Wireshark Lab 2.0. Filtering Packets (V1.1)

OVERVIEW

Wireshark and TShark share a powerful filter engine that helps remove the noise from a packet trace, and lets you see only the packets that interest you.

If a packet meets the requirements expressed in your filter, it is displayed in the list of packets. Display filters let you compare the fields within a protocol against a specific value, compare fields against fields, and check the existence of specified fields or protocols.

OBJECTIVE:

1- How to filter specific data using Wireshark.
2- How to combine multiple filters.
3- How to create quick filter buttons.
4- How to filter conversations.

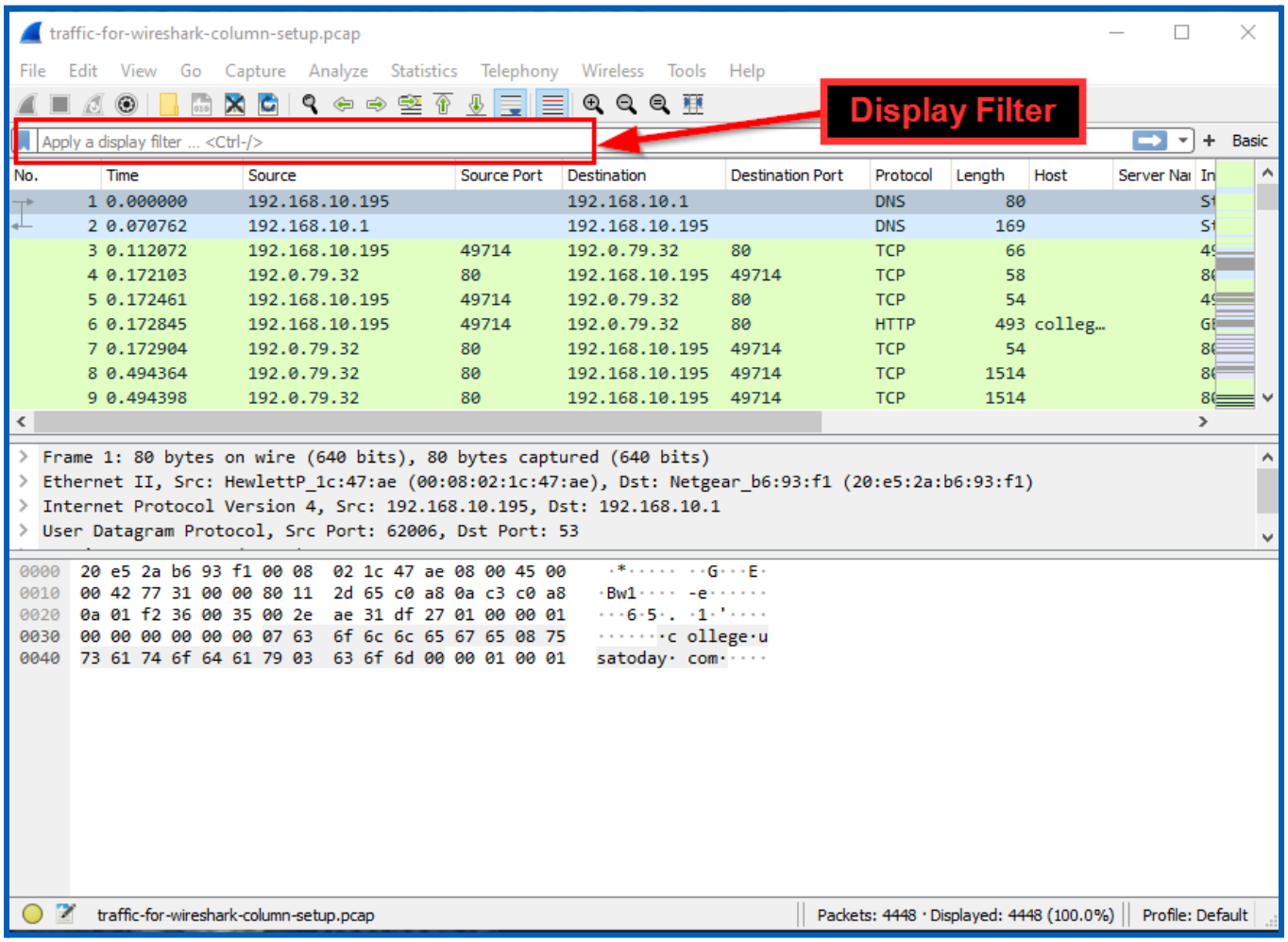
REQUIREMENTS:

- ☐ Wireshark Application
- ☐ OS (Windows, macOS, or Linux)

STEPS:

Part 1 - The Wireshark Display Filter

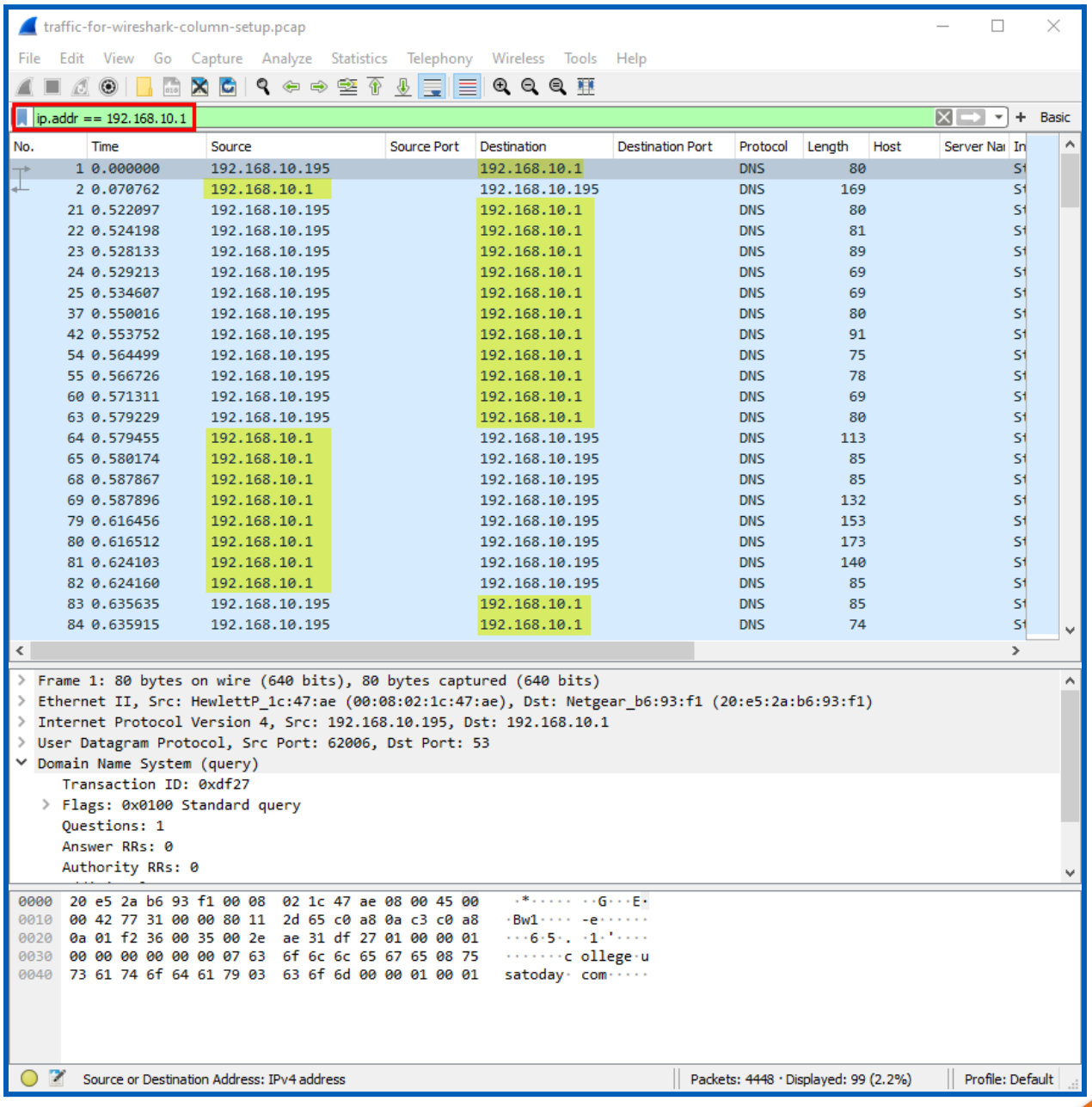
Wireshark's display filter is a bar located right above the column display section. This is where you type expressions to filter the frames, IP packets, or TCP segments that Wireshark displays from a pcap.



