CN LAB

Experiment – 7

PACKET CAPTURING AND ANALYSIS WITH WIRESHARK

Submitted To

Prof. Mangal Singh

Computer Networks

SIT, Pune

Submitted By

Roshan Kumar Yadav

21070126130

AIML B3

AIM

Objective of this lab is to get familiar with the packet sniffer tool “Wireshark” and conduct the packet capturing and packet analysis for various tasks related to HTTP protocol.

THEORY

Wireshark is a network protocol analyzer that is widely used for capturing, analyzing, and inspecting data exchanged over computer networks. When it comes to HTTP (Hypertext Transfer Protocol) message analysis, Wireshark provides the ability to capture and dissect network packets, allowing users to examine the structure and content of HTTP requests and responses. It offers filtering options to isolate HTTP traffic, reassembles streams for a complete view of conversations, and helps in understanding request-response patterns, making it an invaluable tool for network administrators, security professionals, and developers when troubleshooting, optimizing, and securing web applications.

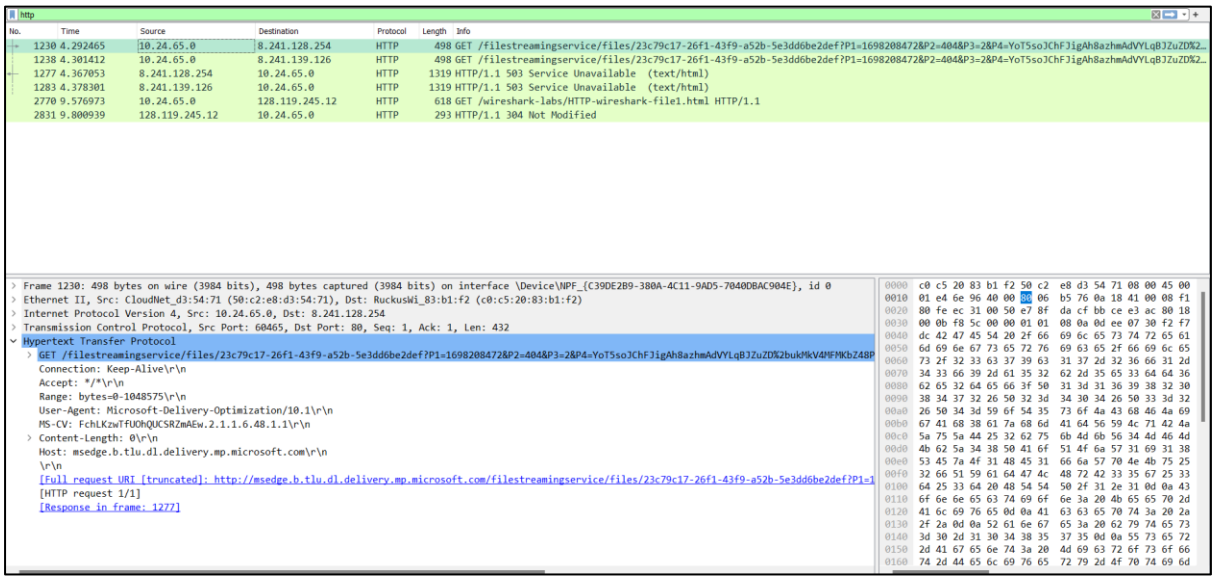
OSERVATION

Getting basic information on HTTP Protocol

1. **Please note down the IP address of your machine and the destination machine (gaia.cs.umass).**

