Winter – SEMESTER 2022 - 23 **Name:** Nidhi Singh
Course Code: MCSE505P **Reg. No:**22MAI0015
Course-Title: – computer Network Lab
DIGITAL ASSIGNMENT - 2
(LAB) **Slot-**L35+L36

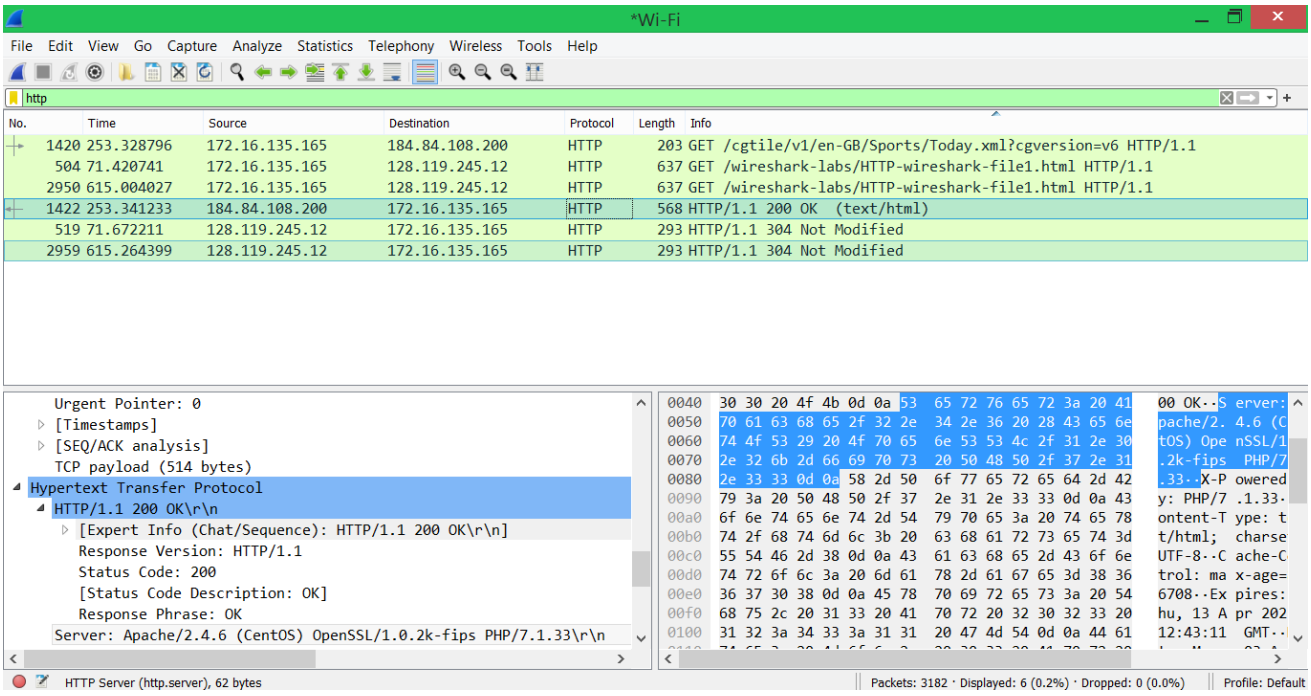
Faculty: - SRIMATHI C - SCOPE

HTTP :-

1. The Basic HTTP GET/response interaction

1. Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server running?

Answer :-
Both of them are version 1.1 (HTTP version information is listed in the item 'Request Version')

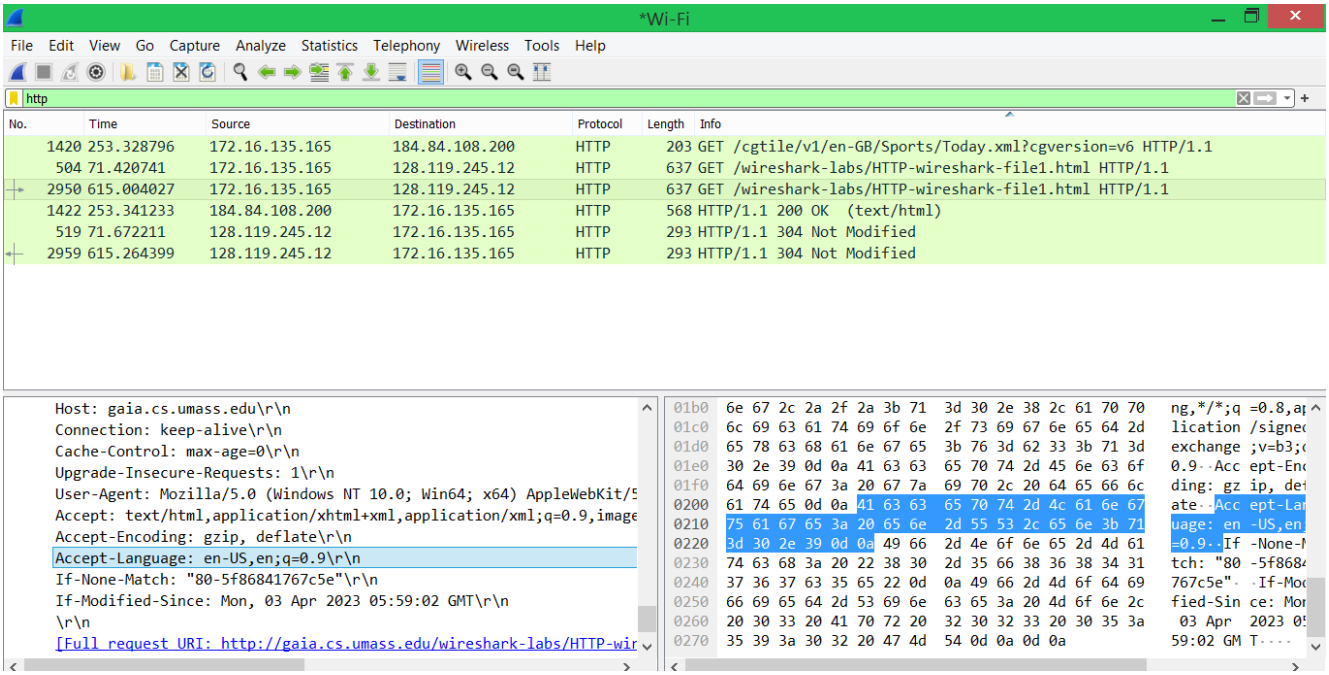


2. What languages (if any) does your browser indicate that it can accept to the server?

Answer :-

en-US

(languages information is listed in the item 'Accept-Language' in the HTTP GET message)

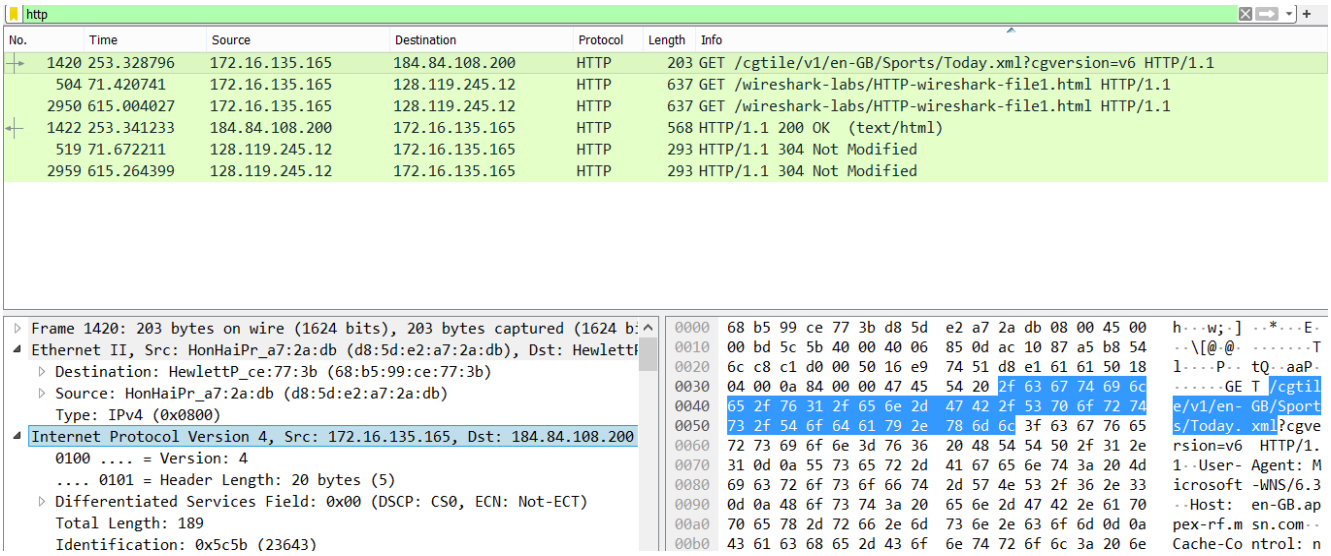


3. What is the IP address of your computer? Of the gaia.cs.umass.edu server?

Answer :-

my computer: 172.16.135.165

gaia.cs.umass.edu: 184.84.108.200

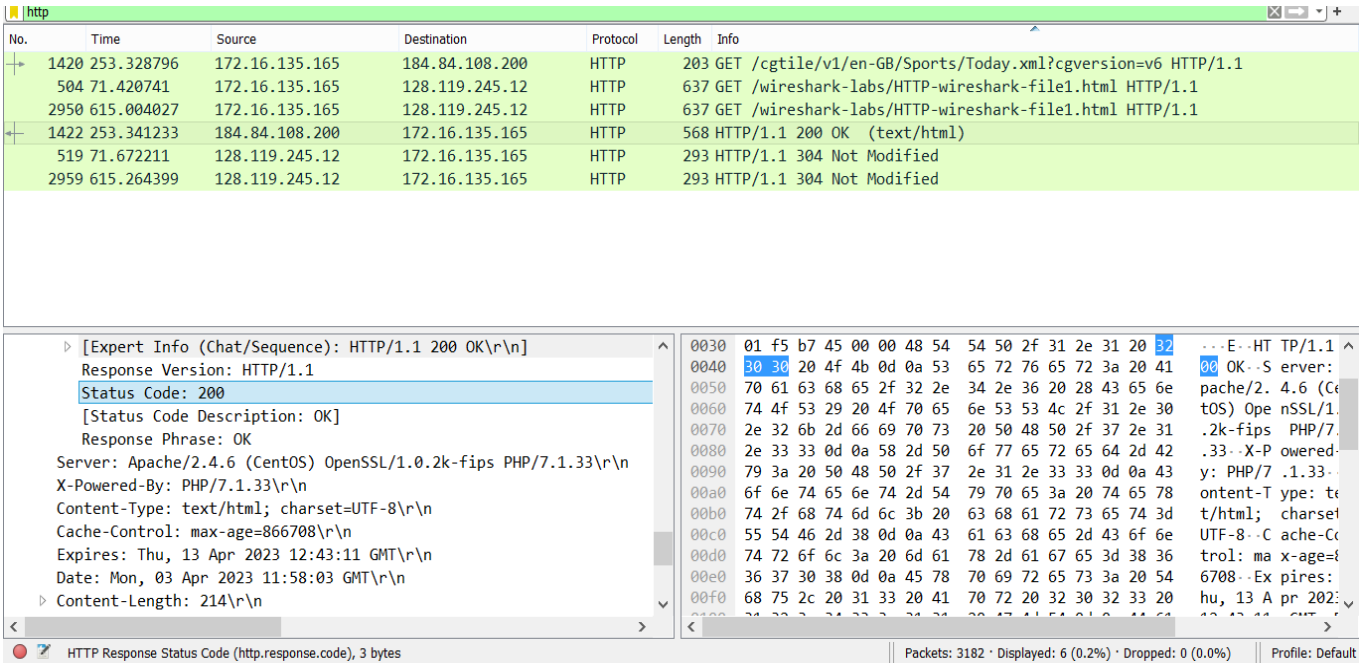


4. What is the status code returned from the server to your browser?

Answer :-

status code:200

(status code information is listed in the HTTP OK message)

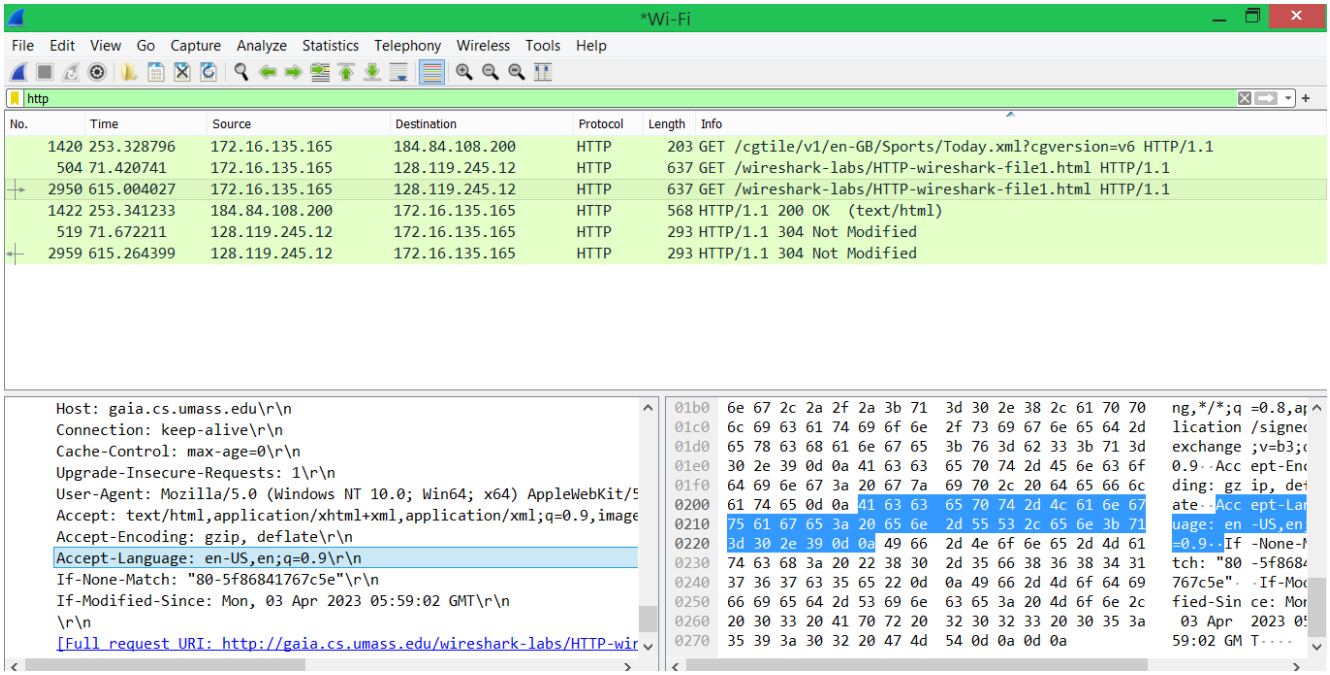


5. When was the HTML file that you are retrieving last modified at the server?

Answer :-

Mon, 03 Apr 2023 05:59:02 GMT\r\n

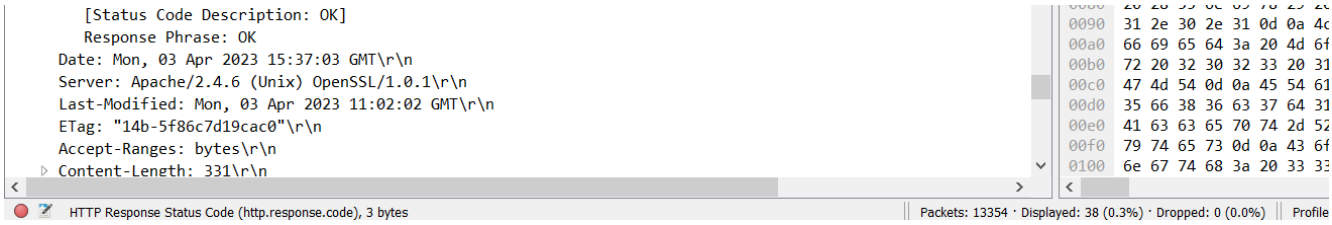
(last modified information is listed in the item 'Last-Modified' in the HTTP OK message)



6. How many bytes of content are being returned to your browser?

Answer :-

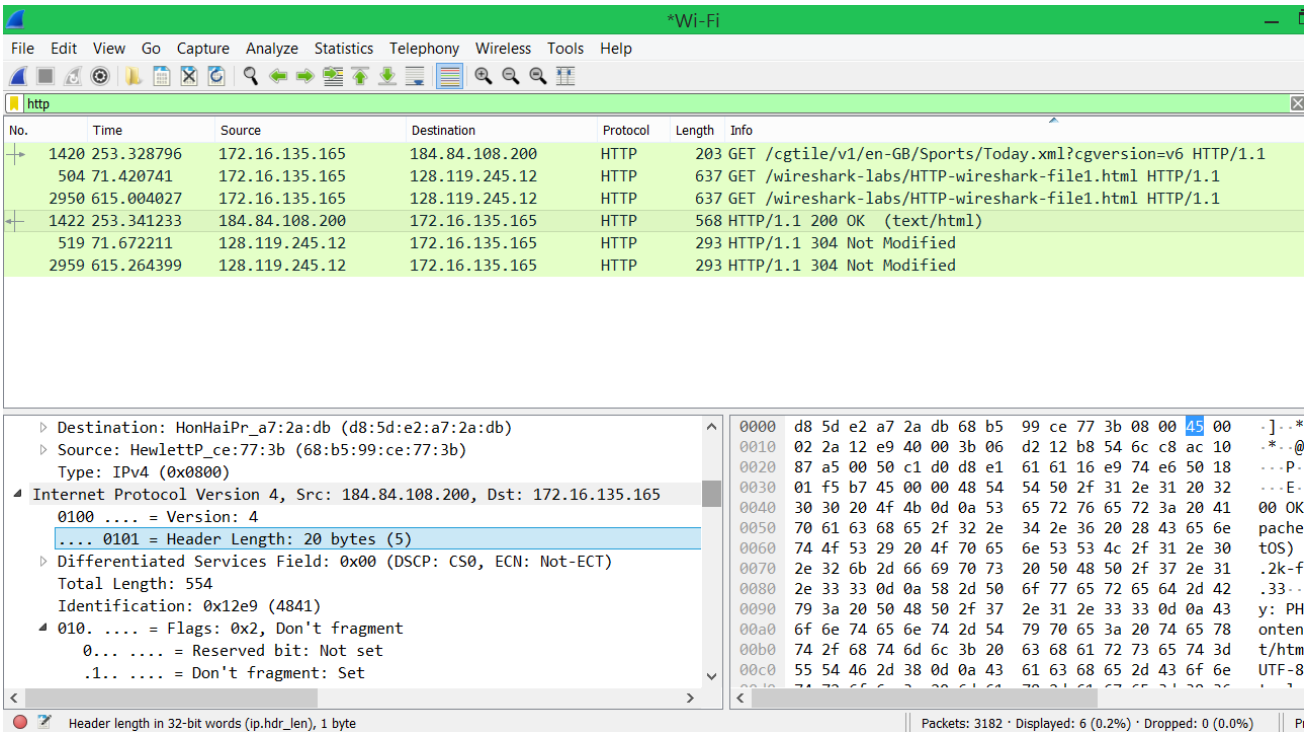
Content length: 331 (Content length information is listed in the item 'Content-Length' in the HTTP OK message)



7. By inspecting the raw data in the packet content window, do you see any headers within the data that are not displayed in the packet-listing window? If so, name one.

Answer :-

No, there is no more headers below.

