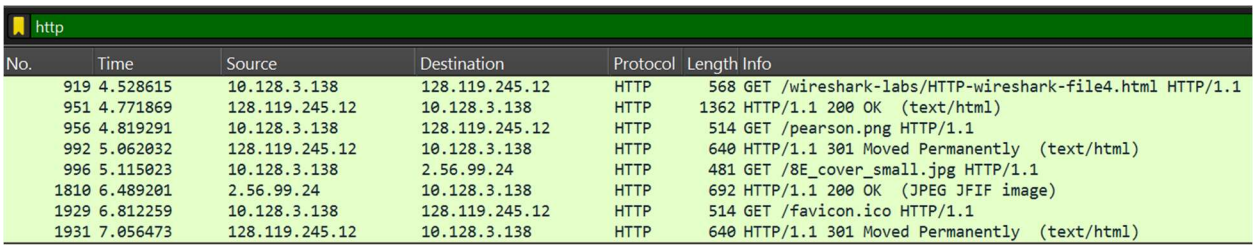
Part 1: Analyzing HTTP Traffic

Contents

1. How many HTTP GET request messages were transmitted by the browser? To which Internet addresses were these requests directed?



http

No.	Time	Source	Destination	Protocol	Length	Info
919	4.528615	10.128.3.138	128.119.245.12	HTTP	568	GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1.1
951	4.771869	128.119.245.12	10.128.3.138	HTTP	1362	HTTP/1.1 200 OK (text/html)
956	4.819291	10.128.3.138	128.119.245.12	HTTP	514	GET /pearson.png HTTP/1.1
992	5.062032	128.119.245.12	10.128.3.138	HTTP	640	HTTP/1.1 301 Moved Permanently (text/html)
996	5.115023	10.128.3.138	2.56.99.24	HTTP	481	GET /8E_cover_small.jpg HTTP/1.1
1810	6.489201	2.56.99.24	10.128.3.138	HTTP	692	HTTP/1.1 200 OK (JPEG JFIF image)
1929	6.812259	10.128.3.138	128.119.245.12	HTTP	514	GET /favicon.ico HTTP/1.1
1931	7.056473	128.119.245.12	10.128.3.138	HTTP	640	HTTP/1.1 301 Moved Permanently (text/html)

Có tổng cộng 3 yêu cầu HTTP GET được trình duyệt gửi đi, không tính favicon.ico:

- GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1.1
- GET /pearson.png HTTP/1.1
- GET /8E_cover_small.jpg HTTP/1.1

(Note: You should ignore any HTTP GET and response for favicon.ico. If you see a reference to this file, it is your browser automatically asking the server if it (the server) has a small icon file that should be displayed next to the displayed URL in your browser. We'll ignore references to this pesky file in this lab.) (Ross, 2005-2016)

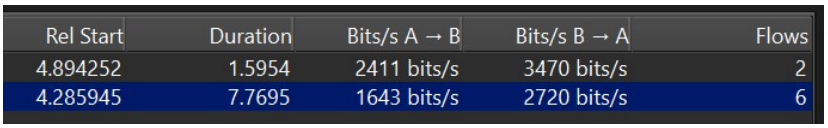
Các yêu cầu này được gửi đến 2 địa chỉ Internet khác nhau:

- 128.119.245.12 - cho file HTML và pearson.png
- 2.56.99.24 - cho 8E_cover_small.jpg

10.128.3.138: địa chỉ IP nguồn của laptop.

2. Determine whether the browser retrieved the two images **sequentially** or **concurrently** from their respective web servers, and provide an explanation for your conclusion by examining the timing of the requests and the source IP addresses.

Trình duyệt tải hai ảnh một cách **song song** từ hai web server khác nhau.

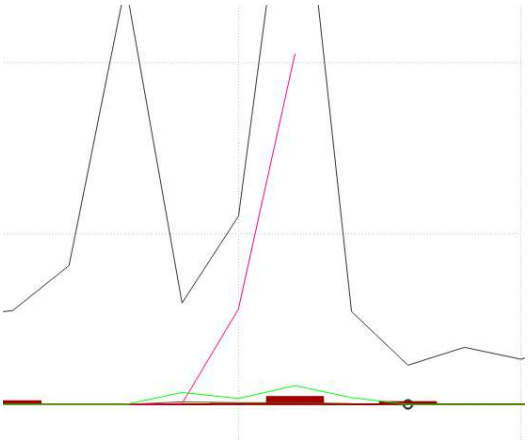


Rel Start	Duration	Bits/s A → B	Bits/s B → A	Flows
4.894252	1.5954	2411 bits/s	3470 bits/s	2
4.285945	7.7695	1643 bits/s	2720 bits/s	6

