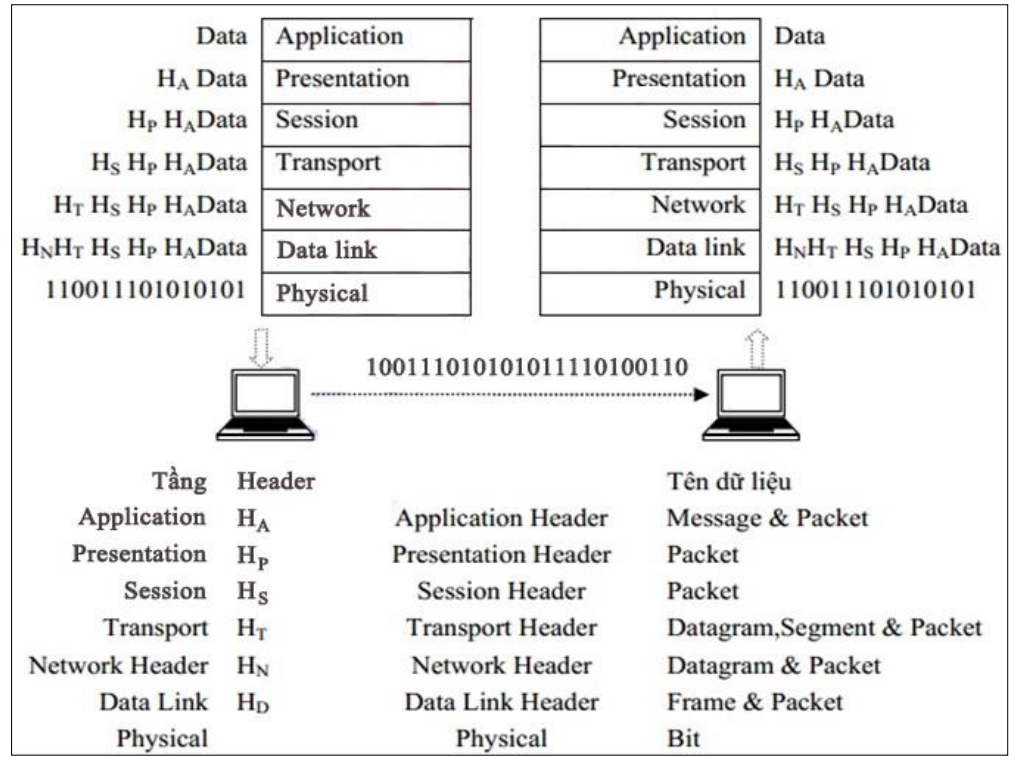
**Mô hình OSI**

Mô hình kết nối các hệ thông mở OSI là mô hình căn bản về các tiến trình truyền thông, thiết lập các tiêu chuẩn kiến trúc mạng ở mức Quốc tế, là cơ sở chung để các hệ thống khác nhau có thể liên kết và truyền thông được với nhau. Mô hình OSI tổ chức các giao thức truyền thông thành 7 tầng, mỗi một tầng giải quyết một phần hẹp của tiến trình truyền thông, chia tiến trình truyền thông thành nhiều tầng và trong mỗi tầng có thể có nhiều giao thức khác nhau thực hiện các nhu cầu truyền thông cụ thể.

**Mô hình OSI bao gồm 7 tầng:**

* **Tầng ứng dụng (Application Layer):**
  + Cung cấp giao diện sử dụng cho người dùng
  + Cung cấp các dịch vụ mạng. Mỗi dịch vụ sử dụng giao thức và cổng dịch vụ riêng
  + Thực hiện thông qua các tiến trình phân tán.
* **Tâng trình bày (Presentation Layer):**
  + Biểu diễn thông tin người sử dụng phù hợp với thông tin làm việc của mạng và ngược lại.
  + Chuyển đổi dữ lliệu đi trên mạng từ một loại biểu diễn này sang một loại khác.
* **Tầng phiên (Session Layer):**
  + Thiết lập các giao dịch (phiên làm việc) giữa các thực thể đầu cuối: thiết lập, duy trì, huỷ bỏ và đồng bộ.
* **Tầng giao vận (Transport Layer):**
  + Thực hiện việc chia các gói tin lớn thành các gói tin nhỏ hơn trước khi gửi đi và đánh số các gói tin và đảm bảo chúng chuyển đúng thứ tự.
  + Vận chuyển giữa các host.
  + Vận chuyển tin cậy
  + Thiết lập, duy trì, kết nối các mạch ảo.
  + Phát hiện lỗi, phục hồi thông tin và điều khiển luồng.
* **Tầng mạng (Network Layer):**
  + Định tuyến các gói tin từ nguồn tới đích có thể trong cùng một mạng hoặc khác mạng nhau
  + Điều khiển tắc nghẽn (Congestion Control)
* **Tầng liên kết dữ liệu (Data link Layer):**
  + Thiết lập các liên kết, duy trì và huỷ bỏ các liên kết dữ liệu.
  + Kiểm soát lỗi và kiểm soát lưu lượng.
  + Chia thông tin thành các khung thông tin (Frame), truyền các khung tuần tự và xử lý các frame xác nhận (Acknowledgement Frame).
  + Tháo gỡ các khung thành chuỗi bit không cấu trúc chuyển xuống tầng vật lý. Tầng 2 bên thu, tái tạo chuỗi bít thành các khung thông tin.
* **Tầng vật lý (Physical Layer):** 
  + Xác định các chức năng, thủ tục về điện, cơ, quang để kích hoạt, duy trì và giải phóng các kết nối vật lý giữa các hệ thống mạng.
  + Cung cấp các cơ chế về điện, cơ thủ tục, … nhằm thực hiện việc kết nối các phần tử của mạng thành một hệ thống bằng các phương pháp vật lý.