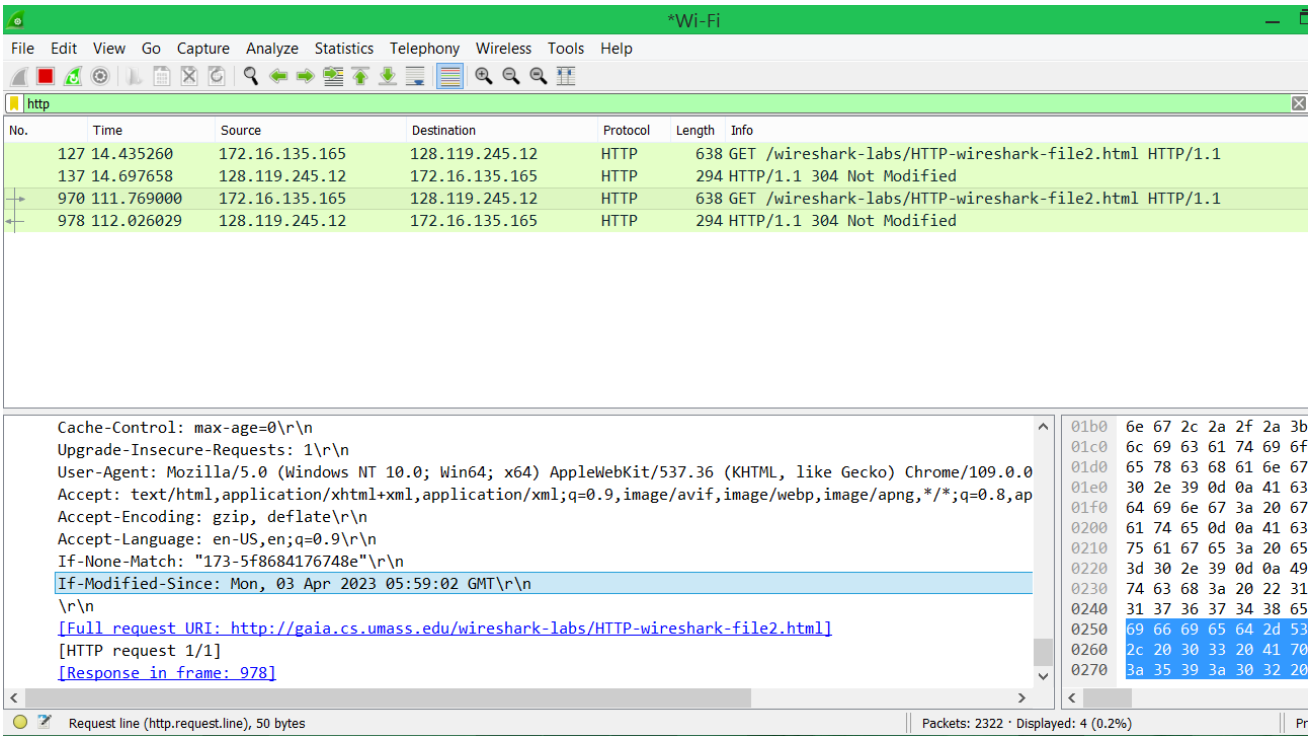


2. The HTTP CONDITIONAL GET/response interaction

8. Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE" line in the HTTP GET?

Answer :-

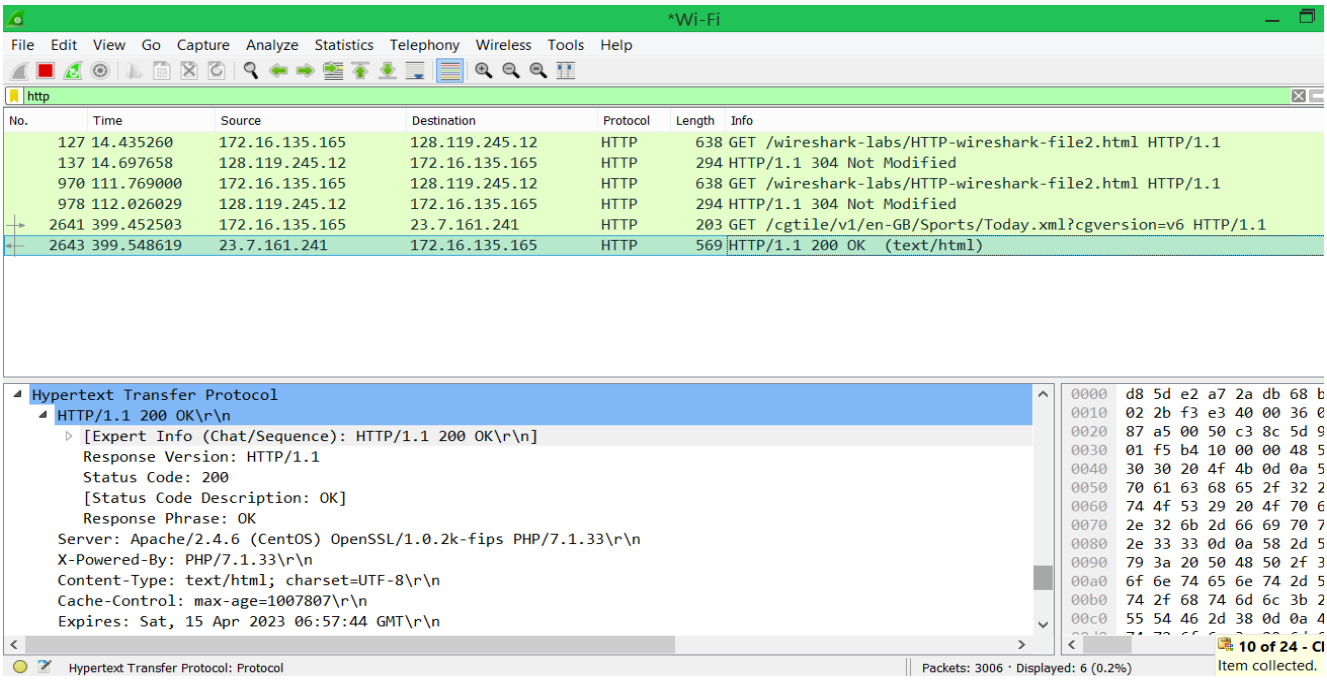
According to the bellow Diagram , there is no IF-MODIFIED-SINCE line in the first HTTP GET, but according to the Diagram bellow, IF-MODIFIED-SINCE is found in the second HTTP GET (the web page cached locally, asking the server side, the local cache need to be updated or not).



9. Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?

Answer :-

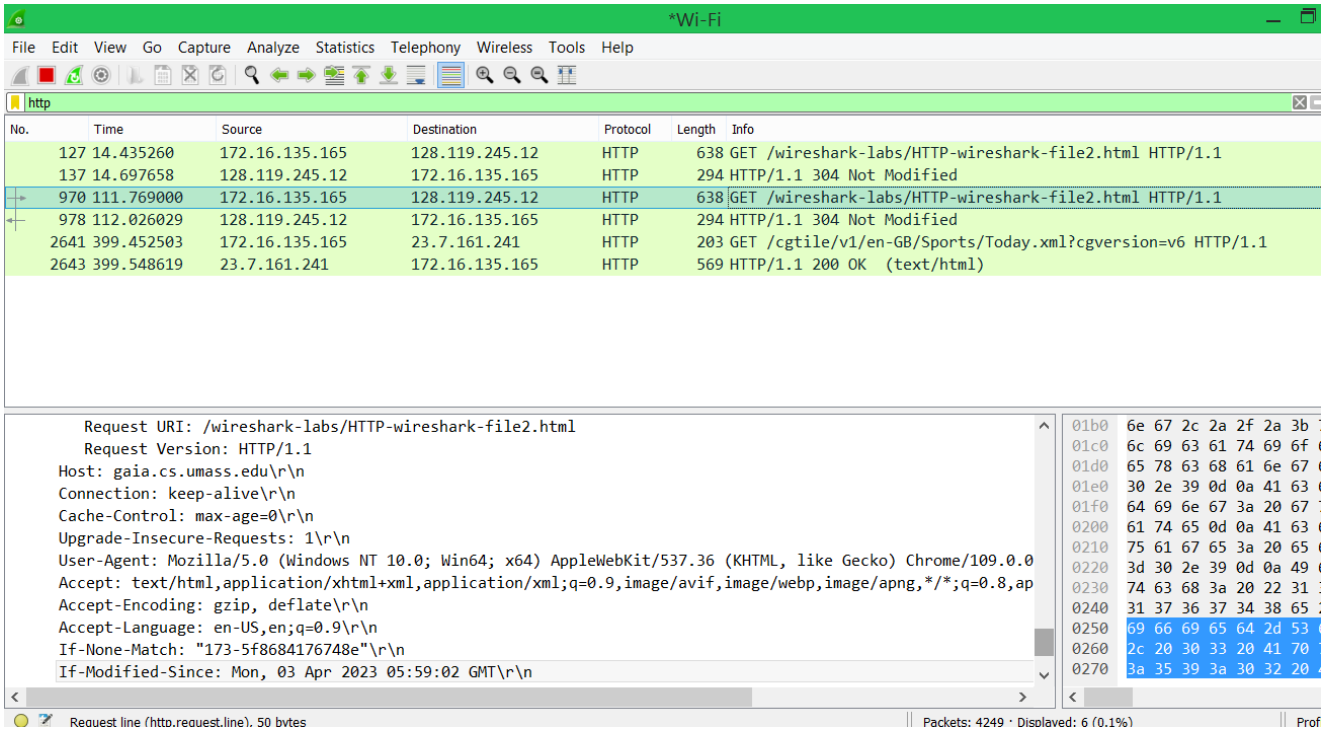
According to bellow Diagram, the server explicitly return the contents of the file, but according to the Diagram 6, the server did not explicitly return the contents of the file since the file had not been modified.



10. Now inspect the contents of the second HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE:" line in the HTTP GET? If so, what information follows the "IF-MODIFIED-SINCE:" header?

Answer :-

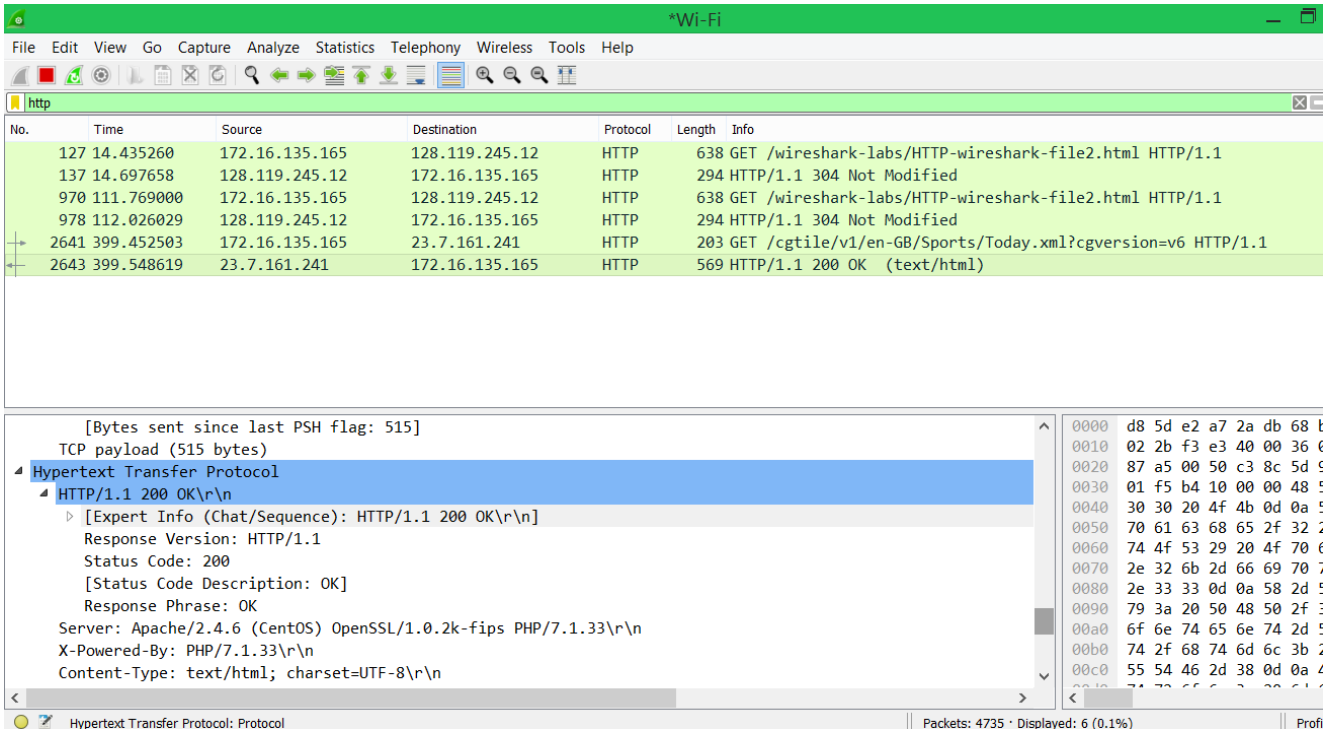
According to the Diagram , IF-MODIFIED-SINCE is found in the second HTTP GET (the web page cached locally, asking the server side, the local cache need to be updated or not). According to the Diagram , the server returning a 304 not modified follows the “IF-MODIFIED- SINCE:” header.



11. What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.

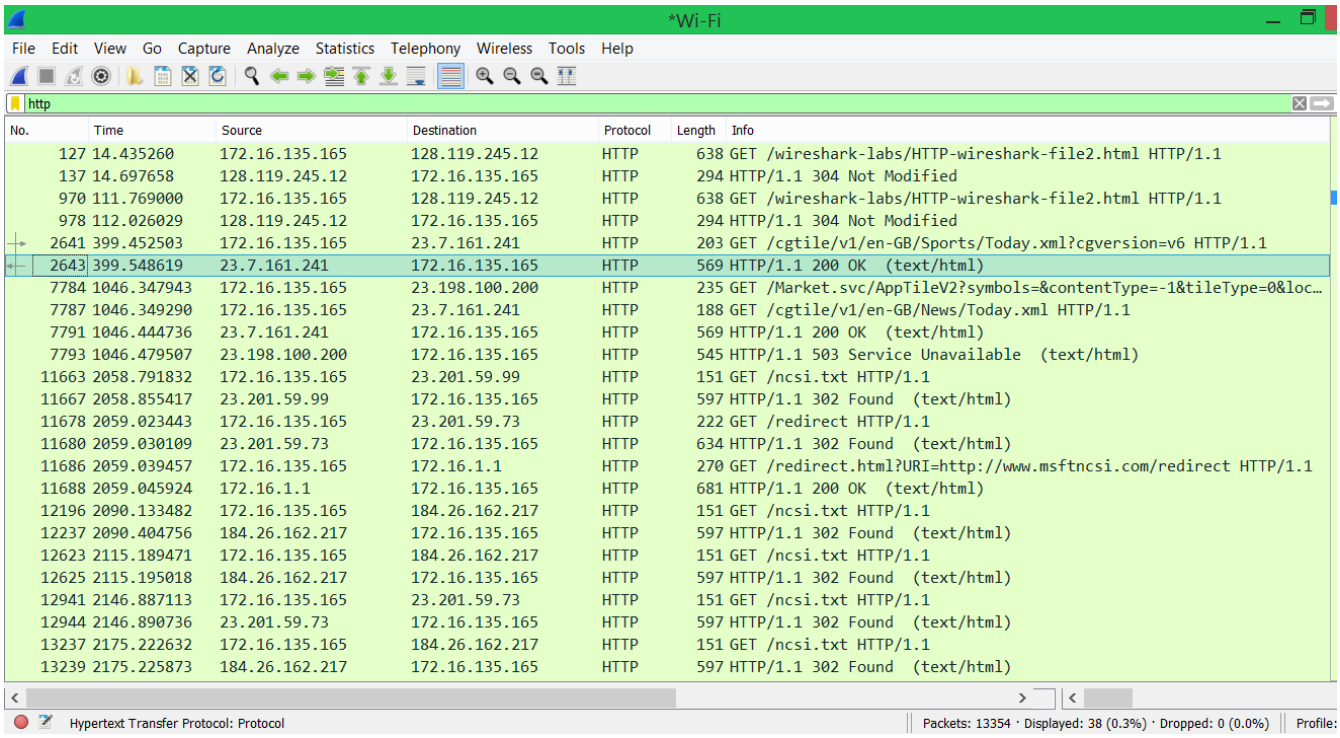
Answer :-

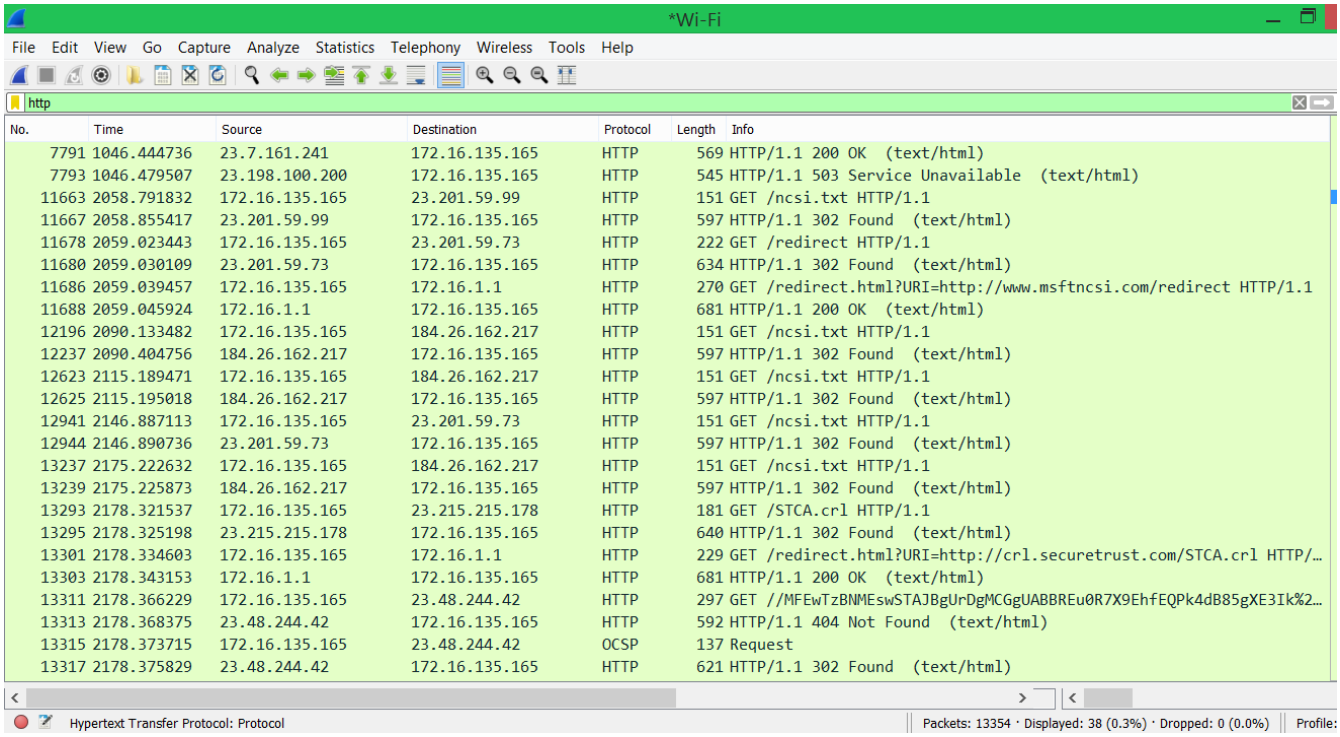
According to the Diagram. The HTTP status code is 304 Not Modified and the server did not explicitly return the contents of the file since the file was cached locally.

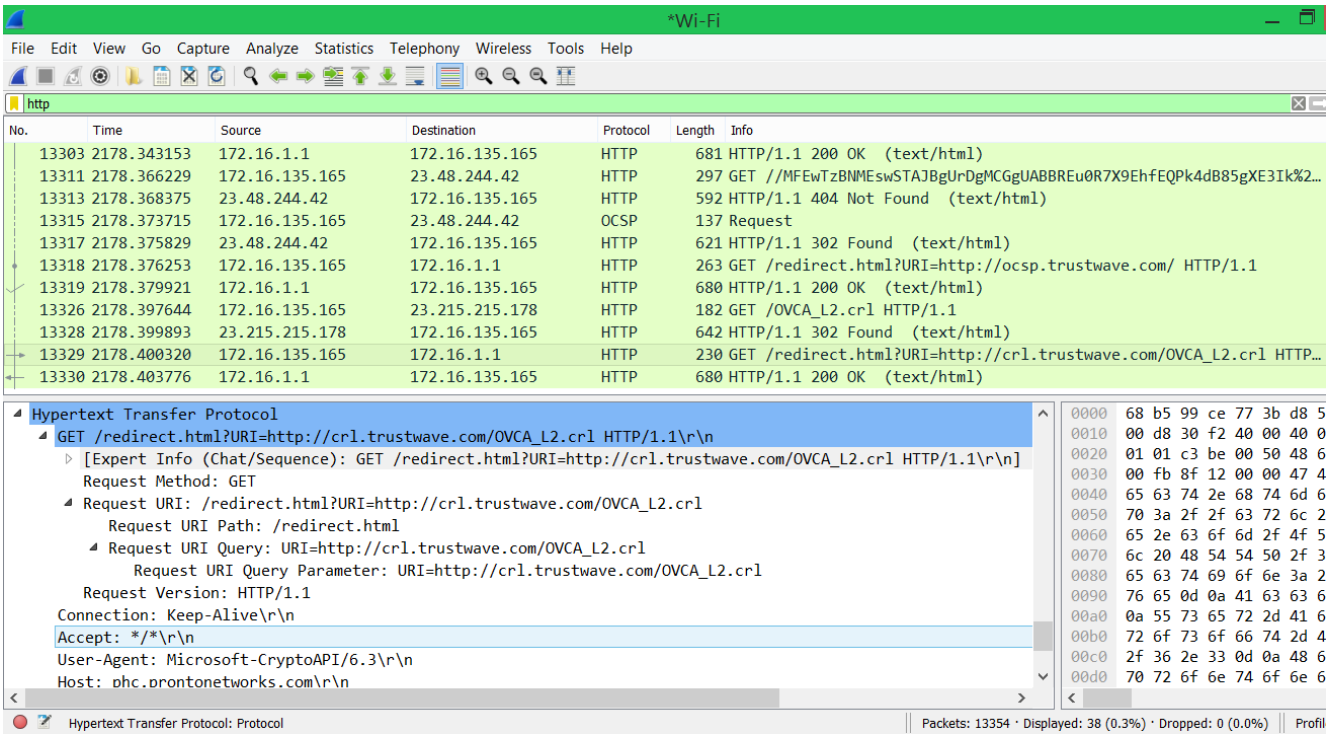


3. Retrieving Long Documents

12. How many HTTP GET request messages did your browser send? Which packet number in the trace contains the GET message for the Bill or Rights?


