TRY HACK ME: Wireshark: Packet Operations Write-Up

Task 1 Introduction-

In the first room, we covered the basics of the Wireshark by focusing on how it operates and how to use it to investigate traffic captures. In this room, we will cover advanced features of the Wireshark by focusing on packet-level details with Wireshark statistics, filters, operators and functions.

Note: A VM is attached to this room. You don't need SSH or RDP; the room provides a "Split View" feature. Access to the machine will be provided in-browser and will deploy in Split View mode in your browser. If you don't see it, use the blue Show Split View button at the top right of this room page to show it. DO NOT directly interact with any domains and IP addresses in this room. The domains and IP addresses are included for reference reasons only

Answer to the questions of this section-

No Answer needed

Task 2 Statistics | Summary-

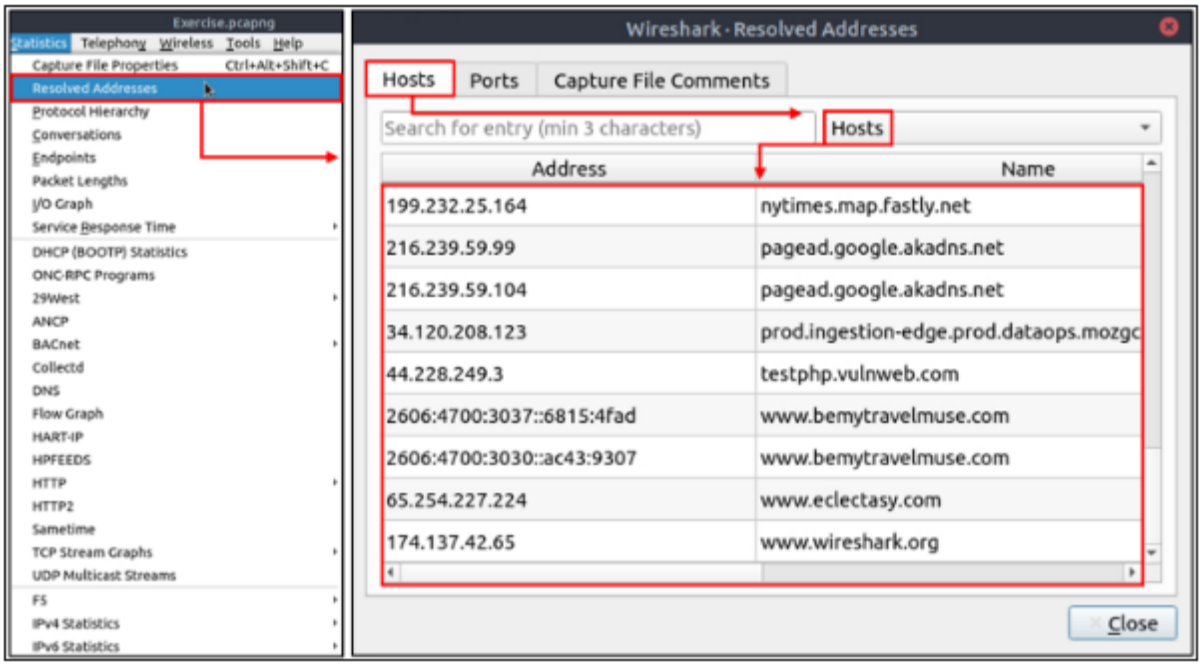
Statistics

This menu provides multiple statistics options ready to investigate to help users see the big picture in terms of the scope of the traffic, available protocols, endpoints and conversations, and some protocol-specific details like DHCP, DNS and HTTP/2. For a security analyst, it is crucial to know how to utilise the statical information. This section provides a quick summary of the processed pcap, which will help analysts create a hypothesis for an investigation. You can use the "Statistics" menu to view all available options. Now start the given VM, open the Wireshark, load the "Exercise.pcapng" file and go through the walkthrough.

Resolved Addresses

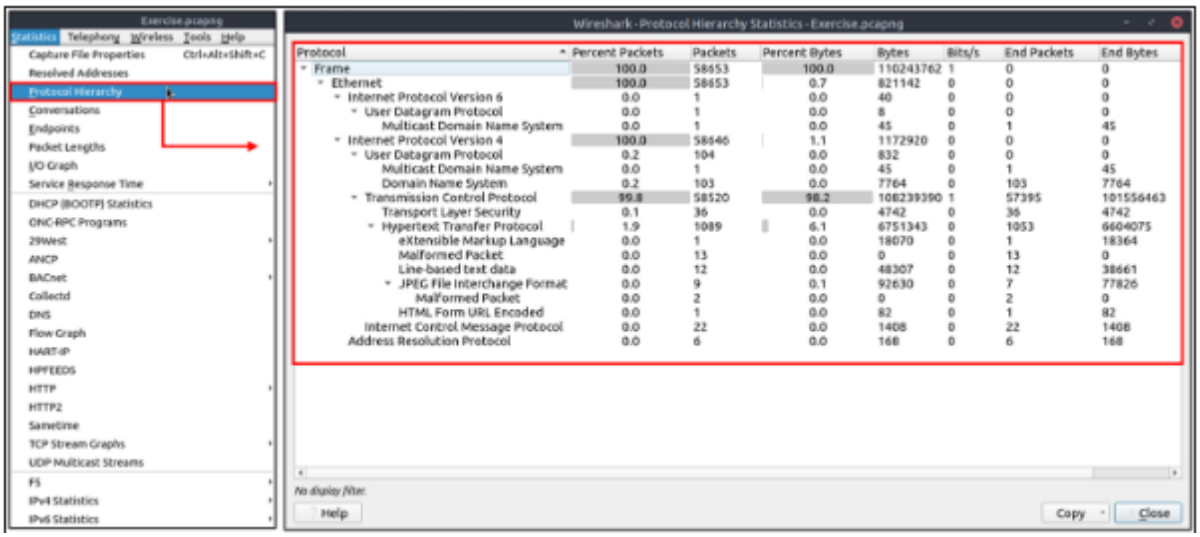
This option helps analysts identify IP addresses and DNS names available in the capture file by providing the list of the resolved addresses and their hostnames. Note that the hostname information is taken from DNS answers in the capture file. Analysts can quickly identify the accessed resources by using this menu. Thus they can spot accessed resources and evaluate them according

to the event of interest. You can use the "Statistics --> Resolved Addresses" menu to view all resolved addresses by Wireshark.



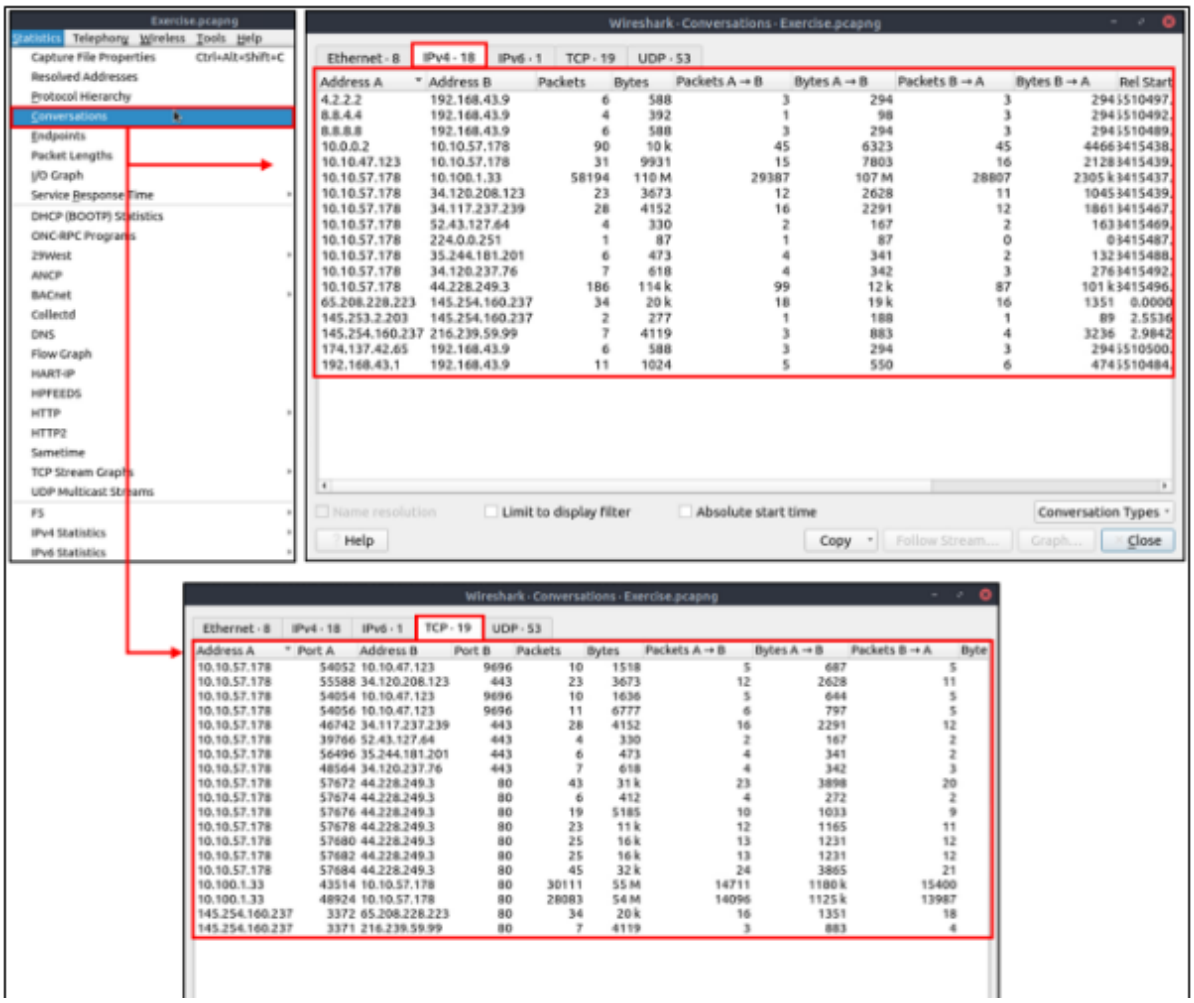
Protocol Hierarchy

This option breaks down all available protocols from the capture file and helps analysts view the protocols in a tree view based on packet counters and percentages. Thus analysts can view the overall usage of the ports and services and focus on the event of interest. The golden rule mentioned in the previous room is valid in this section; you can right-click and filter the event of interest. You can use the "Statistics --> Protocol Hierarchy" menu to view this info.



Conversations

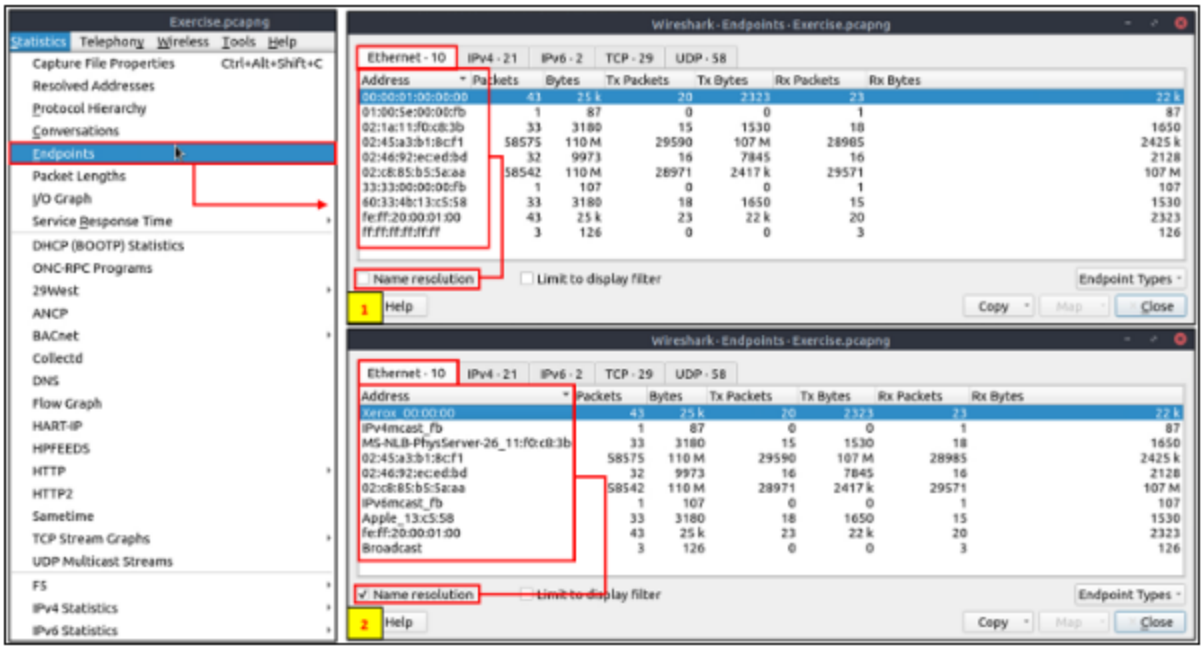
Conversation represents traffic between two specific endpoints. This option provides the list of the conversations in five base formats; ethernet, IPv4, IPv6, TCP and UDP. Thus analysts can identify all conversations and contact endpoints for the event of interest. You can use the "Statistic --> Conversations" menu to view this info.



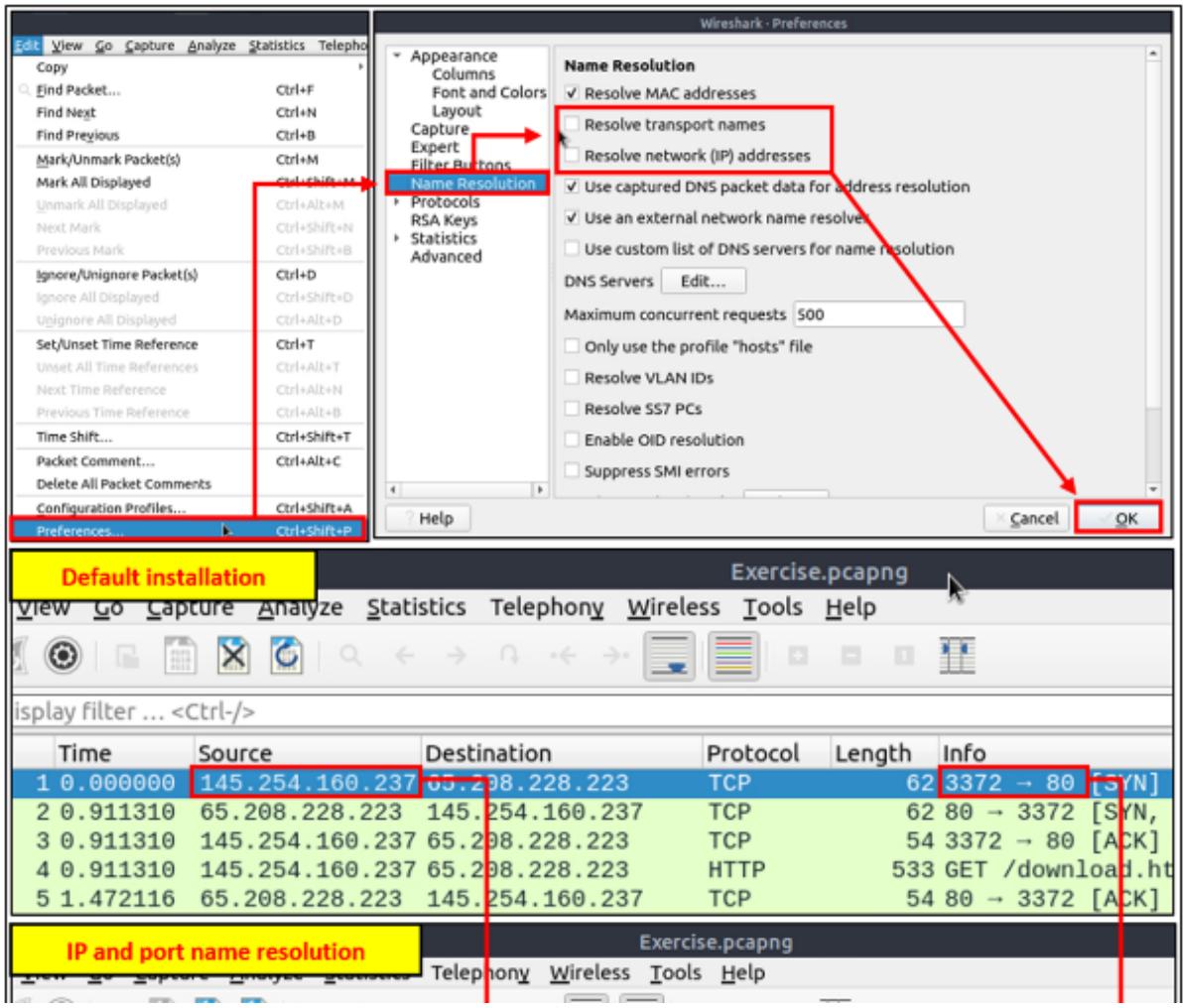
Endpoints

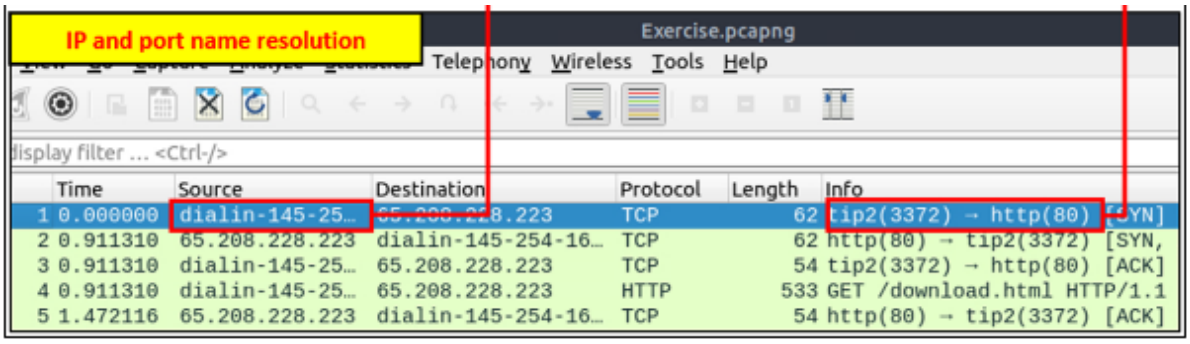
The endpoints option is similar to the conversations option. The only difference is that this option provides unique information for a single information field (Ethernet, IPv4, IPv6, TCP and UDP). Thus analysts can identify the unique endpoints in the capture file and use it for the event of interest. You can use the "Statistics --> Endpoints" menu to view this info.

Wireshark also supports resolving MAC addresses to human-readable format using the manufacturer name assigned by IEEE. Note that this conversion is done through the first three bytes of the MAC address and only works for the known manufacturers. When you review the ethernet endpoints, you can activate this option with the "Name resolution" button in the lower-left corner of the endpoints window.

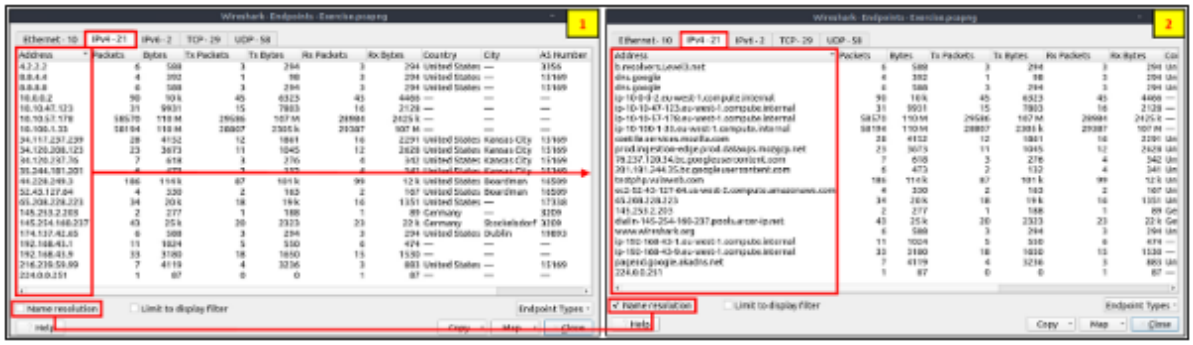


Name resolution is not limited only to MAC addresses. Wireshark provides IP and port name resolution options as well. However, these options are not enabled by default. If you want to use these functionalities, you need to activate them through the "Edit --> Preferences --> Name Resolution" menu. Once you enable IP and port name resolution, you will see the resolved IP address and port names in the packet list pane and also will be able to view resolved names in the "Conversations" and "Endpoints" menus as well.

