

Prelab 2 - HTTP and Network Performance

(100 points)

Having gotten our feet wet with the Wireshark packet sniffer in lab1, we're now ready to use Wireshark packet sniffer to investigate protocols in operation. In this lab, we'll explore several aspects of the HTTP protocol: the basic GET/response interaction, HTTP message formats, retrieving large HTML files, retrieving HTML files with embedded objects, and HTTP authentication and security. Before beginning these labs, make sure to read Section 2.2 of the text.^[1]

Screenshots: For all the questions that require a screenshot, make sure that a **date timestamp** is visible next to your results (you can have a portion of another terminal open with the date command).

References: Chapter 2, Computer Networking: A Top Down Approach 1.

[24 pts] The Basic HTTP request/response interaction

Let's begin our exploration of HTTP by downloading a very simple HTML file - one that is very short, and contains no embedded objects. Do the following: • Start up your web browser in the Mininet VM.

- Start up Wireshark. In the display-filter enter **http**, so that only captured HTTP messages will be displayed later in the packet-listing windows (We're only interested in the HTTP protocol here, and don't want to see the clutter of all captured packets). Then, start a packet capture using the interface 'eth0'. • Clear the cache in your browser^[2] and enter the following:

<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.htm>

| Your browser should display a very simple, one-line HTML file.

- Stop Wireshark packet capture.

Exercises:

Remember that the HTTP headers are extra information in the HTTP message sent by either the client or the server. You can search the vast number of HTTP headers online. By looking at the information in the HTTP GET and response message(s) in Wireshark, answer the following questions:

- a. [3 pts] Is your browser running HTTP version 1.0, 1.1, or 2? What version of HTTP is the server running? How do you know?

The browser is running HTTP version 1.1 and the server is running HTTP version 1.1. In Wireshark, it shows this information under the Info column.

- b. [3 pts] How does your browser indicate to a server the languages it can accept? Based on what you see in Wireshark, what languages does your browser prefer?

The browser indicates to the server what languages it can accept under the "Accept-Language" HTTP request header.

Based on what I see in Wireshark, the browser prefers "en-US."

- c. [3 pts] In this exchange, who is the client and what is its IP address?

The client is the Mininet VM and its IP address is 10.0.2.15

- d. [3 pts] In this exchange, who is the server and what is its IP address?

The server is

<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html> and its IP address is 128.119.245.12

- e. [3 pts] What is the status code returned from the server to your browser?

The status code is 200.

- f. [3 pts] How many bytes of content are being returned to your browser?

540 bytes are being returned to the browser.

- g. [6 pts] Now do a new Wireshark packet capture for "<https://wireshark.org>" **without http filter**- what do you observe is different compared to the previous URL? Can you read the responses from the server? (Hint: Do some investigation into the purpose of HTTPS.) Take note of the different protocols used and mention them in your observations discussion. Take a screenshot and mark it up to help explain your observations.

I am seeing mostly TCP requests and responses. TCP is used to organize data

and ensure a secure transmission between the server and the client. I am also seeing the TLS protocol, which adds a layer of security on top of the TCP protocol. TLS uses encryption to securely send private data, but it can increase latency between client and server communications. The DNS protocol is used to identify websites using readable host names instead of numerical IP addresses.

mininet@cse150:~\$ date
Fri Oct 14 11:22:58 PM PDT 2022
mininet@cse150:~\$

*enp0s3

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol
73	1.505114476	10.0.2.15	172.67.75.39	TLSv1.3
74	1.505335474	172.67.75.39	10.0.2.15	TCP
75	1.822005483	172.67.75.39	10.0.2.15	TLSv1.3
76	1.822080731	10.0.2.15	172.67.75.39	TCP
77	1.823043707	10.0.2.15	172.67.75.39	TLSv1.3
78	1.823343241	172.67.75.39	10.0.2.15	TCP
79	1.825452673	10.0.2.15	128.114.129.33	DNS
80	1.838600009	128.114.129.33	10.0.2.15	DNS
81	1.839349610	10.0.2.15	172.67.75.39	TLSv1.3
82	1.839718804	172.67.75.39	10.0.2.15	TCP
83	1.871628753	172.67.75.39	10.0.2.15	TLSv1.3
84	1.872045977	10.0.2.15	172.67.75.39	TCP
85	1.872452342	172.67.75.39	10.0.2.15	TLSv1.3

User Datagram Protocol, Src Port: 53, Dst Port: 59482

Domain Name System (response)

Transaction ID: 0x853a

Flags: 0x8180 Standard query response, No error

Questions: 1

Answer RRs: 3

Authority RRs: 0

Additional RRs: 1

Queries

Answers

- www.wireshark.org: type A, class IN, addr 172.67.75.39
- www.wireshark.org: type A, class IN, addr 104.20.11.240
- www.wireshark.org: type A, class IN, addr 104.26.10.240

Domain name (DNS)

2. [40 pts] Experiment with wget

A. [20 pts] Running the *wget* command

In the VM terminal, run *wget*

<http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2/http.html> to display the HTTP headers returned by the server **without downloading the actual file**. Also use Wireshark to capture the packets exchanged by *wget*. ***Please ignore the /favicon.ico request/response in Wireshark.***

1. [3 pts] Take a screenshot showing your command and the output

```
mininet@cse150:~$ date
Fri Oct 14 11:26:57 PM PDT 2022
mininet@cse150:~$ wget -S http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2/http.html
--2022-10-14 23:27:00-- http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2/http.html
Resolving www2.cs.uh.edu (www2.cs.uh.edu)... 104.198.203.13
Connecting to www2.cs.uh.edu (www2.cs.uh.edu)|104.198.203.13|:80... connected.
HTTP request sent, awaiting response...
  HTTP/1.1 200 OK
  Date: Sat, 15 Oct 2022 06:27:01 GMT
  Server: Apache/2.4.29 (Ubuntu)
  Last-Modified: Mon, 18 Jun 2018 18:50:37 GMT
  ETag: "b0ed-56eef0a8ff940"
  Accept-Ranges: bytes
  Content-Length: 45293
  Vary: Accept-Encoding
  Keep-Alive: timeout=5, max=100
  Connection: Keep-Alive
  Content-Type: text/html
Length: 45293 (44K) [text/html]
Saving to: 'http.html.2'

http.html.2      100%[=====>]  44.23K  --.-KB/s    in 0.1s

2022-10-14 23:27:00 (391 KB/s) - 'http.html.2' saved [45293/45293]
```

Answer the following questions based on the output of *wget* (mark up your screenshot of the output based on questions below):

2. [2 pts] When was the web page last modified ? What does the last modified field indicate?

```

mininet@cse150:~$ date
Fri Oct 14 11:26:57 PM PDT 2022
mininet@cse150:~$ wget -S http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2/http.html
--2022-10-14 23:27:00-- http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2/http.html
Resolving www2.cs.uh.edu (www2.cs.uh.edu)... 104.198.203.13
Connecting to www2.cs.uh.edu (www2.cs.uh.edu)|104.198.203.13|:80... connected.
HTTP request sent, awaiting response...
HTTP/1.1 200 OK
Date: Sat, 15 Oct 2022 06:27:01 GMT
Server: Apache/2.4.29 (Ubuntu)
Last-Modified: Mon, 18 Jun 2018 18:50:37 GMT
ETag: "b0ed30ecf080ff040"
Accept-Ranges: bytes
Content-Length: 45293
Vary: Accept-Encoding
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
Length: 45293 (44K) [text/html]
Saving to: 'http.html.2'

http.html.2      100%[=====>]  44.23K  --.-KB/s   in 0.1s

2022-10-14 23:27:00 (391 KB/s) - 'http.html.2' saved [45293/45293]

```

The web page was last modified on Monday, June 18th 2018.

The last modified field indicates when the web-page was last updated.

3. [2 pts] What type of HTTP *connection* is used ? Describe what this connection does.

A persistent HTTP connection is being used (aka “Keep-alive” connection). This connection allows multiple objects to be sent over a single TCP connection between the client and server.

4. [2 pts] On what port number is the server listening for this request? How do you determine this?