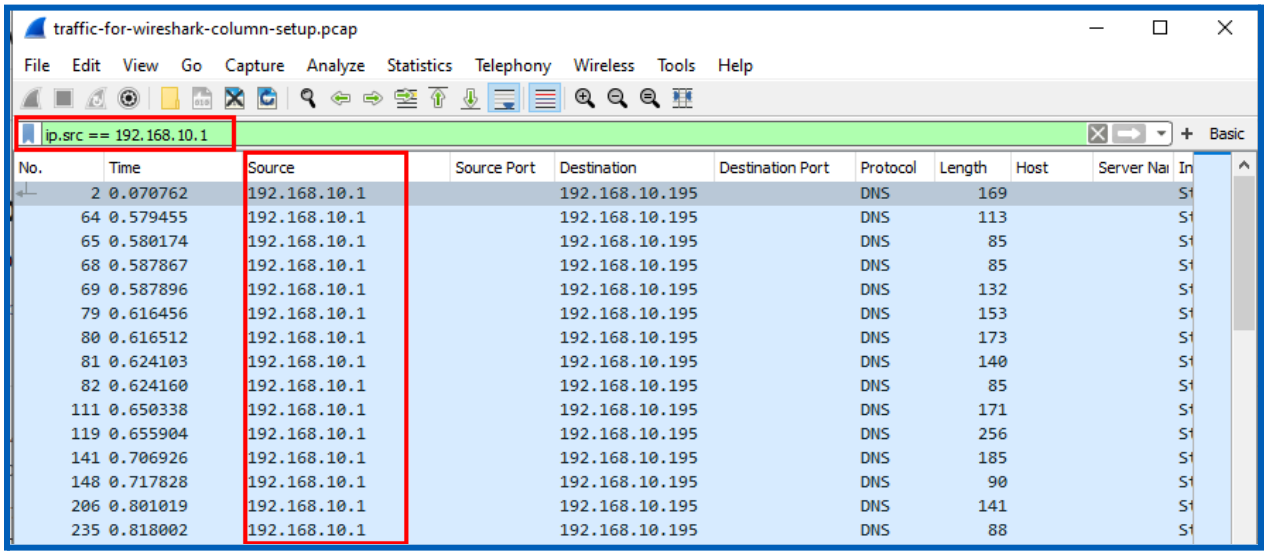
Part 2 - Filtering for IP Addresses , Sources, and Destinations

In this lab, we are going to examine the **(traffic-for-wireshark-column-setup)** PCAP file.

- ☐ Download the PCAP for this lab using this link. **Password: infected**
  - ☐ **Extract the file after you download it, and open the .PCAP file.**
- ☐ Open the .PCAP file in Wireshark.
- ☐ We are going to filter the traffic to just show packets with IP address: 192.168.10.1
- ☐ **Filter : ip.addr == 192.168.10.1**



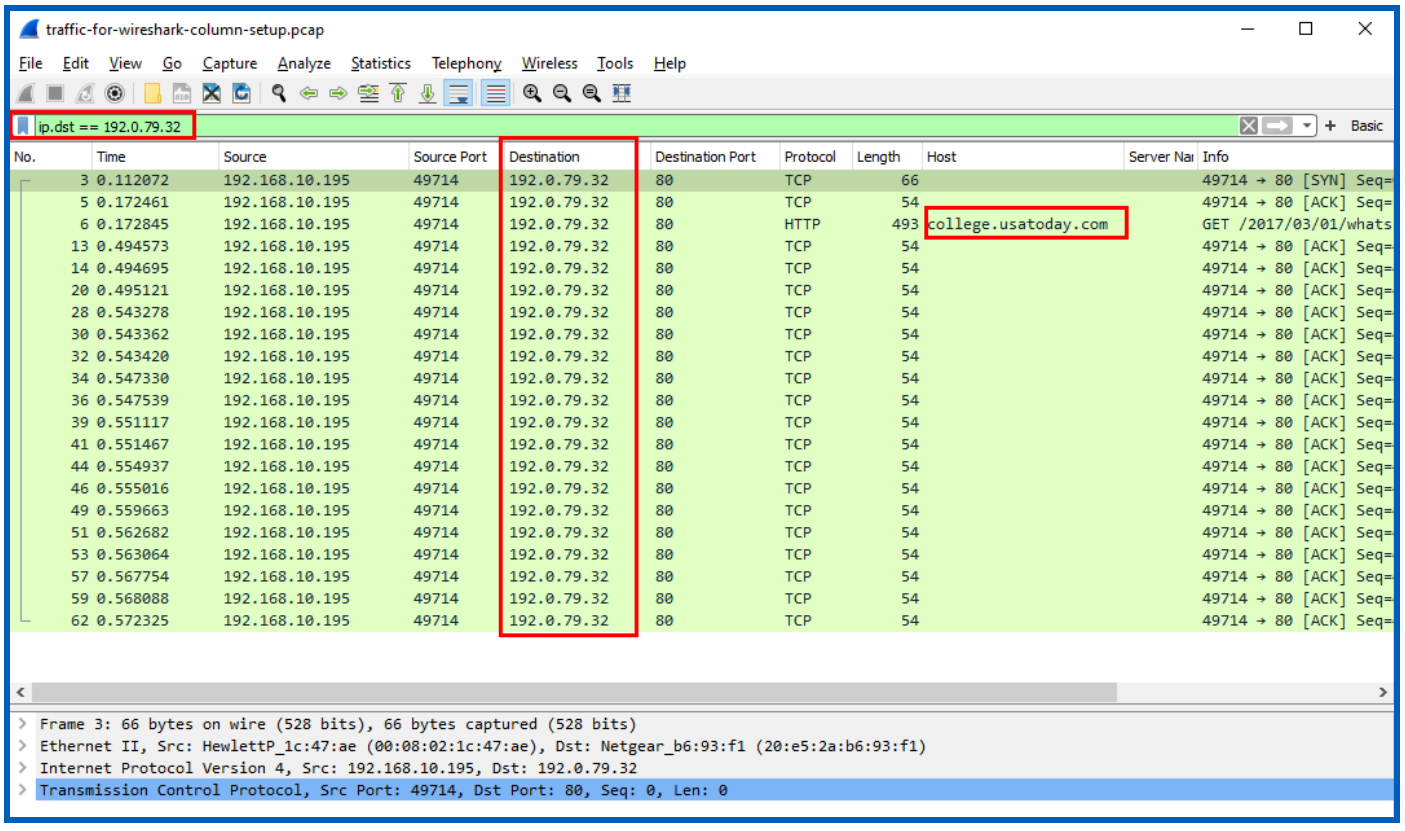
If we only want the packets with the source IP address 192.168.10.1, we apply the following filter: **ip.src == 192.168.10.1**



No.	Time	Source	Source Port	Destination	Destination Port	Protocol	Length	Host	Server Nai	In
2	0.070762	192.168.10.1		192.168.10.195		DNS	169			St
64	0.579455	192.168.10.1		192.168.10.195		DNS	113			St
65	0.580174	192.168.10.1		192.168.10.195		DNS	85			St
68	0.587867	192.168.10.1		192.168.10.195		DNS	85			St
69	0.587896	192.168.10.1		192.168.10.195		DNS	132			St
79	0.616456	192.168.10.1		192.168.10.195		DNS	153			St
80	0.616512	192.168.10.1		192.168.10.195		DNS	173			St
81	0.624103	192.168.10.1		192.168.10.195		DNS	140			St
82	0.624160	192.168.10.1		192.168.10.195		DNS	85			St
111	0.650338	192.168.10.1		192.168.10.195		DNS	171			St
119	0.655904	192.168.10.1		192.168.10.195		DNS	256			St
141	0.706926	192.168.10.1		192.168.10.195		DNS	185			St
148	0.717828	192.168.10.1		192.168.10.195		DNS	90			St
206	0.801019	192.168.10.1		192.168.10.195		DNS	141			St
235	0.818002	192.168.10.1		192.168.10.195		DNS	88			St

college.usatoday.com server has an IP address of 192.0.79.32. We simply want to filter only the destination packets.

