TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

Báo cáo đồ án thực hành

Phân Tích Gói Tin Mạng với Wireshark

Môn học: Mạng Máy Tính

Giáo viên hướng dẫn:
Thầy Lê Ngọc Sơn
Thầy Nguyễn Thanh Quân
Thầy Lê Hà Minh

Thành viên:
Nguyễn Hữu Gia Minh - 24127078
Nguyễn Khánh Linh - 24127197
Trần Hoàng Phúc - 24127505

Ngày 4 tháng 12 năm 2025

Mục lục

1 Giới thiệu

1.1 Mục đích dự án

Dự án này nhằm mục đích thu thập, lọc và phân tích lưu lượng mạng bằng công cụ Wireshark, đồng thời giúp sinh viên hiểu rõ các khái niệm cơ bản về các giao thức mạng khác nhau như HTTP, DHCP, DNS, TCP/IP và Ethernet.

1.2 Giới thiệu công cụ Wireshark

Wireshark là một công cụ phân tích gói tin mạng mã nguồn mở, cho phép bắt và kiểm tra dữ liệu truyền qua mạng ở mức độ chi tiết. Wireshark hỗ trợ hàng trăm giao thức mạng và cung cấp giao diện trực quan để phân tích lưu lượng mạng theo thời gian thực.

1.3 Thông tin nhóm thực hiện

- **Tên học phần:** Mạng máy tính
- **Đồ án thực hiện:** Phân tích gói tin mạng với Wireshark
- **Thời gian thực hiện:** 4/12/2025 đến 11/12/2025

Danh sách thành viên:

STT	MSSV	Họ và tên	Vai trò
01	24127078	Nguyễn Hữu Gia Minh	Thành viên
02	24127197	Nguyễn Khánh Linh	Thành viên
03	24127505	Trần Hoàng Phúc	Nhóm trưởng

2 Phần 1: Phân Tích Lưu Lượng HTTP

2.1 Mô tả các bước thực hiện

- **Bước 1:** Khởi động trình duyệt web và xóa cache của trình duyệt (clear browser cache).
- **Bước 2:** Khởi động Wireshark và bắt đầu capture packets trên network interface đang hoạt động.
- **Bước 3:** Truy cập URL: `http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file4.html` trong trình duyệt.
- **Bước 4:** Đợi trang web tải hoàn tất, sau đó dừng Wireshark capture.
- **Bước 5:** Áp dụng display filter `http` trong Wireshark để chỉ hiển thị các HTTP packets.

2.2 Câu hỏi phân tích và trả lời

2.2.1 Câu 1: Số lượng HTTP GET và địa chỉ đích

Câu hỏi: Có bao nhiêu tin nhắn HTTP GET được trình duyệt gửi đi? Các yêu cầu này được gửi đến địa chỉ Internet nào?

Trả lời:

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)

Hình 1: Các yêu cầu HTTP GET được gửi từ trình duyệt

Có tổng cộng 4 yêu cầu HTTP GET được trình duyệt gửi đi:

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

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

- `128.119.245.12` - cho file HTML, `pearson.png` và `favicon.ico`
- `2.56.99.24` - cho `8E_cover_small.jpg`

`10.128.3.138`: địa chỉ IP nguồn của laptop.

2.2.2 Câu 2: Tải ảnh tuần tự hay đồng thời

Câu hỏi: Xác định xem trình duyệt tải hai hình ảnh tuần tự hay đồng thời từ các web server tương ứng, và giải thích kết luận của bạn bằng cách kiểm tra thời gian của các yêu cầu và địa chỉ IP nguồn.

Trả lời:

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

Ethernet · 1 | IPv4 · 2 | IPv6 | TCP · 2 | UDP

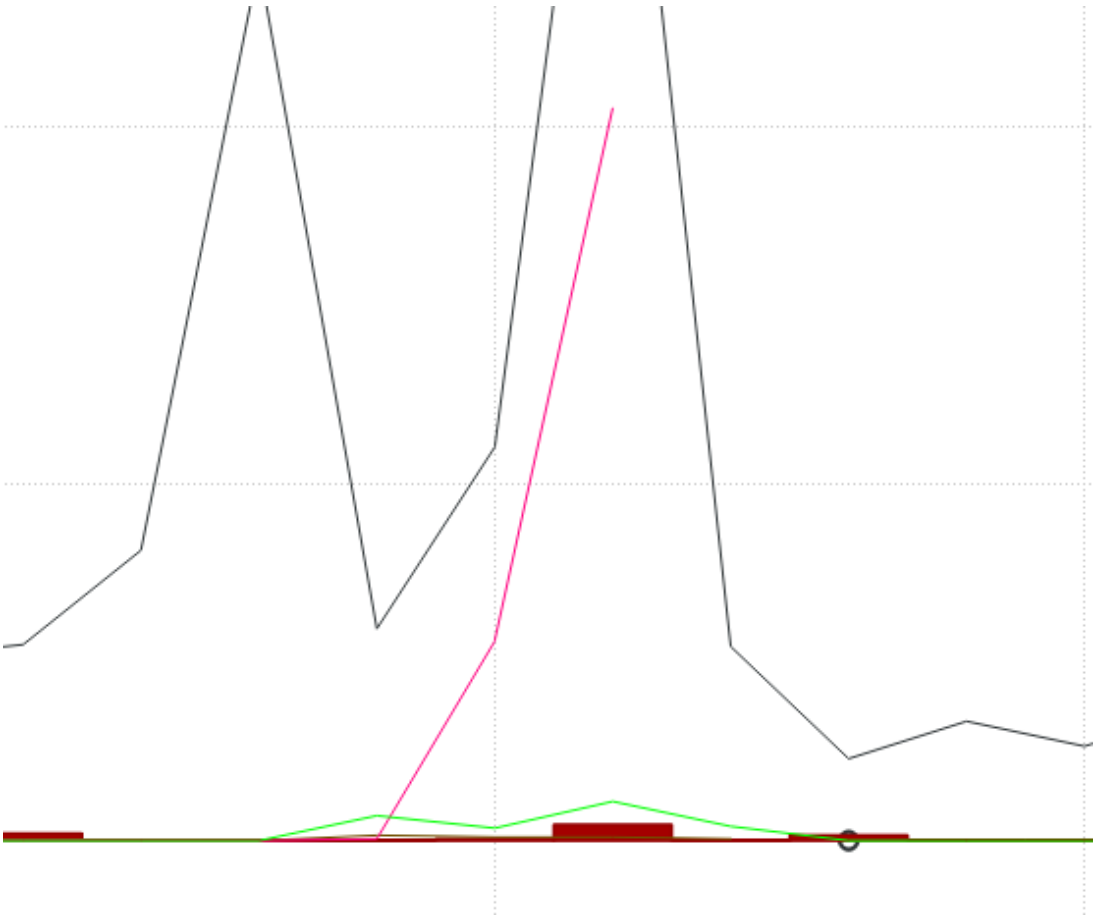
Address A	Port A	Address B	Port B	Packets	Stream ID	Total Packets	Rel Start	Duration	Flows
10.128.3.138	58111	128.119.245.12	80	6	5	15	4.285945	7.7695	6
10.128.3.138	61760	2.56.99.24	80	2	7	521	4.894252	1.5954	2

Hình 2: Timeline của 2 kết nối TCP đến 2 server (statistics -> conversations -> TCP)

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.



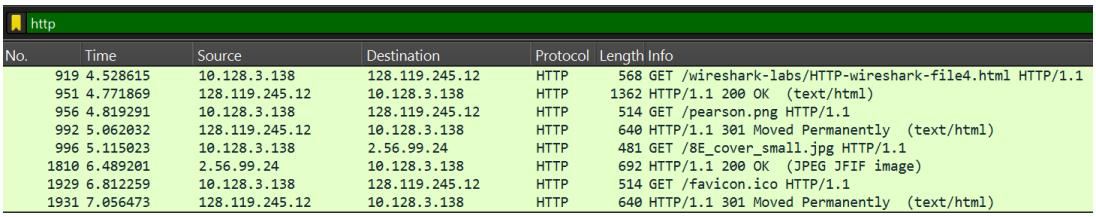
Hình 3: IO Graph thể hiện traffic từ 2 server đồng thời

Nhìn vào IO Graph phía trên (mục statistics → IO Graphs), ta thấy:

- 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.



