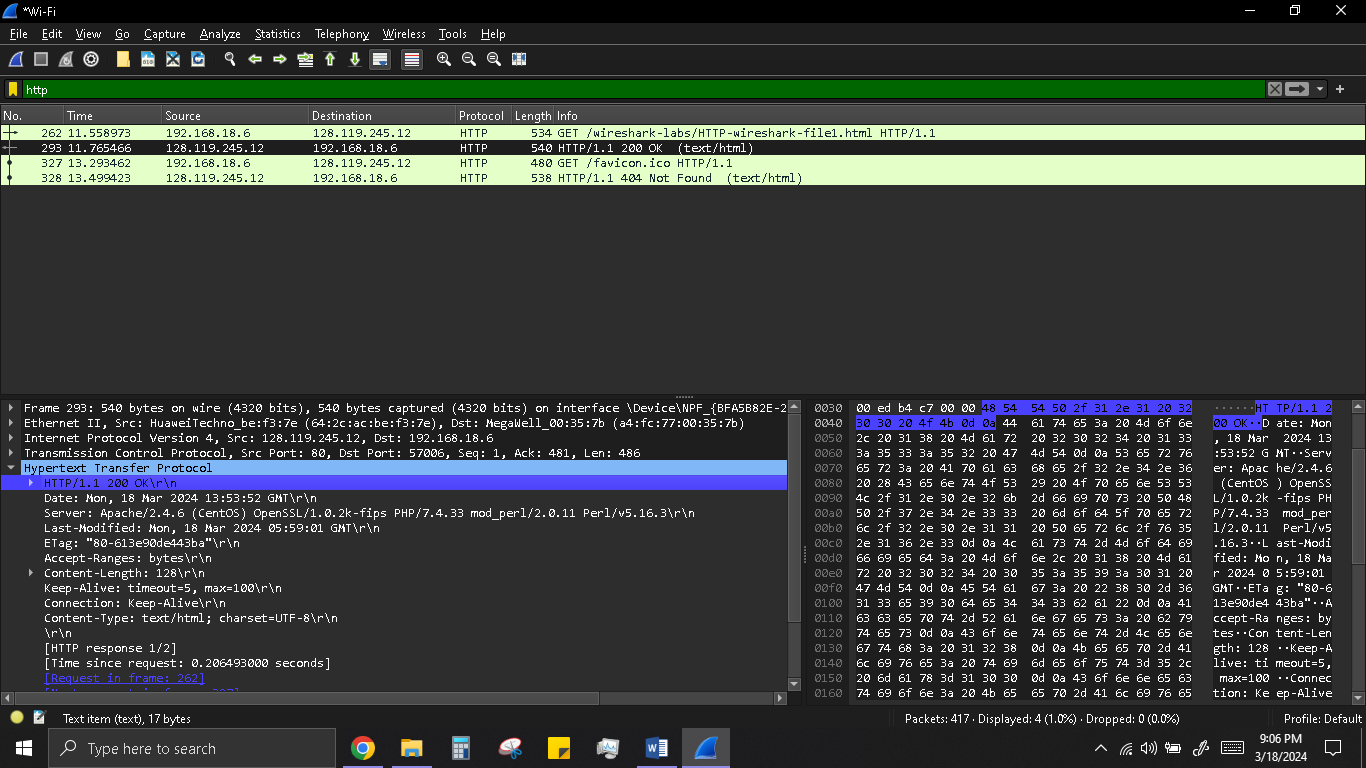
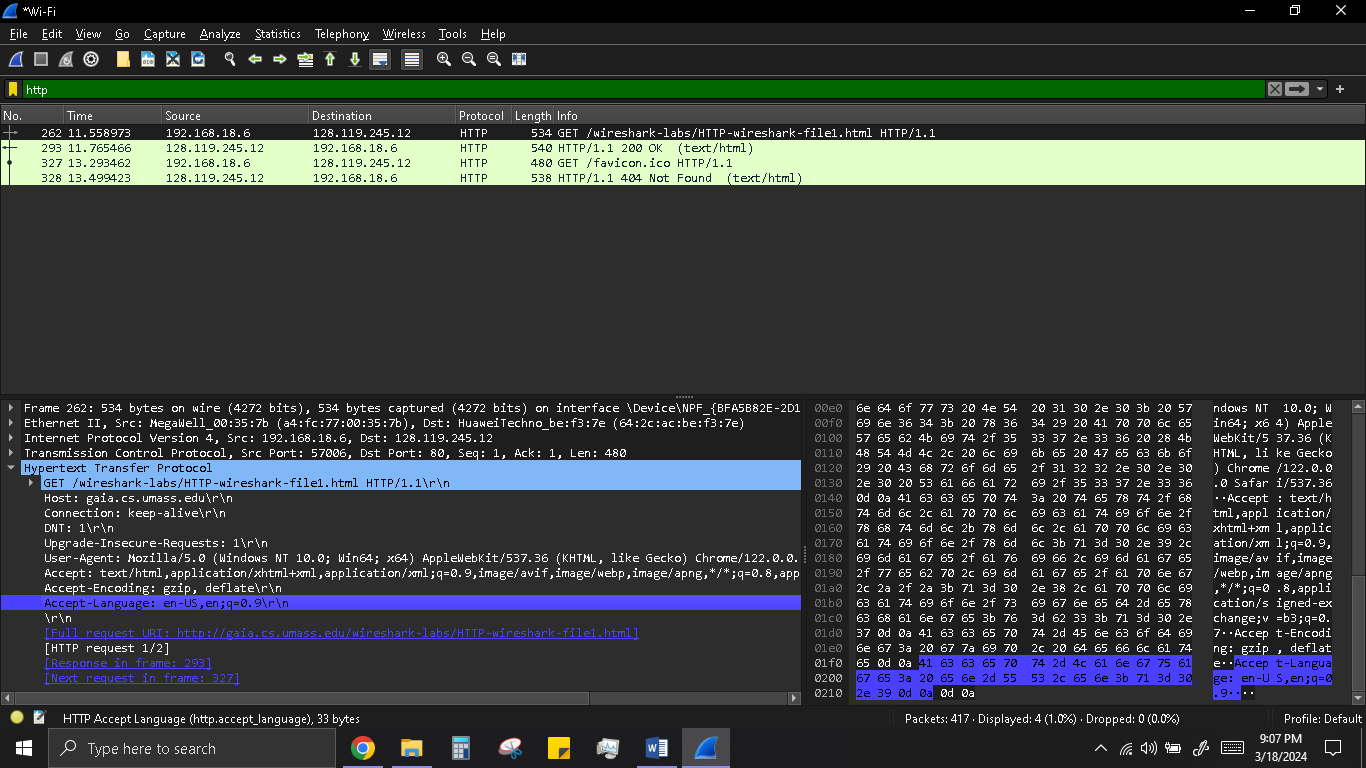
# **The Basic HTTP GET/response interaction**

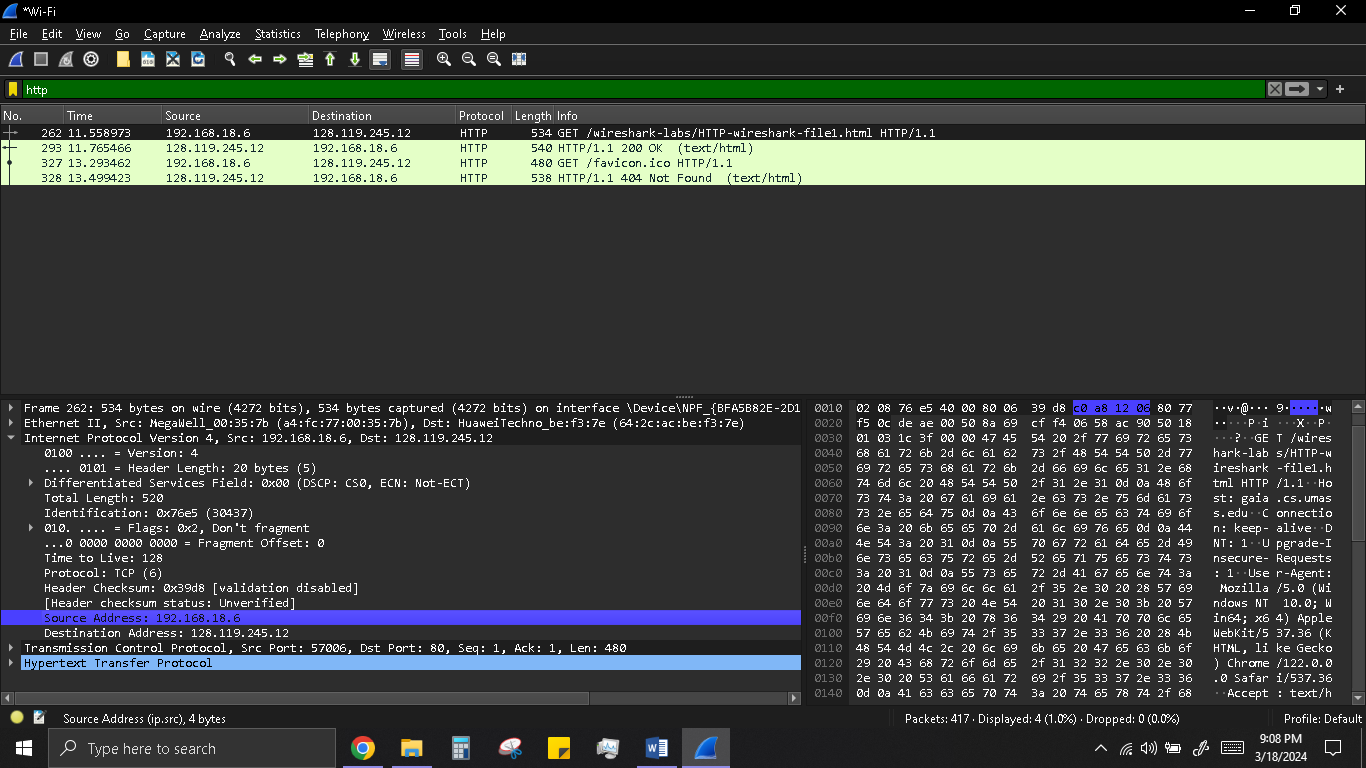
1. Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server running?



