Practical - 09

Title: - Analyze the performance parameters of network using Wire Shark

Aim: - To analyze the performance parameters of a network using Wire Shark.

Lab Objectives: -

To analyze network traffic using network sniffing software.

Description: -

Filtering Packets While Viewing

Wireshark has two filtering languages: **capture filters and display filters**.

- **Capture filters are used for filtering when capturing packets.**
- **Display filters are used for filtering which packets are displayed.**

Display filters can be implemented to locate various types of data:

- Parameters such as the IP address, TCP or UDP port numbers, URLs, and server names
- Conditions such as "packet length shorter than..." and the TCP port range
- Phenomena such as TCP retransmissions, duplicate and other types of ACKs, various protocol error codes, and lag existence
- Various applications parameters such as Short Message Service (SMS) source and destination numbers and Server Message Block (SMB) server names

Any data that is sent over the network can be filtered, and when filtered, you can create statistics and graphs according to it.

In general, a display filter string takes the form of a series of primitive expressions connected by conjunctions (and, or, or something else) and optionally preceded by not:

[not] Expression [and|or] [not] Expression...

Display Filter comparison operators

English	Alias	C-like	Description	Example
eq	any_eq	==	Equal (any if more than one)	`ip.src == 10.0.0.5`
ne	all_ne	!=	Not equal (all if more than one)	`ip.src != 10.0.0.5`
	all_eq	===	Equal (all if more than one)	`ip.src === 10.0.0.5`
	any_ne	!==	Not equal (any if more than one)	`ip.src !== 10.0.0.5`
gt		>	Greater than	`frame.len > 10`
lt		<	Less than	`frame.len < 128`
ge		>=	Greater than or equal to	`frame.len ge 0x100`
le		<=	Less than or equal to	`frame.len <= 0x20`
contains			Protocol, field or slice contains a value	`sip.To contains "a1762"`
matches		~	Protocol or text field matches a Perl-compatible regular expression	`http.host matches "acme\\.(org|com|net)"`

Display Filter Logical Operations

English	C-like	Description	Example
and	&&	Logical AND	`ip.src==10.0.0.5 and tcp.flags.fin`
or	||	Logical OR	`ip.src==10.0.0.5 or ip.src==192.1.1.1`
xor	^^	Logical XOR	`tr.dst[0:3] == 0.6.29 xor tr.src[0:3] == 0.6.29`
not	!	Logical NOT	`not llc`
[...]		Subsequence	See “Slice Operator” below.
in		Set Membership	http.request.method in {"HEAD", "GET"}. See “Membership Operator” below.

The common layer 2 (Ethernet) filters are as follows:

- eth.addr == <MAC Address>: This is used to display a specific MAC address
- eth.src == <MAC Address>: This is used to get the source MAC address
- eth.dst == <MAC Address>: This is used to get the destination MAC address
- eth.type == <Protocol Type (Hexa)>: This is used to get the Ethernet protocol types

The common ARP filters are as follows:

- arp.opcode == <value>: This is used for ARP requests/responses
- arp.src.hw_mac == <MAC Address>: This is used to capture the ARP address of the sender

The common layer 3 (IP) filters are as follows:

- ip.addr == <IP Address>: This is used to get the source or destination IP address
- ip.src == <IP Address>: This is used to get the source IP address
- ip.dst == <IP Address>: This is used to get the destination IP address
- ip.ttl == <value>, ip.ttl < value>, or ip.ttl > <value>: This is used to get IP TTL (Time To Live) values
- ip.len = <value>, ip.len > <value>, or ip.len < <value>: This is used to get IP packet length values
- ip.version == <4/6>: This is used to get the IP protocol version (Version 4 or Version 6)

For TCP or UDP port numbers use the following display filters:

- tcp.port == <value> or udp.port == <value>: This is used for specific TCP or UDP ports (source or destination)
- tcp.dstport == <value> or udp.dstport == <value>: This is used for specific TCP or UDP destination ports
- tcp.srcport == <value> or udp.srcport == <value>: This is used for specific TCP or UDP destination ports

Wireshark Basic Statistics

Wireshark provides a wide range of network statistics which can be accessed via the Statistics menu.

The “Capture File Properties” Dialog - General information about the current capture file.

Resolved Addresses - The Resolved Addresses window shows the list of resolved addresses and their host names. Users can choose the Hosts field to display IPv4 and IPv6 addresses only. In this case, the dialog displays host names for each IP address in a capture file with a known host. This host is typically taken from DNS answers in a capture file. In case of an unknown host name, users can populate it based on a reverse DNS lookup. To do so, follow these steps:

- Enable Resolve Network Addresses in the View → Name Resolution menu as this option is disabled by default.
- Select Use an external network name resolver in the Preferences → Name Resolution menu. This option is enabled by default.