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
No. Time Source Destination Protocol Length Info
127 14.435260 172.16.135.165 128.119.245.12 HTTP 638 GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
137 14.697658 128.119.245.12 172.16.135.165 HTTP 294 HTTP/1.1 304 Not Modified
970 111.769000 172.16.135.165 128.119.245.12 HTTP 638 GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
978 112.026029 128.119.245.12 172.16.135.165 HTTP 294 HTTP/1.1 304 Not Modified
2641 399.452503 172.16.135.165 23.7.161.241 HTTP 203 GET /cgtile/v1/en-GB/Sports/Today.xml?cgversion=v6 HTTP/1.1
2643 399.548619 23.7.161.241 172.16.135.165 HTTP 569 HTTP/1.1 200 OK (text/html)
7784 1046.347943 172.16.135.165 23.198.100.200 HTTP 235 GET /Market.svc/AppTileV2?symbols=&contentType=-1&tileType=0&loc…
7787 1046.349290 172.16.135.165 23.7.161.241 HTTP 188 GET /cgtile/v1/en-GB/News/Today.xml HTTP/1.1
7791 1046.444736 23.7.161.241 172.16.135.165 HTTP 569 HTTP/1.1 200 OK (text/html)
7793 1046.479507 23.198.100.200 172.16.135.165 HTTP 545 HTTP/1.1 503 Service Unavailable (text/html)
11663 2058.791832 172.16.135.165 23.201.59.99 HTTP 151 GET /ncsi.txt HTTP/1.1
11667 2058.855417 23.201.59.99 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
11678 2059.023443 172.16.135.165 23.201.59.73 HTTP 222 GET /redirect HTTP/1.1
11680 2059.030109 23.201.59.73 172.16.135.165 HTTP 634 HTTP/1.1 302 Found (text/html)
11686 2059.039457 172.16.135.165 172.16.1.1 HTTP 270 GET /redirect.html?URI=http://www.msftncsi.com/redirect HTTP/1.1
11688 2059.045924 172.16.1.1 172.16.135.165 HTTP 681 HTTP/1.1 200 OK (text/html)
12196 2090.133482 172.16.135.165 184.26.162.217 HTTP 151 GET /ncsi.txt HTTP/1.1
12237 2090.404756 184.26.162.217 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
12623 2115.189471 172.16.135.165 184.26.162.217 HTTP 151 GET /ncsi.txt HTTP/1.1
12625 2115.195018 184.26.162.217 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
12941 2146.887113 172.16.135.165 23.201.59.73 HTTP 151 GET /ncsi.txt HTTP/1.1
12944 2146.890736 23.201.59.73 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
13237 2175.222632 172.16.135.165 184.26.162.217 HTTP 151 GET /ncsi.txt HTTP/1.1
13239 2175.225873 184.26.162.217 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
Hypertext Transfer Protocol: Protocol
Packets: 13354 · Displayed: 38 (0.3%) · Dropped: 0 (0.0%)
Profile:

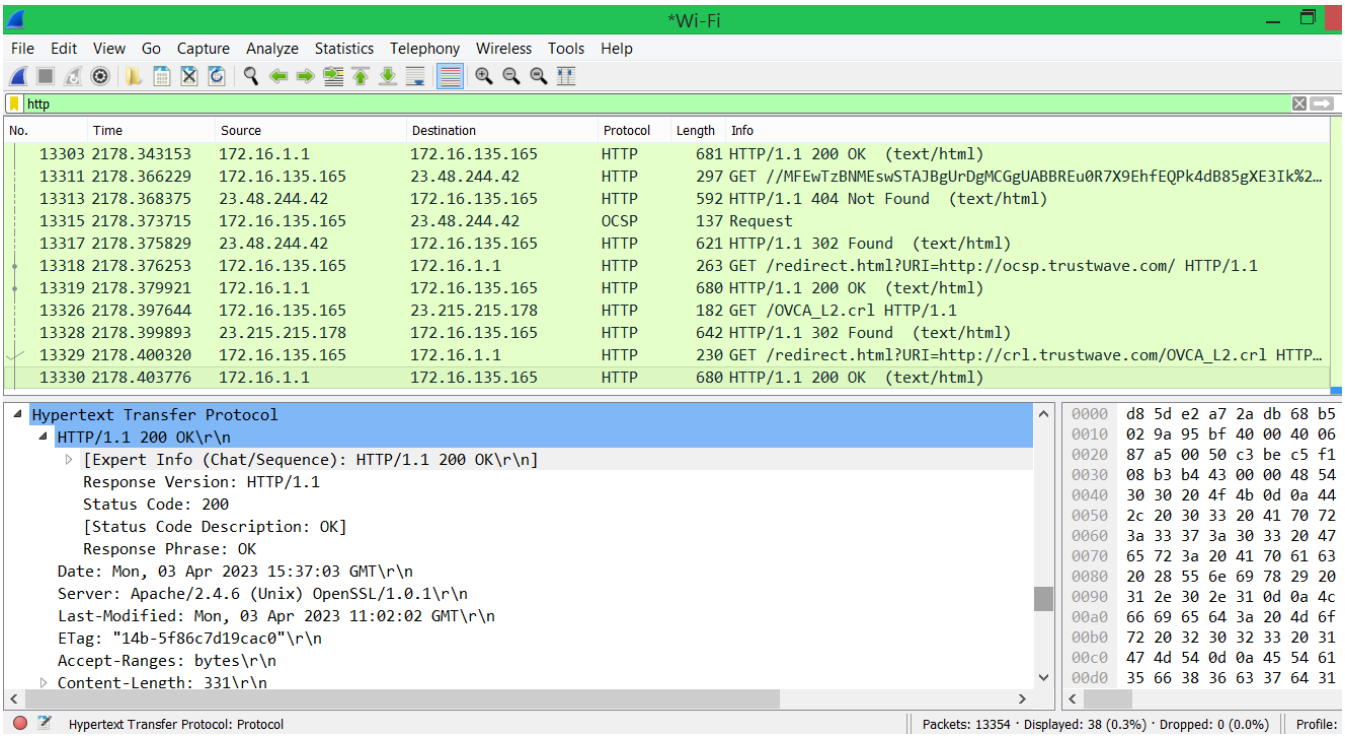

*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
No. Time Source Destination Protocol Length Info
7791 1046.444736 23.7.161.241 172.16.135.165 HTTP 569 HTTP/1.1 200 OK (text/html)
7793 1046.479507 23.198.100.200 172.16.135.165 HTTP 545 HTTP/1.1 503 Service Unavailable (text/html)
11663 2058.791832 172.16.135.165 23.201.59.99 HTTP 151 GET /ncsi.txt HTTP/1.1
11667 2058.855417 23.201.59.99 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
11678 2059.023443 172.16.135.165 23.201.59.73 HTTP 222 GET /redirect HTTP/1.1
11680 2059.030109 23.201.59.73 172.16.135.165 HTTP 634 HTTP/1.1 302 Found (text/html)
11686 2059.039457 172.16.135.165 172.16.1.1 HTTP 270 GET /redirect.html?URI=http://www.msftncsi.com/redirect HTTP/1.1
11688 2059.045924 172.16.1.1 172.16.135.165 HTTP 681 HTTP/1.1 200 OK (text/html)
12196 2090.133482 172.16.135.165 184.26.162.217 HTTP 151 GET /ncsi.txt HTTP/1.1
12237 2090.404756 184.26.162.217 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
12623 2115.189471 172.16.135.165 184.26.162.217 HTTP 151 GET /ncsi.txt HTTP/1.1
12625 2115.195018 184.26.162.217 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
12941 2146.887113 172.16.135.165 23.201.59.73 HTTP 151 GET /ncsi.txt HTTP/1.1
12944 2146.890736 23.201.59.73 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
13237 2175.222632 172.16.135.165 184.26.162.217 HTTP 151 GET /ncsi.txt HTTP/1.1
13239 2175.225873 184.26.162.217 172.16.135.165 HTTP 597 HTTP/1.1 302 Found (text/html)
13293 2178.321537 172.16.135.165 23.215.215.178 HTTP 181 GET /STCA.crl HTTP/1.1
13295 2178.325198 23.215.215.178 172.16.135.165 HTTP 640 HTTP/1.1 302 Found (text/html)
13301 2178.334603 172.16.135.165 172.16.1.1 HTTP 229 GET /redirect.html?URI=http://crl.securetrust.com/STCA.crl HTTP/…
13303 2178.343153 172.16.1.1 172.16.135.165 HTTP 681 HTTP/1.1 200 OK (text/html)
13311 2178.366229 172.16.135.165 23.48.244.42 HTTP 297 GET //MFEwTzBNMEswSTAJBgUrDgMCGgUABBREu0R7X9EhfEQPk4dB85gXE3Ik%2…
13313 2178.368375 23.48.244.42 172.16.135.165 HTTP 592 HTTP/1.1 404 Not Found (text/html)
13315 2178.373715 172.16.135.165 23.48.244.42 OCSP 137 Request
13317 2178.375829 23.48.244.42 172.16.135.165 HTTP 621 HTTP/1.1 302 Found (text/html)
Hypertext Transfer Protocol: Protocol
Packets: 13354 · Displayed: 38 (0.3%) · Dropped: 0 (0.0%)
Profile:



13. Which packet number in the trace contains the status code and phrase associated with the response to the HTTP GET request?

Answer :-

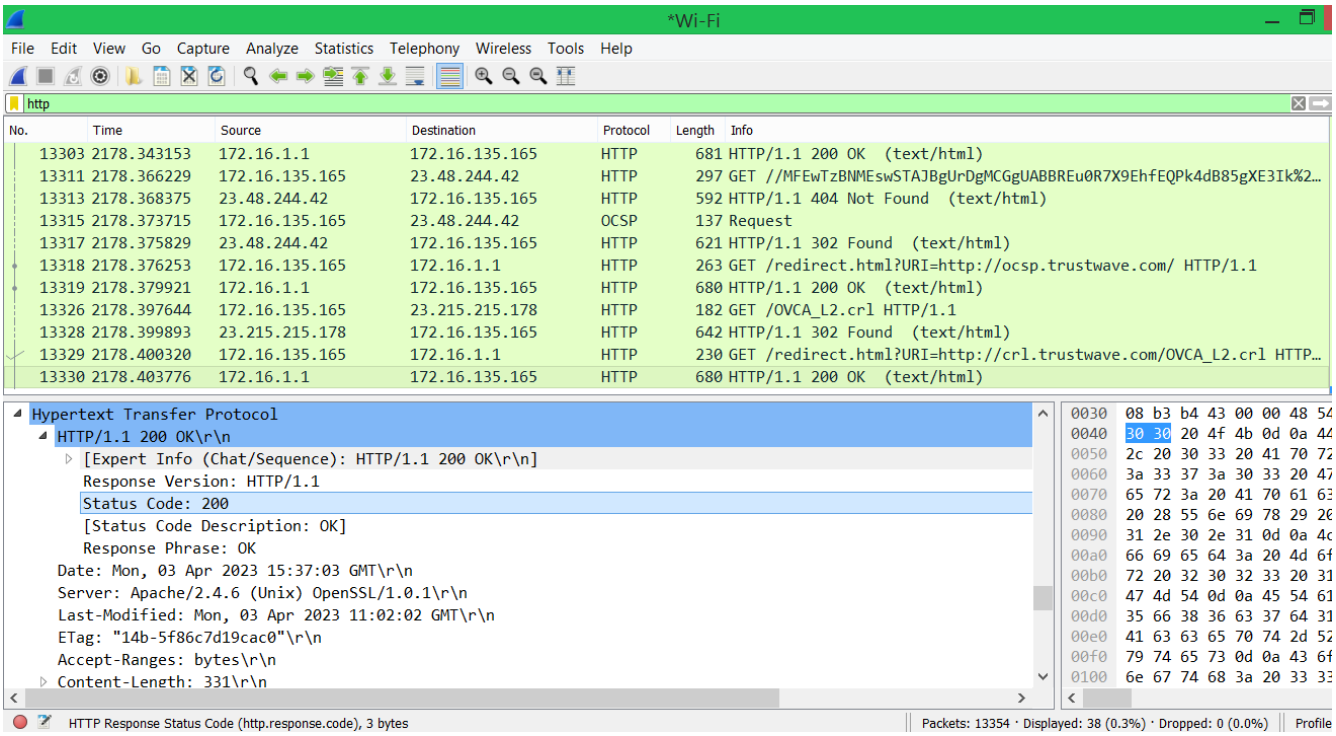
The first response packet (PDU) from the server, packet 10 contains the status code and phrase.



14. What is the status code and phrase in the response?

Answer :-

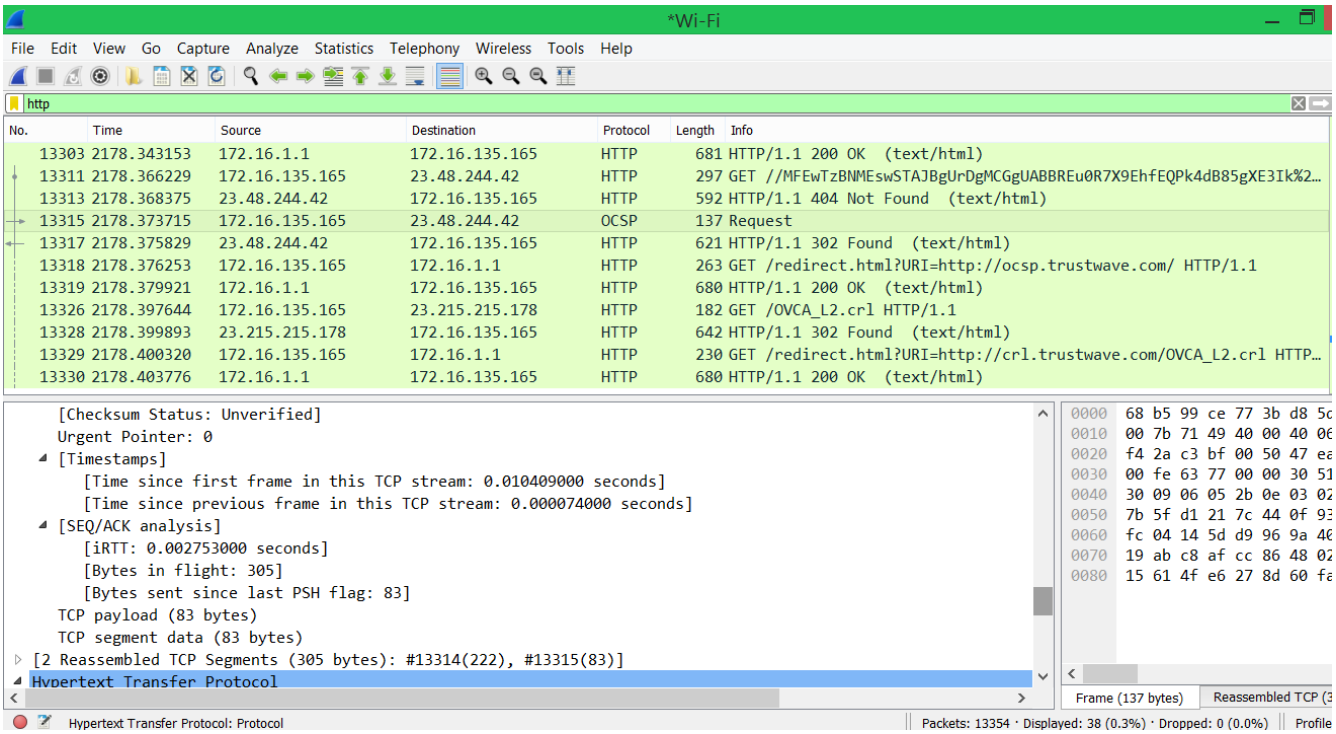
200 OK



15. How many data-containing TCP segments were needed to carry the single HTTP response and the text of the Bill of Rights?

Answer :-

According to the diagram 7, 3 TCP segments (10, 11 and 13) were needed to carry the single HTTP response and the text of the Bill of Rights

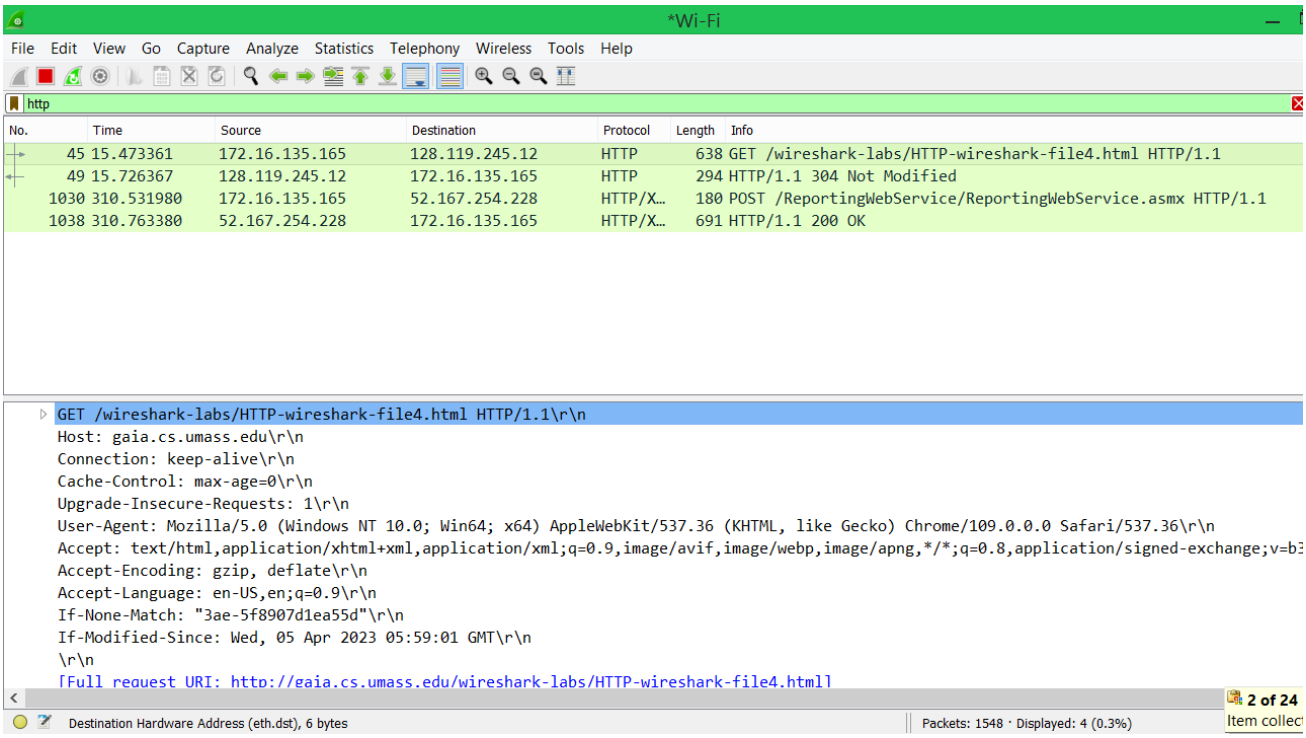


4. HTML Documents with Embedded Objects

16. How many HTTP GET request messages did your browser send? To which Internet addresses were these GET requests sent?

Answer :-

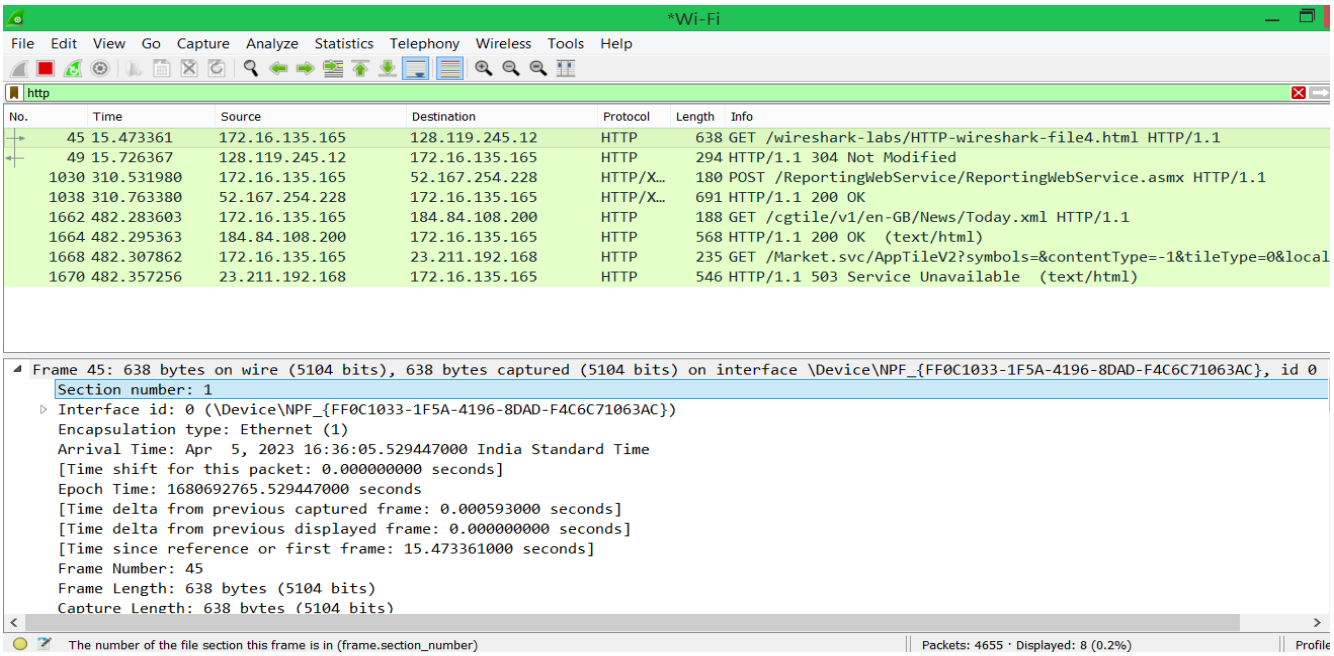
According to the Diagram 8, the browser sent 3 HTTP GET request messages. Packet 10 was sent to 128.119.245.12, packet 17 was sent to 165.193.123.218, and packet 20 was sent to 134.241.6.82.