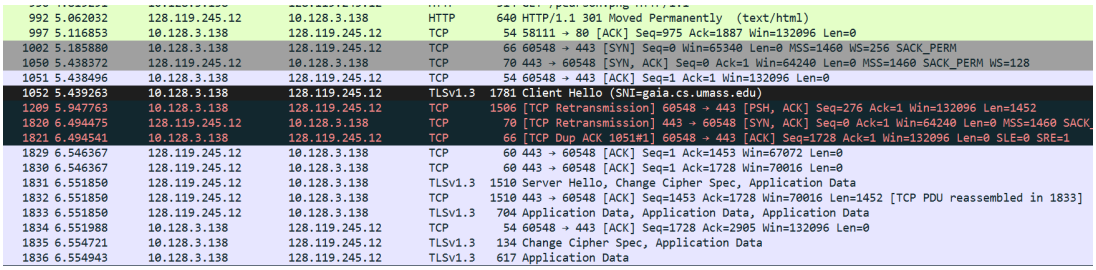
http

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)

Hình 4: HTTP Packet List - Thứ tự các gói GET request

Khi nhìn vào HTTP Packet List, ta thấy các gói GET request xuất hiện có vẻ tuần tự:

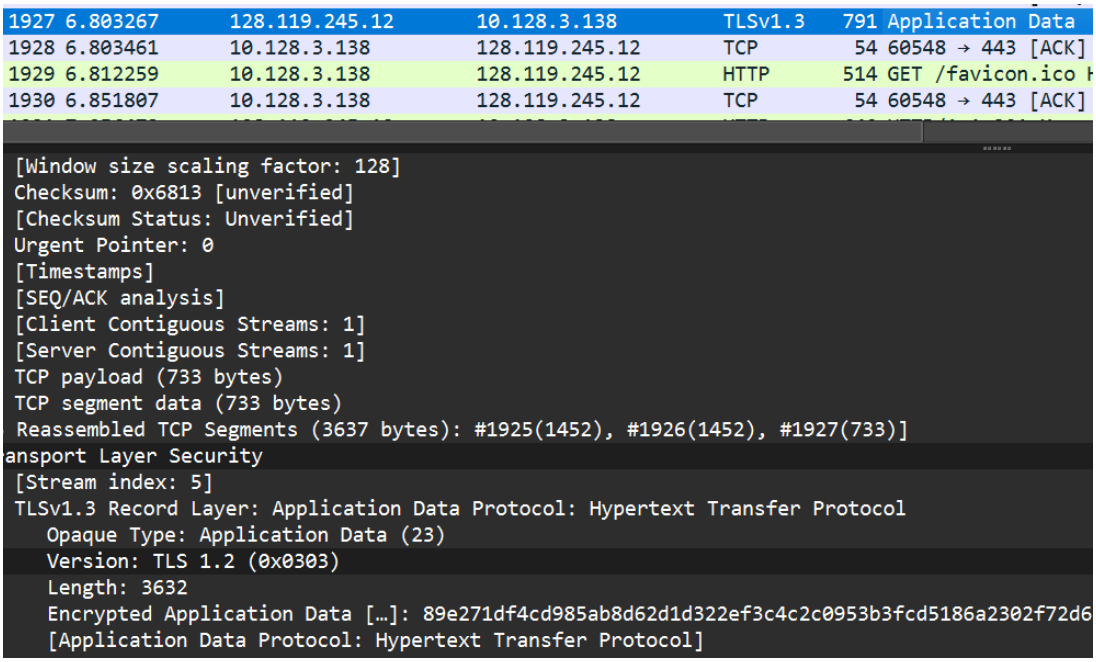
- Gói 956 (4.819s): `GET pearson.png` → Server `128.119.245.12`
- Gói 992 (5.062s): Nhận 301 Response từ Server `128.119.245.12`
- Gói 996 (5.115s): `GET 8E_cover_small.jpg` → Server `2.56.99.24`
- Gói 1929 (6.812s): `GET favicon.ico` → Server `2.56.99.24`



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

Hình 5: Filter ip.addr == 128.119.245.12 - TLS handshake và redirect

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.



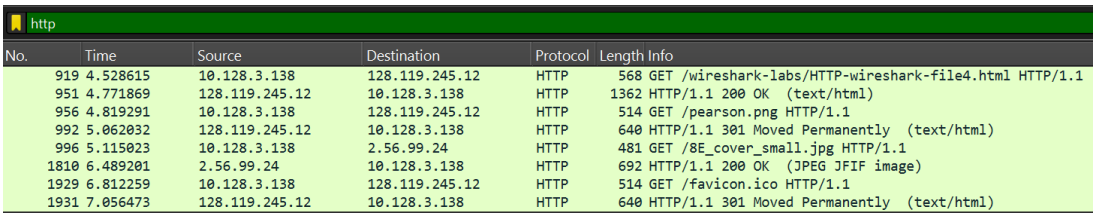
Hình 6: Application Data packet chứa ảnh pearson.png

Trong khi đó, tại giây 6.803s, packet 1927 có info 'Application Data' với kích thước TLS Record là 3632 bytes – đây là **gói tin cuối cùng** chứa dữ liệu ảnh `pearson.png` được gửi về qua HTTPS (dưới dạng mã hóa TLS). Hai quá trình tải diễn ra song song: ảnh `cover_small.jpg` đã hoàn thành (6.49s) trong khi `pearson.png` nhận đủ dữ liệu lúc 6.803s ở packet 1927.

2.2.3 Câu 3: Thông tin phản hồi HTTP của tệp HTML

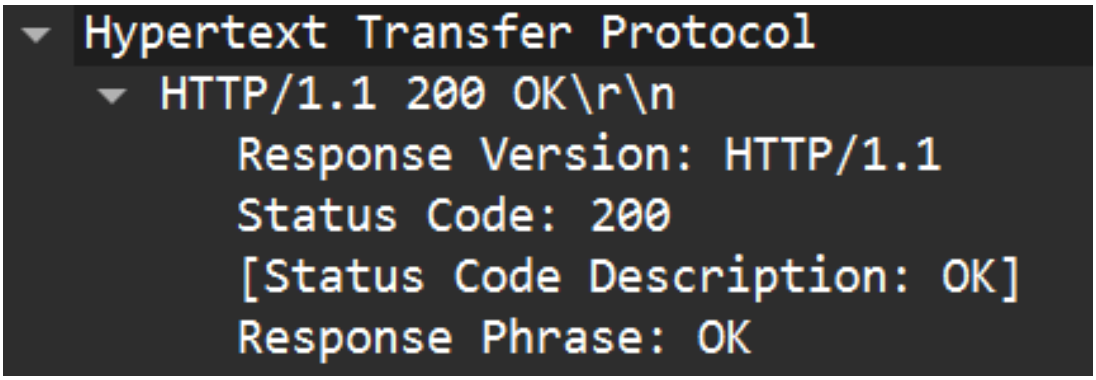
Câu hỏi: Tìm tin nhắn phản hồi HTTP chứa nội dung của trang HTML ban đầu (HTTP-wireshark-file4.html). Status code và status phrase mà server cung cấp là gì?

Trả lời:



Hình 7: Packet trả về của server chứa nội dung HTML

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



Hình 8: Chi tiết HTTP response header

- **Status code:** 200
- **Status phrase:** OK

Mã trạng thái **200 OK** là mã phản hồi HTTP phổ biến nhất.

- Nó biểu thị rằng **yêu cầu (request)** của máy khách (client) đã được **thành công** xử lý.
- Đồng thời, máy chủ (server) đã **cung cấp dữ liệu** được yêu cầu (ví dụ: một trang web, dữ liệu JSON, hình ảnh) trong phản hồi.

