



# Lab Practical #09:

Study Packet capture and header analysis by Wireshark (HTTP, TCP, UDP, IP, etc.)

## Practical Assignment #09:

### Explain usage of Wireshark tool.

**Wireshark** is one of the most popular open-source network protocol analyzers. It captures real-time network traffic and displays it in detail, allowing administrators, developers, and students to study how data flows across the network.

#### Primary Uses of Wireshark:

##### 1. Network Troubleshooting

- Helps diagnose common network issues such as high latency, dropped packets, or connectivity failures.
- Can detect misconfigurations in routing or protocol setups.

##### 2. Protocol Analysis

- Provides detailed inspection of how various protocols (e.g., TCP, UDP, ICMP, HTTP, DNS) function.
- Helps verify whether devices are communicating correctly according to standards.

##### 3. Security Monitoring

- Useful in spotting suspicious or malicious activities (e.g., packet sniffing, man-in-the-middle attacks).
- Aids in detecting unauthorized access attempts and data leaks.

##### 4. Performance Measurement

- Monitors bandwidth usage and data flow patterns.
- Identifies bottlenecks affecting network performance.

##### 5. Application Debugging

- Developers can analyze how their applications exchange data with servers.
- Helps track down bugs in client-server communication.

##### 6. Educational Purposes

- A valuable tool for students and professionals to understand networking protocols and packet structures.

Date: 9/3/2025

**Basic Steps to Use Wireshark:**

**1. Install Wireshark**

- Available on Windows, macOS, and Linux platforms.
- Requires administrative rights for capturing live network data.

**2. Start Capturing Packets**

- Select the correct interface (Ethernet, Wi-Fi, etc.).
- Click the Start button to begin recording network traffic.

**3. Apply Filters**

- Select the correct interface (Ethernet, Wi-Fi, etc.).
- Click the Start button to begin recording network traffic.

**4. Analyze Packets**

- Every packet can be viewed in three panes:
  1. Packet List (summary)
  2. Packet Details (protocol hierarchy)
  3. Packet Bytes (raw data view)
- Information includes IP addresses, ports, flags, and payload.

**5. Save and Export Data**

- Packet captures can be stored as .pcap files for further analysis or sharing.

**6. Use Built-in Tools**

- View statistics like **Protocol Hierarchy**, **Conversations**, and **I/O Graphs**.
- Use **Follow TCP Stream** to see the entire conversation between two endpoints.

**Example Use Case:**

Suppose a website is loading unusually slowly:

1. Start capturing traffic while accessing the website.
2. Apply a filter such as http to focus on web requests.
3. Look for delays in TCP handshakes or slow response times from the server.
4. Examine headers and response codes to find possible misconfigurations.



## Packet capture and header analysis by Wireshark (HTTP, TCP, UDP, IP, etc.)

The screenshot displays two windows side-by-side. The left window is Wireshark, capturing traffic on the 'dns' interface. The packet list shows several DNS queries and responses. The packet details pane for packet 159 (Frame 159) is expanded, showing the 'Domain Name System (query)' section. The query is for 'google.com' with transaction ID 0x0001. The right window is a Windows Command Prompt showing the output of the 'nslookup google.com' command. The output indicates the server is unknown and the address is 10.20.1.1. It also shows a non-authoritative answer for 'google.com' with addresses 2404:6800:4009:800::200e and 142.250.207.142.

No.	Time	Source	Destination	Protocol	Length	Info
159	3.364045	10.20.43.13	10.20.1.1	DNS	82	Standard query 0x0001 PTR 1.1.20.10.in-addr.arpa
163	3.388059	10.20.1.1	10.20.43.13	DNS	82	Standard query response 0x0001 No such name PTR 1.1.20.10.in-addr.arpa
164	3.390192	10.20.43.13	10.20.1.1	DNS	70	Standard query 0x0002 A google.com
165	3.393818	10.20.1.1	10.20.43.13	DNS	86	Standard query response 0x0002 A google.com A 142.250.207.142
166	3.396477	10.20.43.13	10.20.1.1	DNS	70	Standard query 0x0003 AAAA google.com
167	3.398736	10.20.1.1	10.20.43.13	DNS	98	Standard query response 0x0003 AAAA google.com AAAA 2404:6800:4009:800::200e 142.250.207.142

```

D:\>nslookup google.com
Server: Unknown
Address: 10.20.1.1

Non-authoritative answer:
Name: google.com
Addresses: 2404:6800:4009:800::200e
          142.250.207.142

D:\>
  
```

The image displays a network traffic analysis using Wireshark and the corresponding web application interface.

