**Semester: V (2023-24) Subject:** Communication Protocols

**Name: Shreerang Mhatre Class: TY**

**Roll No: 52 Batch: A3**

**Experiment No: 05**

**Name of the Experiment**: Study of Network Protocol Analyser Tool/Software.

**Performed on: 25/10/2023**

**Submitted on: 30/10/2023**

**Aim:** Study of Network Protocol Analyser Tool/Software.

**Prerequisite:**

* Basic knowledge of Data communication and Networking protocol.

**Objectives:**

* Download Wireshark on Windows, Capture data packets and analyse them.

**Components and equipment required/studied:**

* Computer with Operating System installed and Internet connection
* Wireshark software.

**Theory:**

Wireshark is a free application that allows you to capture and view the data traveling back and forth on your network, providing the ability to drill down and read the contents of each packet – filtered to meet your specific needs. It is commonly utilized to troubleshoot network problems as well as to develop and test software. It is used for network troubleshooting, analysis, software and communications protocol development, and education.

This open-source protocol analyser is widely accepted as the industry standard, winning its fair share of awards over the years. There is also a terminal-based (non-GUI) version called Shark. Wireshark, and the other programs distributed with it such as TShark, are free software, released under the terms of the GNU General Public License. Originally known as Ethereal, Wireshark features a user-friendly interface that can display data from hundreds of different protocols on all major network types. These data packets can be viewed in Real-time or analysed offline, with dozens of capture/trace file formats supported including CAP and ERF. Integrated decryption tools allow you to view encrypted packets for several popular protocols such as WEP and WPA/WPA2.

Wireshark lets the user put network interface controllers into promiscuous mode (if supported by the network interface controller), so they can see all the traffic visible on that interface including unicast traffic not sent to that network interface controller's MAC address. However, when capturing with a packet analyser in promiscuous mode on a port on a network switch, not all traffic through the switch is necessarily sent to the port where the capture is done, so capturing in promiscuous mode is not necessarily sufficient to see all network traffic.

Port mirroring or various network taps extend capture to any point on the network. If a remote machine captures packets and sends the captured packets to a machine running Wireshark using the TZSP protocol or the protocol used by OmniPeek, Wireshark dissects those packets, so it can analyse packets captured on a remote machine at the time that they are captured.

**Features:**

Wireshark is a data capturing program that "understands" the structure (encapsulation) of different networking protocols. It can parse and display the fields, along with their meanings as specified by different networking protocols. Wireshark uses pcap to capture packets, so it can only capture packets on the types of networks that pcap supports. Data can be captured "from the wire" from a live network connection or read from a file of already-captured packets. Live data can be read from different types of networks, including Ethernet, IEEE 802.11, PPP, and loopback.

* Captured network data can be browsed via a GUI, or via the terminal (command line) version of the utility, TShark.
* Captured files can be programmatically edited or converted via command-line switches to the "edit-cap" program.
* Data display can be refined using a display filter.
* Plug-ins can be created for dissecting new protocols.
* VoIP calls in the captured traffic can be detected. If encoded in a compatible encoding, the media flow can even be played.
* Raw USB traffic can be captured.
* Wireless connections can also be filtered as long as they traverse the monitored Ethernet.
* Various settings, timers, and filters can be set to provide the facility of filtering the output of the captured traffic.