Cấu trúc thành phần của trong mô hình OSI



* **Lớp Vật lý (Physical Layer):** Các thành phần của gói tin ở lớp này bao gồm bit (0 hoặc 1), đường truyền, tần số sóng, độ dài sóng, điện áp... Độ dài trường dữ liệu không giới hạn nhưng phụ thuộc vào phương tiện truyền thông sử dụng như cáp đồng trục, cáp xoắn đôi hoặc sóng vô tuyến.
* **Lớp Liên kết dữ liệu (Data Link Layer):** Các thành phần của gói tin ở lớp này bao gồm khung (frame), địa chỉ MAC (Media Access Control), kiểm soát lỗi, chuyển tiếp gói tin, đồng bộ hóa. Độ dài trường dữ liệu của tầng này là từ 46 đến 1500 byte.
* **Lớp Mạng (Network Layer):** Các thành phần của gói tin ở lớp này bao gồm giao thức IP (Internet Protocol), địa chỉ IP, bảng định tuyến, giao thức truyền tải... Độ dài trường dữ liệu của tầng này là từ 20 đến 60 byte.
* **Lớp Giao vận (Transport Layer):** Các thành phần của gói tin ở lớp này bao gồm các giao thức như TCP (Transmission Control Protocol), UDP (User Datagram Protocol), cổng (port)... Độ dài trường dữ liệu của tầng này là từ 20 đến 60 byte.
* **Lớp Phiên (Session Layer):** Lớp này thiết lập, quản lý và giải phóng các kết nối phiên làm việc giữa các ứng dụng trên các thiết bị khác nhau. Các thành phần của gói tin ở lớp này bao gồm các yêu cầu bắt đầu phiên, yêu cầu kết thúc phiên, yêu cầu đồng bộ hóa phiên, đồng hồ phiên. Độ dài trường dữ liệu của tầng này là không giới hạn.
* **Lớp trình diễn (Presentation layer):** Ở lớp này, thành phần của gói tin bao gồm dữ liệu được mã hóa và định dạng để có thể hiển thị trên các thiết bị khác nhau. Độ dài trường dữ liệu của tầng này là không giới hạn.
* **Lớp ứng dụng (Application layer):** Tại lớp này, thành phần của gói tin bao gồm các dữ liệu và thông tin điều khiển của ứng dụng. Độ dài trường dữ liệu của tầng này là

**Một số giao thức mạng:**

* **TCP/IP(TransmissionControlProtocol/Internet Protocol):** Là một bộ giao thức được sử dụng để kết nối các thiết bị trên Internet hoặc mạng cục bộ. Giao thức TCP được sử dụng để đảm bảo rằng dữ liệu được truyền tải đến đúng địa chỉ và đảm bảo tính toàn vẹn của dữ liệu. Giao thức IP được sử dụng để định tuyến dữ liệu từ nguồn đến đích.
* **UDP(User Datagram Protocol):** Là một giao thức truyền tải dữ liệu dạng gói (packet) được sử dụng để truyền tải dữ liệu nhanh và hiệu quả hơn TCP. Tuy nhiên, UDP không đảm bảo tính toàn vẹn và độ tin cậy của dữ liệu.
* **HTTP(Hypertext Transfer Protocol):** Là giao thức được sử dụng để truyền tải các trang web và dữ liệu khác trên Internet. HTTP sử dụng phương thức yêu cầu/đáp ứng (request/response) giữa máy khách (client) và máy chủ (server) để truyền tải dữ liệu. Cổng mặc định của HTTP là 80 và 443. Hai cổng này đều được bảo mật.
* **HTTPS(Hypertext Transfer Protocol Secure):** Là phiên bản an toàn của HTTP, sử dụng mã hóa SSL/TLS để bảo vệ dữ liệu khi truyền tải giữa máy khách và máy chủ.
* **SSL/TLS(Secure Sockets Layer/Transport Layer Security):** Là giao thức được sử dụng để bảo mật các thông tin nhạy cảm như mật khẩu, thông tin tài khoản ngân hàng và thẻ tín dụng khi truyền tải trên Internet. Các trình duyệt web hiện đại đều hỗ trợ SSL/TLS và nhiều trang web sử dụng SSL/TLS để đảm bảo an toàn cho người dùng.
* **SMB(Server Message Block):** Là giao thức được sử dụng để chia sẻ tài nguyên và truyền tải tập tin giữa các thiết bị trong mạng Windows. SMB được sử dụng để chia sẻ tập tin, máy in, thư mục và các tài nguyên khác trên mạng.
* **FTP(File Transfer Protocol):** Là giao thức được sử dụng để truyền tải các tệp từ máy tính này sang máy tính khác trong mạng. Nó cho phép người dùng tải lên và tải xuống các tệp từ máy chủ FTP. Cổng mặc định của FTP là 20/21.
* **SMTP (Simple Mail Transfer Protocol):** là giao thức được sử dụng với hai chức năng chính: Chuyển email từ mail server nguồn đến mail server đích và chuyển email từ người dùng cuối sang hệ thống mail. Cổng mặc định của SMTP là 25 và cổng SMTP được bảo mật (SMTPS) là 465 (Không phải tiêu chuẩn).
* **DNS(Domain Name System):** Là giao thức được sử dụng để chuyển đổi tên miền thành địa chỉ IP của máy chủ đó. Đây là giao thức quan trọng trong việc liên kết các tên miền với các địa chỉ IP của các máy chủ.
* **DHCP(Dynamic Host Configuration Protocol):** Là giao thức được sử dụng để cấu hình các thiết bị mạng, như máy tính, để tự động nhận địa chỉ IP và các thông số mạng khác từ máy chủ DHCP.
* **ARP (Address Resolution Protocol):**  là giao thức được sử dụng để tìm địa chỉ MAC của thiết bị từ địa chỉ IP đã biết của nó. Điều này có nghĩa là thiết bị nguồn đã biết địa chỉ IP nhưng không biết địa chỉ MAC của thiết bị đích. Địa chỉ MAC của thiết bị là bắt buộc vì bạn không thể giao tiếp với thiết bị trong mạng cục bộ (Ethernet) mà không biết địa chỉ MAC của nó.
* **RARP (Reverse Address Resolution Protocol):** ngược lại với giao thứ ARP, dung để tìm địa chỉ IP khi đã biết địa chỉ MAC của thiết bị.
* **SNMP(Simple Network Management Protocol):** Là giao thức được sử dụng để quản lý các thiết bị mạng, như định vị và giám sát các thiết bị trong mạng.
* **ICMP(Internet Control Message Protocol):** Là giao thức được sử dụng để gửi các thông báo lỗi và thông báo kiểm tra mạng giữa các thiết bị trong mạng.
* **OSPF(Open Shortest Path First):** Là giao thức định tuyến được sử dụng để tính toán đường đi ngắn nhất giữa các thiết bị trong mạng. Nó được sử dụng chủ yếu trong các mạng lớn và phức tạp.
* **Telnet:** Là phương thức chính được sử dụng để quản lý các thiết bị mạng ở cấp lệnh. Không giống như SSH, Telnet không cung cấp kết nối an toàn, mà chỉ cung cấp kết nối không bảo mật cơ bản. Cổng mặc định của Telnet là 23.
* **SSH (Secured Shell):** Là phương thức chính được sử dụng để quản lý các thiết bị mạng một cách an toàn ở cấp lệnh. SSH thường được sử dụng như sự thay thế cho Telnet, vì giao thức này không hỗ trợ các kết nối an toàn. Cổng mặc định của SSH là 22.