To do that, we use the following filter: **ip.dst == 192.0.79.32**

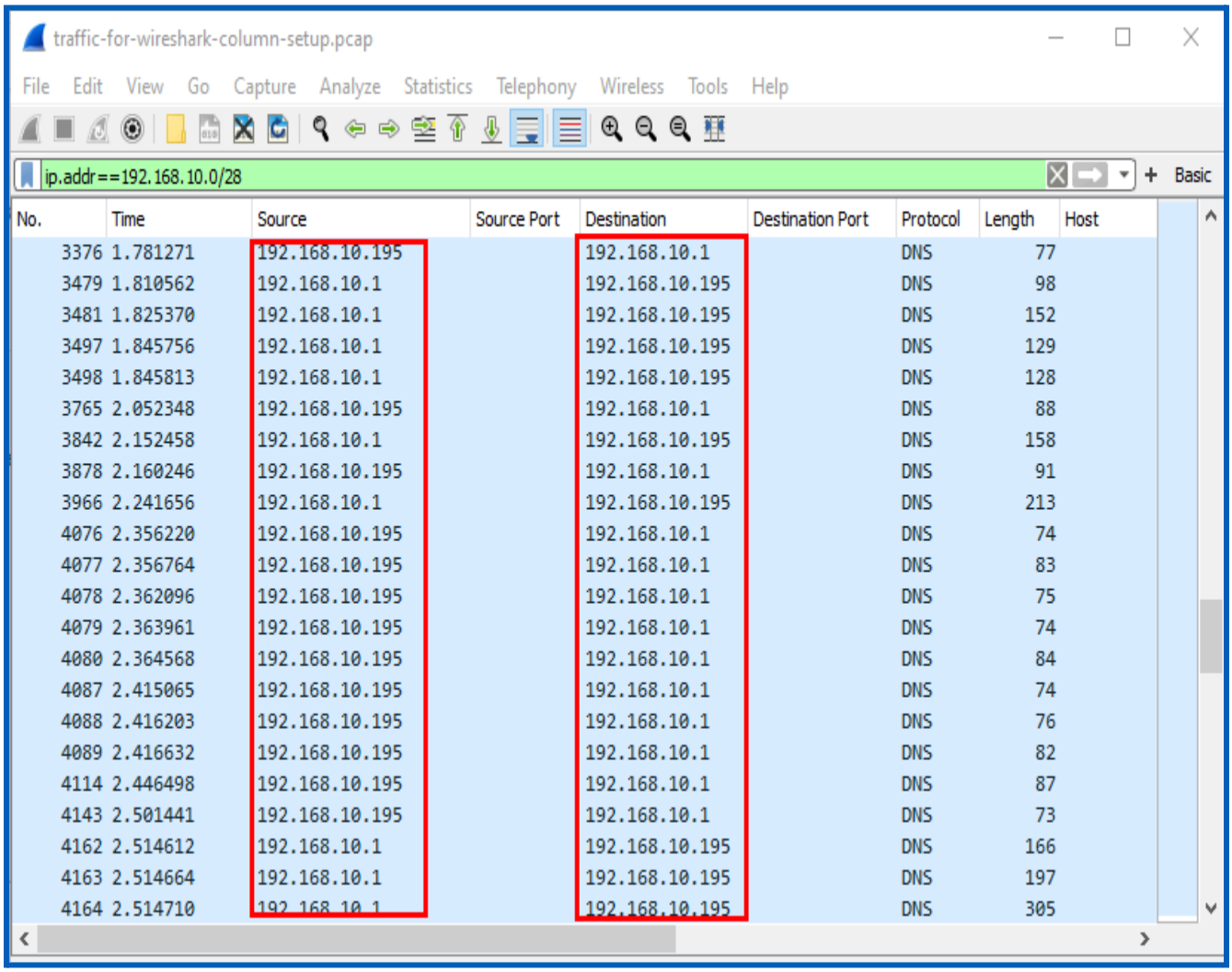


No.	Time	Source	Source Port	Destination	Destination Port	Protocol	Length	Host	Server Nai	Info
3	0.112072	192.168.10.195	49714	192.0.79.32	80	TCP	66			49714 → 80 [SYN] Seq=
5	0.172461	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
6	0.172845	192.168.10.195	49714	192.0.79.32	80	HTTP	493	college.usatoday.com		GET /2017/03/01/whats
13	0.494573	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
14	0.494695	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
20	0.495121	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
28	0.543278	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
30	0.543362	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
32	0.543420	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
34	0.547330	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
36	0.547539	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
39	0.551117	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
41	0.551467	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
44	0.554937	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
46	0.555016	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
49	0.559663	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
51	0.562682	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
53	0.563064	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
57	0.567754	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
59	0.568088	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=
62	0.572325	192.168.10.195	49714	192.0.79.32	80	TCP	54			49714 → 80 [ACK] Seq=

```
> Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
> Ethernet II, Src: HewlettP_1c:47:ae (00:08:02:1c:47:ae), Dst: Netgear_b6:93:f1 (20:e5:2a:b6:93:f1)
> Internet Protocol Version 4, Src: 192.168.10.195, Dst: 192.0.79.32
> Transmission Control Protocol, Src Port: 49714, Dst Port: 80, Seq: 0, Len: 0
```

Another thing we will do is filter for a specified subnet. The purpose here is to view everything that goes to and from a specified subnet.

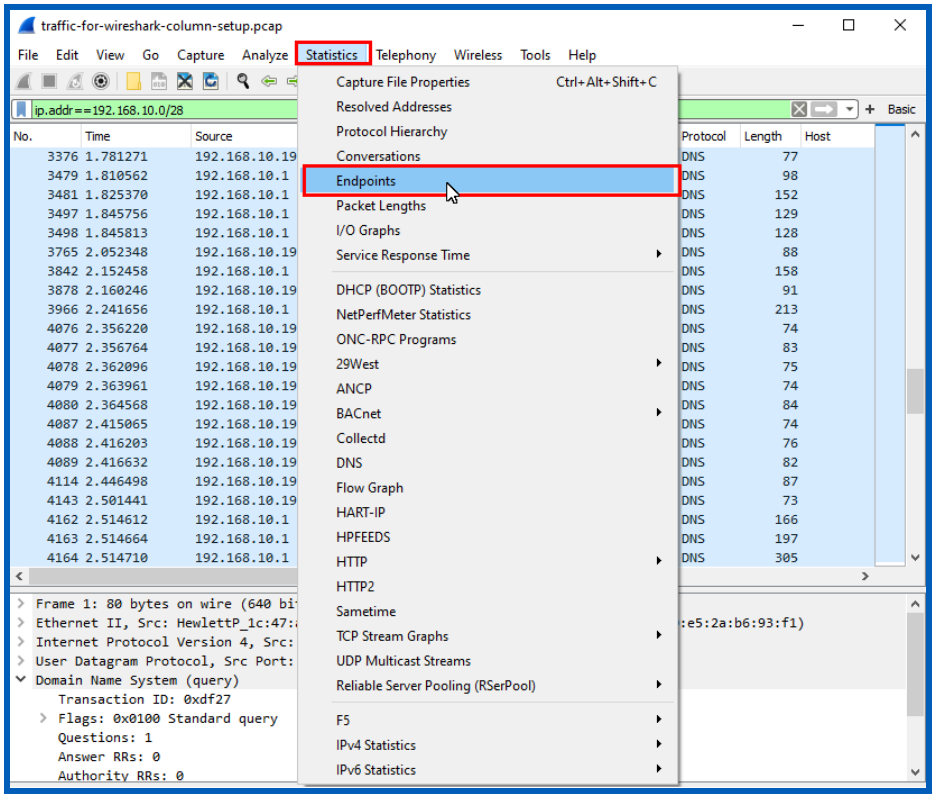
- ☐ You can simply use that format with the **ip.addr ==** or **ip.addr eq** display filter.
- ☐ The network ID that we want to filter is: **192.168.10.0.**
- ☐ The subnet that we are going to use is /28, so our filter will be: **ip.addr==192.168.10.0/28.**



No.	Time	Source	Source Port	Destination	Destination Port	Protocol	Length	Host
3376	1.781271	192.168.10.195		192.168.10.1		DNS	77	
3479	1.810562	192.168.10.1		192.168.10.195		DNS	98	
3481	1.825370	192.168.10.1		192.168.10.195		DNS	152	
3497	1.845756	192.168.10.1		192.168.10.195		DNS	129	
3498	1.845813	192.168.10.1		192.168.10.195		DNS	128	
3765	2.052348	192.168.10.195		192.168.10.1		DNS	88	
3842	2.152458	192.168.10.1		192.168.10.195		DNS	158	
3878	2.160246	192.168.10.195		192.168.10.1		DNS	91	
3966	2.241656	192.168.10.1		192.168.10.195		DNS	213	
4076	2.356220	192.168.10.195		192.168.10.1		DNS	74	
4077	2.356764	192.168.10.195		192.168.10.1		DNS	83	
4078	2.362096	192.168.10.195		192.168.10.1		DNS	75	
4079	2.363961	192.168.10.195		192.168.10.1		DNS	74	
4080	2.364568	192.168.10.195		192.168.10.1		DNS	84	
4087	2.415065	192.168.10.195		192.168.10.1		DNS	74	
4088	2.416203	192.168.10.195		192.168.10.1		DNS	76	
4089	2.416632	192.168.10.195		192.168.10.1		DNS	82	
4114	2.446498	192.168.10.195		192.168.10.1		DNS	87	
4143	2.501441	192.168.10.195		192.168.10.1		DNS	73	
4162	2.514612	192.168.10.1		192.168.10.195		DNS	166	
4163	2.514664	192.168.10.1		192.168.10.195		DNS	197	
4164	2.514710	192.168.10.1		192.168.10.195		DNS	305	

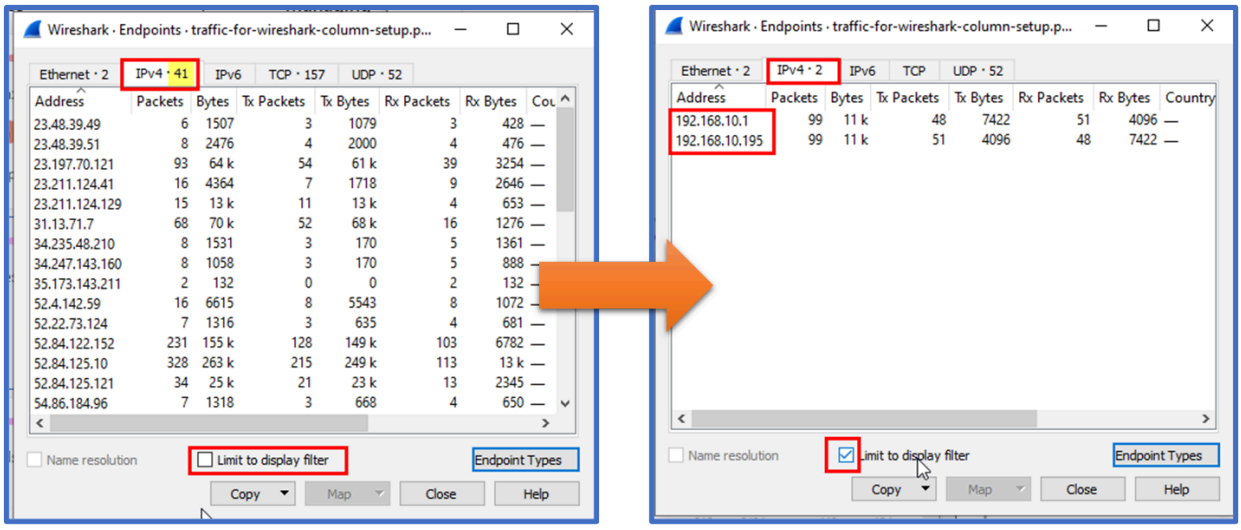
The question here is, **how do you know what addresses are actually involved in that subnet?**

After you set your filter, open the **statistics** view, and then click on **endpoints** as shown in the picture below.



Now, there is a very important checkbox that is very easy to overlook **in the statistics endpoints section because it is small.**

If you come down to the **limit to display filters** as shown in the picture below, only traffic that matches our filter will be displayed.