**Procedure:**

Downloading and Installing Wireshark: <https://www.wireshark.org/download.html>

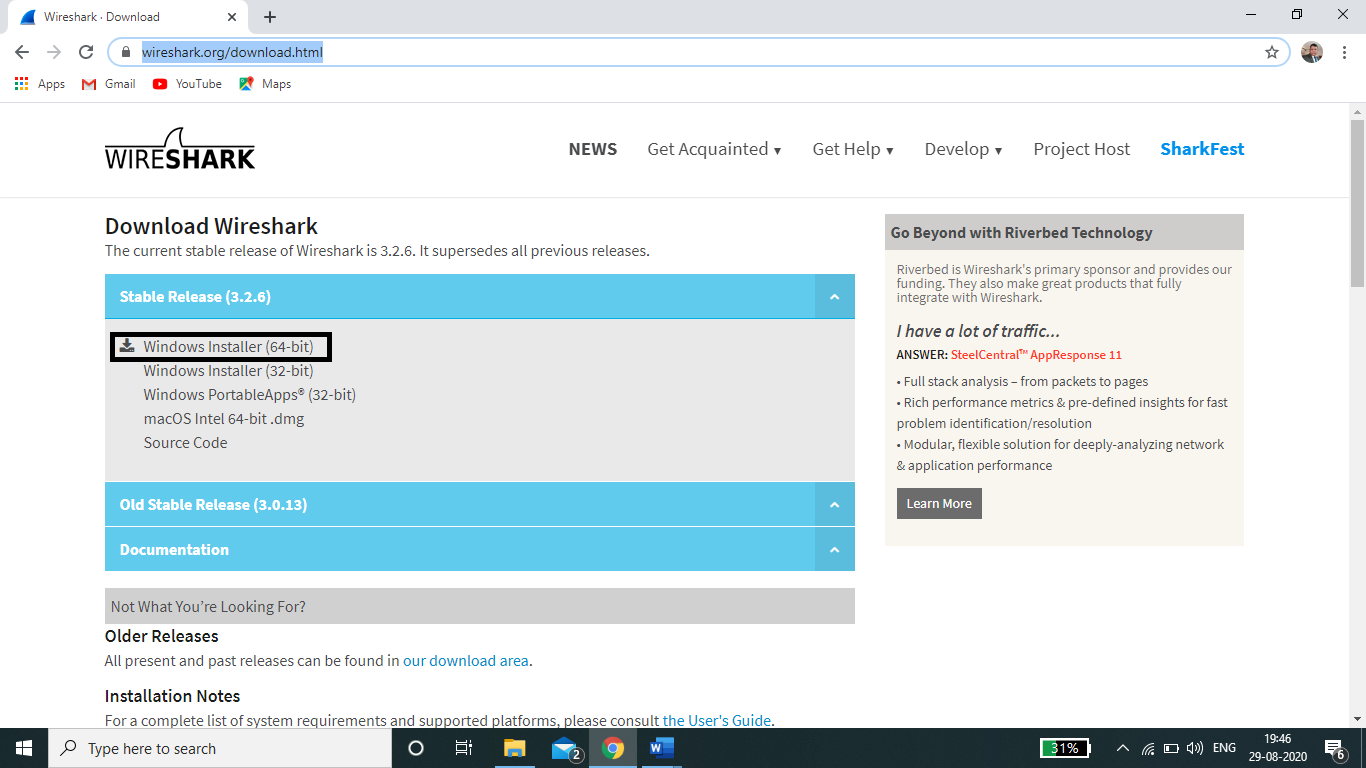
Wireshark can be downloaded at no cost from the Wireshark Foundation website for both macOS and Windows operating systems via above link. It is recommended that to download the latest stable release.

