Department of Electrical Engineering, UET Lahore

EE432: Computer Networks

Course Instructor: Dr. Naveed Nawaz	**Dated:**
	Semester: 7th
	Session: Fall 2025

LAB 2 Application Layer Protocol: HyperText Transfer Protocol (HTTP)

Name	Roll. No.	Report Marks (5)	Viva Marks (5)	Total Marks (10)
Hafiz Muhammad Bilal	2022-EE-014			

Checked on: _____

Signature: _____

Application Layer Protocol: HyperText Transfer Protocol (HTTP)

1.1. Objectives

At the end of this lab, students will have achieved the following goals:

- Explore several aspects of the HTTP protocol
- Observe the basic GET/response interaction, HTTP message formats

1.3. Background

Having introduced the Wireshark packet analyzer in the introductory lab, we're now ready to use Wireshark to investigate protocols in operation, like HTTP, which is a common language of the modern global Internet. The world's web browsers, servers and related web applications all talk to each other through HTTP, the Hypertext Transfer Protocol. Before proceeding to the experiments, read introductions to some general terms used in this lab, to avoid any confusion.

2.3.1. What is a Web page?

A Web page (also called a document) consists of objects. An object is a simple file – such as an HTML file, a JPEG image, a GIF image, a Java applet, an audio clip, etc. – that is addressable by a single URL. Most Web pages consist of a base HTML file and several referenced objects. For example, if a Web page contains HTML text and five JPEG images, then the Web page has six objects: the base HTML file plus the five images. The base HTML file references the other objects in the page with the objects' URLs. Each URL has two components: the host name of the server that houses the object and the object's path name. For example, the URL www.someSchool.edu/someDepartment/picture.gif has www.someSchool.edu for a host name and /someDepartment/picture.gif for a path name.

2.3.2. What is a Web browser?

A browser is a user agent for the Web; it displays to the user the requested Web page and provides numerous navigational and configuration features. Web browsers also implement the client side of HTTP. Thus, in the context of the Web, we will interchangeably use the words "browser" and "client". Popular Web browsers include Google Chrome, Netscape Communicator and Microsoft Explorer.

2.3.3. What is a Web server?

A Web server houses Web objects, each addressable by a URL. Web servers also implement the server side of HTTP. Popular Web servers include Apache, Microsoft Internet Information Server, and the Netscape Enterprise Server. (Netcraft provides a nice survey of Web server penetration [Netcraft].)

Introduction to HTTP

The Hypertext Transfer Protocol (HTTP), the Web's application-layer protocol, is at the heart of the Web. HTTP is implemented in two programs: a client program and server program. The client program and server programs, executing on different end systems, talk to each other by exchanging HTTP messages. HTTP defines the structure of these messages and how the client and server exchange the messages. HTTP defines how Web clients (i.e., browsers) request Web pages from servers (i.e., Web servers) and how servers transfer Web pages to clients. When a user requests a Web page (e.g., clicks on a hyperlink), the browser sends HTTP request messages for the objects in the page to the server. The server receives the requests and responds with HTTP response messages that contain the objects.

1.4. Lab procedure

For all the experiments, we will use Wireshark packet analyzer that we used in the lab 1.

The basic HTTP GET/response interaction

2.4.1.1. Aim of this exercise

We will now learn about what packets are exchanged during an HTTP conversation – we will learn about the HTTP GET message that is sent from the HTTP client to the HTTP server and the HTTP message that is sent as response to this message.

2.4.1.2. Procedure

Follow the steps below to complete this exercise and to provide answers to the questions below:

- Start up your web browser.
- Start up the Wireshark packet sniffer, as described in lab 1 (but don't yet begin packet capture). Enter "http" (just the letters, not the quotation marks) in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window. (We're only interested in the HTTP protocol here, and don't want to see the clutter of all captured packes).
- Wait a bit more than one minute (we'll see why shortly), and then begin Wireshark packet capture.
- Enter the following to your browser http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html.Your browser should display the very simple, one-line HTML file.
- Stop Wireshark packet capture.

The example in Figure 2.1 shows in the packet-listing window that two HTTP messages were captured: the GET message (from your browser to the gaia.cs.umass.edu web server) and the response message from the server to your browser. The packet-contents window shows details of the selected message (in this case the HTTP GET message, which is highlighted in the packet- listing window). Recall that since the HTTP message was carried inside a TCP segment, which was carried inside an IP datagram, which was carried within an Ethernet frame, Wireshark displays the Frame, Ethernet, IP, and TCP packet information as well.