2.2.4 Câu 4: Số lượng kết nối TCP và Stream Index Numbers

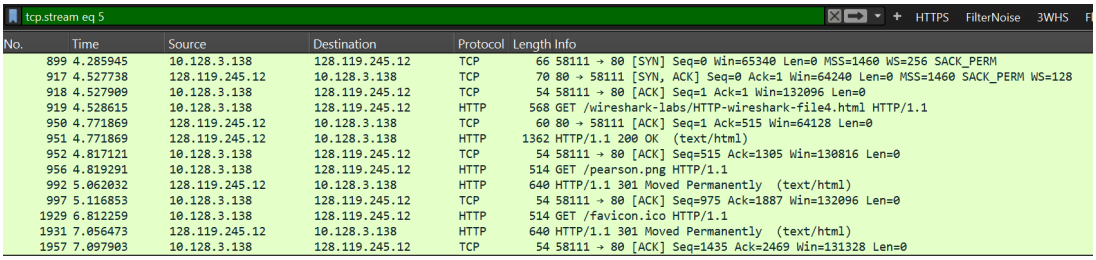
Câu hỏi: Dựa trên câu trả lời ở Câu 1, có bao nhiêu kết nối TCP riêng biệt được thiết lập để tải tệp HTML và hai hình ảnh nhúng? Cung cấp bằng chứng bằng cách liệt kê các Stream Index Numbers duy nhất (ví dụ: `tcp.stream eq X`) được sử dụng cho ba đối tượng này.

Trả lời:

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

Nhấp chuột phải vào một packet của từng destination khác nhau, chọn `Follow` → `TCP Stream` để xem các TCP stream tương ứng.

- **TCP Connection thứ nhất -** `tcp.stream eq 5`**:**



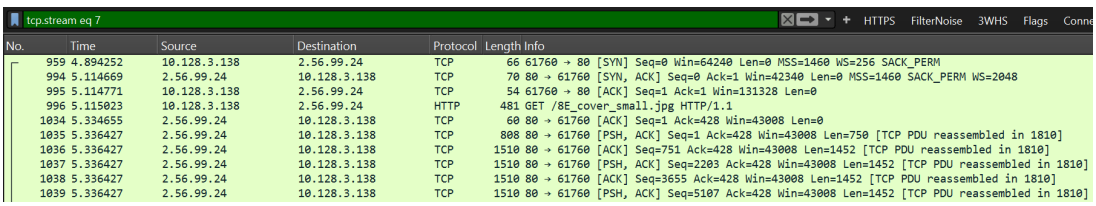
Hình 9: TCP stream 5 - Kết nối đến server 128.119.245.12

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`.
- Packet 951 chứa phản hồi HTTP 200 OK với nội dung của file HTML.
- Packet 956 chứa yêu cầu GET cho hình ảnh `pearson.png`, cũng được gửi đến cùng địa chỉ IP `128.119.245.12`.
- 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`**:**



Hình 10: TCP stream 7 - Kết nối đến server 2.56.99.24

Connection này được thiết lập riêng biệt để tải hình ảnh thứ hai. Trong `tcp.stream eq 7`, chúng ta có thể thấy:

- 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.

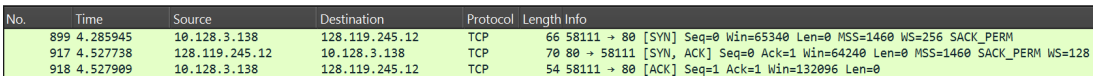
Lý do phải thiết lập một TCP connection mới là vì hình ảnh này nằm trên một web server hoàn toàn khác với địa chỉ IP `2.56.99.24`. Khi trình duyệt muốn tải tài nguyên từ một server khác, nó phải mở một TCP connection mới đến server đó.

2.2.5 Câu 5: TCP Three-Way Handshake

Câu hỏi: Đối với kết nối TCP đã tải tệp HTML ban đầu, xác định ba gói tin tạo thành quá trình Bắt tay ba chiều TCP (TCP Three-Way Handshake). Liệt kê các cờ TCP được đặt trong mỗi gói tin này theo thứ tự.

Trả lời:

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 11: TCP Three-Way Handshake - 3 packets thiết lập kết nối

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)

2.2.6 Câu 6: TCP Window Size Value

Câu hỏi: Chọn gói truyền dữ liệu lớn nhất (một gói có cờ PSH hoặc ACK được đặt và có độ dài lớn) trong quá trình tải một trong các tệp hình ảnh. Kiểm tra giá trị TCP Window Size trong chi tiết gói tin. Giá trị này đại diện cho điều gì, và tại sao nó lại quan trọng đối với Lớp Giao vận (Transport Layer)?

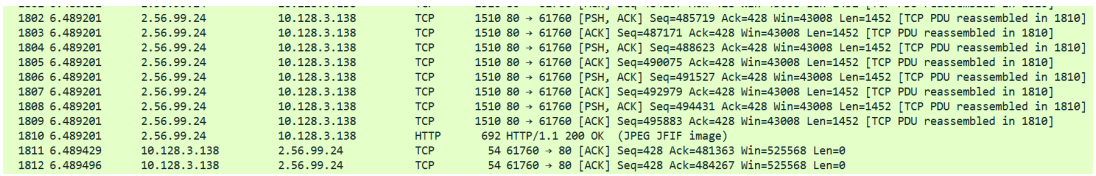
Trả lời:

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

Cờ PSH (Push Flag)

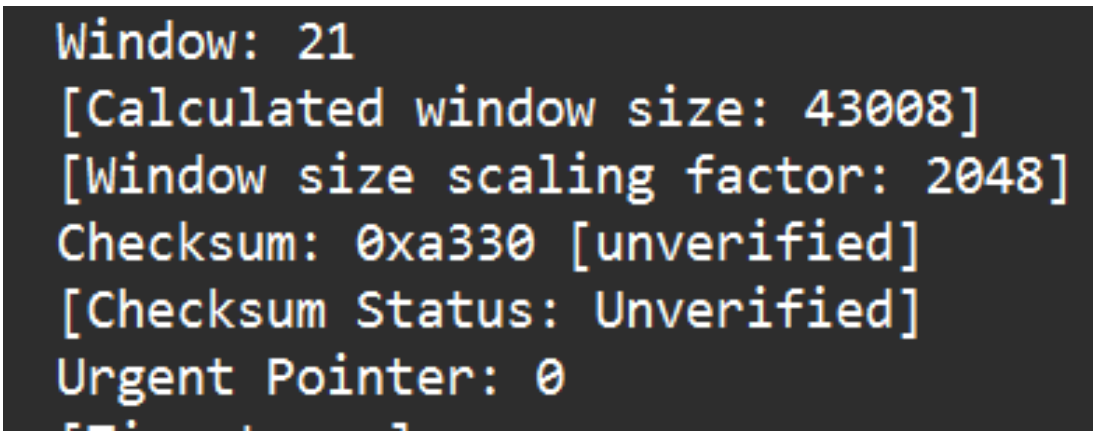
- **Chức năng:** Trong gói tin này, Server bật cờ PSH để yêu cầu Client chuyển ngay lập tức dữ liệu nhận được lên Lớp Ứng dụng (Trình duyệt) thay vì lưu trữ chờ đầy bộ đệm (TCP Buffer).

- **Mục đích:** Giúp trình duyệt nhận dữ liệu ảnh tức thì để giải mã và hiển thị cho người dùng, loại bỏ độ trễ (latency) không cần thiết.



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

Hình 12: Packet 1806 - Largest data transfer packet



Hình 13: Chi tiết TCP Window Size trong packet 1806

Phân tích TCP Window Size:

- **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`).
- **Ý nghĩa:** 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.

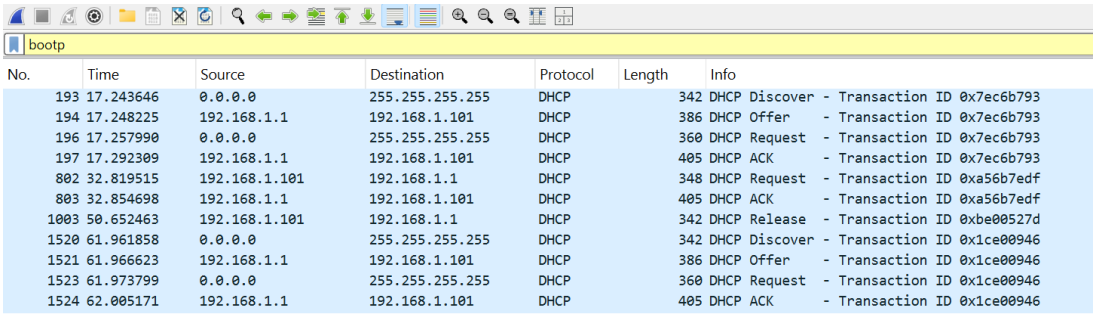
Tại sao TCP Window Size quan trọng đối với Lớp Giao vận:

- **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 $\rightarrow$ Dẫn đến việc bị "drop"dữ liệu, phải gửi lại gây ra hiện tượng tắc nghẽn mạng và lãng phí băng thông.