The "Protocol Hierarchy" Window - This is a tree of all the protocols in the capture. Each row contains the statistical values of one protocol. Two of the columns (Percent Packets and Percent Bytes) serve double duty as bar graphs. If a display filter is set it will be shown at the bottom.

Conversations - A network conversation is the traffic between two specific endpoints. For example, an IP conversation is all the traffic between two IP addresses. Along with addresses, packet counters, and byte counters the conversation window adds four columns: the start time of the conversation ("Rel Start") or ("Abs Start"), the duration of the conversation in seconds, and the average bits (not bytes) per second in each direction.

Endpoints - A network endpoint is the logical endpoint of separate protocol traffic of a specific protocol layer. For each supported protocol, a tab is shown in this window. Each tab label shows the number of endpoints captured (e.g., the tab label "Ethernet · 4" tells you that four ethernet endpoints have been captured).

The "I/O Graphs" Window - Lets you plot packet and protocol data in a variety of ways.

DNS - The DNS statistics window enlists a total count of DNS messages, which are divided into groups by request types (opcodes), response code (rcode), query type, and others.

Flow Graph - The Flow Graph window shows connections between hosts. It displays the packet time, direction, ports and comments for each captured connection. You can filter all connections by ICMP Flows, ICMPv6 Flows, UIM Flows and TCP Flows. Flow Graph window is used for showing multiple different topics. Based on it, it offers different controls.

HTTP Statistics

- HTTP Packet Counter - Statistics for HTTP request types and response codes.
- HTTP Requests - HTTP statistics based on the host and URI.
- HTTP Load Distribution - HTTP request and response statistics based on the server address and host.
- HTTP Request Sequences - HTTP Request Sequences uses HTTP's Referer and Location headers to sequence a capture's HTTP requests as a tree. This enables analysts to see how one HTTP request leads to the next.

TCP Stream Graphs - Show different visual representations of the TCP streams in a capture.

IPv4 Statistics - Internet Protocol version 4 (IPv4) is a core protocol for the internet layer. It uses 32-bit addresses and allows packets routing from one source host to the next one.

Following Protocol Streams

It can be very helpful to see a protocol in the way that the application layer sees it.

To filter to a particular stream, select a TCP, UDP, DCCP, TLS, HTTP, HTTP/2, QUIC or SIP packet in the packet list of the stream/connection you are interested in and then select the menu item Analyze → Follow → TCP Stream

The stream content is displayed in the same sequence as it appeared on the network. Non-printable characters are replaced by dots. Traffic from the client to the server is colored red, while traffic from the server to the client is colored blue.

Packet colorization

A very useful mechanism available in Wireshark is packet colorization. You can set up Wireshark so that it will colorize packets according to a display filter. This allows you to emphasize the packets you might be interested in.

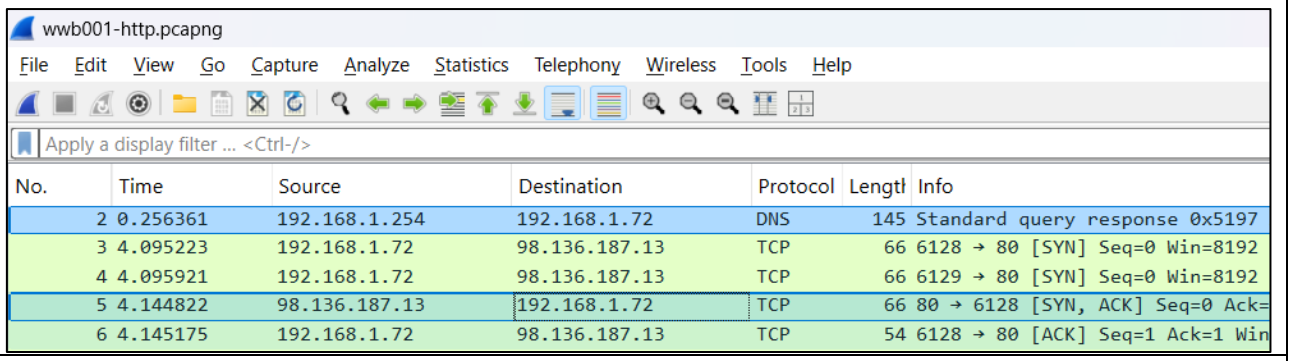
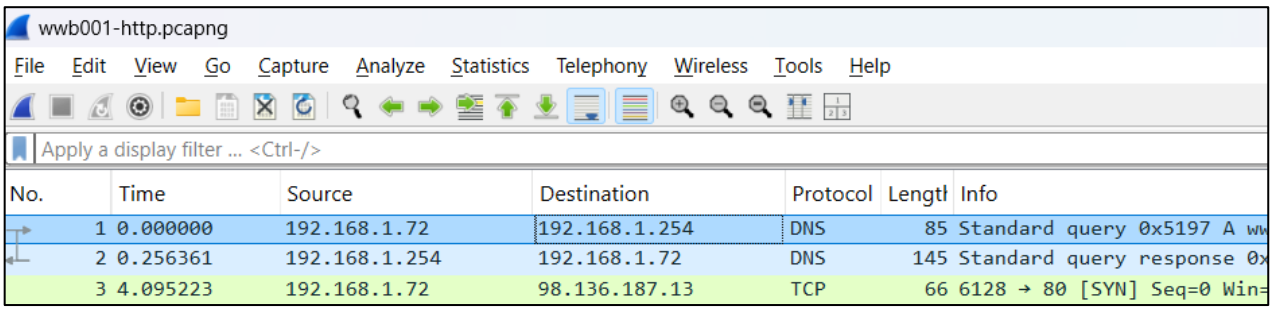
There are two types of coloring rules in Wireshark: **temporary rules** that are only in effect until you quit the program, and **permanent rules** that are saved in a preference file so that they are available the next time you run Wireshark.