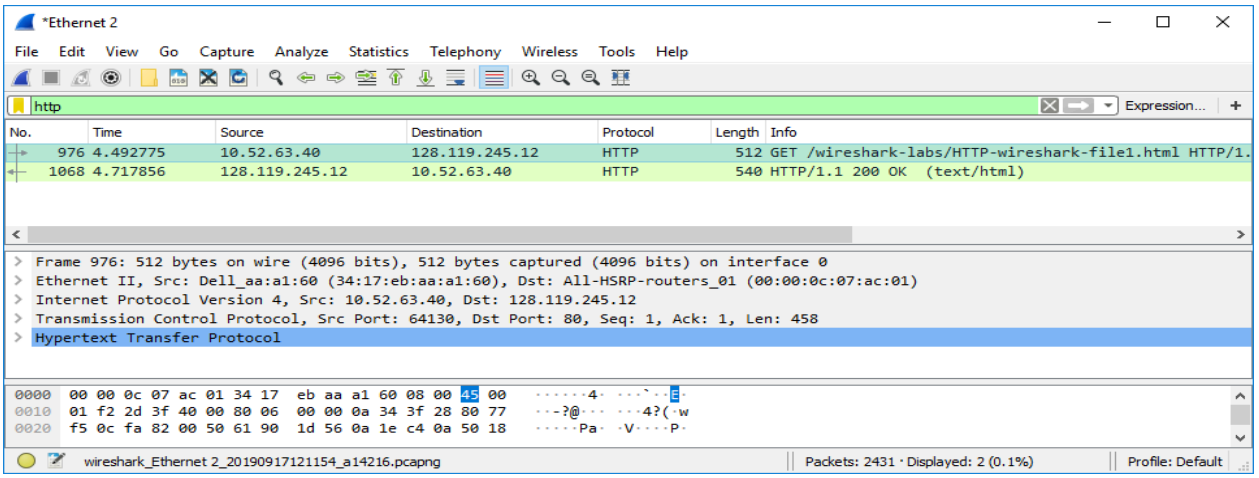


Figure 2.1: Wireshark display after http://gaia.cs.umass.edu/wireshark-labs/ HTTP-wireshark-file1.html has been retrieved by your browser

By looking at the information in the HTTP GET and response messages, answer the following questions.

2.4.1.3. Questions

1. Which version of HTTP is the browser running 1.0 or 1.1? Which HTTP version is the server running? *Paste screenshots and accompanying text to answer this question.*

[3 marks]

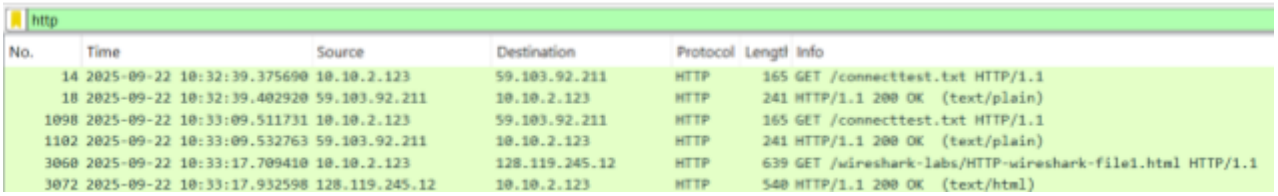
From your screenshot you can see first line (Packet 14): "GET /connecttest.txt HTTP/1.1"

This shows that **browser is running HTTP/1.1**.

Look at the response (from server to browser)

From your screenshot you can see second line (Packet 18): "HTTP/1.1 200 ok (text/plain)"

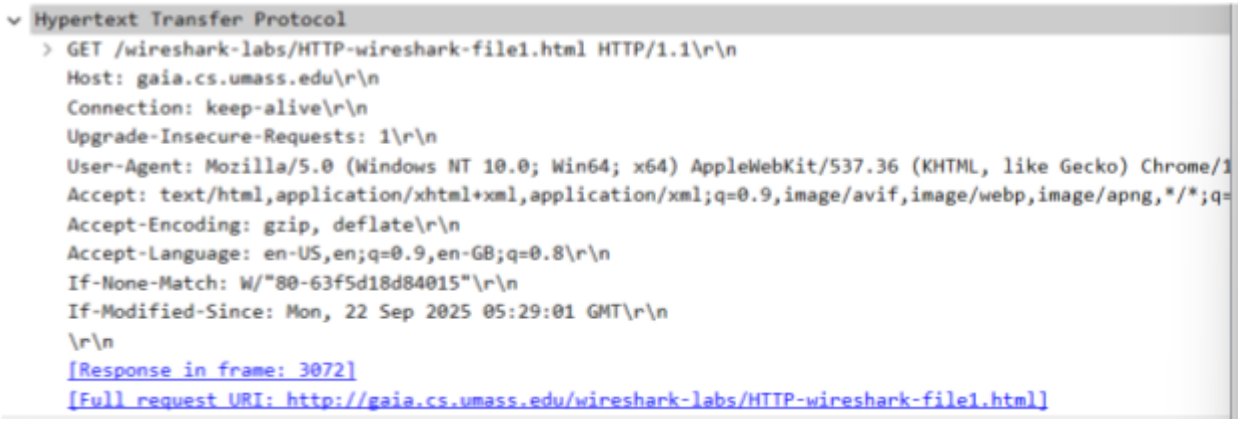
This shows the **server is also running HTTP/1.1**.