Đây là 2 kết nối TCP được tạo để kết nối với 2 server, ta có thể thấy:

- Kết nối đến Server 1 (128.119.245.12) bắt đầu lúc 4.285s và kết thúc lúc ~12.05s
- Kết nối đến Server 2 (2.56.99.24) bắt đầu lúc 4.894s và kết thúc lúc ~6.49s

→ **Thời gian chồng lấn (Overlap):** Từ 4.894s đến 6.49s (~1.6 giây), cả hai kết nối TCP đều đang hoạt động đồng thời.



Nhìn vào IO Graph phía trên:

- Đường màu xanh lá (Server 128.119.245.12): Traffic kéo dài với nhiều đỉnh
- Đường màu đỏ (Server 2.56.99.24): Traffic xuất hiện trong khoảng giữa

-> Hai đường màu giao nhau trong cùng một khoảng thời gian, chứng minh dữ liệu từ cả hai server đang được truyền đồng thời.

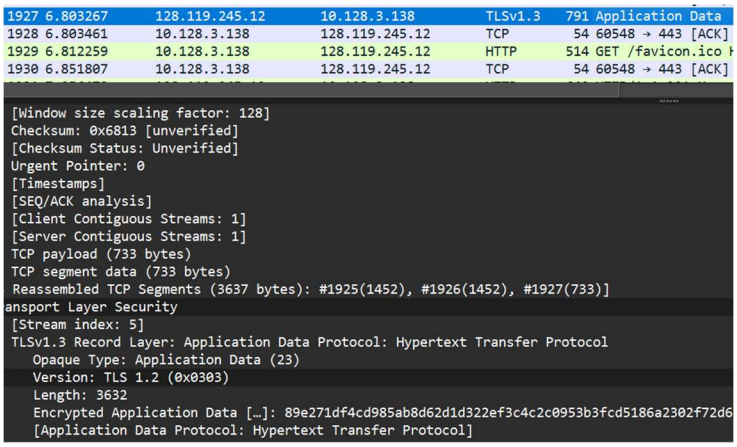
http

No.	Time	Source	Destination	Protocol	Length	Info
919	4.528615	10.128.3.138	128.119.245.12	HTTP	568	GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1.1
951	4.771869	128.119.245.12	10.128.3.138	HTTP	1362	HTTP/1.1 200 OK (text/html)
956	4.819291	10.128.3.138	128.119.245.12	HTTP	514	GET /pearson.png HTTP/1.1
992	5.062032	128.119.245.12	10.128.3.138	HTTP	640	HTTP/1.1 301 Moved Permanently (text/html)
996	5.115023	10.128.3.138	2.56.99.24	HTTP	481	GET /8E_cover_small.jpg HTTP/1.1
1810	6.489201	2.56.99.24	10.128.3.138	HTTP	692	HTTP/1.1 200 OK (JPEG JFIF image)
1929	6.812259	10.128.3.138	128.119.245.12	HTTP	514	GET /favicon.ico HTTP/1.1
1931	7.056473	128.119.245.12	10.128.3.138	HTTP	640	HTTP/1.1 301 Moved Permanently (text/html)

- Khi nhìn vào HTTP Packet List, ta thấy các gói GET request xuất hiện có vẻ tuần tự:
  - Gói 956 (4.819s): GET pearson.png → Server 128.119.245.12
  - Gói 992 (5.062s): Nhận 301 Response từ Server 128.119.245.12
  - Gói 996 (5.115s): GET 8E_cover_small.jpg → Server 2.56.99.24