**Wireshark Packet Capture:**

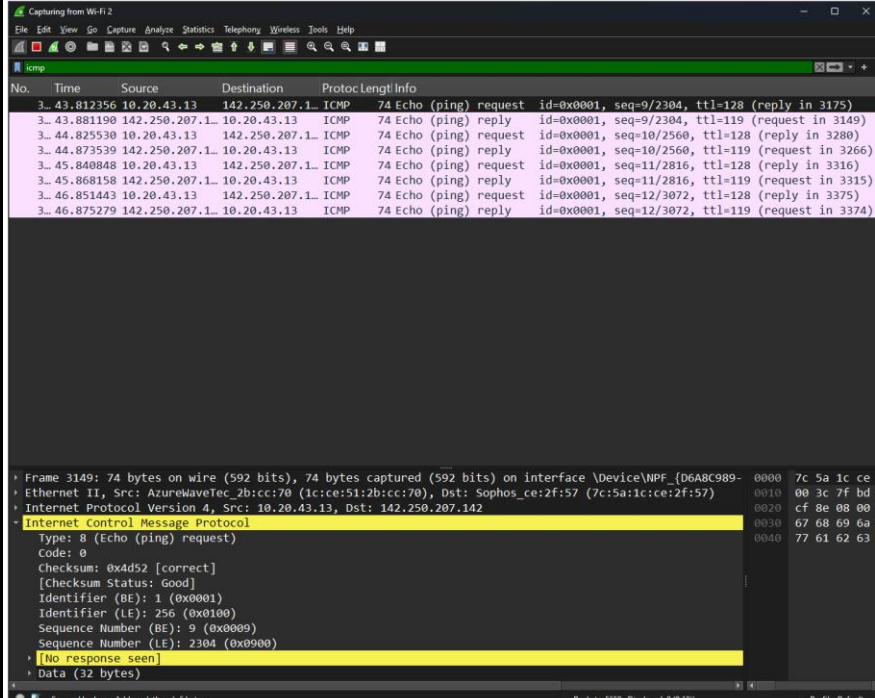
- Packet List:** Shows a series of HTTP requests. Packet 1228 is highlighted, showing a POST request to `/login.xml` with a content type of `application/x-www-form-urlencoded`.
- Packet Details:** For packet 1228, the details pane shows:
  - Frame 16240: 550 bytes on wire (4400 bits), 550 bytes captured (4400 bits) on interface `\Device\NPF_{D6A8...}`
  - Ethernet II, Src: AzureWaveTec\_2b:cc:70 (1c:ce:51:2b:cc:70), Dst: Sophos\_ce:2f:57 (7c:5a:1c:ce:2f:57)
  - Internet Protocol Version 4, Src: 10.20.43.13, Dst: 10.255.1.1
  - Transmission Control Protocol, Src Port: 54075, Dst Port: 8090, Seq: 1, Ack: 1, Len: 496
  - Hypertext Transfer Protocol
  - HTML Form URL Encoded: `application/x-www-form-urlencoded`
    - Form item: `"mode" = "191"`
    - Form item: `"username" = "23010101121"`
    - Form item: `"password" = "Kakadiya@123"`
    - Form item: `"a" = "1753930741680"`
    - Form item: `"producttype" = "0"`
- Packet Bytes:** The raw data is shown in hexadecimal and ASCII. The ASCII portion shows the URL-encoded form data: `mode=191&username=23010101121&password=Kakadiya%40123&a=1753930741680&producttype=0`.

**Sophos Web Interface:**

- The browser shows the Sophos login page.
- The message states: "You are signed in as 23010101121".
- A green checkmark icon indicates successful login.
- A warning message says: "Do not close this page. If you do, you will be signed out".
- A "Sign out" button is visible.
- A link to "Access the User Portal" is provided.