3 Phần 2: Phân Tích Lưu Lượng DHCP

3.1 Mô tả các bước thực hiện

- **Bước 1: Chuẩn bị và giải phóng địa chỉ IP ban đầu**
  - Mở ứng dụng Windows Command Prompt (CMD).
  - Nhập lệnh `ipconfig /release`.
- **Bước 2: Bắt đầu ghi lại dữ liệu**
  - Khởi động Wireshark.
  - Bắt đầu ghi lại dữ liệu gói tin (packet capture).
- **Bước 3: Chu kỳ cấp phát IP lần 1**
  - Quay lại Command Prompt, nhập lệnh `ipconfig /renew`.
- **Bước 4: Chu kỳ cấp phát IP lần 2**
  - Sau khi lệnh trước kết thúc, nhập lại lệnh `ipconfig /renew`.
- **Bước 5: Giải phóng địa chỉ IP**
  - Khi lệnh `ipconfig /renew` lần thứ hai hoàn tất, nhập lệnh `ipconfig /release`.
- **Bước 6: Chu kỳ cấp phát IP lần cuối**
  - Cuối cùng, nhập lệnh `ipconfig /renew` lần nữa.
- **Bước 7: Kết thúc ghi dữ liệu**
  - Quay lại Wireshark và dừng quá trình ghi lại gói tin.



bootp

No.	Time	Source	Destination	Protocol	Length	Info
193	17.243646	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x7ec6b793
194	17.248225	192.168.1.1	192.168.1.101	DHCP	386	DHCP Offer - Transaction ID 0x7ec6b793
196	17.257990	0.0.0.0	255.255.255.255	DHCP	360	DHCP Request - Transaction ID 0x7ec6b793
197	17.292309	192.168.1.1	192.168.1.101	DHCP	405	DHCP ACK - Transaction ID 0x7ec6b793
802	32.819515	192.168.1.101	192.168.1.1	DHCP	348	DHCP Request - Transaction ID 0xa56b7edf
803	32.854698	192.168.1.1	192.168.1.101	DHCP	405	DHCP ACK - Transaction ID 0xa56b7edf
1003	50.652463	192.168.1.101	192.168.1.1	DHCP	342	DHCP Release - Transaction ID 0xbe00527d
1520	61.961858	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x1ce00946
1521	61.966623	192.168.1.1	192.168.1.101	DHCP	386	DHCP Offer - Transaction ID 0x1ce00946
1523	61.973799	0.0.0.0	255.255.255.255	DHCP	360	DHCP Request - Transaction ID 0x1ce00946
1524	62.005171	192.168.1.1	192.168.1.101	DHCP	405	DHCP ACK - Transaction ID 0x1ce00946

Hình 14: Kết quả capture gói tin DHCP

3.2 Câu hỏi phân tích và trả lời

3.2.1 Câu 1: Gói tin ARP trong quá trình trao đổi DHCP

Câu hỏi: Các gói tin ARP có xuất hiện trong giai đoạn trao đổi DHCP không? Nếu có, mục đích của chúng trong quá trình này là gì?

Trả lời:

Dựa vào các bản ghi phân tích gói tin hiển thị, có thể thấy rằng trong suốt 2 chuỗi giao tiếp DORA của DHCP, không có bất kỳ gói tin ARP nào được trao đổi.

No.	Time	Source	Destination	Protocol	Length	Info
188	15.344182	CigShanghai_24:f0:b8	Broadcast	ARP	42	Who has 192.168.1.101? Tell 192.168.1.1
189	16.368233	CigShanghai_24:f0:b8	Broadcast	ARP	42	Who has 192.168.1.101? Tell 192.168.1.1
193	17.243646	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x7ec6b793
194	17.248225	192.168.1.1	192.168.1.101	DHCP	386	DHCP Offer - Transaction ID 0x7ec6b793
196	17.257990	0.0.0.0	255.255.255.255	DHCP	360	DHCP Request - Transaction ID 0x7ec6b793
197	17.292309	192.168.1.1	192.168.1.101	DHCP	405	DHCP ACK - Transaction ID 0x7ec6b793
200	17.325990	Intel_9c:ab:03	Broadcast	ARP	42	Who has 192.168.1.1? Tell 192.168.1.101
201	17.328343	CigShanghai_24:f0:b8	Intel_9c:ab:03	ARP	46	192.168.1.1 is at 94:f7:17:24:f0:b8
206	17.343764	Intel_9c:ab:03	Broadcast	ARP	42	Who has 192.168.1.1? Tell 192.168.1.101
1481	60.704513	Intel_9c:ab:03	Broadcast	ARP	42	ARP Announcement for 169.254.20.22
1520	61.961858	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x1ce00946
1521	61.966623	192.168.1.1	192.168.1.101	DHCP	386	DHCP Offer - Transaction ID 0x1ce00946
1523	61.973799	0.0.0.0	255.255.255.255	DHCP	360	DHCP Request - Transaction ID 0x1ce00946
1524	62.005171	192.168.1.1	192.168.1.101	DHCP	405	DHCP ACK - Transaction ID 0x1ce00946
1558	62.703287	Intel_9c:ab:03	Broadcast	ARP	42	ARP Announcement for 169.254.20.22
1560	62.858509	CigShanghai_24:f0:b8	Broadcast	ARP	42	Who has 169.254.20.22? Tell 192.168.1.1
1561	62.858551	Intel_9c:ab:03	CigShanghai_24:f0:b8	ARP	42	169.254.20.22 is at 48:a4:72:9c:ab:03
1578	63.857449	Intel 9c:ab:03	Broadcast	ARP	42	Who has 192.168.1.1? Tell 192.168.1.101

Hình 15: Không có gói tin ARP nào xuất hiện trong cả 2 chu kì DORA của DHCP

- **Kết luận:** Trong khoảng thời gian của 2 chu kì DORA của DHCP trên, không có gói tin ARP nào được trao đổi.

Giải thích nguyên nhân:

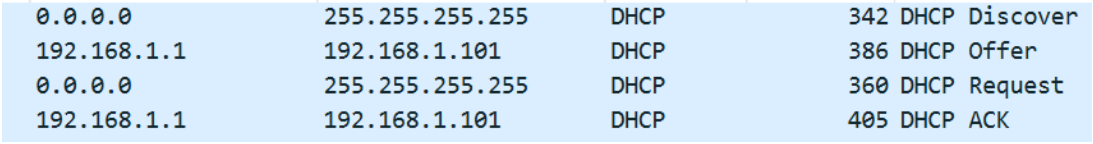
- **Vai trò và cấp độ hoạt động khác nhau của hai giao thức:**
  - DHCP (DORA) hoạt động chủ yếu ở lớp Ứng dụng (Application Layer) và phục vụ mục đích cấp phát địa chỉ IP.
  - ARP hoạt động ở lớp Liên kết dữ liệu (Data Link Layer) và chỉ phục vụ mục đích phân giải địa chỉ MAC từ một địa chỉ IP đã biết.
- **Tuy nhiên có các gói ARP được ghi nhận xuất hiện ngay sau gói DHCP ACK:**
  - **Phân giải địa chỉ MAC của Gateway:** Sau khi nhận IP từ DHCP, máy dùng ARP để tìm MAC của Gateway nhằm gửi dữ liệu ra ngoài mạng LAN.
  - **Kiểm tra xung đột IP (ARP Probe):** Máy gửi ARP Probe đến chính IP vừa được cấp để kiểm tra trùng IP. Nếu có phản hồi → xảy ra xung đột → hủy IP và xin lại DHCP.
  - **Định danh địa chỉ mới (ARP Announcement):** Máy gửi ARP Announcement để thông báo IP–MAC của mình cho toàn mạng, giúp các thiết bị khác cập nhật bảng ARP nhanh hơn.

3.2.2 Câu 2: Địa chỉ IP nguồn và đích trong DHCP messages

Câu hỏi: Địa chỉ IP nguồn và đích được sử dụng trong các Datagram IP mang bốn tin nhắn DHCP (DHCP Discover, DHCP Offer, DHCP Request, DHCP ACK) là gì?

Trả lời:

Quá trình DHCP sử dụng mô hình trao đổi bốn bước được gọi là DORA (Discover, Offer, Request, ACK). Trước khi hoàn tất bước ACK, máy vẫn chưa có IP hợp lệ, nên các gói DHCP phải được đóng gói trong các IP đặc biệt để có thể truyền đi dù chưa có địa chỉ IP chính thức.