**Wireshark:**

1. Wireshark là gì?

Wireshark là một phần mềm phân tích giao thức mạng được sử dụng rộng rãi trên thế giới. Nó cho phép người quản lý mạng xem được những gì xảy ra trên mạng ở cấp độ nhỏ nhất. Wireshark được bắt đầu từ một dự án nhỏ của Gerald Combs từ năm 1998. Phát triển thành công đến ngày nay do sự đóng góp của nhiều chuyên gia mạng toàn cầu.

1. Ứng dụng của Wireshark

- Khắc phục sự cố mạng. Tính năng này được người quản trị mạng sử dụng rất nhiều.

- Kiểm tra các vấn đề bảo mật. Tính năng hữu ích với các kỹ sư bảo mật mạng

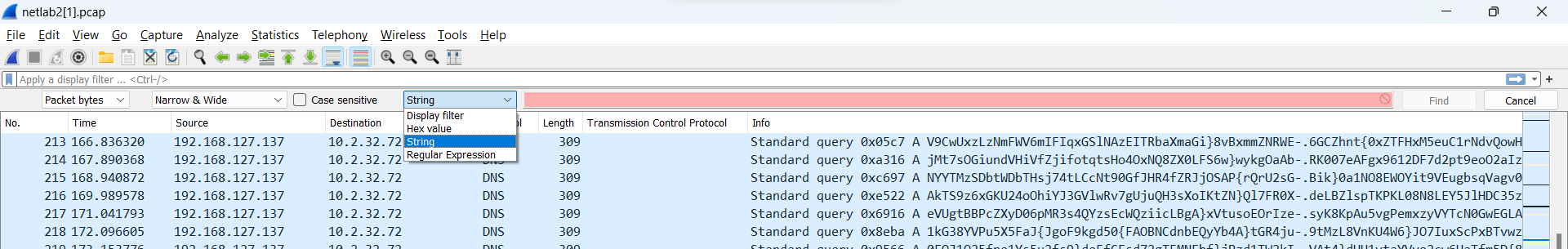
- Các kỹ sư QA dùng Wireshark để xác minh các ứng dụng mạng

- Developer dùng wireshark để gỡ lỗi triển khai giao thức

- Và hầu hết mọi người đều có thể dùng Wireshark để học về những thứ nằm trong giao thức mạng.

1. Các tính năng của Wireshark
   1. **Lọc gói tin**
      1. Tìm kiếm gói tin

Để tìm kiếm gói tin, chúng ta có thể sử dụng thanh công cụ **“Find Packet”**bằng cách bấm phím **Ctrl + F,** một hộp thoại mới sẽ xuất hiện nằm giữa thanh Filter và Packet List:



Chúng ta có thể tìm kiếm packet dựa vào:

* + - Display Filter: Nhập vào một biểu thức filter (expression-based filter), Wireshark sẽ tìm kiếm các gói tin khớp với biểu thức này.
    - Hex value: Tìm kiếm dựa trên giá trị Hex.
    - String: Tìm kiếm dựa trên chuỗi dữ liệu.
    - Regular Expression: Tìm kiểm dựa trên biểu thức Regex

Cú pháp của Wireshark Display Filter phần lớn tuân theo cú pháp:

protocol.feature.subfeature COMPARISION\_OPERATOR value LOGICAL\_OPERATOR protocol.feature.subfeature COMPARISION\_OPERATOR value

Trong đó:

COMPARISION OPERATOR bao gồm

| **Operator** | **Ý nghĩa** |
| --- | --- |
| == | Bằng (equal to) |
| != | Không bằng (not equal to) |
| > | Lớn hơn (greater than) |
| < | Nhỏ hơn |
| >= | Lớn hơn hoặc bằng (greater than or equal) |
| <= | Nhỏ hơn hoặc bằng (less than or equal) |

**LOGICAL OPERATOR**bao gồm:

| **Operator** | **Ý nghĩa** |
| --- | --- |
| and | tất cả các điều kiện phải được thỏa mãn |
| or | một trong các điều kiện được thoả mãn |
| xor | một và chỉ một điều kiện được thỏa mãn |
| not | không điều kiện nào được phép thảo mãn |

