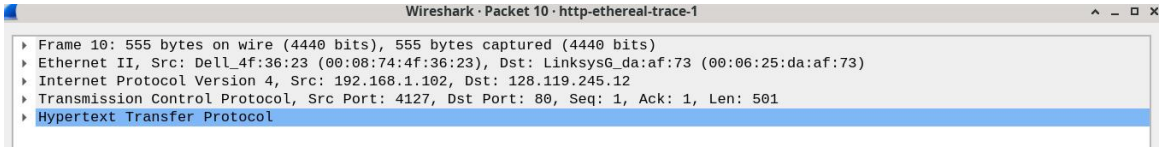
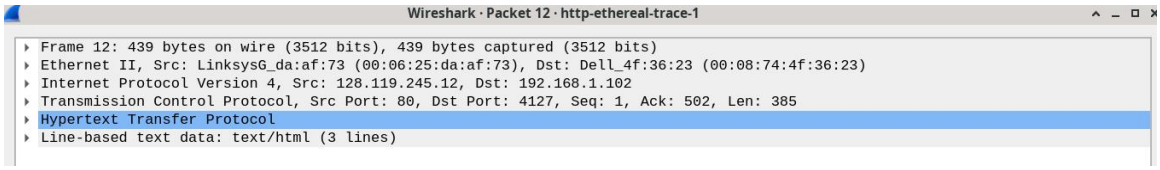
COMP9331 – Lab2

Exercise 3: Using Wireshark to understand basic HTTP request/response messages (2.5 marks)

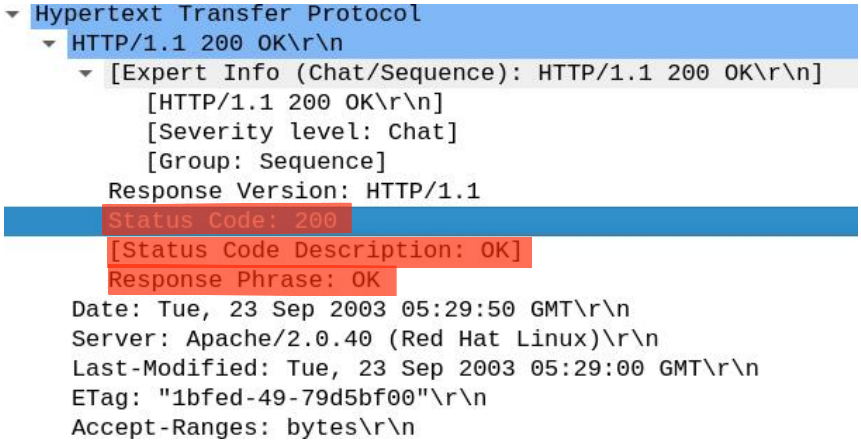
HTTP Request



HTTP Response

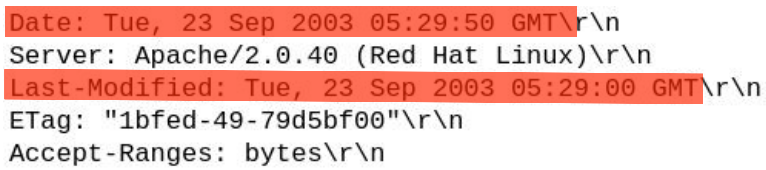


Question 1: What is the status code and phrase returned from the server to the client browser?



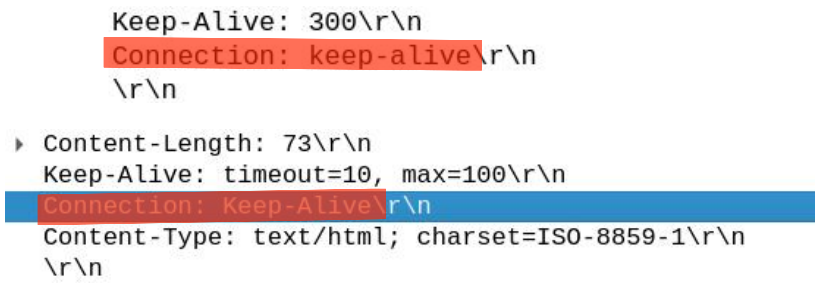
Answer: The server returns a status code of *200* with the phrase *OK*.