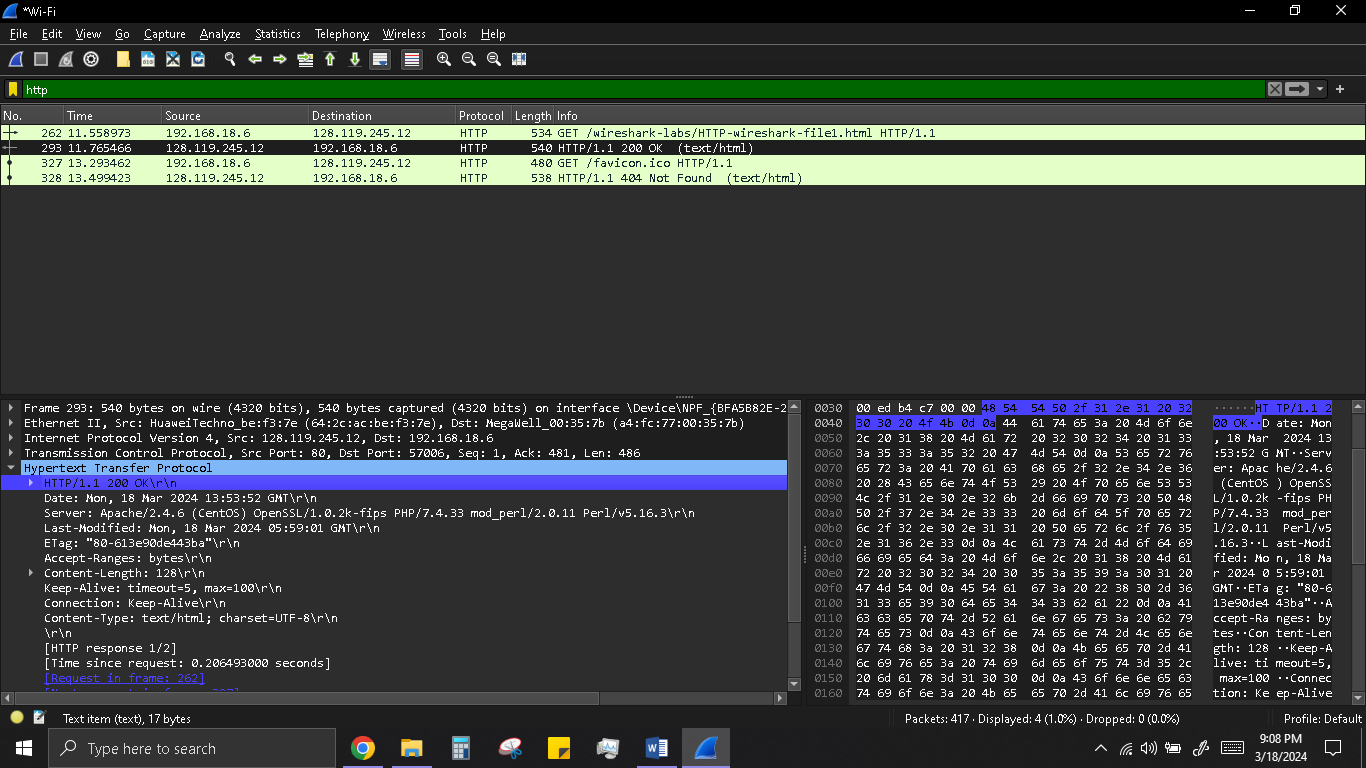
1. What languages (if any) does your browser indicate that it can accept to the server?



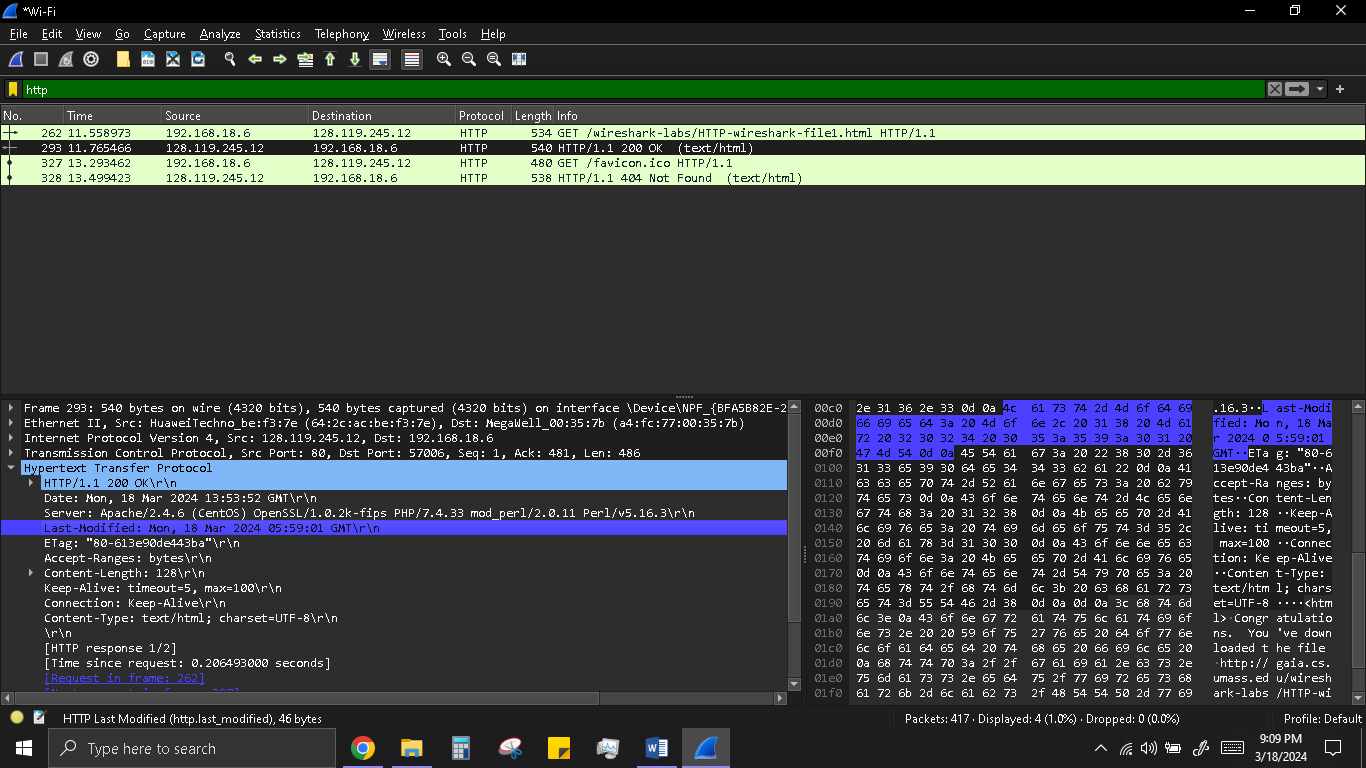
1. What is the IP address of your computer? Of the gaia.cs.umass.edu server?