It is using port 80. The port is located on the “connecting” line and is next to the server IP. (Screenshot below)

```

mininet@cse150:~$ date
Fri Oct 14 11:26:57 PM PDT 2022
mininet@cse150:~$ wget -S http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2/http.html
--2022-10-14 23:27:00-- http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2/http.html
Resolving www2.cs.uh.edu (www2.cs.uh.edu)... 104.198.203.13
Connecting to www2.cs.uh.edu (www2.cs.uh.edu)|104.198.203.13| 80... connected.
HTTP request sent, awaiting response...
  HTTP/1.1 200 OK
  Date: Sat, 15 Oct 2022 06:27:01 GMT
  Server: Apache/2.4.29 (Ubuntu)
  Last-Modified: Mon, 18 Jun 2018 18:50:37 GMT
  ETag: "b0ed-56eef0a8ff940"
  Accept-Ranges: bytes
  Content-Length: 45293
  Vary: Accept-Encoding
  Keep-Alive: timeout=5, max=100
  Connection: Keep-Alive
  Content-Type: text/html
Length: 45293 (44K) [text/html]
Saving to: 'http.html.2'

http.html.2      100%[=====] 44.23K  --.-KB/s   in 0.1s

2022-10-14 23:27:00 (391 KB/s) - 'http.html.2' saved [45293/45293]

```

5. [3 pts] What is the size (in bytes) of the base HTML page?

The size of the base HTML page is 45,293 bytes.

```

mininet@cse150:~$ date
Fri Oct 14 11:26:57 PM PDT 2022
mininet@cse150:~$ wget -S http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2/http.html
--2022-10-14 23:27:00-- http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2/http.html
Resolving www2.cs.uh.edu (www2.cs.uh.edu)... 104.198.203.13
Connecting to www2.cs.uh.edu (www2.cs.uh.edu)|104.198.203.13|:80... connected.
HTTP request sent, awaiting response...
  HTTP/1.1 200 OK
  Date: Sat, 15 Oct 2022 06:27:01 GMT
  Server: Apache/2.4.29 (Ubuntu)
  Last-Modified: Mon, 18 Jun 2018 18:50:37 GMT
  ETag: "b0ed-56eef0a8ff940"
  Accept-Ranges: bytes
  Content-Length: 45293
  Vary: Accept-Encoding
  Keep-Alive: timeout=5, max=100
  Connection: Keep-Alive
  Content-Type: text/html
Length: 45293 (44K) [text/html]
Saving to: 'http.html.2'

http.html.2      100%[=====] 44.23K  --.-KB/s   in 0.1s

2022-10-14 23:27:00 (391 KB/s) - 'http.html.2' saved [45293/45293]

```

Now look at your Wireshark capture and take a screenshot:

mininet@cse150: ~

```
mininet@cse150:~$ date
Fri Oct 14 11:44:02 PM PDT 2022
```

*enp0s3

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol
1	0.000000000	10.0.2.15	239.255.255.250	SSDP
2	0.320147228	10.0.2.15	128.114.129.33	DNS
3	0.325103080	128.114.129.33	10.0.2.15	DNS
4	0.325666382	10.0.2.15	104.198.203.13	TCP
5	0.383010180	104.198.203.13	10.0.2.15	TCP
6	0.383123842	10.0.2.15	104.198.203.13	TCP
7	0.383410192	10.0.2.15	104.198.203.13	HTTP
8	0.383756185	104.198.203.13	10.0.2.15	TCP
9	0.440157878	104.198.203.13	10.0.2.15	TCP
10	0.440187373	10.0.2.15	104.198.203.13	TCP
11	0.440373236	104.198.203.13	10.0.2.15	TCP
12	0.440398602	10.0.2.15	104.198.203.13	TCP
13	0.441662712	104.198.203.13	10.0.2.15	TCP

Frame 7: 226 bytes on wire (1808 bits), 226 bytes captured (1808 bits) on interface

Ethernet II, Src: PcsCompu_a8:d6:ca (08:00:27:a8:d6:ca), Dst: RealtekU_12:35

Internet Protocol Version 4, Src: 10.0.2.15, Dst: 104.198.203.13

Transmission Control Protocol, Src Port: 32794, Dst Port: 80, Seq: 1, Ack: 1

Hypertext Transfer Protocol

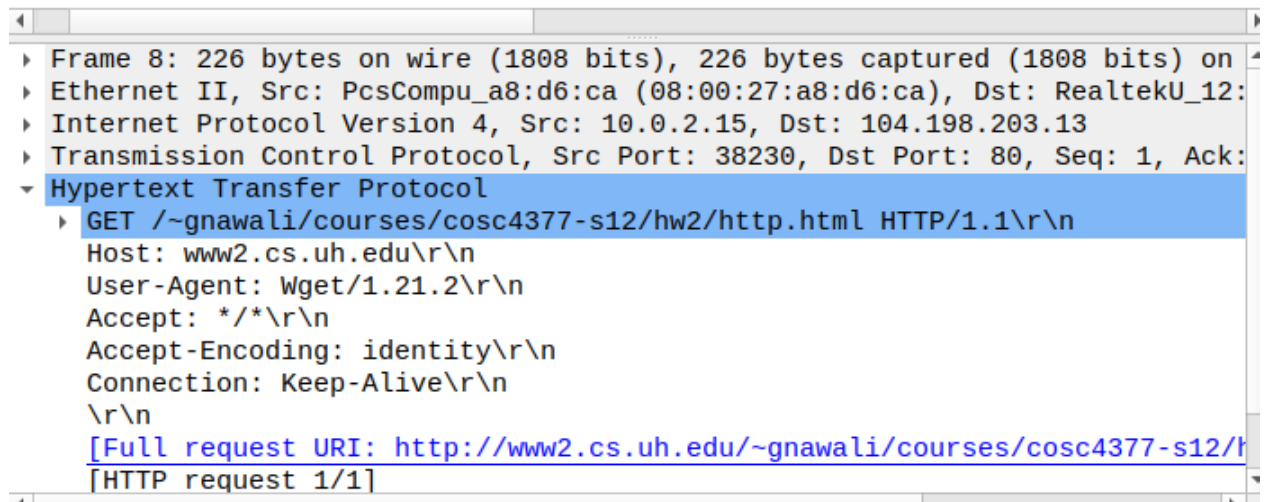
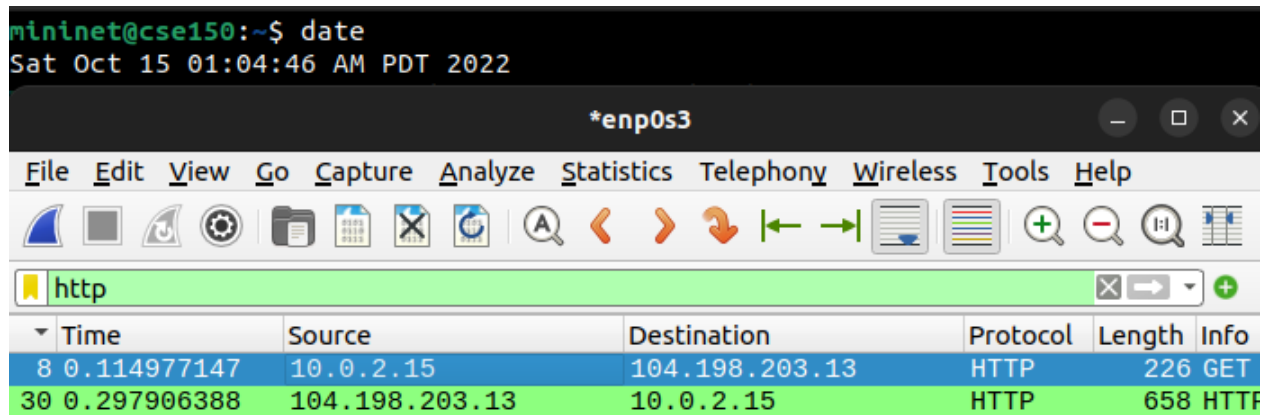
- GET /~gnawali/courses/cosc4377-s12/hw2/http.html HTTP/1.1\r\n
- Host: www2.cs.uh.edu\r\n
- User-Agent: Wget/1.21.2\r\n
- Accept: */*\r\n
- Accept-Encoding: identity\r\n
- Connection: Keep-Alive\r\n
- \r\n
- [Full request URI: http://www2.cs.uh.edu/~gnawali/courses/cosc4377-s12/hw2
- [HTTP request 1/1]

6. [2 pts] What HTTP method is used in the request message? Describe the purpose of this method.

The GET method is being used in this request message. It is used for posting/sending data to the server using a given URL.