17. Can you tell whether your browser downloaded the two images serially, or whether they were downloaded from the two web sites in parallel? Explain.

Answer :-

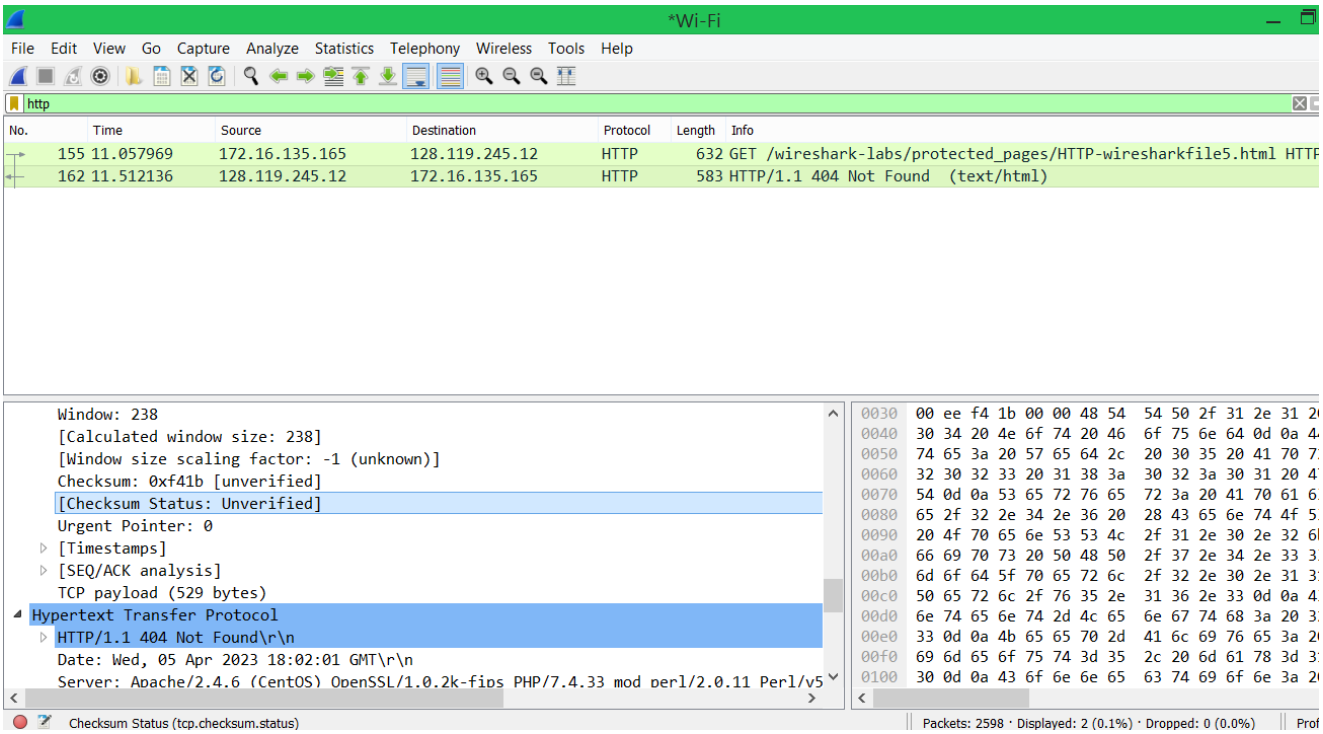
Two images were downloaded in parallel. According to diagram 8, the HTTP GET requests for two images were sent using packet 17 and 20, and the response packets were 25 and 54 which means the request for the second image was made before the first image was received.



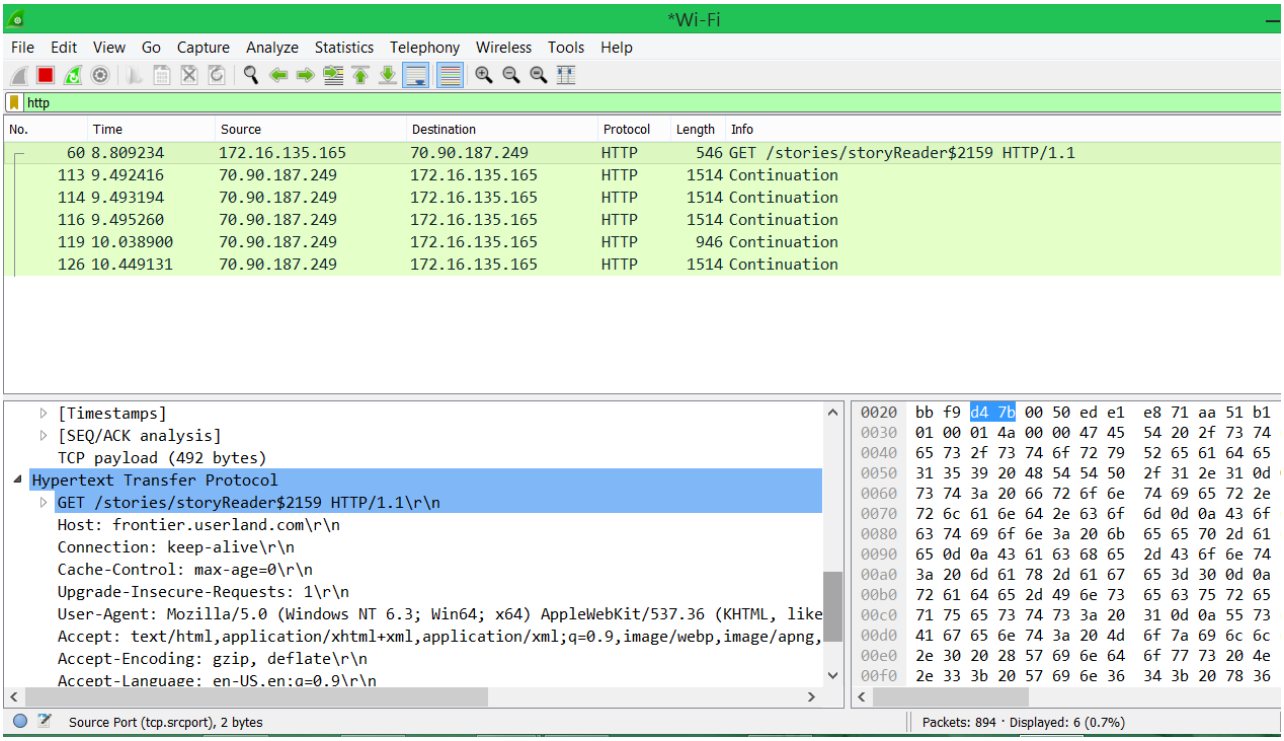
18. What is the server's response (status code and phrase) in response to the initial HTTP GET message from your browser?

Answer :-

According to the diagram , the initial HTTP GET message should be packet 6 and the packet 9 is the response to the packet 6. Thus the server's response is 401 Authorization Required.



19. When your browser's sends the HTTP GET message for the second time, what new field is included in the HTTP GET message?

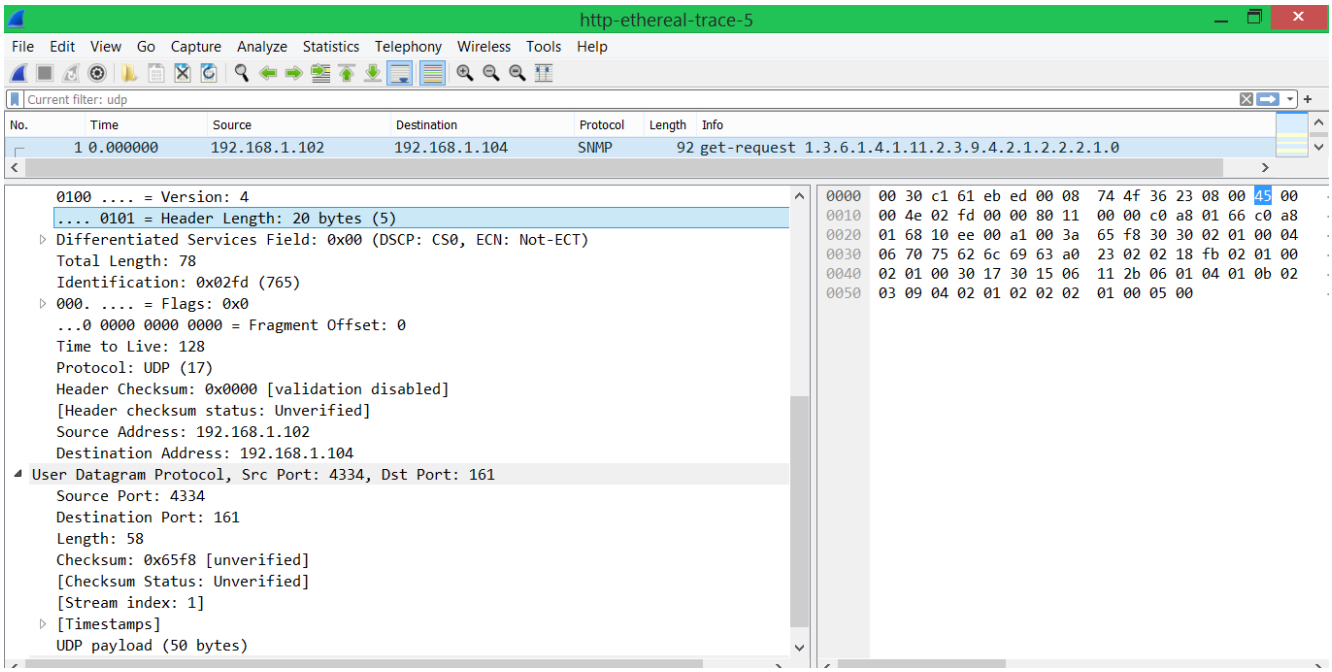


UDP :-

1. Select *one* UDP packet from your trace. From this packet, determine how many fields there are in the UDP header. (You shouldn't look in the textbook! Answer these questions directly from what you observe in the packet trace.) Name these fields.

Solution:

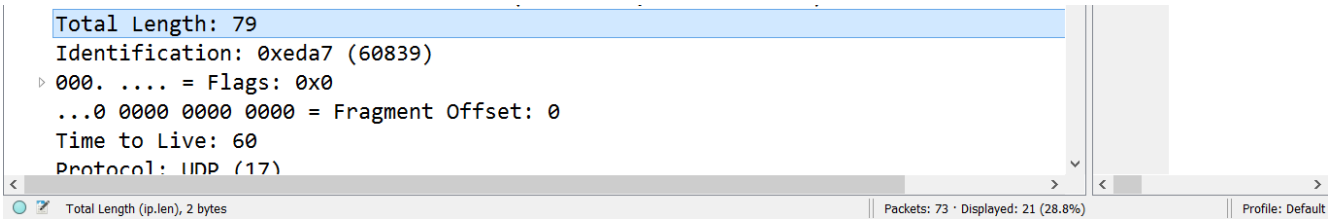
UDP header contains 4 fields:
1. source port; 2. destination port; 3. length; 4. checksum

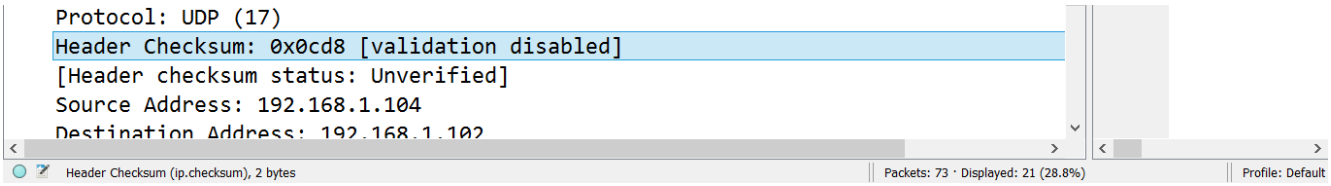


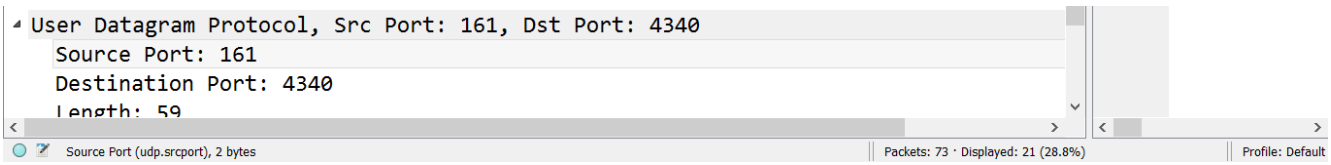
2. By consulting the displayed information in Wireshark's packet content field for this packet, determine the length (in bytes) of each of the UDP header fields.

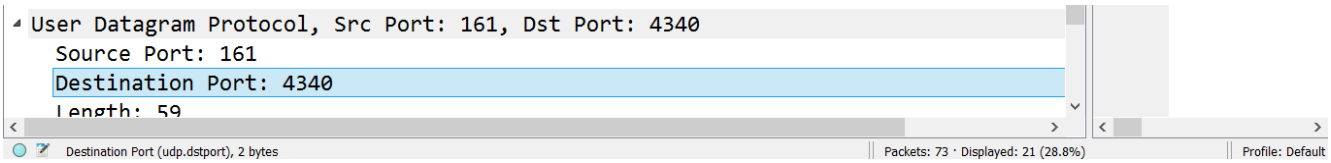
Solution:

The UDP header has a fixed length of 8 bytes. Each of these 4 header fields is 2 bytes long.





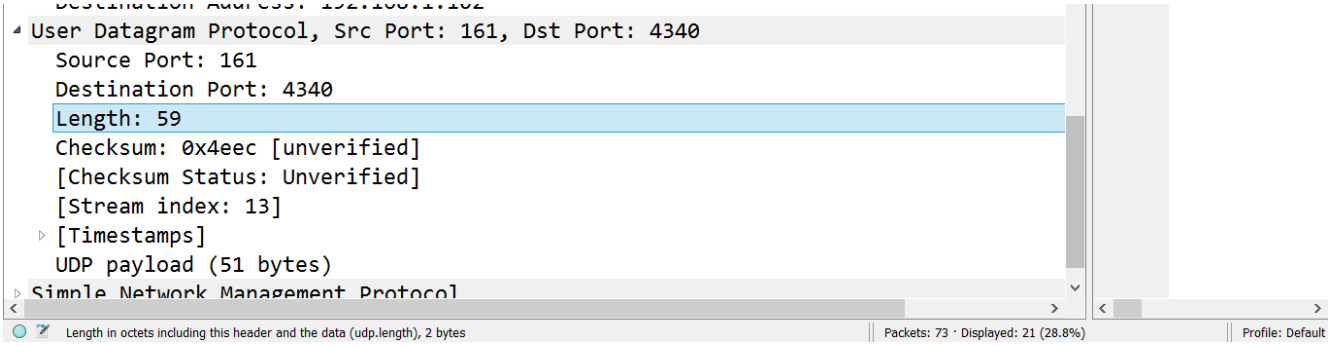




3. The value in the Length field is the length of what? (You can consult the text for this answer). Verify your claim with your captured UDP packet.

Solution:

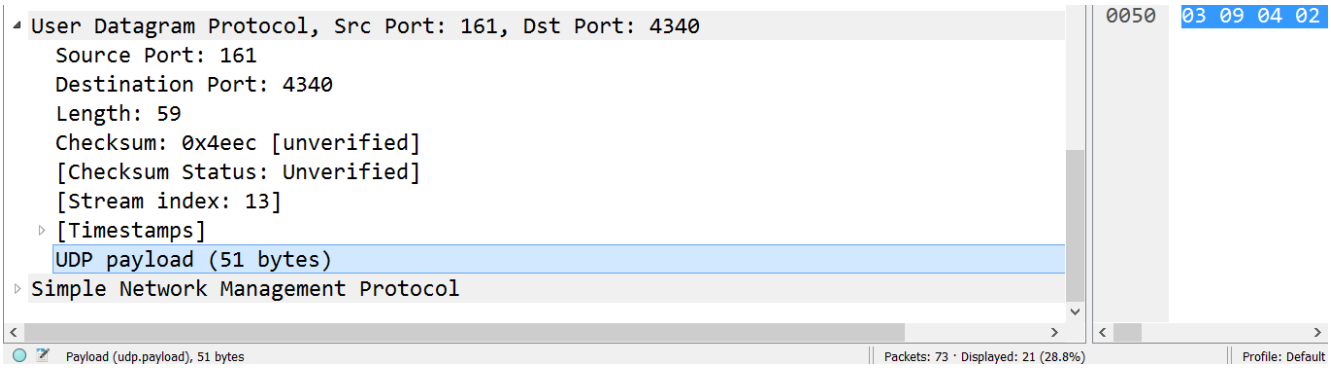
The length field specifies the number of bytes in the UDP segment (header plus data). An explicit length value is needed since the size of the data field may differ from one UDP segment to the next. The length of UDP payload for selected packet is 59 bytes.



4. What is the maximum number of bytes that can be included in a UDP payload? (Hint: the answer to this question can be determined by your answer to 2. above)

Solution:

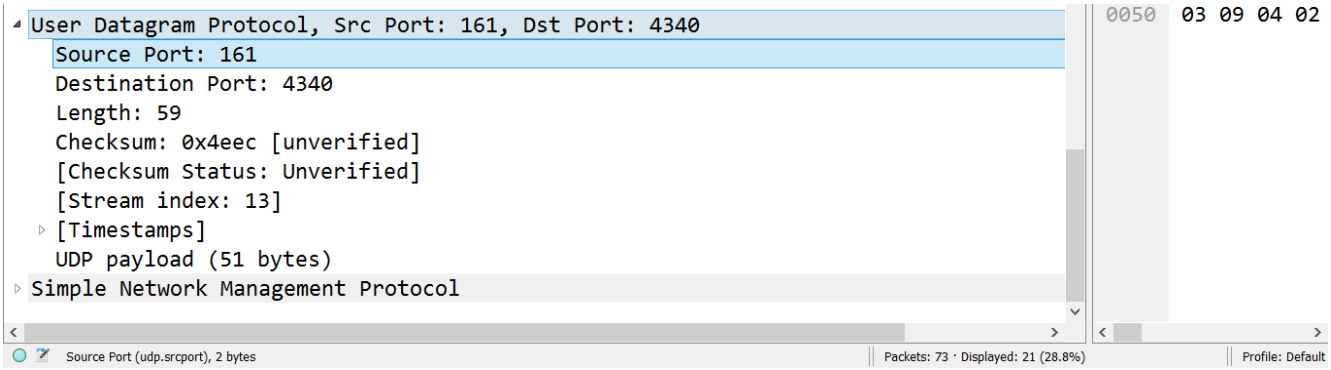
The maximum number of bytes that can be included in a UDP payload is (2^16 – 1) bytes plus the header bytes. This gives 65535 bytes – 8 bytes = 65527 bytes.



