

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

Both my computer and server are running HTTP version 1.1.

2. **What languages (if any) does your browser indicate that it can accept to the server? In the captured session, what other information (if any) does the browser provide the server with regarding the user/browser?**

Swedish, English(US), it also provides the server, name and version of the browser running in the host machine.

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

My IP : 192.168.8.111, gaia.cs.umass.edu: 128.119.245.12

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

Status code : 200 OK

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

Last-modified: Fri, 22 Jan 2021 06:59:02 GMT

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

File Data: 128 bytes

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

NO. There is no HTTP header that is not displayed in the packet-detail window.

TASK A:

In question 1, we found out that the HTTP version can be found in all three sections (packet listing, packet details and packet bytes).

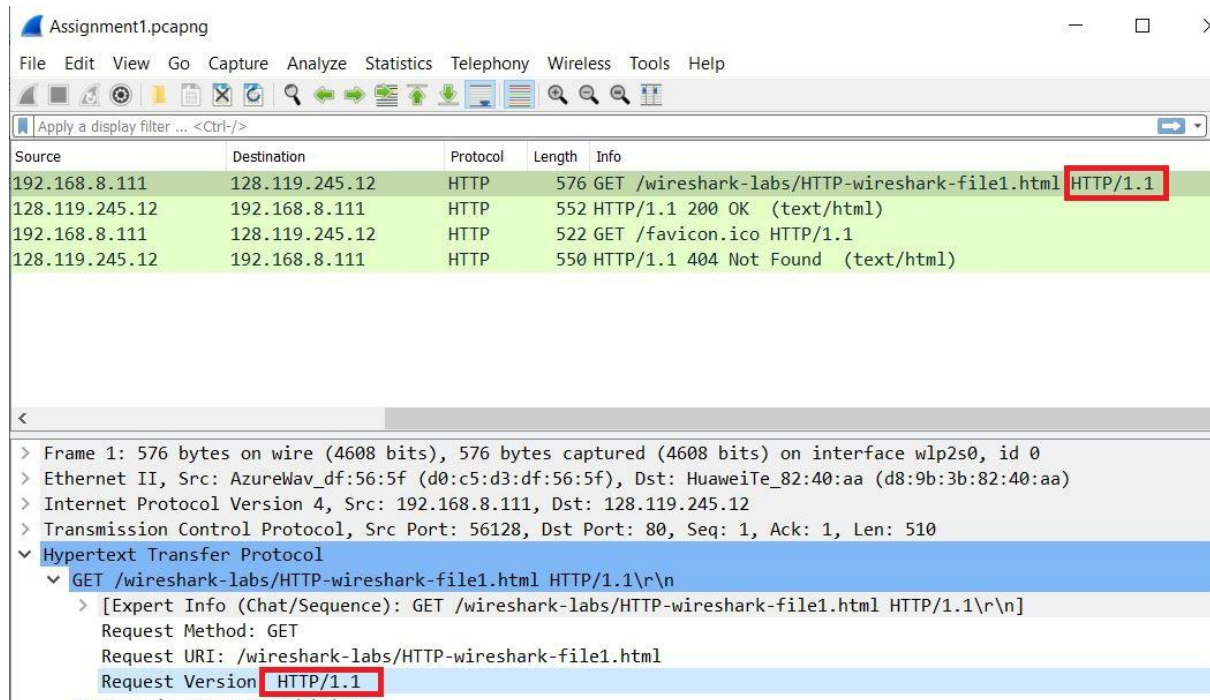


Figure 1

Same for accept-language information. This information also can be found in the packet-details window under the HTTP section. This information can only be found in the packet which my computer is sending to the server but not in the response message.

Accept-Language: sv-SE,sv;q=0.9,en-SE;q=0.8,en;q=0.7,en-US;q=0.6\r\n

Information about IP addresses are found in the packet-listing section as well as in Internet Protocol version in packet details section.