41 IP Addresses before applying the filter

2 IP Addresses after applying the filter

Part 3 - Filtering for Protocols and Port Numbers:

Wireshark has the ability to filter network traffic or packets according to the port or port number.

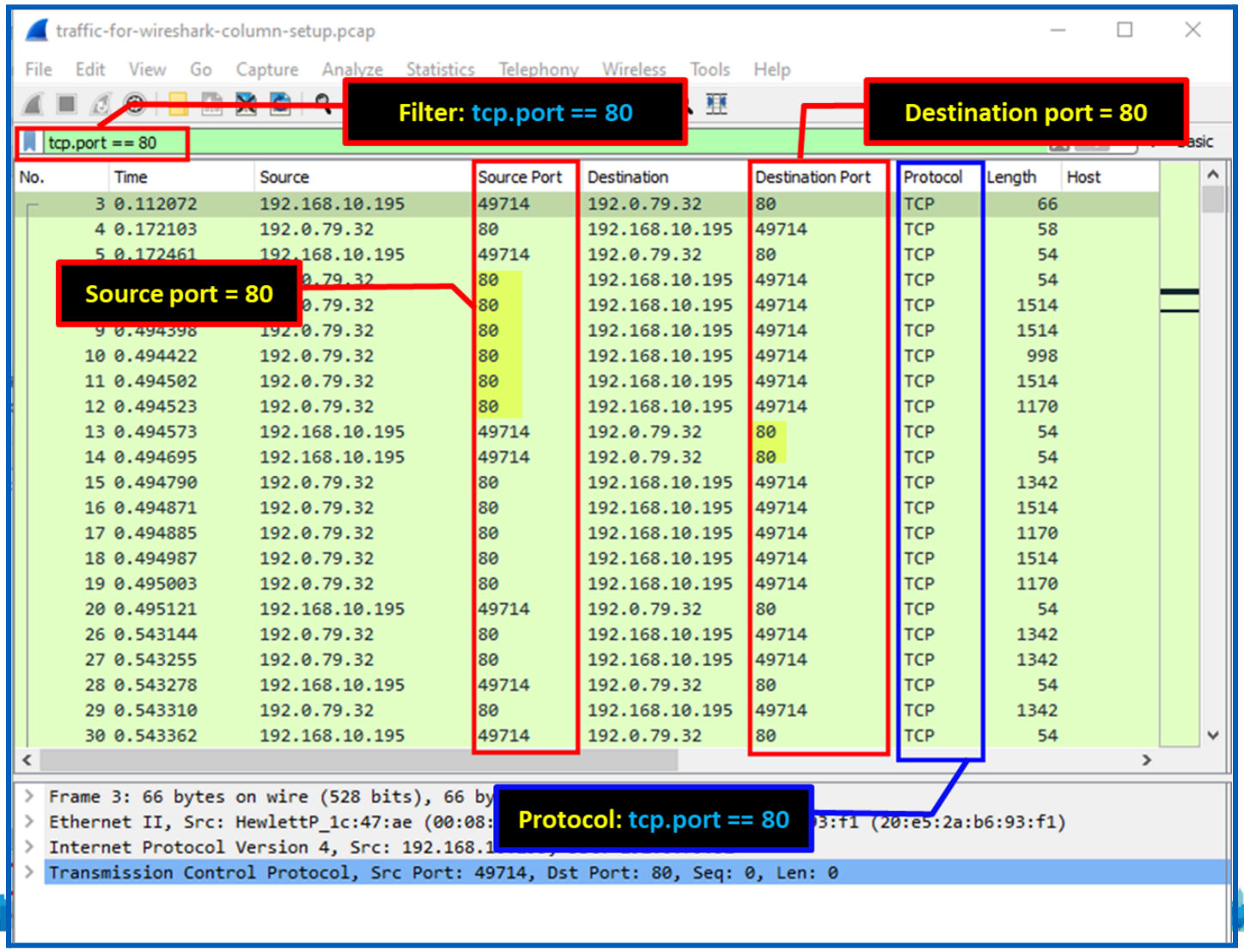
Protocols such as TCP and UDP use port numbers. TCP and UDP are the most common transmission protocols, and most network applications, such as websites, web apps, services, etc. use them.

A- Filter According to TCP or UDP Port Number

HTTP port number is 80 and is using TCP.
As the **tcp.port == 80** is used to filter port number 80, the **==** can be changed with the **eq,** which is the short form of "equal."

Filter : **tcp.port eq 80 or tcp.port == 80**

We are going to use the same .PCAP File
(raffic-for-wireshark-column-setup.PCAP).

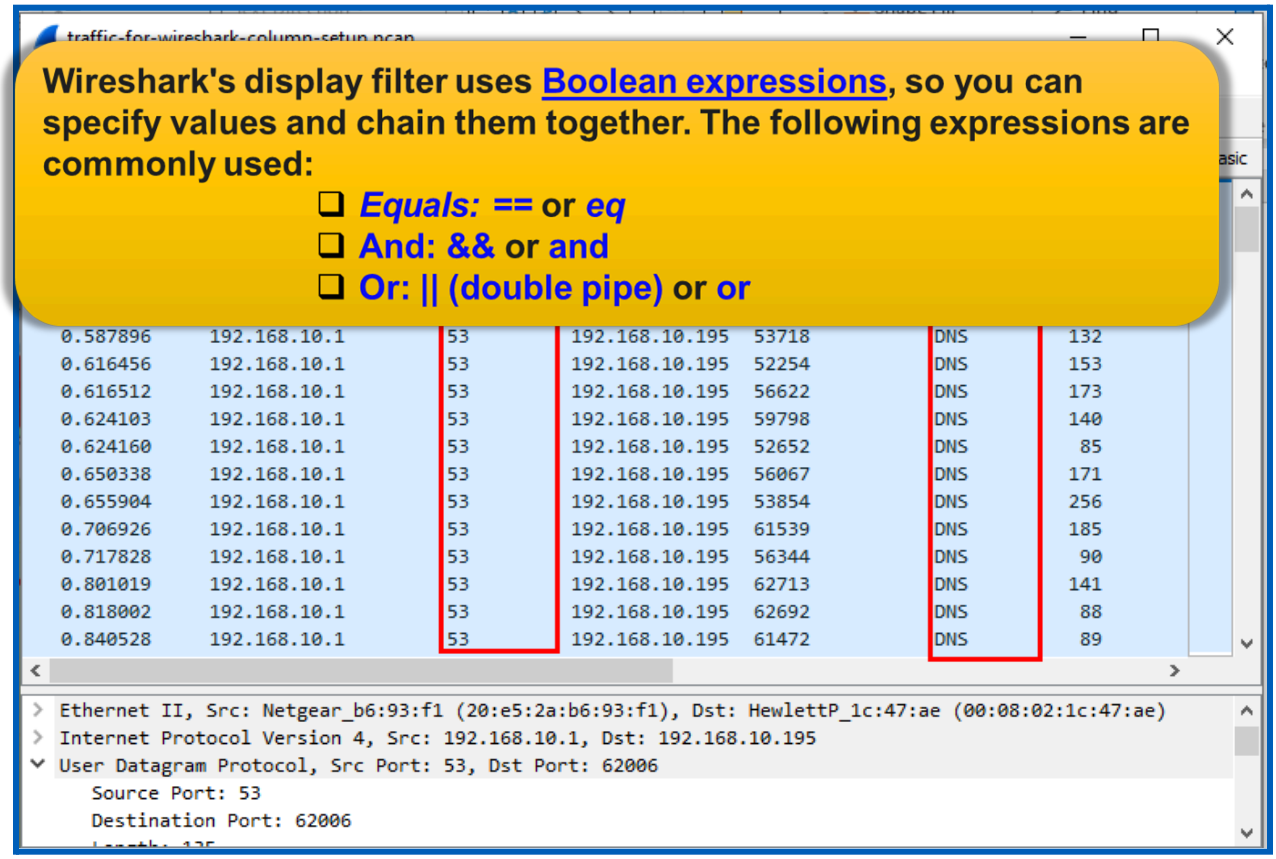


B- Filter According to TCP or UDP with source Port Number

DNS has always been designed to use both **UDP** and **TCP** port **53** from the start, with **UDP** being the default, and falling back to using **TCP** when it is unable to communicate on **UDP**.

Let's first check for **DNS TCP** packets, and hen the **UDP**.
Filter : **tcp.port eq 53 or tcp.port == 53**

As shown in the picture below, there are no **DNC TCP** packets.
Now, let's check for **DNS UCP** packets.
Filter : **udp.port eq 53 or udp.port == 53.**