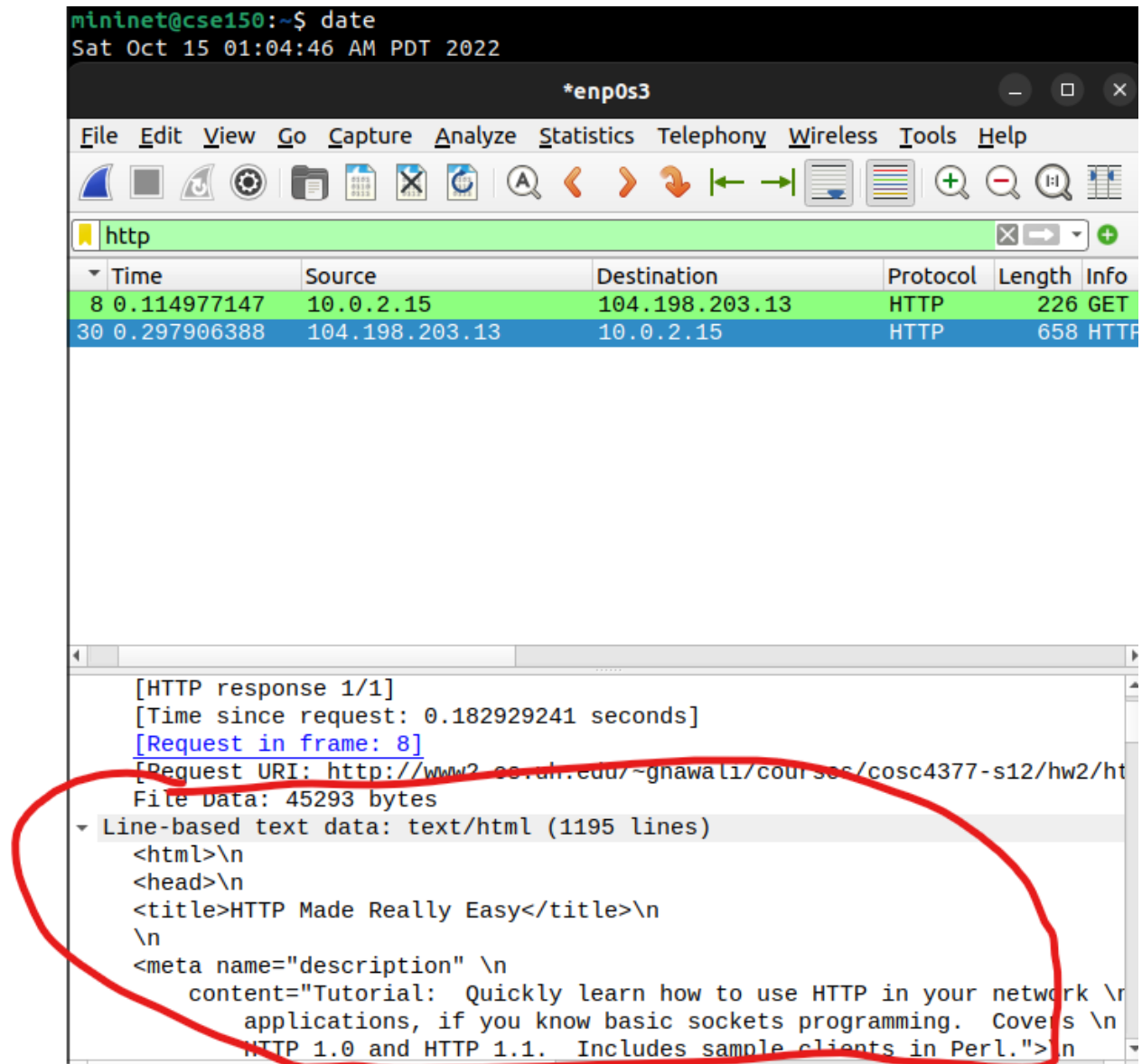
7. [3 pts] How many HTTP request/response messages do you see in total?
Take a screenshot showing the HTTP request/response messages in Wireshark.

I see 2 HTTP request/response messages in Wireshark.



8. [3 pts] Is the base HTML file downloaded? How do you know based on Wireshark? Attach a screenshot showing the same.

Yes, the base HTML file is downloaded, as it shows the total number bytes of the file, and Wireshark proceeds to display the HTML code.



B. [20 pts] Embedded Objects

Let's look at what happens when your browser downloads a file with several embedded objects, i.e., a file that includes other objects that are stored on another server(s).

Do the following:

- Enter <http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file4.html> in your browser and capture the packets in Wireshark.
- Your browser should display a short HTML file with two images. These two images are referenced in the base HTML file. That is, the images themselves are not contained in the HTML; instead the URLs for the images are contained in the downloaded HTML file. As discussed in the textbook, your browser will have to retrieve these objects from the indicated web sites. Stop Wireshark packet capture, and enter “**http**” in the display-filter-specification window, so that only captured HTTP messages will be displayed.
- ***Please ignore the /favicon.ico request/response in the wireshark.***

Answer the following questions:

1. [3 pts] How many HTTP GET request messages did your browser send in total? To which Internet addresses were these GET requests sent? Take a screenshot showing the HTTP request/response messages in Wireshark.

SCREENSHOTS BELOW

There are a total of 3 HTTP GET request messages

The first screenshot shows the GET address is sent to 128.119.245.12

```
mininet@cse150:~$ date
Sat Oct 15 01:25:22 AM PDT 2022
```

The first screenshot shows a Wireshark packet capture on interface *enp0s3. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Length
7	0.073773838	10.0.2.15	128.119.245.12	HTTP	5
13	0.165050313	128.119.245.12	10.0.2.15	HTTP	1
15	0.204344928	10.0.2.15	128.119.245.12	HTTP	4
22	0.284795049	128.119.245.12	10.0.2.15	HTTP	11
26	0.360014200	10.0.2.15	178.79.137.164	HTTP	4
28	0.604689867	178.79.137.164	10.0.2.15	HTTP	2
131	2.472729892	10.0.2.15	128.119.245.12	HTTP	4
133	2.551010613	128.119.245.12	10.0.2.15	HTTP	5

The packet details pane for packet 15 shows the following structure:

- Frame 15: 516 bytes on wire (4128 bits), 516 bytes captured (4128 bits) on Ethernet II, Src: PcsCompu_a8:d6:ca (08:00:27:a8:d6:ca), Dst: RealtekU_12:...
- Internet Protocol Version 4, Src: 10.0.2.15, Dst: 128.119.245.12
- Transmission Control Protocol, Src Port: 47032, Dst Port: 80, Seq: 1, Ack:...
- Hypertext Transfer Protocol
 - GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1.1\r\n
 - Host: gaia.cs.umass.edu\r\n

```
mininet@cse150:~$ date
Sat Oct 15 01:25:22 AM PDT 2022
```