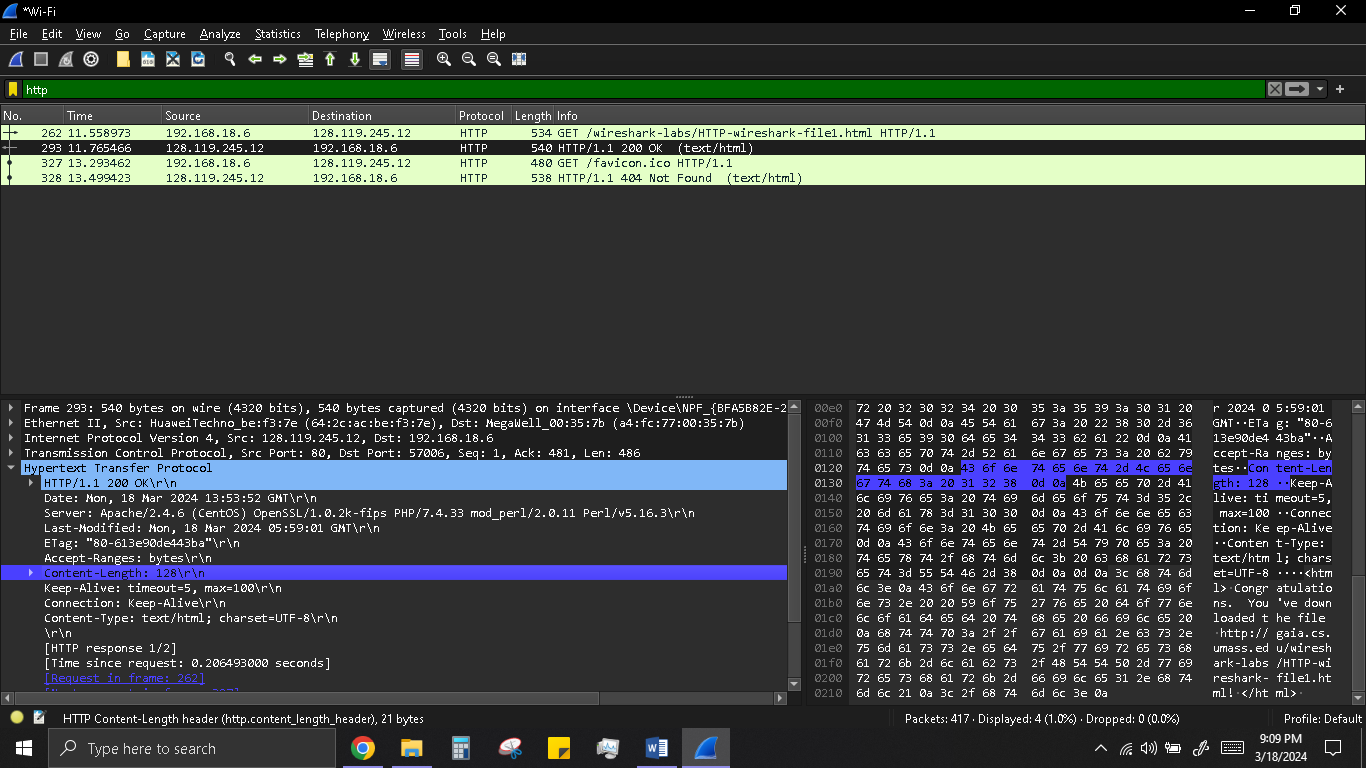
1. What is the status code returned from the server to your browser?



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



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

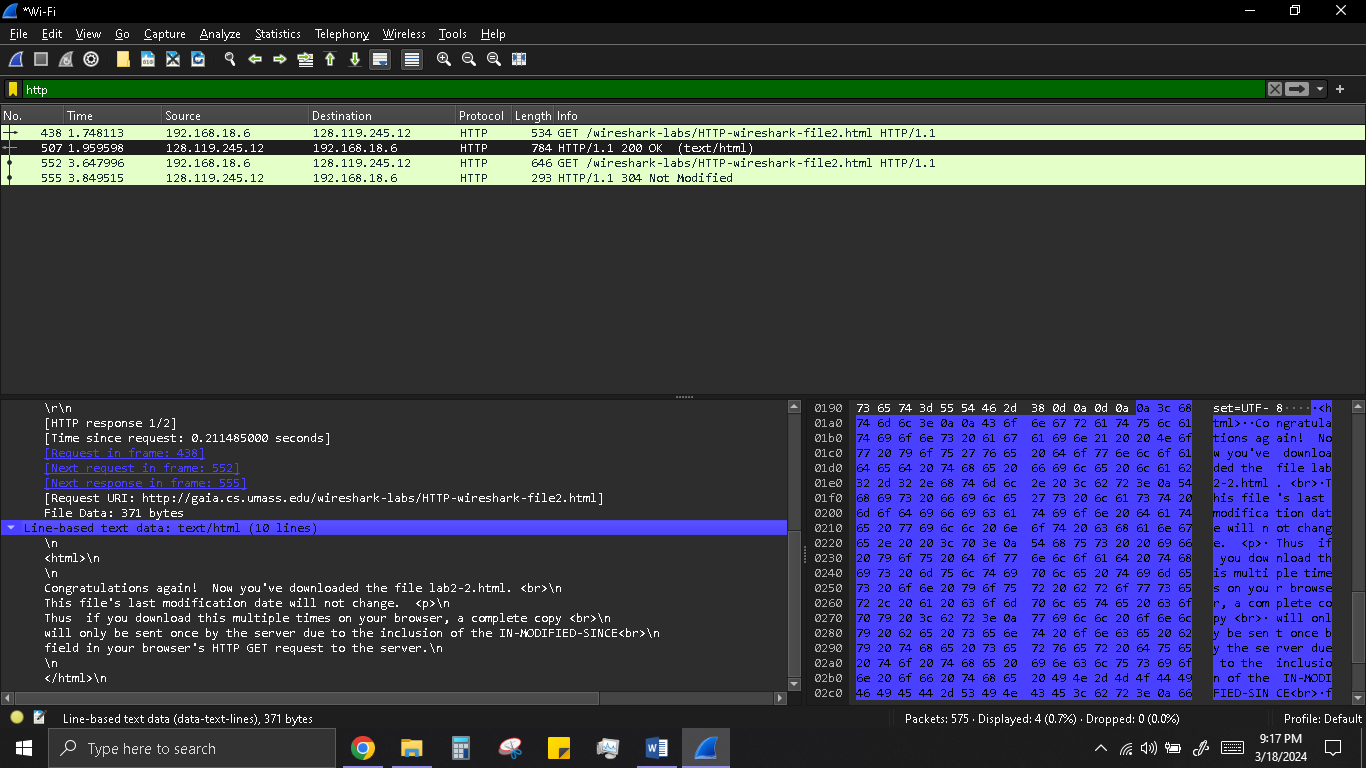


# **The HTTP Conditional GET/response interaction**

1. 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?

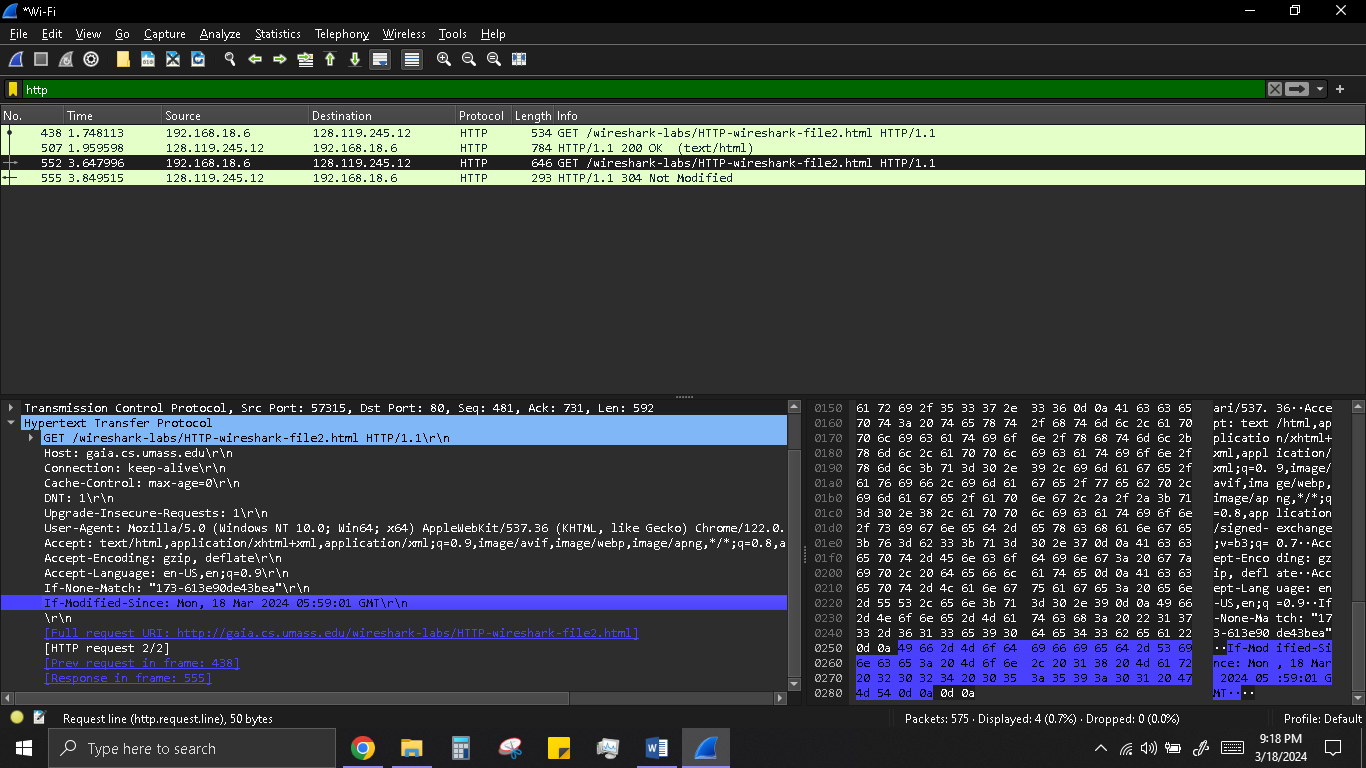
No there is no IF-MODIFIED-SINCE line in the GET message.

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

The server did explicitly return the contents of the file. Wireshark includes a section titled “Line-Based Text Data” which shows what the server sent back to my browser which specifically what the website is showed when I brought it up on my browser.

1. 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?

Yes in the second HTTP message an IF-MODIFIED-SINCE line is included. The information that follows is the date and time that I last accessed the webpage.