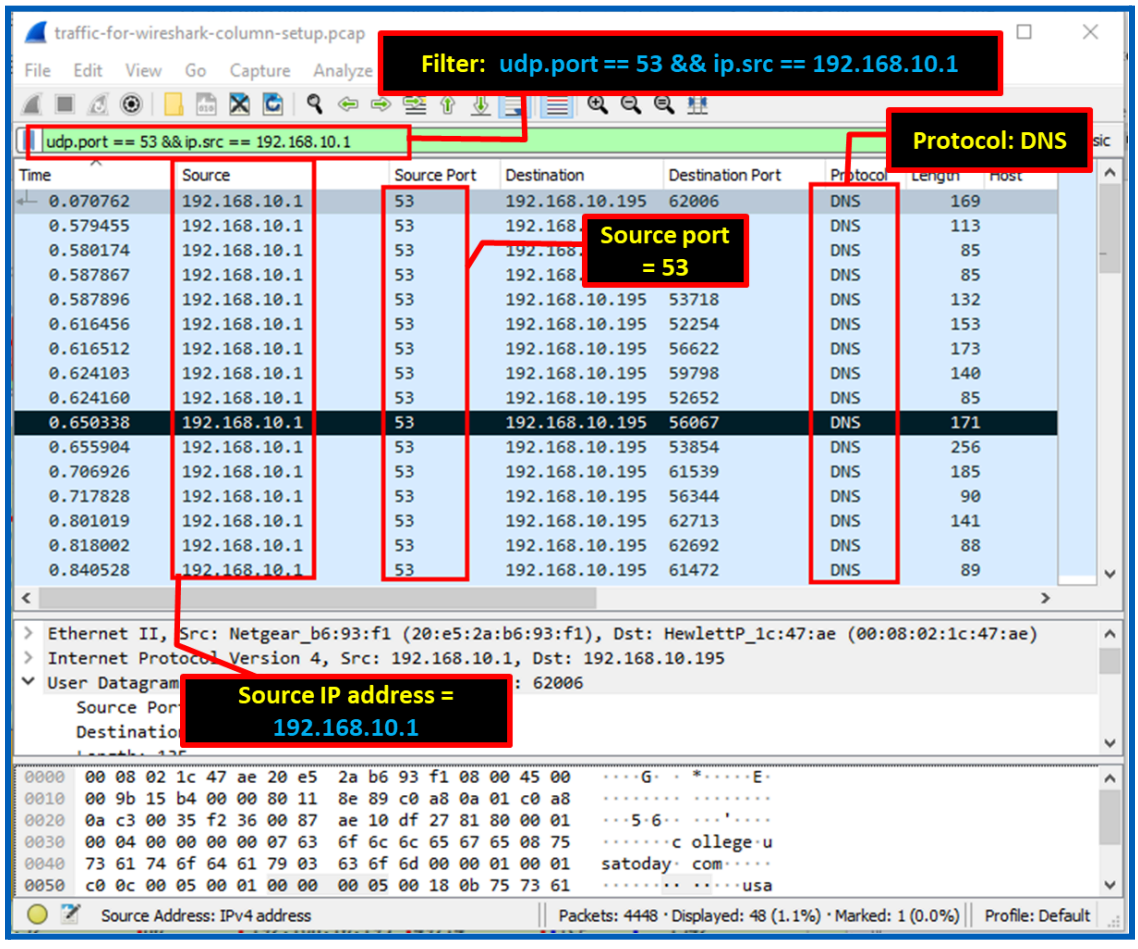


Part 4 - Combining multiple filter queries:

One thing you will need to write a combined display filter is knowledge of **boolean operators**. We are talking about the basic **AND**, **OR,** and **NOT operations**. Using these basic operators, we can also combine multiple filter queries into one. For example, if we are looking for **UDP** traffic and packets utilizing the source IP address **192.168.10.1,** we can write the filter as:

udp.port == 53 and ip.src == 192.168.10.1
Another way is to use the expression:
udp.port == 53 && ip.src == 192.168.10.1

The picture below shows the results after running the following filter:
udp.port == 53 && ip.src == 192.168.10.1



Part 5 - Creating a button that allows you to filter certain packets:

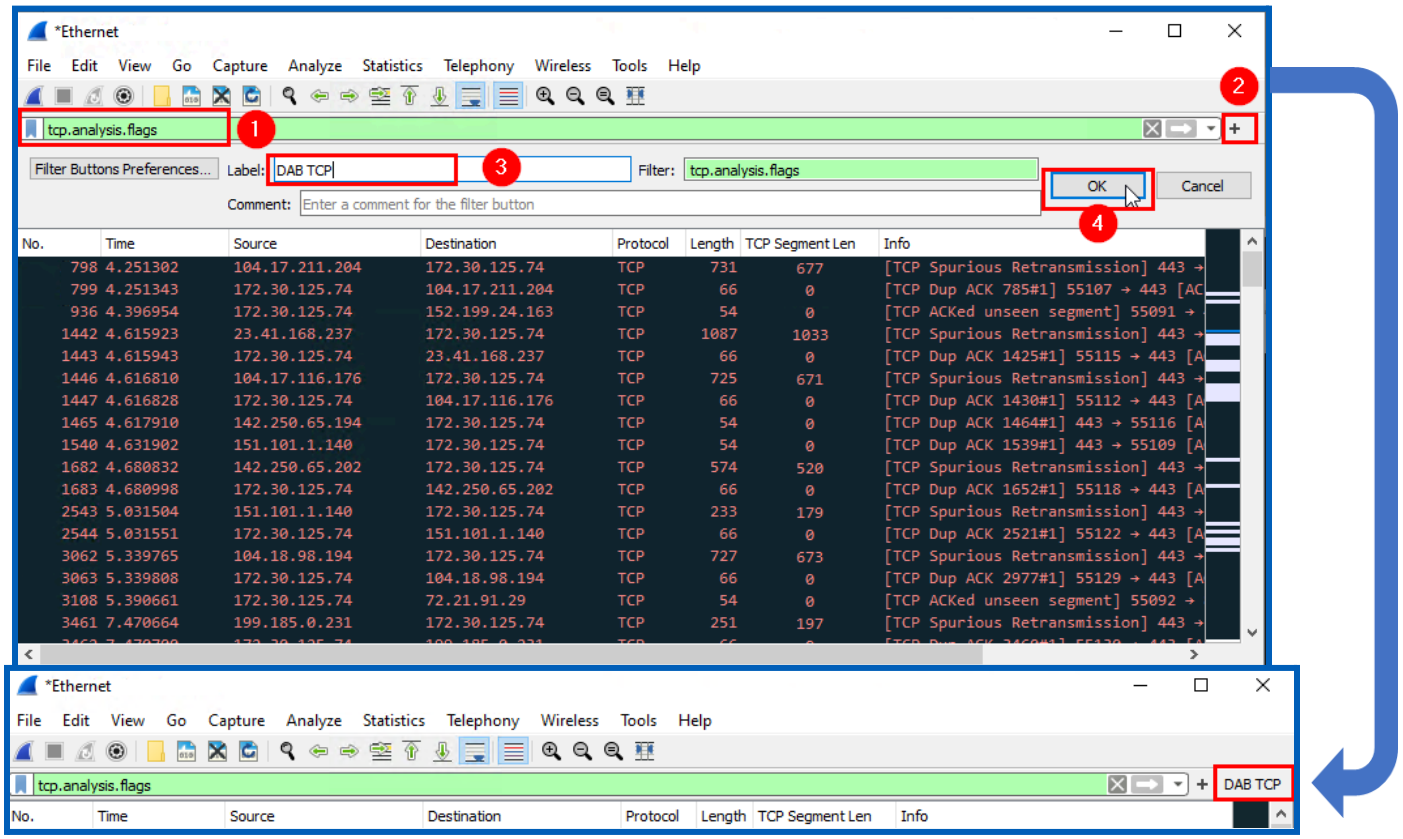
We make buttons to help us quickly filter recorded traffic. For example, we can make a button for all **HTTP.request** packets, **UDP packets**, or **FTP packets**.

What are TCP Analysis Flags?

In a TCP connection, flags are used to indicate a particular state of connection, or to provide additional and useful information for troubleshooting or for handling the control of a particular connection.

In this lab, we are going to create a button for the TCP **analysis flags** filter.

Filter the traffic with **tcp.analysis.flags, and then** follow the steps in the picture below.

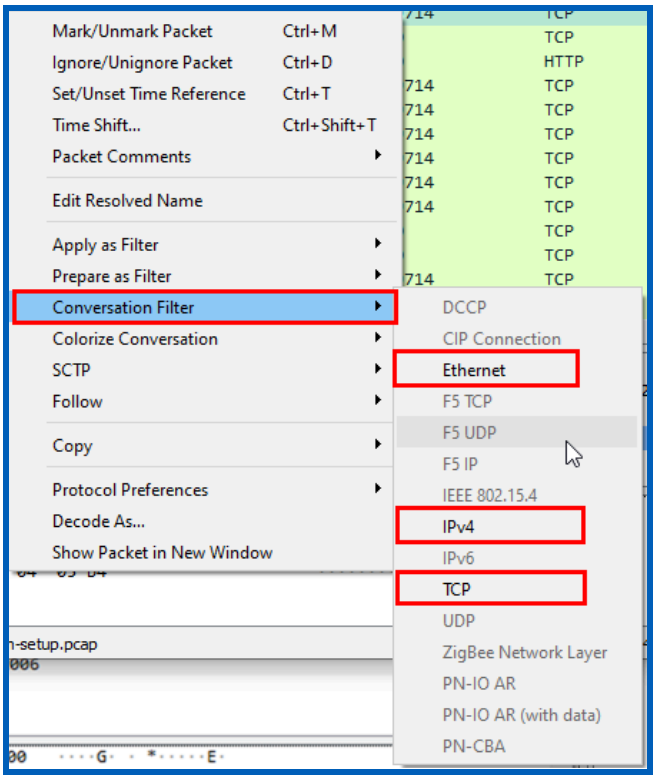


We now have a quick filter search **(BAD TCP)** as shown in the picture above.

Part 6 - Filtering for Conversations

Conversations:

A network conversation is the traffic between two specific endpoints. For example, an IP conversation contains all of the traffic between two IP addresses.



Conversation Filter:

Ethernet: You can filter on everything between two Mac addresses (Layer 2).

IPV4: You can filter on everything between those two IP addresses (Layer 3).