Question 2: When was the HTML file the browser retrieved last modified at the server? Does the response also contain a DATE header? How are these two fields different?



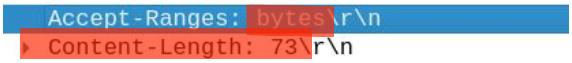
Answer: The HTML file was last modified on *Tue, 23 Sep 2003 05:29:00 GMT*, and the response contains a DATE header with the value *Tue, 23 Sep 2003 05:29:50 GMT*. The last modified header indicates when the resource itself was last changed, while the DATE header shows when the response was sent from the server.

Question 3: Is the connection established between the browser and the server persistent or non-persistent? How can you infer this?



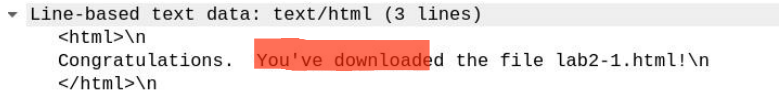
Answer: The connection is persistent. This is indicated by the *Connection: Keep-Alive* header in response, which suggests that the server intends to keep the connection open for further requests.

Question 4: How many bytes of content are being returned to the browser?



Answer: The response body contains *73 bytes* of content, as indicated by the *Content-Length: 73* headers.

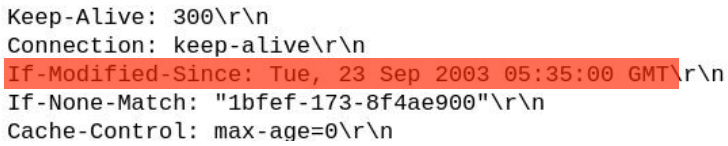
Question 5: What is the data contained inside the HTTP response packet?



Answer: The HTTP response packet includes the following data: *"Congratulations. You've downloaded the file lab2-1.html!"*.

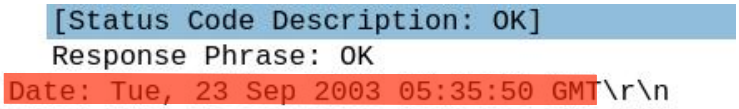
Exercise 4: Using Wireshark to understand the HTTP CONDITIONAL GET/response interaction (2.5 marks)

Question 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?



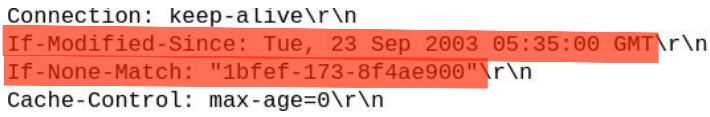
Answer: No *If-Modified-Since* header in the first HTTP GET request, but If-*Modified-Since* header present in the second request.

Question 2: Does the HTTP response from the server indicate the last time the requested file was modified?



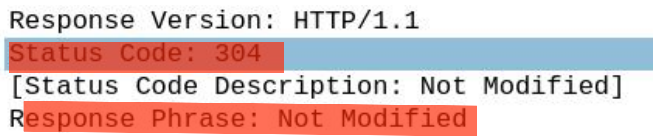
Answer: Yes, the *Last-Modified* header indicates that the requested file was last modified on *Tue, 23 Sep 2003 05:35:00 GMT*.

Question 3: Now inspect the contents of the second HTTP GET request from the browser to the server. Do you see the "IF-MODIFIED-SINCE:" and "IF-NONE-MATCH" lines in the HTTP GET? If so, what information is contained in these headers?