Date: 9/3/2025

### 3. ICMP :



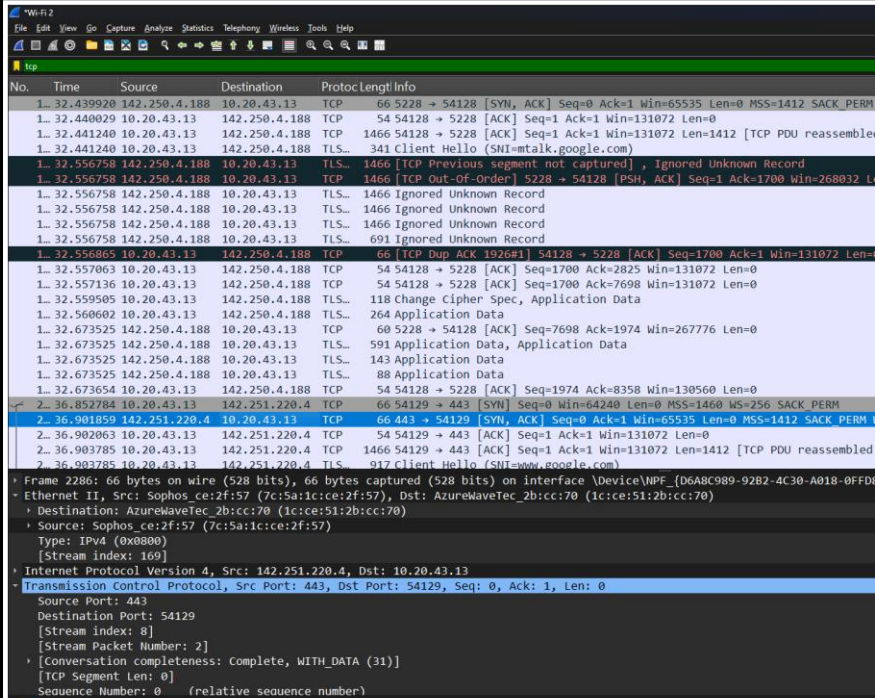
```
D:\>ping google.com

Pinging google.com [142.250.207.142]
with 32 bytes of data:
Reply from 142.250.207.142: bytes=32
time=68ms TTL=119
Reply from 142.250.207.142: bytes=32
time=48ms TTL=119
Reply from 142.250.207.142: bytes=32
time=27ms TTL=119
Reply from 142.250.207.142: bytes=32
time=23ms TTL=119

Ping statistics for 142.250.207.142:
    Packets: Sent = 4, Received = 4,
    Lost = 0 (0% loss),
    Approximate round trip times in milli-
seconds:
        Minimum = 23ms, Maximum = 68ms, A
    verage = 41ms

D:\>
```

### 4. TCP :



```
0000  1c ce 51 2b cc 70 7c 5
0010  00 34 00 00 40 7c 0
0020  2b 0d 01 bb d3 71 df 3
0030  ff ff cd be 00 00 02 0
0040  03 08
```





Date: 9/3/2025

## 5. UDP :

Wireshark packet capture showing UDP traffic. The packet list shows a UDP packet from 10.20.64.94 to 239.255.102.18 on port 51035. The packet details show the UDP header and payload. The packet bytes show the raw data.