TCP: You have everything between those two port numbers, which are also between the two IP addresses. (Layer 3 + Layer4)

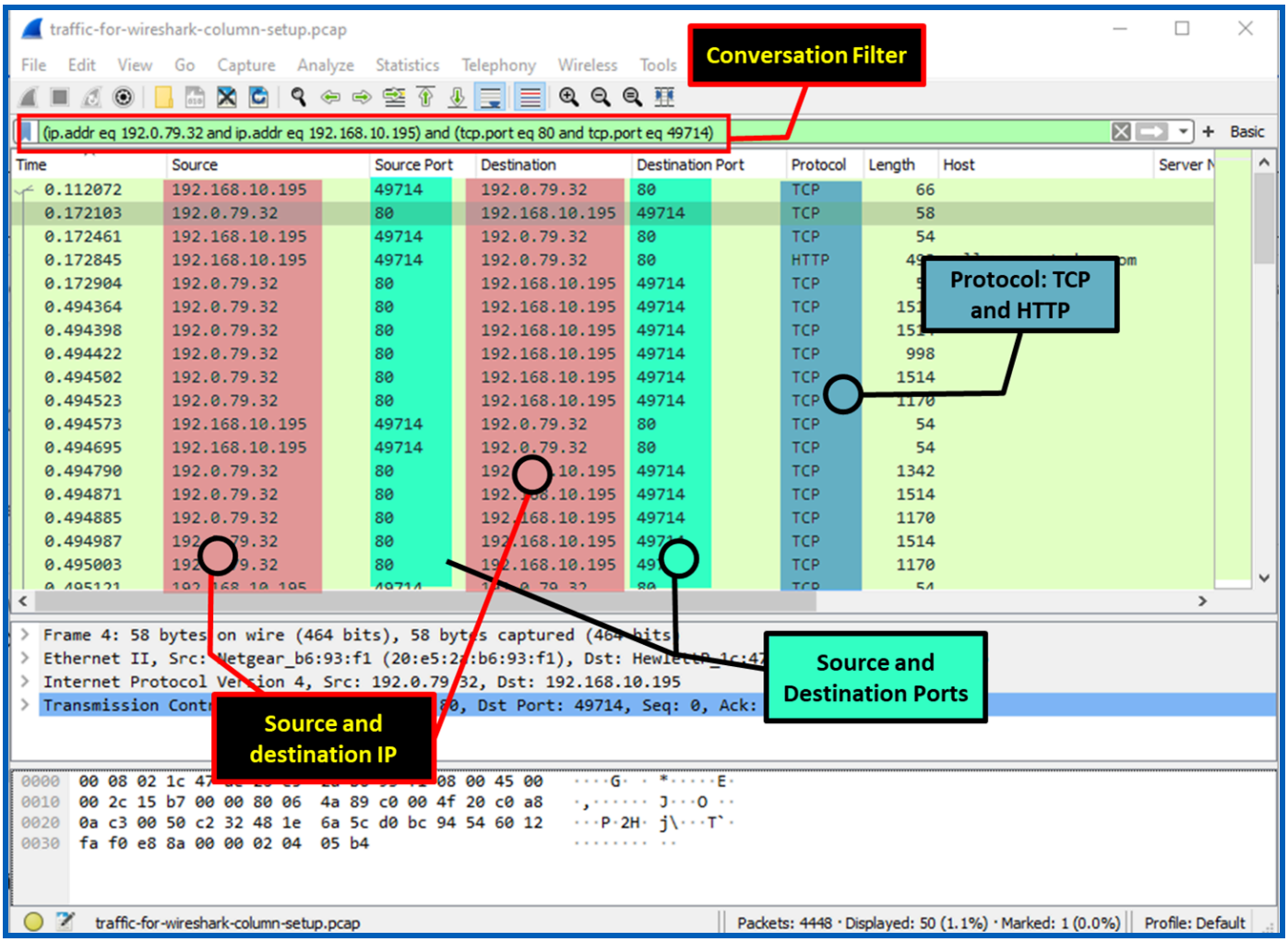
In the following example, we will filter traffic for IP addresses and TCP ports.

Typically, if we want to run a combination filter for two IP addresses and two port numbers, we use the following filter as an example:

(ip.addr eq 192.0.79.32 and ip.addr eq 192.168.10.195) and (tcp.port eq 80 and tcp.port eq 49714)

It is a very long filter to type or to remember, but a Conversation Filter will make your life easier. As illustrated in the image below, **right-click** on the **packet** you want to **filter** and select **Conversation filter -> TCP.**

Results:



This concludes this lab.

Please discuss the following questions with your instructor.

LAB SUBMISSION REQUIREMENTS

Please submit a pdf with the following:

1. **A screenshot of the snapshot taken once the lab is completed.**
2. **One to three screenshots demonstrating the configurations that you made during this lab.**
3. **Discussion questions with your answers.**

DISCUSSION QUESTIONS:

1. *What are the two types of filters in Wireshark?*

Capture Filters

Display Filters

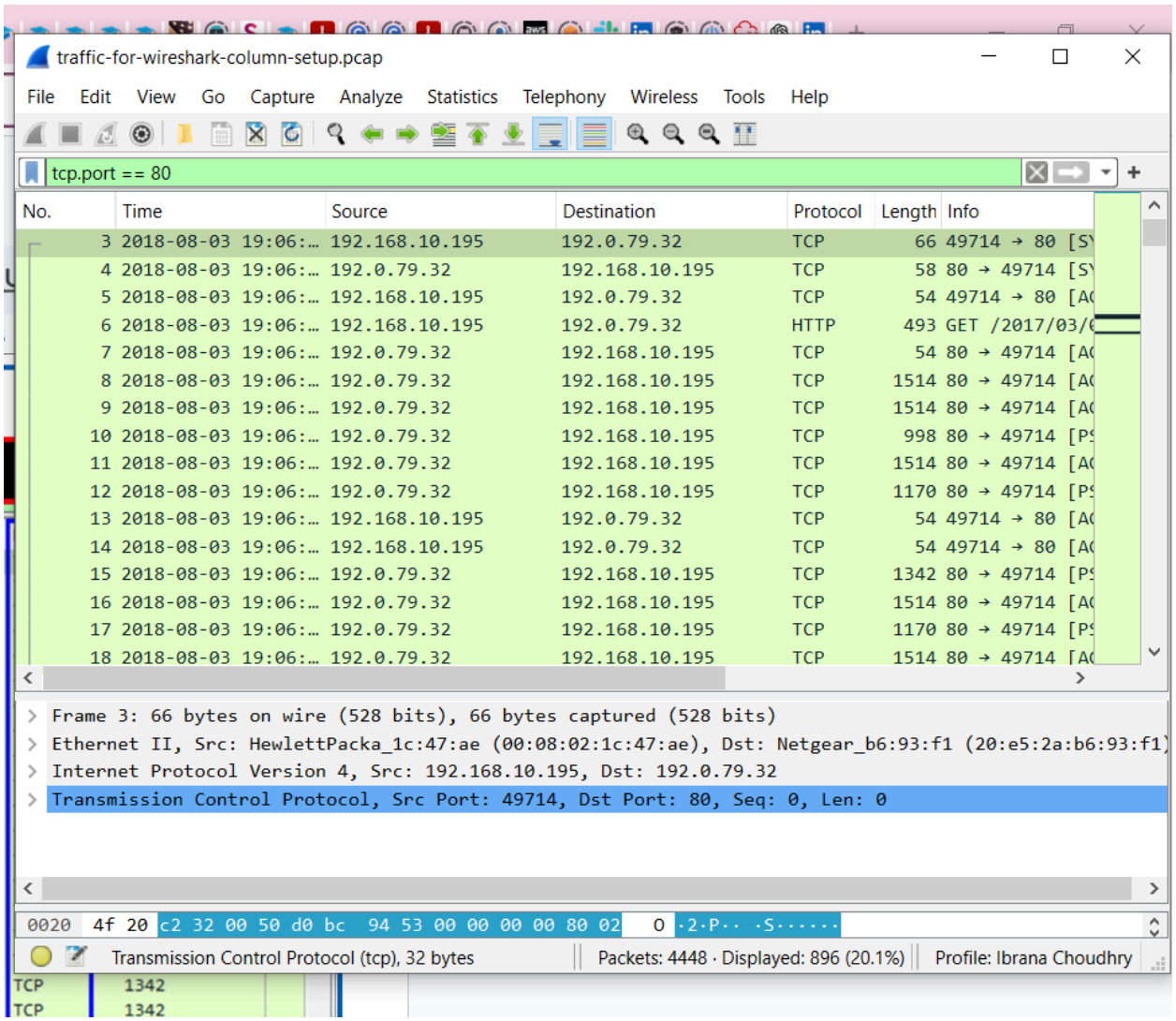
2. *How do you filter specific data in Wireshark??*

