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

   We can see in the GET request that the http version is 1.1 and in the response the http version is also 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?

   In the GET trace we see “accept-language: en-us” so it accepts English. It also provides encoding, charset, file types and user agent.

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

   Gaia: 128.119.245.12

   Self: 192.168.1.102

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

   Its 200 OK seen in the headline of the http response.

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

   “Last-modified: tue, 23 sep 2003 05:29:00 GMT” seen in the response trace.

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

   Seen at the bottom of the response trace, its 73 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 we didn’t find any.

Discussion: We have been examining the data sent between a browser and a webserver. There’s a lot of information but it’s structured in a way that makes it easy to sift through it to find what you’re looking for.

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?

   There is no such header in the first HTTP GET.

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

   We can find the line-based text data in the response, so yes the server returned the contents.

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?

    There is such a header in this HTTP GET. The information that follows is: “Tue, 23 sep 2003 05:35:00 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.

    The status code is 304 not modified, the server did not return the contents this time since it had not modified it since the browser last downloaded the contents.

Discussion: If a file has been modified, the updated version must be downloaded by the browser, but if it has not been modified then the server will just refer to the browser’s cache and confirm that it’s still up to date.

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?

    Only one request was sent. Packet nr.8 was the GET for Bill of Rights.

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?

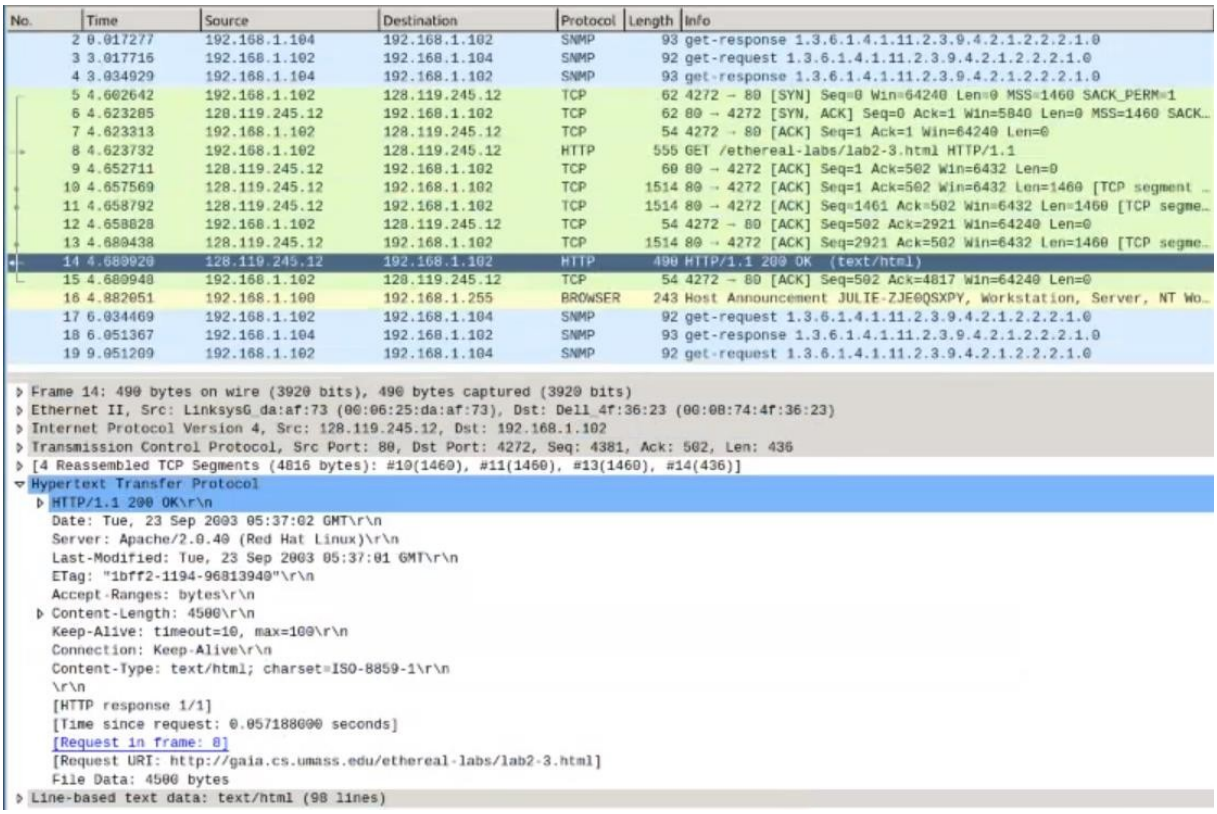
    Packet nr. 14 contains the status code. The code was 200 OK.

