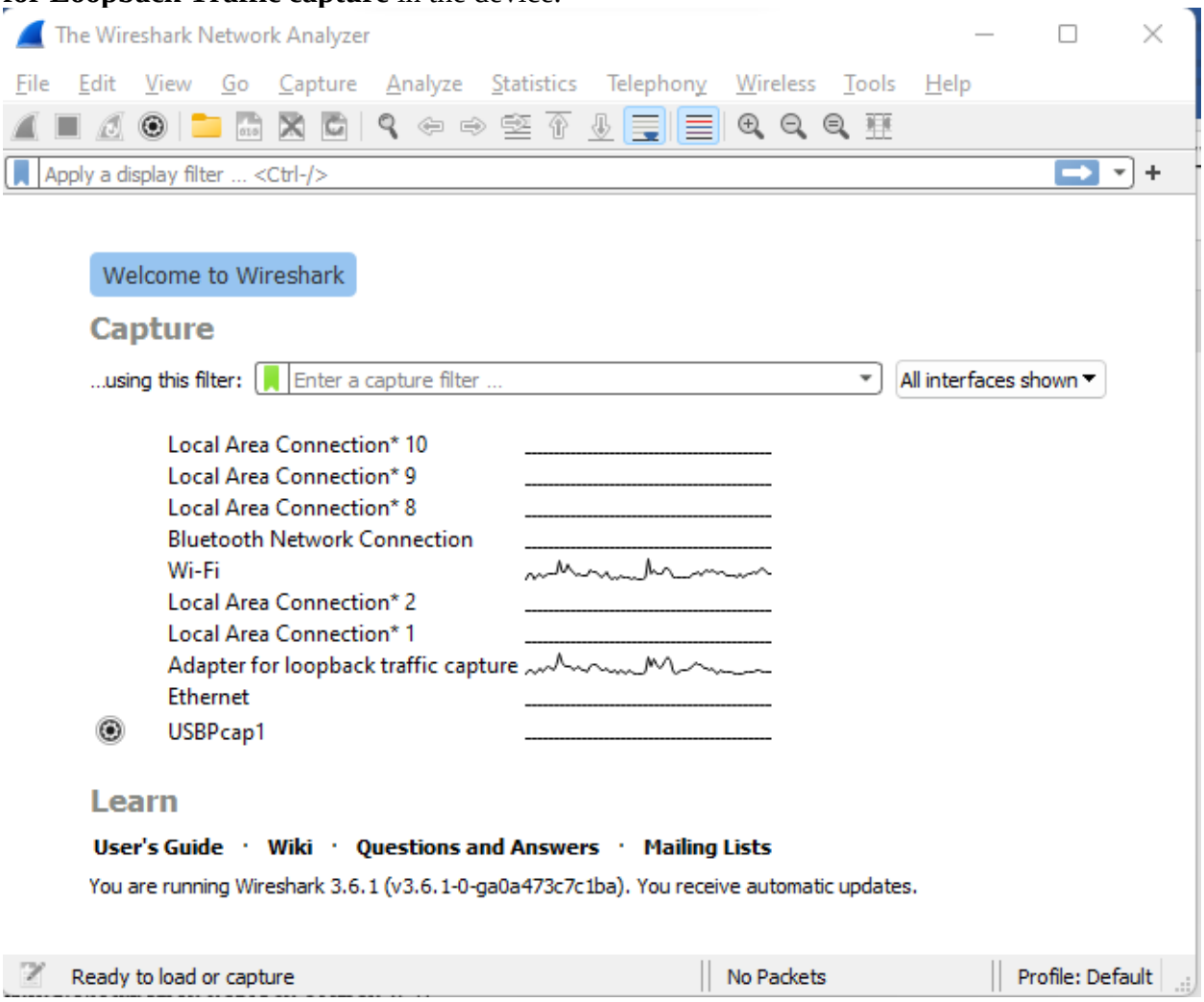
I. Download Wireshark Application from the link: https://www.wireshark.org/#download
II. Once the download is complete, open the installer and install the application on the Device.
III. Note: Once installed you'll have to restart your device to make Wireshark work efficiently.
IV. Some devices have trouble even after restarting so for that, run Wireshark as administrator.