   My Machine IP Address: 10.24.65.0

   Destination Machine IP Address: 8.241.128.254



2. **What do you observe in the HTTP request message.**

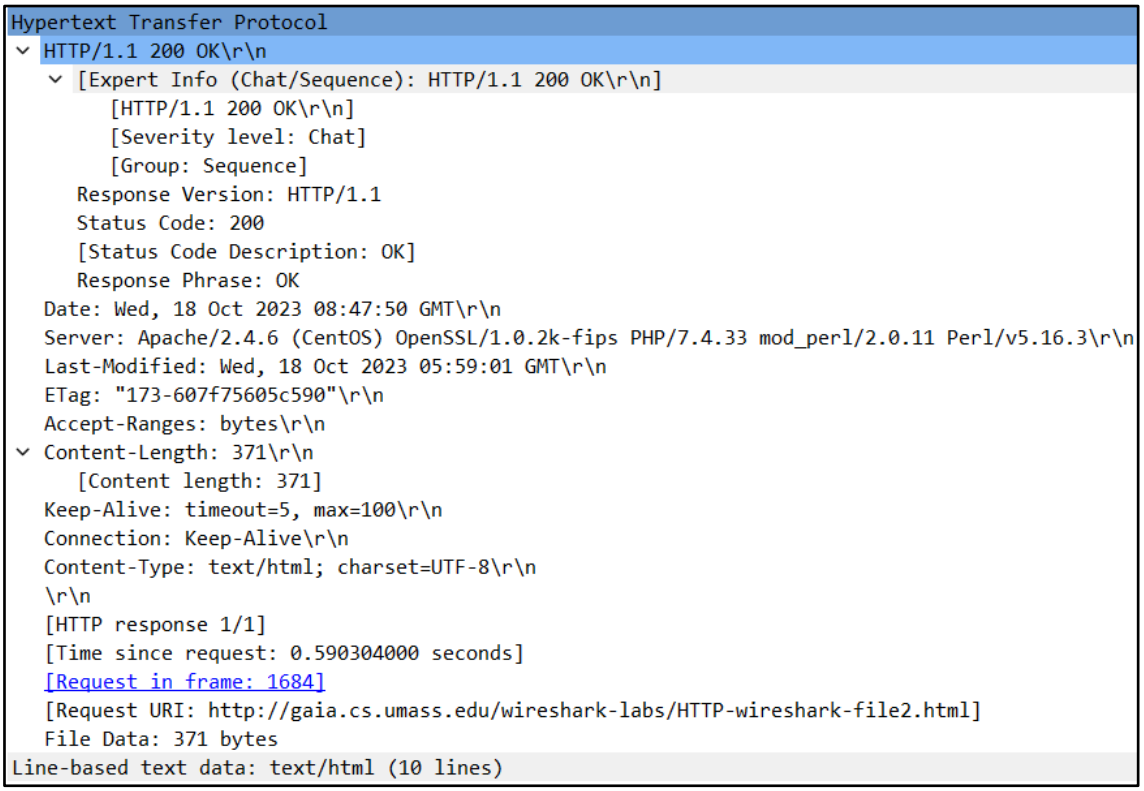
A packet with a source IP address corresponding to the source 10.24.65.0 and a destination IP address 8.241.128.254 corresponding to the web server "http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html". The packet typically uses the HTTP protocol and contain the HTTP (HHTP/1.1) method GET in the message payload. The URI, HTTP version, and various headers, such as User-Agent, Host, Connection, MS-CV, Content-length, Host and Accept, are also visible in the packet.

3. **Write down the details of the HTTP response message such as status code, content length and file modified last time.**

   Status Code: 200

   Content-Length: 371

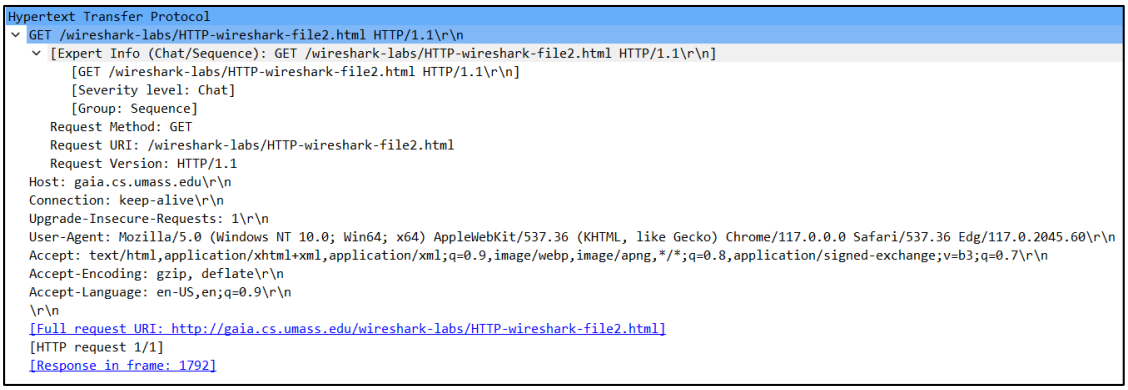
   Last-Modified: Wed, 18 Oct 2023 08:47:50 GMT