Hình 16: Các gói tin DHCP messages

Bảng phân tích địa chỉ IP trong từng thông điệp:

Thông điệp DHCP	IP Nguồn	IP Đích	Vai trò
Discover (D)	0.0.0.0	255.255.255.255	Máy chưa có IP, broadcast tìm kiếm server
Offer (O)	192.168.1.1	192.168.1.101	Server đề xuất IP, gửi unicast đến IP được đề xuất
Request (R)	0.0.0.0	255.255.255.255	Máy khách vẫn chưa có IP, broadcast thông báo chấp nhận
ACK (A)	192.168.1.1	192.168.1.101	Server xác nhận, gửi unicast đến IP đã cấp phát

Bảng 1: Địa chỉ IP trong các thông điệp DHCP

Phân tích các địa chỉ IP được sử dụng trong các bước Discover và Request:

Địa chỉ	Mục đích sử dụng
IP Nguồn: 0.0.0.0	Discover và Request được gửi khi máy chưa có IP, nên dùng địa chỉ 0.0.0.0 - địa chỉ "phi cấu hình" cho phép Host gửi gói tin dù IP Stack chưa khởi tạo đầy đủ.
IP Đích: 255.255.255.255	Đảm bảo gói tin Broadcast đến được tất cả các thiết bị trong mạng, đặc biệt là DHCP Server. Điều này cần thiết trong cả Discover và Request.

Bảng 2: Phân tích địa chỉ IP trong Discover và Request

Sự khác biệt rõ ràng nhất nằm ở cách DHCP Server phản hồi lại trong các bước Offer và ACK (dựa trên capture của nhóm):

Gói tin	IP đích trong lý thuyết	IP đích trong dữ liệu của nhóm
Offer và ACK	255.255.255.255 (Broadcast)	192.168.1.101 (Unicast)

Bảng 3: So sánh IP đích lý thuyết và thực tế

Phân tích sự khác biệt:

- **Hiệu quả (Efficiency):** Server gửi Offer và ACK trực tiếp đến IP mà nó cấp để giảm lưu lượng Broadcast.
- **Khẳng định MAC:** Server biết MAC của Client (từ gói Discover), nên có thể gửi gói tin trực tiếp dù IP chưa được Client xác nhận.
- **Tính tạm thời của IP:** Địa chỉ 192.168.1.101 được dùng ngay trong Offer, thể hiện rằng IP đã có giá trị logic để định danh Host trong suốt quá trình DHCP.

3.2.3 Câu 3: Xử lý khi nhận ARP Reply sau DHCP ACK

Câu hỏi: Theo RFC 2131, nếu client nhận được ARP Reply từ một thiết bị khác sau khi nhận DHCP ACK, client phải làm gì?

Trả lời:

Phát hiện xung đột thông qua ARP Probe:

Sau khi nhận DHCP ACK và coi IP là hợp lệ, Host vẫn phải thực hiện bước kiểm tra cuối cùng để tránh xung đột địa chỉ IP.

- **Cơ chế:** Host gửi một hoặc nhiều gói tin ARP Probe cho chính địa chỉ mới được cấp. Khi Host nhận được gói ARP Reply từ một thiết bị khác trên mạng, điều đó chứng tỏ địa chỉ IP đó đã được sử dụng.

```
ARP          42 Who has 169.254.20.22? (ARP Probe)
ARP          42 Who has 169.254.20.22? (ARP Probe)
ARP          42 Who has 169.254.20.22? (ARP Probe)
```

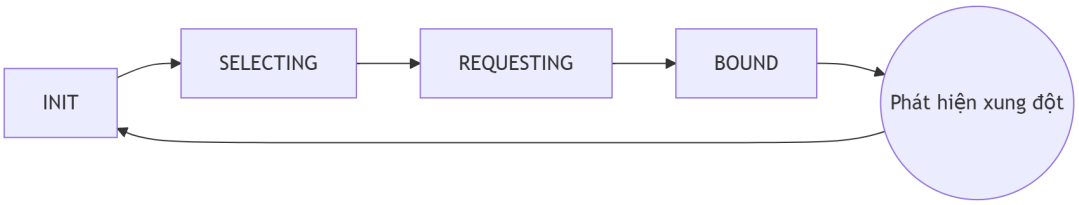
Hình 17: Các gói ARP Probe

Hành động bắt buộc theo tiêu chuẩn RFC 2131:

- **Gửi thông điệp DHCPDECLINE:** Host Client phải ngay lập tức gửi một gói tin DHCPDECLINE đến DHCP Server.
- **Mục đích của DHCPDECLINE:** Thông báo cho DHCP server biết rằng địa chỉ IP được cấp phát không thể sử dụng do xung đột, yêu cầu server đánh dấu địa chỉ này là không khả dụng trong thời gian nhất định.

Sau khi gửi DHCPDECLINE, Host Client không được phép sử dụng địa chỉ IP gây xung đột đó và phải chuyển sang trạng thái mới để xin cấp lại địa chỉ:

- **Từ bỏ IP:** Host phải lập tức hủy bỏ việc sử dụng địa chỉ IP xung đột đó.
- **Chuyển trạng thái:** Host chuyển về trạng thái INITIALIZING (Khởi tạo).
- **Bắt đầu lại:** Host sẽ khởi động lại toàn bộ quá trình xin cấp IP bằng cách gửi một gói DHCP Discover mới.



Hình 18: Quy trình xử lý xung đột IP với DHCPDECLINE

3.2.4 Câu 4: Lý do sử dụng UDP cố định và tính phi kết nối

Câu hỏi: Giải thích lý do tại sao DHCP sử dụng các cổng UDP cố định (68 cho client, 67 cho server) và vai trò của tính chất phi kết nối (connectionless) của UDP trong giao thức DHCP.

Trả lời:

Việc sử dụng các cổng cố định (Port 68 và 67) trong DHCP là bắt buộc và phục vụ 2 mục đích chính:

- **Tiêu chuẩn hóa và nhận diện ứng dụng:**
  - **Port đích (Destination Port 67):** Đây là cổng được chỉ định cho DHCP Server. Mọi Host Client đều biết rằng để nói chuyện với DHCP Server, nó phải gửi gói tin đến Port 67.
  - **Port nguồn (Source Port 68):** Đây là cổng được chỉ định cho DHCP Client. Việc sử dụng cổng cố định này là một phần của tiêu chuẩn DHCP, giúp Server nhận diện ngay lập tức rằng gói tin này đến từ một ứng dụng Client DHCP.
- **Hoạt động trong điều kiện chưa có IP:**
  - DHCP Request là một phần của quá trình DORA. Trong bước Request, Host Client vẫn đang sử dụng Source IP là 0.0.0.0 (vì chưa có IP chính thức).
  - Trong tình huống mà thông tin Lớp Mạng không đầy đủ, việc sử dụng cổng cố định Port 68 đảm bảo rằng khi Server phản hồi lại bằng gói DHCP ACK, nó biết chính xác Port nào (Port 68) mà ứng dụng DHCP Client đang lắng nghe trên Host.

Vai trò của tính chất phi kết nối Của UDP:

UDP là giao thức phi kết nối (connectionless) và không trạng thái (stateless). Chính tính chất này giúp việc sử dụng các cổng cố định trở nên đơn giản và mạnh mẽ trong DHCP.