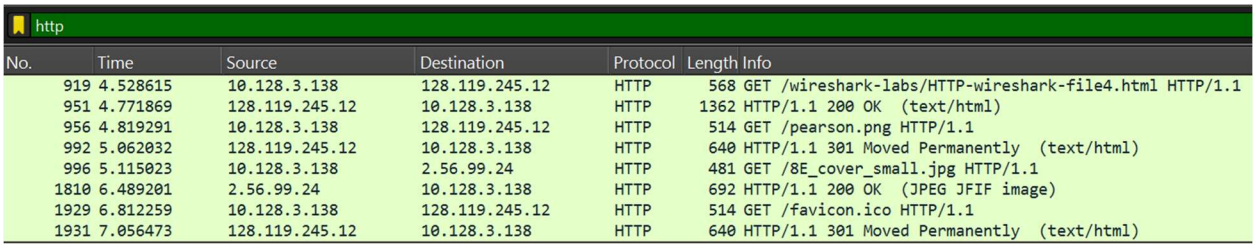
992	5.062032	128.119.245.12	10.128.3.138	HTTP	640	HTTP/1.1 301 Moved Permanently (text/html)
997	5.116853	10.128.3.138	128.119.245.12	TCP	54	58111 → 80 [ACK] Seq=975 Ack=1887 Win=132096 Len=0
1002	5.185880	10.128.3.138	128.119.245.12	TCP	66	60548 → 443 [SYN] Seq=0 Win=65340 Len=0 MSS=1460 WS=256 SACK_PERM
1050	5.438372	128.119.245.12	10.128.3.138	TCP	70	443 → 60548 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
1051	5.438496	10.128.3.138	128.119.245.12	TCP	54	60548 → 443 [ACK] Seq=1 Ack=1 Win=132096 Len=0
1052	5.439263	10.128.3.138	128.119.245.12	TLSv1.3	1781	Client Hello (SNI=gaia.cs.umass.edu)
1209	5.947763	10.128.3.138	128.119.245.12	TCP	1506	[TCP Retransmission] 60548 → 443 [PSH, ACK] Seq=276 Ack=1 Win=132096 Len=1452
1820	6.494475	128.119.245.12	10.128.3.138	TCP	70	[TCP Retransmission] 443 → 60548 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_
1821	6.494541	10.128.3.138	128.119.245.12	TCP	66	[TCP Dup ACK 1051#1] 60548 → 443 [ACK] Seq=1728 Ack=1 Win=132096 Len=0 SLE=0 SRE=1
1829	6.546367	128.119.245.12	10.128.3.138	TCP	60	443 → 60548 [ACK] Seq=1 Ack=1453 Win=67072 Len=0
1830	6.546367	128.119.245.12	10.128.3.138	TCP	60	443 → 60548 [ACK] Seq=1 Ack=1728 Win=70016 Len=0
1831	6.551850	128.119.245.12	10.128.3.138	TLSv1.3	1510	Server Hello, Change Cipher Spec, Application Data
1832	6.551850	128.119.245.12	10.128.3.138	TCP	1510	443 → 60548 [ACK] Seq=1453 Ack=1728 Win=70016 Len=1452 [TCP PDU reassembled in 1833]
1833	6.551850	128.119.245.12	10.128.3.138	TLSv1.3	704	Application Data, Application Data, Application Data
1834	6.551988	10.128.3.138	128.119.245.12	TCP	54	60548 → 443 [ACK] Seq=1728 Ack=2905 Win=132096 Len=0
1835	6.554721	10.128.3.138	128.119.245.12	TLSv1.3	134	Change Cipher Spec, Application Data
1836	6.554943	10.128.3.138	128.119.245.12	TLSv1.3	617	Application Data

Nhìn vào hình ảnh trên, filter: **ip.addr == 128.119.245.12**, ta thấy sau khi server gửi về thông báo "Moved Permanently" nó tiến hành redirect thông qua các TLS handshake như trên, trong thời gian đó hình ảnh thứ 2 "8E_cover_small" đã được lấy hoàn tất vào giây thứ 6.49s.

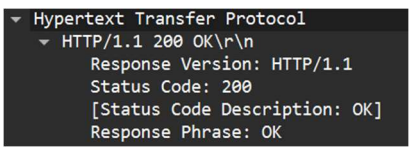


Trong khi đó, vào giây thứ 6.803s ta thấy packet 1927 có info 'Application Data' với protocol được nhận diện là 'Hypertext Transfer Protocol', kích thước 3632 bytes - đây có thể là response chứa ảnh pearson.png được gửi về qua HTTPS. Hai quá trình tải diễn ra song song, ảnh cover_small.jpg đã hoàn thành (6.49s) trong khi ảnh pearson.png mới nhận được dữ liệu lúc 6.803s ở packet 1927.

3. Locate the HTTP response message containing the content of the initial HTML page (HTTP-wireshark-file4.html). What is the **status code and status phrase** provided by the server?



Ta click vào packet trả về của server (Packet thứ 951 trên hình), vào mục Hyptertext Transfer Protocol ở dưới ta sẽ thấy được các status code, status phrase trả về.