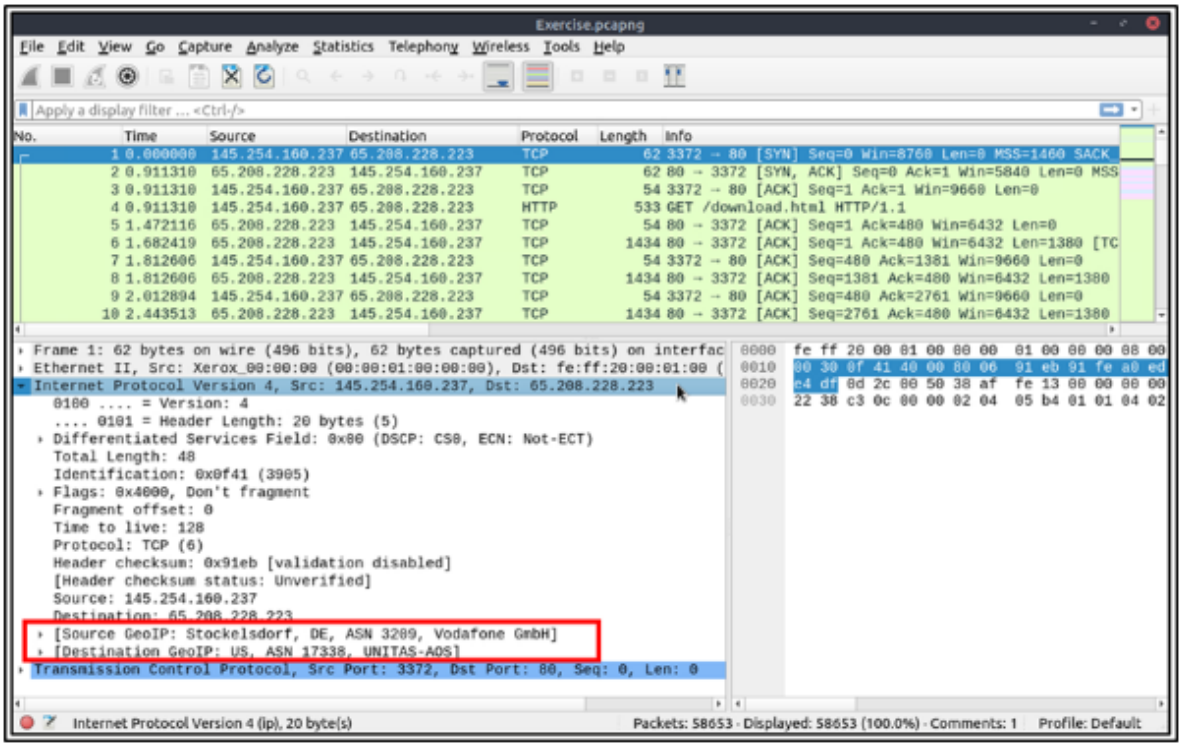




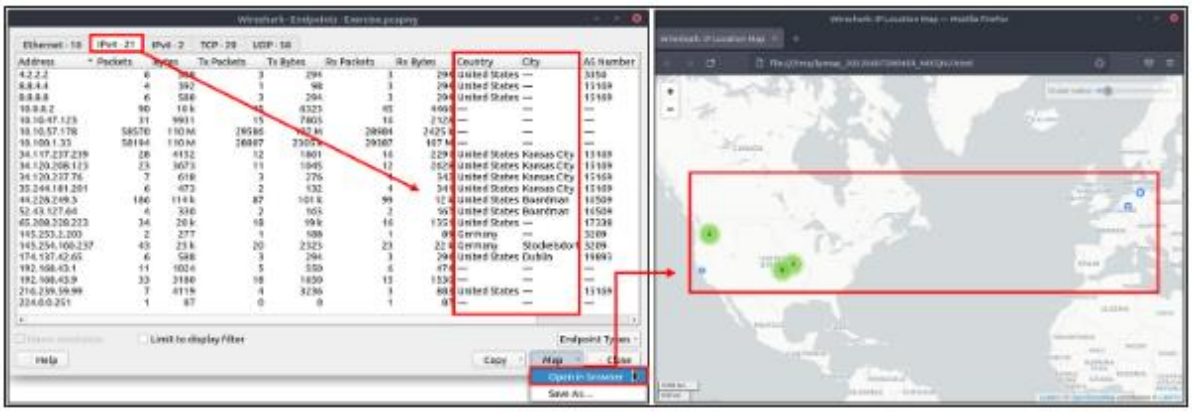
Endpoint menu view with name resolution:



Besides name resolution, Wireshark also provides an IP geolocation mapping that helps analysts identify the map's source and destination addresses. But this feature is not activated by default and needs supplementary data like the GeoIP database. Currently, Wireshark supports MaxMind databases, and the latest versions of the Wireshark come configured MaxMind DB resolver. However, you still need MaxMind DB files and provide the database path to Wireshark by using the "Edit --> Preferences --> Name Resolution --> MaxMind database directories" menu. Once you download and indicate the path, Wireshark will automatically provide GeoIP information under the IP protocol details for the matched IP addresses.



Endpoints and GeoIP view.



Note: You need an active internet connection to view the GeoIP map. The lab machine doesn't have an active internet connection!

Answer to the questions of this section-

Investigate the resolved addresses. What is the IP address of the hostname starts with "bbc"?

199.232.24.81 Correct Answer Hint

What is the number of IPv4 conversations?

435 Correct Answer Hint

How many bytes (k) were transferred from the "Micro-St" MAC address?

7474 Correct Answer Hint

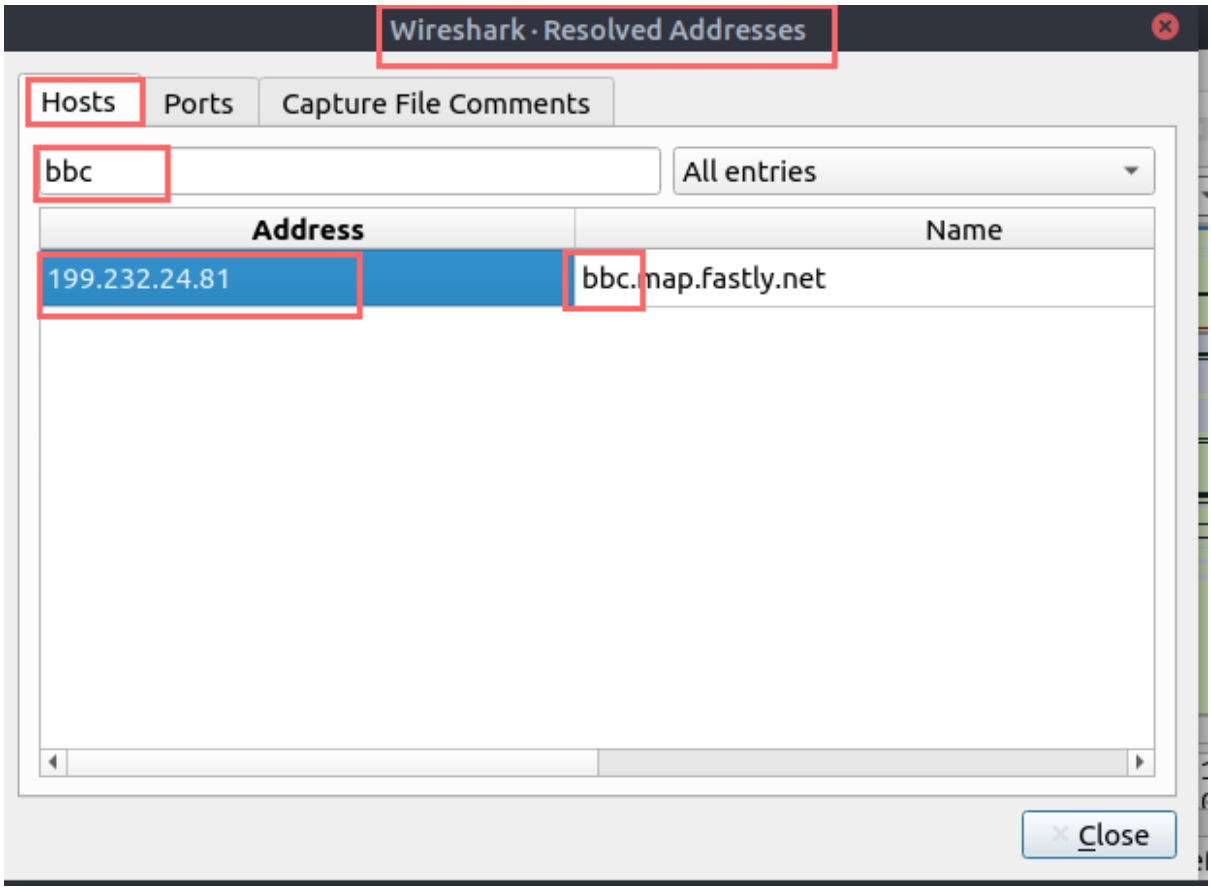
What is the number of IP addresses linked with "Kansas City"?

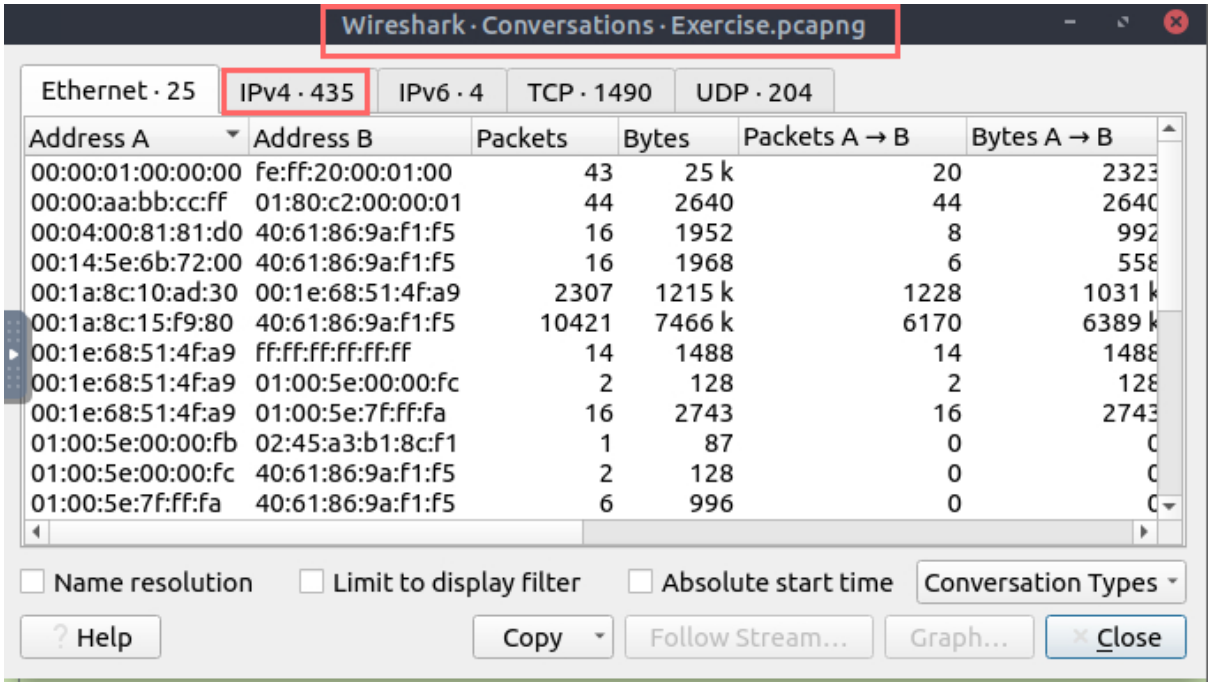
4 Correct Answer Hint

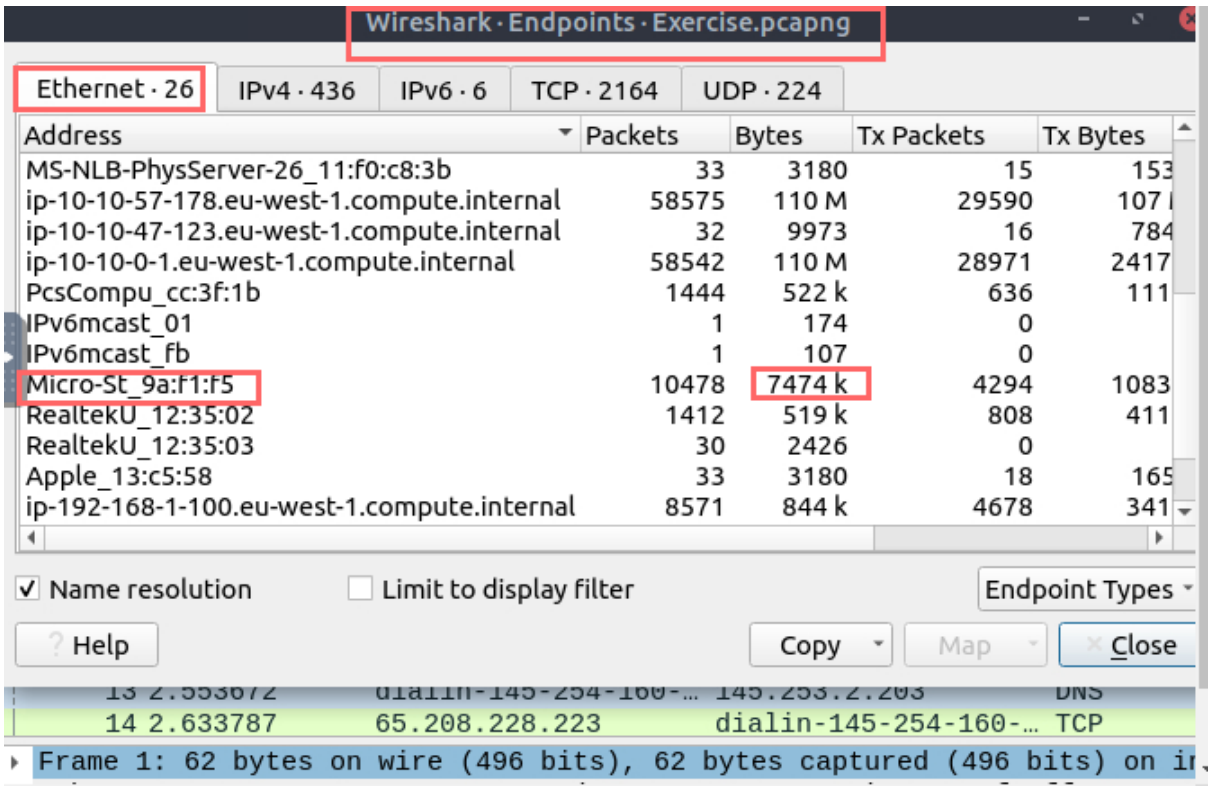
Which IP address is linked with "Blicnet" AS Organisation?

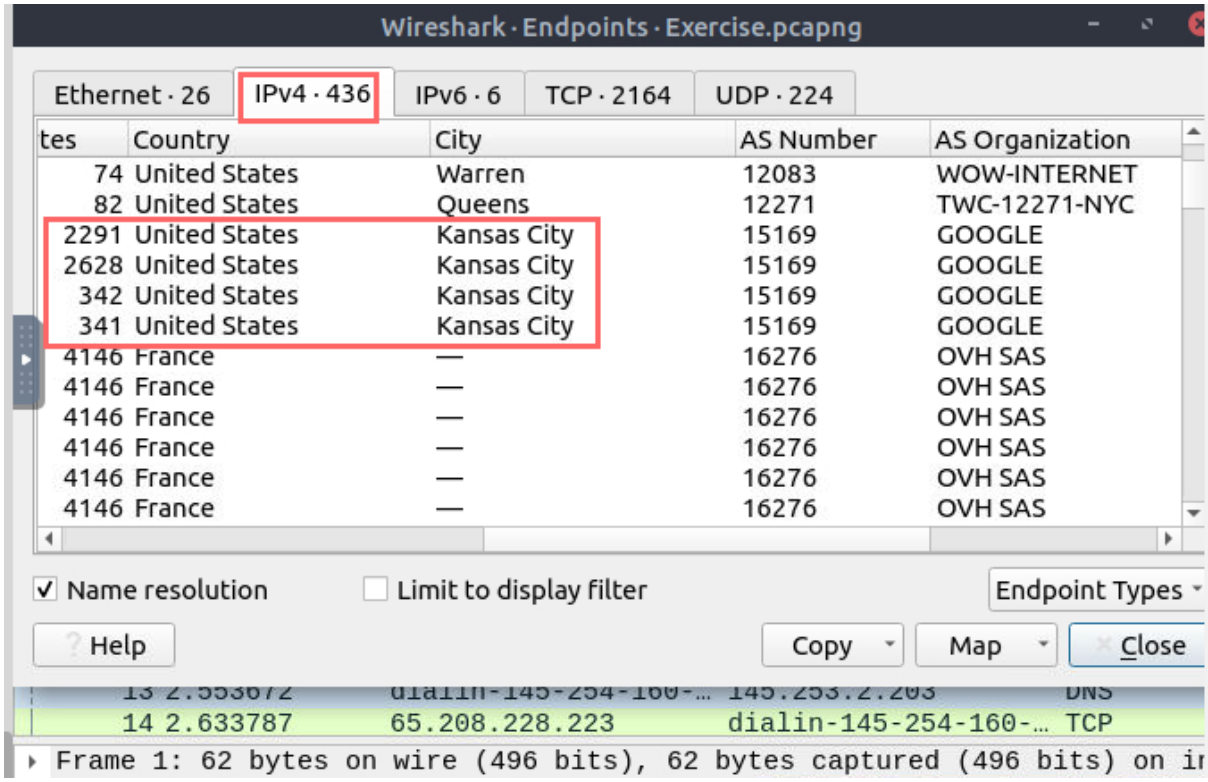
188.246.82.7 Correct Answer Hint

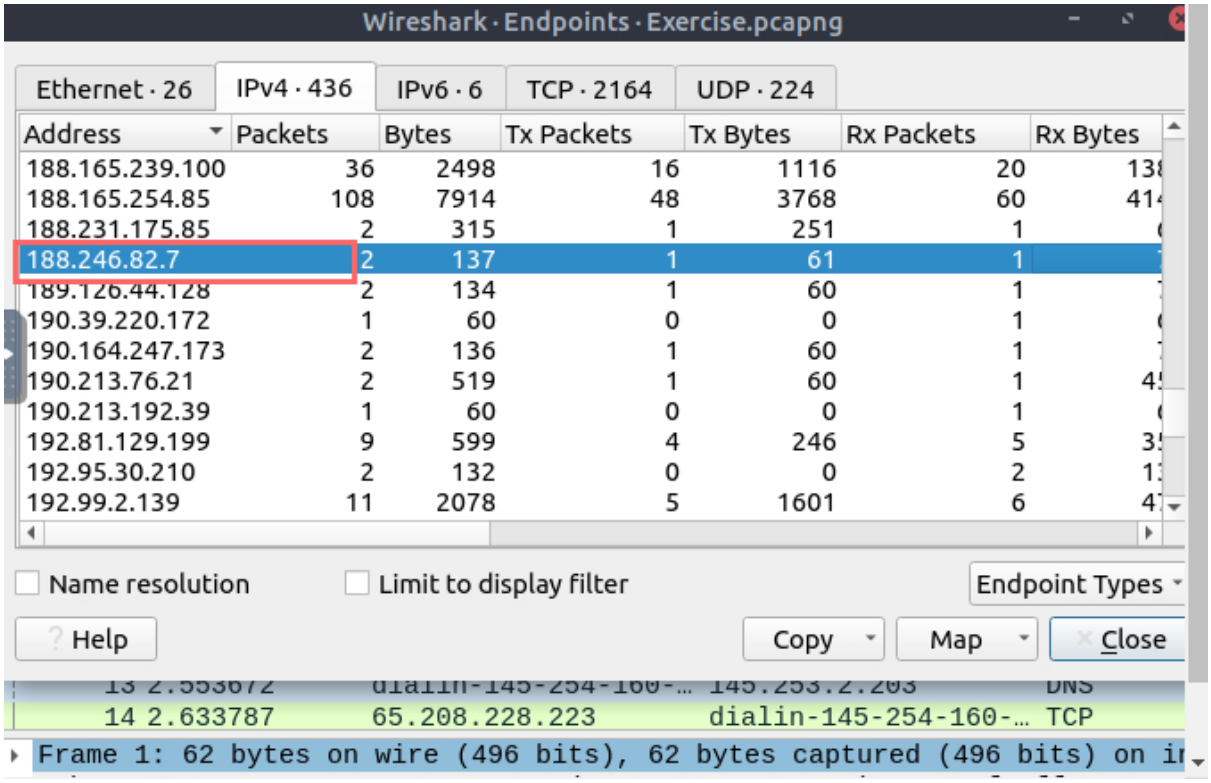
Answers-




Wireshark · Conversations · Exercise.pcapng
Ethernet · 25
IPv4 · 435
IPv6 · 4
TCP · 1490
UDP · 204
Address A
Address B
Packets
Bytes
Packets A → B
Bytes A → B
00:00:01:00:00:00 fe:ff:20:00:01:00 43 25 k 20 2323
00:00:aa:bb:cc:ff 01:80:c2:00:00:01 44 2640 44 2640
00:04:00:81:81:d0 40:61:86:9a:f1:f5 16 1952 8 992
00:14:5e:6b:72:00 40:61:86:9a:f1:f5 16 1968 6 558
00:1a:8c:10:ad:30 00:1e:68:51:4f:a9 2307 1215 k 1228 1031 k
00:1a:8c:15:f9:80 40:61:86:9a:f1:f5 10421 7466 k 6170 6389 k
00:1e:68:51:4f:a9 ff:ff:ff:ff:ff:ff 14 1488 14 1488
00:1e:68:51:4f:a9 01:00:5e:00:00:fc 2 128 2 128
00:1e:68:51:4f:a9 01:00:5e:7f:ff:fa 16 2743 16 2743
01:00:5e:00:00:fb 02:45:a3:b1:8c:f1 1 87 0 0
01:00:5e:00:00:fc 40:61:86:9a:f1:f5 2 128 0 0
01:00:5e:7f:ff:fa 40:61:86:9a:f1:f5 6 996 0 0
Name resolution
Limit to display filter
Absolute start time
Conversation Types
Help
Copy
Follow Stream...
Graph...
Close


Wireshark · Endpoints · Exercise.pcapng
Ethernet · 26
IPv4 · 436
IPv6 · 6
TCP · 2164
UDP · 224
Address
Packets
Bytes
Tx Packets
Tx Bytes
MS-NLB-PhysServer-26_11:f0:c8:3b 33 3180 15 153
ip-10-10-57-178.eu-west-1.compute.internal 58575 110 M 29590 107
ip-10-10-47-123.eu-west-1.compute.internal 32 9973 16 784
ip-10-10-0-1.eu-west-1.compute.internal 58542 110 M 28971 2417
PcsCompu_cc:3f:1b 1444 522 k 636 111
IPv6mcast_01 1 174 0
IPv6mcast_fb 1 107 0
Micro-St_9a:f1:f5 10478 7474 k 4294 1083
RealtekU_12:35:02 1412 519 k 808 411
RealtekU_12:35:03 30 2426 0
Apple_13:c5:58 33 3180 18 165
ip-192-168-1-100.eu-west-1.compute.internal 8571 844 k 4678 341
Name resolution
Limit to display filter
Endpoint Types
Help
Copy
Map
Close
14 2.633787 65.208.228.223 dialin-145-254-160-… TCP
Frame 1: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on