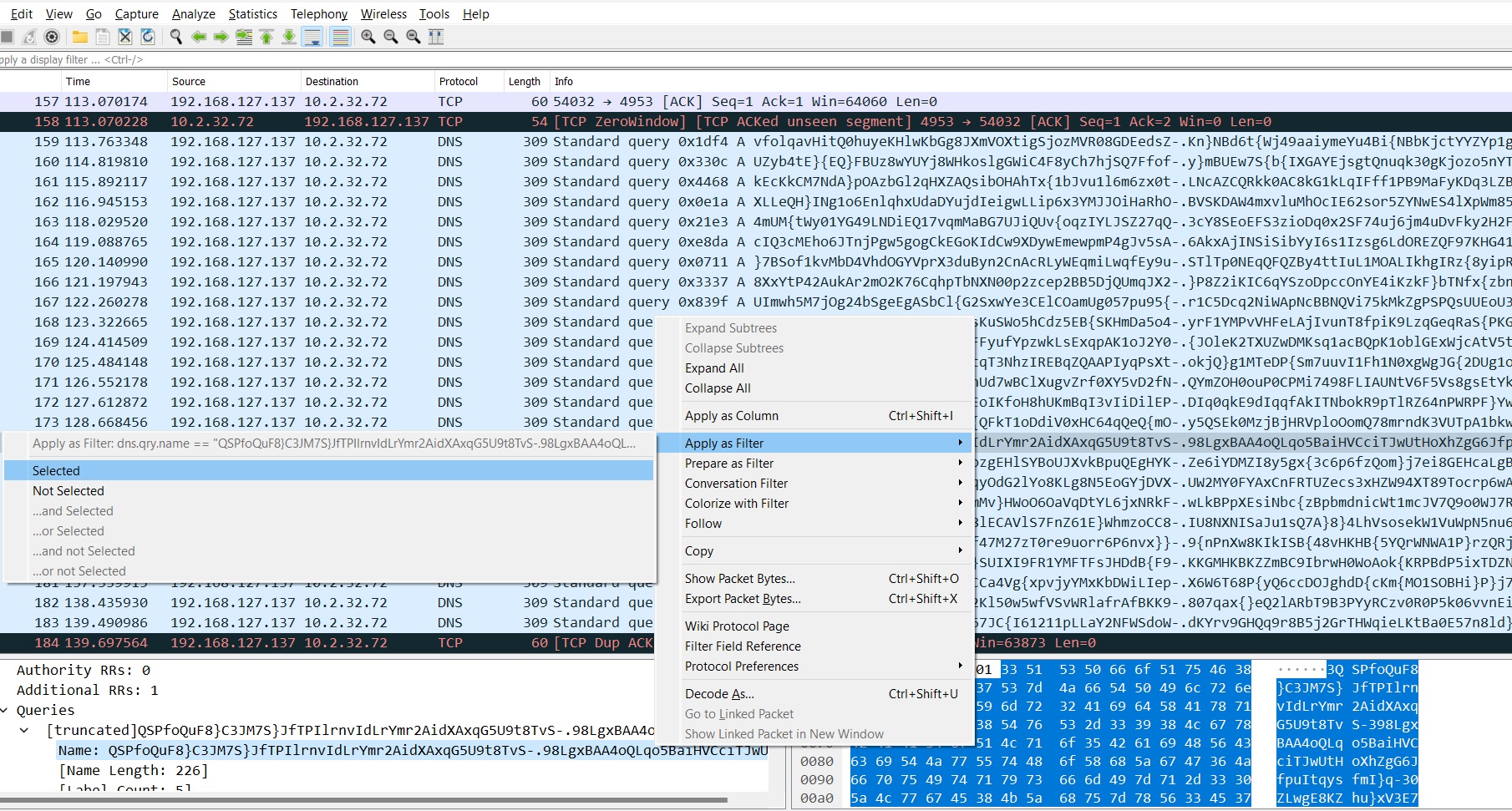
\* Ở khung cửa sổ Packet List sẽ cung cấp cho chúng ta các thông tin như:

* No: Số thứ tự của gói tin trong file capture hiện tại.
* Time: Thời gian tương đối mà gói tin này được bắt, tính từ lúc bắt đầu quá trình bắt gói tin.
* Source: địa chỉ source IP của kết nối.
* Destination: địa chỉ destination IP của kết nối.
* Length: chiều dài của gói tin.
* Protocol: giao thức của gói tin
* Info: các thông tin tổng quan liên quan đến gói tin.

\* Ở khung cửa sổ của Packet Details sẽ cho ta thông tin chi tiết từng Layer của packet như:

* Frame: Interface
* Ethernet: Destination, Source, Mac Address
* Internet: Source IP, Destination IP, TTL, Protocol, Flags, Checksum, Identification, Total Length…
* TCP/UDP/ICMP: Source Port, Destination Port, Sequence Number, ACK Number, Flags, TCP Options …
* Application Layer: HTTP, DNS, SMTP…

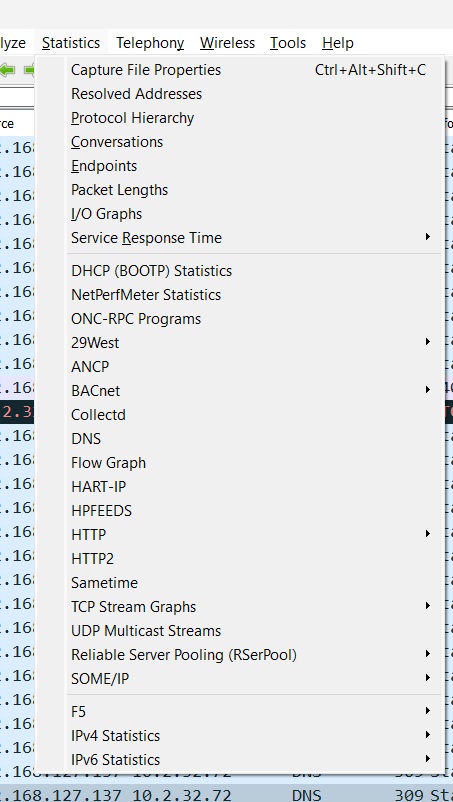
\* Ta cũng có thể tạo thêm các filter ở đây. Chỉ cần click chuột phải vào một trong số các chi tiết, rồi chọn **Apply as Filter** để tạo một filter dựa theo đó.