5. What is the largest possible source port number? (Hint: see the hint in 4.)

Solution:

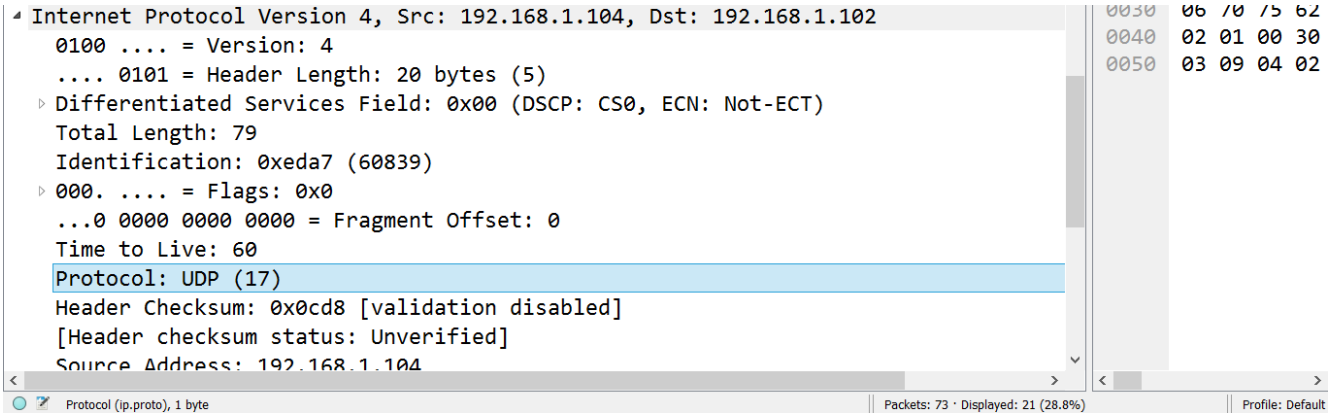
The largest possible source port number is (2^16 – 1) = 65535.



6. What is the protocol number for UDP? Give your answer in both hexadecimal and decimal notation. To answer this question, you'll need to look into the Protocol field of the IP datagram containing this UDP segment (see Figure 4.13 in the text, and the discussion of IP header fields).

Solution:

The IP protocol number for UDP is 0x11 hex, which is 17 in decimal value.

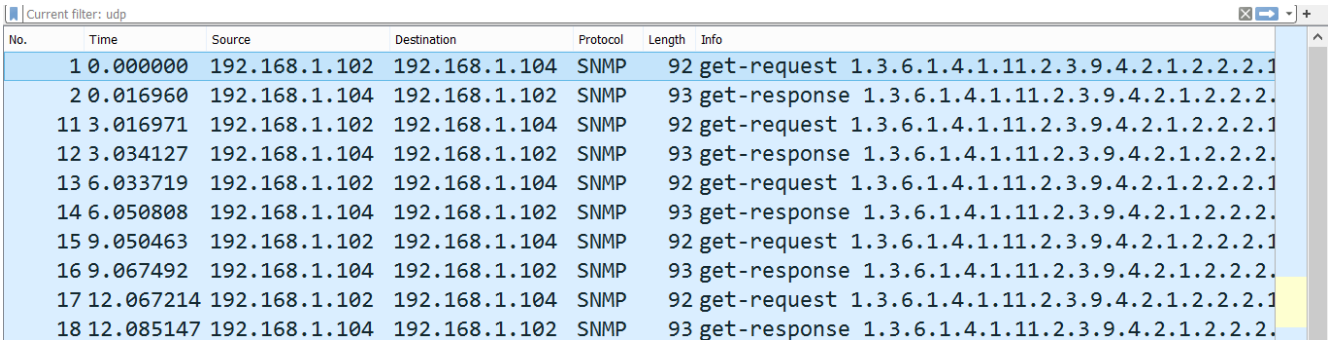


7. Examine a pair of UDP packets in which your host sends the first UDP packet and the second UDP packet is a reply to this first UDP packet. (Hint: for a second packet to be sent in response to a first packet, the sender of the first packet should be the destination of the second packet). Describe the relationship between the port numbers in the two packets.

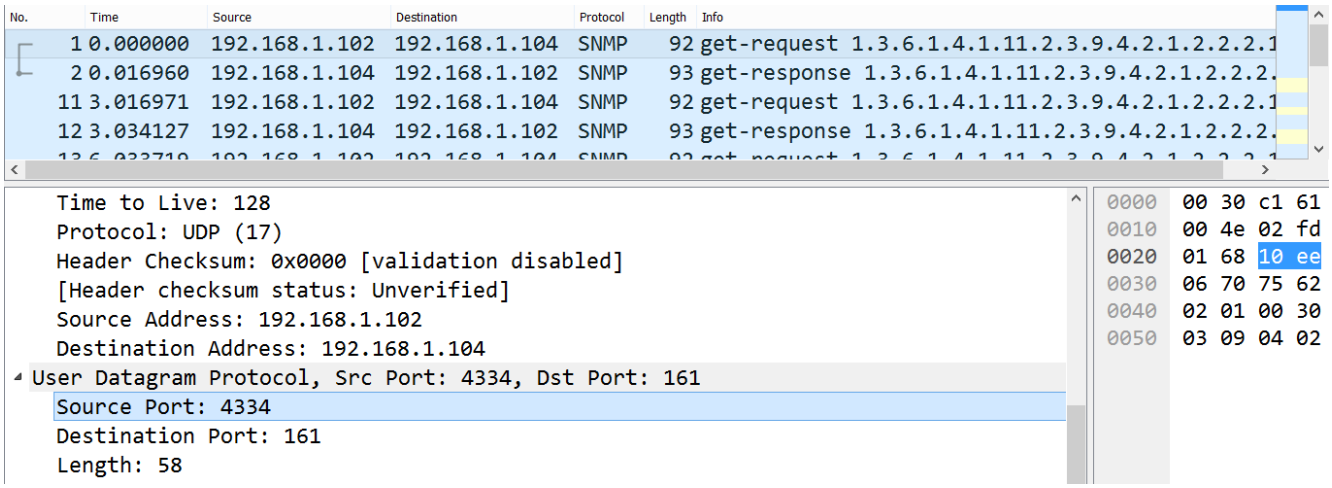
Answer :-

The source port of the UDP packet sent by the host is the same as the destination port of the reply packet, and conversely the destination port of the UDP packet sent by the host is the same as the source port of the reply packet.

You can see that destination of first packet is the sender of second packet and sender of first packet is the destination of second packet. So we can say that communication is perform between to machine only.



Source and Destination port of 1st packet :-

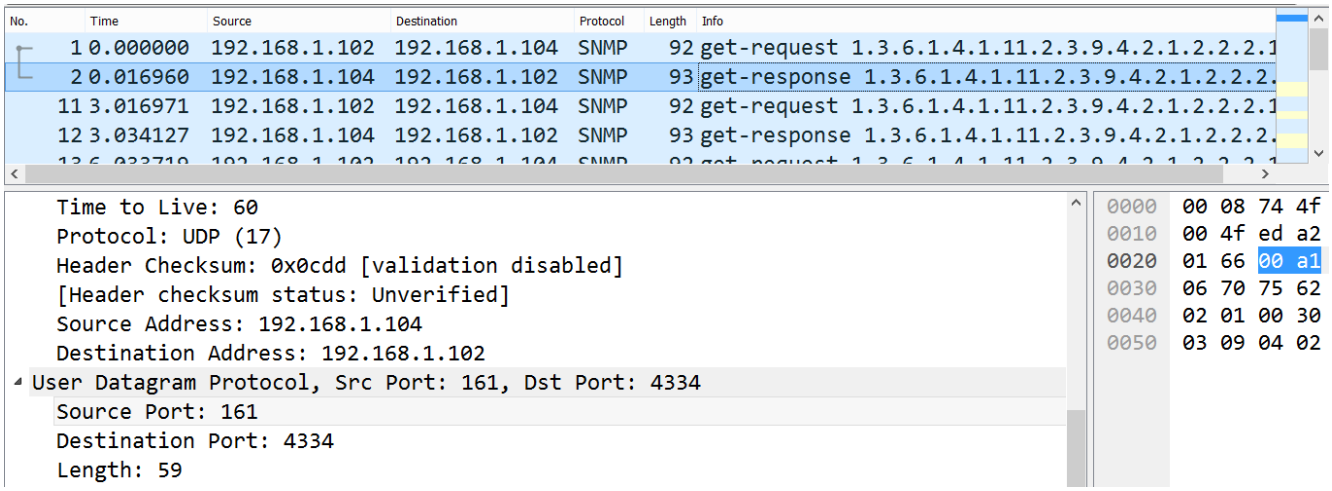


No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.102	192.168.1.104	SNMP	92	get-request 1.3.6.1.4.1.11.2.3.9.4.2.1.2.2.2.1
2	0.016960	192.168.1.104	192.168.1.102	SNMP	93	get-response 1.3.6.1.4.1.11.2.3.9.4.2.1.2.2.2.
11	3.016971	192.168.1.102	192.168.1.104	SNMP	92	get-request 1.3.6.1.4.1.11.2.3.9.4.2.1.2.2.2.1
12	3.034127	192.168.1.104	192.168.1.102	SNMP	93	get-response 1.3.6.1.4.1.11.2.3.9.4.2.1.2.2.2.

```
    Time to Live: 128
    Protocol: UDP (17)
    Header Checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.1.102
    Destination Address: 192.168.1.104
▾ User Datagram Protocol, Src Port: 4334, Dst Port: 161
    Source Port: 4334
    Destination Port: 161
    Length: 58
```

```
0000  00 30 c1 61
0010  00 4e 02 fd
0020  01 68 10 ee
0030  06 70 75 62
0040  02 01 00 30
0050  03 09 04 02
```

Source and Destination port of 2nd packet :-



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.102	192.168.1.104	SNMP	92	get-request 1.3.6.1.4.1.11.2.3.9.4.2.1.2.2.2.1
2	0.016960	192.168.1.104	192.168.1.102	SNMP	93	get-response 1.3.6.1.4.1.11.2.3.9.4.2.1.2.2.2.
11	3.016971	192.168.1.102	192.168.1.104	SNMP	92	get-request 1.3.6.1.4.1.11.2.3.9.4.2.1.2.2.2.1
12	3.034127	192.168.1.104	192.168.1.102	SNMP	93	get-response 1.3.6.1.4.1.11.2.3.9.4.2.1.2.2.2.

```
    Time to Live: 60
    Protocol: UDP (17)
    Header Checksum: 0x0cdd [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.1.104
    Destination Address: 192.168.1.102
▾ User Datagram Protocol, Src Port: 161, Dst Port: 4334
    Source Port: 161
    Destination Port: 4334
    Length: 59
```

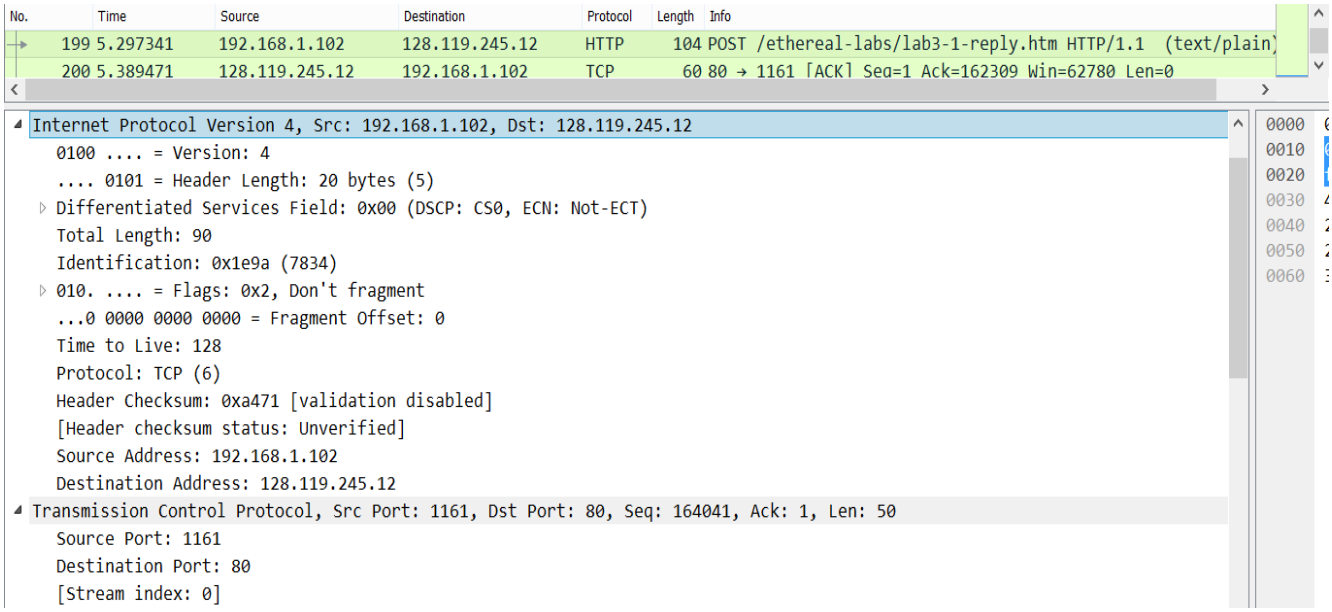
```
0000  00 08 74 4f
0010  00 4f ed a2
0020  01 66 00 a1
0030  06 70 75 62
0040  02 01 00 30
0050  03 09 04 02
```

TCP :-

1. What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu? To answer this question, it's probably easiest to select an HTTP message and explore the details of the TCP packet used to carry this HTTP message, using the "details of the selected packet header window" (refer to Figure 2 in the "Getting Started with Wireshark" Lab if you're uncertain about the Wireshark windows.

Answer :-

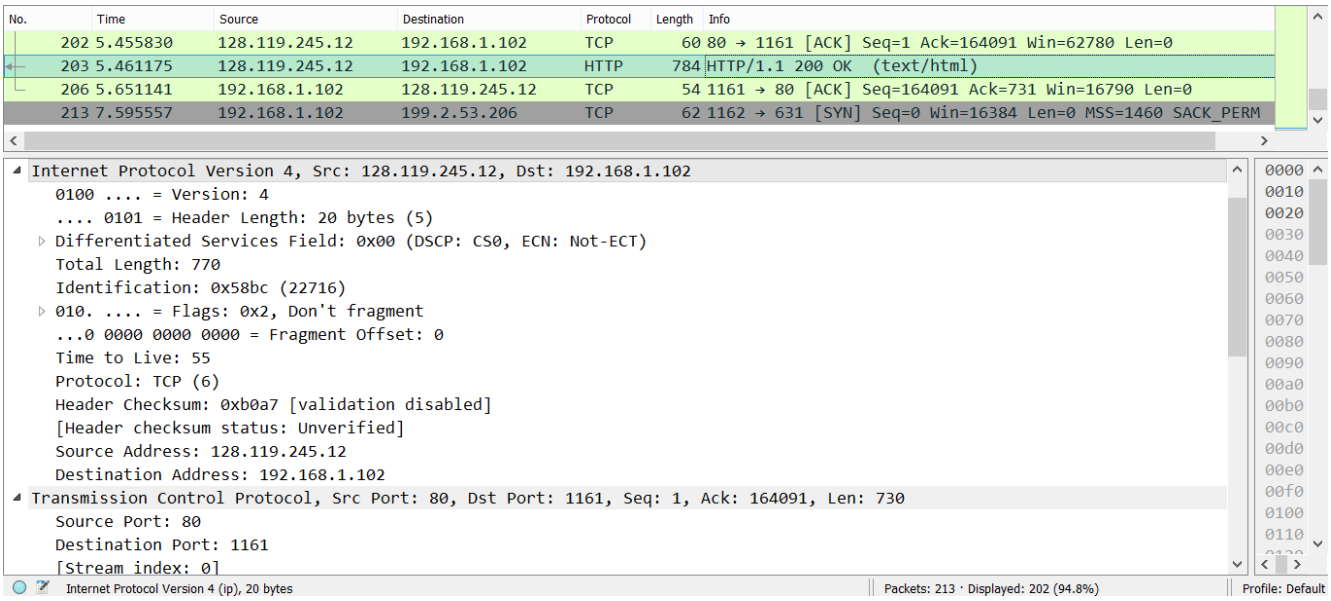
According to bellow figure, the client computer (source)'s IP address is 192.168.1.102 and the TPC source port number is 1161.



2. What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

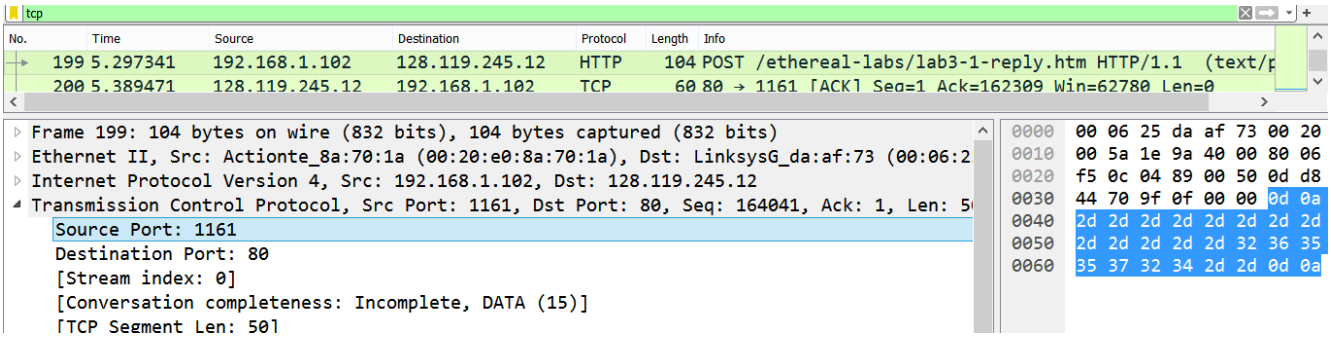
Answer :-

According to bellow figure, the IP address of gaia.cs.umass.edu is 128.119.245.12 and the TCP port number is 80.



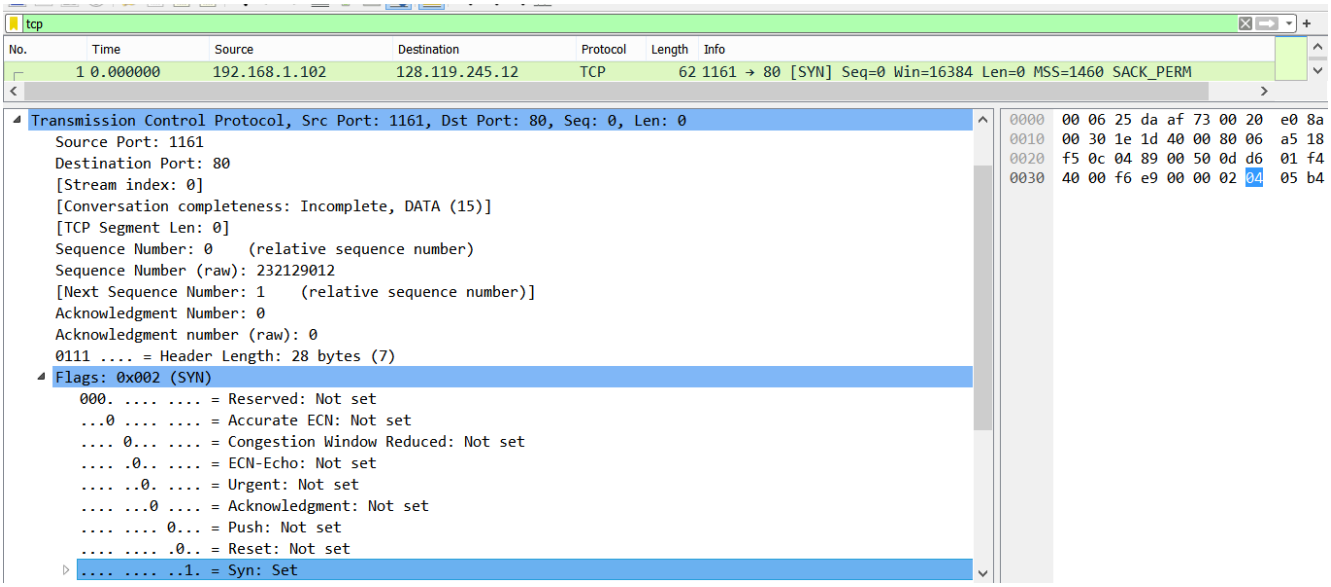
3. What is the IP address and TCP port number used by your client computer (source) to transfer the file to gaia.cs.umass.edu?
Answer :-

According to above figure, my client computer's IP address is 192.168.1.102 and the TCP port is 1161.



4. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?

Answer :-
The sequence number of the TCP SYN segment is 0 since it is used to imitate the TCP connection between the client computer and gaia.cs.umass.edu.
According to bellow figure, in the Flags section, the Syn flag is set to 1 which indicates that this segment is a SYN segment.



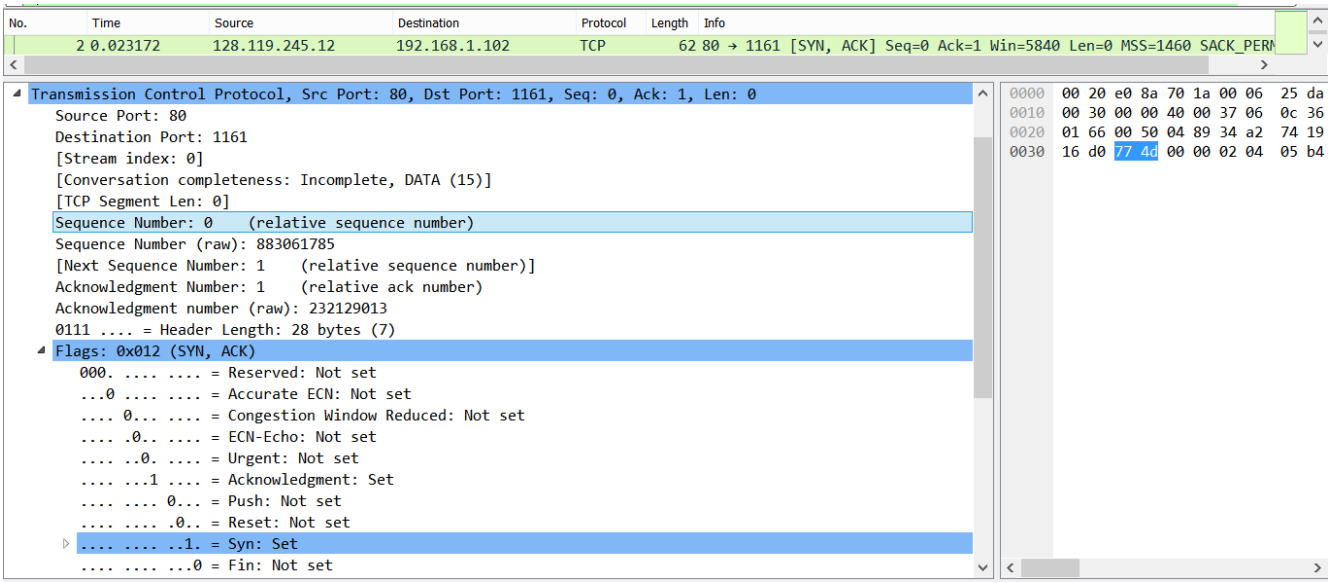
5. What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK segment?

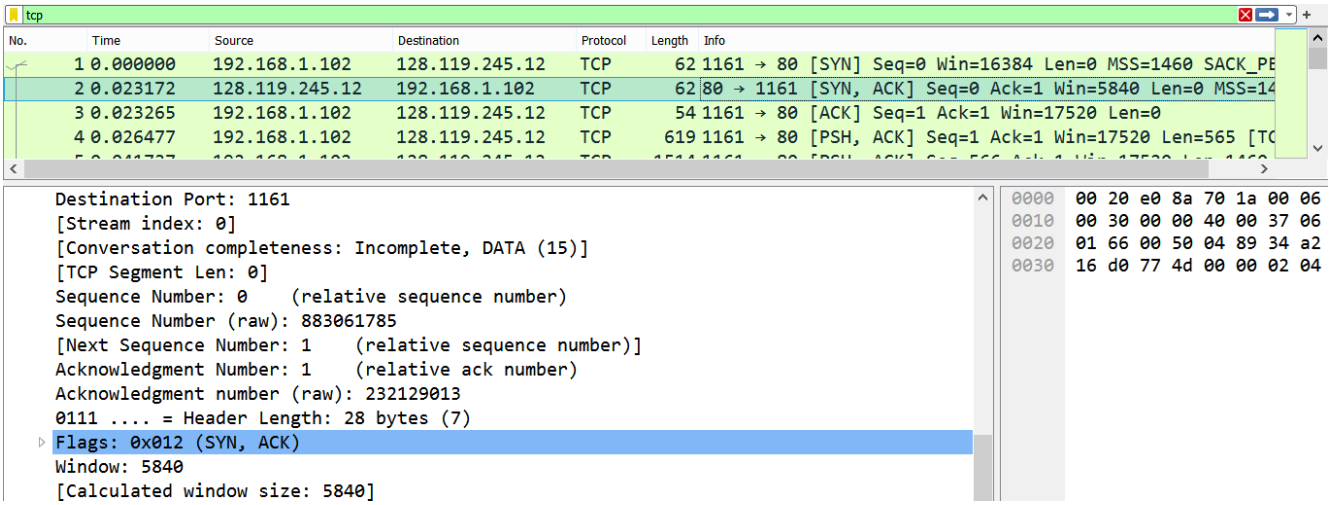
Answer :-

According to the bellow figure, the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN is 0. The value of the acknowledgement field in the SYNACK segment is 1.
The value of the ACKnowledgement field in the SYNACK segment is determined by the server gaia.cs.umass.edu. The server adds 1 to the initial sequence number of SYN segment form the client computer.

For this case, the initial sequence number of SYN segment from the client computer is 0, thus the value of the ACKnowledgement field in the SYNACK segment is 1.

A segment will be identified as a SYNACK segment if both SYN flag and Acknowledgement in the segment are set to 1.

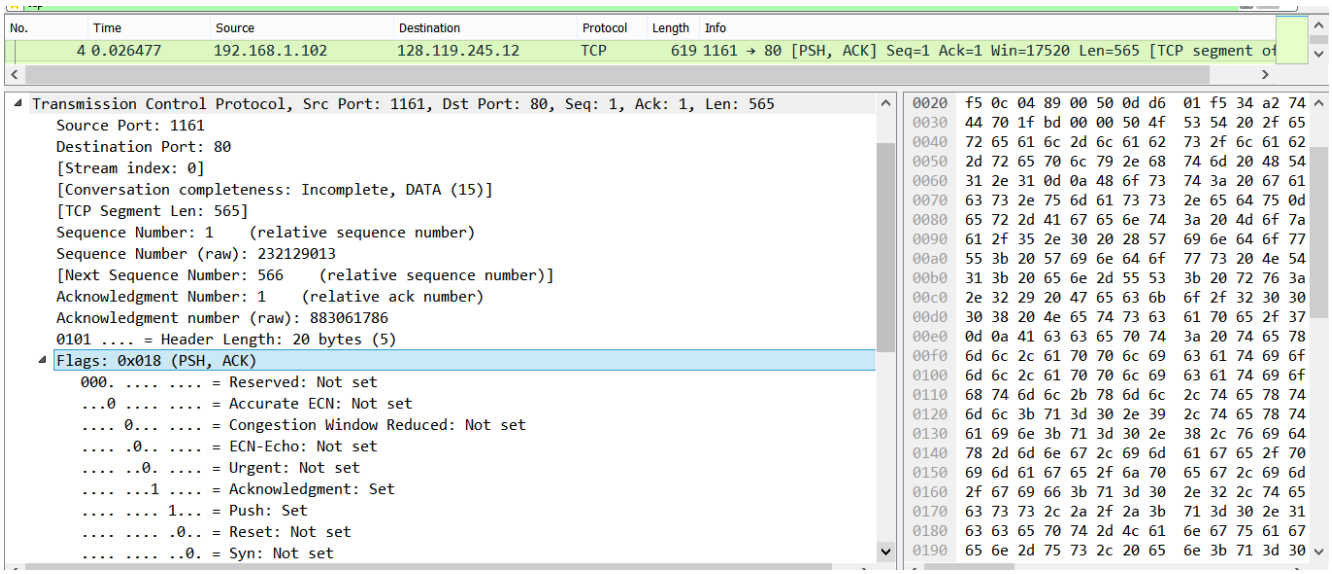


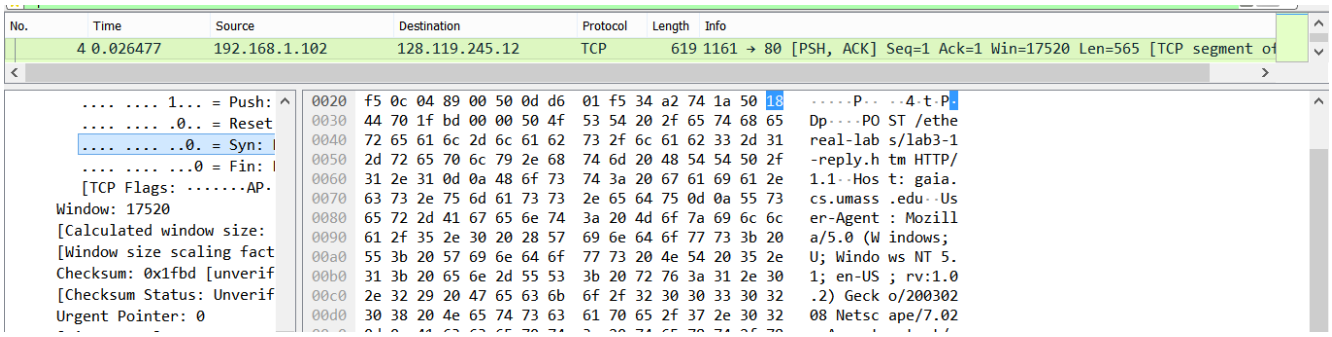


6. What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a "POST" within its DATA field.

Answer :-

According to above figure, the segment No.4 contains the HTTP POST command, the sequence number of this segment is 1.

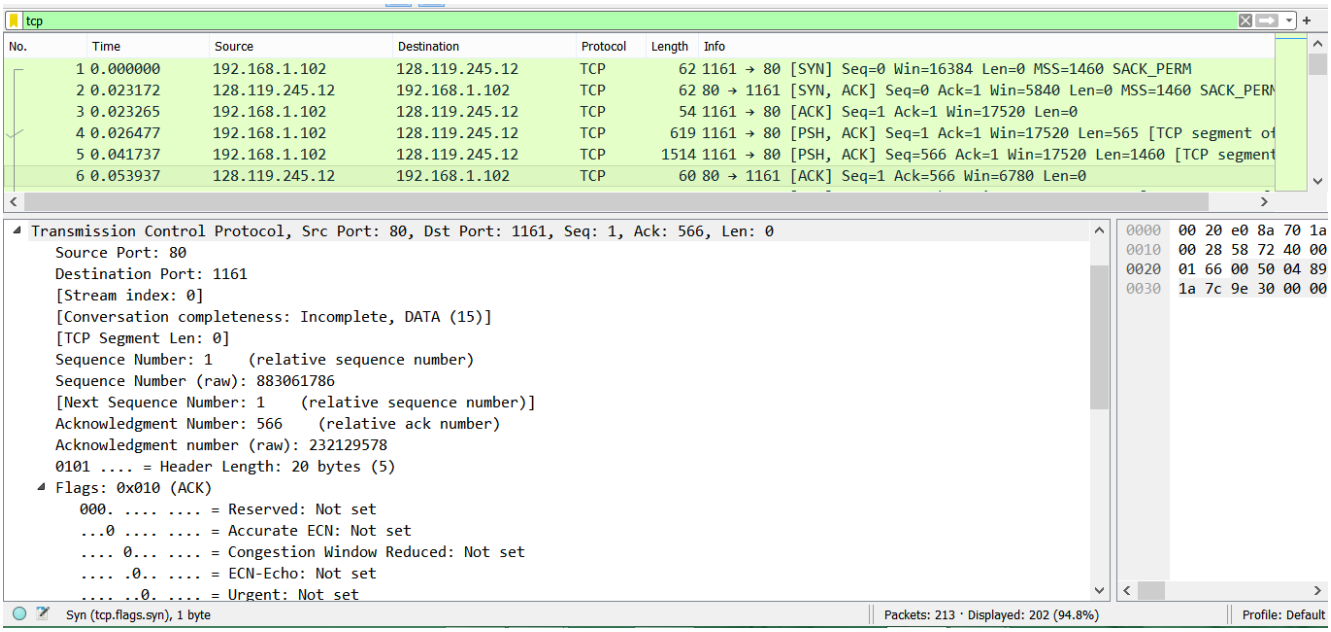


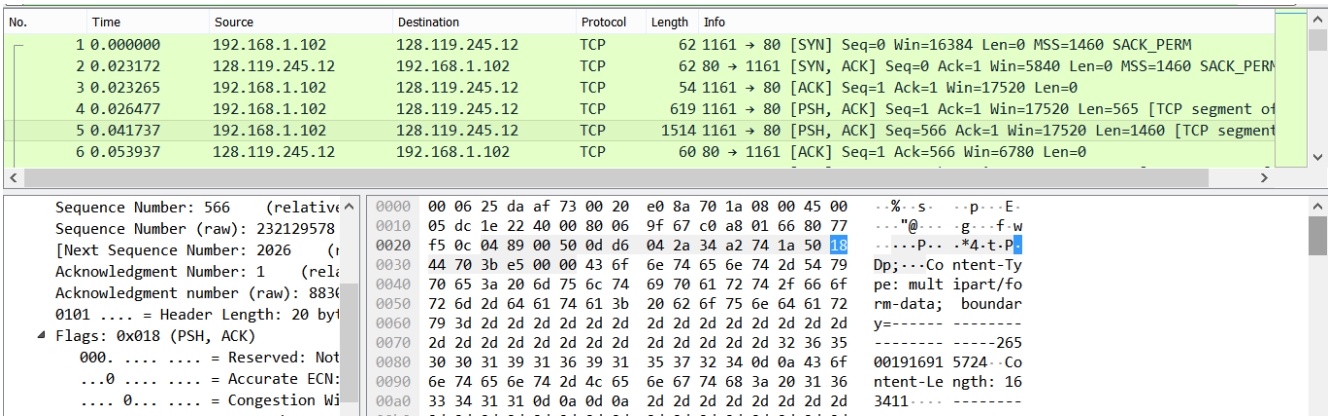


7. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)? At what time was each segment sent? When was the ACK for each segment received? Given the difference between when each TCP segment was sent, and when its acknowledgement was received, what is the RTT value for each of the six segments? What is the EstimatedRTT value (see Section 3.5.3, page 242 in text) after the receipt of each ACK? Assume that the value of the EstimatedRTT is equal to the measured RTT for the first segment, and then is computed using the EstimatedRTT equation on page 242 for all subsequent segments.

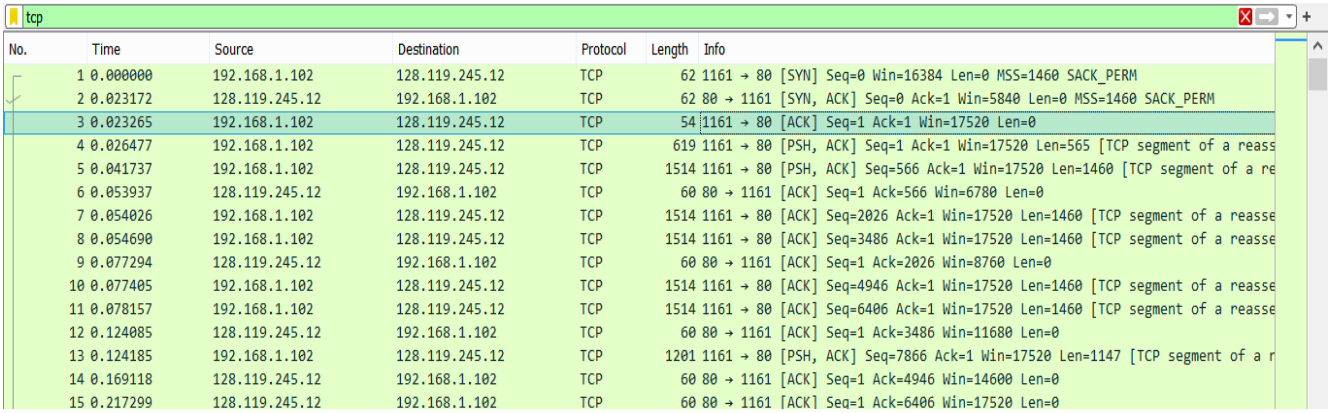
Note: **Wireshark has a nice feature that allows you to plot the RTT for each of the TCP segments sent. Select a TCP segment in the "listing of captured packets" window that is being sent from the client to the gaia.cs.umass.edu server. Then select: *Statistics->TCP Stream Graph- >Round Trip Time Graph.***

At what time was each segment sent? When was the ACK for each segment received?


tcp
No. Time Source Destination Protocol Length Info
1 0.000000 192.168.1.102 128.119.245.12 TCP 62 1161 → 80 [SYN] Seq=0 Win=16384 Len=0 MSS=1460 SACK_PERM
2 0.023172 128.119.245.12 192.168.1.102 TCP 62 80 → 1161 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM
3 0.023265 192.168.1.102 128.119.245.12 TCP 54 1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
4 0.026477 192.168.1.102 128.119.245.12 TCP 619 1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565 [TCP segment of
5 0.041737 192.168.1.102 128.119.245.12 TCP 1514 1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460 [TCP segment
6 0.053937 128.119.245.12 192.168.1.102 TCP 60 80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 Len=0
Transmission Control Protocol, Src Port: 80, Dst Port: 1161, Seq: 1, Ack: 566, Len: 0
Source Port: 80
Destination Port: 1161
[Stream index: 0]
[Conversation completeness: Incomplete, DATA (15)]
[TCP Segment Len: 0]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 883061786
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 566 (relative ack number)
Acknowledgment number (raw): 232129578
0101 .... = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)
000. .... = Reserved: Not set
...0 .... = Accurate ECN: Not set
.... 0... .... = Congestion Window Reduced: Not set
.... .0.. .... = ECN-Echo: Not set
.... ..0. .... = Urgent: Not set
0000 00 20 e0 8a 70 1a
0010 00 28 58 72 40 00
0020 01 66 00 50 04 89
0030 1a 7c 9e 30 00 00
Syn (tcp.flags.syn), 1 byte
Packets: 213 · Displayed: 202 (94.8%)
Profile: Default