Fig. 5.1

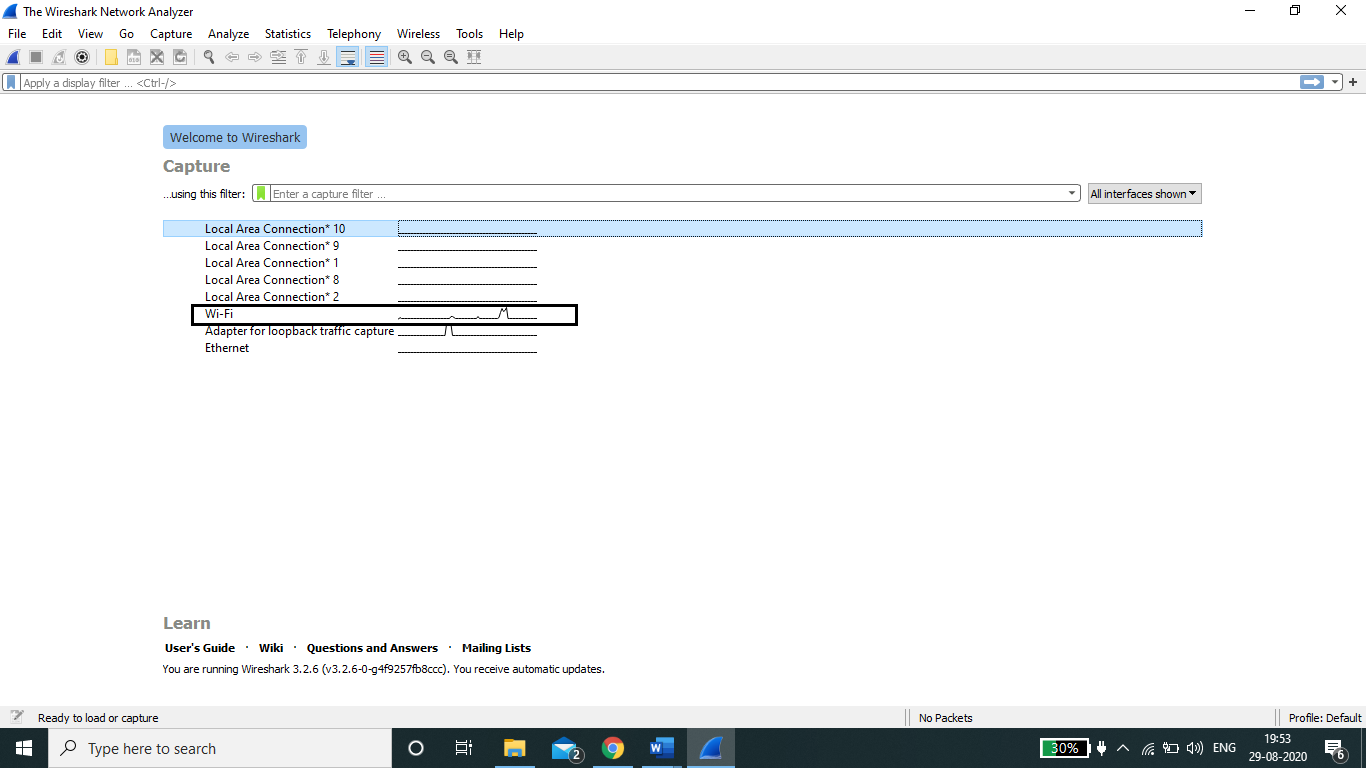
After downloading and installing the app, run the application and it will open a dialog box in which you can see all networks. There you click on Wi-Fi option; you can also take other option if available like Ethernet connection. (Fig. 5.2).

Fig. 5.2

For capturing packets, first select one or more of these networks by clicking on your choice(s) and using the *Shift* or *Ctrl* keys if you’d like to record data from multiple networks simultaneously. Once a connection type is selected for capturing purposes, its background will be shaded in either blue or grey. Click on *Capture* from the main menu, located towards the top of the Wireshark interface. When the dropdown menu appears, select the *Start* option.

You can also initiate packet capturing via one of the following shortcuts.

* **Keyboard:** Press *Ctrl + E*
* **Mouse:** To begin capturing packets from one particular network, simply double-click on its name
* **Toolbar:** Click on the blue shark fin button, located on the far left-hand side of the Wireshark toolbar

The live capture process will now begin, with packet details displayed in the Wireshark window as they are recorded. Perform one of the actions below to stop capturing.

* **Keyboard:** Press *Ctrl + E*
* **Toolbar:** Click on the red stop button, located next to the shark fin on the Wireshark toolbar.