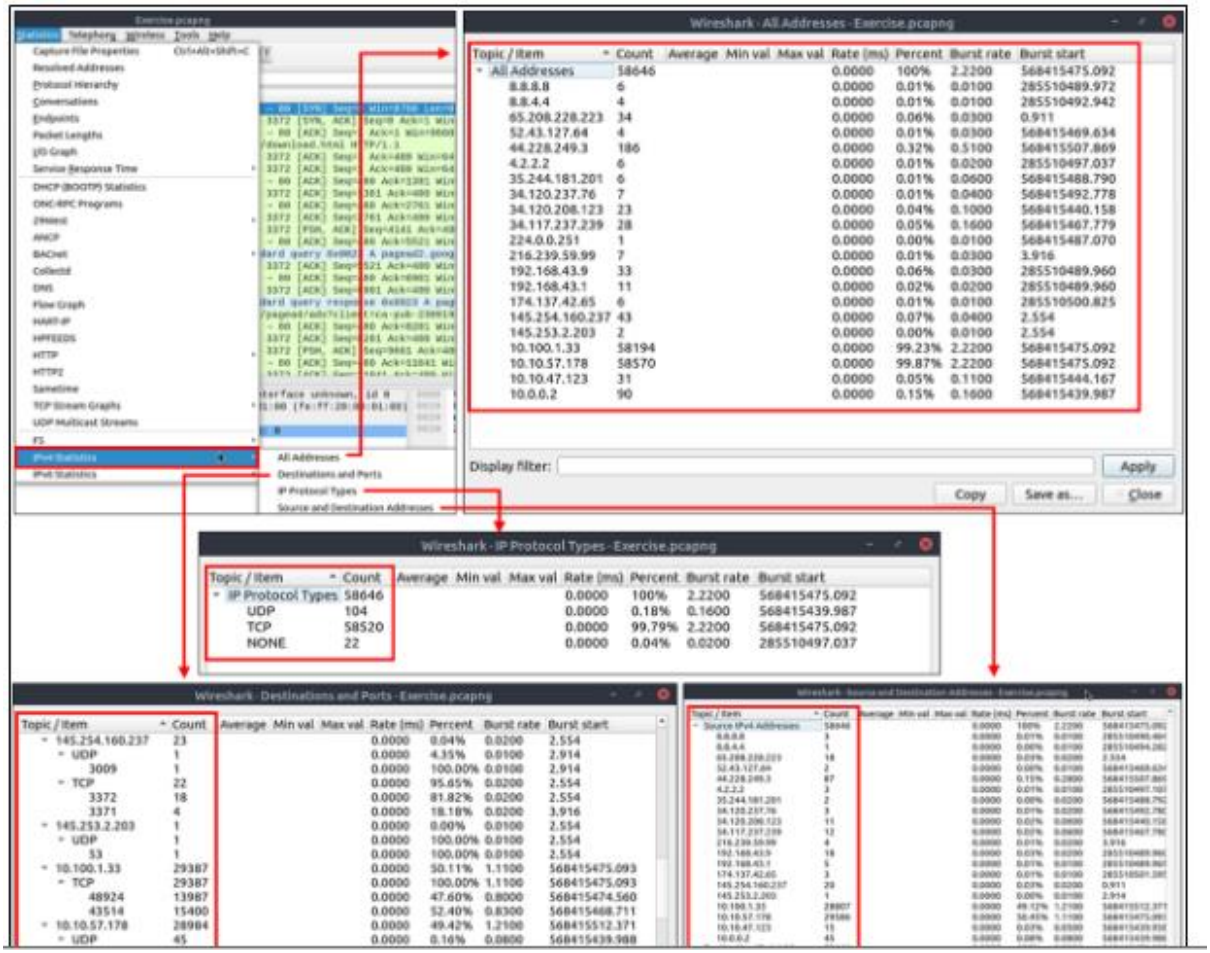


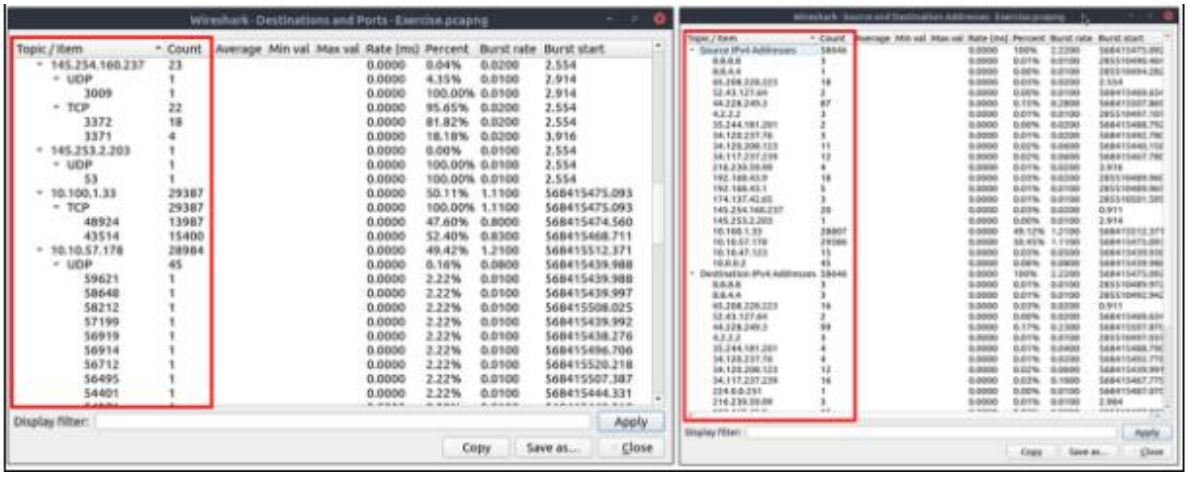


Task 3 Statistics | Protocol Details-

IPv4 and IPv6

Up to here, almost all options provided information that contained both versions of the IP addresses. The statistics menu has two options for narrowing the statistics on packets containing a specific IP version. Thus, analysts can identify and list all events linked to specific IP versions in a single window and use it for the event of interest. You can use the "Statistics --> IPvX Statistics" menu to view this info.

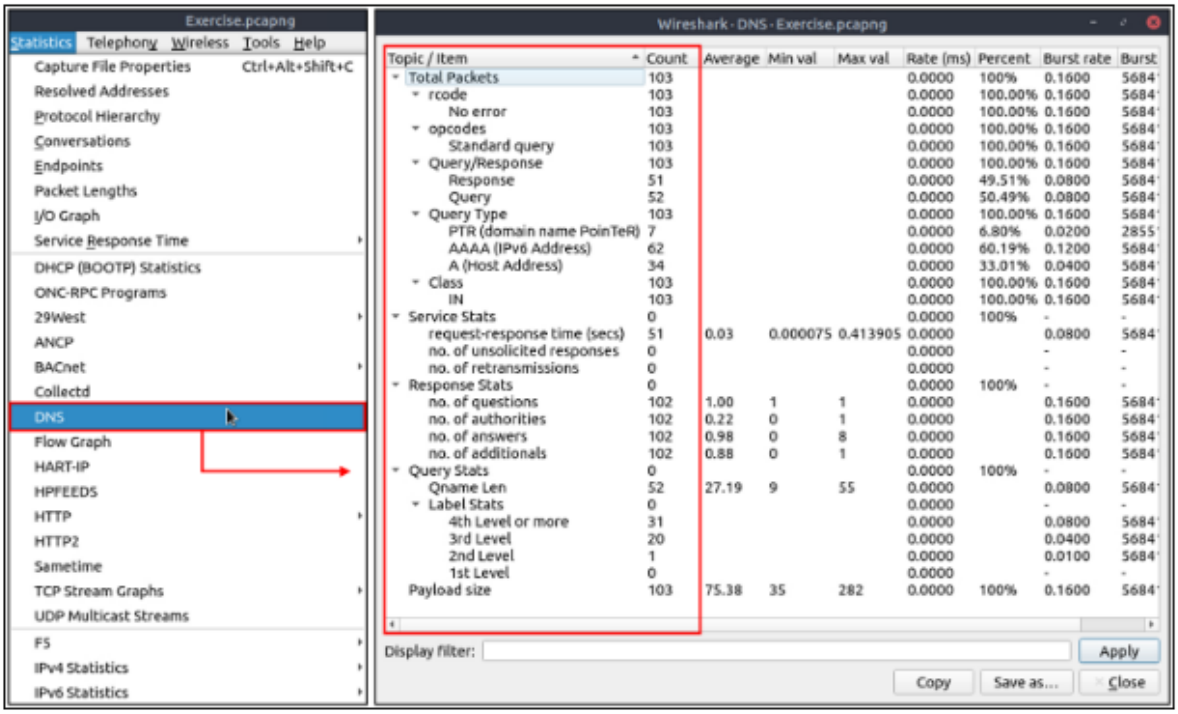




DNS

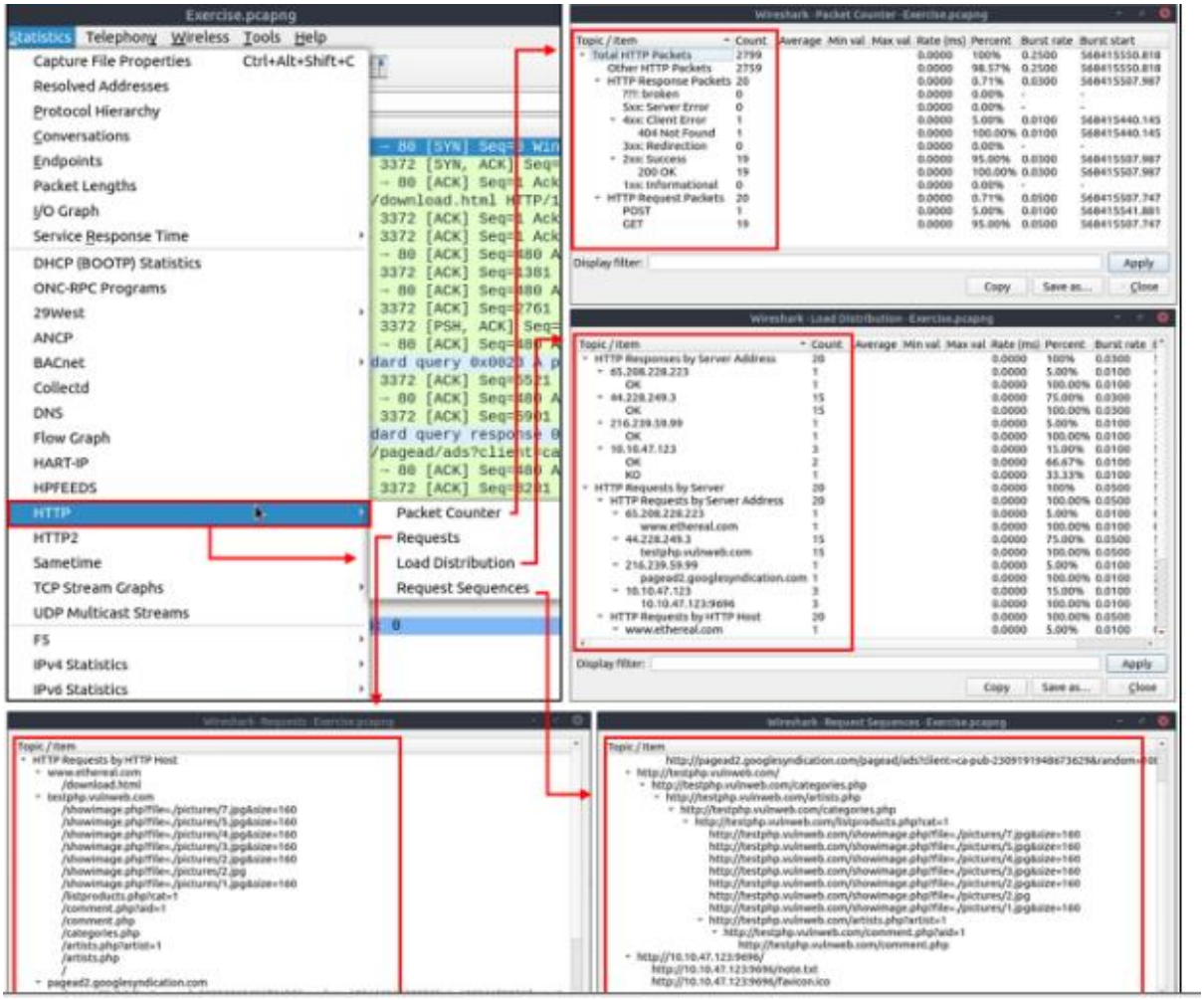
This option breaks down all DNS packets from the capture file and helps analysts view the findings in a tree view based on packet counters and percentages of the DNS protocol. Thus analysts can view

the DNS service's overall usage, including rcode, opcode, class, query type, service and query stats and use it for the event of interest. You can use the "Statistics --> DNS" menu to view this info.



HTTP

This option breaks down all HTTP packets from the capture file and helps analysts view the findings in a tree view based on packet counters and percentages of the HTTP protocol. Thus analysts can view the HTTP service's overall usage, including request and response codes and the original requests. You can use the "Statistics --> HTTP" menu to view this info.



Answer to the questions of this section-

What is the most used IPv4 destination address?

10.100.1.33 Correct Answer Hint

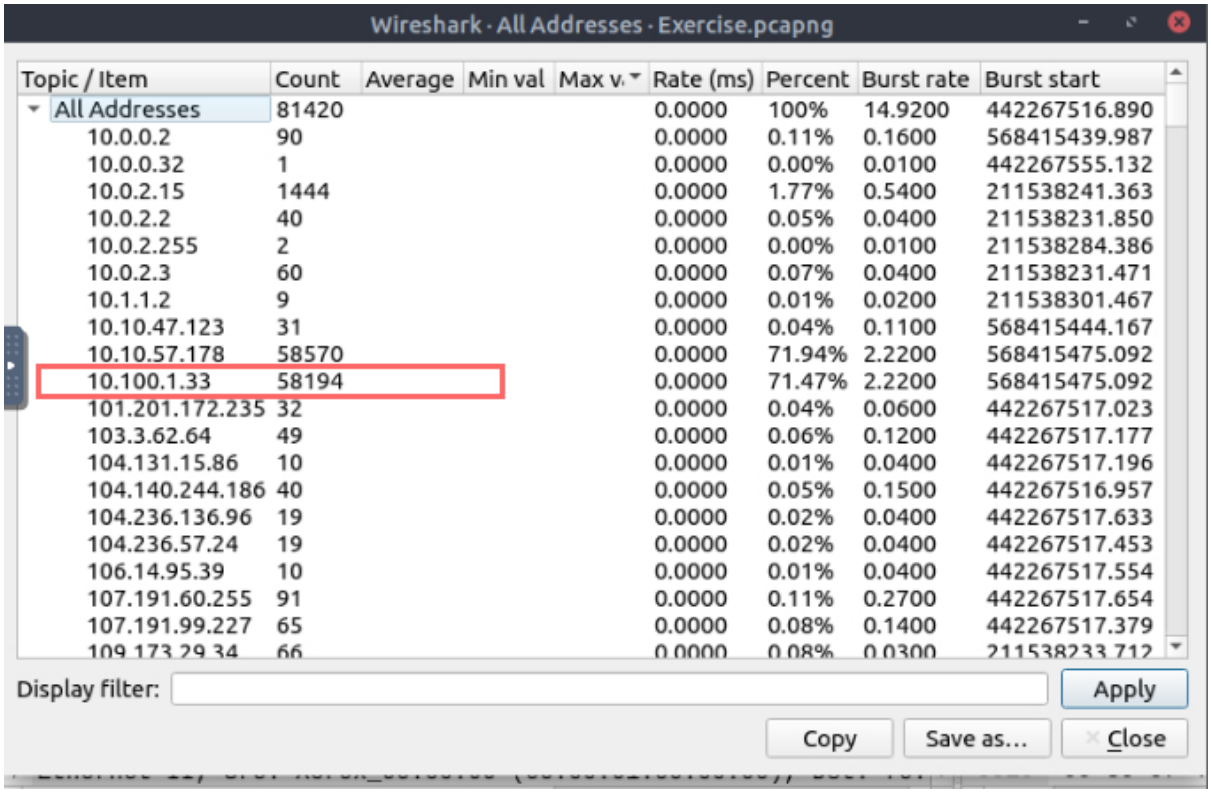
What is the max service request-response time of the DNS packets?

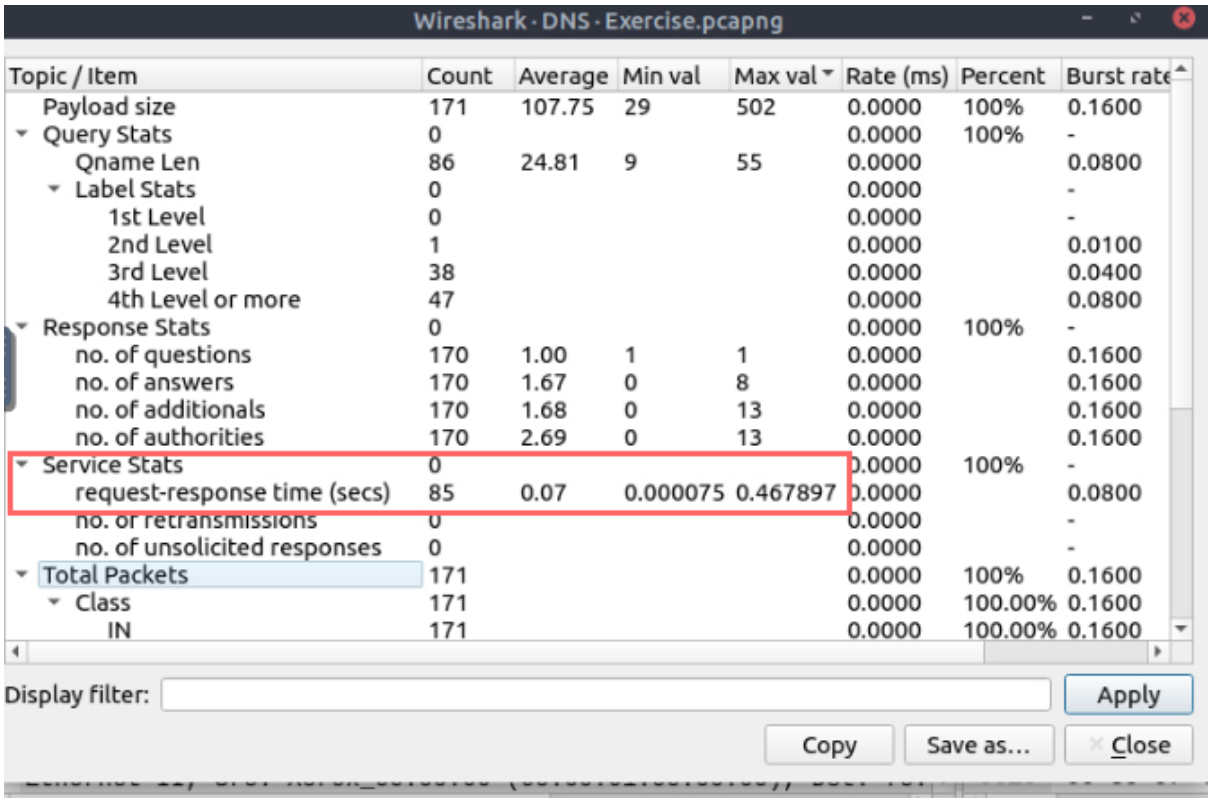
0.467897 Correct Answer Hint

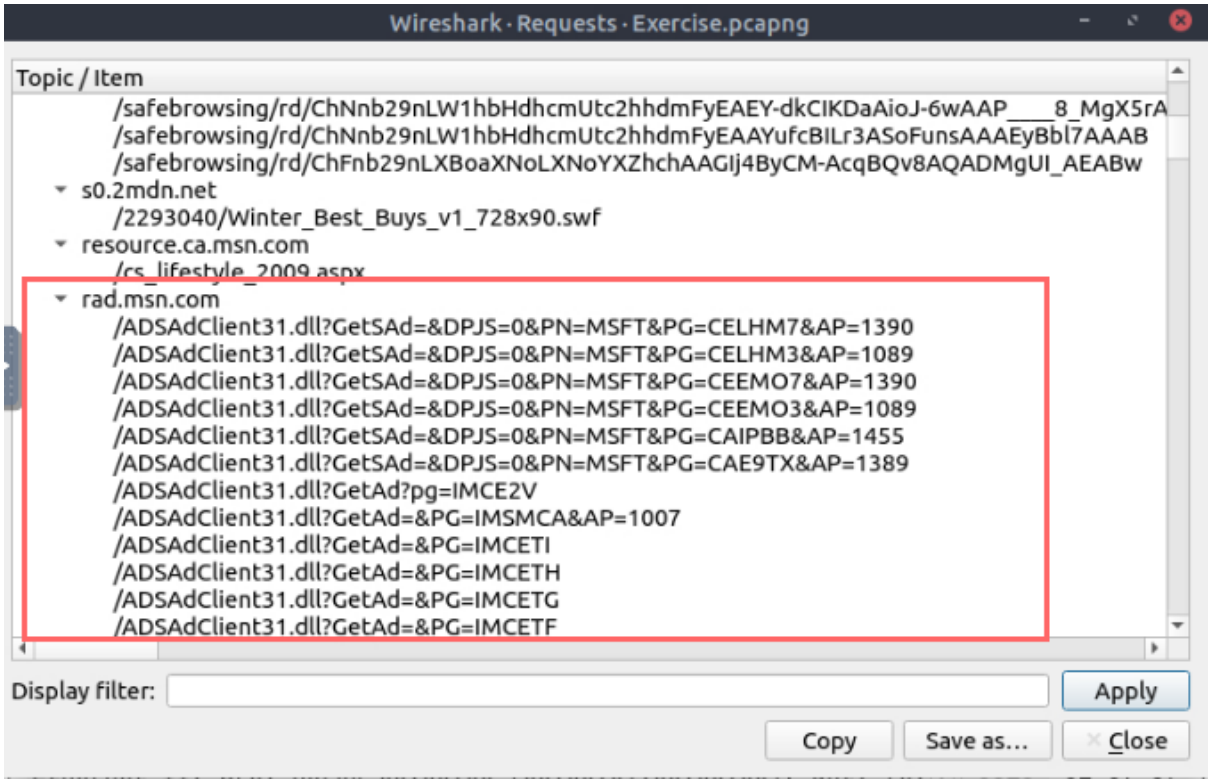
What is the number of HTTP Requests accomplished by "rad[.]msn[.]com?