No.	Time	Source	Destination	Protocol	Length	Info
2...	36.827192	10.20.43.13	10.20.1.1	DNS	74	Standard query 0x7613 A www.google.com
2...	36.827434	10.20.43.13	10.20.1.1	DNS	74	Standard query 0x6c3e HTTPS www.google.com
2...	36.850937	10.20.1.1	10.20.43.13	DNS	90	Standard query response 0x7613 A www.google.com A 142.251.220.4
2...	36.851224	10.20.1.1	10.20.43.13	DNS	99	Standard query response 0x6c3e HTTPS www.google.com HTTPS
2...	37.070046	fe80::2477:fa...	ff02::1:3	LLMNR	88	Standard query 0x9d8b AAAA 05_17_18
2...	37.071529	fe80::2477:fa...	ff02::1:3	LLMNR	88	Standard query 0x7a82 A 05_17_18
2...	37.071529	10.20.27.18	224.0.0.252	LLMNR	68	Standard query 0x9d8b AAAA 05_17_18
2...	37.071529	10.20.27.18	224.0.0.252	LLMNR	68	Standard query 0x7a82 A 05_17_18
2...	37.071529	fe80::4def:cf...	ff02::1:2	DHCP	150	Solicit XID: 0x5c972e CID: 000100012ffe599b7427ea466d11
2...	37.071529	fe80::f05d:20...	ff02::1:3	LLMNR	88	Standard query 0xf1d6 A 05_18_12
2...	37.071529	10.20.27.33	224.0.0.252	LLMNR	68	Standard query 0xf1d6 A 05_18_12
2...	37.072179	fe80::f05d:20...	ff02::1:3	LLMNR	88	Standard query 0x8b62 AAAA 05_18_12
2...	37.072179	10.20.27.33	224.0.0.252	LLMNR	68	Standard query 0x8b62 AAAA 05_18_12
2...	37.285022	10.20.64.94	239.255.102.18	UDP	1239	51035 → 50001 Len=5637
2...	37.291813	10.20.64.94	239.255.102.18	UDP	1239	51036 → 50002 Len=5637
2...	37.303123	10.20.64.94	239.255.102.18	UDP	1239	51037 → 50003 Len=5637
2...	37.304517	10.20.60.23	255.255.255.2	UDP	937	53142 → 29810 Len=895
2...	37.479235	fe80::f05d:20...	ff02::1:3	LLMNR	88	Standard query 0xf1d6 A 05_18_12
2...	37.479235	fe80::f05d:20...	ff02::1:3	LLMNR	88	Standard query 0x8b62 AAAA 05_18_12
2...	37.479235	10.20.27.33	224.0.0.252	LLMNR	68	Standard query 0x8b62 AAAA 05_18_12
2...	37.479235	10.20.27.33	224.0.0.252	LLMNR	68	Standard query 0xf1d6 A 05_18_12
2...	37.786521	10.20.62.99	230.0.0.1	UDP	92	60715 → 6666 Len=50
2...	38.094843	10.20.61.102	224.0.0.251	MDNS	234	Standard query response 0x0000 PTR DESKTOP-P7SHEQ4._dosvc._tcp.local SRV 0 0 7680 DESKTOP-P7SHEQ4.local TXT
2...	38.095580	fe80::e10f:12...	ff02::fb	MDNS	254	Standard query response 0x0000 PTR DESKTOP-P7SHEQ4._dosvc._tcp.local SRV 0 0 7680 DESKTOP-P7SHEQ4.local TXT
2...	38.095580	10.20.61.102	224.0.0.251	MDNS	93	Standard query 0x0000 ANY DESKTOP-P7SHEQ4._dosvc._tcp.local. "P7SHEQ4"

Frame 2316: 1239 bytes on wire (9912 bits), 1239 bytes captured (9912 bits) on interface \Device\NPF\_{D6A0C989-92B2-4C30-A018-0FFD8ECF2EE6}, id 0  
Ethernet II, Src: AzureWaveTec 95:abcde (48:e7:da:95:abcde), Dst: IPv4mcast\_7f:66:12 (01:00:5e:7f:66:12)  
Internet Protocol Version 4, Src: 10.20.64.94, Dst: 239.255.102.18  
User Datagram Protocol, Src Port: 51035, Dst Port: 50001  
Source Port: 51035  
Destination Port: 50001  
Length: 5645  
Checksum: 0x6719 [unverified]  
[Checksum Status: Unverified]  
[Stream index: 351]  
[Stream Packet Number: 1]  
[Timestamps]  
UDP payload (5637 bytes)  
Data (5637 bytes)  
Data [1]: 42d40a86d0d4fb42130000001000015f459e9000399fb3669d34803140eaa7d3b7e06fb80a8ed2901781c23269c4195ce90d5571750fa7237cde542b1224a0d467cef809be7...