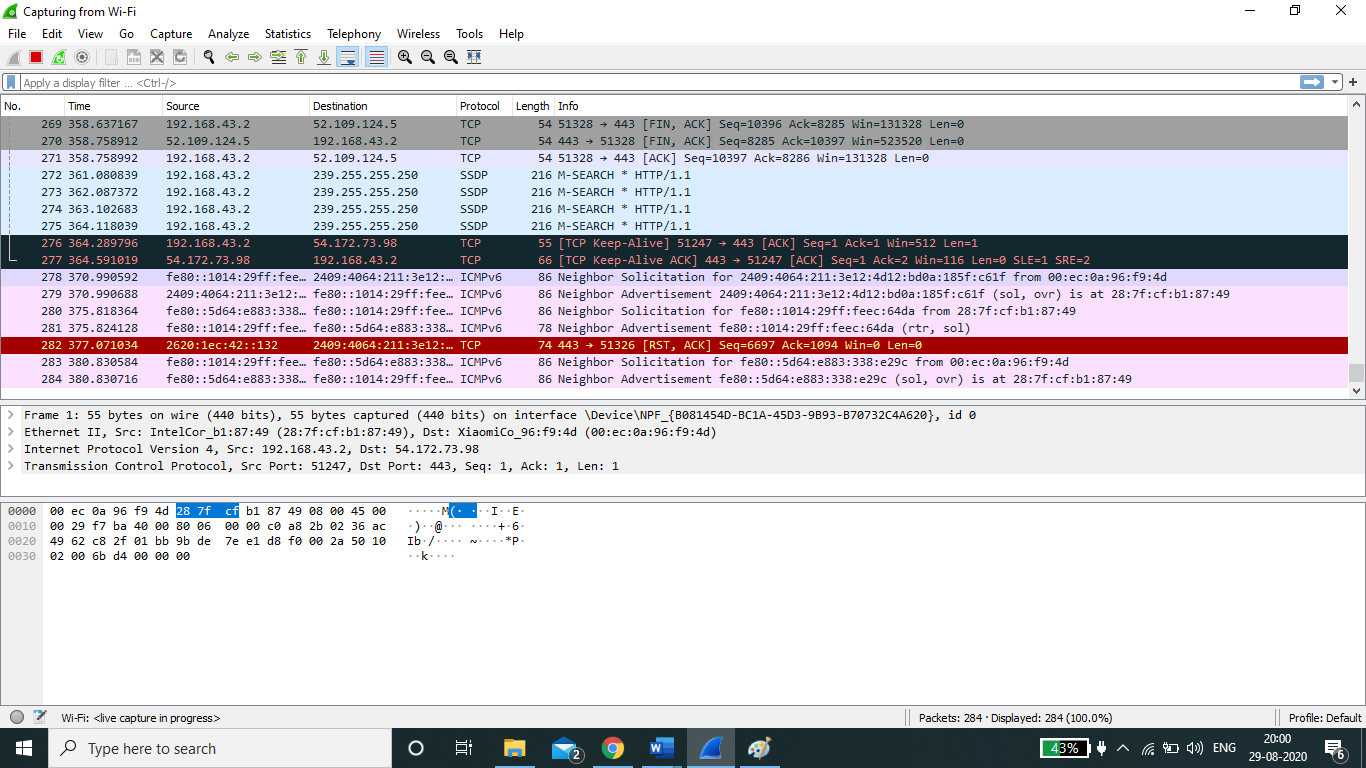


Fig:5.3

Now that you've recorded some network data it's time to take a look at the captured packets. As shown in the screenshot above, the captured data interface contains three main sections: The packet list pane, the packet details pane, and the packet bytes pane. Packet List

The packet list pane, located at the top of the window, shows all packets found in the active capture file. Each packet has its own row and corresponding number assigned to it, along with each of these data points.

* **Time:** The timestamp of when the packet was captured is displayed in this column, with the default format being the number of seconds (or partial seconds) since this specific capture file was first created. To modify this format to something that may be a bit more useful, such as the actual time of day, select the *Time Display Format* option from Wireshark's *View* menu - located at the top of the main interface.
* **Source:** This column contains the address (IP or other) where the packet originated.
* **Destination:** This column contains the address that the packet is being sent to.
* **Protocol:** The packet's protocol name (i.e., TCP) can be found in this column.
* **Length:** The packet length, in bytes, is displayed in this column.
* **Info:** Additional details about the packet are presented here. The contents of this column can vary greatly depending on packet contents.

When a packet is selected in the top pane, you may notice one or more symbols appear in the first column. Open and/or closed brackets, as well as a straight horizontal line, can indicate whether or not a packet or group of packets are all part of the same back-and-forth conversation on the network. A broken horizontal line signifies that a packet is not part of said conversation.