39 Correct Answer Hint

Answers-


Wireshark · All Addresses · Exercise.pcapng
Topic / Item Count Average Min val Max v. Rate (ms) Percent Burst rate Burst start
All Addresses 81420 0.0000 100% 14.9200 442267516.890
10.0.0.2 90 0.0000 0.11% 0.1600 568415439.987
10.0.0.32 1 0.0000 0.00% 0.0100 442267555.132
10.0.2.15 1444 0.0000 1.77% 0.5400 211538241.363
10.0.2.2 40 0.0000 0.05% 0.0400 211538231.850
10.0.2.255 2 0.0000 0.00% 0.0100 211538284.386
10.0.2.3 60 0.0000 0.07% 0.0400 211538231.471
10.1.1.2 9 0.0000 0.01% 0.0200 211538301.467
10.10.47.123 31 0.0000 0.04% 0.1100 568415444.167
10.10.57.178 58570 0.0000 71.94% 2.2200 568415475.092
10.100.1.33 58194 0.0000 71.47% 2.2200 568415475.092
101.201.172.235 32 0.0000 0.04% 0.0600 442267517.023
103.3.62.64 49 0.0000 0.06% 0.1200 442267517.177
104.131.15.86 10 0.0000 0.01% 0.0400 442267517.196
104.140.244.186 40 0.0000 0.05% 0.1500 442267516.957
104.236.136.96 19 0.0000 0.02% 0.0400 442267517.633
104.236.57.24 19 0.0000 0.02% 0.0400 442267517.453
106.14.95.39 10 0.0000 0.01% 0.0400 442267517.554
107.191.60.255 91 0.0000 0.11% 0.2700 442267517.654
107.191.99.227 65 0.0000 0.08% 0.1400 442267517.379
109.173.29.34 66 0.0000 0.08% 0.0300 211538233.712
Display filter:
Apply
Copy
Save as...
Close


Wireshark · DNS · Exercise.pcapng
Topic / Item Count Average Min val Max val Rate (ms) Percent Burst rate
Payload size 171 107.75 29 502 0.0000 100% 0.1600
Query Stats 0 0.0000 100% -
Qname Len 86 24.81 9 55 0.0000 0.0800
Label Stats 0 0.0000 -
1st Level 0 0.0000 -
2nd Level 1 0.0000 0.0100
3rd Level 38 0.0000 0.0400
4th Level or more 47 0.0000 0.0800
Response Stats 0 0.0000 100% -
no. of questions 170 1.00 1 1 0.0000 0.1600
no. of answers 170 1.67 0 8 0.0000 0.1600
no. of additionals 170 1.68 0 13 0.0000 0.1600
no. of authorities 170 2.69 0 13 0.0000 0.1600
Service Stats 0 0.0000 100% -
request-response time (secs) 85 0.07 0.000075 0.467897 0.0000 0.0800
no. of retransmissions 0 0.0000 -
no. of unsolicited responses 0 0.0000 -
Total Packets 171 0.0000 100% 0.1600
Class 171 0.0000 100.00% 0.1600
IN 171 0.0000 100.00% 0.1600
Display filter:
Apply
Copy
Save as...
Close



Task 4 Packet Filtering | Principles-

The typical use case is capturing everything and filtering the packets according to the event of interest. Only experienced professionals use capture filters and sniff traffic. This is why Wireshark supports more protocol types in display filters. Please ensure you thoroughly learn how to use capture filters before using them in a live environment. Remember, you cannot capture the event of interest if your capture filter is not matching the specific traffic pattern you are looking for.

Capture Filters	This type of filter is used to save only a specific part of the traffic. It is set before capturing traffic and not changeable during the capture.
Display Filters	This type of filter is used to investigate packets by reducing the number of visible packets, and it is changeable during the capture.

Capture Filter Syntax

These filters use byte offsets hex values and masks with boolean operators, and it is not easy to understand/predict the filter's purpose at first glance. The base syntax is explained below:

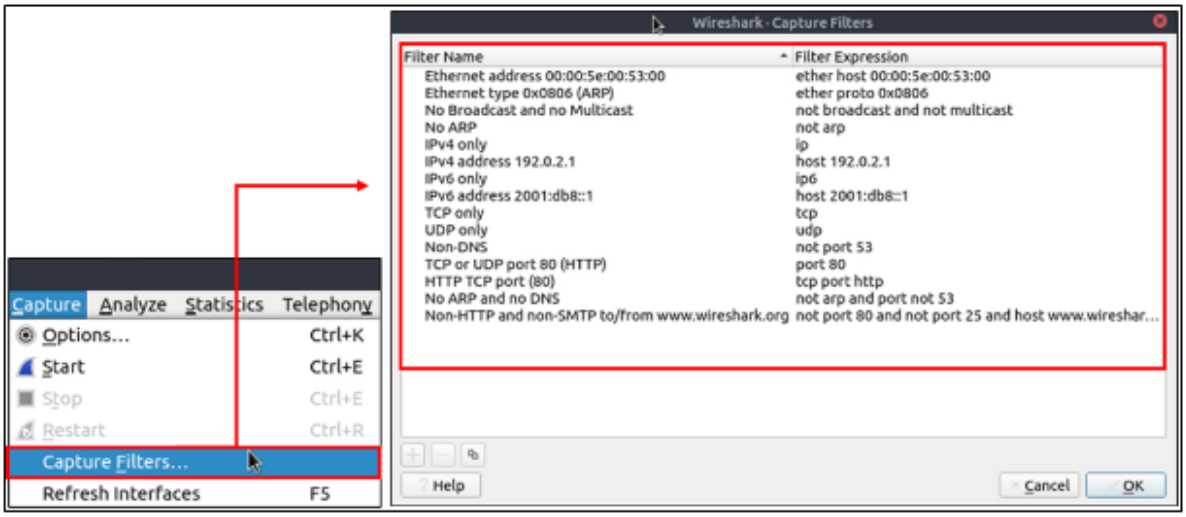
Scope: host, net, port and portrange.

Direction: src, dst, src or dst, src and dst,

Protocol: ether, wlan, ip, ip6, arp, rarp, tcp and udp.

Sample filter to capture port 80 traffic: tcp port 80

You can read more on capture filter syntax from here and here. A quick reference is available under the **"Capture --> Capture Filters"** menu.

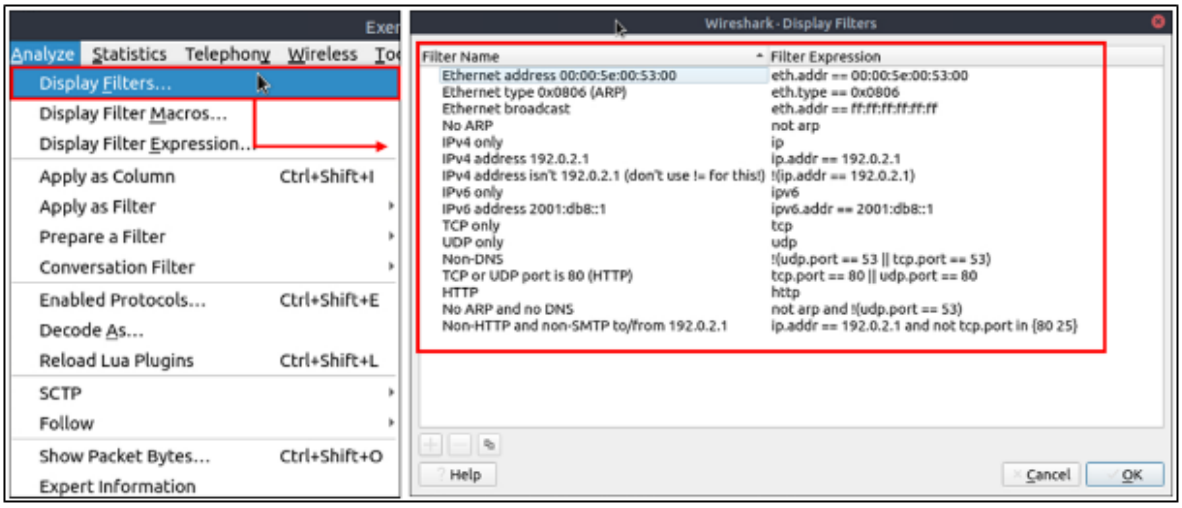


Display Filter Syntax

This is Wireshark's most powerful feature. It supports 3000 protocols and allows conducting packet-level searches under the protocol breakdown. The official "Display Filter Reference" provides all supported protocols breakdown for filtering.

Sample filter to capture port 80 traffic: tcp.port == 80

Wireshark has a built-in option (Display Filter Expression) that stores all supported protocol structures to help analysts create display filters. We will cover the "Display Filter Expression" menu later. Now let's understand the fundamentals of the display filter operations. A quick reference is available under the **"Analyse --> Display Filters"** menu.



Comparison Operators

You can create display filters by using different comparison operators to find the event of interest. The primary operators are shown in the table below.

English	C-Like	Description	Example
eq	==	Equal	`ip.src == 10.10.10.100`
ne	!=	Not equal	`ip.src != 10.10.10.100`
gt	>	Greater than	`ip.ttl > 250`
lt	<	Less Than	`ip.ttl < 10`
ge	>=	Greater than or equal to	`ip.ttl >= 0xFA`
le	<=	Less than or equal to	`ip.ttl <= 0xA`

Note: Wireshark supports decimal and hexadecimal values in filtering. You can use any format you want according to the search you will conduct.

Logical Expressions

Wireshark supports boolean syntax. You can create display filters by using logical operators as well.

English	C-Like	Description	Example
and	&&	Logical AND	`(ip.src == 10.10.10.100) AND (ip.src == 10.10.10.111)`
or	||	Logical OR	`(ip.src == 10.10.10.100) OR (ip.src == 10.10.10.111)`
not	!	Logical NOT	`!(ip.src == 10.10.10.222)` **Note:** Usage of `!=value` is deprecated; using it could provide inconsistent results. Using the `!(value)` style is suggested for more consistent results.

Packet Filter Toolbar

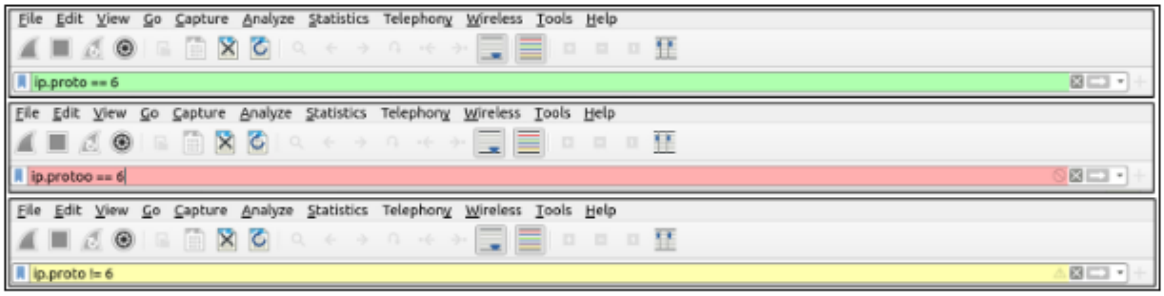
The filter toolbar is where you create and apply your display filters. It is a smart toolbar that helps you create valid display filters with ease. Before starting to filter packets, here are a few tips:

Packet filters are defined in lowercase.

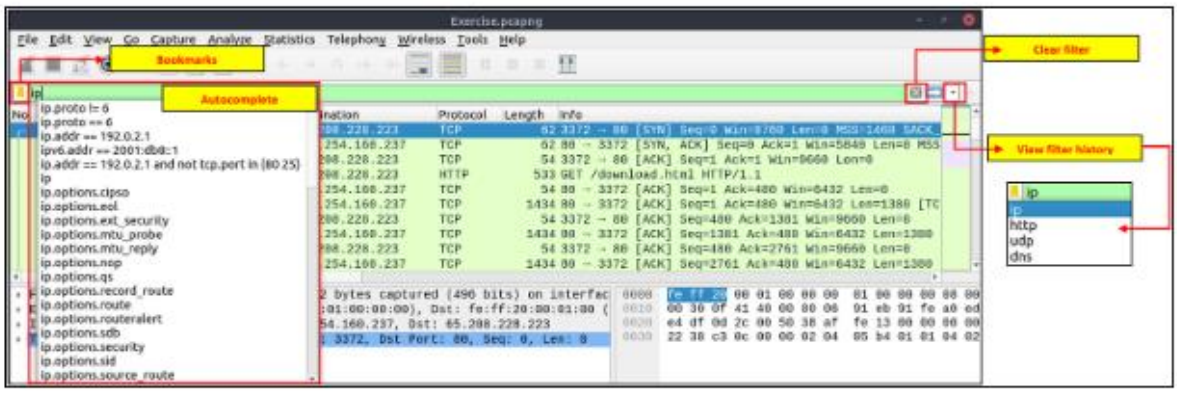
Packet filters have an autocomplete feature to break down protocol details, and each detail is represented by a "dot".

Packet filters have a three-colour representation explained below.

Green	Valid filter
Red	Invalid filter
Yellow	Warning filter. This filter works, but it is unreliable, and it is suggested to char



Filter toolbar features are shown below.



We've covered lots of principles and syntax. Let's put these into practice and start filtering packets in the next task.

Answer to the questions of this section-

No Answer needed

Task 5 Packet Filtering | Protocol Filters-

Protocol Filters

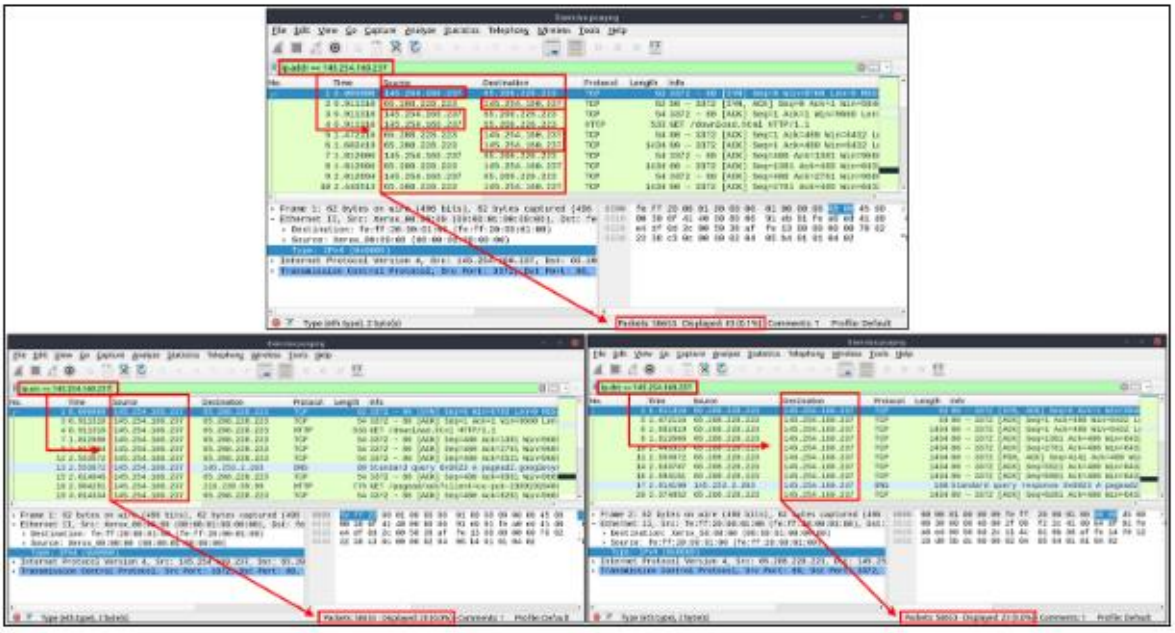
As mentioned in the previous task, Wireshark supports 3000 protocols and allows packet-level investigation by filtering the protocol fields. This task shows the creation and usage of filters against different protocol fields.

IP Filters

IP filters help analysts filter the traffic according to the IP level information from the packets (Network layer of the OSI model). This is one of the most commonly used filters in Wireshark. These filters filter network-level information like IP addresses, version, time to live, type of service, flags, and checksum values.

The common filters are shown in the given table.

Filter	Descripti
`ip`	Show all IP packets.
`ip.addr == 10.10.10.111`	Show all packets containing IP address 10.10.10.111.
`ip.addr == 10.10.10.0/24`	Show all packets containing IP addresses from 10.10.10.0/24
`ip.src == 10.10.10.111`	Show all packets originated from 10.10.10.111
`ip.dst == 10.10.10.111`	Show all packets sent to 10.10.10.111
ip.addr **vs** ip.src/ip.dst	**Note:** The ip.addr filters the traffic without considering the p depending on the packet direction.

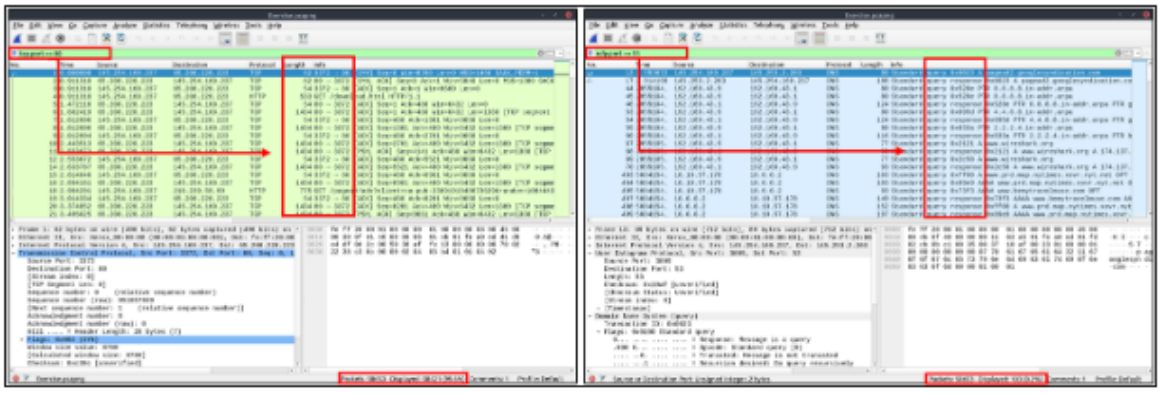


TCP and UDP Filters

TCP filters help analysts filter the traffic according to protocol-level information from the packets (Transport layer of the OSI model). These filters filter transport protocol level information like source

and destination ports, sequence number, acknowledgement number, windows size, timestamps, flags, length and protocol errors.

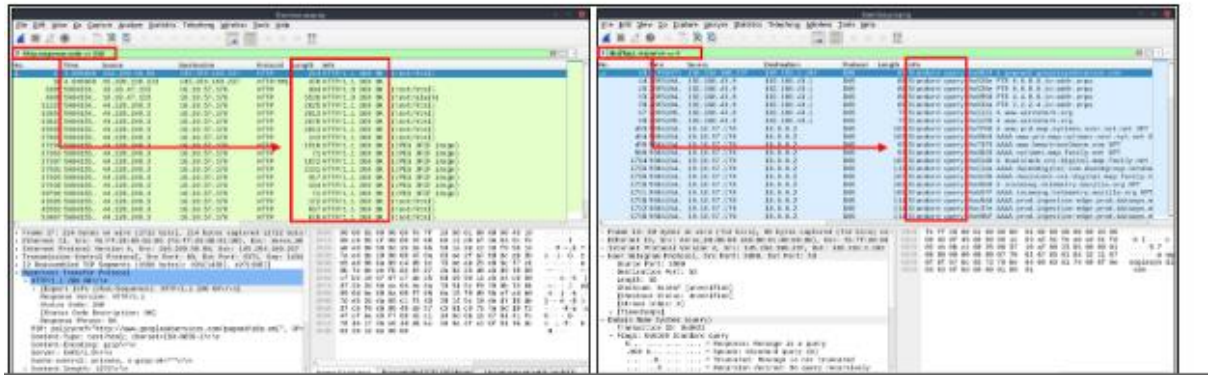
Filter	Description	Filter
`tcp.port == 80`	Show all TCP packets with port 80	`udp.port ==`
`tcp.srcport == 1234`	Show all TCP packets originating from port 1234	`udp.srcport == 1234`
`tcp.dstport == 80`	Show all TCP packets sent to port 80	`udp.dstport == 5353`



Application Level Protocol Filters | HTTP and DNS

Application-level protocol filters help analysts filter the traffic according to application protocol level information from the packets (Application layer of the OSI model). These filters filter application-specific information, like payload and linked data, depending on the protocol type.