B.2 Input and Output:

(Paste your program input and output in following format, If there is error then paste the specific error in the output part. In case of error with due permission of the faculty extension can be given to submit the error free code with output in due course of time. Students will be graded accordingly.)

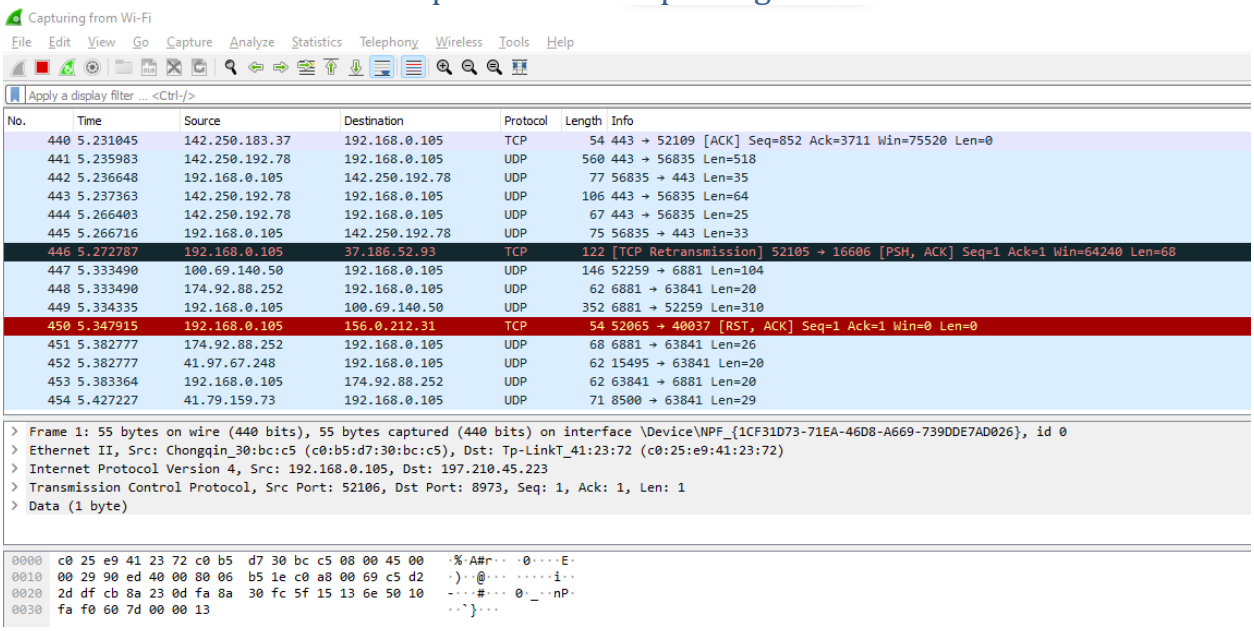
Input:

Below shows the wireshark screen when nothing is running except the **Wifi** and **Adapter for Loopback Traffic capture** in the device.