#### **Wireshark Statistics**

Phần menu Statistic cung cấp những thông tin thống kê có giá trị liên quan đến file capture hiện tại như:

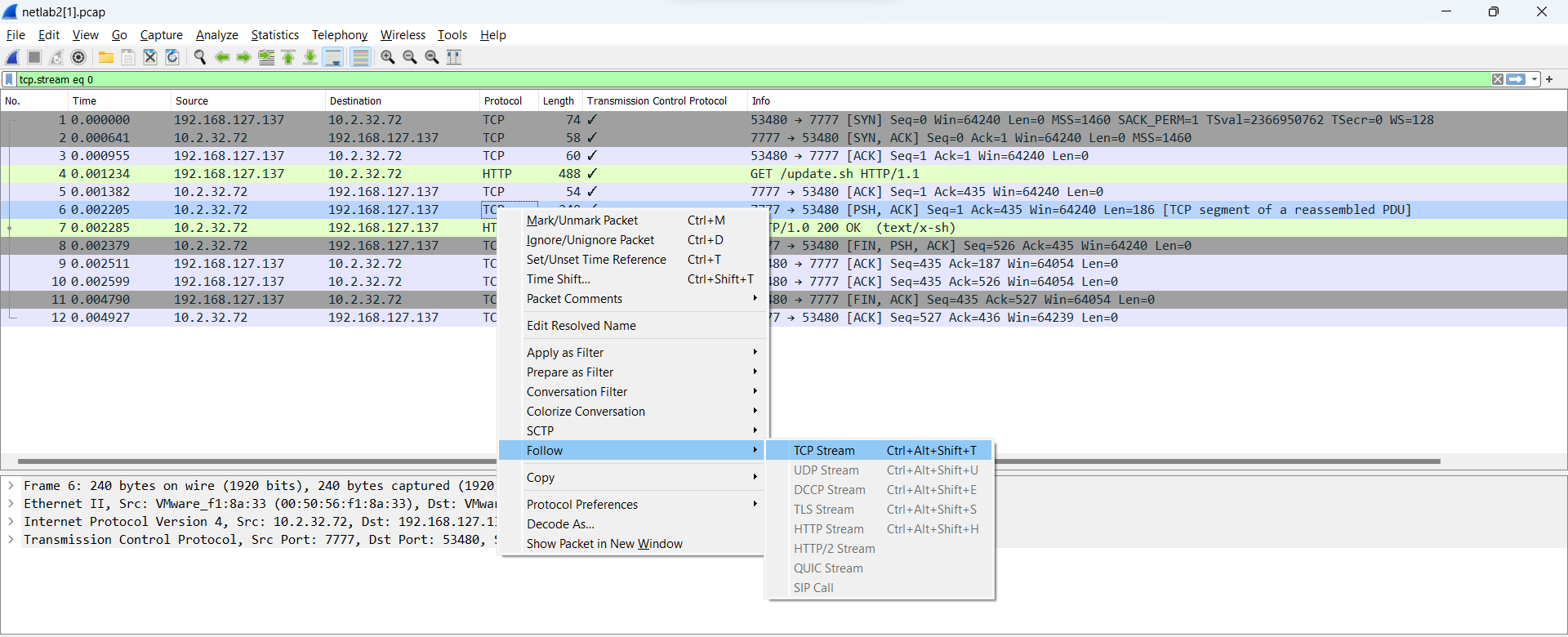
* Capture File Properties: Các thông số tổng quan của file capture
* Protocol Hierarchy: Tổng quan về protocol
* Conversation: thông tin về các luồng trao đổi giữa client và server
* Endpoints: Danh sách những IP tham gia kết nối, số lượng packets và bytes tương ứng.
* Packet Lengths: Thống kê về chiều dài của các gói tin tham gia kết nối.
* I/0 Graph: biểu đồ kết nối.
* Thống kê liên quan đến các giao thức như: HTTP, HTTP2, DNS, DHCP…
* IPv4/IPv6 Statistics: thống kê về danh sách IP, số lượng packet, tần số kết nối.