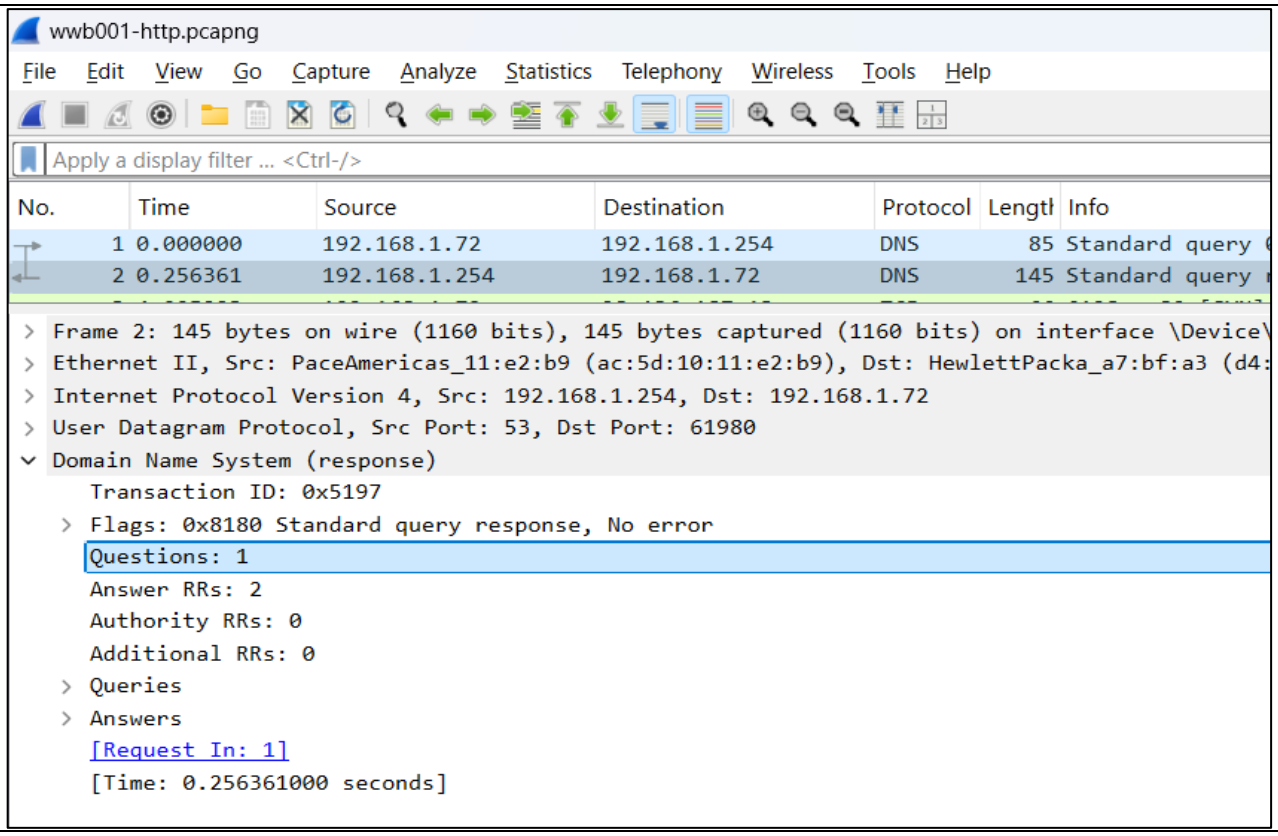
Temporary rules can be added by selecting a packet and pressing the Ctrl key together with one of the number keys. This will create a coloring rule based on the currently selected conversation. It will try to create a conversation filter based on TCP first, then UDP, then IP and at last Ethernet. Temporary filters can also be created by selecting the Colorize with Filter → Color X menu items when right-clicking in the packet detail pane.

To permanently colorize packets, select View → Coloring Rules….