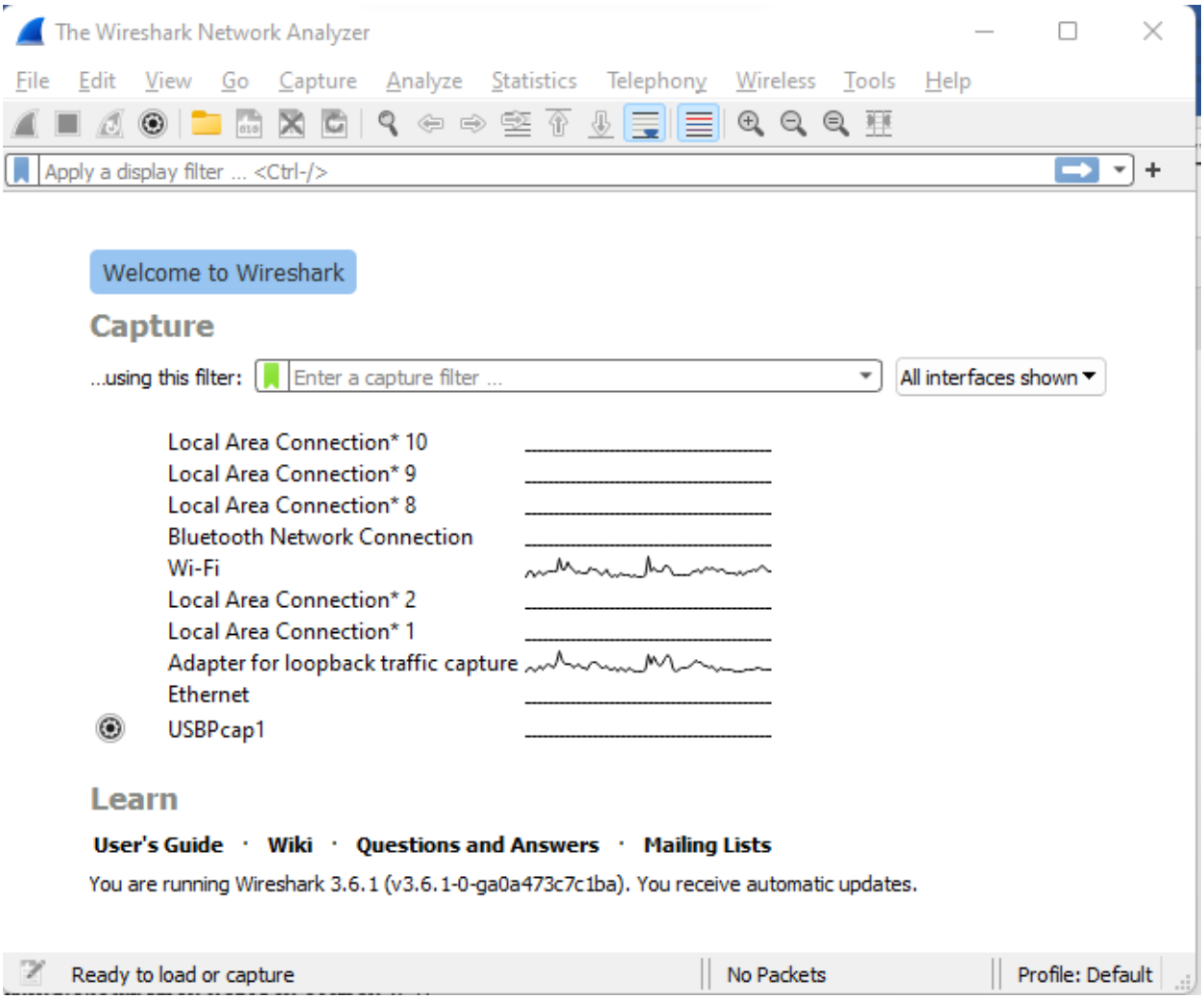


Output:

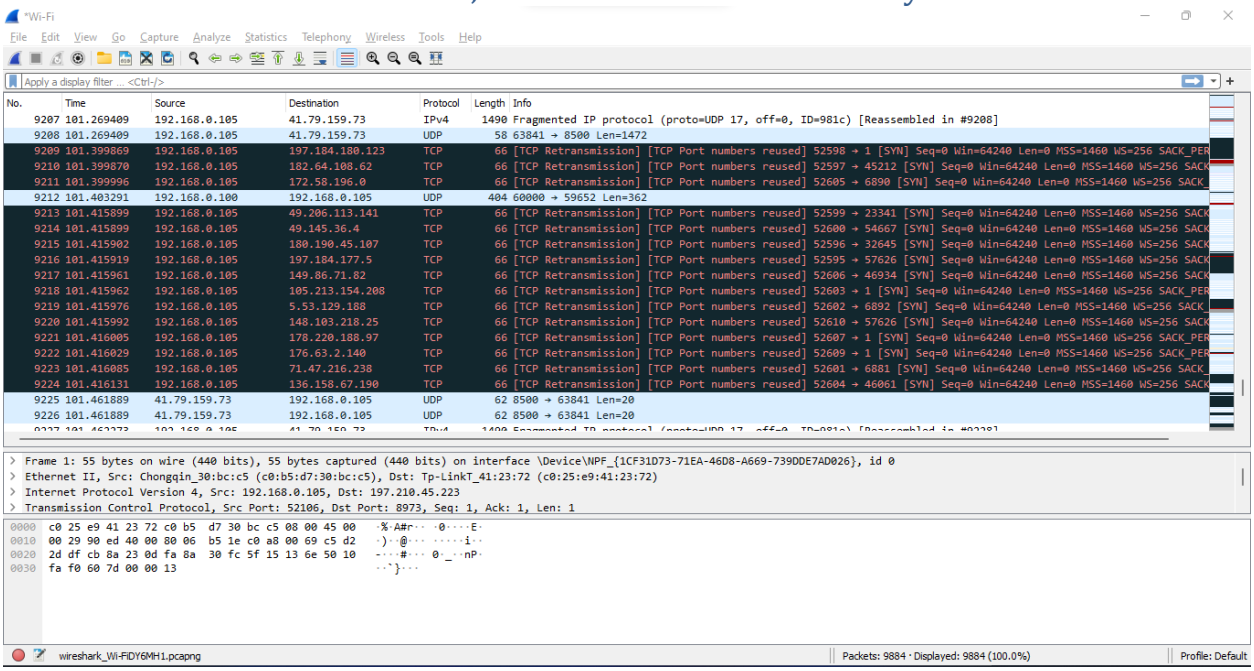
Below is the IP Addresses, Hex codes and their corresponding ASCII Codes and some frame information about the packet that are parsing from WiFi.