## **Packets Details:**

The details pane, found in the middle, presents the protocols and protocol fields of the selected packet in a collapsible format. In addition to expanding each selection, you can also apply individual Wireshark filters based on specific details as well as follow streams of data based on protocol type via the detail’s context menu – accessible by right-clicking your mouse on the desired item within this pane.

Packet Byte: At the bottom is the packet bytes pane, which displays the raw data of the selected packet in a hexadecimal view. This [hex dump](https://www.lifewire.com/xxd-linux-command-unix-command-4097149) contains 16 hexadecimal bytes and 16 ASCII bytes alongside the data offset.

Selecting a specific portion of this data automatically highlights its corresponding section in the packet details pane and vice versa. Any bytes that cannot be printed are instead represented by a period. You can choose to show this data in bit format as opposed to hexadecimal by right-clicking anywhere within the pane and selecting the appropriate option from the context menu.

Fig:5.4

Fig:5.4

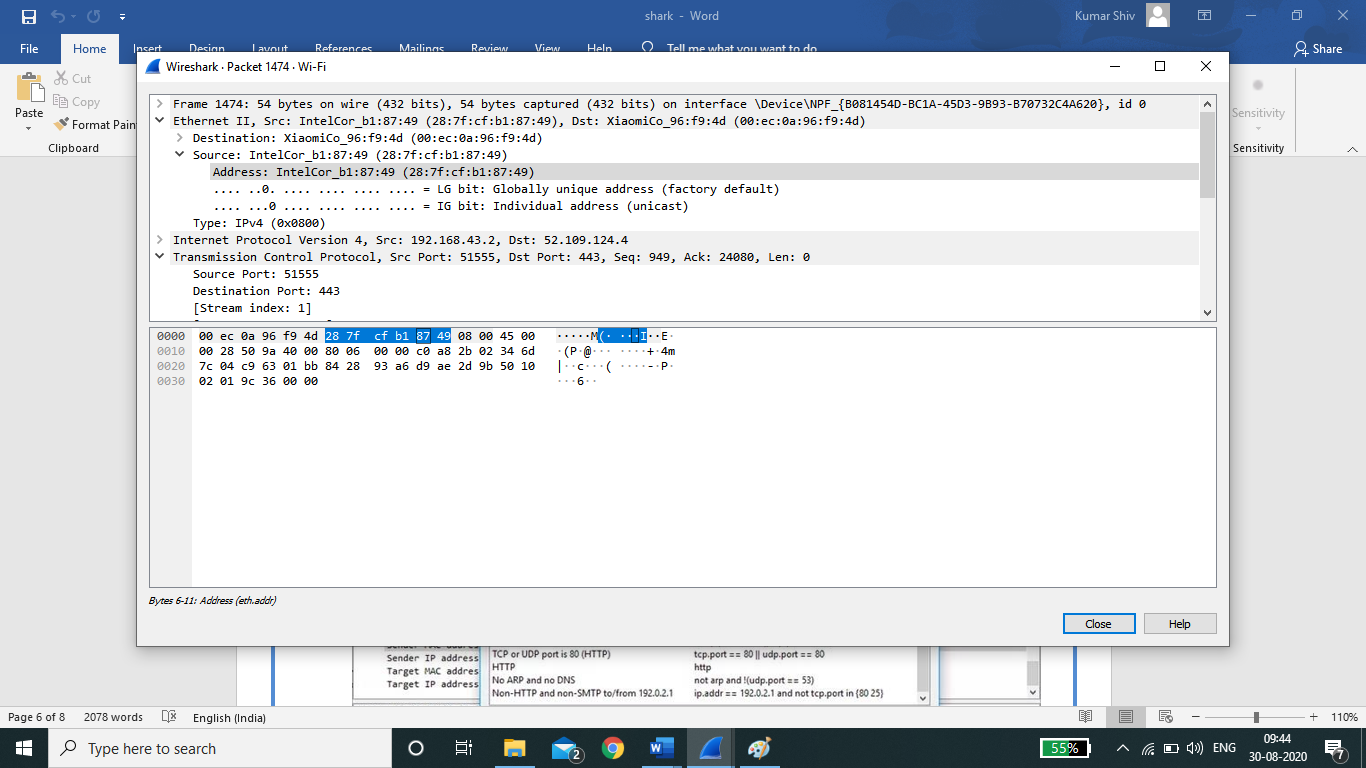


Fig:5.4

**Using Wireshark Filters**

One of the most important feature sets in Wireshark is its filter capabilities, especially when you're dealing with files that are significant in size. Capture filters can be set before the fact, instructing Wireshark to only record those packets that meet your specified criteria.