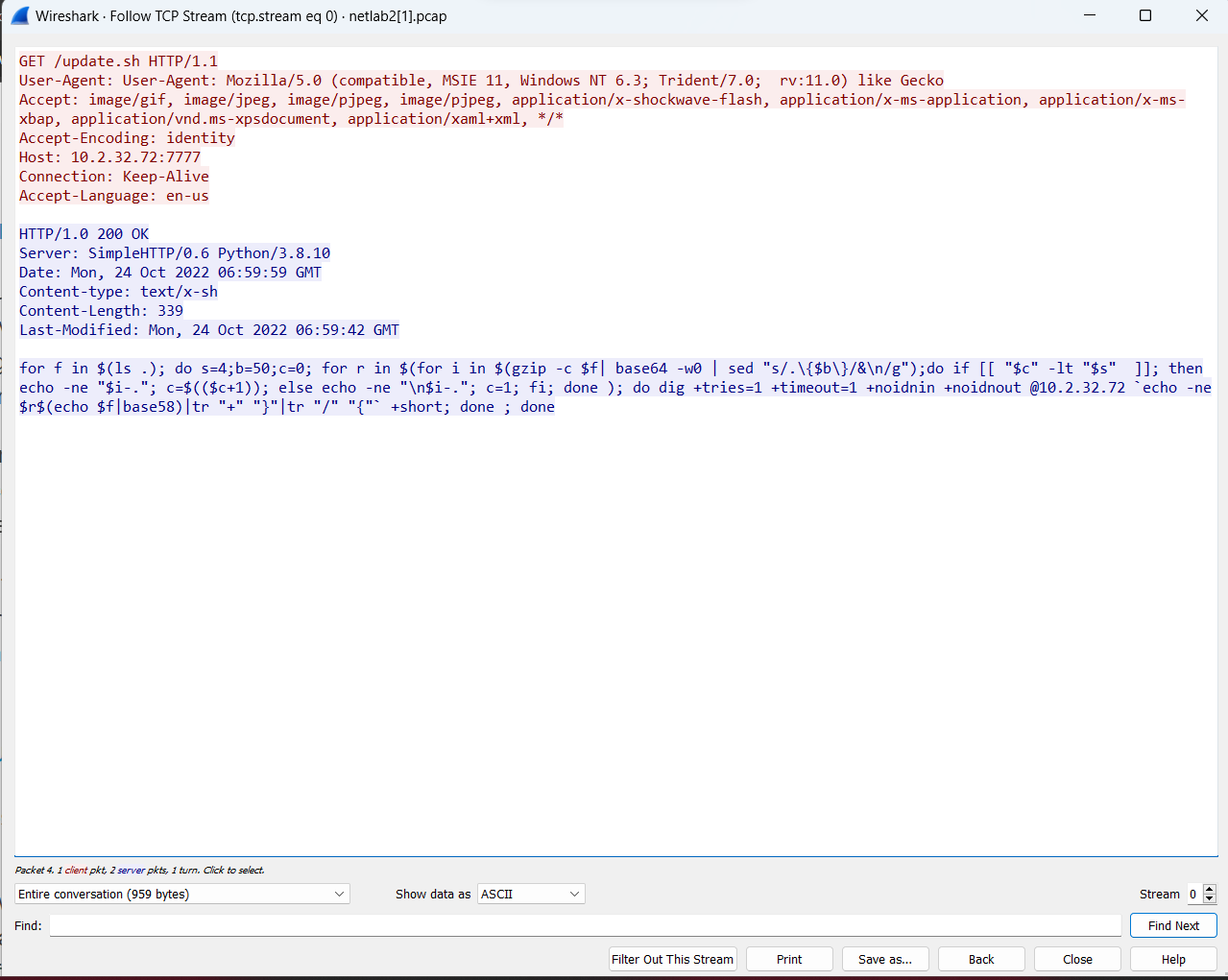


* 1. **Follow**

Một tính năng hữu ích khác là Follow một luồng giao thức xác định. Click chuột phải vào một packet chứa giao thức của cuộc trao đổi ta muốn theo dõi và chọn “**Follow**” và chọn “**Protocol Stream**” đó.

Ví dụ muốn theo dõi dữ liệu trao đổi giữa Client và Server bằng giao thức TCP:

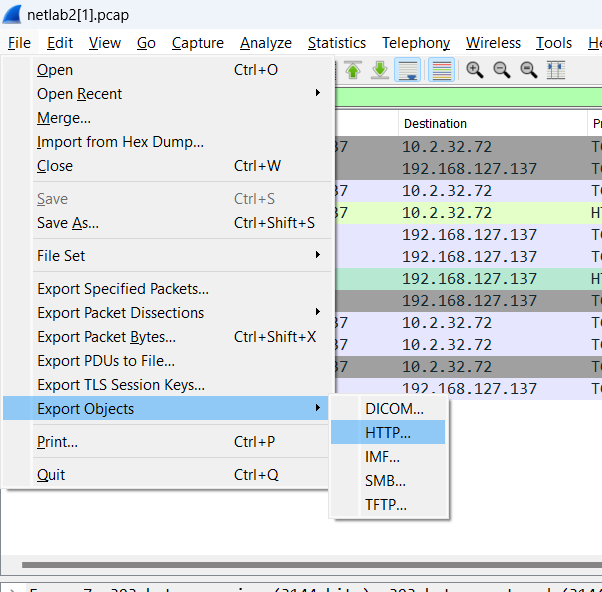




Ta có thể click vào dòng dữ liệu để xem nó nằm trong packet nào.

* 1. **Export Object**

Để sử dụng tính năng này ta chọn “**File > Export Object**”



Tính năng này giúp hiển thị những file được được gửi qua lại trong quá trình trao đổi dữ liệu bằng phương thức được chọn.

**Tshark**

TShark là công cụ phân tích giao thức mạng, một phiên bản terminal của Wireshark được sử dụng khi giao diện đồ họa không có sẵn hoặc không cần thiết. Với TShark bạn có thể bắt gói tin, đọc/phân tích gói tin từ file đã lưu...

**Bắt nhưng gói tin qua interface**

Ta sử dụng câu lệnh

**tshark -i <interface>**

Để xem các danh sách network interface trên máy tính ta sử dụng câu lệnh:

**tshark -D**

Kết hợp với tham số –w để lưu những gói tin vừa bắt được vào file mới

**tshark -i <interface> -w <file-name>.pcap**

**Đoc file pcap**

Với tham số -r ta có thể xem nội dung file .pcap đã lưu

**tshark -r <file-name>.pcap**

**Lọc gói tin**

Sử dụng tham số -f cùng với biểu thức dựa trên cú pháp của Berkeley Packet Filter (BPF)

**tshark -i <interface> -f "expression"**

Ngoài ra ta có thể lọc bằng tham số -Y kết hợp với biểu thức có cấu trúc như trong Display filter của Wireshark

**tshark -i <interface> -Y "expression"**

**Liệt kê trường dữ liệu của gói tin**

Sử dụng tham số -T fields kết hợp với tham số -e cùng tên trường

**tshark -i <interface> -T fields –e <tên\_tường\_1> -e <tên\_trường2> …**

**Các tùy chọn khác của Tshark hỗ trợ phân tích gói tin**

Trong quá trình phân tích ta có thể sử dụng tùy chọn -V để hiển thị chi tiết thông tin từng gói tin như số frame, giao thức...

**tshark -i <interface> -V**

Tùy chọn -O tương tự -V nhưng chỉ hiện thị chi tiết gói tin sử dụng giao thức mà chúng ta chỉ định.