If you start your Google Chrome Browser and open your GMAIL Account, the Wireshark tool shows some data.



This also shows the IP Adresses, the hexa-decimal code for my username as well.



B.3 Observations and learning:

(Students are expected to comment on the output obtained with clear observations and learning for each task/ sub part assigned)

From the above experiment, we can observe that device's WiFi passes some default packets to the device even when there is nothing else using the WiFi. When we open GMAIL via Google Chrome Browser, we see some waves of network in Traffic Capture

B.4 Conclusion:

(Students must write the conclusion as per the attainment of individual outcome listed above and learning/observation noted in section B.3)

We were able to observe the working of Wireshark Tool when there is only wifi working and one when we use the browser, successfully.

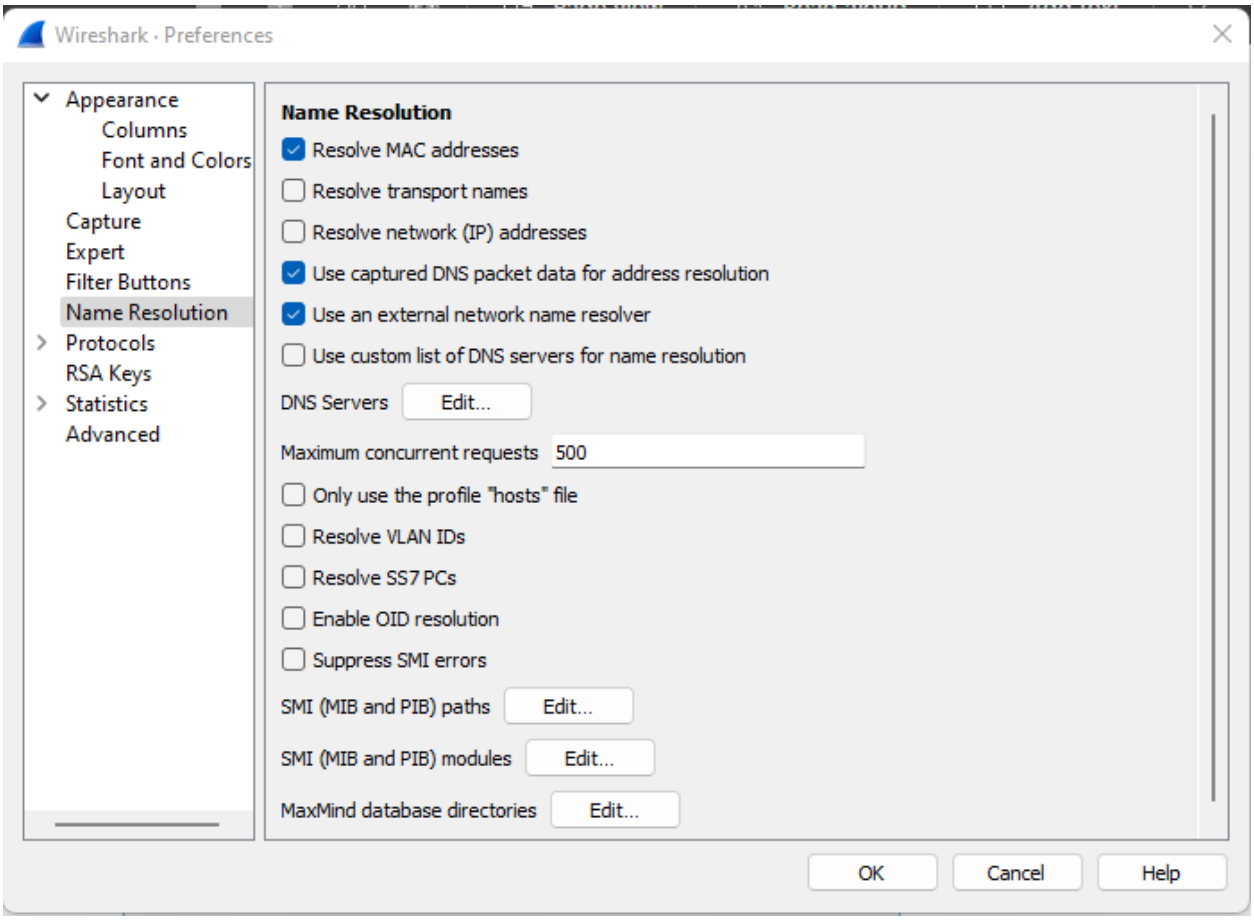
Questions of Curiosity

(To be answered by student based on the practical performed and learning/observations)

Q1: Give the uses of wireshark tool

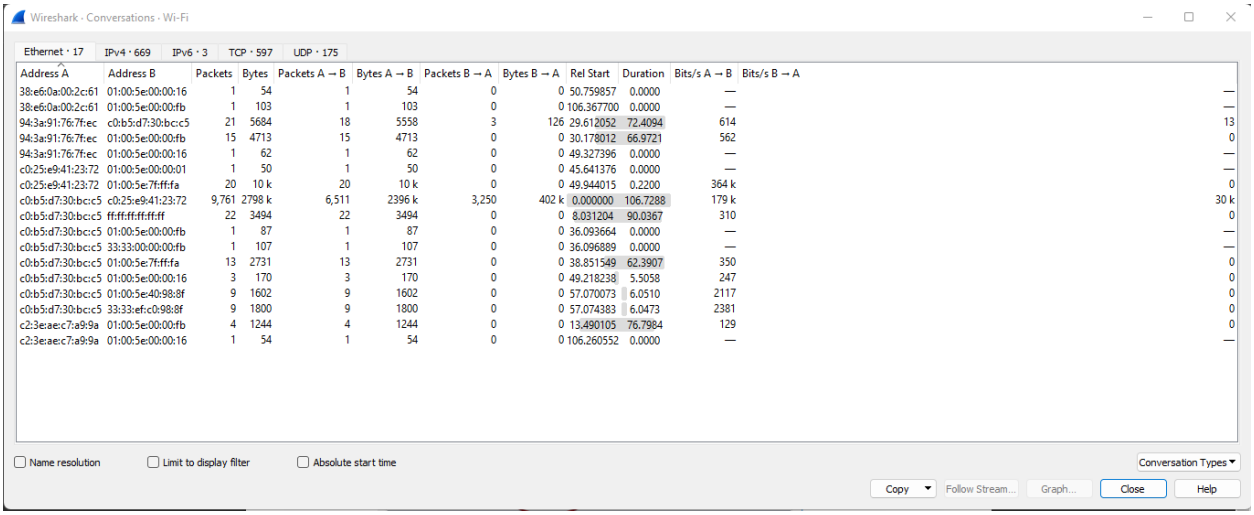
Baselining your traffic:

Baselining is the process of capturing and identifying the "normal" traffic on a network. This traffic may include the auto-update applications on a network, a myriad of broadcast and multicast traffic streams, auto-detect applications scrounging around the network unnecessarily, and more.



Perform Passive Discovery:

Passive discovery is the process of building a map of the network based on what you hear while listening to the traffic. From the Statistics menu, Wireshark can provide a list of visible hosts, conversations (pairs of hosts communicating with each other), resolved addresses, port numbers, and more.



Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
38:e6:0a:00:2c:61	01:00:5e:00:00:16	1	54	1	54	0	0	50.759857	0.0000	—	—
38:e6:0a:00:2c:61	01:00:5e:00:00:fb	1	103	1	103	0	0	106.367700	0.0000	—	—
94:3a:91:76:7f:ec	c0:b5:d7:30:bc:c5	21	5684	18	5558	3	126	29.612052	72.4094	614	13
94:3a:91:76:7f:ec	01:00:5e:00:00:fb	15	4713	15	4713	0	0	30.178012	66.9721	562	0
94:3a:91:76:7f:ec	01:00:5e:00:00:16	1	62	1	62	0	0	49.327396	0.0000	—	—
c0:25:e9:41:23:72	01:00:5e:00:00:01	1	50	1	50	0	0	45.641376	0.0000	—	—
c0:25:e9:41:23:72	01:00:5e:7f:ff:fa	20	10 k	20	10 k	0	0	49.944015	0.2200	364 k	0
c0:b5:d7:30:bc:c5	c0:25:e9:41:23:72	9,761	2798 k	6,511	2396 k	3,250	402 k	0.000000	106.7288	179 k	30 k
c0:b5:d7:30:bc:c5	ff:ff:ff:ff:ff:ff	22	3494	22	3494	0	0	8.031204	90.0367	310	0
c0:b5:d7:30:bc:c5	01:00:5e:00:00:fb	1	87	1	87	0	0	36.093664	0.0000	—	—
c0:b5:d7:30:bc:c5	33:33:00:00:00:fb	1	107	1	107	0	0	36.096889	0.0000	—	—
c0:b5:d7:30:bc:c5	01:00:5e:7f:ff:fa	13	2731	13	2731	0	0	38.851549	62.3907	350	0
c0:b5:d7:30:bc:c5	01:00:5e:00:00:16	3	170	3	170	0	0	49.218238	5.5058	247	0
c0:b5:d7:30:bc:c5	01:00:5e:40:98:8f	9	1602	9	1602	0	0	57.070073	6.0510	2117	0
c0:b5:d7:30:bc:c5	33:33:ef:c0:98:8f	9	1800	9	1800	0	0	57.074383	6.0473	2381	0
c2:3e:ae:c7:a9:9a	01:00:5e:00:00:fb	4	1244	4	1244	0	0	13.490105	76.7984	129	0
c2:3e:ae:c7:a9:9a	01:00:5e:00:00:16	1	54	1	54	0	0	106.260552	0.0000	—	—

Wireshark · Conversations · Wi-Fi

Ethernet · 17 | IPv4 · 669 | IPv6 · 3 | TCP · 597 | UDP · 175

Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.134.60.10	192.168.0.105	4	376	4	376	0	0	93.450631	7.0102	429	0
13.107.6.254	192.168.0.105	1	54	1	54	0	0	77.960923	0.0000	—	—
13.107.213.48	192.168.0.105	1	54	1	54	0	0	77.754908	0.0000	—	—
37.60.91.78	192.168.0.105	5	470	5	470	0	0	67.142870	15.0170	250	0
40.100.138.18	192.168.0.105	1	54	1	54	0	0	78.802007	0.0000	—	—
41.79.159.73	192.168.0.105	3,832	1990 k	1,346	88 k	2,486	1901 k	0.158870	106.5187	6640	142 k
41.97.67.248	192.168.0.105	118	48 k	55	3742	63	44 k	0.448333	105.6034	283	3372
52.72.234.19	192.168.0.105	8	509	4	293	4	216	26.183564	1.8032	1299	958
52.114.14.228	192.168.0.105	10	2097	6	1350	4	747	17.592797	80.3298	134	74
78.8.215.253	192.168.0.105	33	4216	13	1527	20	2689	0.158870	86.1388	141	249
82.79.208.40	192.168.0.105	20	1088	10	540	10	548	10.003337	89.5061	48	48
85.210.31.50	192.168.0.105	8	648	8	648	0	0	8.949306	96.2884	53	0
89.64.84.35	192.168.0.105	1	69	1	69	0	0	6.286162	0.0000	—	—
100.69.140.50	192.168.0.105	163	29 k	84	10 k	79	18 k	2.399848	104.1776	827	1438
117.18.232.200	192.168.0.105	11	675	3	243	8	432	35.155263	58.1367	33	59
131.253.33.254	192.168.0.105	1	54	1	54	0	0	79.880952	0.0000	—	—
142.251.12.189	192.168.0.105	102	7391	51	3563	51	3828	1.228599	95.9215	297	319
162.125.19.131	192.168.0.105	7	2790	4	334	3	2456	35.466374	0.2820	9474	69 k
172.217.27.202	192.168.0.105	192	26 k	96	13 k	96	12 k	1.367597	105.1011	1055	970
192.168.0.1	224.0.0.1	1	50	1	50	0	0	45.641376	0.0000	—	—
192.168.0.1	239.255.255.250	20	10 k	20	10 k	0	0	49.944015	0.2200	364 k	0
192.168.0.100	224.0.0.251	15	4713	15	4713	0	0	30.178012	66.9721	562	0
192.168.0.100	192.168.0.105	13	5252	13	5252	0	0	39.052039	62.9694	667	0