Filters can also be applied to a capture file that has already been created so that only certain packets are shown. These are referred to as display filters.

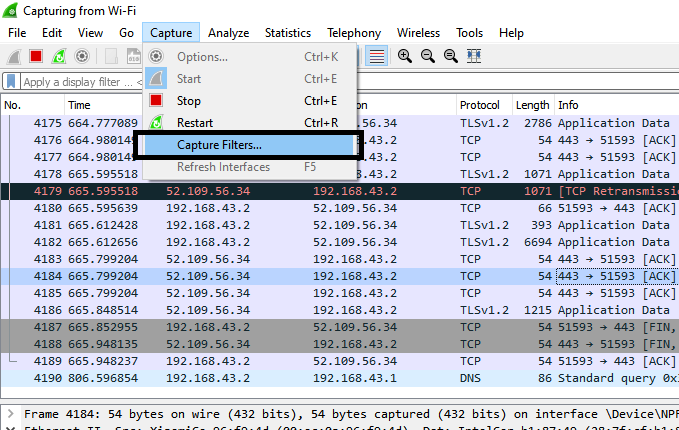


Fig. 5.6

Wireshark provides a large number of predefined filters by default, letting you narrow down the number of visible packets with just a few keystrokes or mouse clicks. To use one of these existing filters, place its name in the *Apply a display filter* entry field (located directly below the Wireshark toolbar) or in the *Enter a capture filter* entry field (located in the centre of the welcome screen).

There are multiple ways to achieve this. If you already know the name of your filter, simply type it into the appropriate field. For example, if you only wanted to display TCP packets you would type *tcp*. Wireshark's autocompleting feature will show suggested names as you begin typing, making it easier to find the correct moniker for the filter you're seeking.

Another way to choose a filter is to click on the bookmark-like icon positioned on the left-hand side of the entry field. This will present a menu containing some of the most commonly-used filters as well as an option to *Manage Capture Filters* or *Manage Display Filters*. If you choose to manage either type an interface will appear allowing you to add, remove or edit filters. One can also access previously-used filters by selecting the down arrow, located on the right-hand side of the entry field, which displays a history drop-down list. Once set, capture filters will be applied as soon as you begin recording network traffic. To apply a display filter, however, you'll need to click on the right arrow button found on the far-right hand side of the entry field.