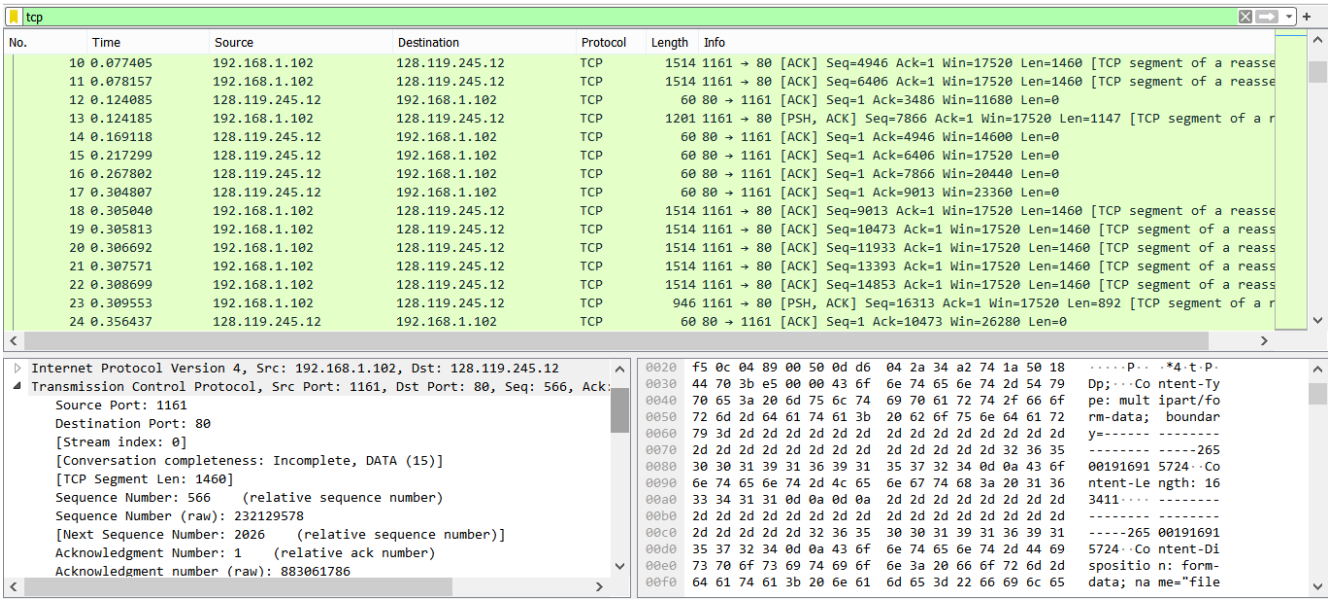

No. Time Source Destination Protocol Length Info
1 0.000000 192.168.1.102 128.119.245.12 TCP 62 1161 → 80 [SYN] Seq=0 Win=16384 Len=0 MSS=1460 SACK_PERM
2 0.023172 128.119.245.12 192.168.1.102 TCP 62 80 → 1161 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM
3 0.023265 192.168.1.102 128.119.245.12 TCP 54 1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
4 0.026477 192.168.1.102 128.119.245.12 TCP 619 1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565 [TCP segment of
5 0.041737 192.168.1.102 128.119.245.12 TCP 1514 1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460 [TCP segment
6 0.053937 128.119.245.12 192.168.1.102 TCP 60 80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 Len=0
Sequence Number: 566 (relative
Sequence Number (raw): 232129578
[Next Sequence Number: 2026 (
Acknowledgment Number: 1 (rela
Acknowledgment number (raw): 883
0101 .... = Header Length: 20 byt
Flags: 0x018 (PSH, ACK)
000. .... = Reserved: Not
...0 .... = Accurate ECN:
.... 0... .... = Congestion Wi
0000 00 06 25 da af 73 00 20 e0 8a 70 1a 08 00 45 00 ··%··s· ··p···E·
0010 05 dc 1e 22 40 00 80 06 9f 67 c0 a8 01 66 80 77 ···"@··· ·g···f·w
0020 f5 0c 04 89 00 50 0d d6 04 2a 34 a2 74 1a 50 18 ·····P·· ·*4·t·P·
0030 44 70 3b e5 00 00 43 6f 6e 74 65 6e 74 2d 54 79 Dp;···Co ntent-Ty
0040 70 65 3a 20 6d 75 6c 74 69 70 61 72 74 2f 66 6f pe: mult ipart/fo
0050 72 6d 2d 64 61 74 61 3b 20 62 6f 75 6e 64 61 72 rm-data; boundar
0060 79 3d 2d 2d 2d 2d 2d 2d 2d 2d 2d 2d 2d 2d 2d 2d y=------ --------
0070 2d 2d 2d 2d 2d 2d 2d 2d 2d 2d 2d 2d 32 36 35 -------- -----265
0080 30 30 31 39 31 36 39 31 35 37 32 34 0d 0a 43 6f 00191691 5724··Co
0090 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 31 36 ntent-Le ngth: 16
00a0 33 34 31 31 0d 0a 0d 0a 2d 2d 2d 2d 2d 2d 2d 2d 3411···· --------

Segments 1-6



tcp

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.102	128.119.245.12	TCP	62	1161 → 80 [SYN] Seq=0 Win=16384 Len=0 MSS=1460 SACK_PERM
2	0.023172	128.119.245.12	192.168.1.102	TCP	62	80 → 1161 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM
3	0.023265	192.168.1.102	128.119.245.12	TCP	54	1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
4	0.026477	192.168.1.102	128.119.245.12	TCP	619	1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565 [TCP segment of a reass
5	0.041737	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460 [TCP segment of a re
6	0.053937	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 Len=0
7	0.054026	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=2026 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
9	0.077294	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
10	0.077405	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
12	0.124085	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=3486 Win=11680 Len=0
13	0.124185	192.168.1.102	128.119.245.12	TCP	1201	1161 → 80 [PSH, ACK] Seq=7866 Ack=1 Win=17520 Len=1147 [TCP segment of a r
14	0.169118	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=4946 Win=14600 Len=0
15	0.217299	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=6406 Win=17520 Len=0



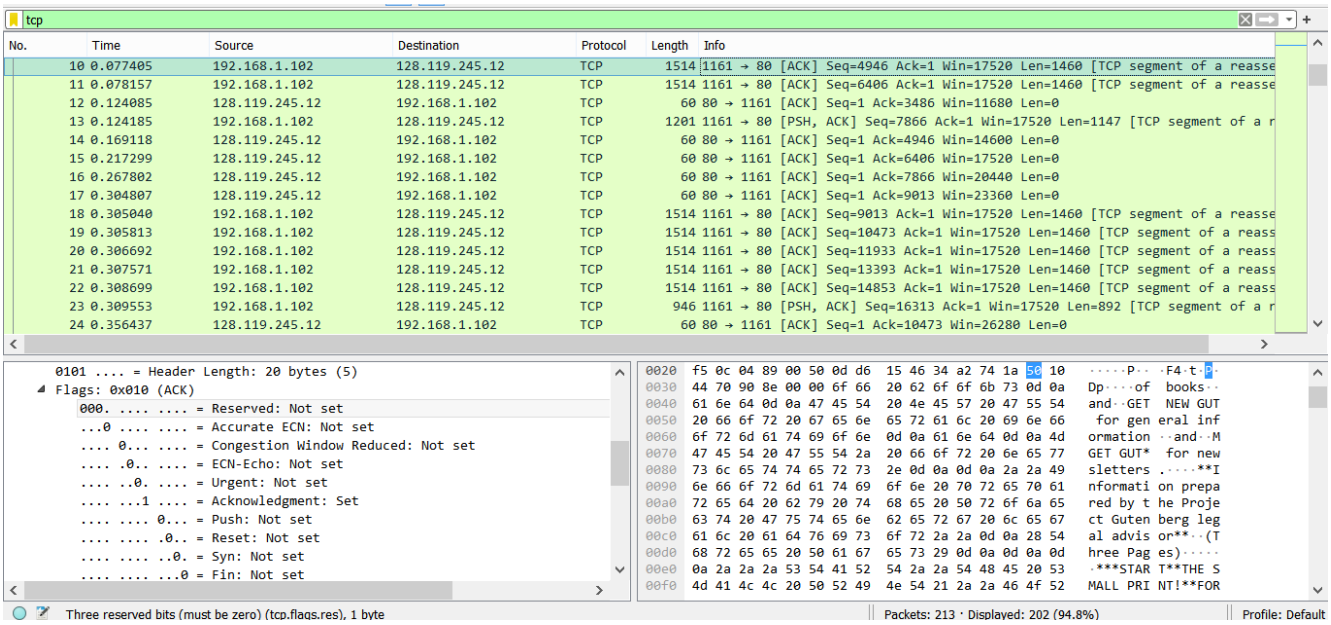
tcp

No.	Time	Source	Destination	Protocol	Length	Info
10	0.077405	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
12	0.124085	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=3486 Win=11680 Len=0
13	0.124185	192.168.1.102	128.119.245.12	TCP	1201	1161 → 80 [PSH, ACK] Seq=7866 Ack=1 Win=17520 Len=1147 [TCP segment of a r
14	0.169118	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=4946 Win=14600 Len=0
15	0.217299	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=6406 Win=17520 Len=0
16	0.267802	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=7866 Win=20440 Len=0
17	0.304807	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=9013 Win=23360 Len=0
18	0.305040	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=9013 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
19	0.305813	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=10473 Ack=1 Win=17520 Len=1460 [TCP segment of a reass
20	0.306692	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=11933 Ack=1 Win=17520 Len=1460 [TCP segment of a reass
21	0.307571	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=13393 Ack=1 Win=17520 Len=1460 [TCP segment of a reass
22	0.308699	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=14853 Ack=1 Win=17520 Len=1460 [TCP segment of a reass
23	0.309553	192.168.1.102	128.119.245.12	TCP	946	1161 → 80 [PSH, ACK] Seq=16313 Ack=1 Win=17520 Len=892 [TCP segment of a r
24	0.356437	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=10473 Win=26280 Len=0


```
▷ Internet Protocol Version 4, Src: 192.168.1.102, Dst: 128.119.245.12
◢ Transmission Control Protocol, Src Port: 1161, Dst Port: 80, Seq: 566, Ack:
    Source Port: 1161
    Destination Port: 80
    [Stream index: 0]
    [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 1460]
    Sequence Number: 566    (relative sequence number)
    Sequence Number (raw): 232129578
    [Next Sequence Number: 2026    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 883061786
```



```
0020  f5 0c 04 89 00 50 0d d6  04 2a 34 a2 74 1a 50 18   ·····P·· ·*4·t·P·
0030  44 70 3b e5 00 00 43 6f  6e 74 65 6e 74 2d 54 79   Dp;···Co ntent-Ty
0040  70 65 3a 20 6d 75 6c 74  69 70 61 72 74 2f 66 6f   pe: mult ipart/fo
0050  72 6d 2d 64 61 74 61 3b  20 62 6f 75 6e 64 61 72   rm-data;  boundar
0060  79 3d 2d 2d 2d 2d 2d 2d  2d 2d 2d 2d 2d 2d 2d 2d   y=------ --------
0070  2d 2d 2d 2d 2d 2d 2d 2d  2d 2d 2d 2d 2d 32 36 35   -------- -----265
0080  30 30 31 39 31 36 39 31  35 37 32 34 0d 0a 43 6f   00191691 5724··Co
0090  6e 74 65 6e 74 2d 4c 65  6e 67 74 68 3a 20 31 36   ntent-Le ngth: 16
00a0  33 34 31 31 0d 0a 0d 0a  2d 2d 2d 2d 2d 2d 2d 2d   3411···· --------
00b0  2d 2d 2d 2d 2d 2d 2d 2d  2d 2d 2d 2d 2d 2d 2d 2d   -------- --------
00c0  2d 2d 2d 2d 2d 32 36 35  30 30 31 39 31 36 39 31   -----265 00191691
00d0  35 37 32 34 0d 0a 43 6f  6e 74 65 6e 74 2d 44 69   5724··Co ntent-Di
00e0  73 70 6f 73 69 74 69 6f  6e 3a 20 66 6f 72 6d 2d   sposition: form-
00f0  64 61 74 61 3b 20 6e 61  6d 65 3d 22 66 69 6c 65   data; na me="file
```



ACK of segments 1-6

According to above figures, the segments 1-6 are No. 4,5,7,8,10 and 11. The ACK of segments 1-6 are No. 6,9,12,14,15,16 and 17.

Segment 1 sequence number is 1
Segment 2 sequence number is 566
Segment 3 sequence number is 2026
Segment 4 sequence number is 3486
Segment 5 sequence number is 4946
Segment 6 sequence number is 6406

Recording the sending time and received time of ACKs:

Segments no	Sent time	ACK received time	RTT
Segment 1	**0.026477**	**0.053937**	**0.02746**
Segment 2	**0.041737**	**0.077294**	**0.035557**
Segment 3	**0.054026**	**0.124085**	**0.070059**
Segment 4	**0.054690**	**0.169118**	**0.114428**
Segment 5	**0.077405**	**0.217299**	**0.139894**
Segment 6	**0.078157**	**0.267802**	**0.189645**

According to the formula:

EstimatedRTT = 0.875 * EstimatedRTT + 0.125 * SampleRTT

EstimatedRTT after the receipt of the ACK of segment 1:

EstimatedRTT = RTT for Segment 1 = 0.02746 s

EstimatedRTT after the receipt of the ACK of segment 2:

EstimatedRTT = 0.875 * 0.02746 + 0.125 * 0.035557=0.028472125 s

EstimatedRTT after the receipt of the ACK of segment 3:

EstimatedRTT = 0.875 * 0.02847125+ 0.125 * 0.070059= 0.0336704844 s

EstimatedRTT after the receipt of the ACK of segment 4:

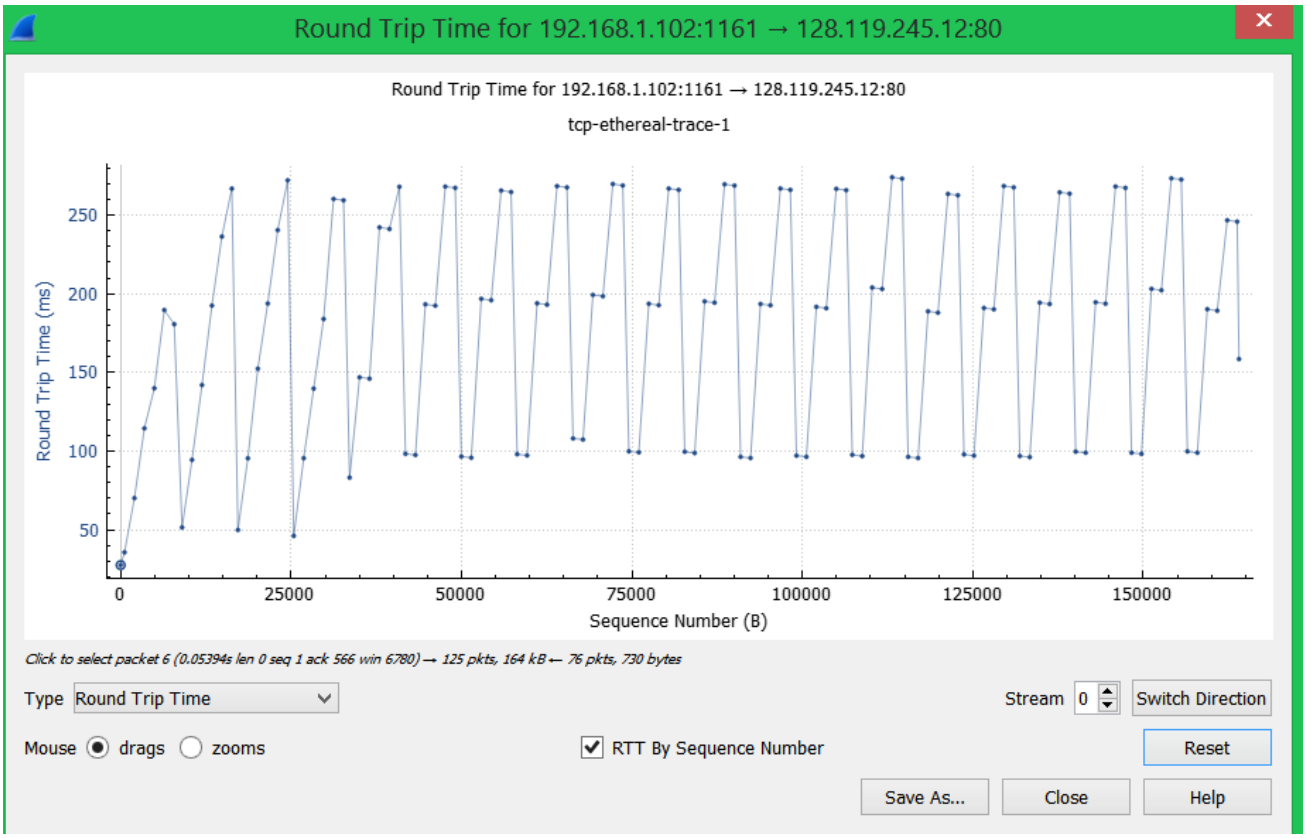
EstimatedRTT = 0.875 *0.0336704844 + 0.125 * 0.114428=0.0437651738 s

EstimatedRTT after the receipt of the ACK of segment 5:

EstimatedRTT = 0.875 * 0.0437651738 + 0.125 * 0.139894 s= 0.0557812771s

EstimatedRTT after the receipt of the ACK of segment 6:

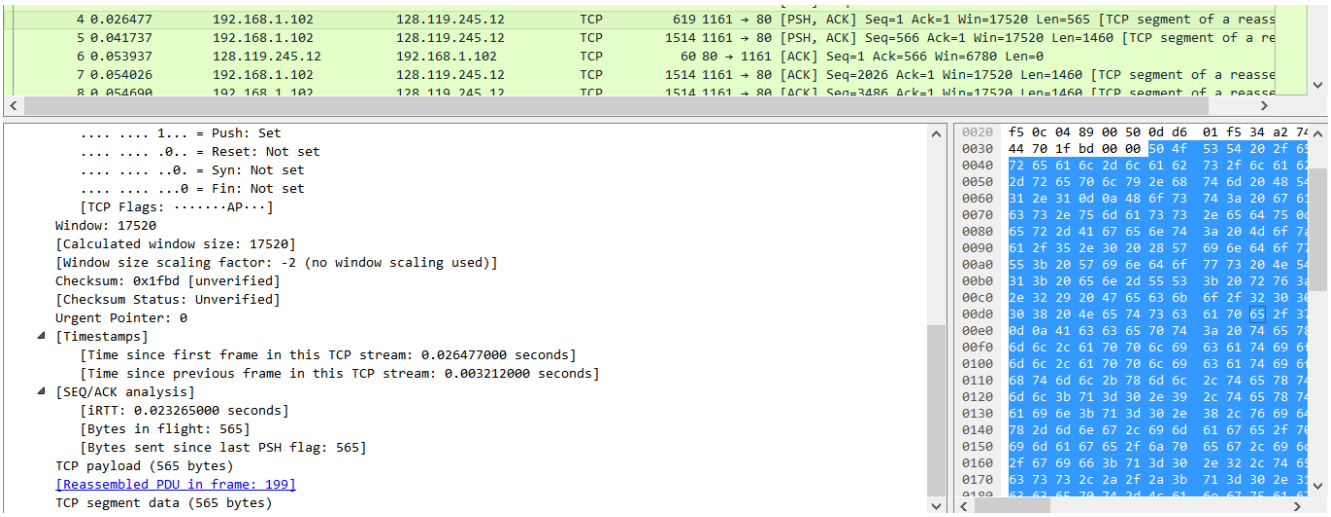
EstimatedRTT = 0.875 * 0.0557812771+ 0.125 *0.189645 = 0.0725142425 s