Exercise

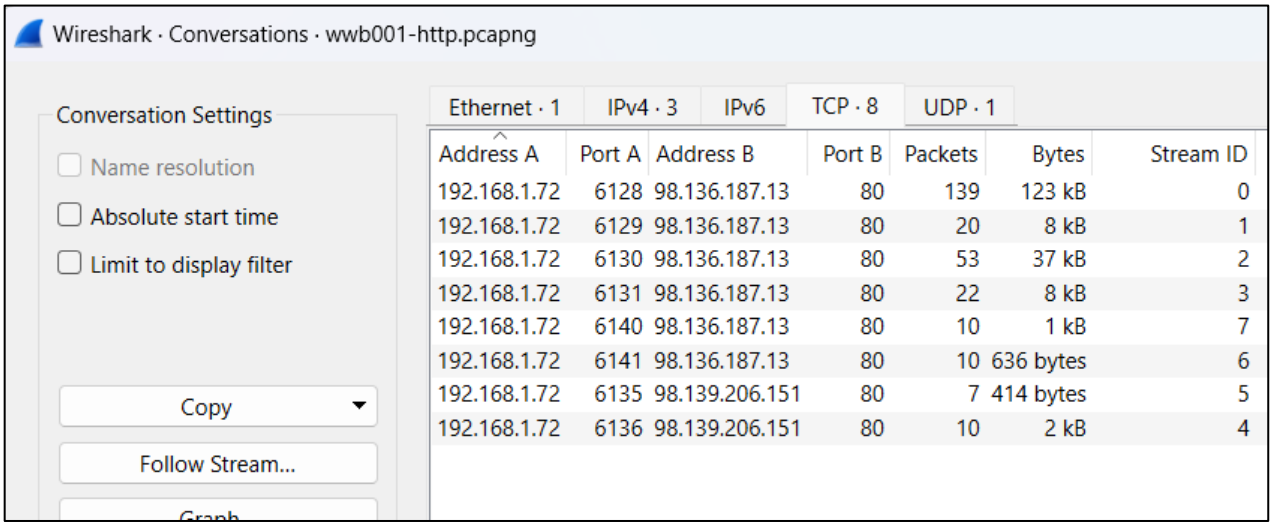
1. Use the trace file wwb001-http.pcapng and answer the following questions. Support your answers with appropriate screenshot.

1.	What is the IP address of the DNS/HTTP client in this trace file?
Sol	Frame 1 is a DNS request sent from 192.168.1.72. Frame 3 is a TCP SYN packet sent from 192.168.1.72 to an HTTP server at port 80. That is the IP address of the DNS/HTTP client in this trace file. You could also look for GET requests to see the IP address of the HTTP client in this trace file. 
2.	What is the IP address of the DNS server?
Sol	Frame 1 is a DNS packet addressed to 192.168.1.254. If the DNS traffic did not occur right at the top of the trace file, you could apply a display filter for dns to find the IP address of the DNS server 
3.	What DNS response time is seen in this trace file?
Sol	You can look at the *Time* column for this value since the DNS request and response packets are the first two packets of the trace file. The *Time* column indicates the DNS response arrived 0.256361 seconds (just over 256 ms) after the DNS request. Alternately, you can expand the DNS section of the DNS response packet and look for the *[Time:]* value. This is Wireshark's measurement from the related DNS request to this DNS response in the trace file.



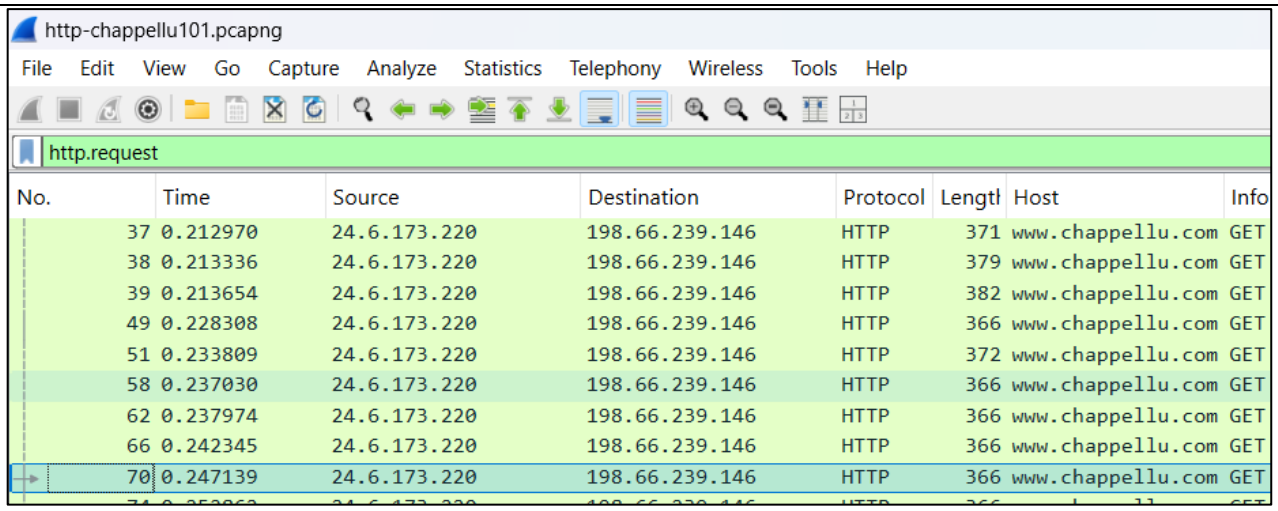
4. What are the IP addresses of the HTTP servers to which the client successfully connected?