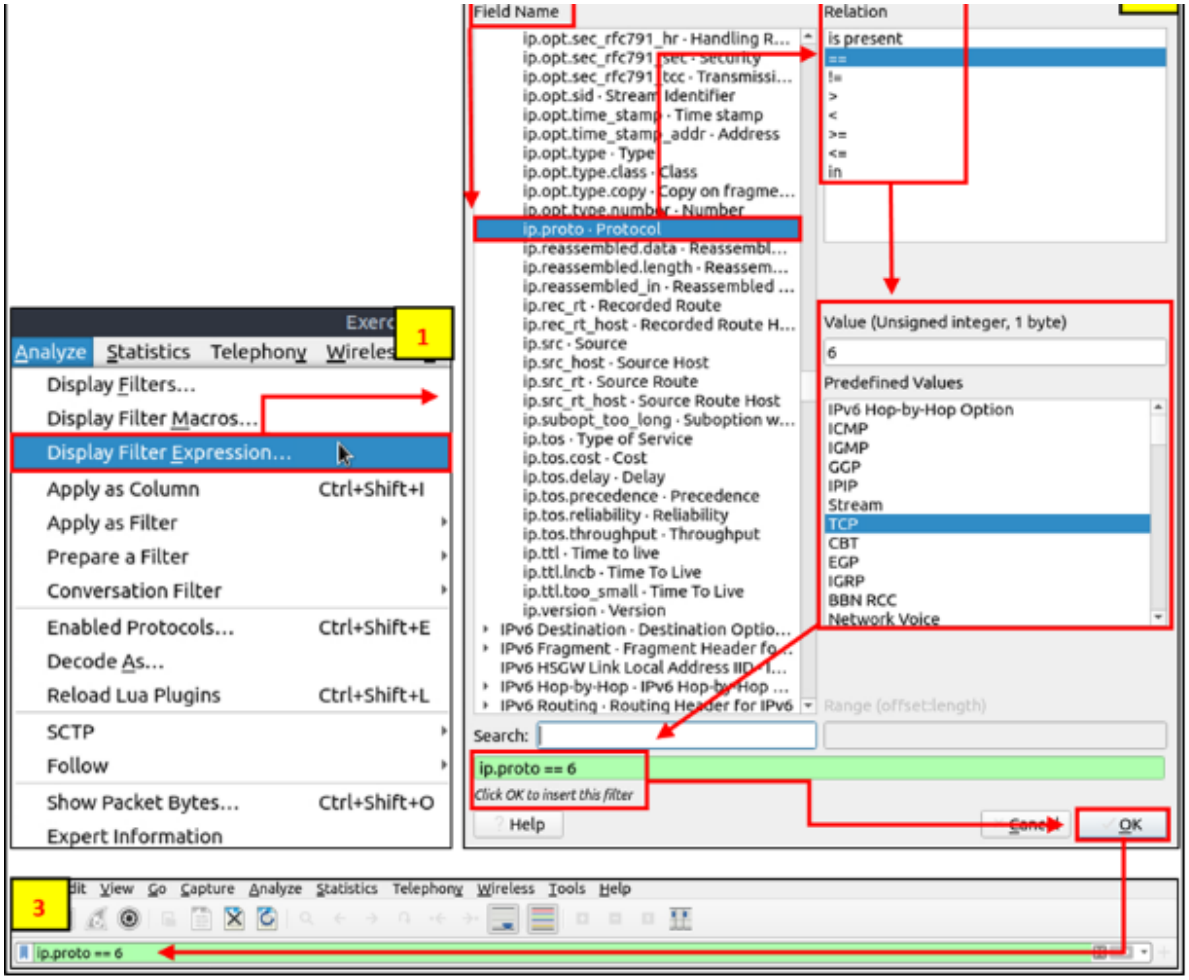
`http`	Show all HTTP packets
`http.response.code == 200`	Show all packets with HTTP response code "200"
`http.request.method == "GET"`	Show all HTTP GET requests
`http.request.method == "POST"`	Show all HTTP POST requests



Display Filter Expressions

As mentioned earlier, Wireshark has a built-in option (Display Filter Expression) that stores all supported protocol structures to help analysts create display filters. When an analyst can't recall the required filter for a specific protocol or is unsure about the assignable values for a filter, the Display Filter Expressions menu provides an easy-to-use display filter builder guide. It is available under the **"Analyse --> Display Filter Expression"** menu.

It is impossible to memorise all details of the display filters for each protocol. Each protocol can have different fields and can accept various types of values. The Display Filter Expressions menu shows all protocol fields, accepted value types (integer or string) and predefined values (if any). Note that it will take time and require practice to master creating filters and learning the protocol filter fields.



Note: The first room introduced the "Colouring Rules" (Task-2). Now you know how to create display filters and filter the event of interest. You can use the "View --> Coloring Rules" menu to assign colours to highlight your display filter results.

Answer to the questions of this section-

What is the number of IP packets?

81420 Correct Answer

What is the number of packets with a "TTL value less than 10"?

66 Correct Answer

What is the number of packets which uses "TCP port 4444"?

632 Correct Answer

What is the number of "HTTP GET" requests sent to port "80"?

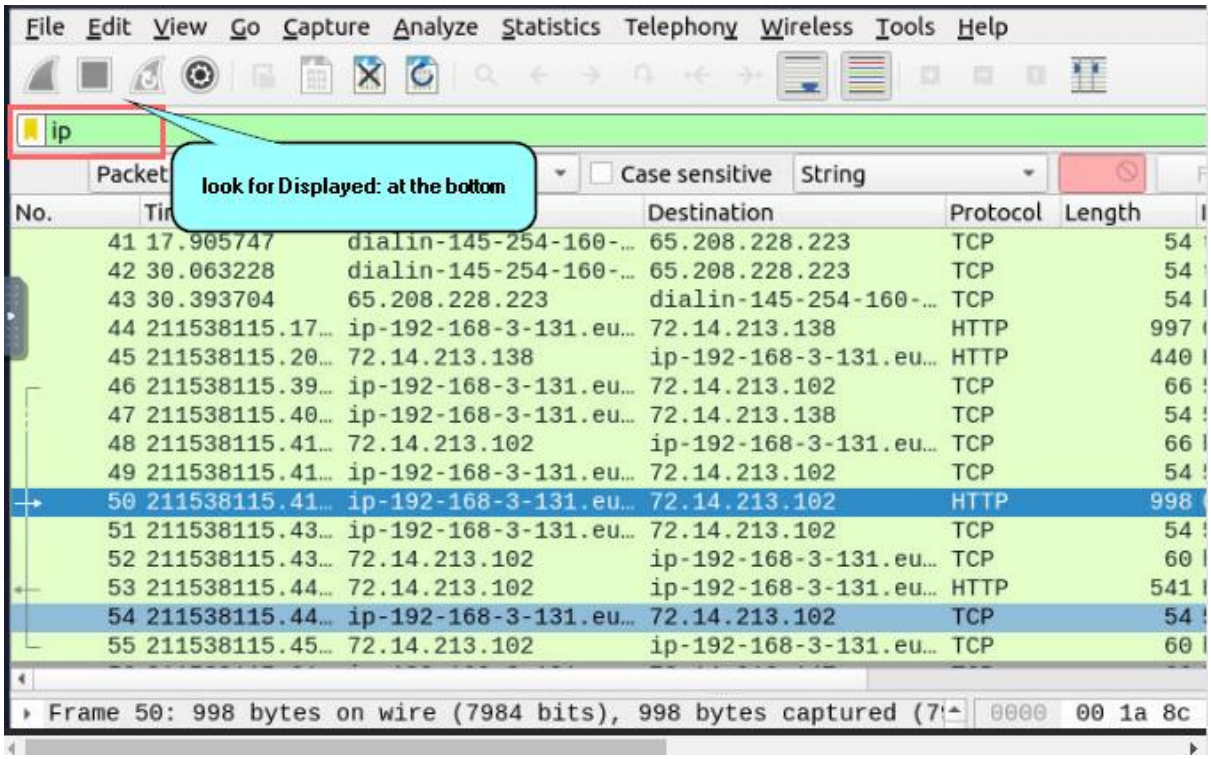
527 Correct Answer

What is the number of "type A DNS Queries"?

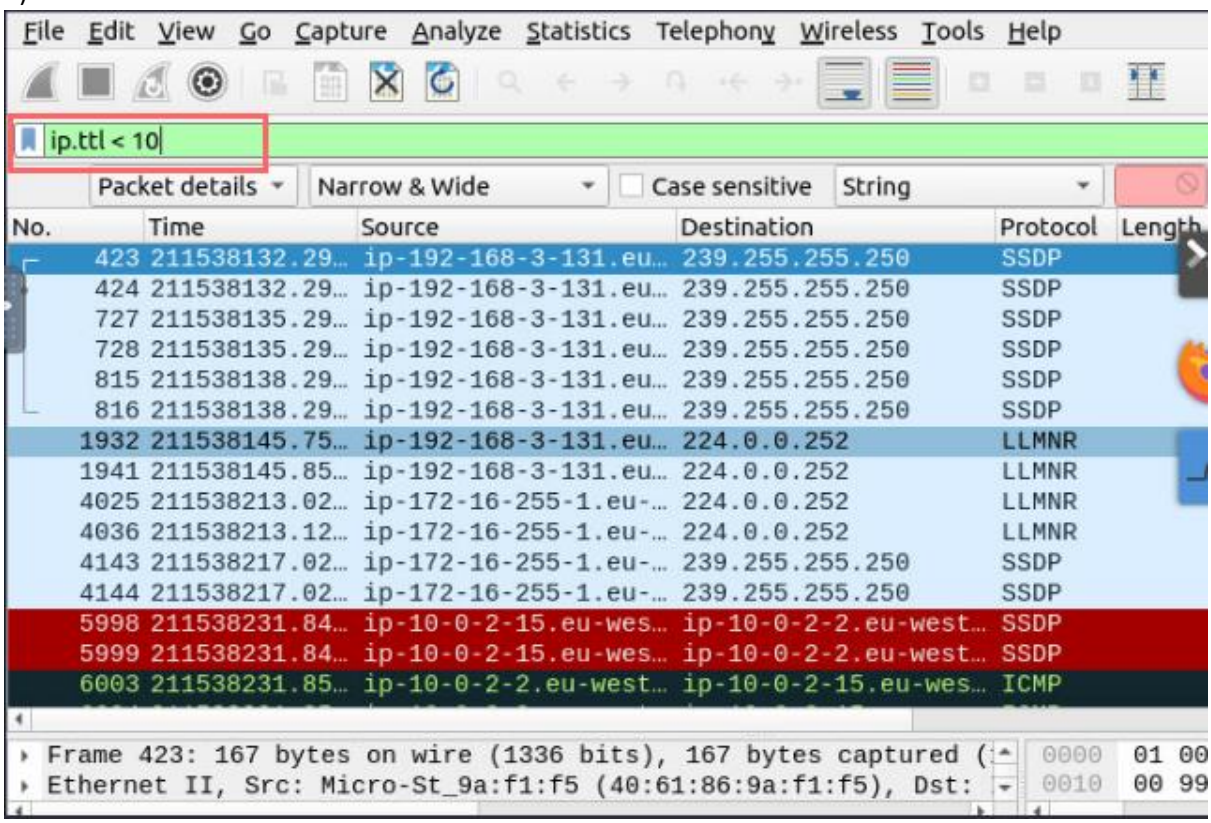
51 Correct Answer

Answers-

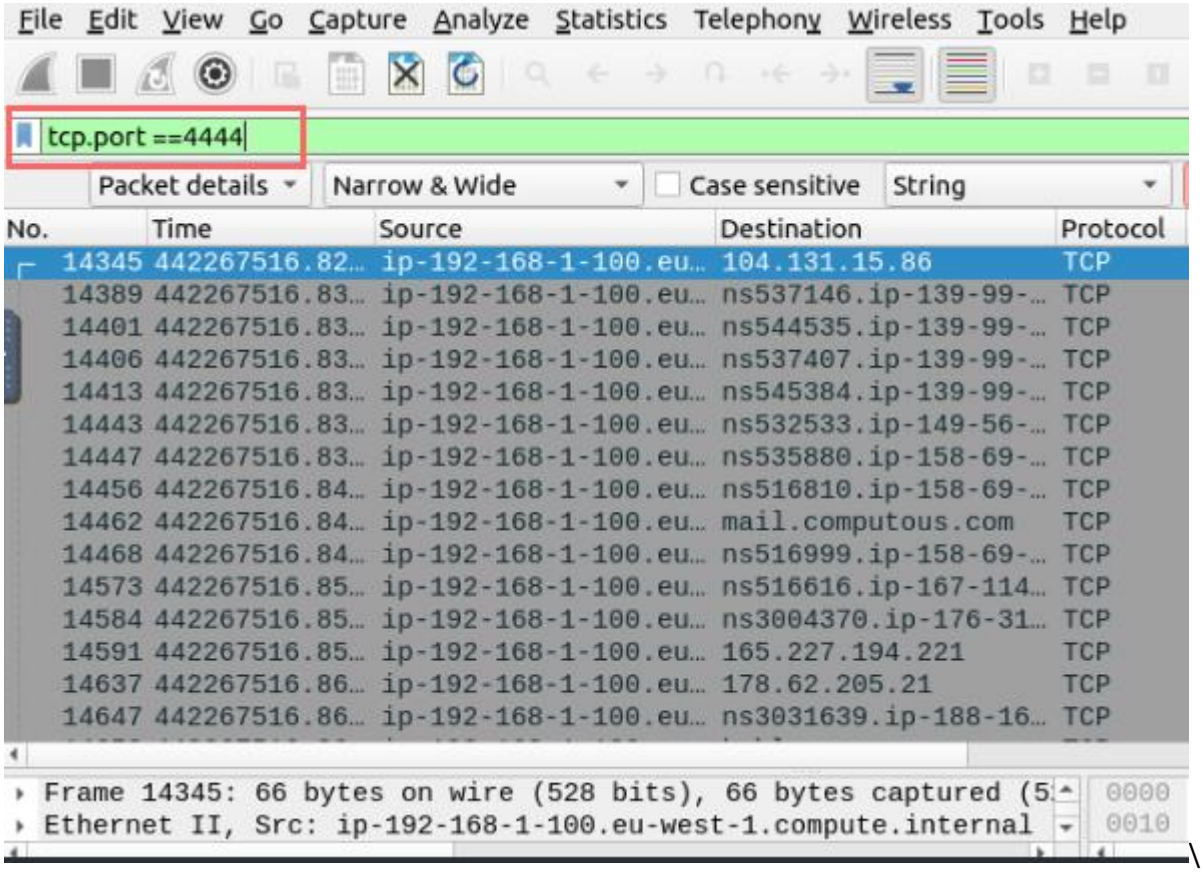
1)



2)

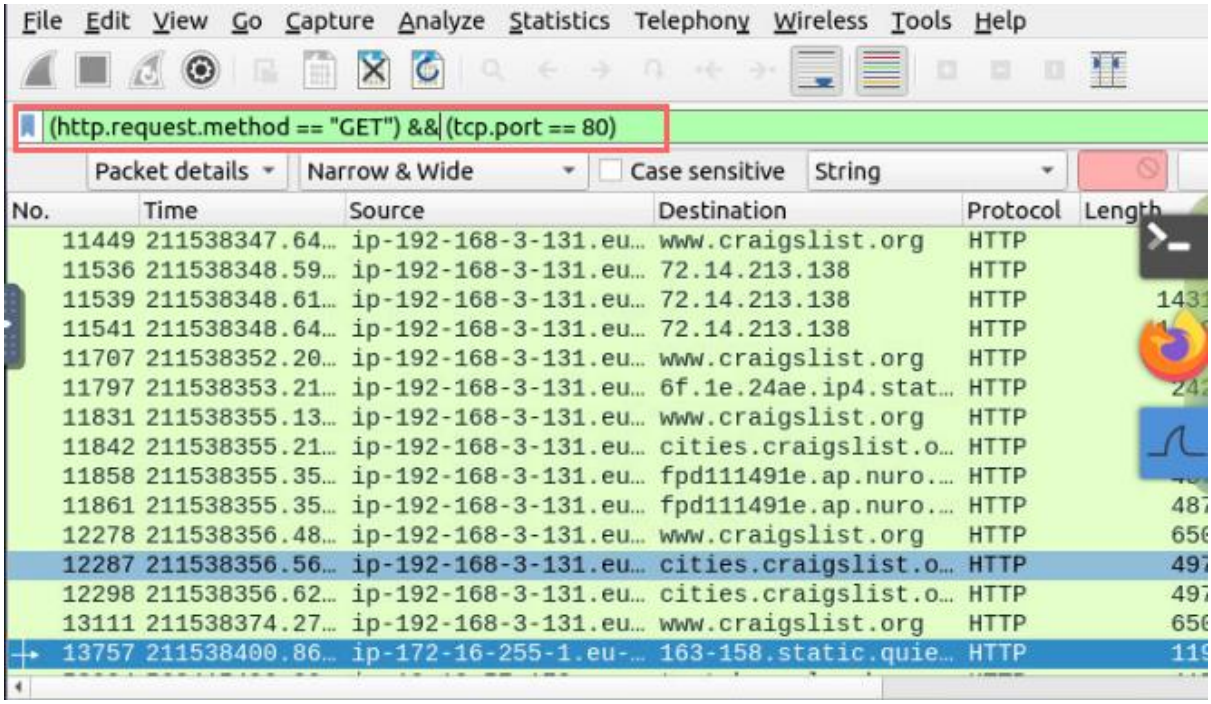

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ip.ttl < 10
Packet details
Narrow & Wide
Case sensitive
String
No. Time Source Destination Protocol Length
423 211538132.29… ip-192-168-3-131.eu… 239.255.255.250 SSDP
424 211538132.29… ip-192-168-3-131.eu… 239.255.255.250 SSDP
727 211538135.29… ip-192-168-3-131.eu… 239.255.255.250 SSDP
728 211538135.29… ip-192-168-3-131.eu… 239.255.255.250 SSDP
815 211538138.29… ip-192-168-3-131.eu… 239.255.255.250 SSDP
816 211538138.29… ip-192-168-3-131.eu… 239.255.255.250 SSDP
1932 211538145.75… ip-192-168-3-131.eu… 224.0.0.252 LLMNR
1941 211538145.85… ip-192-168-3-131.eu… 224.0.0.252 LLMNR
4025 211538213.02… ip-172-16-255-1.eu-… 224.0.0.252 LLMNR
4036 211538213.12… ip-172-16-255-1.eu-… 224.0.0.252 LLMNR
4143 211538217.02… ip-172-16-255-1.eu-… 239.255.255.250 SSDP
4144 211538217.02… ip-172-16-255-1.eu-… 239.255.255.250 SSDP
5998 211538231.84… ip-10-0-2-15.eu-wes… ip-10-0-2-2.eu-west… SSDP
5999 211538231.84… ip-10-0-2-15.eu-wes… ip-10-0-2-2.eu-west… SSDP
6003 211538231.85… ip-10-0-2-2.eu-west… ip-10-0-2-15.eu-wes… ICMP
Frame 423: 167 bytes on wire (1336 bits), 167 bytes captured (
Ethernet II, Src: Micro-St_9a:f1:f5 (40:61:86:9a:f1:f5), Dst:
0000 01 00
0010 00 99

3)

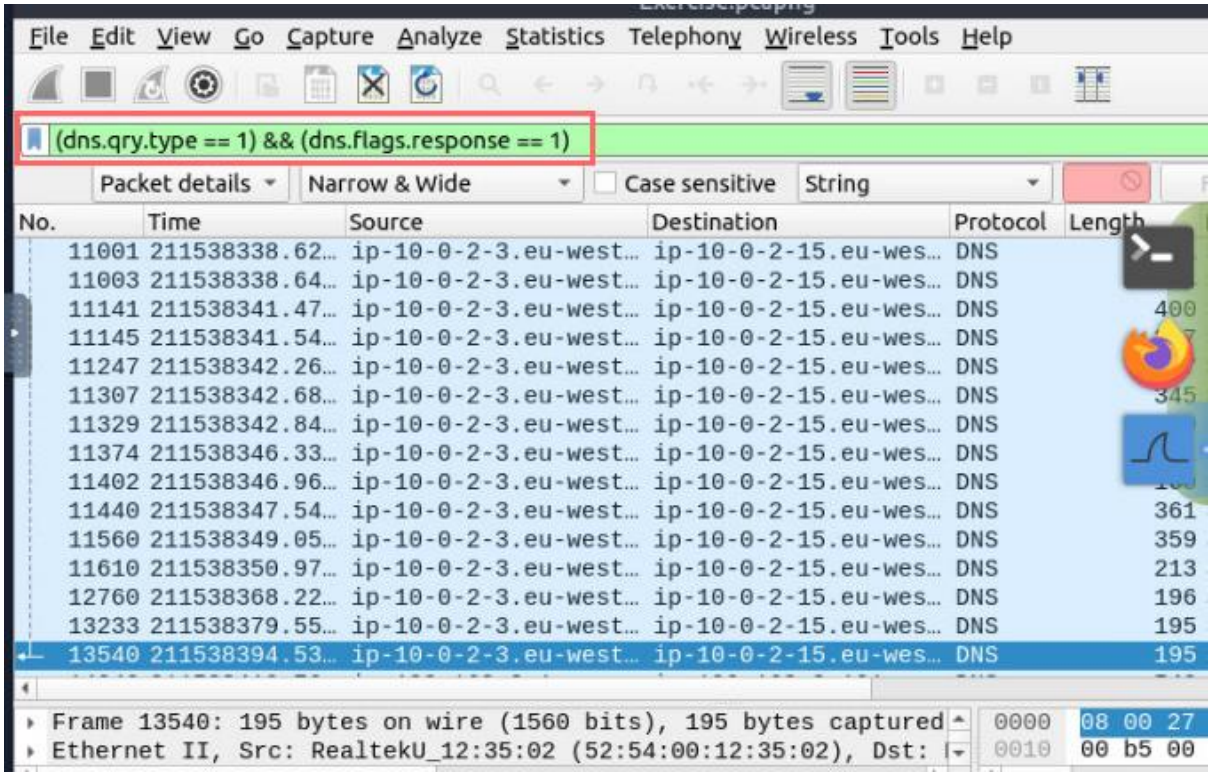

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.port ==4444
Packet details
Narrow & Wide
Case sensitive
String
No. Time Source Destination Protocol
14345 442267516.82… ip-192-168-1-100.eu… 104.131.15.86 TCP
14389 442267516.83… ip-192-168-1-100.eu… ns537146.ip-139-99-… TCP
14401 442267516.83… ip-192-168-1-100.eu… ns544535.ip-139-99-… TCP
14406 442267516.83… ip-192-168-1-100.eu… ns537407.ip-139-99-… TCP
14413 442267516.83… ip-192-168-1-100.eu… ns545384.ip-139-99-… TCP
14443 442267516.83… ip-192-168-1-100.eu… ns532533.ip-149-56-… TCP
14447 442267516.83… ip-192-168-1-100.eu… ns535880.ip-158-69-… TCP
14456 442267516.84… ip-192-168-1-100.eu… ns516810.ip-158-69-… TCP
14462 442267516.84… ip-192-168-1-100.eu… mail.computous.com TCP
14468 442267516.84… ip-192-168-1-100.eu… ns516999.ip-158-69-… TCP
14573 442267516.85… ip-192-168-1-100.eu… ns516616.ip-167-114… TCP
14584 442267516.85… ip-192-168-1-100.eu… ns3004370.ip-176-31… TCP
14591 442267516.85… ip-192-168-1-100.eu… 165.227.194.221 TCP
14637 442267516.86… ip-192-168-1-100.eu… 178.62.205.21 TCP
14647 442267516.86… ip-192-168-1-100.eu… ns3031639.ip-188-16… TCP
Frame 14345: 66 bytes on wire (528 bits), 66 bytes captured (5
Ethernet II, Src: ip-192-168-1-100.eu-west-1.compute.internal
0000
0010