Name resolution | Limit to display filter | Absolute start time | Conversation Types

Copy | Follow Stream... | Graph... | Close | Help

Wireshark · Conversations · Wi-Fi

Ethernet · 17 | IPv4 · 669 | IPv6 · 3 | TCP · 597 | UDP · 175

Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
::	ff02::16	2	180	2	180	0	0	29.612052	0.6758	2130	0
fe80::b504:ff2d:1d1f:bffc	ff02::fb	1	107	1	107	0	0	36.096889	0.0000	—	—
fe80::b504:ff2d:1d1f:bffc	ff15::efc0:988f	9	1800	9	1800	0	0	57.074383	6.0473	2381	0

Name resolution | Limit to display filter | Absolute start time | Conversation Types

Copy | Follow Stream... | Graph... | Close | Help

Wireshark · Conversations · Wi-Fi

Ethernet · 17 | IPv4 · 669 | IPv6 · 3 | TCP · 597 | UDP · 175

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
13.107.6.254	443	192.168.0.105	51916	1	54	1	54	0	0	77.960923	0.0000	—	—
13.107.213.48	443	192.168.0.105	51936	1	54	1	54	0	0	77.754908	0.0000	—	—
40.100.138.18	443	192.168.0.105	51938	1	54	1	54	0	0	78.802007	0.0000	—	—
52.72.234.19	443	192.168.0.105	51397	8	509	4	293	4	216	26.183564	1.8032	1299	958
52.114.14.228	443	192.168.0.105	50210	10	2097	6	1350	4	747	17.592797	80.3298	134	74
78.8.215.253	1073	192.168.0.105	52104	8	1300	4	675	4	625	0.158870	7.2899	740	685
82.79.208.40	48797	192.168.0.105	50119	20	1088	10	540	10	548	10.003337	89.5061	48	48
117.18.232.200	443	192.168.0.105	51417	11	675	3	243	8	432	35.155263	58.1367	33	59
131.253.33.254	443	192.168.0.105	51937	1	54	1	54	0	0	79.880952	0.0000	—	—
162.125.19.131	443	192.168.0.105	50356	7	2790	4	334	3	2456	35.466374	0.2820	9474	69 k
192.168.0.105	52106	197.210.45.223	8973	6	329	6	329	0	0	0.000000	29.8488	88	0
192.168.0.105	52101	197.210.47.235	41403	4	219	4	219	0	0	0.291088	32.2778	54	0
192.168.0.105	52097	197.210.47.115	2535	3	164	3	164	0	0	0.322832	22.9823	57	0
192.168.0.105	52105	37.186.52.93	16606	6	664	6	664	0	0	0.402550	50.2501	105	0
192.168.0.105	52096	41.214.3.65	35817	1	54	1	54	0	0	0.668593	0.0000	—	—
192.168.0.105	52107	5.102.210.74	18264	9	824	5	596	4	228	0.671810	0.8560	5569	2130
192.168.0.105	49902	52.114.36.180	443	9	795	6	495	3	300	2.121544	80.4761	49	29
192.168.0.105	52108	35.186.224.25	443	29	6681	14	3939	15	2742	2.548692	3.6810	8560	5959
192.168.0.105	52098	186.65.223.63	6881	3	164	3	164	0	0	2.935700	31.0630	42	0
192.168.0.105	52100	197.210.64.12	31641	3	164	3	164	0	0	3.112686	28.4891	46	0