**tshark -i <interface> -O <protocol>**

**Thư viện Scapy**

**Hàm rdpcap()**

Cú pháp: *rdpcap(r“PATH”)*

Trong đó PATH là đường dẫn tới file .pcap

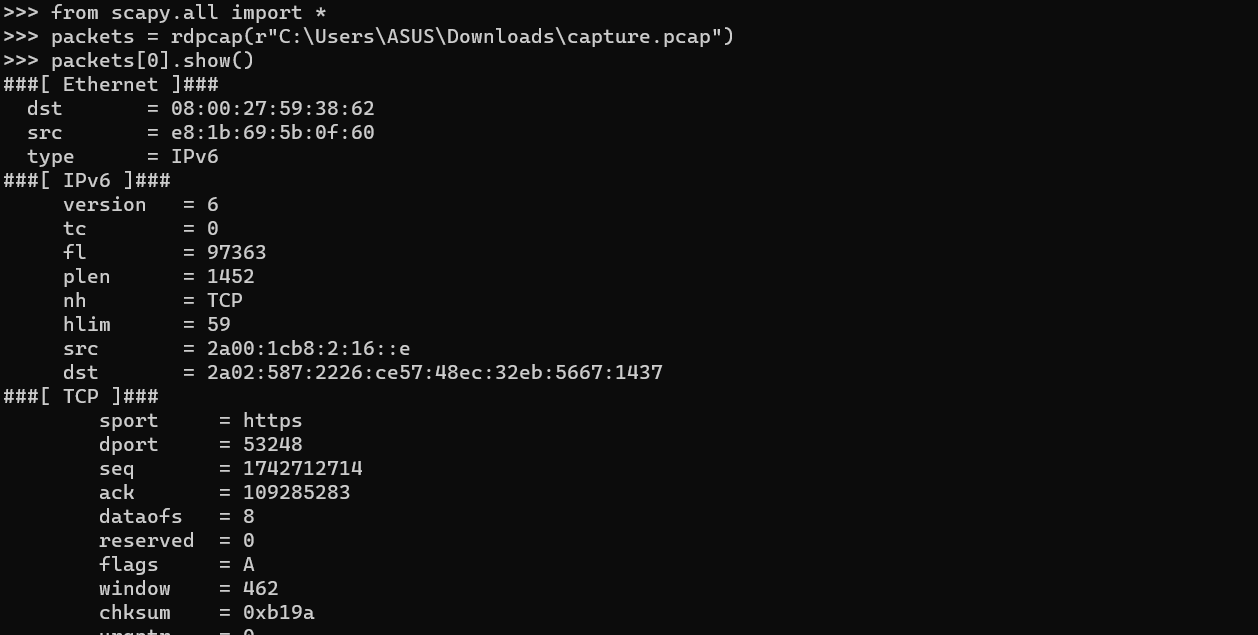
Hàm rdpcap() trả về danh sách các gói tin trong file .

**Hàm show()**

Cú pháp: *packet.show()*

Hàm show() in ra thông tin chi tiết của một gói tin

Ví dụ:



**Lấy dữ liệu trong trường gói tin**

Cú pháp: *packet[‘layer’].tên\_trường*

Ví dụ



**Hàm haslayer()**

Cú pháp: *packet.haslayer(‘tên\_layer’)*

Kiểm tra xem gói tin kiểm tra xem gói tin có layer được truyển vào hay không

**Hàm wpcap()**

Cú pháp: *wpcap(‘tên\_file.pcap’,packet\_list)*

Hàm wpcap() lưu những gói tin trong packet\_list vào một file mới.

**Hàm sniff()**

* Với tham số *iface=‘interface’* trả về danh sách gói tin qua interface truyền vào
* Với tham số *offline=‘path\_to\_file\_pcap’* trả về danh sách gói tin trong trong file
* Với tham số *filter=‘expression’* để lọc gói tin với theo biểu thức expression (dưa trên cú pháp giống Display Filter trong Wireshark)
* Với tham số *count=<số\_packet>* trả về số\_packet đầu tiên trong danh sách
* Với tham số *prn=<function>* truyền mỗi gói tin trong danh sách qua hàm truyền vào rồi trả về danh sách sau khi xử lý qua hàm.

**Log web**

**Log web là gì?**

Log web là các tệp ghi lại các hoạt động truy cập trang web. Các tệp log này thường được tạo ra bởi máy chủ web khi người dùng truy cập vào trang web, và chúng bao gồm thông tin về địa chỉ IP của người dùng, trang web được truy cập, thời gian truy cập, trình duyệt web được sử dụng và các thông tin khác liên quan đến hoạt động truy cập.

**Cấu trúc của một file log web**

Cấu trúc của log web có thể khác nhau tùy thuộc vào phần mềm máy chủ web được sử dụng. Tuy nhiên, thường thì các tệp log web sẽ có cấu trúc tương tự nhau, với mỗi dòng trong tệp log đại diện cho một yêu cầu truy cập trang web.

Các trường thông tin chính trong một tệp log web có thể bao gồm:

* + Địa chỉ IP của người dung
  + Ngày và giờ truy cập
  + Phương thức truy cập (GET, POST, PUT, DELETE, ...)
  + Đường dẫn tới tài nguyên được yêu cầu (URL)
  + Mã trạng thái HTTP (200, 404, 500, ...)
    - Dưới đây là thông tin của mã trạng thái HTTP.

|  |  |
| --- | --- |
| 1xx | Thông tin |
| 2xx | Thành công |
| 3xx | Redirection |
| 4xx | Lỗi máy khách |
| 5xx | Lỗi máy chủ |

* + Kích thước tệp được truy cập
  + Thông tin về trình duyệt và hệ điều hành được sử dụng
  + Thông tin về ngôn ngữ và miền địa lý của người dung

Ngoài ra, tùy vào cấu hình của máy chủ web, tệp log web cũng có thể chứa các trường thông tin bổ sung như tên người dùng và mật khẩu được sử dụng để xác thực người dùng, hoặc các thông tin về tài khoản đăng nhập của người dùng. Tuy nhiên, để đảm bảo tính riêng tư của người dùng, các thông tin nhạy cảm này thường sẽ được mã hóa hoặc che đi trong các tệp log.

Ví dụ một dòng trong log web:

192.168.1.1 - - [10/Mar/2022:09:47:23 -0700] "GET /index.html HTTP/1.1" 200 2326 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"

Trong đó:

* ***192.168.1.1*:** Địa chỉ IP của người dùng
* **‘-‘** :Tên người dùng (nếu được sử dụng)
* **‘-‘** :ID người dùng (nếu được sử dụng)
* ***[10/Mar/2022:09:47:23 -0700]*:** Ngày và giờ truy cập
* ***"GET /index.html HTTP/1.1"*:** Phương thức truy cập và đường dẫn tới tài nguyên được yêu cầu
* **200:** Mã trạng thái HTTP trả về (trong trường hợp này là thành công)
* **2326:** Kích thước của tệp được truy cập
* **"-"**: Thông tin về trang web hoặc ứng dụng được sử dụng (nếu có)
* **"*Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36":*** Thông tin về trình duyệt và hệ điều hành được sử dụng.