4. **Write down your interesting observations for the GET request and response messages.**

GET Message

1. **HTTP Method (GET)**: Requests the specified resource.
2. **URI ("/wireshark-labs/HTTP-wireshark-file2.html")**: Specifies the path to the requested resource.
3. **HTTP Version (HTTP/1.1)**: Indicates the version of the HTTP protocol used.
4. **Host (gaia.cs.umass.edu)**: Specifies the domain of the server to which the request is sent.
5. **Connection (keep-alive)**: Requests that the connection be kept open for potential future requests, reducing latency.
6. **Upgrade-Insecure-Requests (1)**: Indicates a preference for secure HTTPS connections.
7. **User-Agent**: Provides information about the client making the request, including the browser and operating system.
8. **Accept**: Specifies the types of media (MIME types) that the client can handle.
9. **Accept-Encoding (gzip, deflate)**: Informs the server about the compression methods supported by the client.
10. **Accept-Language (en-US, en;q=0.9)**: Indicates the preferred languages for content, with a preference for American English.



Response Message

1. **Status Code (200 OK)**: Indicates that the request was successful.
2. **Date (Wed, 18 Oct 2023 08:47:50 GMT)**: Shows the response's generation timestamp.
3. **Server (Apache/2.4.6, CentOS, etc.)**: Provides server software and version details.
4. **Last-Modified (Wed, 18 Oct 2023 05:59:01 GMT)**: Indicates when the resource was last modified.
5. **ETag ("173-607f75605c590")**: An entity tag for resource version tracking.
6. **Accept-Ranges (bytes)**: Signifies support for byte range requests.

7. **Content-Length (371)**: Specifies the response body size in bytes.
8. **Keep-Alive (timeout=5, max=100)**: Describes connection management settings.
9. **Connection (Keep-Alive)**: Informs that the connection will be maintained for future requests.
10. **Content-Type (text/html; charset=UTF-8)**: Declares the content as HTML with UTF-8 encoding.

```
Hypertext Transfer Protocol
▾ HTTP/1.1 200 OK\r\n
  ▾ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      [HTTP/1.1 200 OK\r\n]
      [Severity level: Chat]
      [Group: Sequence]
    Response Version: HTTP/1.1
    Status Code: 200
    [Status Code Description: OK]
    Response Phrase: OK
  Date: Wed, 18 Oct 2023 08:47:50 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\r\n
  Last-Modified: Wed, 18 Oct 2023 05:59:01 GMT\r\n
  ETag: "173-607f75605c590"\r\n
  Accept-Ranges: bytes\r\n
▾ Content-Length: 371\r\n
    [Content length: 371]
  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 1/1]
  [Time since request: 0.590304000 seconds]
  [Request in frame: 1684]
  [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)
```

5. **As you are retrieving long document, how many request packets are sent from the client to the server.**