Đặc điểm của UDP	**Ứng dụng trong DHCP**	**Lợi ích**
Không bắt tay 3 bước	Client gửi Discover ngay lập tức mà không cần thiết lập kết nối trước	Giảm overhead mạng, tăng tốc độ khởi động (bootstrap) cho Client
Không trạng thái (Stateless)	Mỗi message DHCP độc lập (Discover, Offer, Request, ACK)	Server không cần lưu trữ trạng thái kết nối (connection state) hoặc phiên làm việc (session state), giúp tăng khả năng mở rộng
Không có sequence number	Không cần theo dõi thứ tự của các gói tin gửi/nhận	Đơn giản hóa việc triển khai cả ở Client và Server; giảm thiểu xử lý ở tầng giao vận
Không có acknowledgment (ACK)	DHCP sử dụng Application-level ACK	Linh hoạt trong cơ chế thử lại (retry mechanism) và quản lý thời gian chờ (timeout)

Bảng 4: Tính chất phi kết nối của UDP trong DHCP

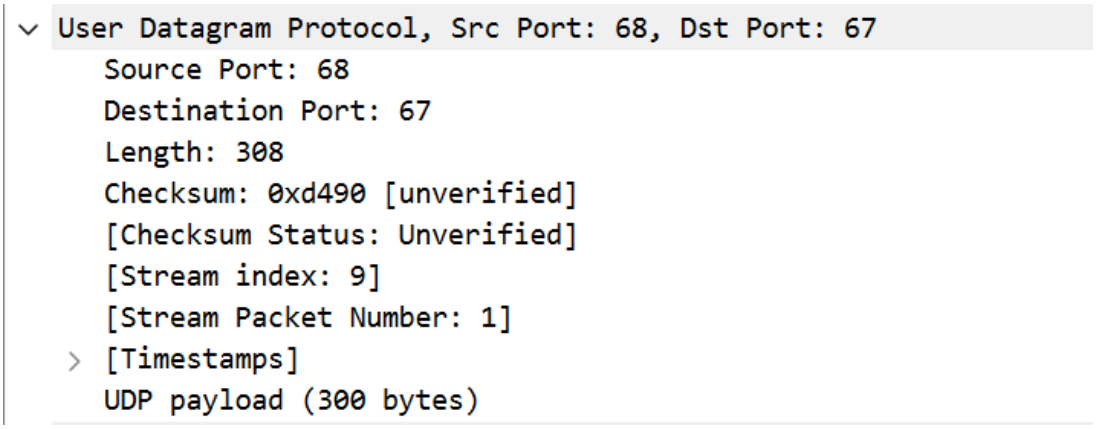
3.2.5 Câu 5: Checksum UDP và xử lý lỗi

Câu hỏi: Kiểm tra trường Checksum UDP trong gói DHCP Discover. Nếu Checksum không chính xác, bộ xử lý tầng Giao vận (UDP handler) sẽ thực hiện hành động gì và tại sao điều này lại khiến quá trình DHCP bị treo?

Trả lời:

a. Trạng thái Checksum trong gói DHCP Discover:

Trong gói tin DHCP Discover, khi mở rộng lớp User Datagram Protocol (UDP), trường Checksum hiển thị như sau:



Hình 19: Trường Checksum UDP trong gói DHCP Discover

- **Giá trị Checksum:** 0xd490
- **Trạng thái:** [unverified]
- **Checksum Status:** Unverified

Kết luận: Wireshark không thể xác minh được tính toàn vẹn của Checksum, do đó không hiển thị trạng thái "good"hay "incorrect".

Nguyên nhân kỹ thuật (Checksum Offloading):

- **Checksum Offloading** là kỹ thuật tối ưu hóa hiệu suất, trong đó việc tính toán checksum được chuyển giao (offload) từ CPU sang phần cứng của card mạng (NIC).
- **Hậu quả khi Capture:** Wireshark, chạy ở Lớp Liên kết dữ liệu hoặc ngay trên Lớp Mạng bắt giữ gói tin trước khi card mạng NIC kịp hoàn thành việc tính toán Checksum. Do đó, giá trị Checksum Wireshark ghi lại có thể là giá trị tạm thời, không chính xác, hoặc là giá trị unverified (0xd490), khiến Wireshark không thể xác minh tính toàn vẹn của gói tin.

b. Hành động của Transport Layer khi Checksum không chính xác:

Nếu Checksum của gói tin DHCP Discover được tính toán là không chính xác (incorrect), UDP handler của thiết bị nhận sẽ drop gói ngay lập tức, không chuyển lên DHCP Server → quá trình DHCP bị treo vì server không bao giờ nhận được gói.

Lý do:

- **Tính toàn vẹn dữ liệu:** Checksum dùng để kiểm tra tính toàn vẹn dữ liệu; nếu checksum sai, nghĩa là gói tin đã bị thay đổi hoặc hỏng trong quá trình truyền.

- **Không có cơ chế sửa lỗi:** UDP là giao thức không kết nối và không đáng tin cậy, không có cơ chế sửa lỗi hay gửi lại. Vì vậy, khi phát hiện lỗi, nó chỉ có thể loại bỏ gói tin hỏng.

Lý do quá trình DHCP bị đình trệ (stall):

- **Gói Discover không đến được Server:** Gói DHCP Discover bị drop ở lớp Transport của server thì DHCP Server không bao giờ nhận được yêu cầu cấp IP từ client.
- **Không có phản hồi Offer:** Nếu Server không nhận được gói Discover, nó sẽ không thể tạo và gửi gói DHCP Offer cho Client.
- **Hết thời gian chờ (Timeout):** Client sẽ tiếp tục chờ DHCP Offer. Khi hết thời gian chờ, nó gửi lại DHCP Discover. Nếu các gói vẫn lỗi checksum và bị drop, quá trình cứ lặp lại cho đến khi client bỏ cuộc và tự gán địa chỉ APIPA (169.254.x.x), chỉ giao tiếp được với các thiết bị APIPA cùng subnet.

4 Phần 3: Phân Tích Lớp Mạng và Lớp Liên Kết

4.1 Mô tả các bước thực hiện

- **Bước 1:** Tìm địa chỉ Default Gateway bằng lệnh `ipconfig`.
- **Bước 2:** Bắt đầu capture trong Wireshark.
- **Bước 3:** Chạy lệnh `nslookup google.com 8.8.8.8` để thực hiện DNS query.
- **Bước 4:** Dừng capture.
- **Bước 5:** Áp dụng bộ lọc `dns` để chỉ hiển thị các gói tin DNS.

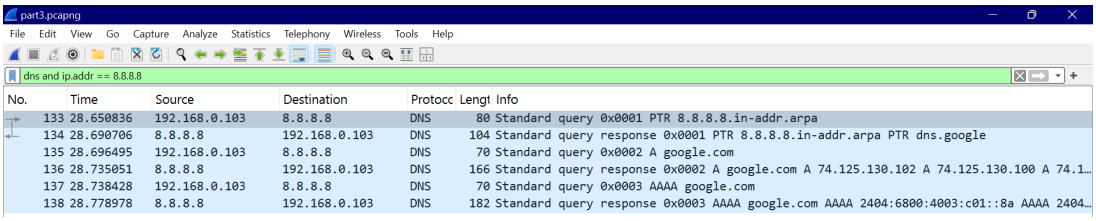
4.2 Câu hỏi phân tích và trả lời

4.2.1 Câu 1: Địa chỉ IP trong DNS Query

Câu hỏi: Địa chỉ IP nguồn/đích:
Xác định gói tin DNS Query được gửi từ host của bạn tới máy chủ Google DNS.
Địa chỉ IP nguồn là gì?
Địa chỉ IP đích là gì?
Giải thích vì sao các địa chỉ IP nguồn và đích này vẫn giữ nguyên khi gói tin truyền qua Internet đến máy chủ Google DNS.
Trả lời:

Trong gói DNS Query gửi tới Google DNS (gói số 181 trong hình):

- **Địa chỉ IP nguồn:** `192.168.0.103` → Địa chỉ IPv4 của Wi-Fi.
- **Địa chỉ IP đích:** `8.8.8.8` → Google Public DNS.



Hình 20: Gói DNS Query gửi tới Google DNS (8.8.8.8)

Vì sao 2 địa chỉ IP này "giữ nguyên"khi đi qua Internet?

Việc địa chỉ IP Nguồn (Source IP Address) và địa chỉ IP Đích (Destination IP Address) không thay đổi khi gói tin đi qua Internet là một nguyên tắc cốt lõi của thiết kế lớp Mạng (Network Layer), đảm bảo dịch vụ truyền thông end-to-end (từ đầu cuối đến đầu cuối) trên toàn mạng lưới.

- Khi host nguồn tạo datagram, nó gán IP nguồn là địa chỉ IP của chính nó và IP đích là địa chỉ IP của host đích cuối cùng (ở đây là máy chủ DNS 8.8.8.8).
- Các router trung gian chỉ thực hiện chức năng chuyển tiếp (forwarding) cục bộ: chúng đọc trường Địa chỉ IP Đích trong header IP, tra bảng chuyển tiếp (forwarding table) để chọn cổng ra thích hợp, rồi gửi tiếp gói tin.