2. What languages (if any) does the browser indicate that it can accept to the server? *Paste screenshot (containing referenced item) and accompanying text to answer this question.*

[2 marks]

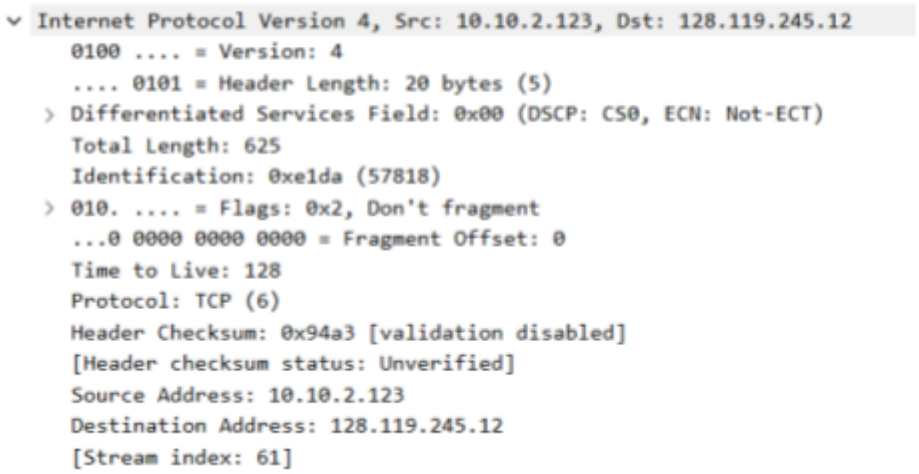
By expanding the Hypertext Transfer Protocol section, in this section we see that "**Accept-Language: en-US,en;q=0.9,en-GB;q=0.8\r\n**"
In this case, the browser specify that **Accept-Language: en-US**. It means the browser prefer **English (United State)**.



3. What is the IP address of your computer and of the gaia.cs.umass.edu server? *Describe how you determined these IP addresses.*

From the screen shot we can see the source and destination IP address. In this case
IP address of my computer is 10.10.2.123
IP address of giga.cs.umass.edu is 128.199.245.12
How to determine:
Selected the **HTTP GET packet** in Wireshark.

I Expanded the **Internet Protocol (IP)** section.
Read the **Source** and **Destination** IP addresses.
Source = my machine
Destination = the server I'm contacting (gaia.cs.umass.edu).



Application Layer Protocol: HyperText Transfer Protocol (HTTP)

[2 marks]

4. What is the status code returned from the server to your browser? *Paste screenshot (containing referenced item) and accompanying text to answer this question.*

[2 marks]

From this screen shot we can see status code return from server, and that is "HTTP/1.1 200 OK". This indicates that the request was successful and server return the file.

```
Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Mon, 22 Sep 2025 05:33:18 GMT\r\n
    Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.
    Last-Modified: Mon, 22 Sep 2025 05:33:02 GMT\r\n
    ETag: "80-63f5d272e8061"\r\n
    Accept-Ranges: bytes\r\n
  > Content-Length: 128\r\n
    Keep-Alive: timeout=5, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: text/html; charset=UTF-8\r\n
    \r\n
    [Request in frame: 3060]
```

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

[2 marks]

From this screen shot we can see Last Modified "Last-Modified: Mon, 22 Sep 2025 05:33:18 GMT". This indicates that the server last updated the file at that specific date and time..

```
Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Mon, 22 Sep 2025 05:33:18 GMT\r\n
    Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips
    Last-Modified: Mon, 22 Sep 2025 05:33:02 GMT\r\n
```

6. How many bytes of content are being returned to your browser? *Describe how you determined this*

[2 marks]

From this screen shot we can see that the number of bytes of content returned is given by the content length header in the HTTP response. In this case, the server specifies: Content-Length: 128

```
Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Mon, 22 Sep 2025 05:33:18 GMT\r\n
    Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips
    Last-Modified: Mon, 22 Sep 2025 05:33:02 GMT\r\n
    ETag: "80-63f5d272e8061"\r\n
    Accept-Ranges: bytes\r\n
  > Content-Length: 128\r\n
```

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. *Paste screenshot (containing referenced item) and accompanying text to answer this question.*

By inspecting the raw data in the packet-content window, I found additional HTTP headers that are not shown in the packet-listing summary.
And the response contains: “Connection: Keep-Alive”
This header was visible in the raw data but not displayed in the packet-listing window

[2 marks]

The HTTP CONDITIONAL GET/response interaction

2.4.1.4. Aim of this exercise

We will now learn about a variant of the HTTP GET request message that we've seen earlier. We will note how the HTTP CONDITIONAL GET request and the reply to such a request differs from a simple HTTP GET request (which we talked about in exercise 2.4.2).