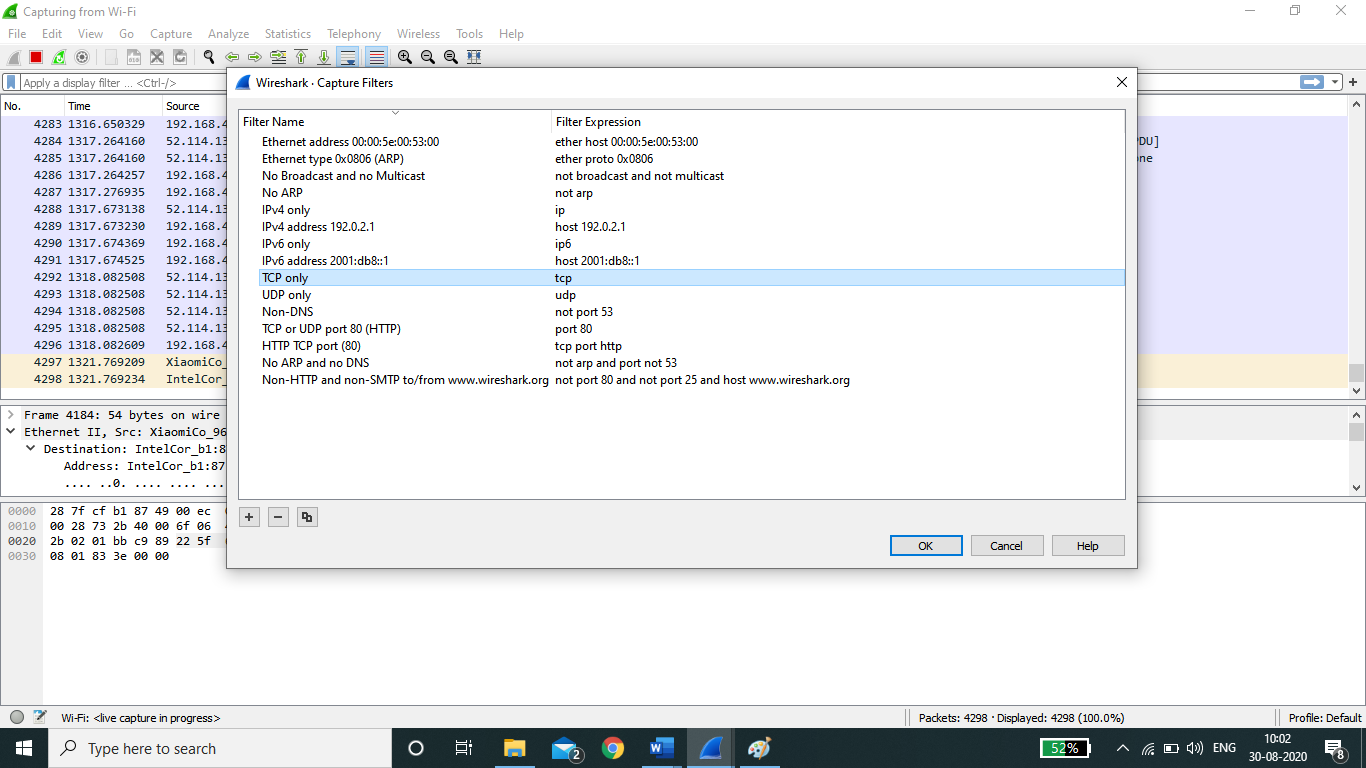


Fig.5.7

**Colouring Rules:**

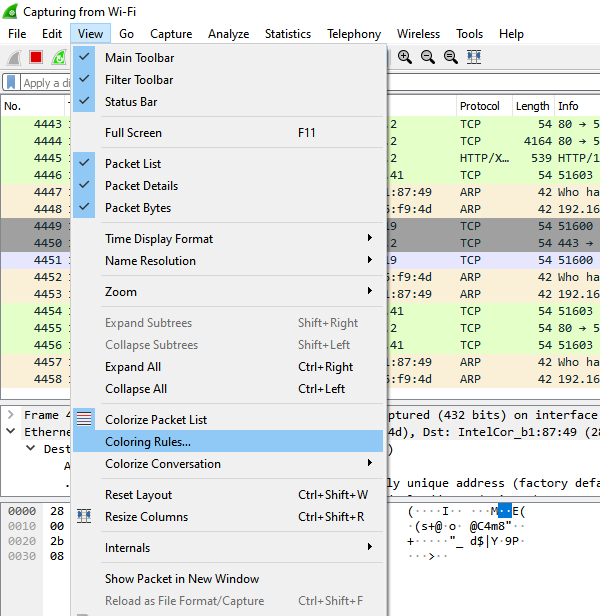


Fig. 5.8

While Wireshark's capture and display filters allow you to limit which packets are recorded or shown on the screen, its colorization functionality takes things a step further by making it easy to distinguish between different packet types based on their individual hue.

This handy feature lets you quickly locate certain packets within a saved set by their row's colour scheme in the packet list pane. Wireshark can colour packets based on rules that match particular fields in packets, to help the user identify the types of traffic at a glance.