8. What is the length of each of the first six TCP segments?

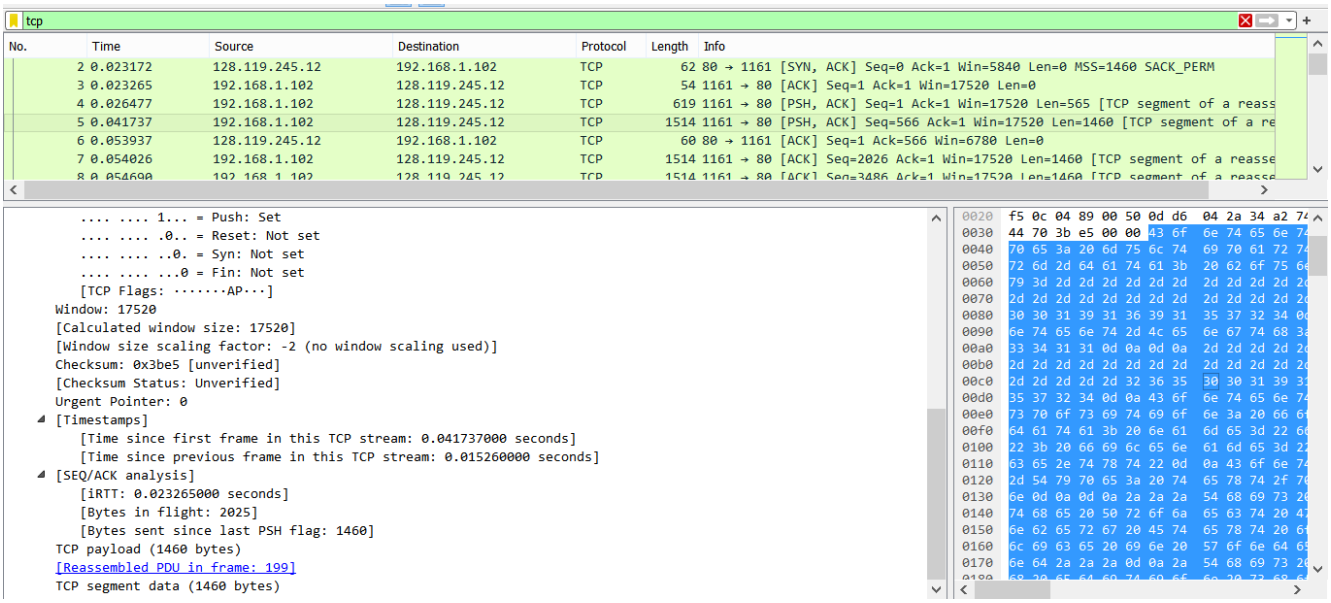
1st segment length :-

565 bytes



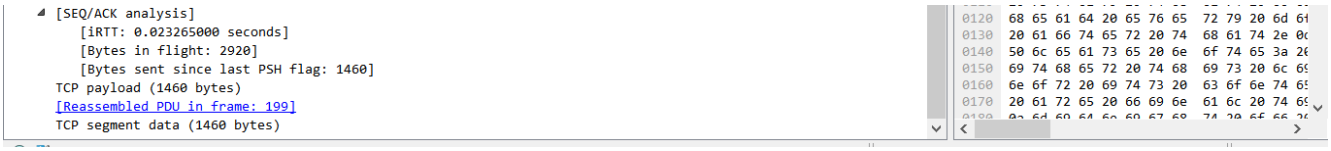
2nd segment length :-

1460 bytes



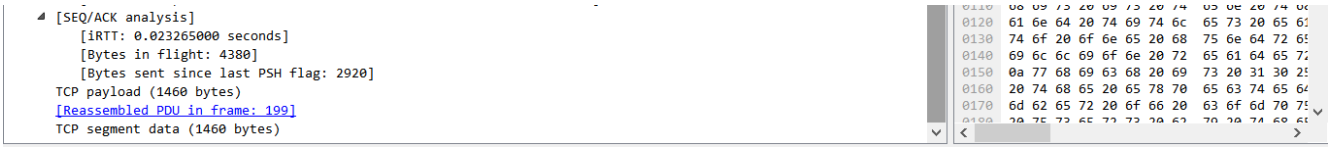
3rd segment length :-

1468 bytes

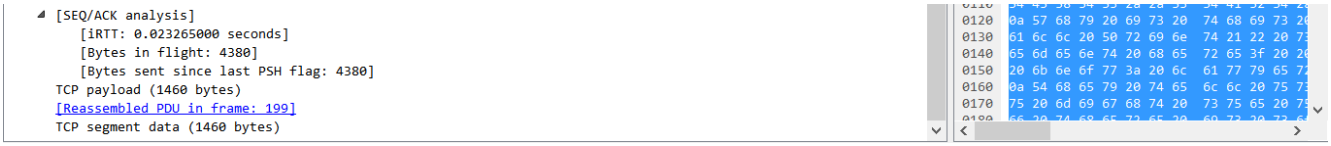


4rth segment length :-

1460 bytes

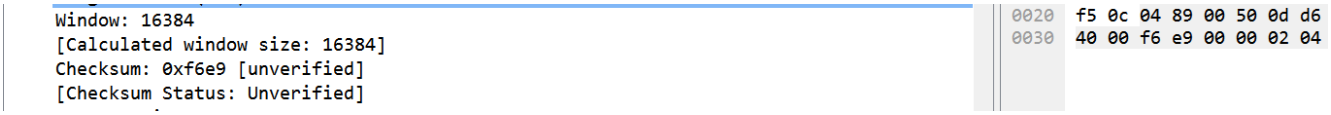


5fth segment length :-
1460 bytes



6th segment length :-

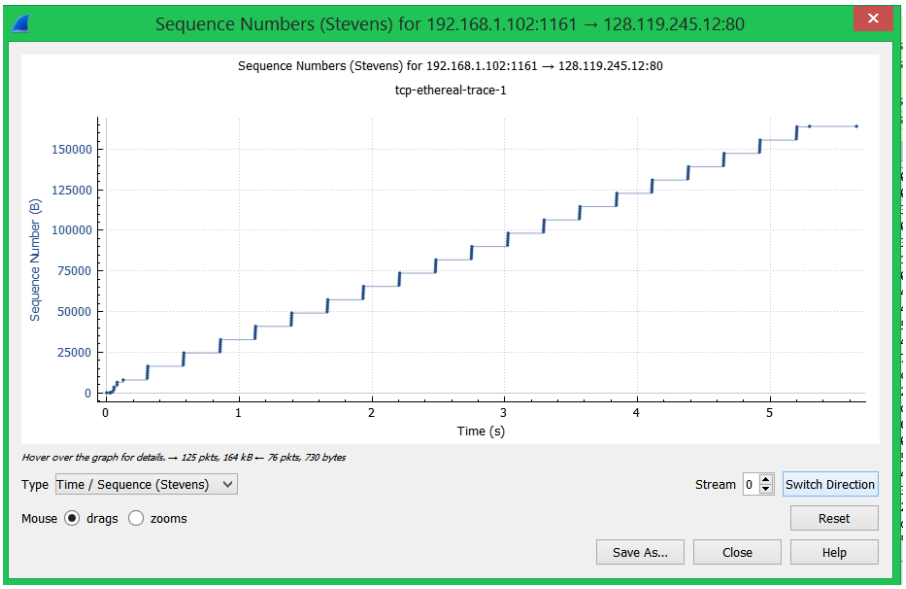
9. What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?



The minimum amount of available buffer space advertised at the received for the entire trace is indicated first ACK from the server, its value is 16384 bytes (shown in above figure).
This reviver window grows until it reaches the maximum receiver buffer size of 62780 bytes. According to the trace, the sender is never throttled due to lacking of receiver buffer space.

10. Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?

There is no retransmitted segments in the trace file since in the time sequence graph (stevens), all sequence numbers are monotonically increasing.



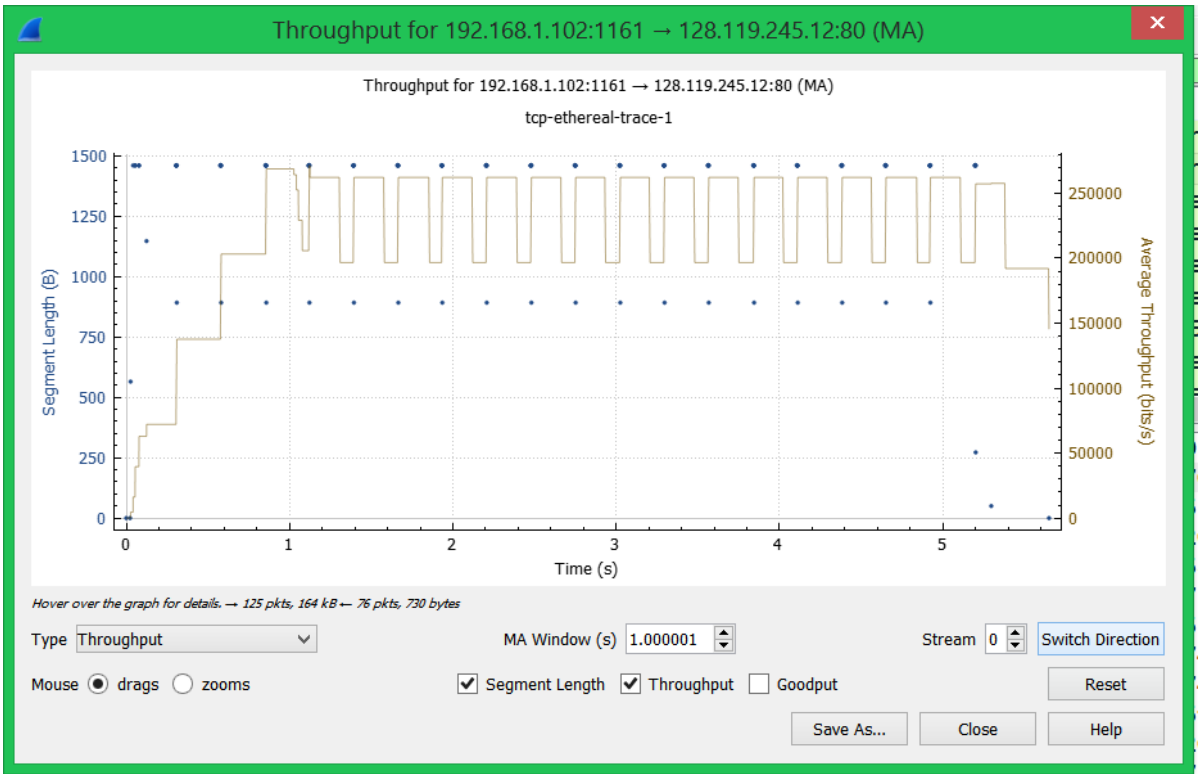
11. How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment (see Table 3.2 on page 250 in the text).

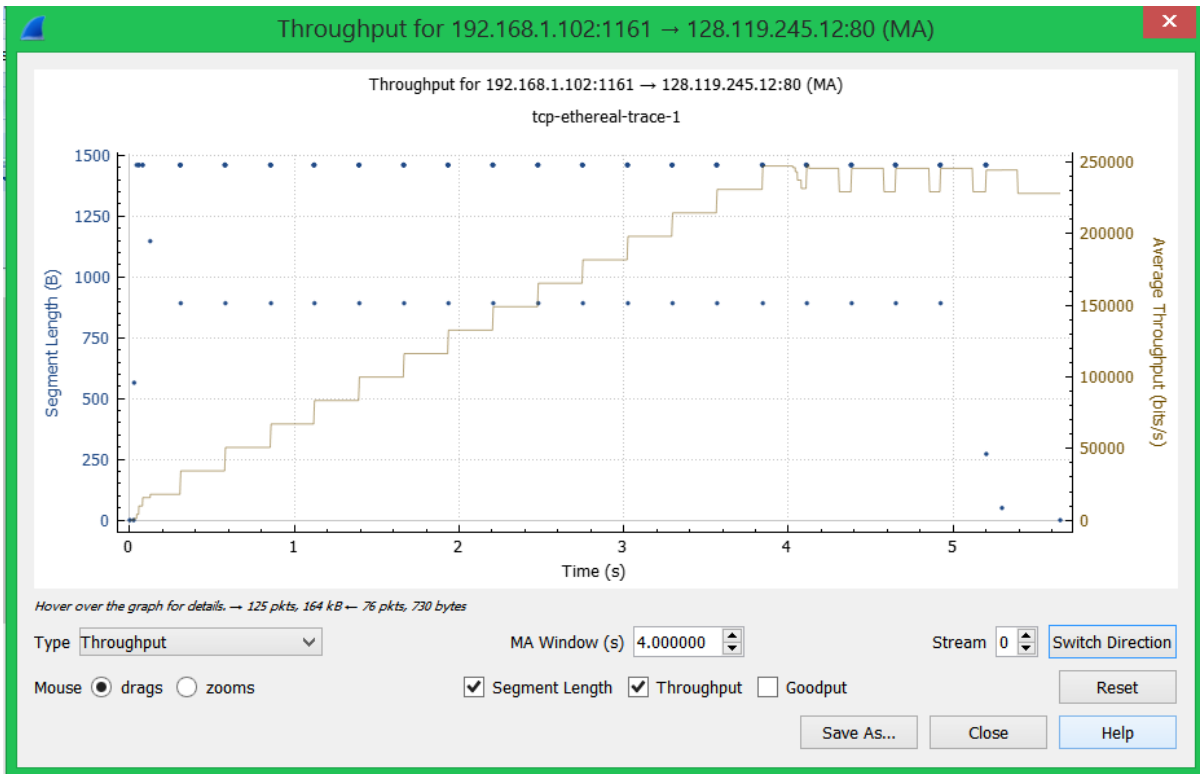
7	0.054026	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=2026 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
9	0.077294	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
10	0.077405	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
12	0.124085	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=3486 Win=11680 Len=0
13	0.124185	192.168.1.102	128.119.245.12	TCP	1201 1161 → 80 [PSH, ACK] Seq=7866 Ack=1 Win=17520 Len=1147 [TCP segment of a r
14	0.169118	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=4946 Win=14600 Len=0
15	0.217299	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=6406 Win=17520 Len=0
16	0.267802	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=7866 Win=20440 Len=0
17	0.304807	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=9013 Win=23360 Len=0
18	0.305040	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=9013 Ack=1 Win=17520 Len=1460 [TCP segment of a reasse
19	0.305813	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=10473 Ack=1 Win=17520 Len=1460 [TCP segment of a reass
20	0.306692	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=11933 Ack=1 Win=17520 Len=1460 [TCP segment of a reass
21	0.307571	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=13393 Ack=1 Win=17520 Len=1460 [TCP segment of a reass
22	0.308699	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq=14853 Ack=1 Win=17520 Len=1460 [TCP segment of a reass
23	0.309553	192.168.1.102	128.119.245.12	TCP	946 1161 → 80 [PSH, ACK] Seq=16313 Ack=1 Win=17520 Len=892 [TCP segment of a r
24	0.356437	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=10473 Win=26280 Len=0
25	0.400164	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=11933 Win=29200 Len=0
26	0.448613	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=13393 Win=32120 Len=0

The difference between the acknowledged sequence numbers of two consecutive ACKs indicates the data received by the server between these two ACKs. The receiver is ACKing every other segment. For example, segment of No. 13 acknowledged data with 1460 bytes.

12. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

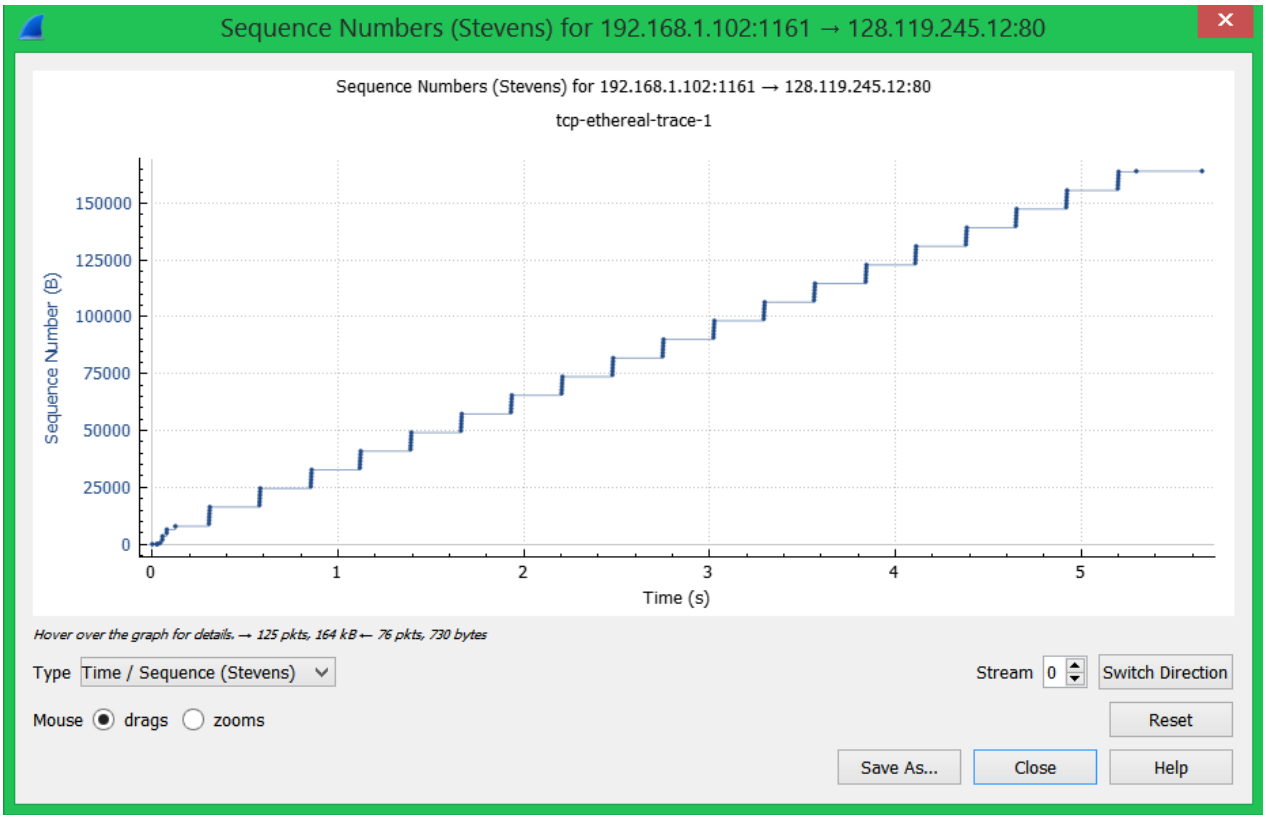
The alice.txt on the hard drive is 152,138 bytes, and the download time is 1.578736000 (First TCP segment) - 0.026477 (last ACK) = 1.552259 second. Therefore, the throughput for the TCP connection is computed as 152,138/1.552259=98010.7056876462 bytes/second.


Throughput for 192.168.1.102:1161 → 128.119.245.12:80 (MA)
Throughput for 192.168.1.102:1161 → 128.119.245.12:80 (MA)
tcp-ethereal-trace-1
Segment Length (B)
Average Throughput (bits/s)
Time (s)
Hover over the graph for details. → 125 pkts, 164 kB ← 76 pkts, 730 bytes
Type Throughput
MA Window (s) 1.000001
Stream 0
Switch Direction
Mouse drags zooms
Segment Length
Throughput
Goodput
Reset
Save As...
Close
Help

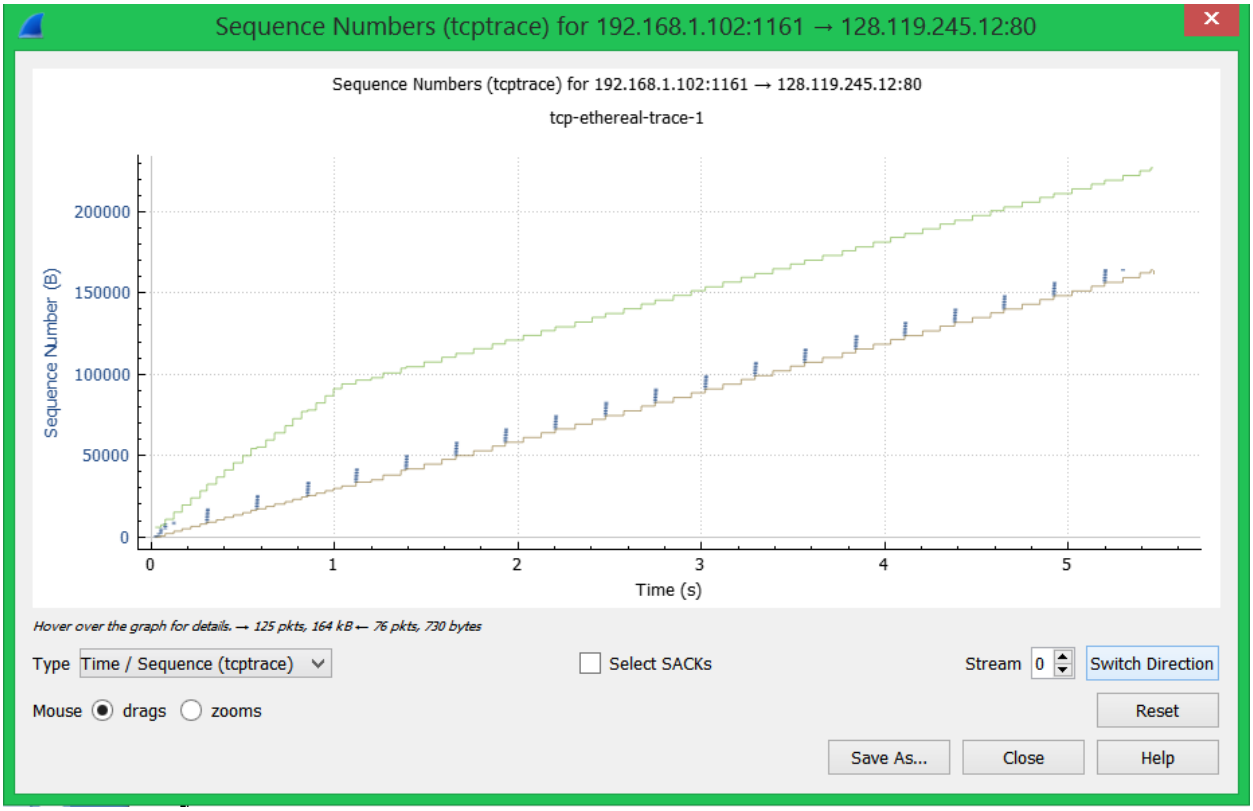

Throughput for 192.168.1.102:1161 → 128.119.245.12:80 (MA)
Throughput for 192.168.1.102:1161 → 128.119.245.12:80 (MA)
tcp-ethereal-trace-1
Segment Length (B)
Average Throughput (bits/s)
Time (s)
Hover over the graph for details. → 125 pkts, 164 kB ← 76 pkts, 730 bytes
Type Throughput
MA Window (s) 4.000000
Stream 0
Switch Direction
Mouse drags zooms
Segment Length
Throughput
Goodput
Reset
Save As...
Close
Help

13. Use the *Time-Sequence-Graph(Stevens)* plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server. Can you identify where TCP's slowstart phase begins and ends, and where congestion avoidance takes over? Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.

The slow start of the TCP seems to begin at about 0 seconds and then ends at about 0.17 seconds. Congestion avoidance takes over at about 0.52 seconds because it cut down the amount being sent.



14. Answer each of two questions above for the trace that you have gathered when you transferred a file from your computer to gaia.cs.umass.edu


Sequence Numbers (tcptrace) for 192.168.1.102:1161 → 128.119.245.12:80
Sequence Numbers (tcptrace) for 192.168.1.102:1161 → 128.119.245.12:80
tcp-ethereal-trace-1
Sequence Number (B)
200000
150000
100000
50000
0
0
1
2
3
4
5
Time (s)
Hover over the graph for details. → 125 pkts, 164 kB ← 76 pkts, 730 bytes
Type
Time / Sequence (tcptrace)
Select SACKs
Stream
0
Switch Direction
Mouse
drags
zooms
Reset
Save As...
Close
Help