Status code indicates the result of a request. It can be found in HTTP response, in the packet-listing section. We got 200 OK, which shows that we got the result successfully.

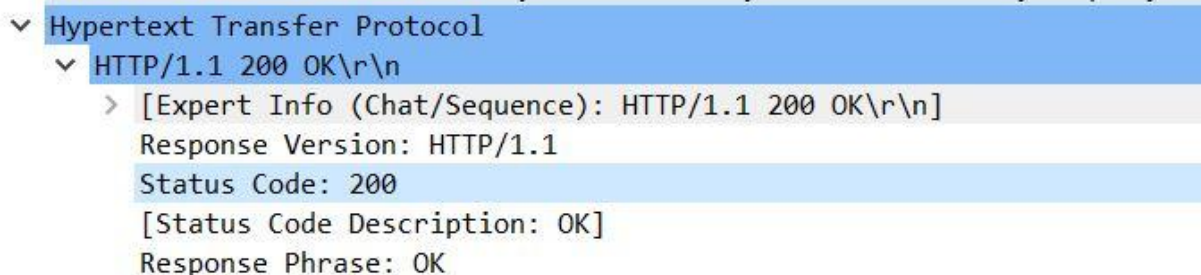


Figure 2

The answer to question 5 is obtained from the packet-detail window in which details for server response is. Following screenshot is taken from this section.

Last-Modified: Fri, 22 Jan 2021 06:59:02 GMT\r\n

The amount of data being returned can be found in the packet-details window in the HTTP section.

```

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.14 mod_perl/2.0.11 Perl/v5.16.3\r\n
Last-Modified: Fri, 22 Jan 2021 06:59:02 GMT\r\n
ETag: "80-5b977b926b6ff"\r\n
Accept-Ranges: bytes\r\n
✓ Content-Length: 128\r\n
  [Content length: 128]
Keep-Alive: timeout=5, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=UTF-8\r\n
\r\n
[HTTP response 1/2]

```

We could not find any difference by comparing the headers in the packet-detail pane and packet-content pane. All headers in packet-details could be found in the content of the packet. But according to our observation, most HTTP headers in packet-content are not displayed in packet-listing (by packet-listing pane we refer to the top pane in the Wireshark window and not the middle one). In our recent answer we were comparing top pane with bottom pane and that was why we found some differences but if we compare packet-content pane with HTTP headers in middle-pane which is packet-detail pane we will see that all HTTP headers match in both panes.

0040	45 90 47 45 54 20 2f 77	69 72 65 73 68 61 72 6b	E·GET /w ireshark
0050	2d 6c 61 62 73 2f 48 54	54 50 2d 77 69 72 65 73	-labs/HT TP-wires
0060	68 61 72 6b 2d 66 69 6c	65 31 2e 68 74 6d 6c 20	hark-file1.html
0070	48 54 54 50 2f 31 2e 31	0d 0a 48 6f 73 74 3a 20	HTTP/1.1 ··Host:
0080	67 61 69 61 2e 63 73 2e	75 6d 61 73 73 2e 65 64	gaia.cs.umass.ed
0090	75 0d 0a 43 6f 6e 6e 65	63 74 69 6f 6e 3a 20 6b	u··Conne ction: k
00a0	65 65 70 2d 61 6c 69 76	65 0d 0a 55 70 67 72 61	ee-p-aliv e··Upgra
00b0	64 65 2d 49 6e 73 65 63	75 72 65 2d 52 65 71 75	de-Insec ure-Requ
00c0	65 73 74 73 3a 20 31 0d	0a 55 73 65 72 2d 41 67	ests: 1··User-Ag
00d0	65 6e 74 3a 20 4d 6f 7a	69 6c 6c 61 2f 35 2e 30	ent: Moz illa/5.0
00e0	20 28 58 31 31 3b 20 4c	69 6e 75 78 20 78 38 36	(X11; L inux x86
00f0	5f 36 34 29 20 41 70 70	6c 65 57 65 62 4b 69 74	_64) App leWebKit

Figure 3

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?