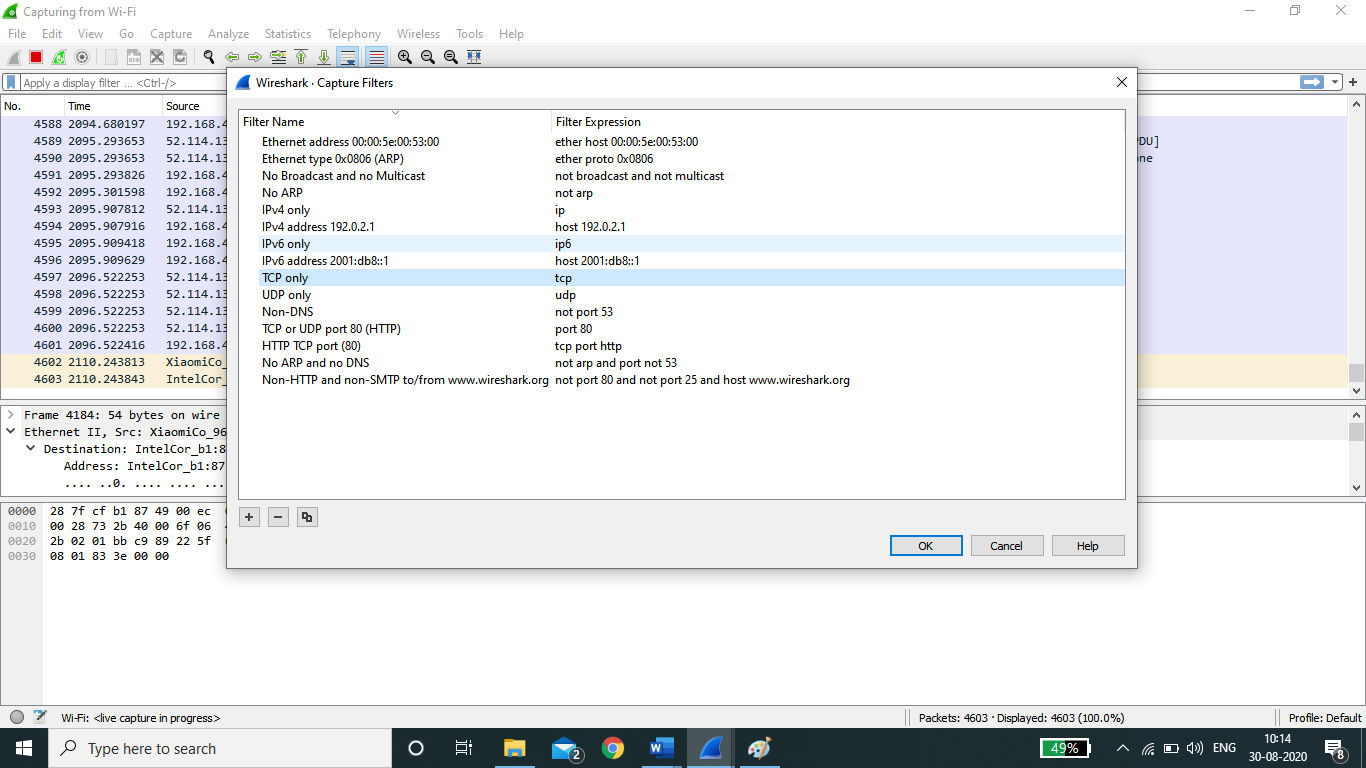
Wireshark comes with about 20 default colouring rules built in; each which can be edited, disabled, or deleted if you wish. You can also add new shade-based filters through the colouring rules interface, accessible from the *View* menu. In addition to defining a name and filter criteria for each rule, you are also asked to associate both a background colour and a text colour. Packet colorization can be toggled off and on via the *Colorize Packet List* option, also found within the *View* menu.

Fig. 5.9

**Simulation Packet Capture:**

Wireshark can also be used to capture packets from most network simulation tools such as [ns,](https://en.wikipedia.org/wiki/Ns_(simulator)) [OPNET](https://en.wikipedia.org/wiki/OPNET) Modeler, GNS3 and [NetSim.](https://en.wikipedia.org/w/index.php?title=NetSim&action=edit&redlink=1)

**Conclusion:**

----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Post Lab Questions:**

1. Enlist the other packet software tools you know.
2. What do we need to do packet capturing and its analysis?
3. If you capture TCP and UDP packets, what change you will observer in these packets.
4. What is the role of Filters in Wireshark?
5. How can one can use Wireshark software in day to day life and in Industry?