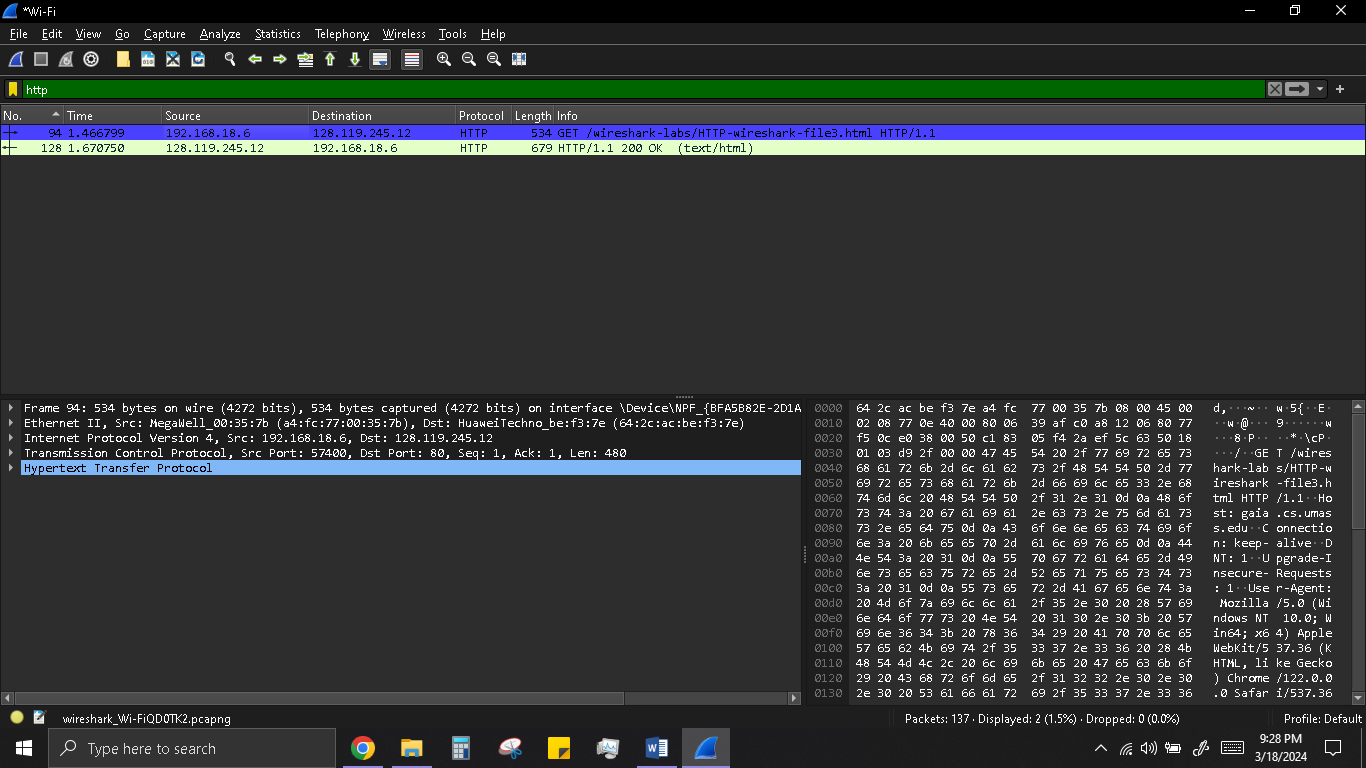


1. What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.

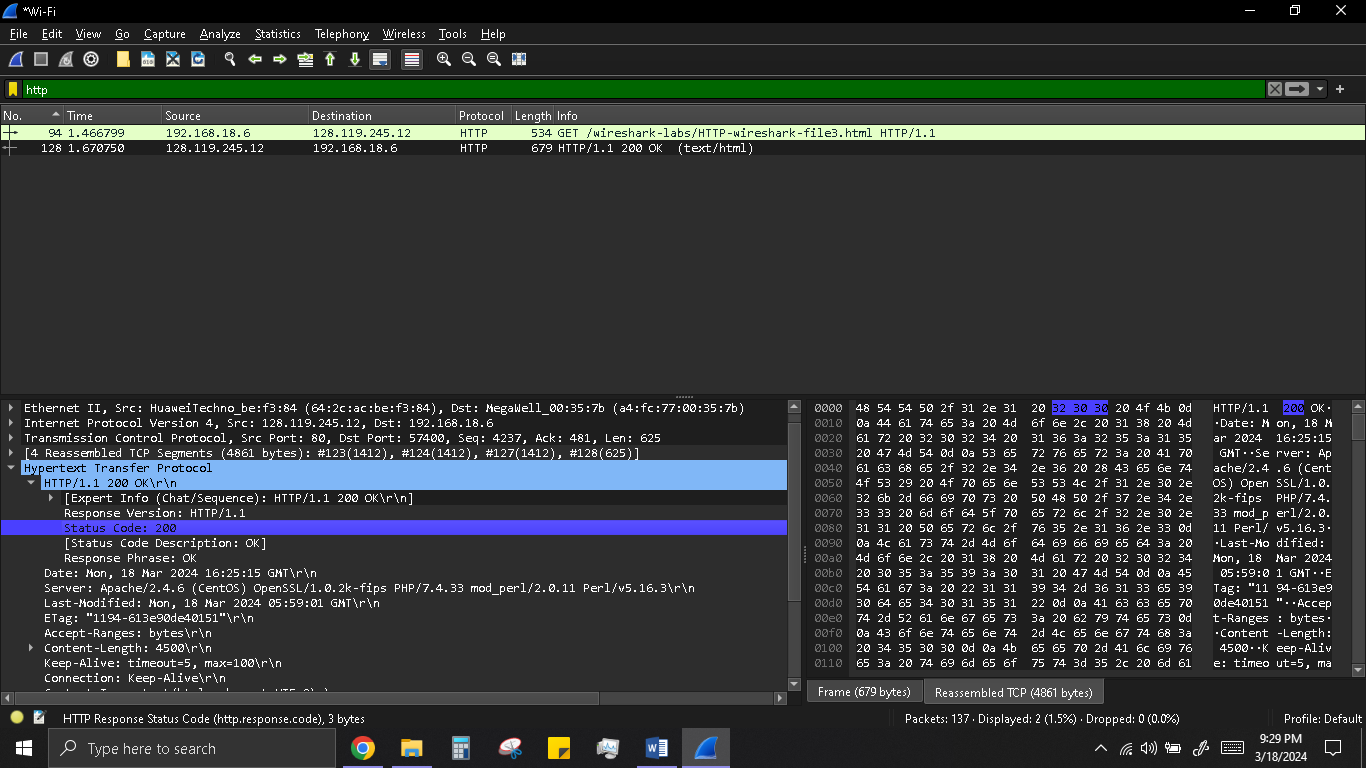
The HTTP status code is “304: Not Modified”. The server did not return the contents of the file because the browser simply retrieved the contents from its cache. Had the file been modified since it was last accessed, it would have returned the contents of the file, and instead it simply told my browser to retrieve the old file from its cached memory.

# **Retrieving Long Documents**

1. 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?

My browser only sent 1 HTTP GET request to the server. The Packet that contained the GET message was packet number 94. 

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

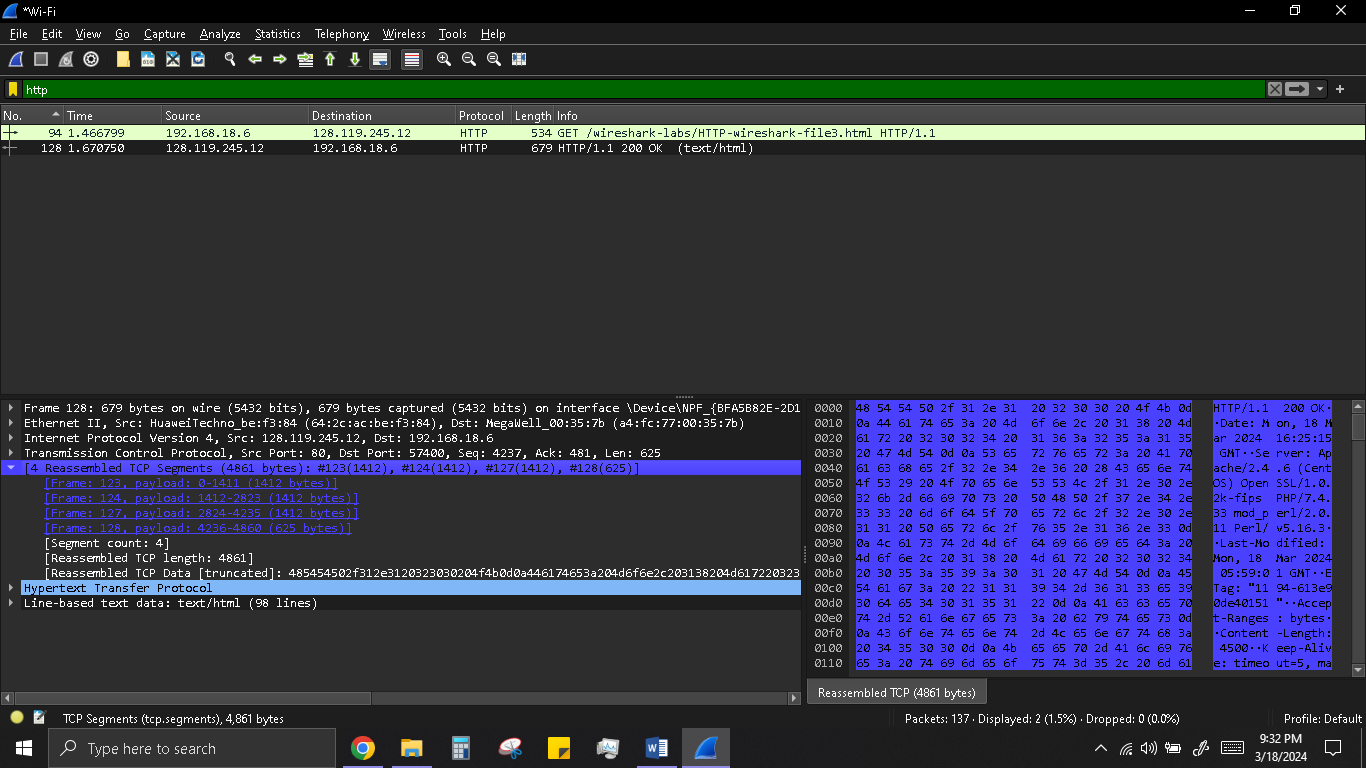
The packet that contains the status code and phrase which the server sent in response to the GET message was packet number 128. 