Sol There are many ways to obtain this information. You can select *Statistics | Conversations* and look under the *TCP* tab. For clarity, sort the *Address B* column and you can clearly see the two HTTP servers listed with port 80.



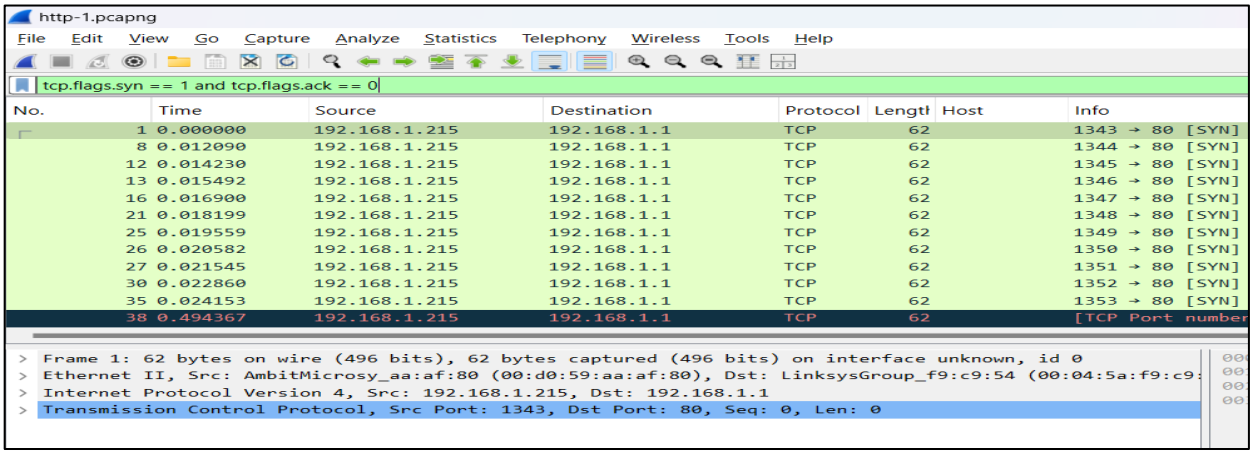
5. What are the HTTP host names of the target HTTP servers?

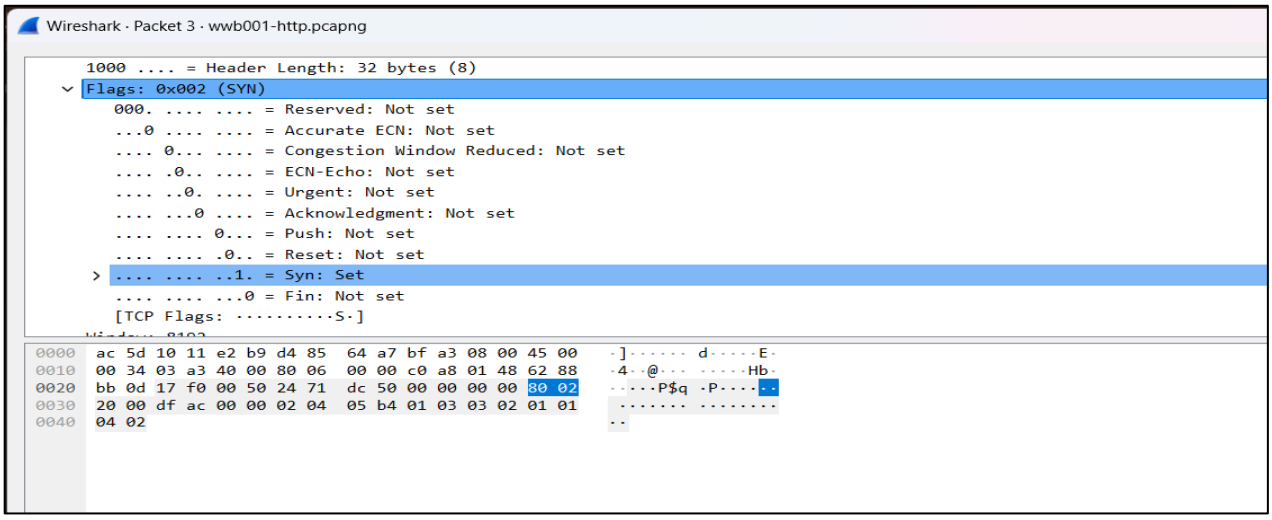
Sol HTTP requests contain a *Host* field (http.host). Although you could scroll through the packets to look for this field separately in each HTTP request, there is almost always a better way to locate something than scrolling.