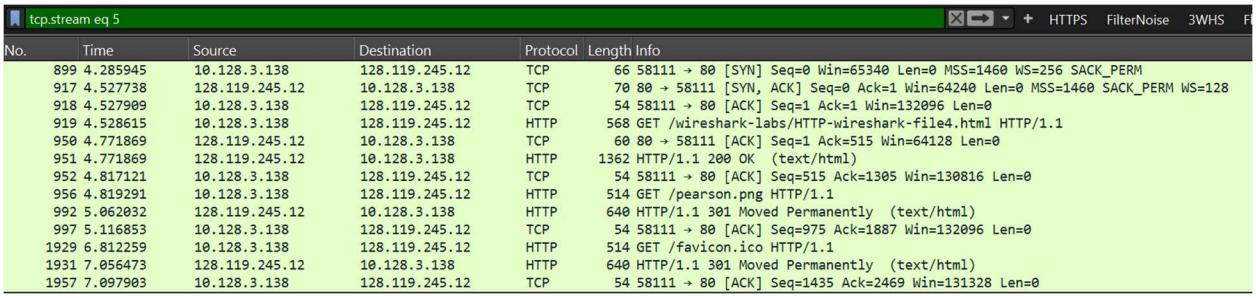
Status code: 200

Status phrase: OK

4. Based on your answer to Question 1, how many **distinct TCP connections** were established to fetch the HTML file and the two embedded images? Provide evidence by listing the unique **Stream Index Numbers** (e.g., tcp.stream eq X) that were used for these three objects.

Có tổng cộng 2 TCP connections được thiết lập để tải file HTML và hai hình ảnh nhúng.

TCP Connection thứ nhất - tcp.stream eq 5:



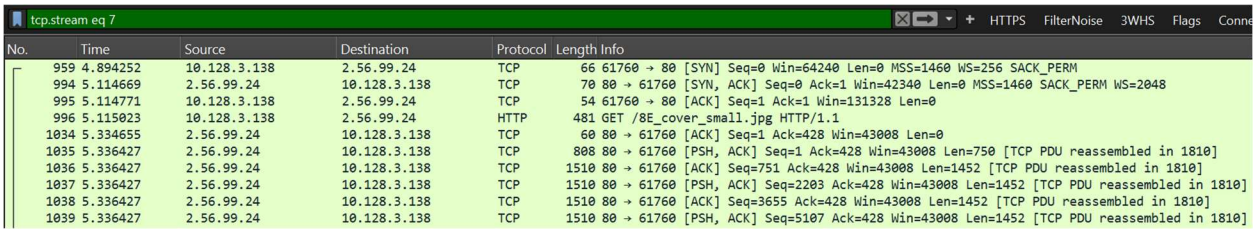
No.	Time	Source	Destination	Protocol	Length	Info
899	4.285945	10.128.3.138	128.119.245.12	TCP	66	58111 → 80 [SYN] Seq=0 Win=65340 Len=0 MSS=1460 WS=256 SACK_PERM
917	4.527738	128.119.245.12	10.128.3.138	TCP	70	80 → 58111 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
918	4.527909	10.128.3.138	128.119.245.12	TCP	54	58111 → 80 [ACK] Seq=1 Ack=1 Win=132096 Len=0
919	4.528615	10.128.3.138	128.119.245.12	HTTP	568	GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1.1
950	4.771869	128.119.245.12	10.128.3.138	TCP	60	80 → 58111 [ACK] Seq=1 Ack=515 Win=64128 Len=0
951	4.771869	128.119.245.12	10.128.3.138	HTTP	1362	HTTP/1.1 200 OK (text/html)
952	4.817121	10.128.3.138	128.119.245.12	TCP	54	58111 → 80 [ACK] Seq=515 Ack=1305 Win=130816 Len=0
956	4.819291	10.128.3.138	128.119.245.12	HTTP	514	GET /pearson.png HTTP/1.1
992	5.062032	128.119.245.12	10.128.3.138	HTTP	640	HTTP/1.1 301 Moved Permanently (text/html)
997	5.116853	10.128.3.138	128.119.245.12	TCP	54	58111 → 80 [ACK] Seq=975 Ack=1887 Win=132096 Len=0
1929	6.812259	10.128.3.138	128.119.245.12	HTTP	514	GET /favicon.ico HTTP/1.1
1931	7.056473	128.119.245.12	10.128.3.138	HTTP	640	HTTP/1.1 301 Moved Permanently (text/html)
1957	7.097903	10.128.3.138	128.119.245.12	TCP	54	58111 → 80 [ACK] Seq=1435 Ack=2469 Win=131328 Len=0