\

4)


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
(http.request.method == "GET") && (tcp.port == 80)
Packet details
Narrow & Wide
Case sensitive
String
No. Time Source Destination Protocol Length
11449 211538347.64… ip-192-168-3-131.eu… www.craigslist.org HTTP
11536 211538348.59… ip-192-168-3-131.eu… 72.14.213.138 HTTP
11539 211538348.61… ip-192-168-3-131.eu… 72.14.213.138 HTTP
11541 211538348.64… ip-192-168-3-131.eu… 72.14.213.138 HTTP
11707 211538352.20… ip-192-168-3-131.eu… www.craigslist.org HTTP
11797 211538353.21… ip-192-168-3-131.eu… 6f.1e.24ae.ip4.stat… HTTP
11831 211538355.13… ip-192-168-3-131.eu… www.craigslist.org HTTP
11842 211538355.21… ip-192-168-3-131.eu… cities.craigslist.o… HTTP
11858 211538355.35… ip-192-168-3-131.eu… fpd111491e.ap.nuro.… HTTP
11861 211538355.35… ip-192-168-3-131.eu… fpd111491e.ap.nuro.… HTTP
12278 211538356.48… ip-192-168-3-131.eu… www.craigslist.org HTTP
12287 211538356.56… ip-192-168-3-131.eu… cities.craigslist.o… HTTP
12298 211538356.62… ip-192-168-3-131.eu… cities.craigslist.o… HTTP
13111 211538374.27… ip-192-168-3-131.eu… www.craigslist.org HTTP
13757 211538400.86… ip-172-16-255-1.eu-… 163-158.static.quie… HTTP

5)


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
(dns.qry.type == 1) && (dns.flags.response == 1)
Packet details
Narrow & Wide
Case sensitive
String
No. Time Source Destination Protocol Length
11001 211538338.62… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11003 211538338.64… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11141 211538341.47… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11145 211538341.54… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11247 211538342.26… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11307 211538342.68… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11329 211538342.84… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11374 211538346.33… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11402 211538346.96… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11440 211538347.54… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11560 211538349.05… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
11610 211538350.97… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
12760 211538368.22… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
13233 211538379.55… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
13540 211538394.53… ip-10-0-2-3.eu-west… ip-10-0-2-15.eu-wes… DNS
Frame 13540: 195 bytes on wire (1560 bits), 195 bytes captured
Ethernet II, Src: RealtekU_12:35:02 (52:54:00:12:35:02), Dst:
0000 08 00 27
0010 00 b5 00

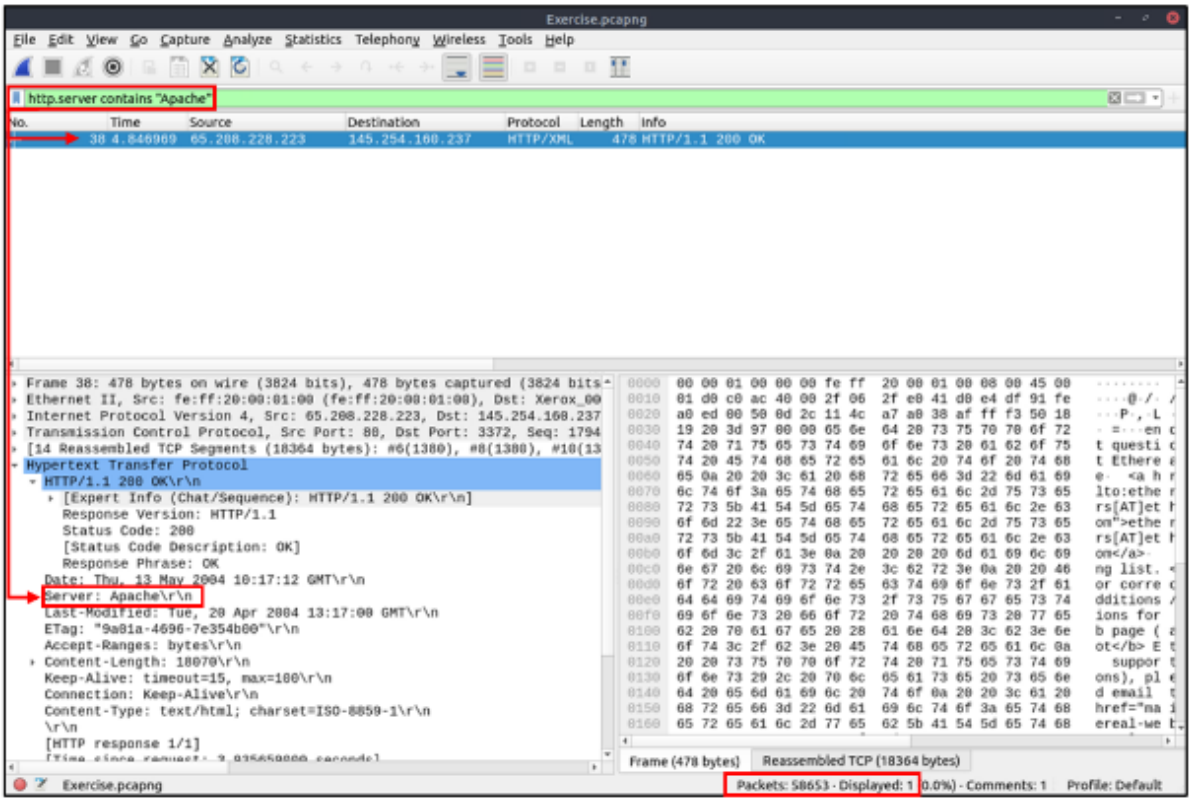
Task 6 Advanced Filtering-

Advanced Filtering

So far, you have learned the basics of packet filtering operations. Now it is time to focus on specific packet details for the event of interest. Besides the operators and expressions covered in the previous room, Wireshark has advanced operators and functions. These advanced filtering options help the analyst conduct an in-depth analysis of an event of interest.

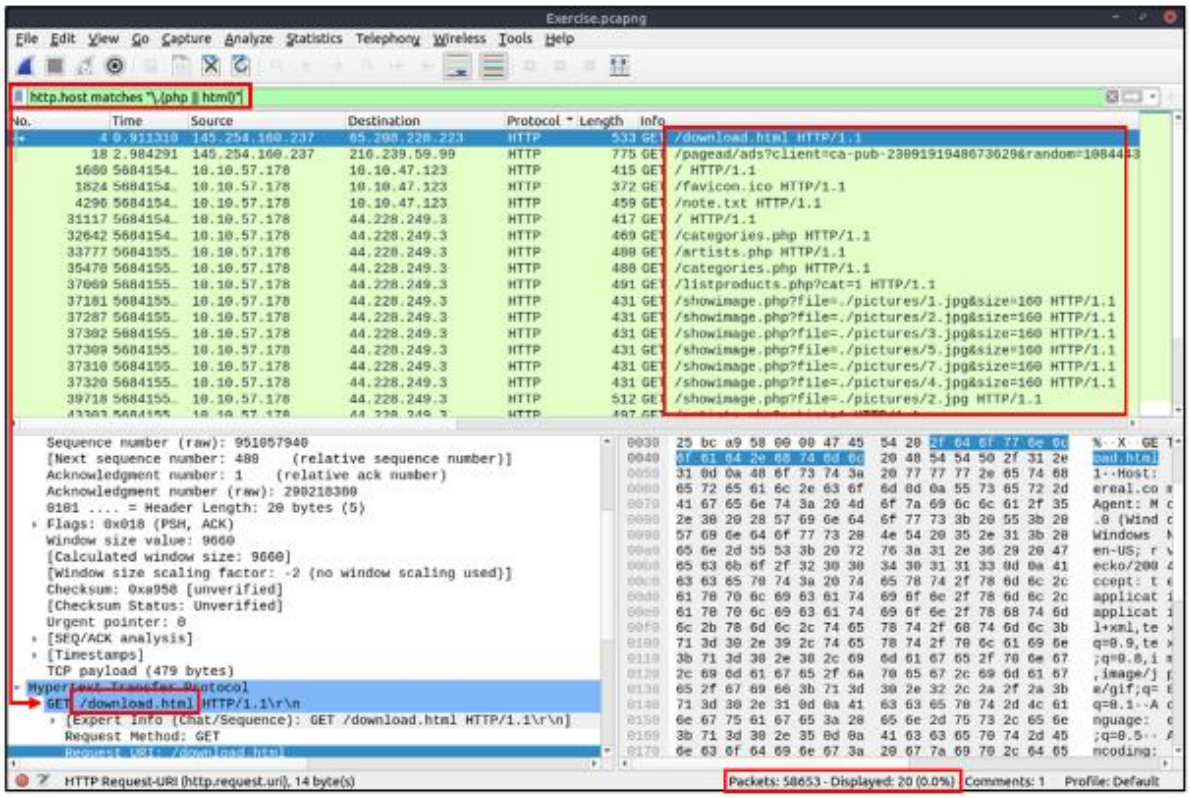
Filter: "contains"

Filter	**contains**
Type	Comparison Operator
Description	Search a value inside packets. It is case-sensitive and provides similar functionality to the "Find" option by focusing on a specific field.
Example	Find all "Apache" servers.
Workflow	List all HTTP packets where packets' "server" field contains the "Apache" keyword.
Usage	`http.server contains "Apache"`



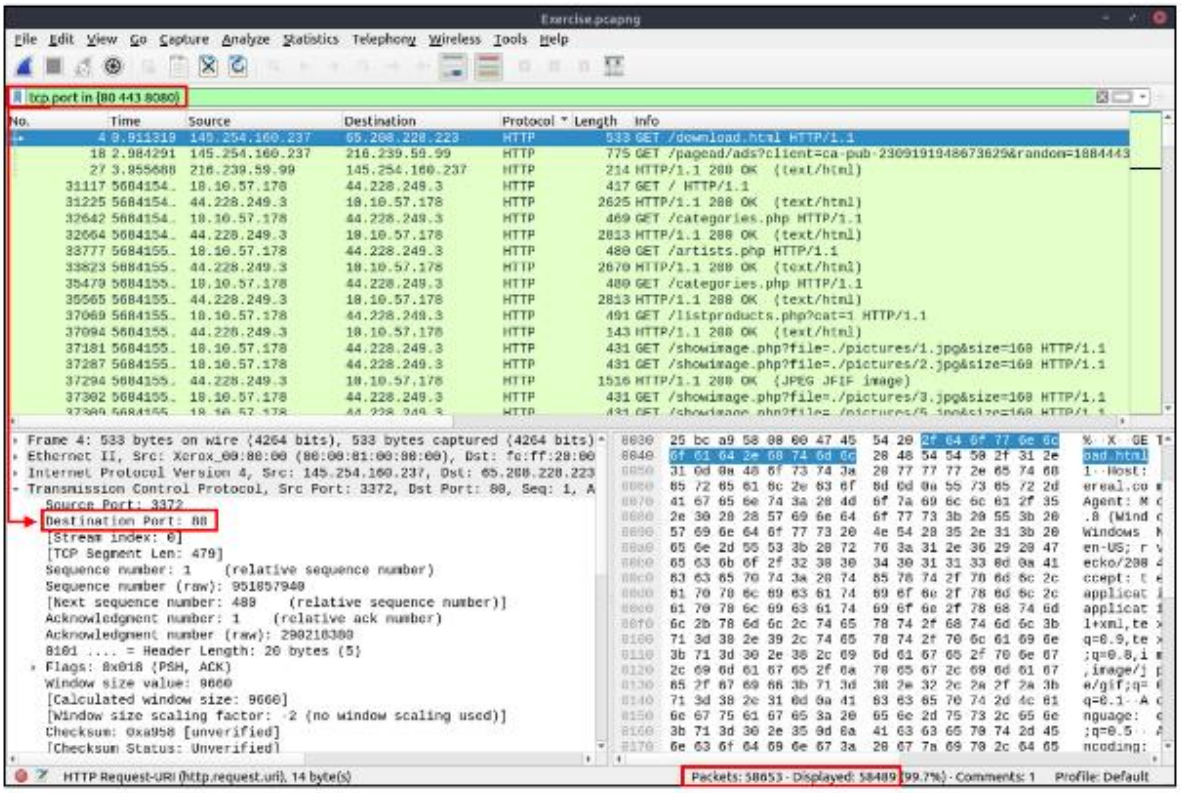
Filter: "matches"

Filter	**matches**
Type	Comparison Operator
Description	Search a pattern of a regular expression. It is case insensitive, and comple
Example	Find all .php and .html pages.
Workflow	List all HTTP packets where packets' "host" fields match keywords ".php"
Usage	`http.host matches "\.(php || html)"`



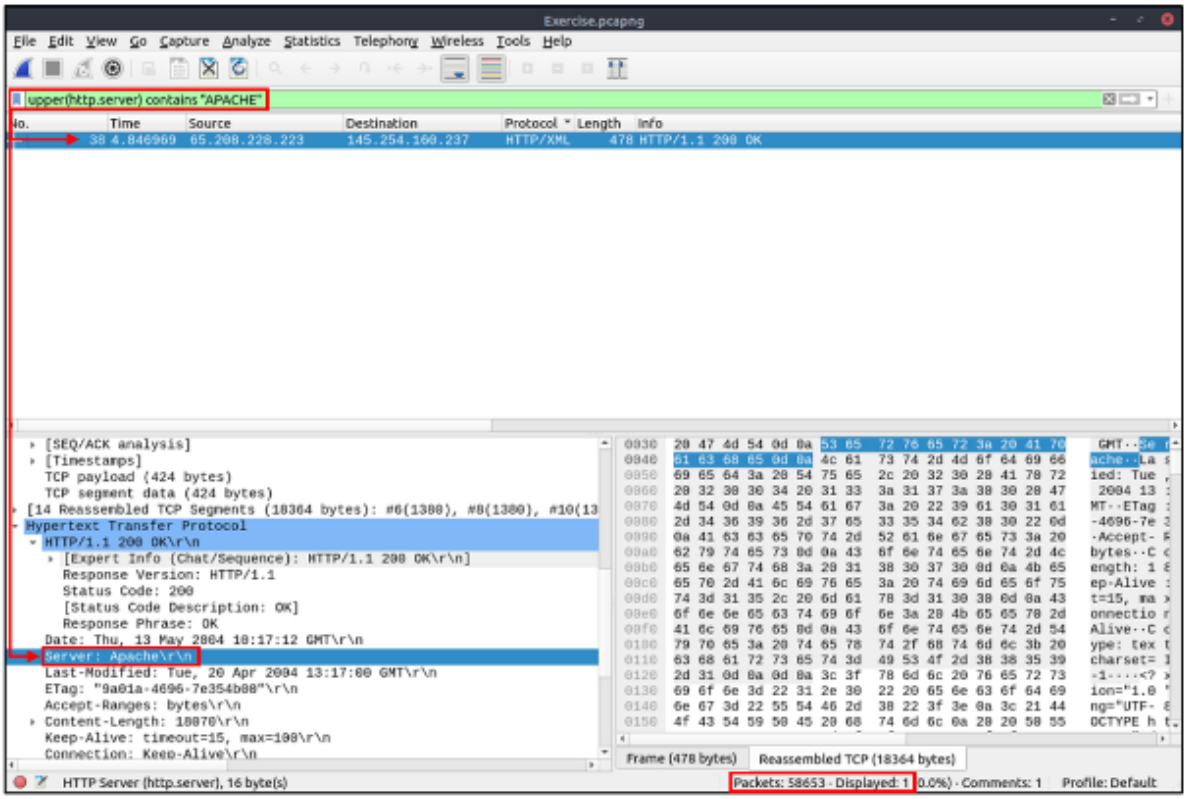
Filter: "in"

Filter	in
Type	Set Membership
Description	Search a value or field inside of a specific scope/range.
Example	Find all packets that use ports 80, 443 or 8080.
Workflow	List all TCP packets where packets' "port" fields have values 80, 443
Usage	`tcp.port in {80 443 8080}`



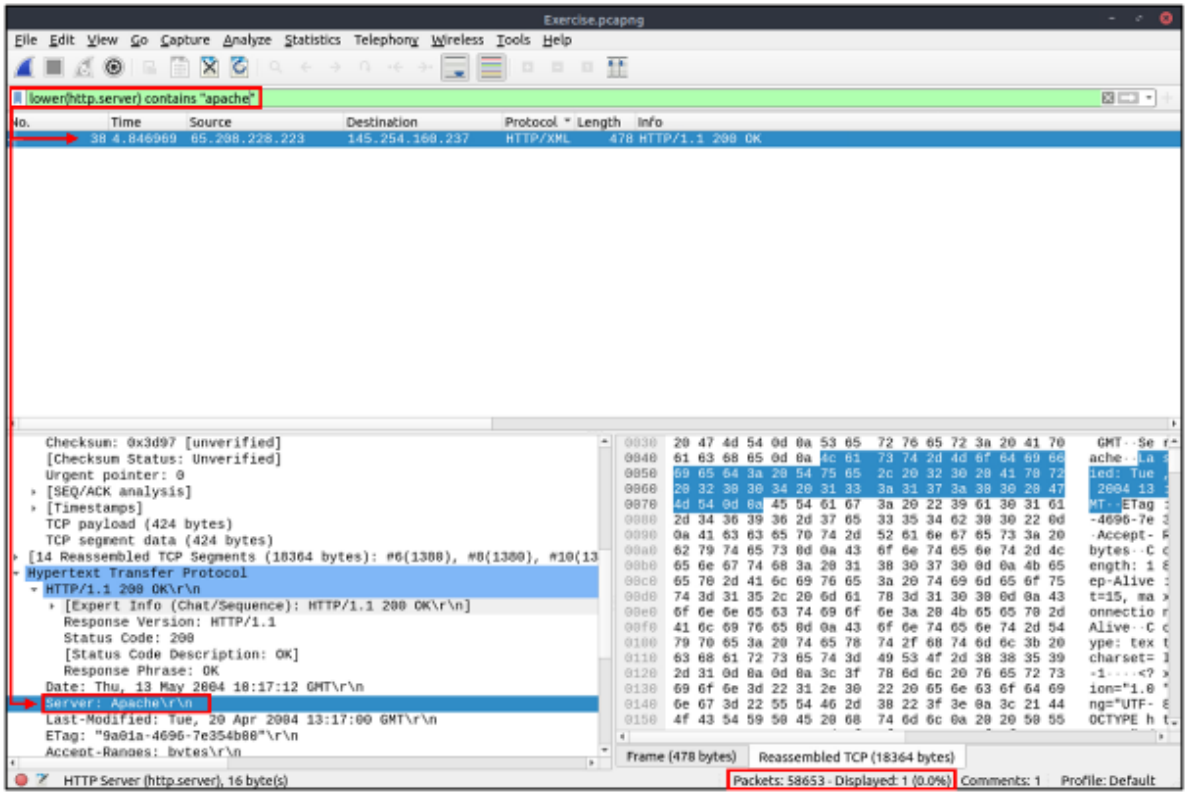
Filter: "upper"

Filter	**upper**
Type	Function
Description	Convert a string value to uppercase.
Example	Find all "APACHE" servers.
Workflow	Convert all HTTP packets' "server" fields to uppercase and list packets tha
Usage	`upper(http.server) contains "APACHE"`



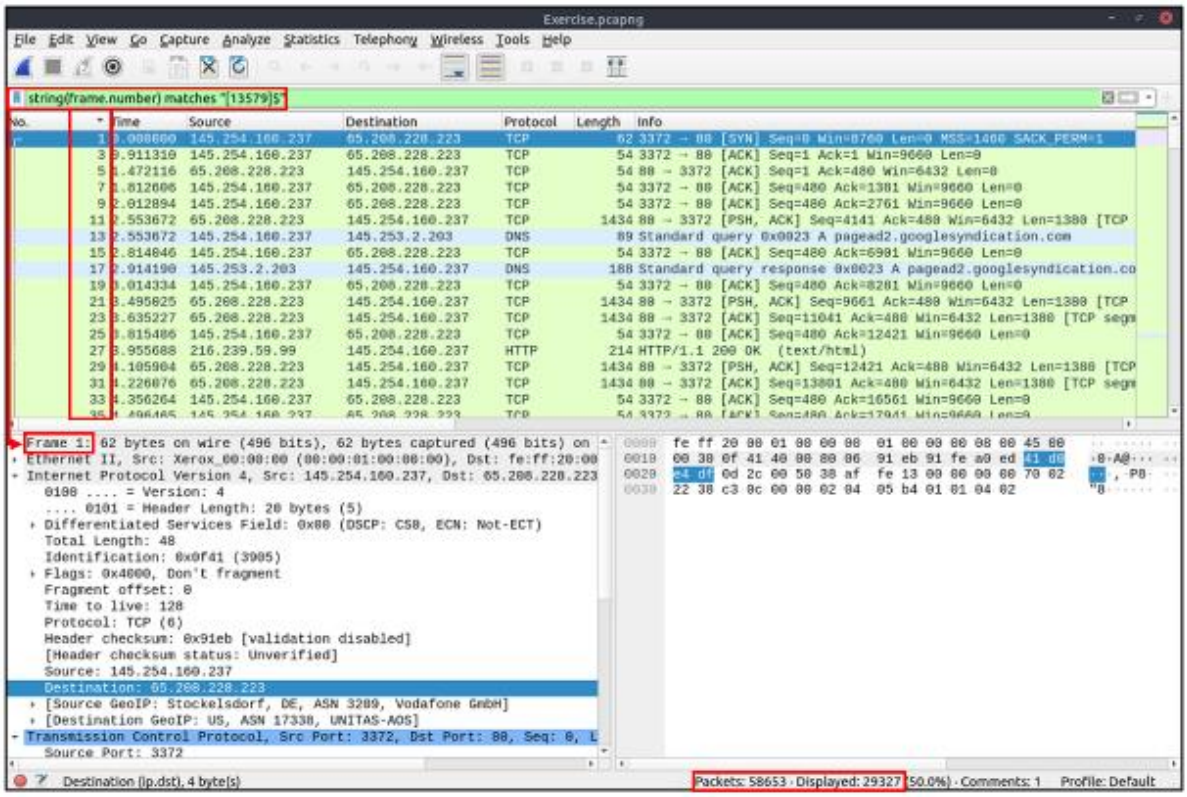
Filter: "lower"