Name resolution | Limit to display filter | Absolute start time | Conversation Types

Copy | Follow Stream... | Graph... | Close | Help

Wireshark · Conversations · Wi-Fi

Ethernet · 17 | IPv4 · 669 | IPv6 · 3 | TCP · 597 | UDP · 175

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
41.79.159.73	8500	192.168.0.105	63841	2,620	184 k	1,346	88 k	1,274	95 k	0.158870	106.5187	6640	7206
41.97.67.248	15495	192.168.0.105	63841	117	46 k	55	3742	62	43 k	0.448333	105.6034	283	3259
85.210.31.50	6881	192.168.0.105	63841	6	380	6	380	0	0	8.949306	96.2884	31	0
85.210.31.50	6881	192.168.0.105	6881	2	268	2	268	0	0	12.495062	6.3423	338	0
89.64.84.35	37103	192.168.0.105	63841	1	69	1	69	0	0	6.286162	0.0000	—	—
100.69.140.50	12848	192.168.0.105	6881	3	186	3	186	0	0	2.399848	15.1298	98	0
100.69.140.50	52259	192.168.0.105	6881	2	498	1	146	1	352	5.333490	0.0008	—	—
100.69.140.50	1032	192.168.0.105	6881	2	497	1	146	1	351	6.315889	0.0009	—	—
100.69.140.50	1950	192.168.0.105	6881	2	497	1	146	1	351	8.610503	0.0019	—	—
100.69.140.50	1062	192.168.0.105	6881	2	497	1	146	1	351	12.015231	0.0017	—	—
100.69.140.50	6881	192.168.0.105	6881	6	1491	3	438	3	1053	12.220579	83.0470	42	101
100.69.140.50	63629	192.168.0.105	6881	2	498	1	146	1	352	13.589824	0.0016	—	—
100.69.140.50	37385	192.168.0.105	6881	20	2540	9	790	11	1750	15.422073	82.6282	76	169
100.69.140.50	11265	192.168.0.105	6881	6	1494	3	438	3	1056	15.893029	17.2586	203	489
100.69.140.50	1108	192.168.0.105	6881	2	497	1	146	1	351	21.995170	0.0009	—	—
100.69.140.50	63639	192.168.0.105	6881	9	1269	4	538	5	731	32.184866	74.3926	57	78
100.69.140.50	1109	192.168.0.105	6881	6	1491	3	438	3	1053	35.946230	55.0518	63	153
100.69.140.50	1041	192.168.0.105	6881	2	497	1	146	1	351	39.823402	0.0087	134 k	322 k
100.69.140.50	1047	192.168.0.105	6881	2	497	1	146	1	351	42.029889	0.0020	—	—
100.69.140.50	50928	192.168.0.105	6881	17	2180	9	1085	8	1095	49.230609	31.3681	276	279

Name resolution | Limit to display filter | Absolute start time | Conversation Types

Copy | Follow Stream... | Graph... | Close | Help

Detect Unsecured Application

Detect Suspicious Protocols and Application

Q2: List some other packet sniffing tools.

SolarWinds Deep Packet Inspection and Analysis Tool

Gives detailed insights into what causes network slowness and uses deep packet inspection to allow you to resolve the root causes. You can identify traffic by application, category and risk level to eliminate and filter problem traffic. With a great user interface, this excellent packet sniffing software is perfect for network analysis.

ManageEngine NetFlow Analyzer

A traffic analysis tool that works with NetFlow, J-Flow, sFlow Netstream, IPFIX, and AppFlow Paessler Packet Capture Tool A packet sniffer, a NetFlow sensor, an sFlow sensor, and a J-Flow sensor built into Paessler PRTG.

Omnipeek Network Protocol Analyzer

A network monitor that can be extended to capture packets.

tcpdump

The essential free packet capture tool that every network manager needs in his toolkit.

tshark

A lightweight answer to those who want the functionality of Wireshark, but the slim profile of tcpdump.

NetworkMiner

A Windows-based network analyzer with a no-frills free version.

Fiddler

A packet capture tool that focuses on HTTP traffic.

Capsa

Written for Windows, the free packet capture tool can be upgraded for payment to add on analytical features.