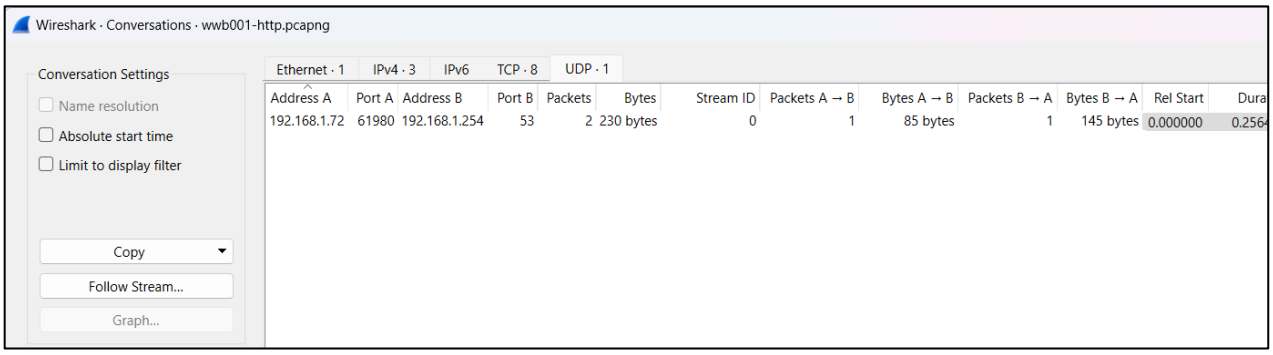
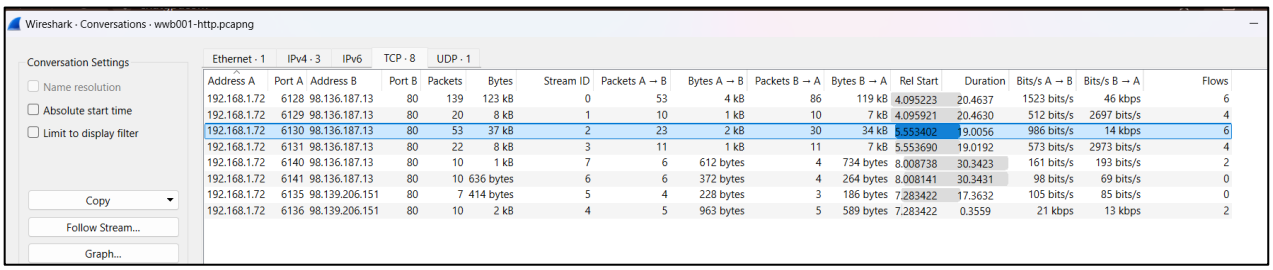
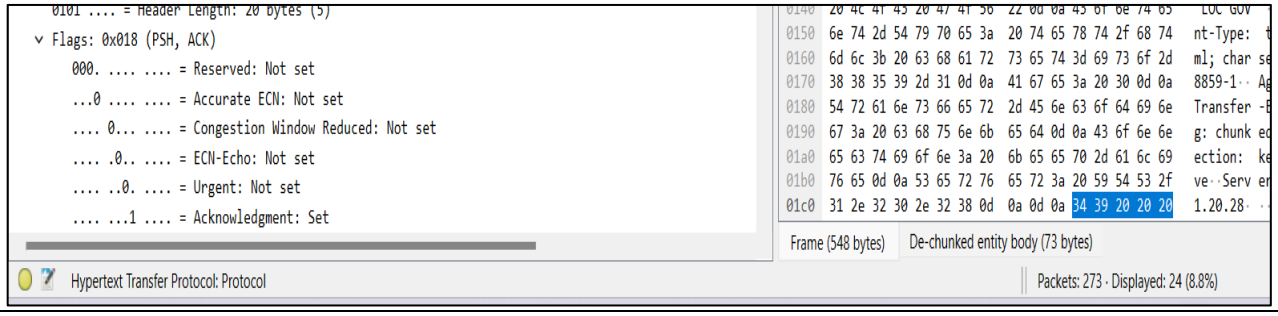
6. How many TCP SYN packets did the client send to the HTTP servers? Display Flow Graph limited to display filter.