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

The code and phrase in the response was 200 OK. See above image.

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

The data was sent in 4 TCP segments to the browser, then reassembled.



# **HTML Documents with Embedded Objects**

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

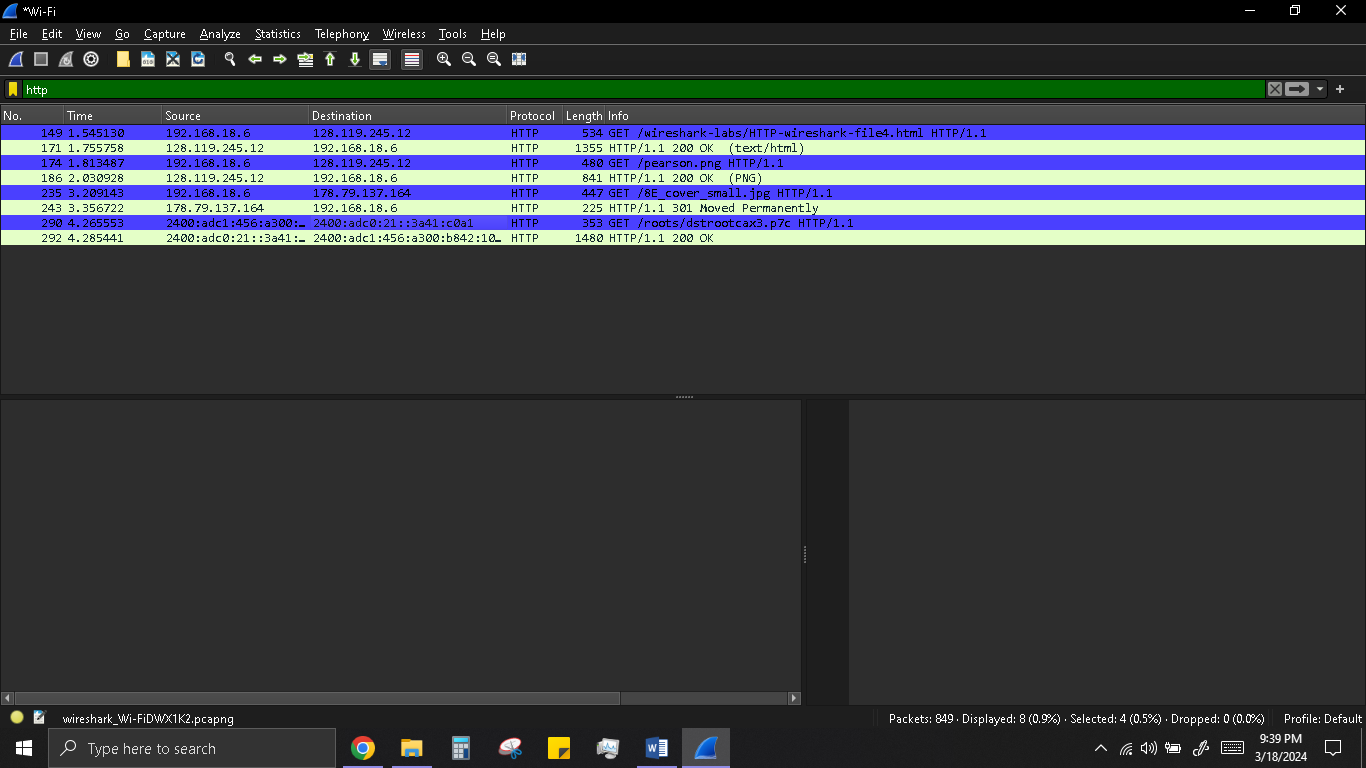
My browser sent 4 http GET message requests. One each to each for each of the following: The initial page, the Pearson logo, the cover of the Pearson book, 5th Edition, and a .p7c authentication file.

Initial Page address: 128.119.245.12

Pearson Logo: 128.119.245.12

Pearson book, 5th Edition: 178.79.137.164

Authentication File: 2400:adc0:21::3a41:c0a1



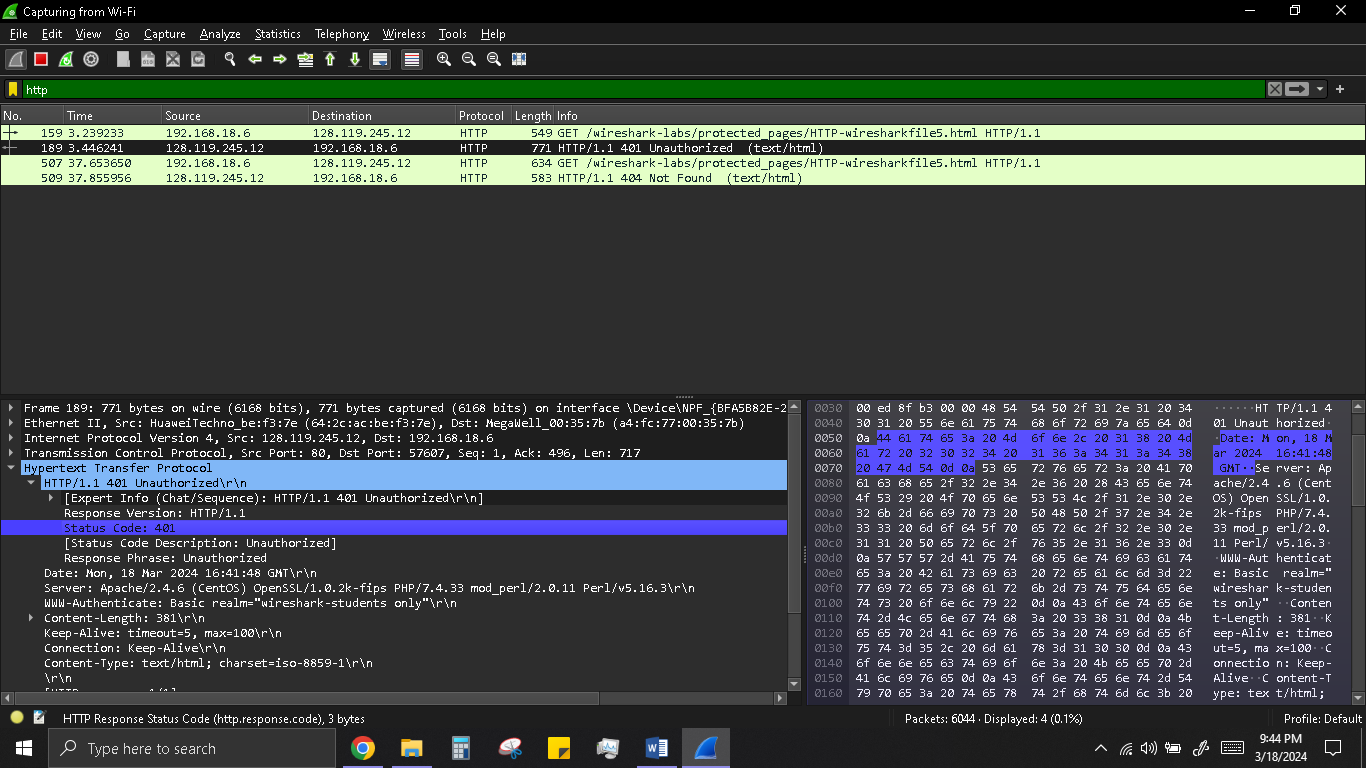
1. Can you tell whether your browser downloaded the two images serially, or whether they were downloaded from the two web sites in parallel? Explain.

The browser downloaded the two images in serially. I believe this to be the case because the first image was requested and sent before the second image was requested by the browser. Had they been running in parallel, both files would have been requested then would have returned in the same time period. In this case however, the second image was only requested after the first image came back.