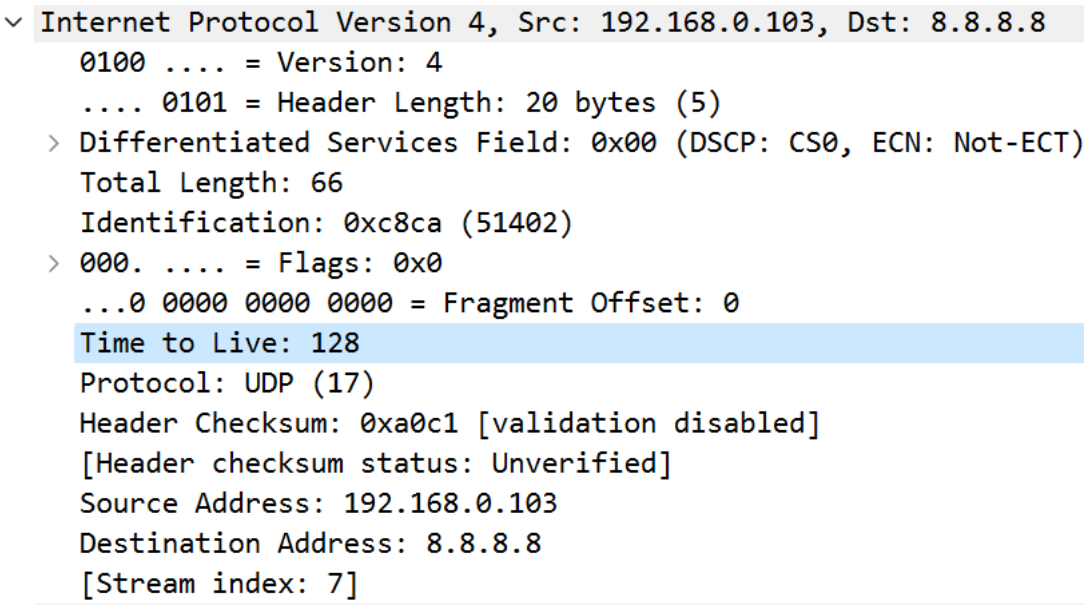
- Nếu Địa chỉ IP Đích bị thay đổi giữa chừng, các router kế tiếp sẽ mất thông tin cần thiết để định tuyến gói tin đến host nhận cuối cùng. Do đó, Địa chỉ IP Đích phải giữ nguyên từ host nguồn đến host đích.
- Vì vậy, từ lúc rời laptop cho đến khi tới Google DNS, datagram vẫn giữ nguyên cặp địa chỉ IP 192.168.0.103 → 8.8.8.8, đảm bảo dịch vụ truyền thông end-to-end giữa hai host.

4.2.2 Câu 2: Time To Live (TTL)

Câu hỏi: Tìm trường Time to Live (TTL) trong phần tiêu đề IP của gói tin DNS Query. Giá trị TTL ban đầu này biểu thị điều gì, và router/cổng (gateway) cục bộ sẽ điều chỉnh giá trị đó như thế nào khi thực hiện chuyển tiếp (forward) gói tin?
Trả lời:

Đầu tiên, lọc gói DNS Query đầu tiên gửi tới máy chủ DNS bằng filter: `dns && ip.dst == 8.8.8.8`. Khi mở phần **Internet Protocol** của gói tin này, trường TTL hiển thị như sau:

- `Time to live: 128`



Hình 21: Gói DNS Query - Trường TTL trong IP header

Giá trị TTL = 128 là **TTL ban đầu** do host nguồn (máy tính chạy hệ điều hành Windows) gán khi tạo IP datagram. Trường TTL cho biết *số bước nhảy tối đa* (max hops) mà gói tin được phép đi qua trước khi bị loại bỏ, giúp ngăn chặn các *vòng lặp định tuyến* khiến gói tin tồn tại vô hạn trên mạng.

Cách default gateway cục bộ xử lý TTL: Default gateway (router Wi-Fi trong mạng LAN) là **bước nhảy (hop) đầu tiên** trên đường đi của gói DNS Query ra Internet. Tại gateway này, TTL được xử lý theo chuẩn như sau:

1. Router nhận gói IP với TTL ban đầu là 128.
2. Router **giảm TTL đi 1** rồi mới chuyển tiếp (forward) gói tin:
   - Khi gói rời khỏi mạng LAN, TTL còn **127**.
3. Mỗi router trung gian tiếp theo trên đường đi tiếp tục lặp lại quy trình này, mỗi hop lại giảm TTL xuống 1.
4. Nếu tại một router nào đó, sau khi giảm mà TTL trở thành **0**, router sẽ:
   - Loại bỏ (drop) gói tin.
   - Gửi ngược về host nguồn một bản tin `ICMP Time Exceeded` để báo gói tin đã hết "tuổi thọ" trên mạng.

Trong capture trên Wireshark, chỉ quan sát được TTL = 128 tại thời điểm gói vừa rời host nguồn. Tuy nhiên, về mặt hoạt động chuẩn, ngay khi gói đi qua default gateway cục bộ, TTL của gói DNS Query đã được giảm xuống còn 127.

4.2.3 Câu 3: Địa chỉ MAC của Router/Gateway

Câu hỏi: Xác định địa chỉ MAC của router/cổng mặc định (gateway) cục bộ của bạn (bằng cách sử dụng lệnh arp -a trong cửa sổ dòng lệnh, hoặc bằng cách tìm địa chỉ MAC gắn với địa chỉ IP gateway trong file bắt gói tin (capture)).

Trả lời:

- Địa chỉ IP của router (Default Gateway) thu được từ lệnh `ipconfig` trong Command Prompt là `192.168.0.1`:

```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::e06c:7bb9:7844:c30e%4
   IPv4 Address. . . . . . . . . . . : 192.168.0.103
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.0.1
```

Hình 22: Kết quả lệnh ipconfig - Địa chỉ IP của router

- Địa chỉ MAC của router/gateway cục bộ, xác định từ kết quả lệnh `arp -a` tương ứng với IP Gateway `192.168.0.1`, là `5c-a6-e6-e1-90-01`:

```
C:\Users\User>arp -a

Interface: 192.168.0.103 --- 0x4
  Internet Address      Physical Address      Type
  192.168.0.1           5c-a6-e6-e1-90-01     dynamic
```

Hình 23: Kết quả lệnh arp -a - Địa chỉ MAC của router/gateway

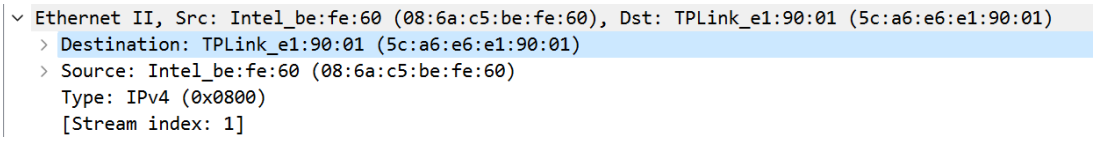
4.2.4 Câu 4: Địa chỉ trong Link Layer Header

Câu hỏi: Các địa chỉ nguồn và đích trong tiêu đề tầng Liên kết (Link Layer) (Ethernet hoặc 802.11) là gì? Về bản chất, những địa chỉ này khác gì so với các địa chỉ IP mà bạn đã tìm được ở tầng Mạng?

Trả lời:

Trong header Ethernet (Link Layer) của gói DNS Query, địa chỉ Nguồn và địa chỉ Đích là:

- **Địa chỉ MAC nguồn:** `08-6a-c5-be-fe-60` → card Wi-Fi của laptop.
- **Địa chỉ MAC đích:** `5c-a6-e6-e1-90-01` → router/gateway Wi-Fi.