Filter	**lower**
Type	Function
Description	Convert a string value to lowercase.
Example	Find all "apache" servers.
Workflow	Convert all HTTP packets' "server" fields info to lowercase and list packets
Usage	`lower(http.server) contains "apache"`



Filter: "string"

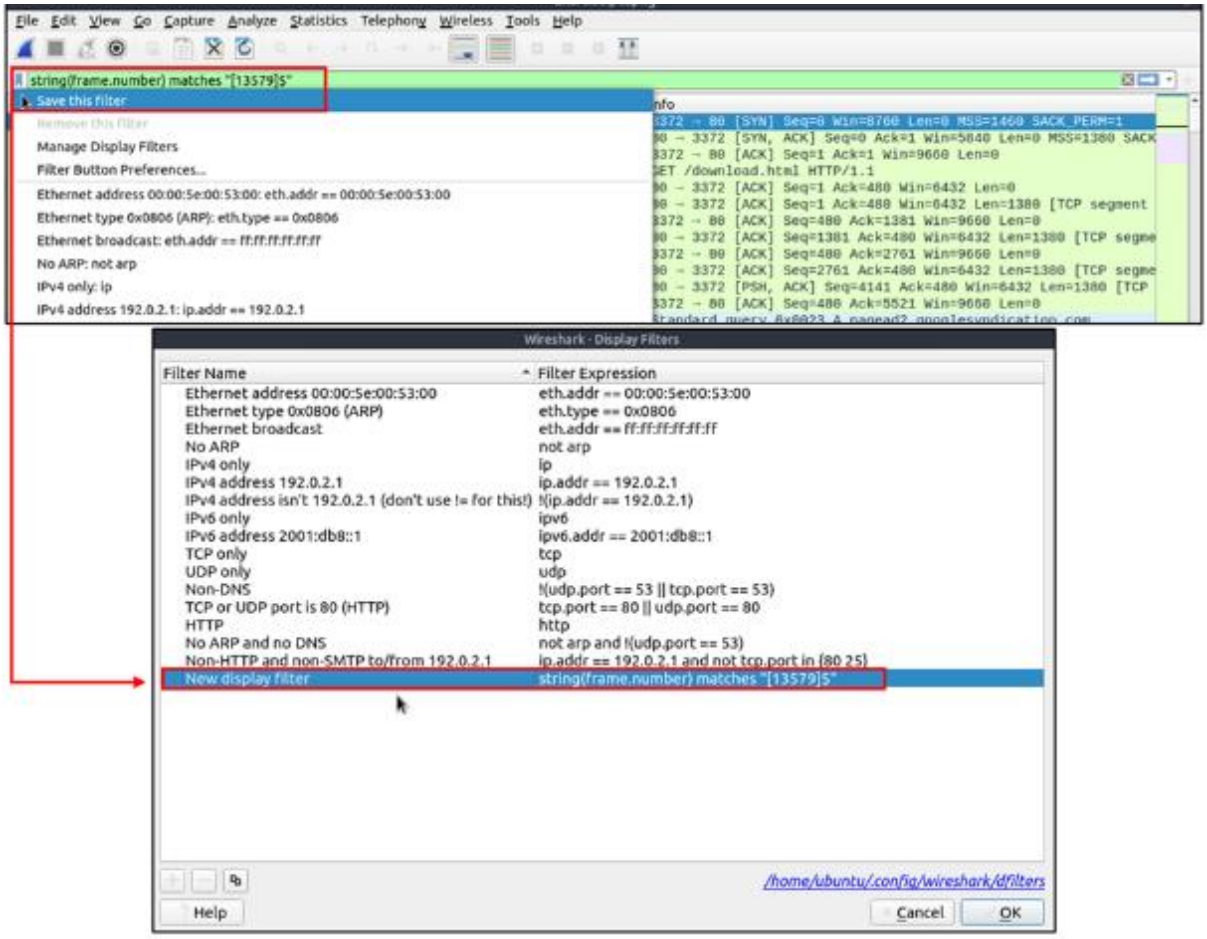
Filter	**string**
Type	Function
Description	Convert a non-string value to a string.
Example	Find all frames with odd numbers.
Workflow	Convert all "frame number" fields to string values, and list frames end
Usage	`string(frame.number) matches "[13579]$"`



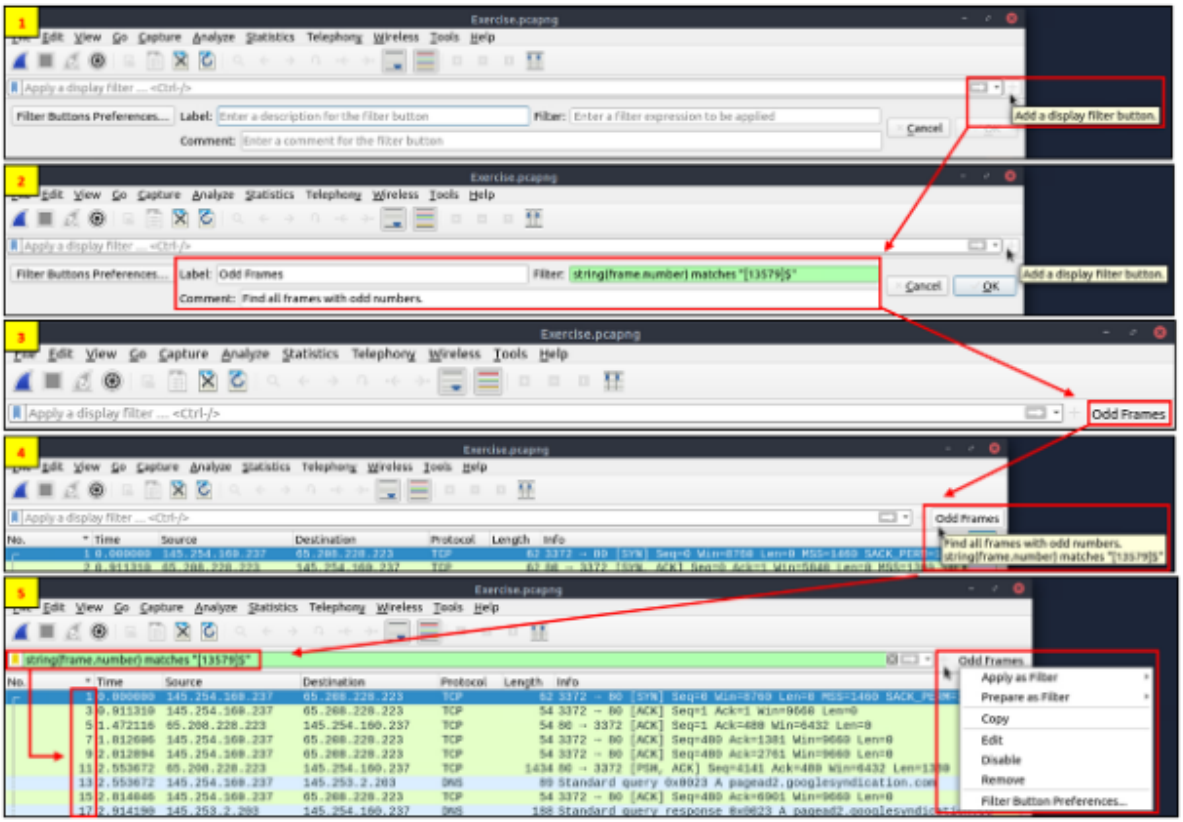
Bookmarks and Filtering Buttons

We've covered different types of filtering options, operators and functions. It is time to create filters and save them as bookmarks and buttons for later usage. As mentioned in the previous task, the filter toolbar has a filter bookmark section to save user-created filters, which helps analysts re-use favourite/complex filters with a couple of clicks. Similar to bookmarks, you can create filter buttons ready to apply with a single click.

Creating and using bookmarks.

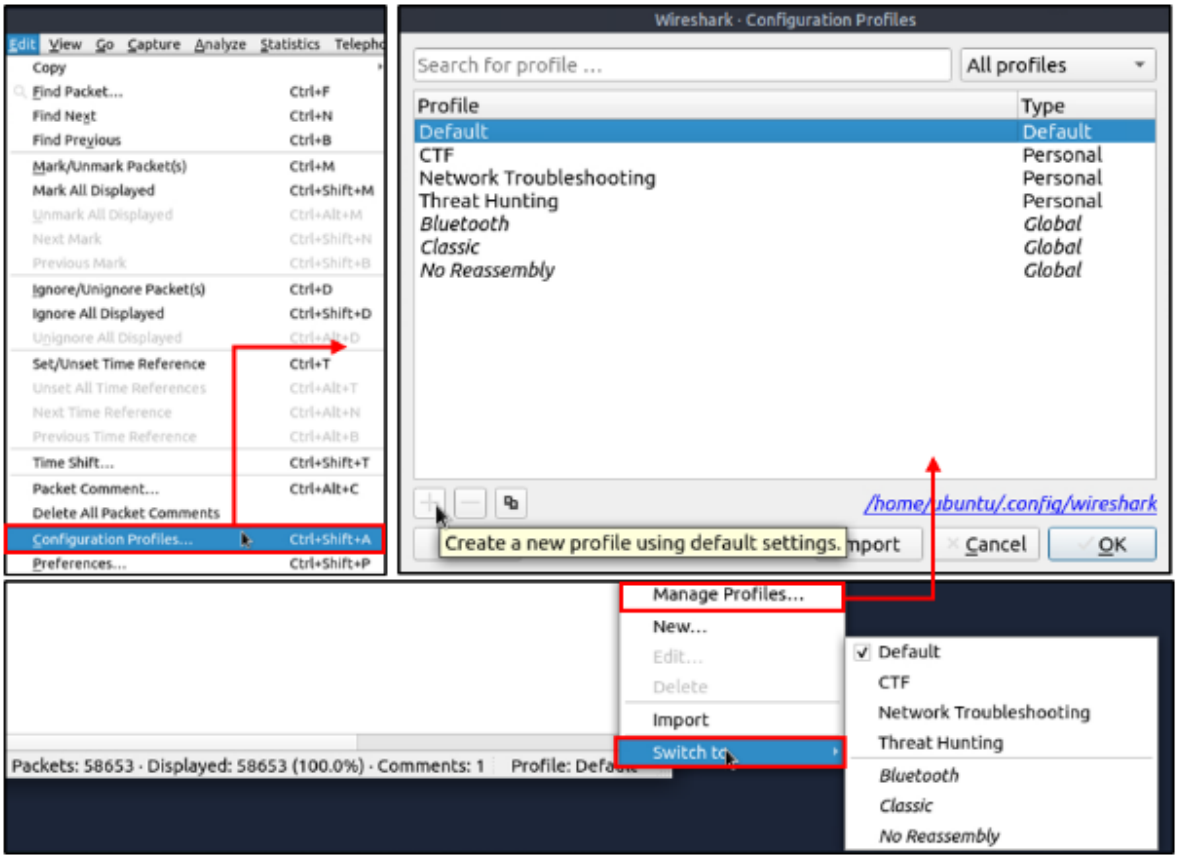


Creating and using display filter buttons.



Profiles

Wireshark is a multifunctional tool that helps analysts to accomplish in-depth packet analysis. As we covered during the room, multiple preferences need to be configured to analyse a specific event of interest. It is cumbersome to re-change the configuration for each investigation case, which requires a different set of colouring rules and filtering buttons. This is where Wireshark profiles come into play. You can create multiple profiles for different investigation cases and use them accordingly. You can use the "Edit --> Configuration Profiles" menu or the "lower right bottom of the status bar --> Profile" section to create, modify and change the profile configuration.



Answer to the questions of this section-

Find all Microsoft IIS servers. What is the number of packets that did not originate from "port 80"?

21 Correct Answer Hint

Find all Microsoft IIS servers. What is the number of packets that have "version 7.5"?

71 Correct Answer Hint

What is the total number of packets that use ports 3333, 4444 or 9999?

2235 Correct Answer Hint

What is the number of packets with "even TTL numbers"?

77289 Correct Answer Hint

Change the profile to "Checksum Control". What is the number of "Bad TCP Checksum" packets?

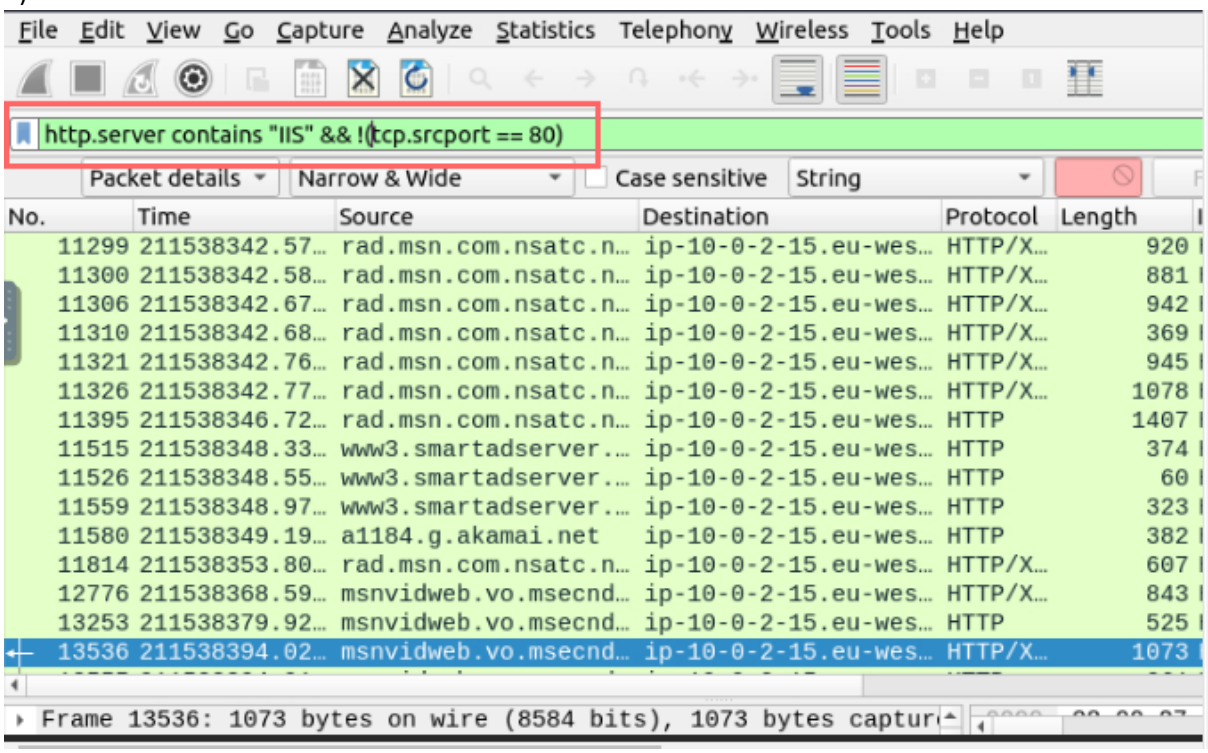
34185 Correct Answer Hint

Use the existing filtering button to filter the traffic. What is the number of displayed packets?

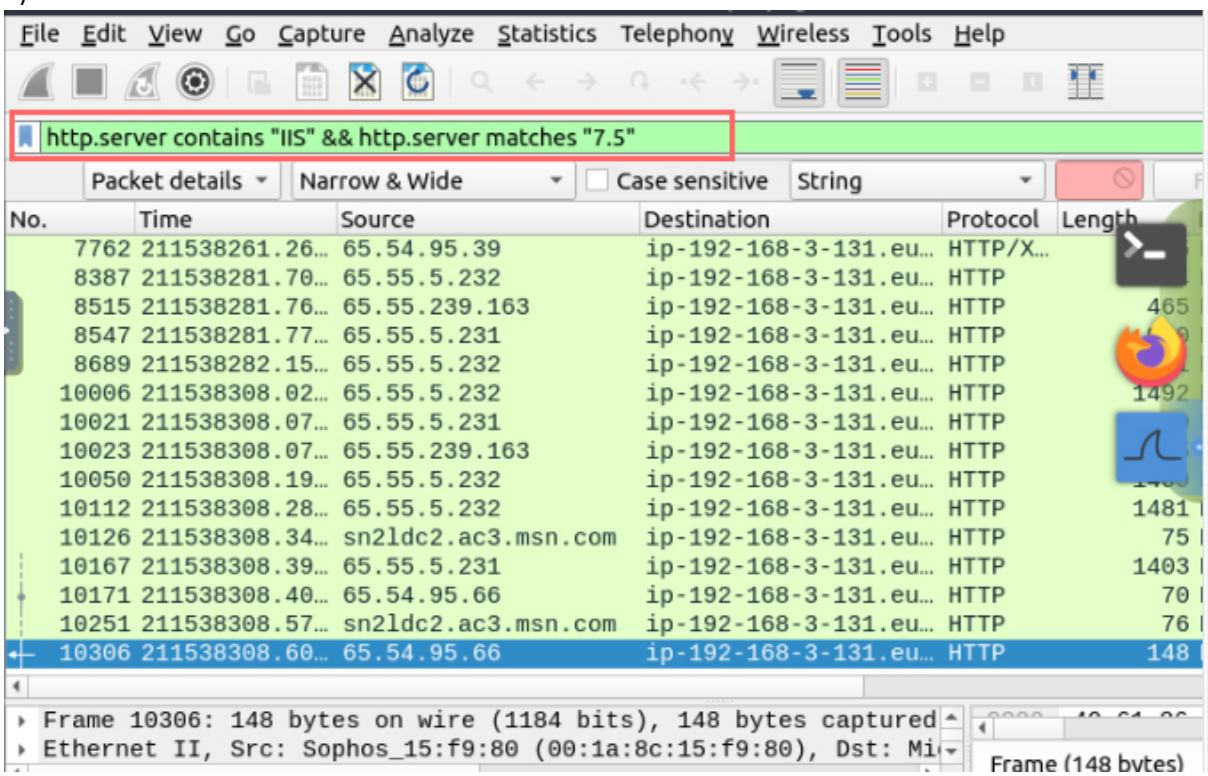
261 Correct Answer Hint

Answers-

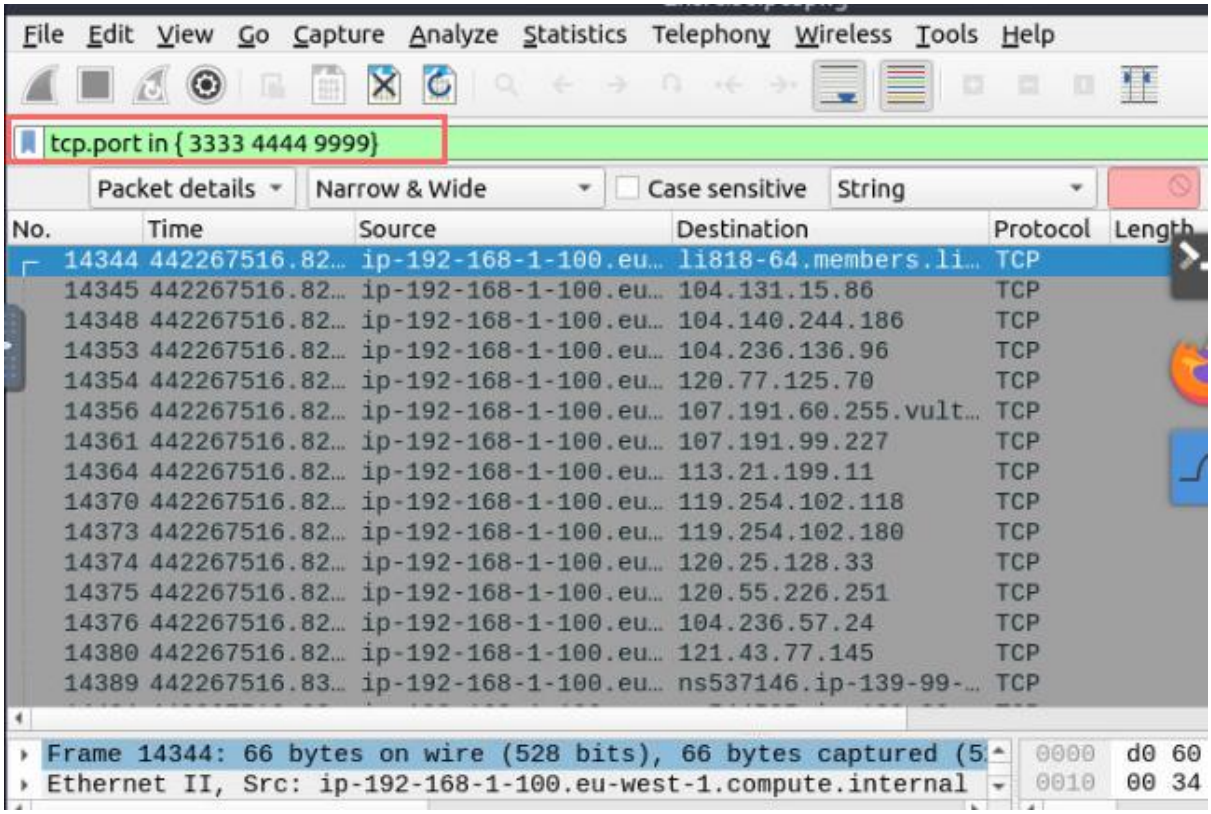
1)