**Phát hiện điểm đáng nghi**

Có thể sử dụng các công cụ và kỹ thuật khác nhau để phát hiện các điểm nghi ngờ trong web logs. Sau đây là một số phương pháp thông dụng:

* Xem xét các địa chỉ IP không được xác thực hoặc không phải là người dùng quen thuộc: Các địa chỉ IP không xác thực hoặc không phải là người dùng thường xuyên truy cập có thể là dấu hiệu của các hành vi đáng ngờ hoặc tấn công từ máy tính hoặc khu vực mạng không an toàn.
* Phát hiện các yêu cầu không hợp lệ: Các yêu cầu không hợp lệ có thể là dấu hiệu của các tấn công từ xa, chẳng hạn như các cuộc tấn công SQL injection hoặc cross-site scripting (XSS).
* Kiểm tra tần suất truy cập và thời gian truy cập: Các yêu cầu truy cập tần suất cao hoặc truy cập vào các trang web vào thời gian không thường có thể là dấu hiệu của các hành vi tấn công hoặc kiểm tra khả năng xâm nhập.
* Kiểm tra kích thước và kiểu tệp: Các tệp có kích thước lớn hoặc không xác định có thể bao gồm mã độc hoặc các dữ liệu không mong muốn.

**Write up**

**Bài Netlab2:**

Như thói quen, mở file ra thì đầu tiên em check TCP steam và Export HTTP ra thì em được một file update.sh. Nhìn vào file thì thấy file có chức năng là nén những file bên trong thư mục hiện tại và truy vấn lên DNS tại địa chỉ **10.2.32.72**

Mở file netlab2.pcap và lọc **‘dns && ip.dst == 10.2.32.72’** để lấy những gói tin dns truy vấn tới địa chỉ **10.2.32.72** và lưu thành một file khác

Sử dụng thư viện Scapy để dịch ngược dự liệu:

import base58

import base64

from scapy.all import \*

packets = rdpcap(r"C:\Users\ASUS\Desktop\capture.pcap")

res =[]

for p in packets:

    string = (str(p[DNSQR].qname)[2:-2]).replace('}','+').replace('{','/')

    res.append(string.split('-.'))

name\_file = []

data\_file = []

for i in res:

    name\_file.append(i[len(i)-1])

    str =''

    for j in range(len(i)-1):

        str=str+i[j]

    data\_file.append(str)

for i in range(len(name\_file)):

    name\_file[i]=name\_file[i].strip()

    name = base58.b58decode(name\_file[i]).decode('utf-8')

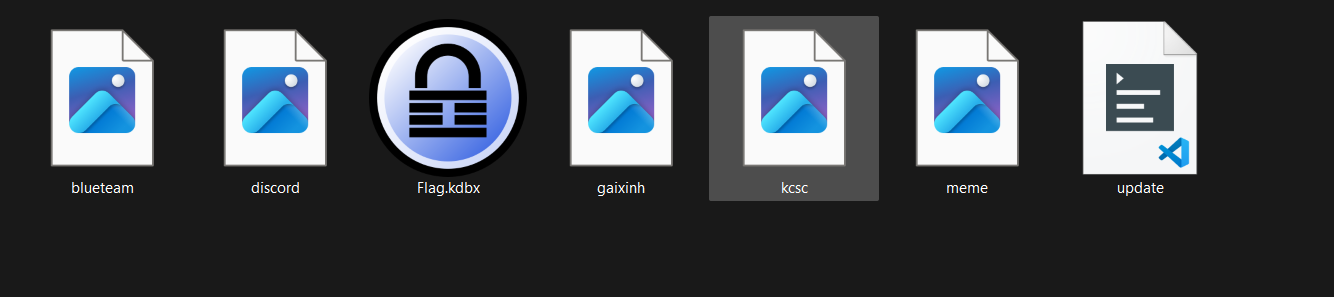
    name = name.strip()

    f = open(name,'a')

    f.write(data\_file[i])

    f.close()

Sau khi chạy chương trình ta được nhưng file sau:



Decode base64 và giải nén file Flag.kdbx được một file Flag.kbdx

Sử dụng tool **Keepass2john** và **John** crack, tìm được mật khẩu là ‘iloveyou’. Mở file ra ta được flag : Flag{NetLab2\_DNS\_3xf1ltr4t10n\_15\_5t3al7hy}

**Netlab5:**

Xem qua file, thì em thấy những gói tin được gửi từ địa chỉ ‘192.52.12.51’ đến ‘192.168.134.52’ có byte TCP payload có nghĩa, nên sử dụng scapy để gộp những byte đó lại:

from scapy.all import \*

packets = rdpcap(r"C:\Users\ASUS\Downloads\netlab5\netlab5.pcap")

for p in packets:

    if p.haslayer('IP'):

        if (p['IP'].proto == 6) and (p['IP'].src == '192.52.12.51') and (p['IP'].dst == '192.168.134.52'):

            print(str(p['Raw'].load)[2:-1],end='')

Sau khi chạy , em được đoạn văn bản sau:

I miss the days when life was so simple Felt like the glass was always half full Where did that go? And every second with you was so special Back when we didn't fear the unknowns But that was long ago Who can say where the path will go? Philosophers guess but they just don't know Maybe that's why We had our head in the clouds Thought we had it all figured out Planning to fly away To escape everything on the ground But like a plane up in space We slowly drifted away And every plan that we made And dream that we chased Are just memories now They're just memories now I'm not sure where everything went wrong But I know that we landed where we both belong (Where we both belong) I just wish we weren't scared to say That there's expiration dates on the friends you make As hard as that may sound Who can say where the path will go? Philosophers guess but they just don't know Maybe that's why We had our head in the clouds Thought we had it all figured out Planning to fly away To escape everything on the ground But like a plane up in space We slowly drifted away And every plan that we made And dream that we chased Are just memories now They're just memories now Who can say where the path will go? (Where the path will go? Where the path will go?) Philosophers guess but they just don't know (But they just don't know)|||||||||||||TCP SYN flood (a.k.a. SYN flood) is a type of Distributed Denial of Service (DDoS) attack that exploits part of the normal TCP three-way handshake to consume resources on YWxlcnQoJ0ZsYWd7TmV0TGFiNV9TWU5fRkwwb2RfYjRuTjNkfScp the targeted server and render it unresponsive.  Essentially, with SYN flood DDoS, the offender sends TCP connection requests faster than the targeted machine can process them, causing network saturation.

Decode base64 chuỗi **‘YWxlcnQoJ0ZsYWd7TmV0TGFiNV9TWU5fRkwwb2RfYjRuTjNkfScp’** được flag: Flag{NetLab5\_SYN\_FL0od\_b4nN3d}