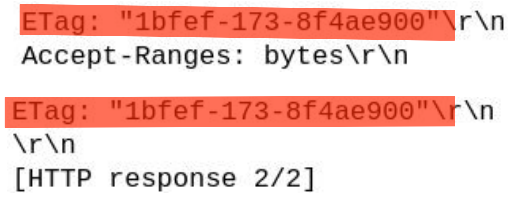
Answer: Yes, the second HTTP GET request includes both the *If-Modified-Since* and *If-None-Match headers*. If-Modified-Since: *Tue, 23 Sep 2003 05:35:00 GMT*. It shows the client's last known modification time for the requested resource. If-None-Match: *"lbfef-173-8f4ae900"* , which is the entity tag (ETag) assigned to the resource during its last modification. These headers are used by the client to check if the resource has been modified since the specified date or if its ETag matches the current version on the server.

Question 4: 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 file's contents? Explain.



Answer: The server responds to the second HTTP GET request with a status code of *304* and the response phrase *Not Modified*. This indicates that the requested resource has not been modified since the client's last request. Therefore, the server does not return the file's contents explicitly.

Question 5: What is the value of the ETag field in the 2nd response message, and how is it used? Is the ETag value the same as in the 1st response?



Answer: The value of the *ETag* in the second response message is "lbfef-173-8f4ae900", which matches the value in the first response. An ETag is a unique identifier assigned by the server to a specific version of a resource. By comparing the ETag value sent by the client in the *If-None-Match* header with the current ETag, the server can determine whether the client's cached version is up to date.

Exercise 5: Ping Client (5 marks, submit source code separately, include sample output)

Sample Output:

```
z5484442@vx22:~/9331/lab02$ python3 PingClient.py 127.0.0.1 12000
PING to 127.0.0.1, seq=11113, rtt=timeout
PING to 127.0.0.1, seq=11114, rtt=17 ms
PING to 127.0.0.1, seq=11115, rtt=timeout
PING to 127.0.0.1, seq=11116, rtt=timeout
PING to 127.0.0.1, seq=11117, rtt=timeout
PING to 127.0.0.1, seq=11118, rtt=timeout
PING to 127.0.0.1, seq=11119, rtt=47 ms
PING to 127.0.0.1, seq=11120, rtt=28 ms
PING to 127.0.0.1, seq=11121, rtt=75 ms
PING to 127.0.0.1, seq=11122, rtt=26 ms
PING to 127.0.0.1, seq=11123, rtt=1 ms
PING to 127.0.0.1, seq=11124, rtt=timeout
PING to 127.0.0.1, seq=11125, rtt=timeout
PING to 127.0.0.1, seq=11126, rtt=timeout
PING to 127.0.0.1, seq=11127, rtt=timeout

----- Detailed Report -----
Total packets sent: 15
Packets acknowledged: 6
Packet loss: 60.0%
Minimum RTT: 1 ms, Maximum RTT: 75 ms, Average RTT: 32.33 ms
Total transmission time: 5605 ms
Jitter: 34.00 ms
z5484442@vx22:~/9331/lab02$ python3 PingClient.py 127.0.0.1 12000
PING to 127.0.0.1, seq=16303, rtt=135 ms
PING to 127.0.0.1, seq=16304, rtt=149 ms
PING to 127.0.0.1, seq=16305, rtt=97 ms
PING to 127.0.0.1, seq=16306, rtt=58 ms
PING to 127.0.0.1, seq=16307, rtt=timeout
PING to 127.0.0.1, seq=16308, rtt=131 ms
PING to 127.0.0.1, seq=16309, rtt=86 ms
PING to 127.0.0.1, seq=16310, rtt=42 ms
PING to 127.0.0.1, seq=16311, rtt=199 ms
PING to 127.0.0.1, seq=16312, rtt=timeout
PING to 127.0.0.1, seq=16313, rtt=39 ms
PING to 127.0.0.1, seq=16314, rtt=136 ms
PING to 127.0.0.1, seq=16315, rtt=timeout
PING to 127.0.0.1, seq=16316, rtt=timeout
PING to 127.0.0.1, seq=16317, rtt=13 ms

----- Detailed Report -----
Total packets sent: 15
Packets acknowledged: 11
Packet loss: 26.7%
Minimum RTT: 13 ms, Maximum RTT: 199 ms, Average RTT: 98.64 ms
Total transmission time: 3489 ms
Jitter: 80.40 ms
```