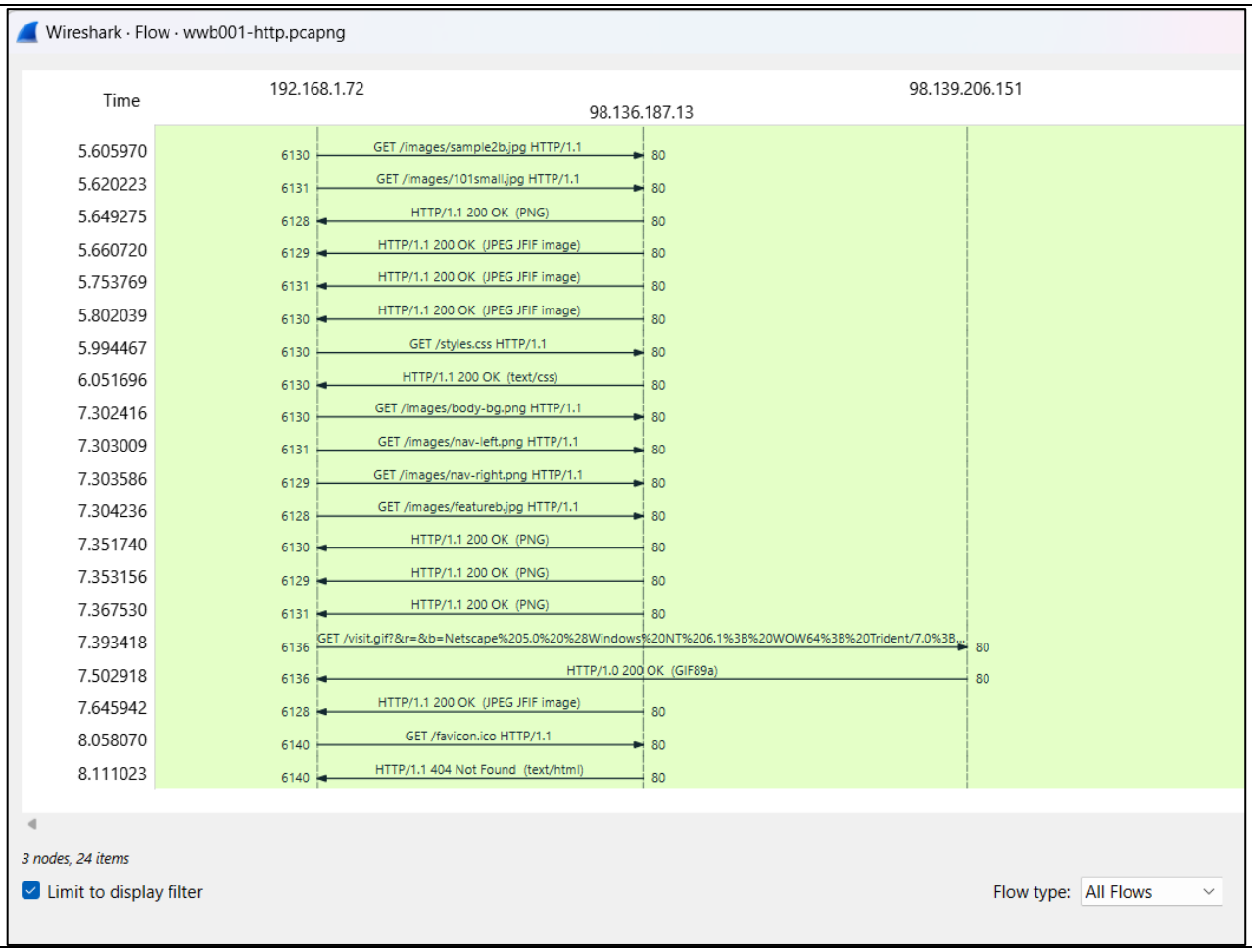
Sol Note that we are not counting SYN/ACK packets, just SYN packets. So, let's consider creating a filter for what we really want to see; packets that only have the SYN bit set and not the ACK bit.
tcp.flags.syn**==1 &&** tcp.flags.ack==0





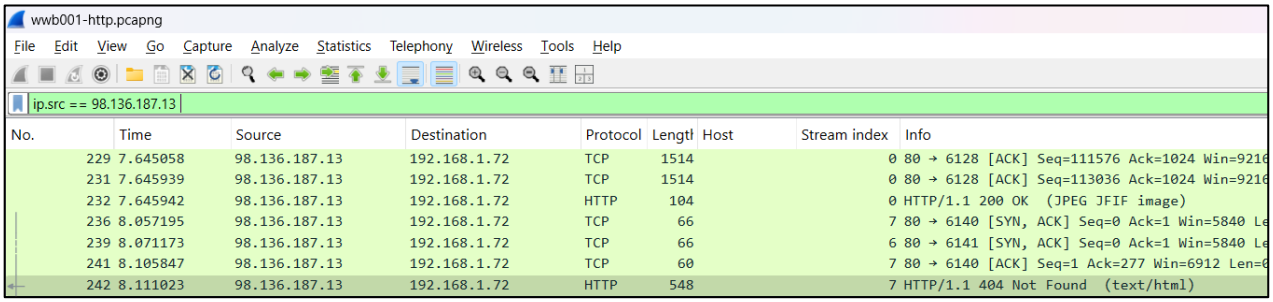
7. How many UDP streams are in this trace file? Follow UDP Stream.

