

# Lab-10 Using-Wireshark---analyzing-web-browser-artifacts-email-header-analysis

---

## AIM:

---

To use Wireshark to analyze web browser activities and inspect email headers from captured network traffic.

## DESIGN STEPS:

---

### Step 1:

Launch Wireshark and start capturing traffic on the appropriate network interface.

### Step 2:

Use filters like http, dns, or tcp.port == 80 to monitor web browser artifacts such as visited URLs, cookies, and user-agent strings.

### Step 3:

Apply filters like smtp, pop, or imap to locate and analyze email header details (e.g., sender, receiver, subject) from email communications.

## PROGRAM:

---

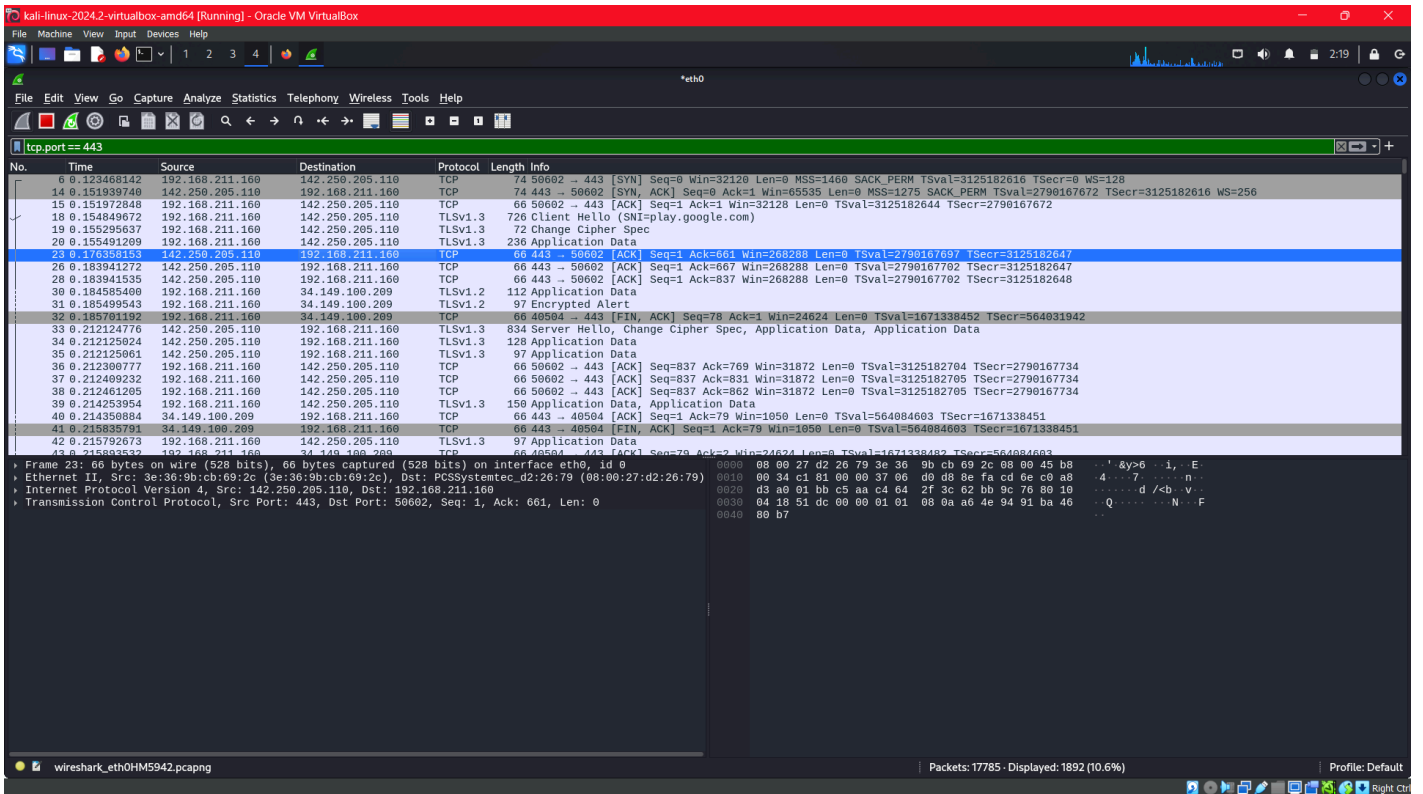
Wireshark Web and Email Traffic Filtering Steps