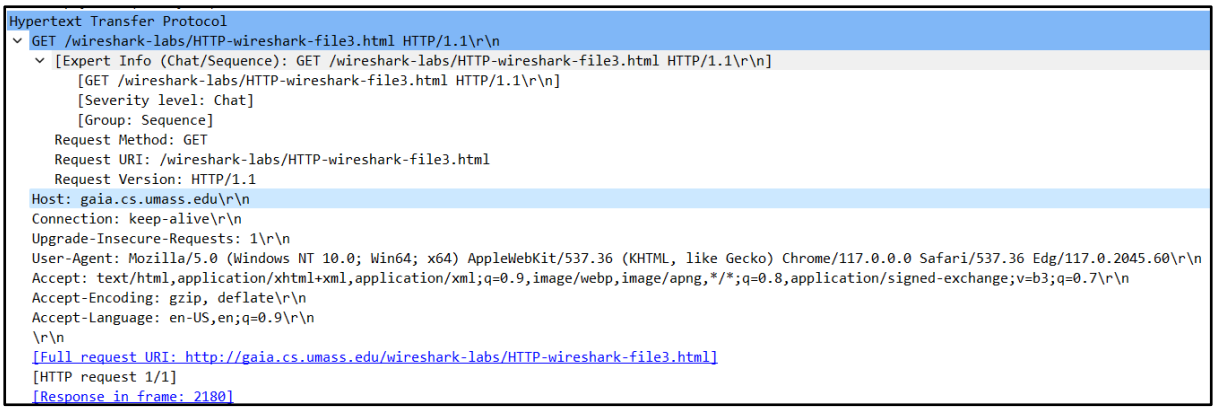
http

No.	Time	Source	Destination	Protocol	Length	Info
2116	5.171141	10.24.65.0	128.119.245.12	HTTP	533	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
2180	5.415156	128.119.245.12	10.24.65.0	HTTP	535	HTTP/1.1 200 OK (text/html)

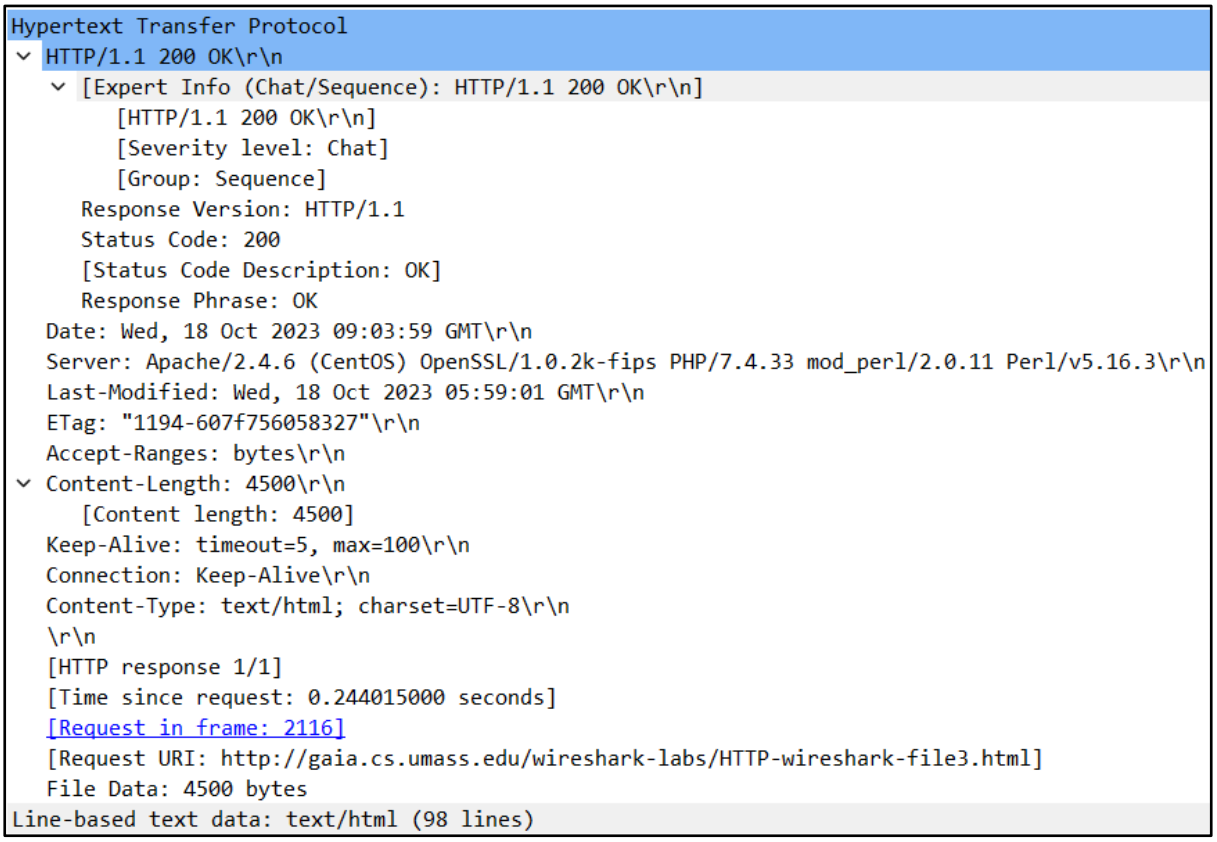
1 request packet (Packet No. 2116) was sent as HTTP GET message.