Hình 24: Ethernet header của gói DNS Query - Địa chỉ MAC nguồn và đích

So sánh địa chỉ MAC và địa chỉ IP:

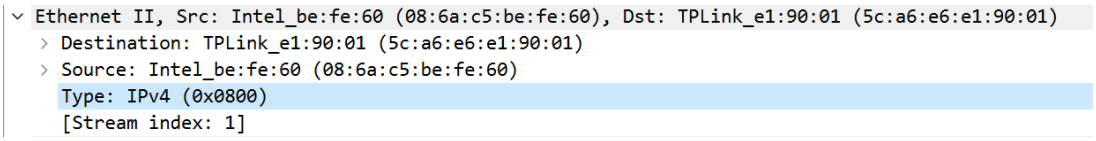
Đặc điểm	**Địa chỉ MAC (Lớp Liên kết / Lớp 2)**	**Địa chỉ IP (Lớp Mạng / Lớp 3)**
Phạm vi	Chỉ sử dụng trong **mạng cục bộ** (Local Network Segment). Gói tin cần đổi MAC ở mỗi bước nhảy (hop) khi đi qua router.	Sử dụng trên **toàn cầu** (Internet). Địa chỉ IP Nguồn và Đích không thay đổi từ máy gửi đến máy nhận cuối cùng (trừ khi có NAT).
Cấp phát	Được **nhà sản xuất** thiết bị mạng gán cố định (Physical Address) và thường là duy nhất trên toàn thế giới.	Được **Quản trị viên Mạng/ISP** hoặc máy chủ DHCP cấp phát (Logical Address) và có thể thay đổi.
Định dạng	**48 bit**, thường được viết dưới dạng 6 nhóm 2 ký tự hệ thập lục phân (Hex), cách nhau bằng dấu gạch ngang hoặc dấu hai chấm (ví dụ: `00:6a:c5:be:fe:60`).	**32 bit** (IPv4) hoặc **128 bit** (IPv6), thường được viết dưới dạng thập phân có dấu chấm (ví dụ: `172.20.10.4`).
Mục đích	Cho phép các thiết bị vật lý **trong cùng một mạng** tìm thấy nhau để truyền gói tin.	Cho phép xác định **vị trí logic** của mạng và máy chủ trên mạng toàn cầu.

Bảng 5: So sánh địa chỉ MAC và địa chỉ IP

4.2.5 Câu 5: Trường Type trong Link Layer Header

Câu hỏi: Quan sát tiêu đề tầng Liên kết (Ethernet II hoặc tương tự). Giá trị của trường Type là bao nhiêu, và trường này cho thiết bị nhận (router/gateway của bạn) biết điều gì về loại giao thức tầng trên (protocol) nằm ngay sau tiêu đề Ethernet?

Trả lời:



Hình 25: Trường Type trong Ethernet header

Giá trị `0x0800` là một mã chuẩn (EtherType) và nó cho thiết bị nhận (router/gateway hoặc thiết bị mạng kế tiếp) biết thông tin quan trọng sau:

- **Giao thức Tiếp theo:** Nó chỉ ra rằng giao thức được đóng gói (encapsulated) ngay sau tiêu đề Ethernet là Giao thức Internet phiên bản 4 (IPv4).
- **Chức năng Phân phối:** Sau khi nhận được gói tin, router sẽ nhìn vào giá trị này để biết rằng nó cần chuyển gói dữ liệu này lên Lớp Mạng (Layer 3) và xử lý nó bằng cách sử dụng logic của giao thức IP.

Ví dụ các giá trị EtherType phổ biến:

- `0x0800` → IPv4
- `0x86DD` → IPv6
- `0x0806` → ARP

5 Kết luận

Qua quá trình thực hiện dự án phân tích gói tin mạng với Wireshark, nhóm đã thu được những kiến thức và kinh nghiệm thực tế quý báu về cách thức hoạt động của các giao thức mạng ở nhiều tầng khác nhau. Những bài học quan trọng nhất bao gồm:

5.1 Từ Phần 1: HTTP Traffic Analysis

- **Cơ chế tối ưu hiệu năng của trình duyệt (Concurrency):** Giúp nhận thấy trình duyệt không tải dữ liệu một cách tuần tự mà thực hiện tải song song các tài nguyên (như hình ảnh) để giảm độ trễ.
- **Quy trình thiết lập kết nối tin cậy (3-Way Handshake):** Trước khi bất kỳ dữ liệu HTTP nào (như GET request) được truyền đi, thấy rõ quy trình "bắt tay ba bước"để đồng bộ hóa. Điều này đảm bảo kết nối được thiết lập chắc chắn trước khi trao đổi dữ liệu tầng ứng dụng.
- **Vai trò của Window Size trong tầng giao vận (Flow Control):** Thông qua giá trị Calculated window size, hiểu được cách TCP thực hiện kiểm soát luồng. Giá trị này thông báo cho bên gửi biết dung lượng bộ nhớ đệm còn trống của bên nhận. Đây là cơ chế thiết yếu để ngăn chặn việc gửi dữ liệu quá nhanh gây tràn bộ đệm (buffer overflow), từ đó giảm thiểu việc mất gói tin và tắc nghẽn mạng.

5.2 Từ Phần 2: DHCP Traffic Analysis

- **Hiểu rõ quy trình cấp phát địa chỉ IP qua DHCP (DORA) và cách thức vận hành của từng gói tin:** Qua Wireshark, quá trình Discover → Offer → Request → ACK được quan sát trực quan, giúp hiểu chính xác thời điểm máy khách chưa có IP, thời điểm server đề xuất IP, cũng như sự khác biệt giữa broadcast/unicast trong từng bước.
- **Nhận diện vai trò của ARP trong quá trình xác minh IP sau DHCP:** Dù ARP không xuất hiện trong DORA, các gói ARP Probe và ARP Announcement xuất hiện ngay sau DHCP ACK cho thấy vai trò quan trọng của ARP trong kiểm tra xung đột IP và phân giải MAC của Default Gateway.
- **Nắm được các cơ chế kỹ thuật thực tế của hệ thống mạng:** Kết quả phân tích làm rõ các cơ chế như: Checksum Offloading, cách DHCP xử lý xung đột IP, và lý do sử dụng cổng UDP cố định 67–68 để đảm bảo tính thống nhất, đơn giản và tin cậy trong môi trường chưa có IP.

5.3 Từ Phần 3: Network & Link Layer Analysis

- **Hiểu rõ sự khác biệt và vai trò của IP và MAC:** Qua bài tập giúp hiểu rõ rằng IP là địa chỉ logic dùng để định tuyến end-to-end, nên không thay đổi khi đi qua Internet; còn MAC là địa chỉ vật lý trong LAN, và sẽ thay đổi ở mỗi hop. Điều này giúp em nắm chắc mối quan hệ giữa Layer 2 và Layer 3 trong mạng.
- **Nắm vững cơ chế TTL và lý do phải giảm tại mỗi router:** TTL giúp ngăn gói tin chạy vòng lặp vô hạn. Mỗi router giảm TTL xuống 1; nếu TTL = 0, gói bị drop và gửi ICMP Time Exceeded. Bài tập giúp hiểu chính xác cách router xử lý gói IP và vì sao TTL là thông số quan trọng khi phân tích mạng.

- **Biết cách phân tích gói tin và xác định router/gateway bằng Wireshark:** Học được cách dùng Wireshark và `arp -a` để nhận diện MAC của gateway, đọc Ethernet header, EtherType và các trường IP/UDP/DNS. Nhờ đó mà tự tin hơn trong việc phân tích các gói tin và hiểu rõ quá trình encapsulation từ Layer 2 → Layer 3 → Layer 4 → Layer 7.

Dự án này không chỉ củng cố kiến thức lý thuyết mà còn trang bị cho nhóm các kỹ năng thực hành cần thiết trong việc giám sát, phân tích và khắc phục sự cố mạng.

6 Tài liệu tham khảo

1. Kurose, J. F., & Ross, K. W. (2021). *Computer Networking: A Top-Down Approach* (7th ed.). Pearson.

2. Kurose, J. F., & Ross, K. W. (2005-2016). *Wireshark Lab: HTTP v7.0*. Supplement to Computer Networking: A Top-Down Approach (7th ed.).

3. Kurose, J. F., & Ross, K. W. (2005-2016). *Wireshark Lab: DHCP v7.0*. Supplement to Computer Networking: A Top-Down Approach (7th ed.).

4. Kurose, J. F., & Ross, K. W. (2005-2016). *Wireshark Lab: DNS v7.0*. Supplement to Computer Networking: A Top-Down Approach (7th ed.).

5. Giới Thiệu về Mạng Máy Tính. (2024). *Wireshark Network Analysis Tutorial* [Video]. YouTube. Retrieved from https://www.youtube.com/watch?v=qTaOZrDnMzQ

6. PowerCert Animated Videos. (2024). *DHCP Explained - Dynamic Host Configuration Protocol* [Video]. YouTube. Retrieved from https://www.youtube.com/watch?v=TkCSr30UojM