## OUTPUT:

---

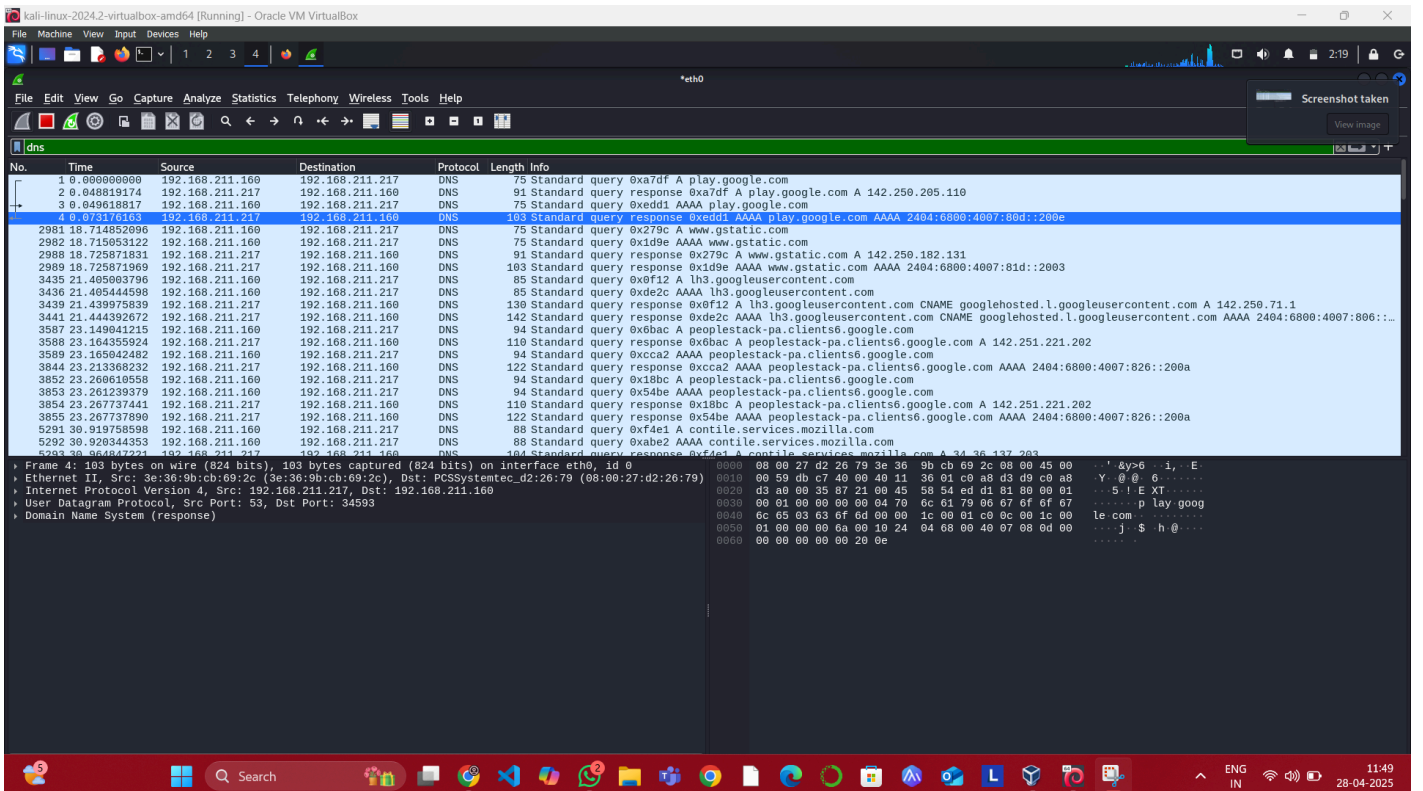
### A. Capturing Traffic in Wireshark

1. Open Wireshark and start capturing on the active interface (Wi-Fi/Ethernet).
2. Perform activities like opening a website or sending an email through a client (e.g., Gmail via browser or Thunderbird).
3. Stop the capture once done.



Analyze DNS Queries: o Filter: dns

o Reveal domains the browser tried to resolve.