1 response packet (Packet No. 2180) was received as HTTP response message.

GET Message



Response Message



6. **Write down your understanding on how the HTTP long file is supported by underlying TCP.**

HTTP long file support is achieved using TCP's reliable and ordered delivery of data. When a client requests a large file from a server, the server breaks the file down into smaller chunks. These chunks are then sent to the client one at a time. The client reassembles the chunks in the correct order to create the original file.

TCP ensures that all the chunks are delivered to the client in the correct order. This is done by using sequence numbers. Each chunk is assigned a unique sequence number. The client acknowledges each chunk that it receives. The server then sends the next chunk in the sequence. If the client does not acknowledge a chunk, the server will resend it.

TCP also ensures that all the chunks are delivered reliably. This is done by using a checksum. Each chunk is given a checksum. The client calculates the checksum of the chunk and compares it to the checksum that was sent by the server. If the checksums do not match, the client knows that the chunk has been corrupted and requests that the server resend it.

HTTP supports long files by using the *Content-Length* header and TCP's *window size* mechanism. These mechanisms allow HTTP to send and receive files of any size.

```
[2 Reassembled TCP Segments (4861 bytes): #283(4380), #284(481)]
   [Frame: 283, payload: 0-4379 (4380 bytes)]
   [Frame: 284, payload: 4380-4860 (481 bytes)]
   [Segment count: 2]
   [Reassembled TCP length: 4861]
   [Reassembled TCP Data: 485454502f312e3120323030204f4b0d0a446174653a2053756e2c203232204f63742032…]
```

```
Transmission Control Protocol, Src Port: 60643, Dst Port: 80, Seq: 1, Ack: 1, Len: 479
   Source Port: 60643
   Destination Port: 80
   [Stream index: 3]
   [Conversation completeness: Complete, WITH_DATA (31)]
   [TCP Segment Len: 479]
   Sequence Number: 1    (relative sequence number)
   Sequence Number (raw): 4251822730
   [Next Sequence Number: 480    (relative sequence number)]
   Acknowledgment Number: 1    (relative ack number)
   Acknowledgment number (raw): 4028137535
   0101 .... = Header Length: 20 bytes (5)
 > Flags: 0x018 (PSH, ACK)
   Window: 513
   [Calculated window size: 131328]
   [Window size scaling factor: 256]
   Checksum: 0xe3d4 [unverified]
   [Checksum Status: Unverified]
   Urgent Pointer: 0
 v [Timestamps]
      [Time since first frame in this TCP stream: 0.206500000 seconds]
      [Time since previous frame in this TCP stream: 0.000489000 seconds]
 v [SEQ/ACK analysis]
      [iRTT: 0.206011000 seconds]
      [Bytes in flight: 479]
      [Bytes sent since last PSH flag: 479]
   TCP payload (479 bytes)
```

7. **Inspect the packet which contains the status code and phrase of the response message.**

Packet No. 284

```
Frame 284: 535 bytes on wire (4280 bits), 535 bytes captured (4280 bits) on interface \Device\NPF_{C39DE2B9-380A-4C11-9AD5-7040DBAC904E}, id 0
   Section number: 1
 > Interface id: 0 (\Device\NPF_{C39DE2B9-380A-4C11-9AD5-7040DBAC904E})
   Encapsulation type: Ethernet (1)
   Arrival Time: Oct 22, 2023 12:04:19.081469000 India Standard Time
   [Time shift for this packet: 0.000000000 seconds]
   Epoch Time: 1697956459.081469000 seconds
   [Time delta from previous captured frame: 0.000000000 seconds]
   [Time delta from previous displayed frame: 0.209847000 seconds]
   [Time since reference or first frame: 4.269644000 seconds]
   Frame Number: 284
   Frame Length: 535 bytes (4280 bits)
   Capture Length: 535 bytes (4280 bits)
   [Frame is marked: False]
   [Frame is ignored: False]
   [Protocols in frame: eth:ethertype:ip:tcp:http:data-text-lines]
   [Coloring Rule Name: HTTP]
   [Coloring Rule String: http || tcp.port == 80 || http2]
```