Connection này được sử dụng để tải cả file HTML chính và hình ảnh đầu tiên. Cụ thể, trong tcp.stream eq 5, chúng ta có thể thấy các packet sau:

Packet 919 chứa yêu cầu GET cho file HTTP-wireshark-file4.html được gửi đến địa chỉ IP đích 128.119.245.12. Tiếp theo, packet 951 chứa phản hồi HTTP 200 OK với nội dung của file HTML. Sau đó, packet 956 chứa yêu cầu GET cho hình ảnh pearson.png, cũng được gửi đến cùng địa chỉ IP 128.119.245.12, và packet 992 chứa phản hồi với status code 301 Moved Permanently cho hình ảnh này.

Tất cả các giao tiếp này diễn ra trên cùng một TCP connection vì chúng đều được gửi đến cùng một server có địa chỉ 128.119.245.12.

TCP Connection thứ hai - tcp.stream eq 7:



No.	Time	Source	Destination	Protocol	Length	Info
959	4.894252	10.128.3.138	2.56.99.24	TCP	66	61760 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
994	5.114669	2.56.99.24	10.128.3.138	TCP	70	80 → 61760 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1460 SACK_PERM WS=2048
995	5.114771	10.128.3.138	2.56.99.24	TCP	54	61760 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=0
996	5.115023	10.128.3.138	2.56.99.24	HTTP	481	GET /8E_cover_small.jpg HTTP/1.1
1034	5.334655	2.56.99.24	10.128.3.138	TCP	60	80 → 61760 [ACK] Seq=1 Ack=428 Win=43008 Len=0
1035	5.336427	2.56.99.24	10.128.3.138	TCP	808	80 → 61760 [PSH, ACK] Seq=1 Ack=428 Win=43008 Len=750 [TCP PDU reassembled in 1810]
1036	5.336427	2.56.99.24	10.128.3.138	TCP	1510	80 → 61760 [ACK] Seq=751 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1037	5.336427	2.56.99.24	10.128.3.138	TCP	1510	80 → 61760 [PSH, ACK] Seq=2203 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1038	5.336427	2.56.99.24	10.128.3.138	TCP	1510	80 → 61760 [ACK] Seq=3655 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1039	5.336427	2.56.99.24	10.128.3.138	TCP	1510	80 → 61760 [PSH, ACK] Seq=5107 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]

Connection này được thiết lập riêng biệt để tải hình ảnh thứ hai. Trong tcp.stream eq 7, packet 996 chứa yêu cầu GET cho file 8E_cover_small.jpg được gửi đến địa chỉ IP đích 2.56.99.24. Các packet tiếp theo trong cùng stream này chứa dữ liệu phản hồi của hình ảnh được chia thành nhiều TCP segments.

Lý do phải thiết lập một TCP connection mới là vì hình ảnh này nằm trên một web server hoàn toàn khác với địa chỉ IP 2.56.99.24. Khi trình duyệt muốn tải tài nguyên từ một server khác, nó phải mở một TCP connection mới đến server đó.

5. For the TCP connection that retrieved the *initial HTML file*, identify the three packets that form the **TCP Three-Way Handshake**. List the **TCP flags** set in each of these three packets in order

TCP Connection thứ nhất - tcp.stream eq 5:

No.	Time	Source	Destination	Protocol	Length Info
899	4.285945	10.128.3.138	128.119.245.12	TCP	66 58111 → 80 [SYN] Seq=0 Win=65340 Len=0 MSS=1460 WS=256 SACK_PERM
917	4.527738	128.119.245.12	10.128.3.138	TCP	70 80 → 58111 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
918	4.527909	10.128.3.138	128.119.245.12	TCP	54 58111 → 80 [ACK] Seq=1 Ack=1 Win=132096 Len=0

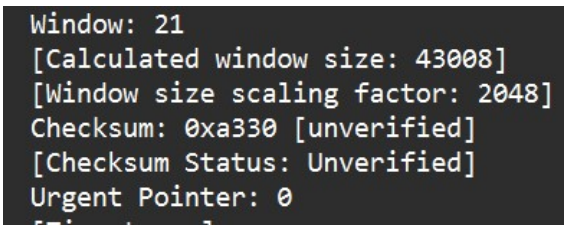
Hình ảnh trên hiển thị 3 packets thể hiện quá trình "TCP Three-Way Handshake" (bắt tay ba bước) để thiết lập kết nối đáng tin cậy giữa client và server:

1. **Packet 899**: Client gửi yêu cầu kết nối tới Server với cờ TCP là [SYN], seq = 0. (Đây là bước khởi tạo, client yêu cầu đồng bộ hóa sequence number để bắt đầu kết nối)
2. **Packet 917:** Server phản hồi lại Client với cờ TCP là [SYN, ACK], seq = 0, ack = 1. (Bước này server đồng ý thiết lập kết nối, xác nhận seq của client và gửi seq của mình, sẵn sàng giao tiếp)
3. **Packet 918**: Client xác nhận hoàn tất kết nối với cờ TCP là [ACK], seq = 1, ack = 1. (Sau packet này, TCP connection đã được thiết lập hoàn toàn và sẵn sàng truyền dữ liệu, như HTTP GET sau đó)

6. Select the largest data transfer packet (a packet with the **PSH** or **ACK** flag set and a large length) during the download of one of the image files. Examine the **TCP Window Size Value** in the packet details. What does this value represent, and why is it essential for the Transport Layer?

Trong TCP connection: tcp.stream.eq 7

1802	6.489201	2.56.99.24	10.128.3.138	TCP	1510 80 → 61760 [PSH, ACK] Seq=485719 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1803	6.489201	2.56.99.24	10.128.3.138	TCP	1510 80 → 61760 [ACK] Seq=487171 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1804	6.489201	2.56.99.24	10.128.3.138	TCP	1510 80 → 61760 [PSH, ACK] Seq=488623 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1805	6.489201	2.56.99.24	10.128.3.138	TCP	1510 80 → 61760 [ACK] Seq=490075 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1806	6.489201	2.56.99.24	10.128.3.138	TCP	1510 80 → 61760 [PSH, ACK] Seq=491527 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1807	6.489201	2.56.99.24	10.128.3.138	TCP	1510 80 → 61760 [ACK] Seq=492979 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1808	6.489201	2.56.99.24	10.128.3.138	TCP	1510 80 → 61760 [PSH, ACK] Seq=494431 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1809	6.489201	2.56.99.24	10.128.3.138	TCP	1510 80 → 61760 [ACK] Seq=495883 Ack=428 Win=43008 Len=1452 [TCP PDU reassembled in 1810]
1810	6.489201	2.56.99.24	10.128.3.138	HTTP	692 HTTP/1.1 200 OK (JPEG JFIF image)
1811	6.489429	10.128.3.138	2.56.99.24	TCP	54 61760 → 80 [ACK] Seq=428 Ack=481363 Win=525568 Len=0
1812	6.489496	10.128.3.138	2.56.99.24	TCP	54 61760 → 80 [ACK] Seq=428 Ack=484267 Win=525568 Len=0



Trong TCP connection: tcp.stream eq 7 (dùng để download file hình ảnh 8E_cover_small.jpg từ server 2.56.99.24), em chọn packet 1806 làm ví dụ cho largest data transfer packet (với flag [PSH, ACK] và length=1510 bytes)

- **Window (Raw)**: 21 (Đây là giá trị thô ghi trong header của gói tin).

- **Window size scaling factor**: 2048 (Đây là hệ số nhân đã được thỏa thuận trong quá trình bắt tay 3 bước của TCP handshake mục WS (window scale)).
- **Calculated window size**: 43008 (21 * 2048).

Giá trị 43008 là kích thước vùng đệm nhận của thiết bị gửi gói tin này (trong trường hợp này là Server 2.56.99.24).

- Thông báo cho thiết bị nhận gói tin này (Client) biết rằng thiết bị gửi (Server) hiện đang có bộ nhớ đệm (buffer) trống là 43008 bytes. Điều này cho phép Client có thể gửi liên tiếp lượng dữ liệu tối đa là 43008 bytes ngược lại cho Server mà không cần dừng lại chờ tín hiệu xác nhận (ACK) cho từng gói.

Điều này là thiết yếu đối với tầng giao vận vì:

- Ngăn chặn quá tải, đảm bảo Sender không gửi dữ liệu vượt quá khả năng xử lý của bên Receiver.
- Đảm bảo độ tin cậy, nếu gửi khi mà bộ đệm không có khả năng chứa hết → Dẫn đến việc bị "drop" dữ liệu, phải gửi lại gây ra hiện tượng tắc nghẽn mạng và lãng phí băng thông.

References

Ross, J. K. (2005-2016). *Wireshark Lab: HTTP v7.0.*