## Email Header Analysis

1. Apply relevant filters:
- 2.

o For POP3: tcp.port == 110

o For SMTP: tcp.port == 25 or 587

o For IMAP: tcp.port == 143 or 993

4. Locate email data:

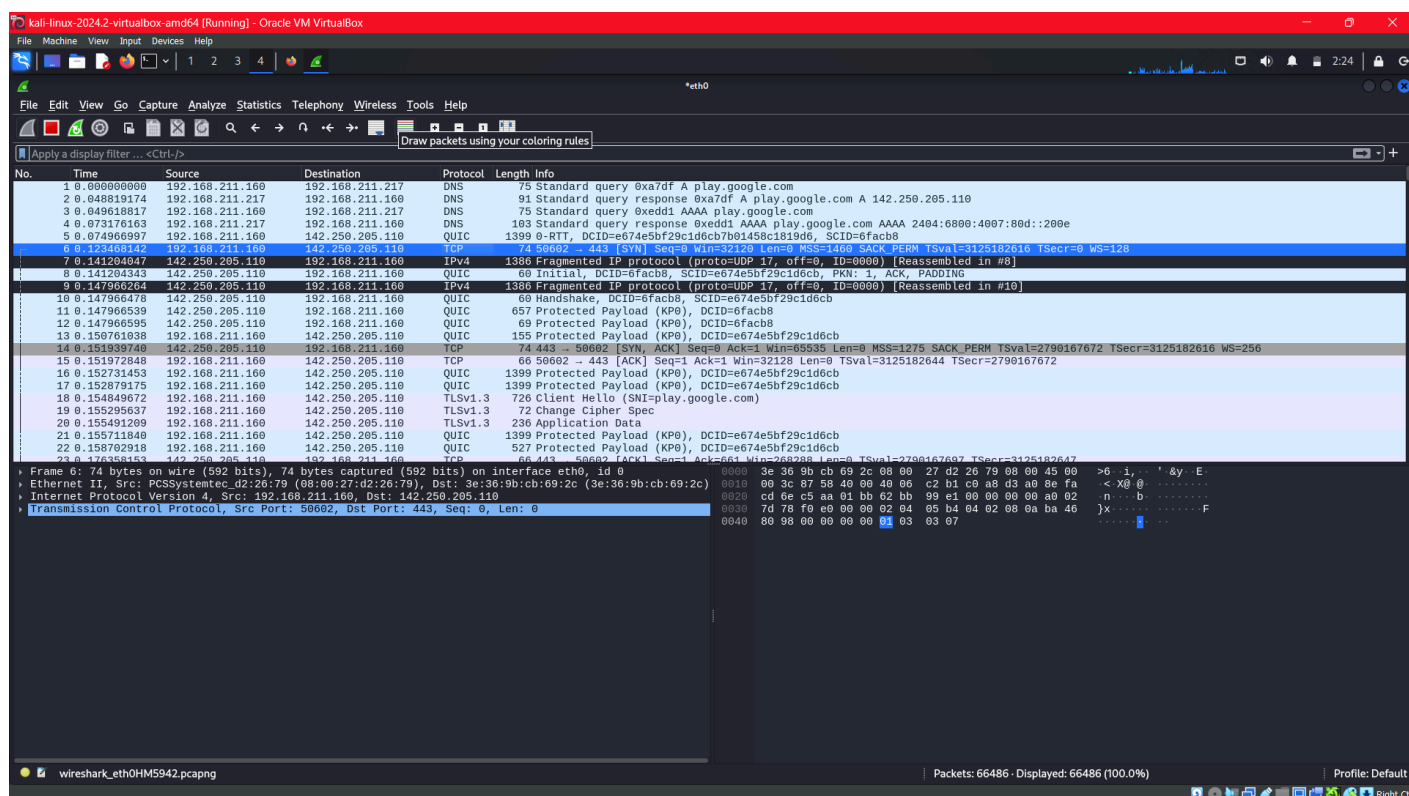
5.

o Look for SMTP packets to see sender/receiver email addresses.

o Use "Follow TCP Stream" to view the full email headers and body if unencrypted.

## Extract Email Header Fields:

o Analyze From, To, Subject, Date, Message-ID, and relay servers used in sending the email.



```

> Frame 6: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface eth0, id 0
> Ethernet II, Src: PCSSystemtec_d2:26:79 (08:00:27:d2:26:79), Dst: 3e:36:9b:cb:69:2c (3e:36:9b:cb:69:2c)
> Internet Protocol Version 4, Src: 192.168.211.160, Dst: 142.250.205.110
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 60
  Identification: 0x8758 (34648)
> 010. .... = Flags: 0x2, Don't fragment
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 64
  Protocol: TCP (6)
  Header Checksum: 0xc2b1 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 192.168.211.160
  Destination Address: 142.250.205.110
> Transmission Control Protocol, Src Port: 50602, Dst Port: 443, Seq: 0, Len: 0
  Source Port: 50602
  Destination Port: 443
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  ..0. .... = RST: Absent
  ...1 .... = FIN: Present
  1 .... = Data: Present

```

```

Destination Address: 142.250.205.110
Transmission Control Protocol, Src Port: 50602, Dst Port: 443, Seq: 0, Len: 0
Source Port: 50602
Destination Port: 443
[Stream index: 0]
[Conversation completeness: Complete, WITH_DATA (31)]
..0. .... = RST: Absent
...1 .... = FIN: Present
.... 1... = Data: Present
.... .1.. = ACK: Present
.... ..1. = SYN-ACK: Present
.... ...1 = SYN: Present
[Completeness Flags: .FDASS]
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 1656461793
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1010 .... = Header Length: 40 bytes (10)
Flags: 0x002 (SYN)
000. .... = Reserved: Not set
...0 .... = Accurate ECN: Not set

[Calculated window size: 32120]
Checksum: 0xf0e0 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window s
  TCP Option - Maximum segment size: 1460 bytes
    Kind: Maximum Segment Size (2)
    Length: 4
    MSS Value: 1460
  TCP Option - SACK permitted
    Kind: SACK Permitted (4)
    Length: 2
  TCP Option - Timestamps: TSval 3125182616, TSecr 0
  TCP Option - No-Operation (NOP)
    Kind: No-Operation (1)
  TCP Option - Window scale: 7 (multiply by 128)
    Kind: Window Scale (3)
    Length: 3
    Shift count: 7
    [Multiplier: 128]
[Timestamps]
[Time since first frame in this TCP stream: 0.000000000 seconds]
[Time since previous frame in this TCP stream: 0.000000000 seconds]

```

## RESULT:

Web browser artifacts and email headers were successfully analyzed using Wireshark