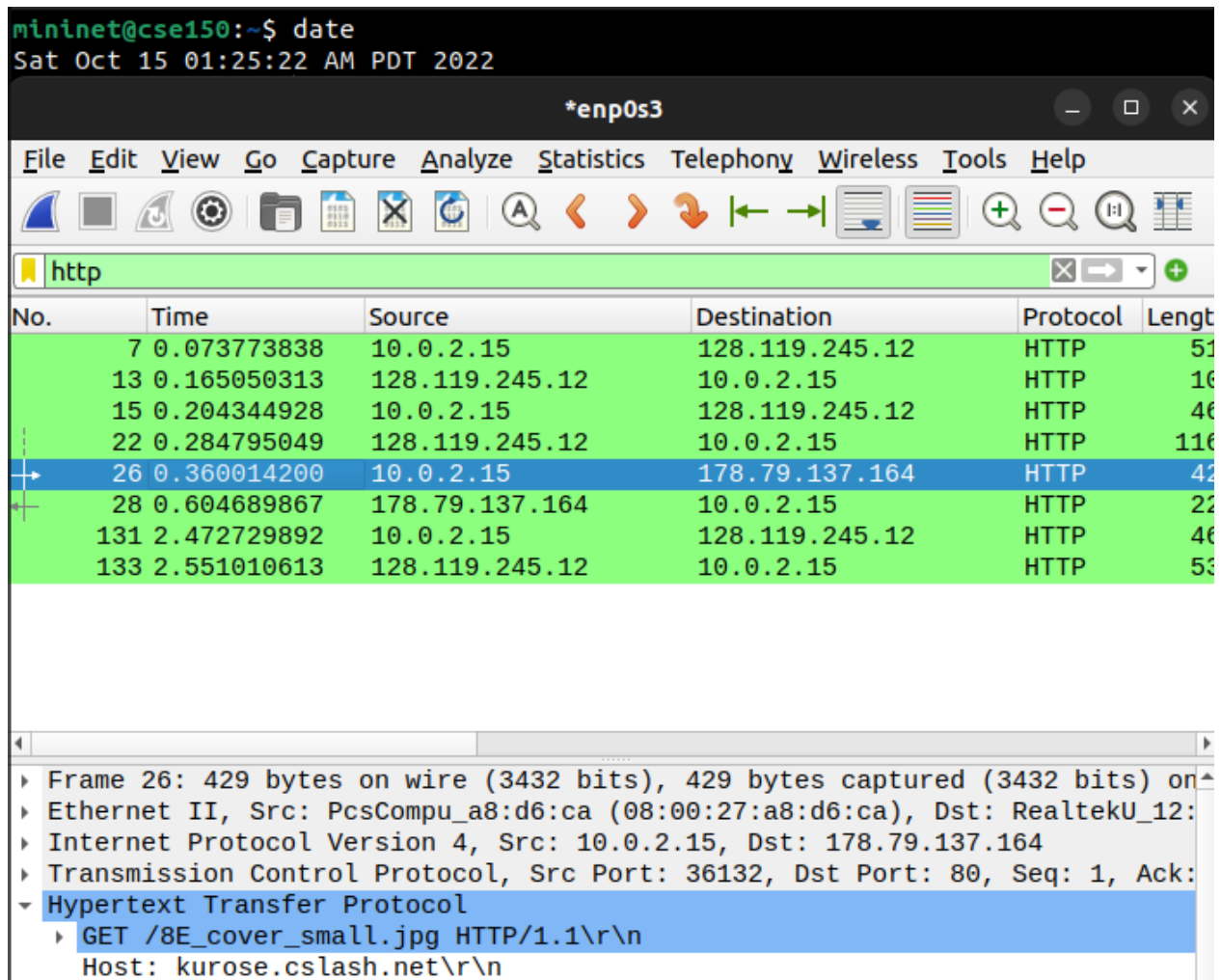
The second screenshot shows the same Wireshark packet capture. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Length
7	0.073773838	10.0.2.15	128.119.245.12	HTTP	5
13	0.165050313	128.119.245.12	10.0.2.15	HTTP	1
15	0.204344928	10.0.2.15	128.119.245.12	HTTP	4
22	0.284795049	128.119.245.12	10.0.2.15	HTTP	11
26	0.360014200	10.0.2.15	178.79.137.164	HTTP	4
28	0.604689867	178.79.137.164	10.0.2.15	HTTP	2
131	2.472729892	10.0.2.15	128.119.245.12	HTTP	4
133	2.551010613	128.119.245.12	10.0.2.15	HTTP	5

The packet details pane for packet 15 shows the following structure:

- Frame 15: 462 bytes on wire (3696 bits), 462 bytes captured (3696 bits) on Ethernet II, Src: PcsCompu_a8:d6:ca (08:00:27:a8:d6:ca), Dst: RealtekU_12:...
- Internet Protocol Version 4, Src: 10.0.2.15, Dst: 128.119.245.12
- Transmission Control Protocol, Src Port: 47032, Dst Port: 80, Seq: 463, Ac...
- Hypertext Transfer Protocol
 - GET /pearson.png HTTP/1.1\r\n
 - Host: gaia.cs.umass.edu\r\n

The second screenshot shows the GET request is sent to 128.119.245.12



The third screenshot shows the GET request being sent to 178.79.137.164

2. [3 pts] Describe the purpose of each of the Request/Response messages

The purpose of the 1st request/response message is to download the content of the base HTML file.

The purpose of the 2nd request/response message is to download the content for the Pearson logo image.

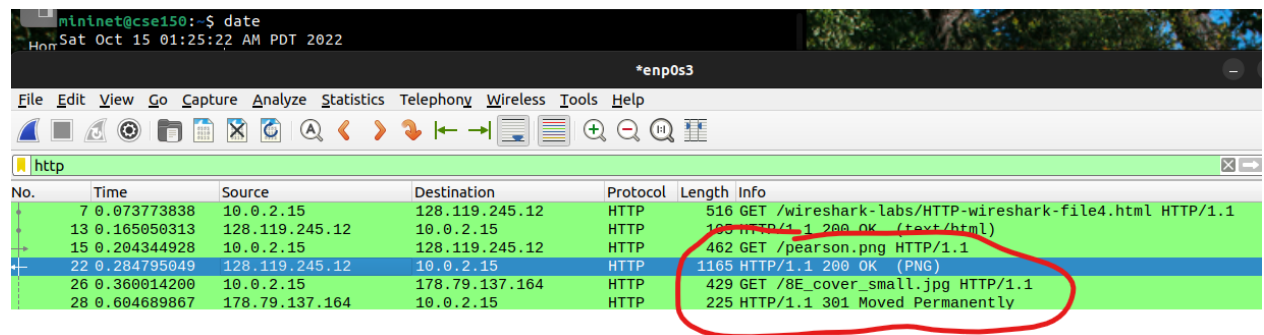
The purpose of the 3rd request/response message is to download the content for the textbook cover image.

3.[5 pts] Can you tell whether your browser downloaded the embedded images serially, or whether they were downloaded from the two websites in parallel?

Explain and support with screenshots from your Wireshark trace.

The browser downloaded the embedded images serially as the first image

is downloaded after the second GET request, and the second image is downloaded after the 3rd GET request.



mininet@cse150:~\$ date
Sat Oct 15 01:25:22 AM PDT 2022

*enp0s3

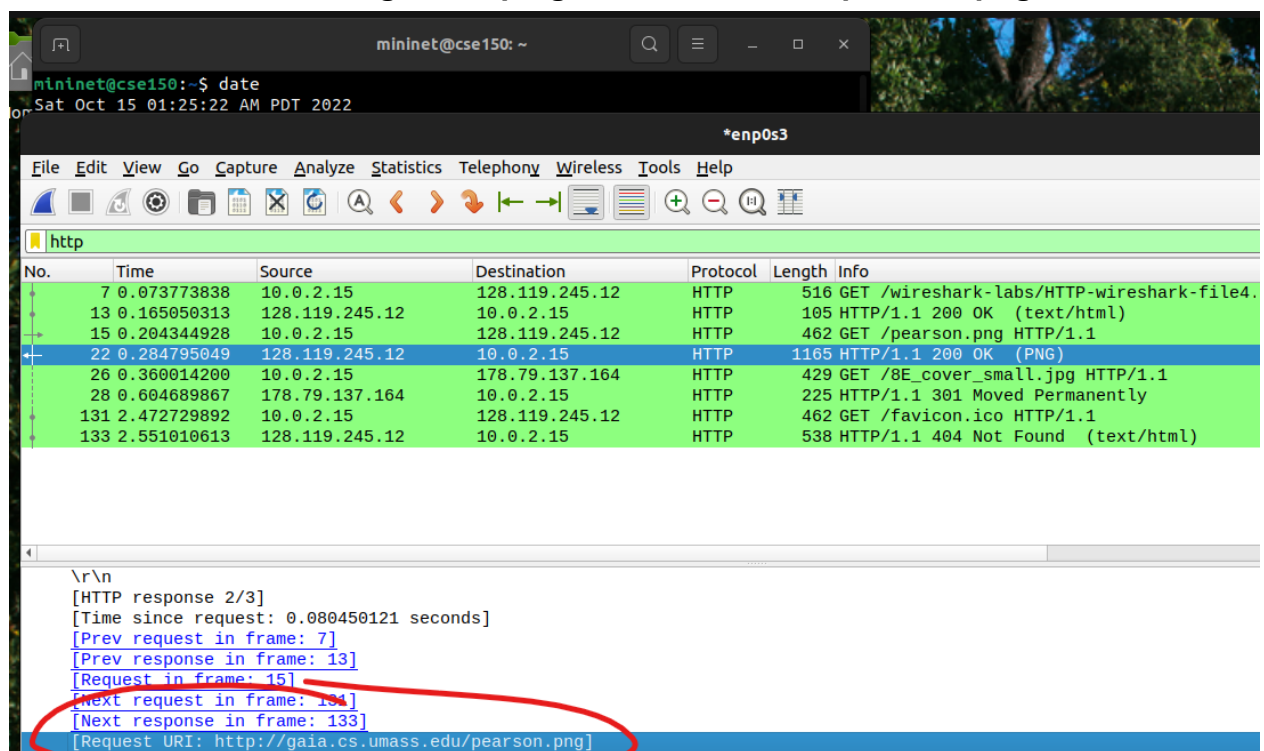
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
7	0.073773838	10.0.2.15	128.119.245.12	HTTP	516	GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1.1
13	0.165050313	128.119.245.12	10.0.2.15	HTTP	105	HTTP/1.1 200 OK (text/html)
15	0.204344928	10.0.2.15	128.119.245.12	HTTP	462	GET /pearson.png HTTP/1.1
22	0.284795049	128.119.245.12	10.0.2.15	HTTP	1165	HTTP/1.1 200 OK (PNG)
26	0.360014200	10.0.2.15	178.79.137.164	HTTP	429	GET /8E_cover_small.jpg HTTP/1.1
28	0.604689867	178.79.137.164	10.0.2.15	HTTP	225	HTTP/1.1 301 Moved Permanently

4. [3 pts] What are the URLs for the images? Support with screenshots from your Wireshark trace.