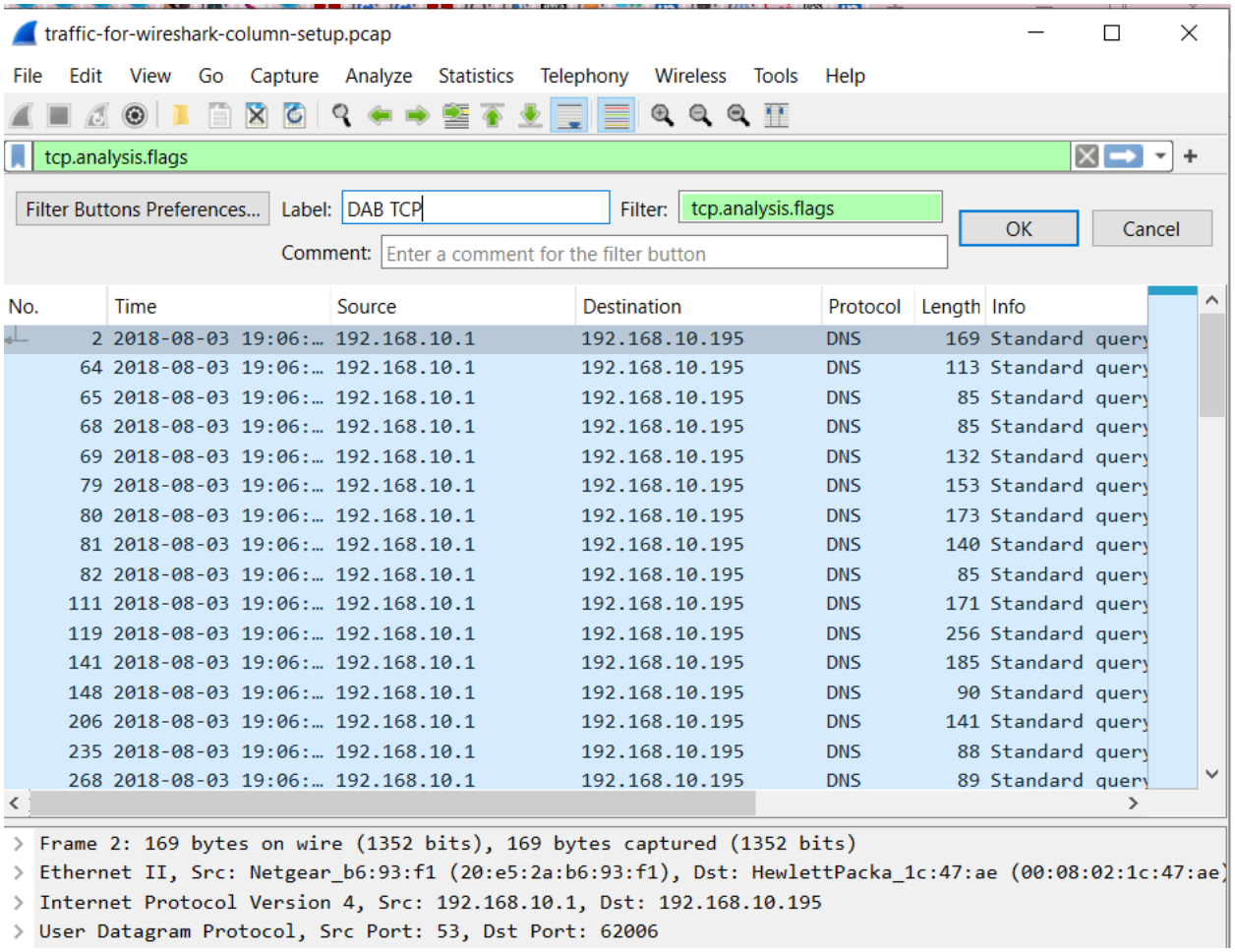
Filtering specific data in Wireshark involves using capture filters and display filters, depending on whether you want to filter packets during the capture process or after they have been captured.

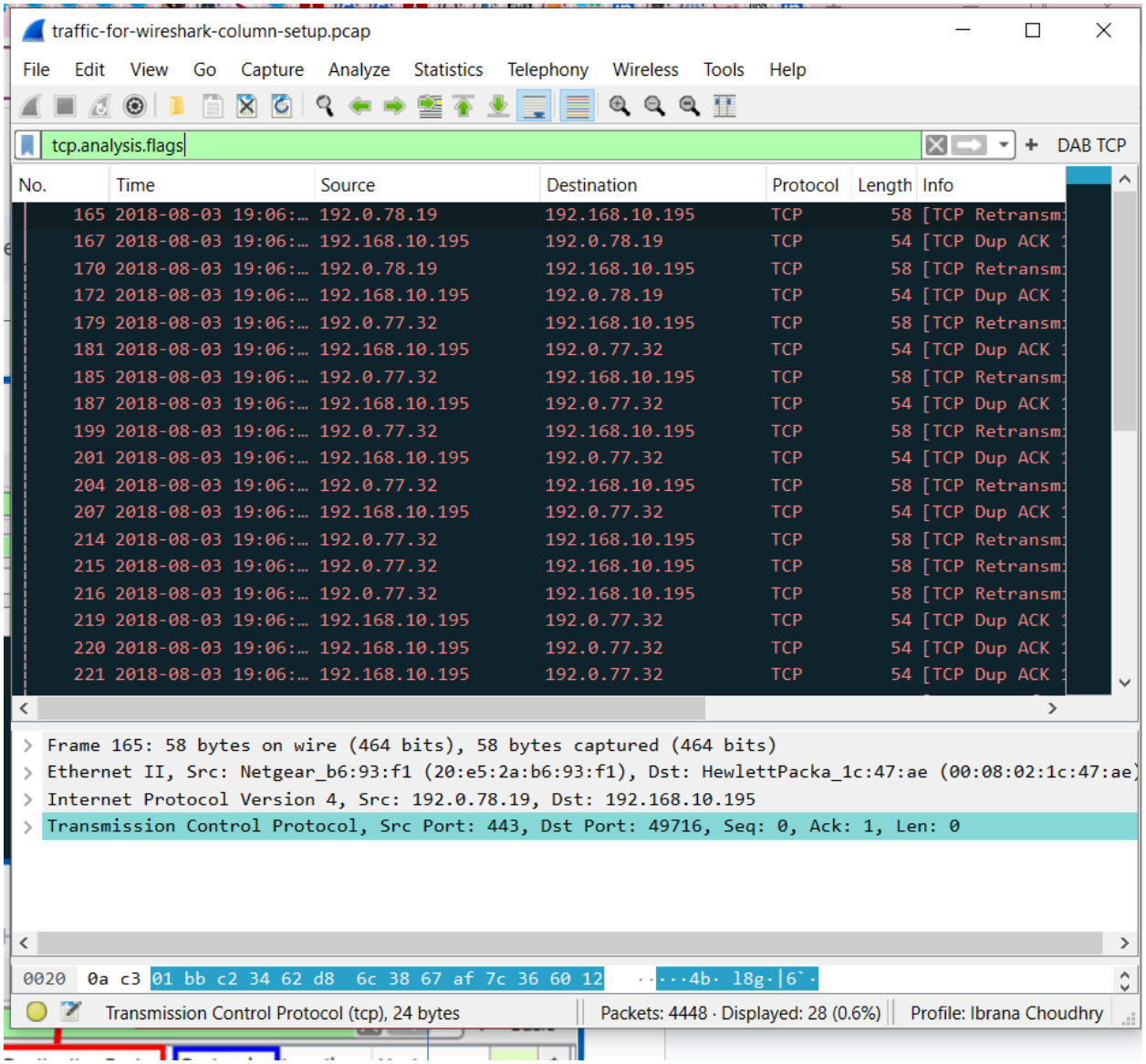
Display filters are used to refine and narrow down the view of the captured packets in Wireshark.

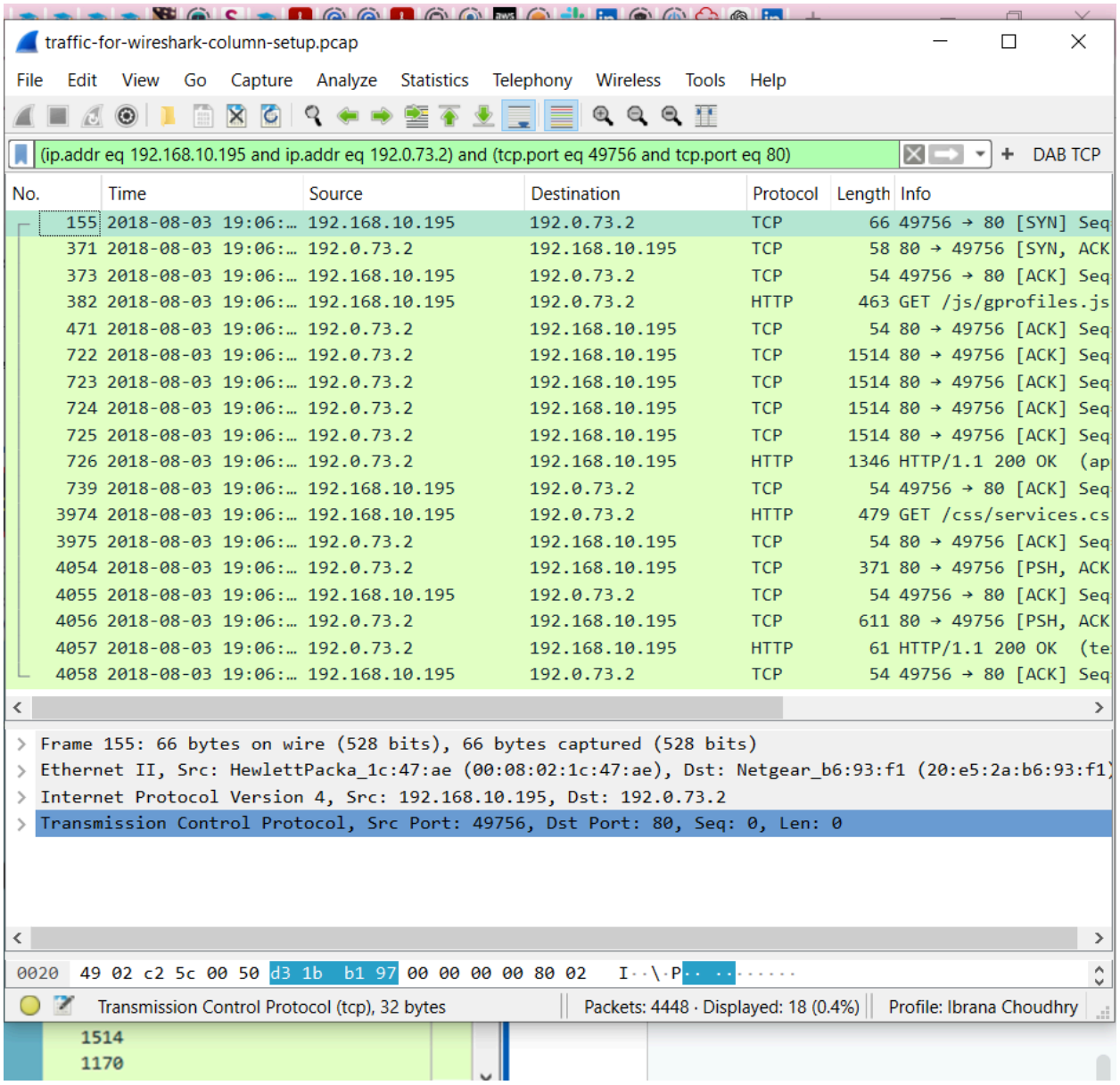
3. *What is the difference between a capture filter and a display filter?*