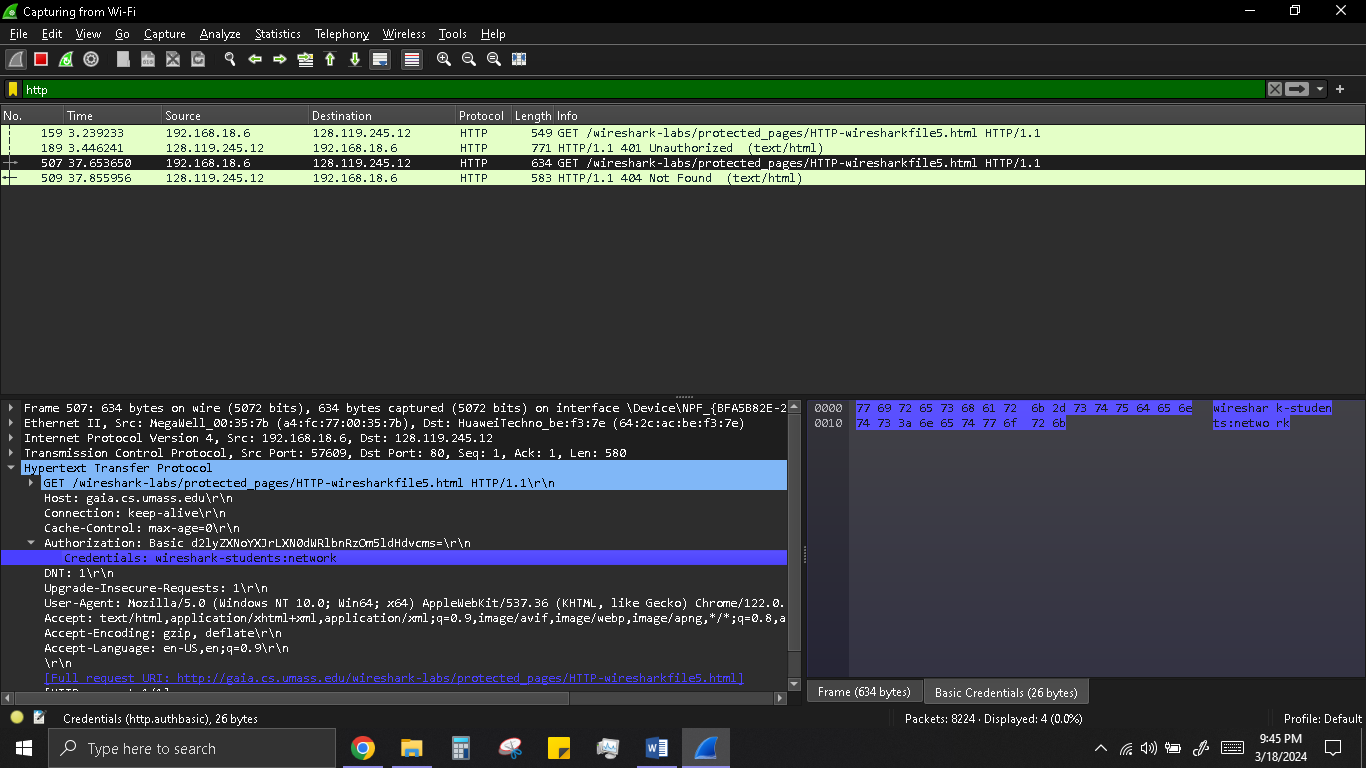
# **HTTP Authentication**

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

The servers intial response was “401 Authentication Required”.

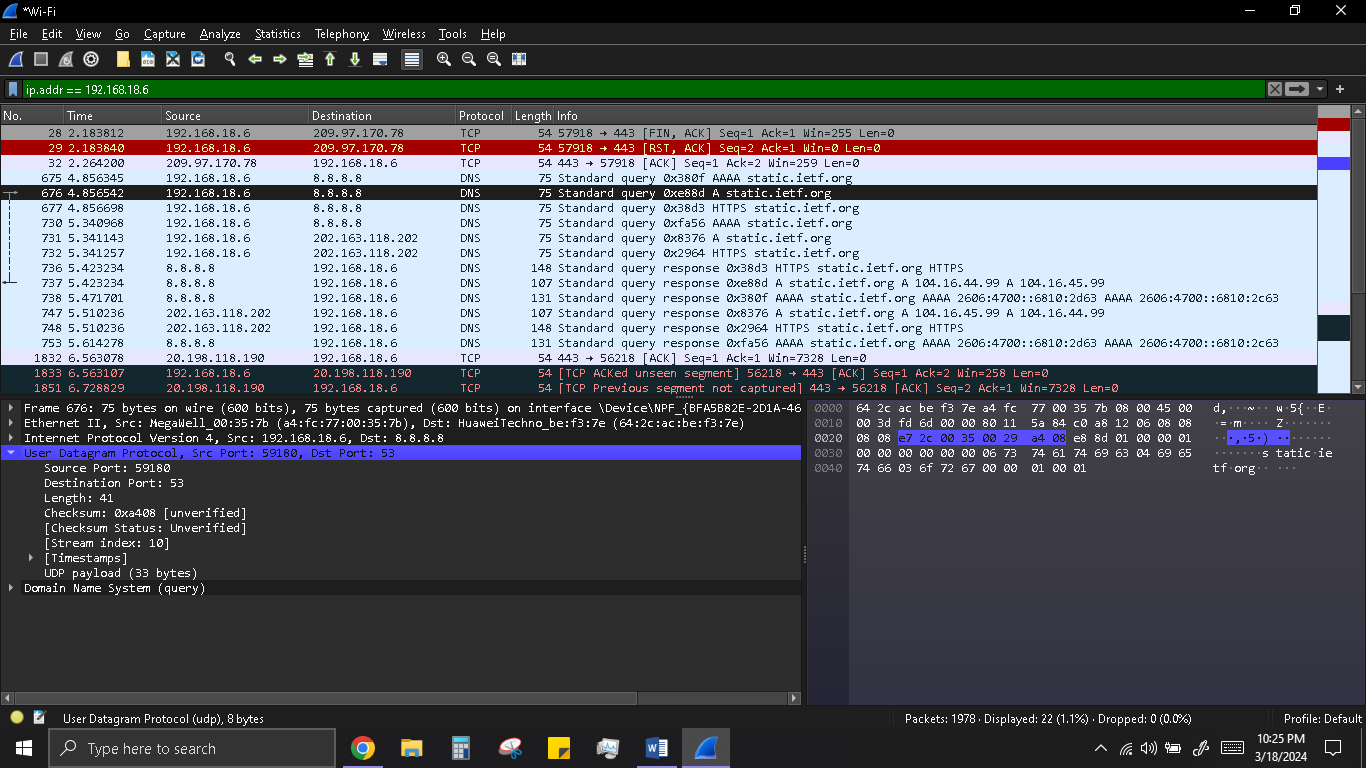


1. When your browser’s sends the HTTP GET message for the second time, what new field is included in the HTTP GET message?

The new field that is now included is the authorization field. This is included because we sent the server a username and password along with our request stating that we were authorized to receive the page. 

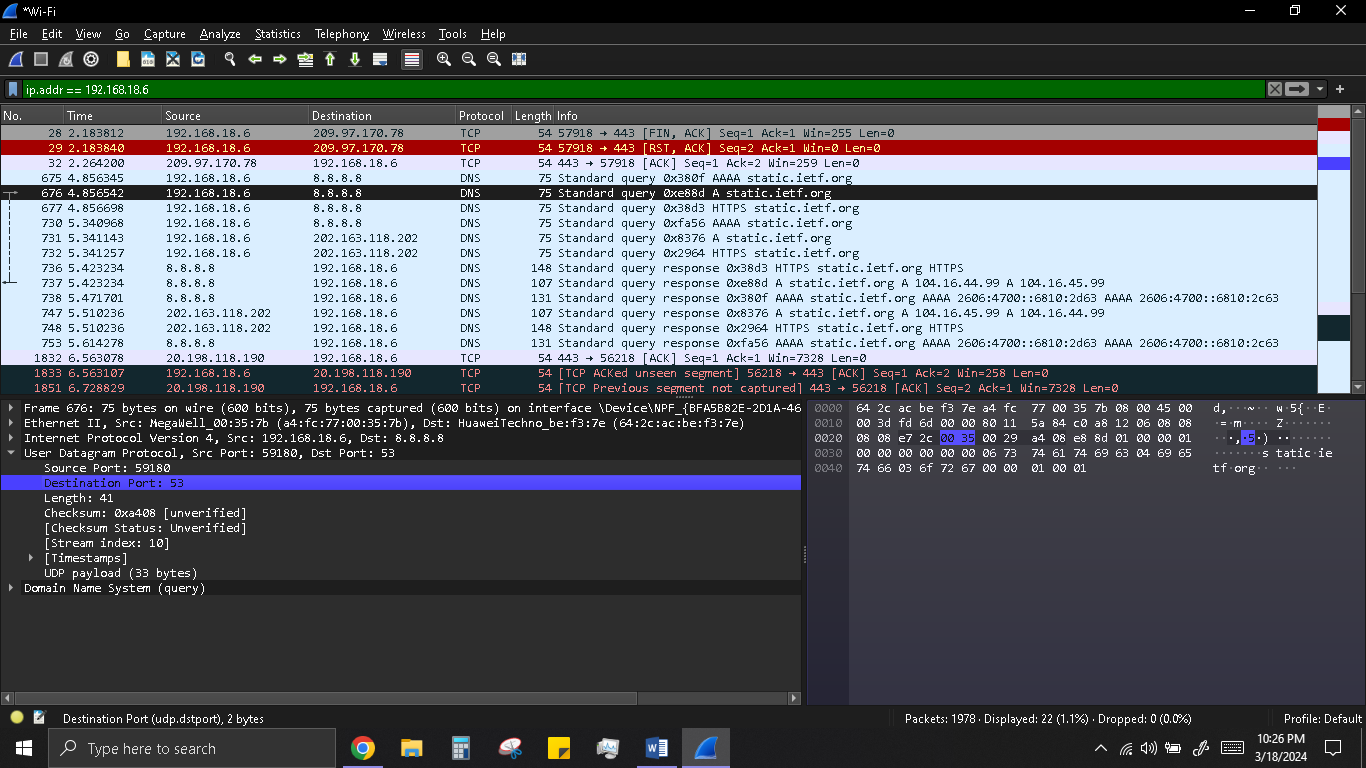
# **Network Traffic Analysis of DNS Using Wireshark**

1. Locate the DNS query and response messages. Are then sent over UDP or TCP?

The DNS query and response messages are sent over UDP. 