The URL for the first image is <http://gaia.cs.umass.edu/pearson.png>



mininet@cse150:~\$ date
Sat Oct 15 01:25:22 AM PDT 2022

*enp0s3

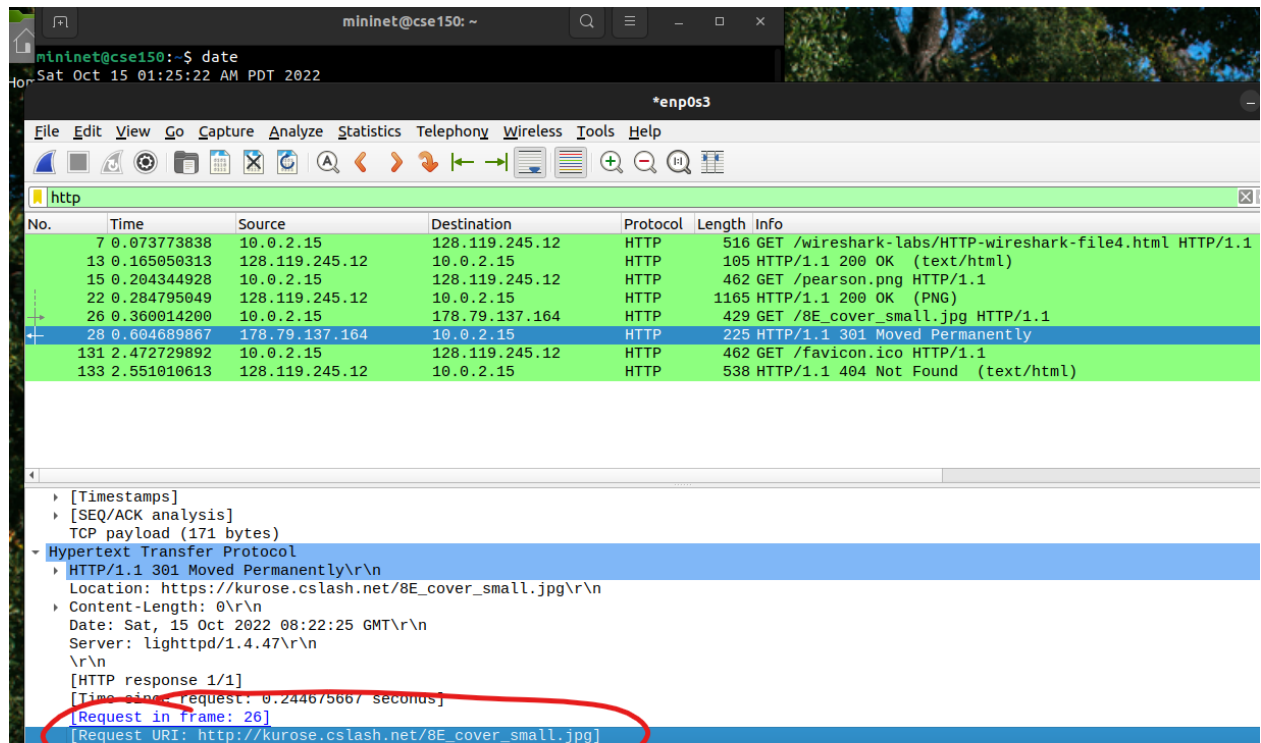
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
7	0.073773838	10.0.2.15	128.119.245.12	HTTP	516	GET /wireshark-labs/HTTP-wireshark-file4.
13	0.165050313	128.119.245.12	10.0.2.15	HTTP	105	HTTP/1.1 200 OK (text/html)
15	0.204344928	10.0.2.15	128.119.245.12	HTTP	462	GET /pearson.png HTTP/1.1
22	0.284795049	128.119.245.12	10.0.2.15	HTTP	1165	HTTP/1.1 200 OK (PNG)
26	0.360014200	10.0.2.15	178.79.137.164	HTTP	429	GET /8E_cover_small.jpg HTTP/1.1
28	0.604689867	178.79.137.164	10.0.2.15	HTTP	225	HTTP/1.1 301 Moved Permanently
131	2.472729892	10.0.2.15	128.119.245.12	HTTP	462	GET /favicon.ico HTTP/1.1
133	2.551010613	128.119.245.12	10.0.2.15	HTTP	538	HTTP/1.1 404 Not Found (text/html)

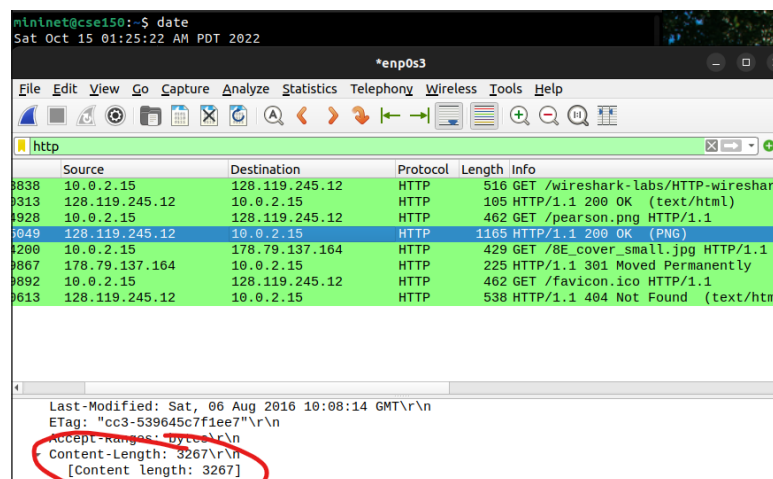
\r\n
[HTTP response 2/3]
[Time since request: 0.080450121 seconds]
[Prev request in frame: 7]
[Prev response in frame: 13]
[Request in frame: 15]
[Next request in frame: 134]
[Next response in frame: 133]
[Request URI: http://gaia.cs.umass.edu/pearson.png]

The URL for the second image is http://kurose.cslash.net/8E_cover_small.jpg



- [3 pts] Draw a picture with a client and server, and use arrows to indicate the Requests and Responses to fully load the web page.
- [3 pts] What are the sizes of the embedded images? Attach the screenshot with markup on the size of the image.

First image size is 3,267 bytes



Wireshark says the second image size is 0 bytes, but that is because the image is at a different location.

mininet@cse150:~\$ date
Sat Oct 15 01:25:22 AM PDT 2022

*enp0s3

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

	Source	Destination	Protocol	Length	Info
8838	10.0.2.15	128.119.245.12	HTTP	516	GET /wireshark-labs/HTTP-wireshark
9313	128.119.245.12	10.0.2.15	HTTP	105	HTTP/1.1 200 OK (text/html)
9928	10.0.2.15	128.119.245.12	HTTP	462	GET /pearson.png HTTP/1.1
10049	128.119.245.12	10.0.2.15	HTTP	1165	HTTP/1.1 200 OK (PNG)
10200	10.0.2.15	178.79.137.164	HTTP	429	GET /8E_cover_small.jpg HTTP/1.1
10867	178.79.137.164	10.0.2.15	HTTP	225	HTTP/1.1 301 Moved Permanently
10892	10.0.2.15	128.119.245.12	HTTP	462	GET /favicon.ico HTTP/1.1
109613	128.119.245.12	10.0.2.15	HTTP	538	HTTP/1.1 404 Not Found (text/html)

Response Version: HTTP/1.1
Status Code: 301
[Status Code Description: Moved Permanently]
Response Phrase: Moved Permanently
Location: https://kurose.cslash.net/8E_cover_small.jpg\r\n
Content-Length: 0\r\n
[Content length: 0]
Date: Sat, 15 Oct 2022 08:22:25 GMT\r\n
Server: lighttpd/1.4.47\r\n
\r\n
[HTTP response 1/1]
[Time since request: 0.244675667 seconds]
[Request in frame: 26]
[Request URI: http://kurose.cslash.net/8E_cover_small.jpg]

3. [16 pts] HTTP Conditional GET

Recall from Section 2.2.5 of the text, that most web browsers perform object caching and thus often perform a conditional GET when retrieving an HTTP object. Before performing the steps below, make sure your browser's cache is empty^[2]. Now do the following:

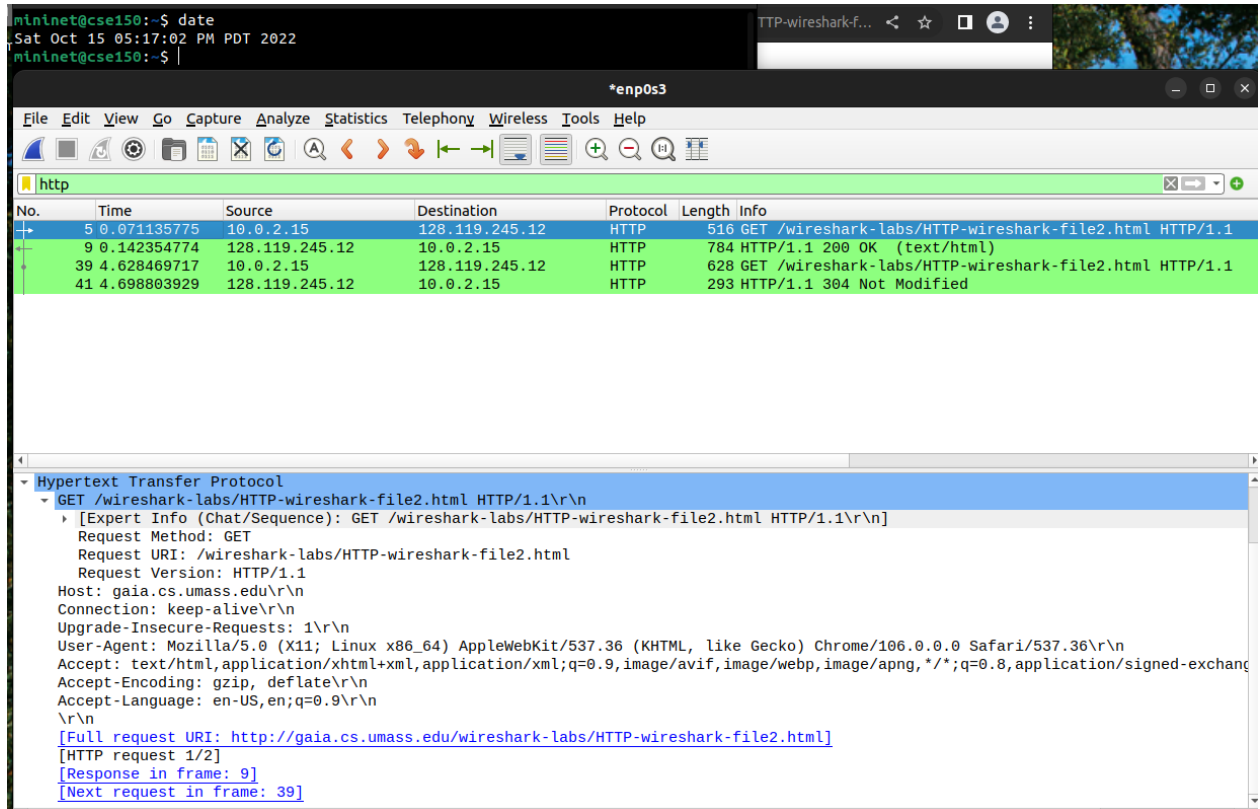
- Start up your web browser, and make sure your browser's cache is cleared.
- Start up the Wireshark packet sniffer
- Enter the following URL into your browser
<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.htm>
! Your browser should display a very simple five-line HTML file.
- Quickly enter the same URL into your browser again (or simply select the refresh button on your browser)
- Stop Wireshark packet capture, and enter "http" (again, in lower case without the quotation marks) in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.

- Please ignore the /favicon.ico request/response in Wireshark.

Answer the following questions:

- [4 pts] Inspect the contents of the first HTTP GET^[3] request from your browser to the server. Do you see an “IF-MODIFIED-SINCE” line in the HTTP GET? Show a screenshot. What is the purpose of this header?

There is no “IF-MODIFIED-SINCE” line in the HTTP GET request. The purpose of the header is to show the last time the browser received the requested data from the server.



- [4 pts] Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell? Support with screenshots from your Wireshark trace.

Yes the server did explicitly return the contents of the file as it shows the content of the message in the response. (SCREENSHOT BELOW)

mininet@cse150:~\$ date
Sat Oct 15 05:17:02 PM PDT 2022
mininet@cse150:~\$

Wireshark interface showing an HTTP trace on interface *enp0s3. The trace contains three packets:

No.	Time	Source	Destination	Protocol	Length	Info
5	0.071135775	10.0.2.15	128.119.245.12	HTTP	516	GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
9	0.142354774	128.119.245.12	10.0.2.15	HTTP	784	HTTP/1.1 200 OK (text/html)
39	4.628469717	10.0.2.15	128.119.245.12	HTTP	628	GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
41	4.698803929	128.119.245.12	10.0.2.15	HTTP	293	HTTP/1.1 304 Not Modified

The details pane for the selected packet (No. 39) shows the following information:

- [HTTP response 1/2]
- [Time since request: 0.071218999 seconds]
- [Request in frame: 5]
- [Next request in frame: 39]
- [Next response in frame: 41]
- [Request URI: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html]
- File Data: 371 bytes
- Line-based text data: text/html (10 lines)

```

\r\n
[HTTP response 1/2]
[Time since request: 0.071218999 seconds]
[Request in frame: 5]
[Next request in frame: 39]
[Next response in frame: 41]
[Request URI: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html]
File Data: 371 bytes
Line-based text data: text/html (10 lines)
\r\n
<html>\r\n
\r\n
Congratulations again! Now you've downloaded the file lab2-2.html. <br>\r\n
This file's last modification date will not change. <p>\r\n
Thus if you download this multiple times on your browser, a complete copy <br>\r\n
will only be sent once by the server due to the inclusion of the IN-MODIFIED-SINCE<br>\r\n
field in your browser's HTTP GET request to the server.\r\n
\r\n
</html>\r\n

```

- c. [4 pts] 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? Support with screenshots from your Wireshark trace.

There is an “IF-MODIFIED-SINCE” line in the second HTTP GET request. It shows the date: Saturday, October 15, 2022, which is the date in the first server response. (SCREENSHOTS BELOW)

mininet@cse150: ~
Sat Oct 15 05:17:02 PM PDT 2022
mininet@cse150: ~\$

shark-labs/HTTP-wireshark-f...
ab2-2.html.

*enp0s3

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
5	0.071135775	10.0.2.15	128.119.245.12	HTTP	516	GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
9	0.142354774	128.119.245.12	10.0.2.15	HTTP	784	HTTP/1.1 200 OK (text/html)
39	4.628469717	10.0.2.15	128.119.245.12	HTTP	628	GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
41	4.698803929	128.119.245.12	10.0.2.15	HTTP	293	HTTP/1.1 304 Not Modified

Transmission Control Protocol, Src Port: 57718, Dst Port: 80, Seq: 463, Ack: 731, Len: 574

Hypertext Transfer Protocol

GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1\r\n

[Expert Info (Chat/Sequence): GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1\r\n]

Request Method: GET

Request URI: /wireshark-labs/HTTP-wireshark-file2.html

Request Version: HTTP/1.1

Host: gaia.cs.umass.edu\r\n

Connection: keep-alive\r\n

Cache-Control: max-age=0\r\n

Upgrade-Insecure-Requests: 1\r\n

User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36\r\n

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange\r\n

Accept-Encoding: gzip, deflate\r\n

Accept-Language: en-US,en;q=0.9\r\n

If-None-Match: "172-5b0c706-0800"\r\n

If-Modified-Since: Sat, 15 Oct 2022 05:59:01 GMT\r\n

mininet@cse150: ~\$ date
Sat Oct 15 05:17:02 PM PDT 2022
mininet@cse150: ~\$

shark-labs/HTTP-wireshark-f...
ab2-2.html.