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

    4 packets, as seen in the [4 reassembled TCP segments…].

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.

    The segmentation is the transport layer's responsibility which means it has nothing to to with HTTP or it's headers. So no there is no such data in the HTTP header information.



Discussion: Big packages cannot be sent in one piece, instead they must be split up into different segments. This process takes place in the transport layer, meaning the application layer does not need to worry about this.

16. How many HTTP GET request messages were sent by your browser? To which Internet addresses were these GET requests sent?
    3 were sent to ethereal-labs/lab2-4.html, catalog/images/pearson-logo-footer.gif, ~kurose/cover.jpg.

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

They were not done in parallel. This can be seen in TCP sequence numbers. The first image ends with segment 25, and the second image start with segment 29.

No.	Time	Source	Destination	Protocol	Length	Info
10	7.236929	192.168.1.102	128.119.245.12	HTTP	555	GET /ethereal-labs/lab2-4.html HTTP/1.1
12	7.260813	128.119.245.12	192.168.1.102	HTTP	1057	HTTP/1.1 200 OK (text/html)
17	7.305485	192.168.1.102	165.193.123.218	HTTP	625	GET /catalog/images/pearson-logo-footer.gif HTTP/1.1
20	7.308803	192.168.1.102	134.241.6.82	HTTP	609	GET /~kurose/cover.jpg HTTP/1.1
25	7.333054	165.193.123.218	192.168.1.102	HTTP	912	HTTP/1.1 200 OK (GIF89a)
54	7.589877	134.241.6.82	192.168.1.102	HTTP	1096	HTTP/1.0 200 Document follows (JPEG JFIF image)

Discussion: The browser can download images in parallel to speed up the overall experience of the website. When downloading images the packet counts quickly balloon compared to just text files.

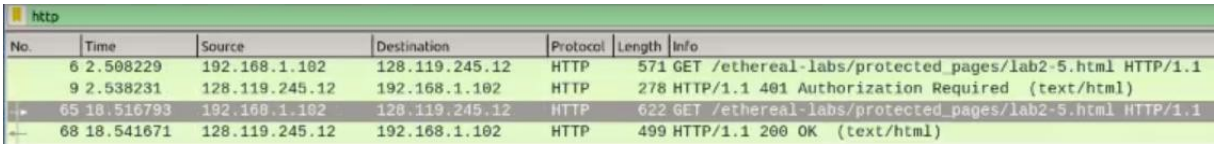
18. What is the server's response (status code and phrase) in response to the initial HTTP GET message from your browser?

401 Authorization required.

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

The authorization field.

Discussion: Websites may require authorization for some actions, to comply with this the browser adds the authorization field to confirm the identity of the user to the server.



No.	Time	Source	Destination	Protocol	Length	Info
6	2.508229	192.168.1.102	128.119.245.12	HTTP	571	GET /ethereal-labs/protected_pages/lab2-5.html HTTP/1.1
9	2.538231	128.119.245.12	192.168.1.102	HTTP	278	HTTP/1.1 401 Authorization Required (text/html)
65	18.516793	192.168.1.102	128.119.245.12	HTTP	622	GET /ethereal-labs/protected_pages/lab2-5.html HTTP/1.1
68	18.541671	128.119.245.12	192.168.1.102	HTTP	499	HTTP/1.1 200 OK (text/html)

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

It’s to determine if the connection is persistent or not. E.g., if subsequent requests to the same server are to be ignored or complied with. (Without doing another Syn/ack process). Keeping connections alive reduces CPU usage and memory usage for the server so it’s preferable in most cases. All modern browsers use this if the server cooperates.