No, we could not find any header named "IF-MODIFIED-SINCE".

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

By clicking the server response, there is a new row added which explicitly returns the contents of the file.

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?

Yes, and it has the following information: “If-Modified-Since: Sun, 24 Jan 2021 06:59:01 GMT”

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.

Status code: 304. No, the server did not return any content to the client because the content of the file in the HTTP server was not modified since “Sun, 24 Jan 2021 06:59:01 GMT”.

TASK B:

In question 8, there was no “IF-MODIFIED-SINCE” header to be found. header line is exactly equal to the value of the “Last-Modified”.

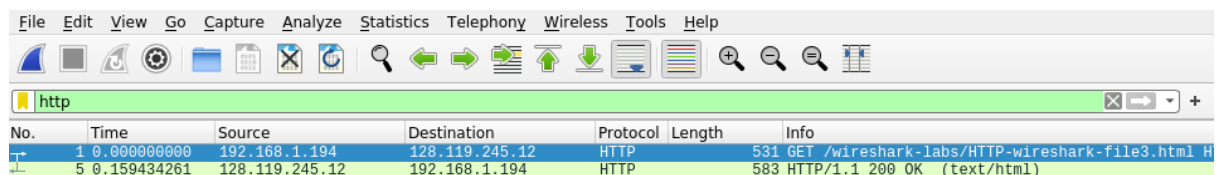
In question 9, We see that a new row called “Line-based text data” appears, which explicitly shows the content of the file.

In question 10, now in the second request we see that “IF-MODIFIED-SINCE” exists there which shows the date and time of the last version that was opened by the client. This second GET request attaches this information to the HTTP message to ask the server to send a newer version of the content if it exists.

In question 11, we got the code 304, which tells the cache that it can go ahead and forward its cached copy of the object to the requesting browser.

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

1 GET request message. Packet number 1 contains the GET message for the Bill of Rights.



The image shows a Wireshark packet capture window. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons. A filter bar shows 'http'. The packet list pane displays two packets:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.1.194	128.119.245.12	HTTP	531	GET /wireshark-labs/HTTP-wireshark-file3.html H
5	0.159434261	128.119.245.12	192.168.1.194	HTTP	583	HTTP/1.1 200 OK (text/html)

figure 4

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

Packet number 5, and it contains 200 (OK) status code.

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

4 TCP segments.

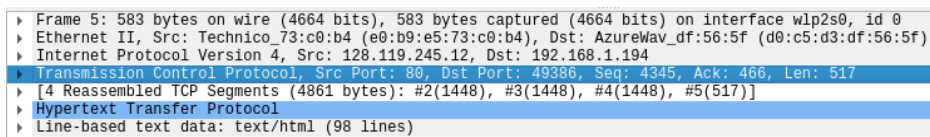


Figure 5 shows a Wireshark packet capture details view for packet 5. The packet is 583 bytes on wire (4664 bits) and 583 bytes captured (4664 bits) on interface wlp2s0, id 0. The layers shown are Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and Hypertext Transfer Protocol. The TCP layer shows [4 Reassembled TCP Segments (4861 bytes): #2(1448), #3(1448), #4(1448), #5(517)] and the HTTP layer shows Line-based text data: text/html (98 lines).

Figure 5

- 15. Is there any HTTP header information in the transmitted data associated with TCP segmentation? For this question you may want to think about at what layer each protocol operates, and how the protocols at the different layers interoperate.**

No, there is not.

TASK C:

In question 12, the browser sent only one HTTP GET request to the server as shown in figure 4.

In question 13, we got status code 200(OK) that indicates the result of a request. In our case it shows that we got the result successfully.

In question 14, there were 4 data-containing TCP segments needed to carry the single HTTP response and the text of the BoR.