1. What is the destination port for the DNS query message? What is the source port of DNS response message?

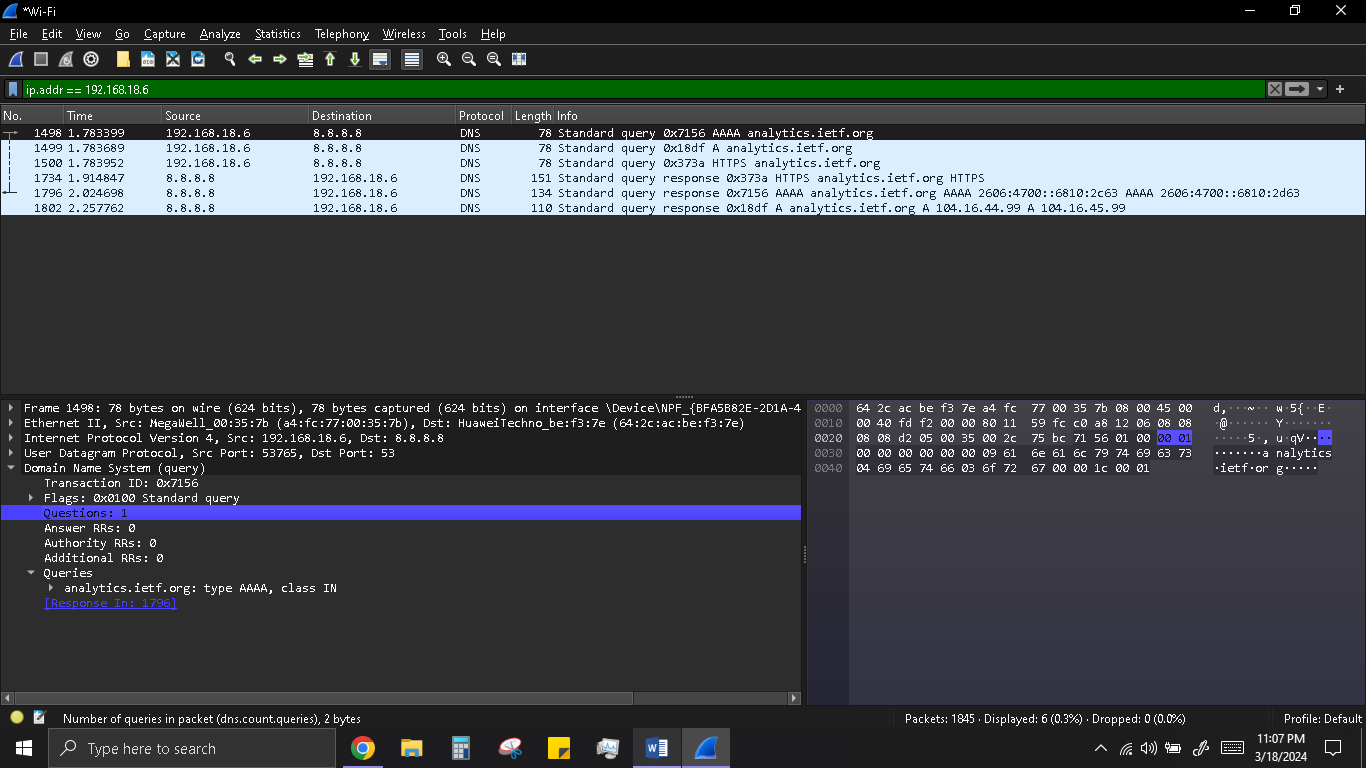
The destination port is 53

The source port is 59180

1. To what IP address is the DNS query message sent? Use ipconfig to determine the IP address of your local DNS server. Are these two IP addresses the same?

The DNS query was sent to IP address 8.8.8.8. No, it is not the same IP address as that of my local DNS server.

1. Examine the DNS query message. What “Type” of DNS query is it? Does the query message contain any “answers”?

The query message was a type “AAAA” query, but the message did not contain any “answers.” 

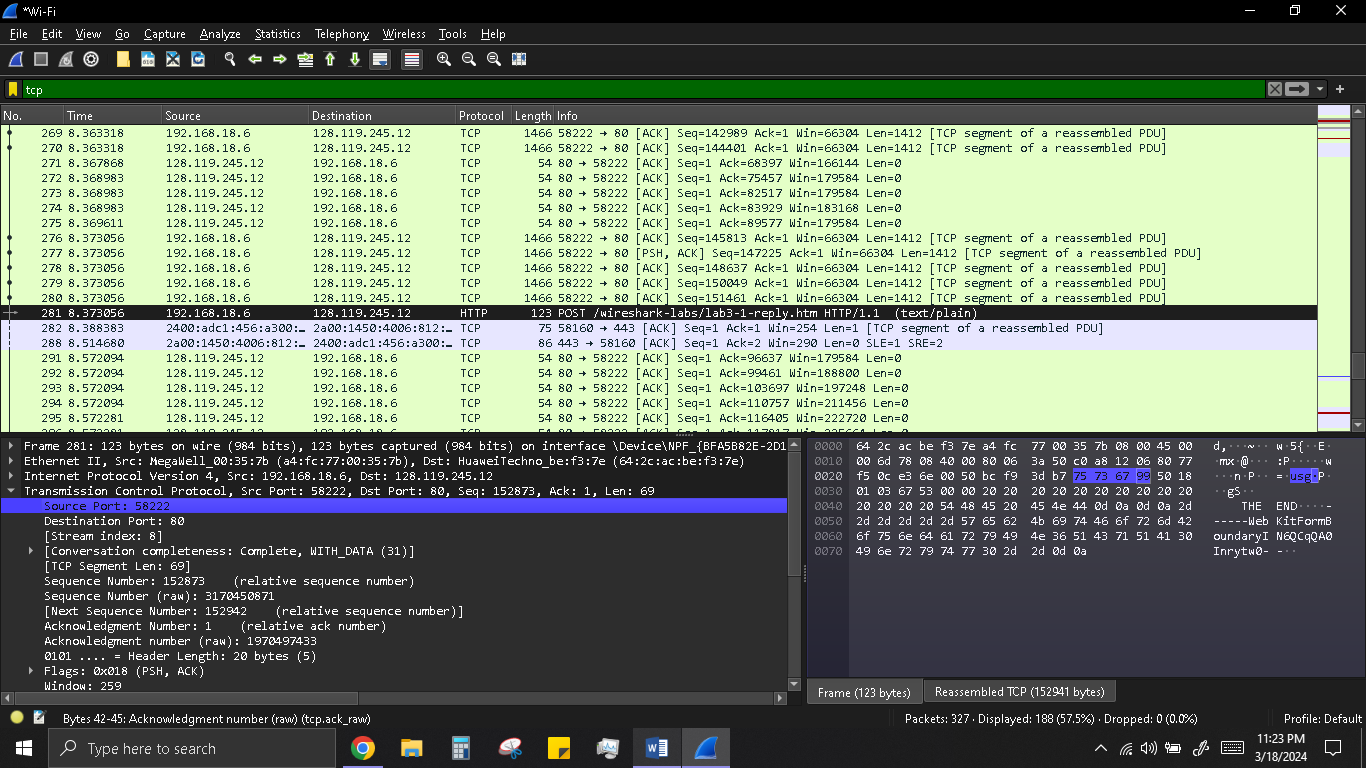
1. Examine the DNS response message. How many “answers” are provided? What do each of these answers contain?

The response contained 2 answers, both of which had an address to the website.