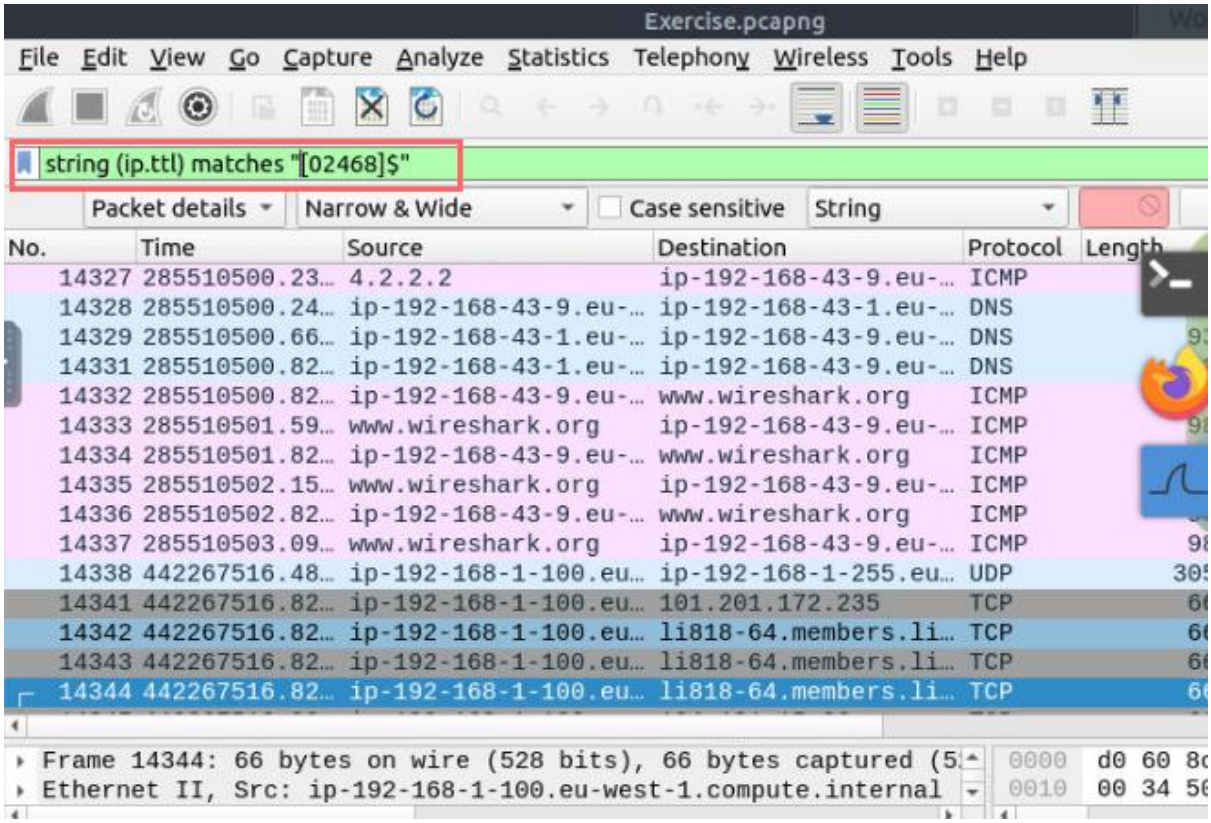
2)


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http.server contains "IIS" && http.server matches "7.5"
Packet details
Narrow & Wide
Case sensitive
String
No. Time Source Destination Protocol Length
7762 211538261.26… 65.54.95.39 ip-192-168-3-131.eu… HTTP/X…
8387 211538281.70… 65.55.5.232 ip-192-168-3-131.eu… HTTP
8515 211538281.76… 65.55.239.163 ip-192-168-3-131.eu… HTTP 465
8547 211538281.77… 65.55.5.231 ip-192-168-3-131.eu… HTTP
8689 211538282.15… 65.55.5.232 ip-192-168-3-131.eu… HTTP
10006 211538308.02… 65.55.5.232 ip-192-168-3-131.eu… HTTP 1492
10021 211538308.07… 65.55.5.231 ip-192-168-3-131.eu… HTTP
10023 211538308.07… 65.55.239.163 ip-192-168-3-131.eu… HTTP
10050 211538308.19… 65.55.5.232 ip-192-168-3-131.eu… HTTP
10112 211538308.28… 65.55.5.232 ip-192-168-3-131.eu… HTTP 1481
10126 211538308.34… sn2ldc2.ac3.msn.com ip-192-168-3-131.eu… HTTP 75
10167 211538308.39… 65.55.5.231 ip-192-168-3-131.eu… HTTP 1403
10171 211538308.40… 65.54.95.66 ip-192-168-3-131.eu… HTTP 70
10251 211538308.57… sn2ldc2.ac3.msn.com ip-192-168-3-131.eu… HTTP 76
10306 211538308.60… 65.54.95.66 ip-192-168-3-131.eu… HTTP 148
Frame 10306: 148 bytes on wire (1184 bits), 148 bytes captured
Ethernet II, Src: Sophos_15:f9:80 (00:1a:8c:15:f9:80), Dst: Mi
Frame (148 bytes)

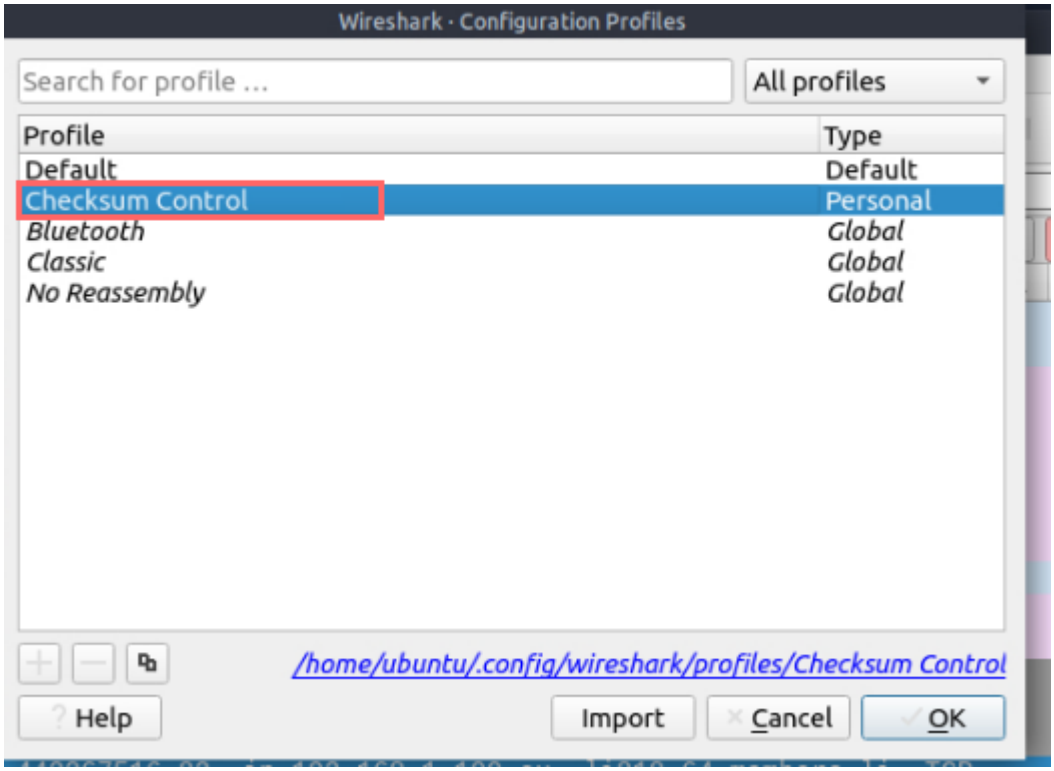
3)

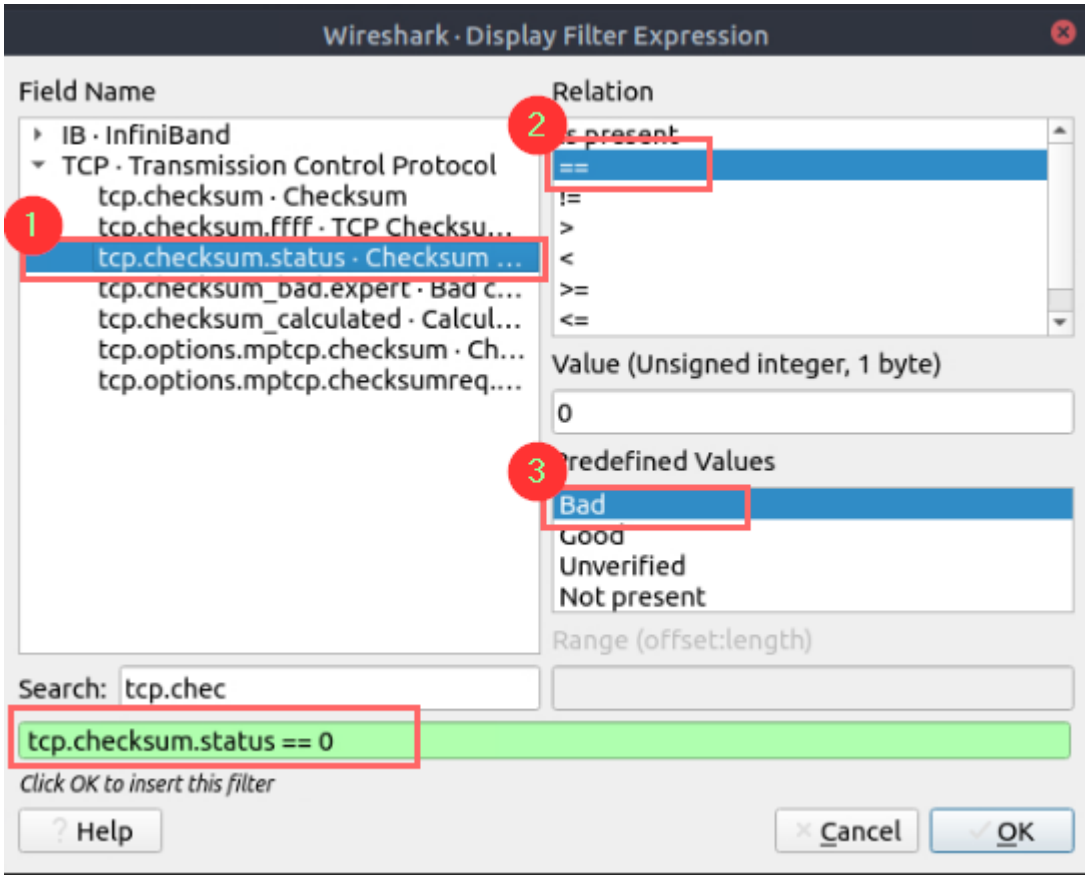

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.port in {3333 4444 9999}
Packet details
Narrow & Wide
Case sensitive
String
No. Time Source Destination Protocol Length
14344 442267516.82… ip-192-168-1-100.eu… li818-64.members.li… TCP
14345 442267516.82… ip-192-168-1-100.eu… 104.131.15.86 TCP
14348 442267516.82… ip-192-168-1-100.eu… 104.140.244.186 TCP
14353 442267516.82… ip-192-168-1-100.eu… 104.236.136.96 TCP
14354 442267516.82… ip-192-168-1-100.eu… 120.77.125.70 TCP
14356 442267516.82… ip-192-168-1-100.eu… 107.191.60.255.vult… TCP
14361 442267516.82… ip-192-168-1-100.eu… 107.191.99.227 TCP
14364 442267516.82… ip-192-168-1-100.eu… 113.21.199.11 TCP
14370 442267516.82… ip-192-168-1-100.eu… 119.254.102.118 TCP
14373 442267516.82… ip-192-168-1-100.eu… 119.254.102.180 TCP
14374 442267516.82… ip-192-168-1-100.eu… 120.25.128.33 TCP
14375 442267516.82… ip-192-168-1-100.eu… 120.55.226.251 TCP
14376 442267516.82… ip-192-168-1-100.eu… 104.236.57.24 TCP
14380 442267516.82… ip-192-168-1-100.eu… 121.43.77.145 TCP
14389 442267516.83… ip-192-168-1-100.eu… ns537146.ip-139-99-… TCP
Frame 14344: 66 bytes on wire (528 bits), 66 bytes captured (5
Ethernet II, Src: ip-192-168-1-100.eu-west-1.compute.internal
0000 d0 60
0010 00 34

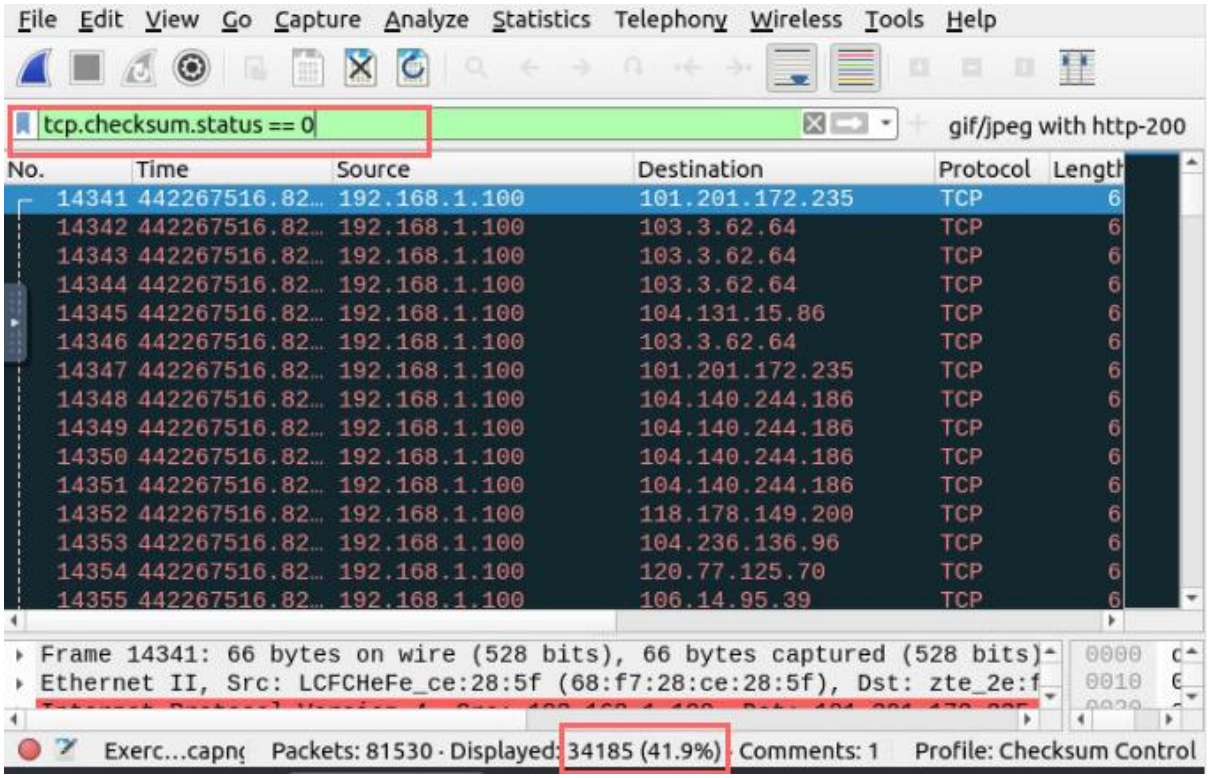
4) We used the TTL value earlier by looking at ip.ttl, but now we have to convert it to a string and see if it is an even number. This is done using the string and matches operators along with “[02468]$” to see if it ends in an even number. The filter will look like “string(ip.ttl) matches “[02468]$””.



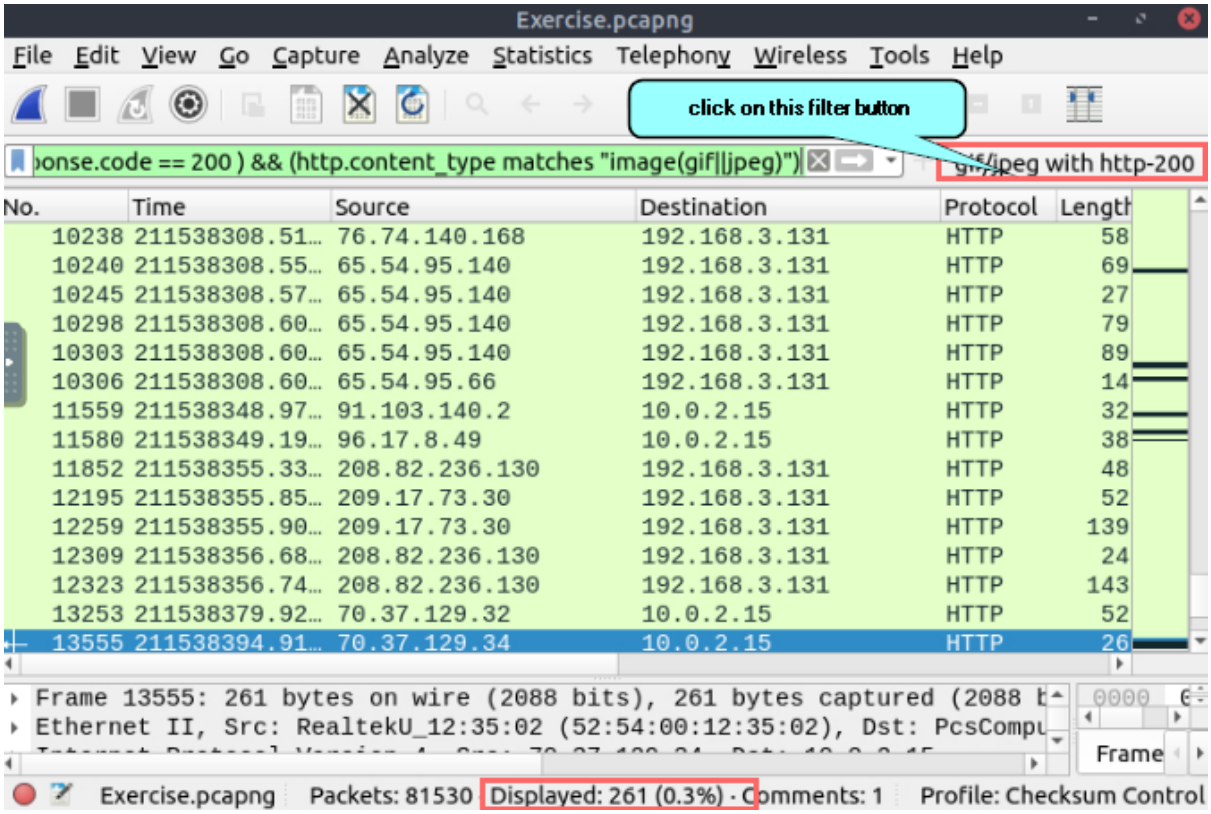
5)




Wireshark · Display Filter Expression
Field Name
Relation
IB · InfiniBand
TCP · Transmission Control Protocol
tcp.checksum · Checksum
tcp.checksum.ffff · TCP Checksu...
tcp.checksum.status · Checksum ...
tcp.checksum_bad.expert · Bad c...
tcp.checksum_calculated · Calcul...
tcp.options.mptcp.checksum · Ch...
tcp.options.mptcp.checksumreq....
==
!=
>
<
>=
<=
Value (Unsigned integer, 1 byte)
0
Predefined Values
Bad
Good
Unverified
Not present
Range (offset:length)
Search: tcp.chec
tcp.checksum.status == 0
Click OK to insert this filter
Help
Cancel
OK
1
2
3


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.checksum.status == 0
gif/jpeg with http-200
No. Time Source Destination Protocol Length
14341 442267516.82... 192.168.1.100 101.201.172.235 TCP 6
14342 442267516.82... 192.168.1.100 103.3.62.64 TCP 6
14343 442267516.82... 192.168.1.100 103.3.62.64 TCP 6
14344 442267516.82... 192.168.1.100 103.3.62.64 TCP 6
14345 442267516.82... 192.168.1.100 104.131.15.86 TCP 6
14346 442267516.82... 192.168.1.100 103.3.62.64 TCP 6
14347 442267516.82... 192.168.1.100 101.201.172.235 TCP 6
14348 442267516.82... 192.168.1.100 104.140.244.186 TCP 6
14349 442267516.82... 192.168.1.100 104.140.244.186 TCP 6
14350 442267516.82... 192.168.1.100 104.140.244.186 TCP 6
14351 442267516.82... 192.168.1.100 104.140.244.186 TCP 6
14352 442267516.82... 192.168.1.100 118.178.149.200 TCP 6
14353 442267516.82... 192.168.1.100 104.236.136.96 TCP 6
14354 442267516.82... 192.168.1.100 120.77.125.70 TCP 6
14355 442267516.82... 192.168.1.100 106.14.95.39 TCP 6
Frame 14341: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
Ethernet II, Src: LCFCHeFe_ce:28:5f (68:f7:28:ce:28:5f), Dst: zte_2e:f
Exerc...capng Packets: 81530 · Displayed: 34185 (41.9%) Comments: 1 Profile: Checksum Control

6)



That is all for this Write-up, hoping this will help you in solving the challenges of Wireshark: Packet Operations. Have Fun and Enjoy Hacking! Do visit other rooms and modules on TryHackMe for more learning.

-by Shefali Kumai

For more cyber security learning follow me here-

https://github.com/ctf-time

https://www.youtube.com/channel/UCf-F-eATCUXYaUVk8Xl7OOQ

https://www.instagram.com/cybersecurity.cyber_seek/

https://twitter.com/Shefali37920461