2.4.1.5. Procedure

The following are the steps for this exercise:

1. Start up your web browser, and make sure your browser's cache is cleared, as discussed above.
2. Start up the Wireshark packet sniffer.
3. Enter the following URL into your browser http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html. Your browser should display a very simple five-line HTML file.
4. Quickly enter the same URL into your browser again (or simply select the refresh button on your browser).
5. Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.
6. Filter out all the non-HTTP packets and focus on the HTTP header information in the packet-header detail window.

7. By looking at the information in the HTTP GET and response messages (the first two messages), answer the following questions.

2.4.1.6. Questions

1. Inspect the contents of the first HTTP GET request from the browser to the server. Do you see an "IF-MODIFIED-SINCE" line in the HTTP GET? *Paste screenshot (containing referenced item) and accompanying text to answer this question.*

[2 marks]

In the first HTTP GET request (Frame 3473), I did not seen and the browser did not include an If-Modified_Since header. As we can see in the screenshot , the request only contains headers such as Host, Connection etc. This confirms that the first request was a normal HTTP GET.

```
Hypertext Transfer Protocol
  GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1\r\n
  Host: gaia.cs.umass.edu\r\n
  Connection: keep-alive\r\n
  Upgrade-Insecure-Requests: 1\r\n
  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36 Edg/140.0.0.0\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;v=b3;q=0.7\r\n
  Accept-Encoding: gzip, deflate\r\n
  Accept-Language: en-US,en;q=0.9,en-GB;q=0.8\r\n
  \r\n
  [Response in frame: 3528]
  [Full request URI: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html]
```

2. Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell? *Paste screenshot (containing referenced item) and accompanying text to answer this question.*

[3 marks]

In the server's first response (Frame 3528), the status code is 200 OK. This confirms that the server explicitly returned the contents of the requested file HTTP-wireshark-file2.html. The presence of Content-Length and the HTML file data in the packet further indicates that the file was indeed transmitted to the browser.

In this screen shot we can see **status code** is 200 OK and the second screen shot confirms the return content.

```
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Mon, 22 Sep 2025 06:56:14 GMT\r\n
  Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.33 mod_perl/2.0.11 Perl/v5.16.3\
  Last-Modified: Mon, 22 Sep 2025 05:59:01 GMT\r\n
  ETag: "173-63f5d84242929"\r\n
  Accept-Ranges: bytes\r\n
  Content-Length: 371\r\n
  Keep-Alive: timeout=5, max=100\r\n
  Connection: Keep-Alive\r\n
  Content-Type: text/html; charset=UTF-8\r\n
  \r\n
  [Request in frame: 3473]
```

3. Does the response indicate the last time that the requested file was modified? *Paste screenshot (containing referenced item) and accompanying text to answer this question.* **[2 marks]**

Yes ,the response in this screenshot shows the **"Last Modified"** column, which indicates the last time the requested file was modified. "**Last -Modified: Mon 22 sep 2025 05:59:01 GMT**".
This confirms that the response **does indicate the last time that the requested file was modified**.

```
Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Mon, 22 Sep 2025 06:56:14 GMT\r\n
    Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.33 mod_perl/2.0.11 Perl/v5.16.3\
    Last-Modified: Mon, 22 Sep 2025 05:59:01 GMT\r\n
```

4. Now inspect the contents of the second HTTP GET request from the browser to the server. Do you see an "IF-MODIFIED-SINCE:" line in the HTTP GET? If so, what information is contained in the "IF-MODIFIED-SINCE:" header? *Paste screenshot (containing referenced item) and accompanying text to answer this question.*

[2 marks]

Yes in the second HTTP GET request, the header **If-Modified-Since:** is present.
"**If-Modified-Since: Mon, 22 Sep 2025 05:59:01 GMT**"
This shows that the browser is asking the server only to return the file if it has been modified after Mon, 22 Sep 2025 05:59:01 GMT.

```
v Hypertext Transfer Protocol
  > GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1\r\n
    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 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHT
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: en-US,en;q=0.9,en-GB;q=0.8\r\n
    If-None-Match: "173-63f5d84242929"\r\n
    If-Modified-Since: Mon, 22 Sep 2025 05:59:01 GMT\r\n
```

5. 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. *Paste screenshot (containing referenced item) and accompanying text to answer this question.*

[3 marks]

In the packet capture, the server responds with: " **HTTP/1.1 304 Not Modified**"
Status code = 304
Status phrase = Not Modified
This means the server **did not explicitly return the contents of the file**. Instead, it told the browser:
"Your cached copy is still valid, so keep using that. No need to resend the whole file."