Capture filters only keep copies of packets that match the filter. Display filters are used when you've captured everything, but need to cut through the noise to analyze specific packets or flows. Capture filters and display filters are created using different syntaxes.


traffic-for-wireshark-column-setup.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.port == 80
No. Time Source Destination Protocol Length Info
3 2018-08-03 19:06:… 192.168.10.195 192.0.79.32 TCP 66 49714 → 80 [SY
4 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 58 80 → 49714 [SY
5 2018-08-03 19:06:… 192.168.10.195 192.0.79.32 TCP 54 49714 → 80 [AC
6 2018-08-03 19:06:… 192.168.10.195 192.0.79.32 HTTP 493 GET /2017/03/
7 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 54 80 → 49714 [AC
8 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 1514 80 → 49714 [AC
9 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 1514 80 → 49714 [AC
10 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 998 80 → 49714 [PS
11 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 1514 80 → 49714 [AC
12 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 1170 80 → 49714 [PS
13 2018-08-03 19:06:… 192.168.10.195 192.0.79.32 TCP 54 49714 → 80 [AC
14 2018-08-03 19:06:… 192.168.10.195 192.0.79.32 TCP 54 49714 → 80 [AC
15 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 1342 80 → 49714 [PS
16 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 1514 80 → 49714 [AC
17 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 1170 80 → 49714 [PS
18 2018-08-03 19:06:… 192.0.79.32 192.168.10.195 TCP 1514 80 → 49714 [AC
Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: HewlettPacka_1c:47:ae (00:08:02:1c:47:ae), Dst: Netgear_b6:93:f1 (20:e5:2a:b6:93:f1)
Internet Protocol Version 4, Src: 192.168.10.195, Dst: 192.0.79.32
Transmission Control Protocol, Src Port: 49714, Dst Port: 80, Seq: 0, Len: 0
0020 4f 20 c2 32 00 50 d0 bc 94 53 00 00 00 00 80 02 0 ·2·P·· ·S·····
Transmission Control Protocol (tcp), 32 bytes
Packets: 4448 · Displayed: 896 (20.1%)
Profile: Ibrana Choudhry
TCP 1342
TCP 1342


traffic-for-wireshark-column-setup.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.analysis.flags
Filter Buttons Preferences...
Label: DAB TCP
Filter: tcp.analysis.flags
OK
Cancel
Comment: Enter a comment for the filter button
No. Time Source Destination Protocol Length Info
2 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 169 Standard query
64 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 113 Standard query
65 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 85 Standard query
68 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 85 Standard query
69 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 132 Standard query
79 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 153 Standard query
80 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 173 Standard query
81 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 140 Standard query
82 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 85 Standard query
111 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 171 Standard query
119 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 256 Standard query
141 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 185 Standard query
148 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 90 Standard query
206 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 141 Standard query
235 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 88 Standard query
268 2018-08-03 19:06:… 192.168.10.1 192.168.10.195 DNS 89 Standard query
Frame 2: 169 bytes on wire (1352 bits), 169 bytes captured (1352 bits)
Ethernet II, Src: Netgear_b6:93:f1 (20:e5:2a:b6:93:f1), Dst: HewlettPacka_1c:47:ae (00:08:02:1c:47:ae)
Internet Protocol Version 4, Src: 192.168.10.1, Dst: 192.168.10.195
User Datagram Protocol, Src Port: 53, Dst Port: 62006


traffic-for-wireshark-column-setup.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.analysis.flags
DAB TCP
No. Time Source Destination Protocol Length Info
165 2018-08-03 19:06:… 192.0.78.19 192.168.10.195 TCP 58 [TCP Retransm
167 2018-08-03 19:06:… 192.168.10.195 192.0.78.19 TCP 54 [TCP Dup ACK
170 2018-08-03 19:06:… 192.0.78.19 192.168.10.195 TCP 58 [TCP Retransm
172 2018-08-03 19:06:… 192.168.10.195 192.0.78.19 TCP 54 [TCP Dup ACK
179 2018-08-03 19:06:… 192.0.77.32 192.168.10.195 TCP 58 [TCP Retransm
181 2018-08-03 19:06:… 192.168.10.195 192.0.77.32 TCP 54 [TCP Dup ACK
185 2018-08-03 19:06:… 192.0.77.32 192.168.10.195 TCP 58 [TCP Retransm
187 2018-08-03 19:06:… 192.168.10.195 192.0.77.32 TCP 54 [TCP Dup ACK
199 2018-08-03 19:06:… 192.0.77.32 192.168.10.195 TCP 58 [TCP Retransm
201 2018-08-03 19:06:… 192.168.10.195 192.0.77.32 TCP 54 [TCP Dup ACK
204 2018-08-03 19:06:… 192.0.77.32 192.168.10.195 TCP 58 [TCP Retransm
207 2018-08-03 19:06:… 192.168.10.195 192.0.77.32 TCP 54 [TCP Dup ACK
214 2018-08-03 19:06:… 192.0.77.32 192.168.10.195 TCP 58 [TCP Retransm
215 2018-08-03 19:06:… 192.0.77.32 192.168.10.195 TCP 58 [TCP Retransm
216 2018-08-03 19:06:… 192.0.77.32 192.168.10.195 TCP 58 [TCP Retransm
219 2018-08-03 19:06:… 192.168.10.195 192.0.77.32 TCP 54 [TCP Dup ACK
220 2018-08-03 19:06:… 192.168.10.195 192.0.77.32 TCP 54 [TCP Dup ACK
221 2018-08-03 19:06:… 192.168.10.195 192.0.77.32 TCP 54 [TCP Dup ACK
Frame 165: 58 bytes on wire (464 bits), 58 bytes captured (464 bits)
Ethernet II, Src: Netgear_b6:93:f1 (20:e5:2a:b6:93:f1), Dst: HewlettPacka_1c:47:ae (00:08:02:1c:47:ae)
Internet Protocol Version 4, Src: 192.0.78.19, Dst: 192.168.10.195
Transmission Control Protocol, Src Port: 443, Dst Port: 49716, Seq: 0, Ack: 1, Len: 0
0020 0a c3 01 bb c2 34 62 d8 6c 38 67 af 7c 36 60 12
Transmission Control Protocol (tcp), 24 bytes
Packets: 4448 · Displayed: 28 (0.6%)
Profile: Ibrana Choudhry


traffic-for-wireshark-column-setup.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
(ip.addr eq 192.168.10.195 and ip.addr eq 192.0.73.2) and (tcp.port eq 49756 and tcp.port eq 80)
DAB TCP
No. Time Source Destination Protocol Length Info
155 2018-08-03 19:06:… 192.168.10.195 192.0.73.2 TCP 66 49756 → 80 [SYN] Seq
371 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 TCP 58 80 → 49756 [SYN, ACK
373 2018-08-03 19:06:… 192.168.10.195 192.0.73.2 TCP 54 49756 → 80 [ACK] Seq
382 2018-08-03 19:06:… 192.168.10.195 192.0.73.2 HTTP 463 GET /js/gprofiles.js
471 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 TCP 54 80 → 49756 [ACK] Seq
722 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 TCP 1514 80 → 49756 [ACK] Seq
723 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 TCP 1514 80 → 49756 [ACK] Seq
724 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 TCP 1514 80 → 49756 [ACK] Seq
725 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 TCP 1514 80 → 49756 [ACK] Seq
726 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 HTTP 1346 HTTP/1.1 200 OK (ap
739 2018-08-03 19:06:… 192.168.10.195 192.0.73.2 TCP 54 49756 → 80 [ACK] Seq
3974 2018-08-03 19:06:… 192.168.10.195 192.0.73.2 HTTP 479 GET /css/services.cs
3975 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 TCP 54 80 → 49756 [ACK] Seq
4054 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 TCP 371 80 → 49756 [PSH, ACK
4055 2018-08-03 19:06:… 192.168.10.195 192.0.73.2 TCP 54 49756 → 80 [ACK] Seq
4056 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 TCP 611 80 → 49756 [PSH, ACK
4057 2018-08-03 19:06:… 192.0.73.2 192.168.10.195 HTTP 61 HTTP/1.1 200 OK (te
4058 2018-08-03 19:06:… 192.168.10.195 192.0.73.2 TCP 54 49756 → 80 [ACK] Seq
Frame 155: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: HewlettPacka_1c:47:ae (00:08:02:1c:47:ae), Dst: Netgear_b6:93:f1 (20:e5:2a:b6:93:f1)
Internet Protocol Version 4, Src: 192.168.10.195, Dst: 192.0.73.2
Transmission Control Protocol, Src Port: 49756, Dst Port: 80, Seq: 0, Len: 0
0020 49 02 c2 5c 00 50 d3 1b b1 97 00 00 00 00 80 02
Transmission Control Protocol (tcp), 32 bytes
Packets: 4448 · Displayed: 18 (0.4%)
Profile: Ibrana Choudhry
1514
1170