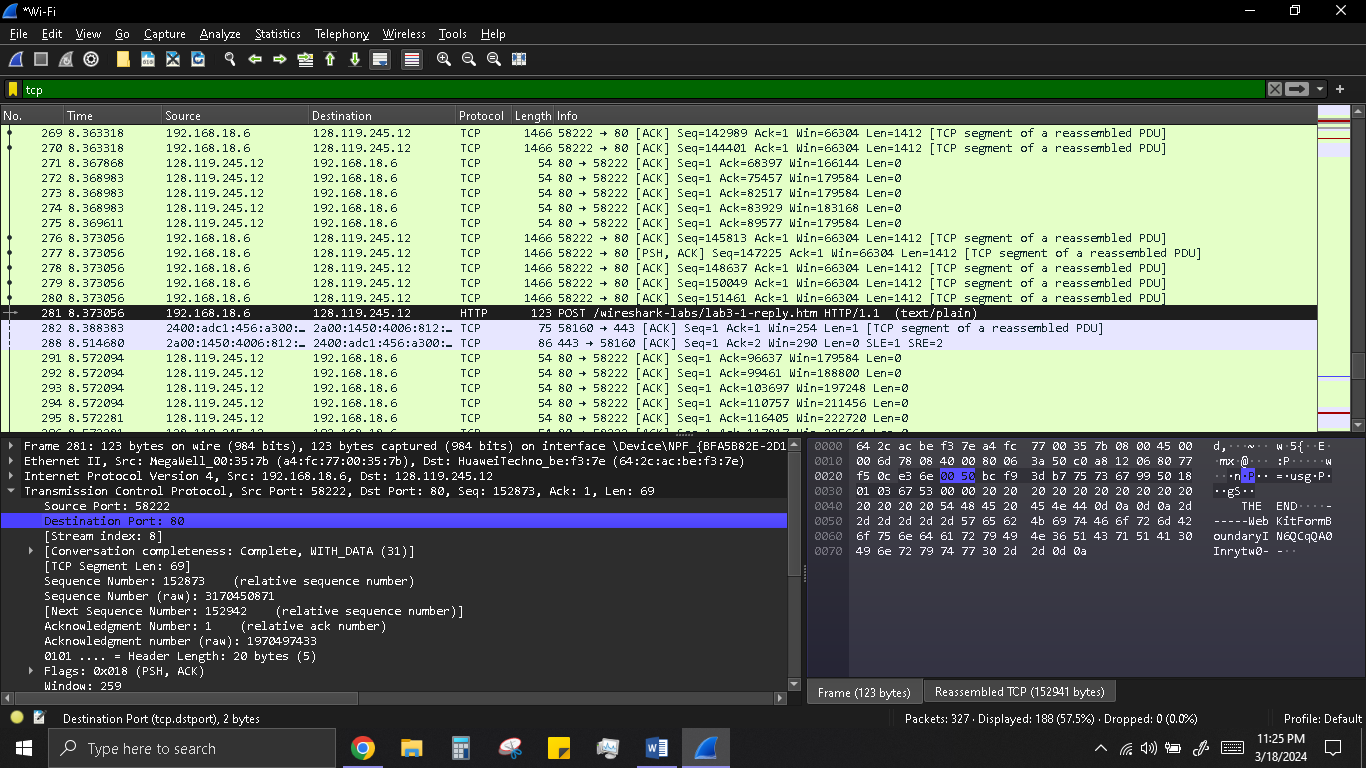
# **Capturing a Bulk TCP transfer from your computer to a Remote Server**

# **A First Look at The Captured Trace**

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”.

The source IP address was 192.168.18.6 using source port 58222. 

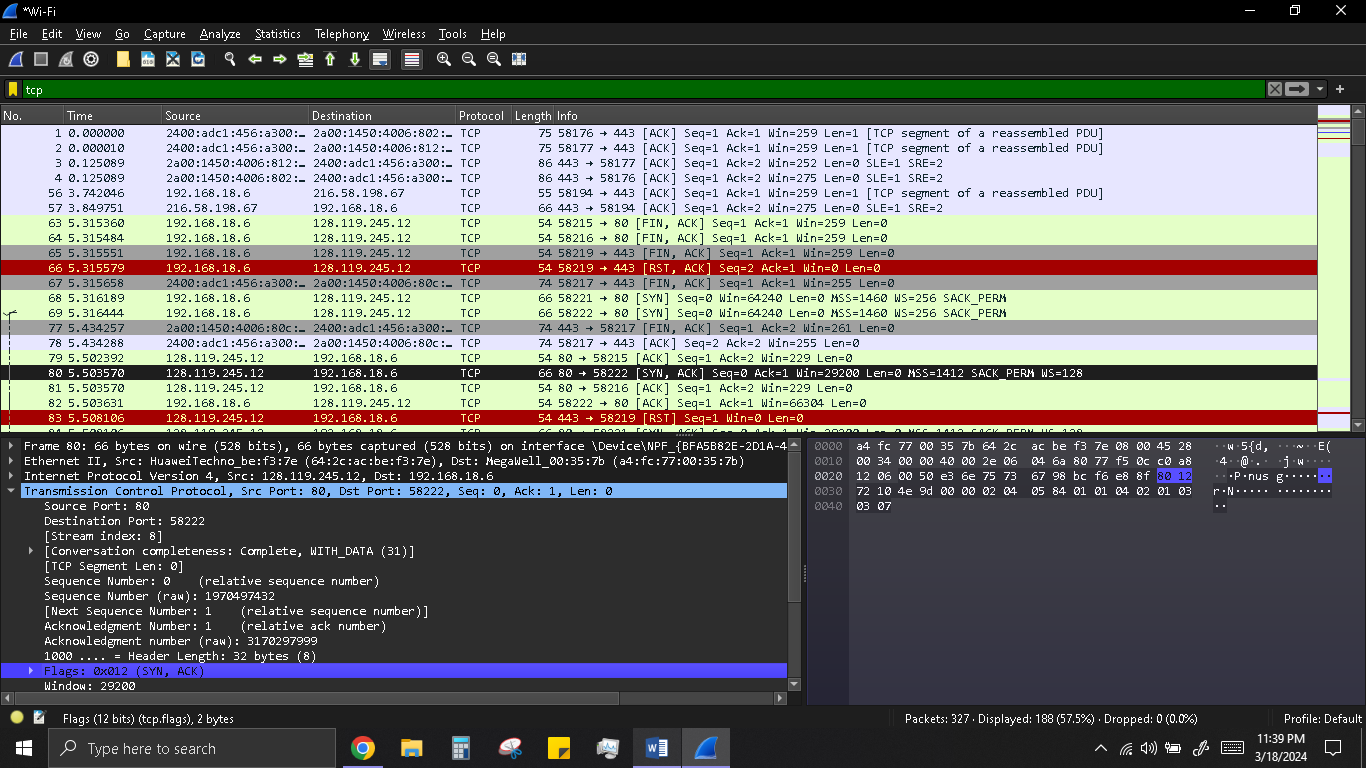
1. What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

The destination IP address is 128.119.245.12 receiving on port 80. 

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

Done above.

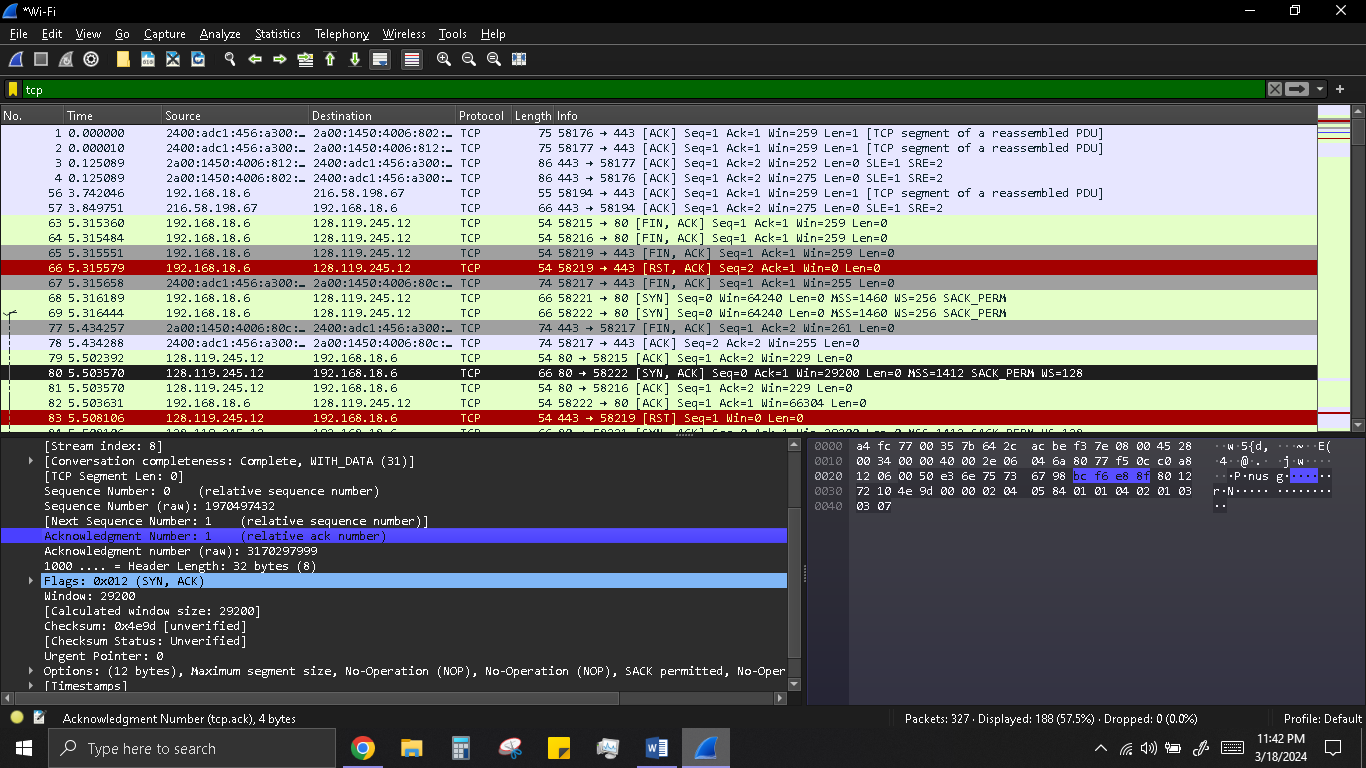
1. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?

The sequence number of the segment used to initiate the TCP connection is 0.  We can see that the message contains a SYN flag indicating that it is a SYN segment. 

1. 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?

The sequence number of the SYNACK segment is 0.

The value of the acknowledgement field is 1. This value is determined by the initial sequence number +1.

The message carries flags that show it to be a SYN ACK message. 

1. 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.

The sequence number of the TCP segment containing the HTTP Post Command is 152873. 