*enp0s3

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
5	0.071135775	10.0.2.15	128.119.245.12	HTTP	516	GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
9	0.142354774	128.119.245.12	10.0.2.15	HTTP	784	HTTP/1.1 200 OK (text/html)
39	4.628469717	10.0.2.15	128.119.245.12	HTTP	628	GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
41	4.698803929	128.119.245.12	10.0.2.15	HTTP	293	HTTP/1.1 304 Not Modified

Hypertext Transfer Protocol

HTTP/1.1 200 OK\r\n

[Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]

Response Version: HTTP/1.1

Status Code: 200

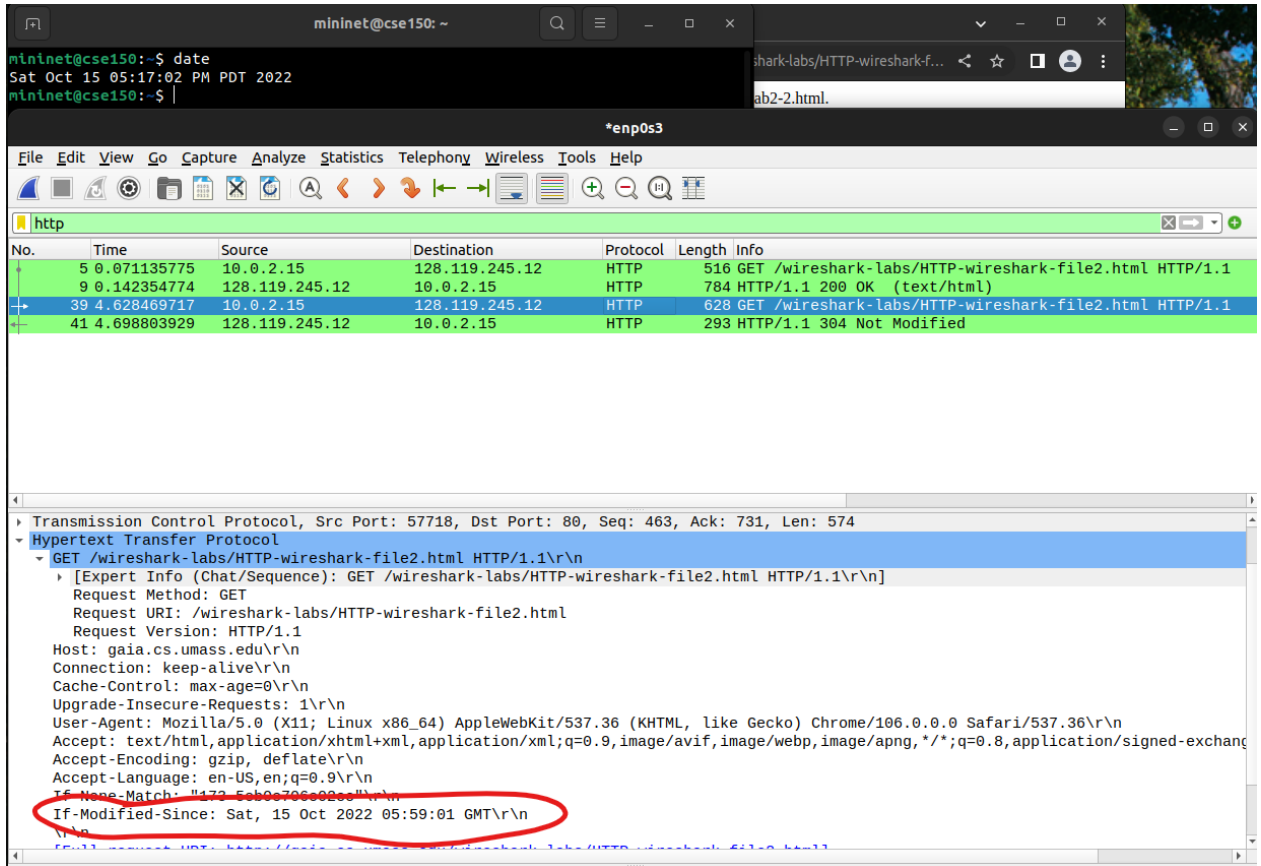
[Status Code Description: OK]

Response Phrase: OK

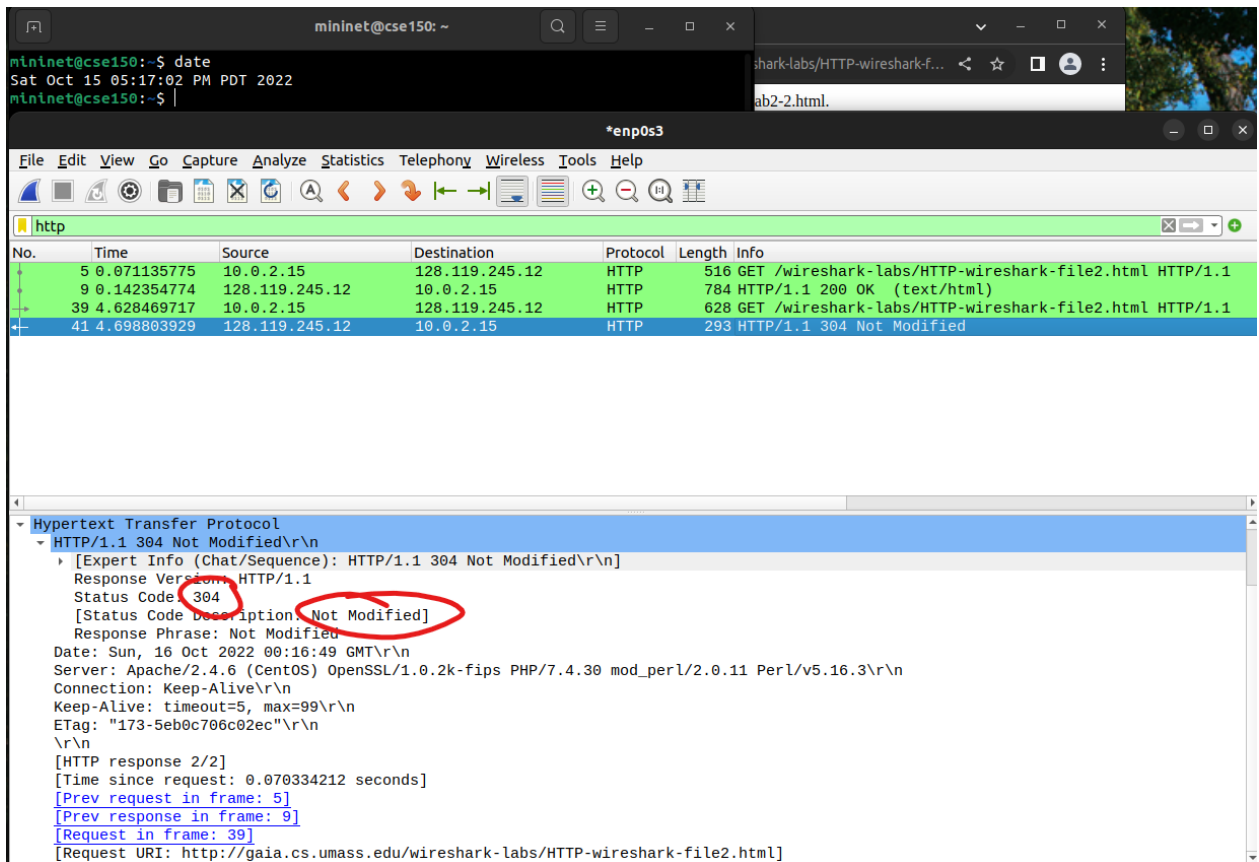
Date: Sun, 16 Oct 2022 00:16:45 GMT\r\n

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.30 mod_perl/2.0.11 Perl/v5.16.3\r\n

Last-Modified: Sat, 15 Oct 2022 05:59:01 GMT\r\n



- d. [4 pts] 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 and support with screenshots from your Wireshark trace.
- The status code and phrase of the server response to the second HTTP GET is 304: Not Modified. The server did not explicitly return the contents of the file because nothing was modified since the first HTTP GET response. (SCREENSHOT BELOW)**



4. [20 pts] Measuring Performance in Mininet

Using the *iperf* tool on your topology from lab1:

- a. [2 pts] Research the *iperf* tool and describe it in your own words.

The *iperf* tool is used to measure throughput of a network and tune it. It can be used to test TCP and UDP.