Status Code: 200

Response phrase: OK

```
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
```

8. Write down your interesting observations for the request and response messages while performing this task.

```
2243 4.007220   10.24.65.0      128.119.245.12   HTTP   548 GET /wireshark-labs/protected_pages/HTTP-wiresharkfile5.html HTTP/1.1
2304 4.215787   128.119.245.12  10.24.65.0       HTTP   771 HTTP/1.1 401 Unauthorized  (text/html)
3222 8.538516   10.24.65.0      23.213.0.186     HTTP   487 GET /filestreamingservice/files/2342e9ca-cbe9-4c24-a8a2-c1230c4d4894?P1=1697959196&P2=404&P3=2&P4=LVMtaxeyhU1Pz7XKfYyeFQkDktklCVORds…
3228 8.561051   23.213.0.186    10.24.65.0       HTTP  1319 HTTP/1.1 503 Service Unavailable  (text/html)
3264 8.681054   10.24.65.0      128.119.245.12   HTTP   633 GET /wireshark-labs/protected_pages/HTTP-wiresharkfile5.html HTTP/1.1
3296 8.889824   128.119.245.12  10.24.65.0       HTTP   582 HTTP/1.1 404 Not Found  (text/html)
```

Request Message

```
Frame 2243: 548 bytes on wire (4384 bits), 548 bytes captured (4384 bits) on interface \Device\NPF_{C39DE2B9-380A-4C11-9AD5-7040DBAC904E}, id 0
Ethernet II, Src: CloudNet_d3:54:71 (50:c2:e8:d3:54:71), Dst: RuckusWi_83:b1:f2 (c0:c5:20:83:b1:f2)
Internet Protocol Version 4, Src: 10.24.65.0, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 62840, Dst Port: 80, Seq: 1, Ack: 1, Len: 494
Hypertext Transfer Protocol
 GET /wireshark-labs/protected_pages/HTTP-wiresharkfile5.html HTTP/1.1\r\n
   [Expert Info (Chat/Sequence): GET /wireshark-labs/protected_pages/HTTP-wiresharkfile5.html HTTP/1.1\r\n]
   Request Method: GET
   Request URI: /wireshark-labs/protected_pages/HTTP-wiresharkfile5.html
   Request Version: HTTP/1.1
 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/118.0.0.0 Safari/537.36 Edg/118.0.2088.57\r\n
 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,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\r\n
 \r\n
 [Full request URI: http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wiresharkfile5.html]
 [HTTP request 1/2]
 [Response in frame: 2304]
 [Next request in frame: 3264]
```

```
Frame 3264: 633 bytes on wire (5064 bits), 633 bytes captured (5064 bits) on interface \Device\NPF_{C39DE2B9-380A-4C11-9AD5-7040DBAC904E}, id 0
Ethernet II, Src: CloudNet_d3:54:71 (50:c2:e8:d3:54:71), Dst: RuckusWi_83:b1:f2 (c0:c5:20:83:b1:f2)
Internet Protocol Version 4, Src: 10.24.65.0, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 62840, Dst Port: 80, Seq: 495, Ack: 718, Len: 579
Hypertext Transfer Protocol
  GET /wireshark-labs/protected_pages/HTTP-wiresharkfile5.html HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET /wireshark-labs/protected_pages/HTTP-wiresharkfile5.html HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /wireshark-labs/protected_pages/HTTP-wiresharkfile5.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
  Authorization: Basic d2lyZXNoYXJrLXN0dWRlbnRzOm5ldHdvcms=\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/118.0.0.0 Safari/537.36 Edg/118.0.2088.57\r\n
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,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\r\n
  \r\n
  [Full request URI: http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wiresharkfile5.html]
  [HTTP request 2/2]
  [Prev request in frame: 2243]
  [Response in frame: 3296]
```