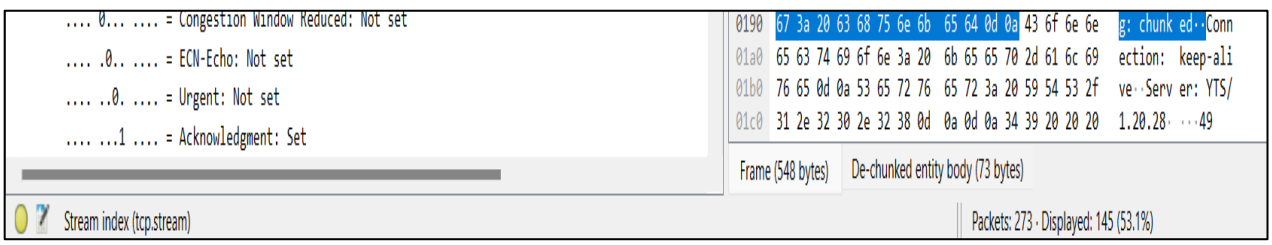
Sol	You can use the *Statistics	Conversations* window to identify the number of UDP and TCP streams in a trace file. 
8.	How many TCP streams are in this trace file? Follow TCP Stream 2.
Sol	You can use the *Statistics	Conversations* window to identify the number of TCP streams in a trace file. 
9.	What is total number of http packets in this trace file? Show its flow graph.
Sol	To determine the total number of HTTP packets in the trace file, we use a display filter that isolates only HTTP-related traffic. Once filtered, Wireshark's status bar shows the count of matching packets. After that, we generate a flow graph to visualize the communication between client(s) and server(s). 



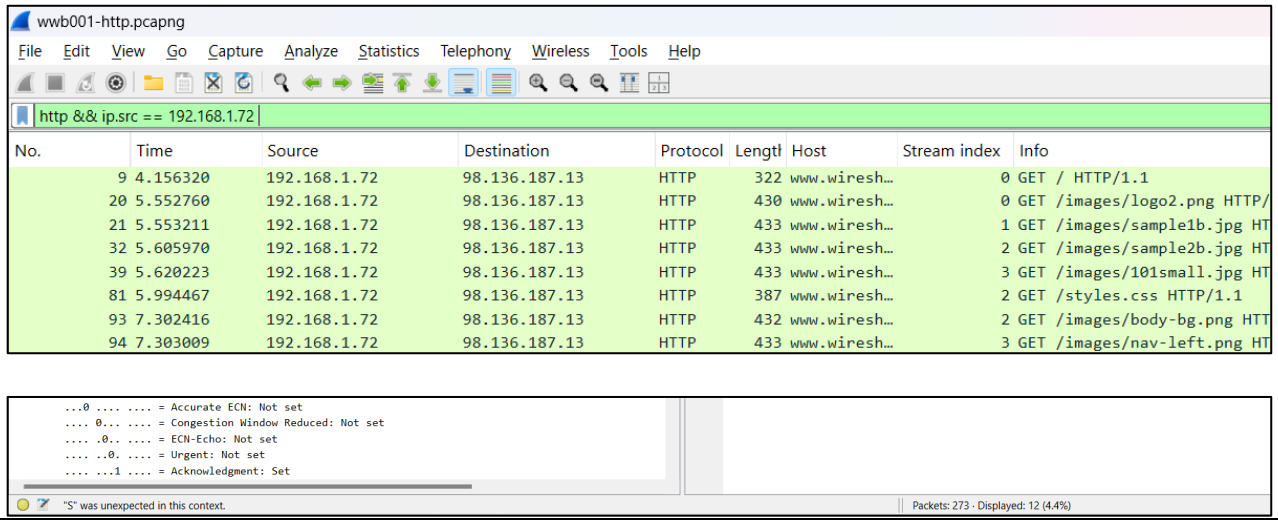
10 Display packets whose source ip address is 98.136.187.13

Sol To filter packets based on a specific source IP address in Wireshark, we use a display filter that matches the ip.src field.





11.	Display http packets received from 192.168.1.72
Sol	To display only HTTP packets **sent by** the IP address 192.168.1.72 (meaning they were **received by** another host), we use a display filter combining the http protocol and the ip.src field.



Conclusion: Analyzed the traffic on network using Wireshark.

References

https://www.wireshark.org/docs/wsug_html_chunked/index.html