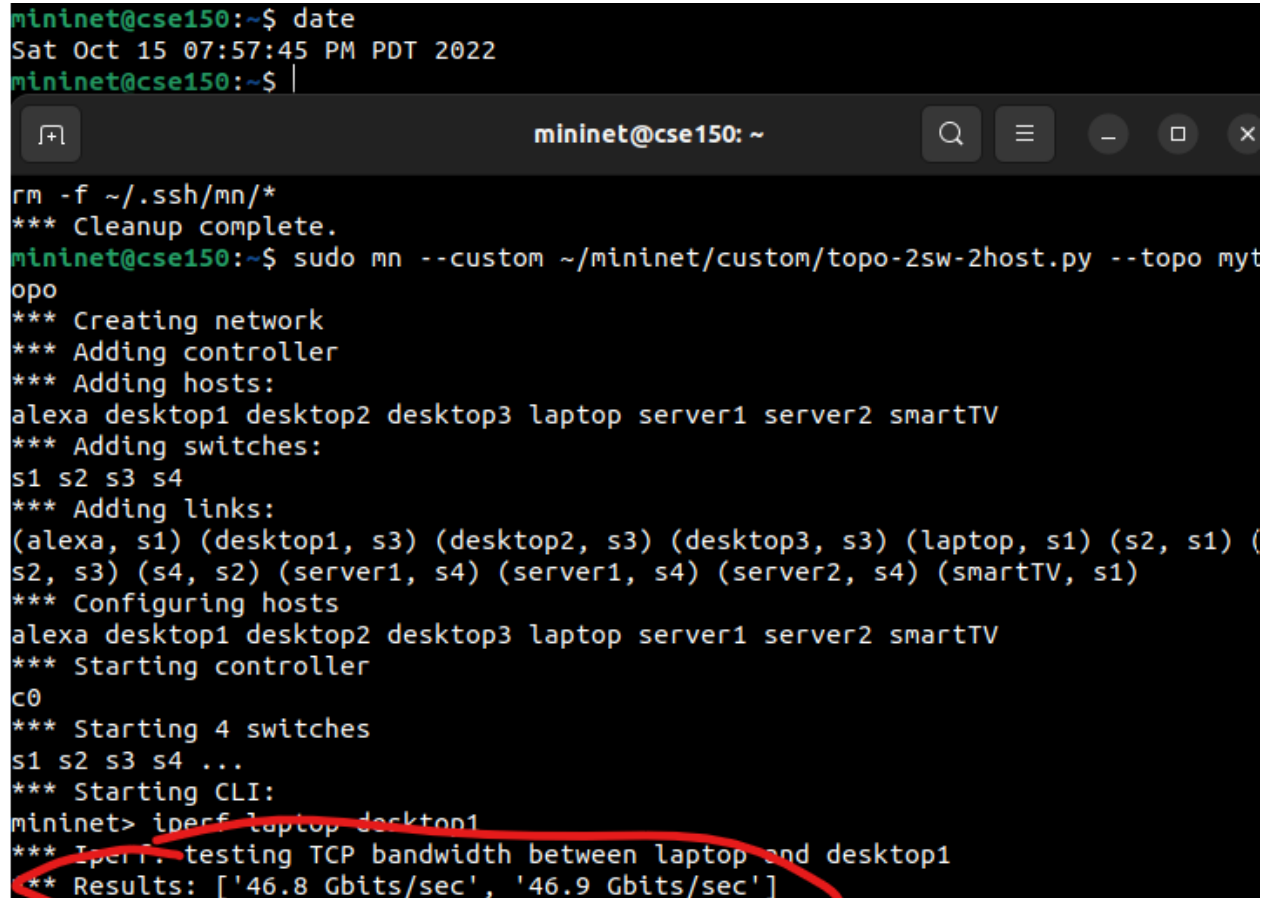
- b. [4 pts] Then run the *iperf* command between Laptop and Desktop1 in the Mininet command line and screenshot the output. Describe the results, referencing your screenshot, and determine how fast the connection is.

Laptop is streaming data to desktop1 at a speed of 46.8 Gigabits/sec and vice-versa is 46.9 Gigabits/sec.

```

mininet@cse150:~$ date
Sat Oct 15 07:57:45 PM PDT 2022
mininet@cse150:~$ |

```



```

rm -f ~/.ssh/mn/*
*** Cleanup complete.
mininet@cse150:~$ sudo mn --custom ~/mininet/custom/topo-2sw-2host.py --topo myt
opo
*** Creating network
*** Adding controller
*** Adding hosts:
alexas desktop1 desktop2 desktop3 laptop server1 server2 smartTV
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(alexas, s1) (desktop1, s3) (desktop2, s3) (desktop3, s3) (laptop, s1) (s2, s1) (
s2, s3) (s4, s2) (server1, s4) (server1, s4) (server2, s4) (smartTV, s1)
*** Configuring hosts
alexas desktop1 desktop2 desktop3 laptop server1 server2 smartTV
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet> iperf laptop desktop1
*** Iperf: testing TCP bandwidth between laptop and desktop1
** Results: ['46.8 Gbits/sec', '46.9 Gbits/sec']

```

- c. [3 pts] Given an example of an application or test case that would benefit from running the iperf tool.

An application that would benefit from running the iperf tool is testing the network speed between a central company server and the employee desktop stations of the company, to make sure the connection is healthy for data uploads and downloads.

Set the bandwidth between all the links between Laptop and Desktop1 (4 links total) to 50 Mbps by adding a `bw=` parameter to `addLink`, which takes its argument as a number representing the bandwidth in Mbps.

- d. [4 pts] Measure the throughput between Laptop and Desktop1 with *iperf* three times and take the average of all the Mb/s/sec figures printed. What is your result? Include a screenshot of the results.

43.51 Gbits/sec is the average number of all the results

```

mininet@cse150:~$ date
Sat Oct 15 08:29:47 PM PDT 2022

*** Adding hosts:
alexas desktop1 desktop2 desktop3 laptop server1 server2 smartTV
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(alexas, s1) (desktop1, s3) (desktop2, s3) (desktop3, s3) (laptop, s1) (s2, s1) (
s2, s3) (s4, s2) (server1, s4) (server1, s4) (server2, s4) (smartTV, s1)
*** Configuring hosts
alexas desktop1 desktop2 desktop3 laptop server1 server2 smartTV
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet> iperf laptop desktop1
*** Iperf: testing TCP bandwidth between laptop and desktop1
*** Results: ['46.7 Gbits/sec', '46.7 Gbits/sec']
mininet> iperf laptop desktop1
*** Iperf: testing TCP bandwidth between laptop and desktop1
*** Results: ['39.2 Gbits/sec', '39.3 Gbits/sec']
mininet> iperf laptop desktop1
*** Iperf: testing TCP bandwidth between laptop and desktop1
*** Results: ['44.6 Gbits/sec', '44.6 Gbits/sec']

```

e. [4 pts] Now remove all the bw= parameters you added except for the one between switch1 and switch2. Run iperf between the Laptop and Desktop1 another three times and take the average of all the Mbits/sec figures printed. How many Mbits/sec do you get now? Include a screenshot of the results.

46.09 Gbits/sec is my new average


```

mininet@cse150:~$ date
Sat Oct 15 08:35:19 PM PDT 2022

*** Adding hosts:
alexsa desktop1 desktop2 desktop3 laptop server1 server2 smartTV
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(alexsa, s1) (desktop1, s3) (desktop2, s3) (desktop3, s3) (laptop, s1) (s2, s1) (
s2, s3) (s4, s2) (server1, s4) (server1, s4) (server2, s4) (smartTV, s1)
*** Configuring hosts
alexsa desktop1 desktop2 desktop3 laptop server1 server2 smartTV
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet> iperf laptop desktop1
*** Iperf: testing TCP bandwidth between laptop and desktop1
*** Results: ['46.4 Gbits/sec', '46.4 Gbits/sec']
mininet> iperf laptop desktop1
*** Iperf: testing TCP bandwidth between laptop and desktop1
*** Results: ['46.8 Gbits/sec', '46.9 Gbits/sec']
mininet> iperf laptop desktop1
*** Iperf: testing TCP bandwidth between laptop and desktop1
*** Results: ['45.0 Gbits/sec', '45.0 Gbits/sec']
mininet>

```

f. [3 pts] Do your answers for d. and e. vary significantly? Discuss why or why not?
The answers do not vary significantly because the hosts are all based on the same machine.

Submission:

You will submit 1 file for this assignment directly on Canvas: a PDF with all of your solutions to the questions.

Naming convention: name the PDF file as [YourCruzID].pdf.

[1] References to figures and sections are for the 8th edition of our text, *Computer Networks, A Top-down Approach*, 8th ed., J.F. Kurose and K.W. Ross, Addison-Wesley/Pearson, 2020.

[2] See <https://www.howtogeek.com/304218/how-to-clear-your-history-in-any-browser/> for instructions on clearing your browser cache.

[3] *Hint:* ideally, you should see an If-Modified-Since header since you've just downloaded this page a few seconds ago. However, depending on the browser you're using, and the format of the server's earlier response to your initial GET, your browser may not include an If-Modified-Since even if the document has been downloaded and cached. The Chrome browser is pretty good at regularly using If-Modified-Since. But Safari and Firefox are much more finicky about when to use If-Modified-Since. Life isn't always as easy in practice as it is in theory!