Response Message

```
Frame 2304: 771 bytes on wire (6168 bits), 771 bytes captured (6168 bits) on interface \Device\NPF_{C39DE2B9-380A-4C11-9AD5-7040DBAC904E}, id 0
Ethernet II, Src: RuckusWi_83:b1:f2 (c0:c5:20:83:b1:f2), Dst: CloudNet_d3:54:71 (50:c2:e8:d3:54:71)
Internet Protocol Version 4, Src: 128.119.245.12, Dst: 10.24.65.0
Transmission Control Protocol, Src Port: 80, Dst Port: 62840, Seq: 1, Ack: 495, 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
  Date: Sun, 22 Oct 2023 07:12:04 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\r\n
  WWW-Authenticate: Basic realm="wireshark-students only"\r\n
  Content-Length: 381\r\n
  Keep-Alive: timeout=5, max=100\r\n
  Connection: Keep-Alive\r\n
  Content-Type: text/html; charset=iso-8859-1\r\n
  \r\n
  [HTTP response 1/2]
  [Time since request: 0.208567000 seconds]
  [Request in frame: 2243]
  [Next request in frame: 3264]
  [Next response in frame: 3296]
  [Request URI: http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wiresharkfile5.html]
  File Data: 381 bytes
```

```
Frame 3296: 582 bytes on wire (4656 bits), 582 bytes captured (4656 bits) on interface \Device\NPF_{C39DE2B9-380A-4C11-9AD5-7040DBAC904E}, id 0
Ethernet II, Src: RuckusWi_83:b1:f2 (c0:c5:20:83:b1:f2), Dst: CloudNet_d3:54:71 (50:c2:e8:d3:54:71)
Internet Protocol Version 4, Src: 128.119.245.12, Dst: 10.24.65.0
Transmission Control Protocol, Src Port: 80, Dst Port: 62840, Seq: 718, Ack: 1074, Len: 528
Hypertext Transfer Protocol
  HTTP/1.1 404 Not Found\r\n
    [Expert Info (Chat/Sequence): HTTP/1.1 404 Not Found\r\n]
      Response Version: HTTP/1.1
      Status Code: 404
      [Status Code Description: Not Found]
      Response Phrase: Not Found
  Date: Sun, 22 Oct 2023 07:12:09 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\r\n
  Content-Length: 253\r\n
  Keep-Alive: timeout=5, max=99\r\n
  Connection: Keep-Alive\r\n
  Content-Type: text/html; charset=iso-8859-1\r\n
  \r\n
  [HTTP response 2/2]
  [Time since request: 0.208770000 seconds]
  [Prev request in frame: 2243]
  [Prev response in frame: 2304]
  [Request in frame: 3264]
  [Request URI: http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wiresharkfile5.html]
  File Data: 253 bytes
```

Observations

Request Message:

1. **HTTP Request Method**: The request message uses the "GET" method, indicating that the client is requesting a resource from the server.
2. **Host Header**: The "Host" header specifies the destination server as "gaia.cs.umass.edu," indicating the target of the request.
3. **User-Agent**: The User-Agent header reveals that the client is using the Chrome web browser on a Windows platform.
4. **Accept-Encoding**: The "Accept-Encoding" header shows that the client is willing to accept content encoded using gzip or deflate compression.

Response Message (401 Unauthorized):

1. **HTTP Status Code**: The server responds with a "401 Unauthorized" status code, indicating that access to the requested resource is denied.
2. **WWW-Authenticate**: The response includes a "WWW-Authenticate" header specifying "Basic" authentication with a realm of "wireshark-students only." This suggests that access to the resource requires authentication.
3. **Content-Length**: The "Content-Length" header indicates that the response body is 381 bytes in size.
4. **HTML Content**: The response body contains an HTML page with a message stating that access is unauthorized and explaining potential reasons for the denial.

Response Message (404 Not Found):

1. **HTTP Status Code**: In the second response, the server returns a "404 Not Found" status code, indicating that the requested URL does not exist on the server.
2. **Content-Length**: The "Content-Length" header in this response specifies that the response body is 253 bytes.
3. **HTML Content**: The response body contains an HTML page with a message indicating that the requested URL was not found on the server.
4. **Server Information**: The "Server" header in both response messages identifies the server as "Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.33 mod_perl/2.0.11 Perl/v5.16.3."
5. **Keep-Alive**: The "Connection" header in both responses indicates the use of "Keep-Alive" for persistent connections, with specified timeout values.
6. **Authorization Header**: In the second request, there is an "Authorization" header with a "Basic" authentication token, suggesting that the client is providing credentials to access the protected resource.