In question 15, *HTTP is located in the application layer and has no header information about TCP layer which is the layer underneath.*

- 16. How many HTTP GET request messages were sent by your browser? To which Internet addresses were these GET requests sent?**

3 GET request messages sent. All of them were sent to 128.119.245.12.

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

Parallel

TASK D:

In question 16, there were 3 GET request messages sent in total but to two different addresses. Two of them are sent to get the website content and the image which is embedded in the same server (pearson.png), and one of them is sent to another server where the second image (8E_cover_small.jpg) is located.

No.	Time	Source	Destination	Protocol	Length	Info
1589	2021-02-03 15:56:09,285450	192.168.0.122	128.119.245.12	HTTP	530	GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1.1
1617	2021-02-03 15:56:09,411574	128.119.245.12	192.168.0.122	HTTP	1355	HTTP/1.1 200 OK (text/html)
1624	2021-02-03 15:56:09,456545	192.168.0.122	128.119.245.12	HTTP	476	GET /pearson.png HTTP/1.1
1636	2021-02-03 15:56:09,503884	192.168.0.122	178.79.137.164	HTTP	443	GET /8E_cover_small.jpg HTTP/1.1
1643	2021-02-03 15:56:09,543400	178.79.137.164	192.168.0.122	HTTP	225	HTTP/1.1 301 Moved Permanently
1655	2021-02-03 15:56:09,583539	128.119.245.12	192.168.0.122	HTTP	745	HTTP/1.1 200 OK (PNG)

We observed that the GET requests are sent to two different addresses in order to get HTML site and two images. Because each GET request for downloading the images has a different destination so we guess that those images have been sent to our browser in parallel and independently of each other.

Figure 6

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

Status code: 401 Unauthorized.

374	2021-01-24 16:42:44,679904	192.168.0.122	128.119.245.12	HTTP	546	GET /wireshark-labs/protected_pages/H
388	2021-01-24 16:42:44,807081	128.119.245.12	192.168.0.122	HTTP	771	HTTP/1.1 401 Unauthorized (text/html)
1379	2021-01-24 16:42:58,277995	192.168.0.122	128.119.245.12	HTTP	631	GET /wireshark-labs/protected_pages/H
1398	2021-01-24 16:42:58,411034	128.119.245.12	192.168.0.122	HTTP	544	HTTP/1.1 200 OK (text/html)

> Internet Protocol Version 4, Src: 128.119.245.12, Dst: 192.168.0.122
> Transmission Control Protocol, Src Port: 80, Dst Port: 55444, Seq: 1, Ack: 493, Len: 717
> Hypertext Transfer Protocol
> HTTP/1.1 401 Unauthorized\r\n
> [Expert Info (Chat/Sequence): HTTP/1.1 401 Unauthorized\r\n]
> Response Version: HTTP/1.1
> Status Code: 401
> [Status Code Description: Unauthorized]
> Response Phrase: Unauthorized

Figure 7

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

*A new field called **Authorization: Basic** is added*

```
✓ Authorization: Basic d2lyZXNoYXJrLXN0dWR1bnRzOm5ldHdvcm5=\r\n
  Credentials: wireshark-students:network
```

Figure 8

TASK E:

In question 18, we got **401 Unauthorized** status code from the server, which means that we do not have the required authorization to access the server.

In question 19, the added field includes credentials for entering the website. This Get request with correct authentications let us receive the content of the page.

20. What does the "Connection: close" and "Connection: keep-alive" header field imply in HTTP protocol? When should one be used over the other?

Connection: close as it sounds will close the connection between server and client which means that you may need to provide authentication each time you are sending a GET request to the same page. It is called a non-persistent connection in computer networking. Server will terminate the TCP connection right after sending a response back to the client.

Connection : alive keep the connection so you do not need to use authentication for multiple HTTP requests. This type of connection is the default mode of HTTP connection but application developers can change it to non-persistent connection if needed.