Task – 5 :

Network Traffic Capture and Analysis Using Wireshark

Objective

The objective of this lab is to capture live network traffic using Wireshark on Kali Linux, filter and identify various network protocols, and analyze the traffic to understand communication between network devices.

Tools and Environment

- **Wireshark** – GUI-based network protocol analyzer
- **Kali Linux** – Security-focused Linux distribution
- **Firefox** – Used to generate HTTP traffic
- **Terminal (Bash)** – Used to generate ICMP traffic (ping)

Procedure

1. Install and Launch Wireshark

Wireshark was installed (if not already) and launched using the following command: sudo wireshark

Wireshark was granted permission to capture traffic on the active interface.

2. Identify and Select Network Interface

Used:

```
ip a
```

to identify the active interface (eth0 in this case). Selected this interface in Wireshark for live packet capture.

3. Start Packet Capture

Live capture was started by clicking the capture icon in Wireshark on the selected interface.

4. Generate Network Traffic Traffic was manually generated using:

- **Ping (ICMP):**

```
ping google.com
```

- **Web browsing (HTTP/DNS):**
  - Visited https://example.com via Firefox to generate DNS and HTTP traffic.

---

5. Stop Capture

After ~60 seconds, packet capture was stopped.

---

6. Apply Protocol Filters

Applied filters to identify and analyze:

- DNS traffic: dns
- HTTP traffic: http
- ICMP traffic: icmp
- TCP handshakes and sessions: tcp

Analysis and Findings

1. DNS (Domain Name System)

- **Port:** UDP 53
- **Observation:** DNS query for example.com and google.com.
- **Packet Example:**

less

```
Standard query 0x37d1 A google.com

Response: A record for google.com -> 142.250.190.78
```

---

2. HTTP (HyperText Transfer Protocol)

- **Port:** TCP 80
- **Observation:** HTTP GET request when accessing example.com.
- **Packet Example:**

makefile

```
GET / HTTP/1.1

Host: example.com

User-Agent: Mozilla/5.0 (Linux x86_64; rv:89.0)
```

---

3. ICMP (Internet Control Message Protocol)

- **No Port** (network layer protocol)
- **Observation:** Echo requests and replies using the ping command.
- **Packet Example:**

Echo (ping) request id=0x01 seq=1

Echo (ping) reply id=0x01 seq=1

---

4. TCP (Transmission Control Protocol)

- **Observation:** TCP 3-way handshake observed prior to HTTP data exchange.

- **Packet Sequence:** arduino

1. SYN (Client -> Server)
2. SYN-ACK (Server -> Client)
3. ACK (Client -> Server)

Capture File

- File Name: network_traffic.pcap
- Size: ~500 KB (varies based on traffic)
- Protocols Included: DNS, HTTP, TCP, ICMP

Key Takeaways

- Wireshark effectively captures and dissects network communication.
- DNS requests precede HTTP traffic as domain names are resolved.
- HTTP GET requests clearly show human-readable URLs and headers.
- ICMP packets are useful for diagnostic purposes (reachability and latency).
- TCP ensures reliable communication through handshakes and ACKs.

Conclusion

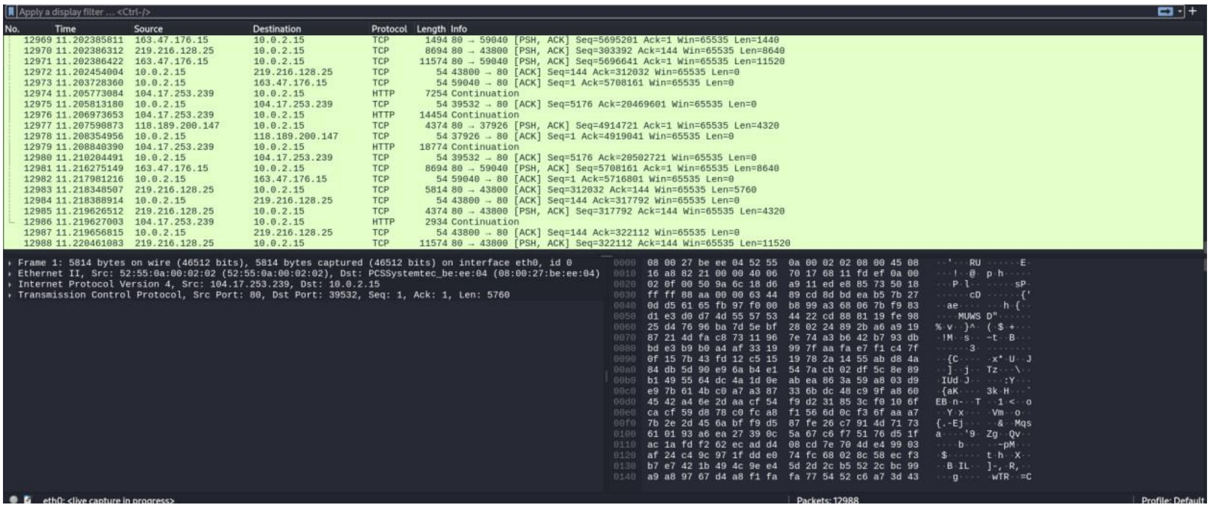
This activity demonstrated how to:

- Use Wireshark to capture real-time network traffic
- Apply filters to isolate protocol-specific packets
- Interpret packet contents and understand network behavior

- Identify and analyze at least 3 core Internet protocols

References

- Wireshark Official Documentation: https://www.wireshark.org/docs/
- Protocol Port Numbers: IANA Service Name and Port Number Registry



```
No.   Time          Source            Destination       Protocol Length Info
12969 11.202385811  163.47.176.15     10.0.2.15         TCP       1494 80 → 59040 [PSH, ACK] Seq=5695201 Ack=1 Win=65535 Len=1440
12970 11.202386312  219.216.128.25    10.0.2.15         TCP       8694 80 → 43800 [PSH, ACK] Seq=303392 Ack=144 Win=65535 Len=8640
12971 11.202386422  163.47.176.15     10.0.2.15         TCP      11574 80 → 59040 [PSH, ACK] Seq=5696641 Ack=1 Win=65535 Len=11520
12972 11.202454004  10.0.2.15         219.216.128.25    TCP         54 43800 → 80 [ACK] Seq=144 Ack=312032 Win=65535 Len=0
12973 11.203728360  10.0.2.15         163.47.176.15     TCP         54 59040 → 80 [ACK] Seq=1 Ack=5708161 Win=65535 Len=0
12974 11.205773084  104.17.253.239    10.0.2.15         HTTP      7254 Continuation
12975 11.205813180  10.0.2.15         104.17.253.239    TCP         54 39532 → 80 [ACK] Seq=5176 Ack=20469601 Win=65535 Len=0
12976 11.206973653  104.17.253.239    10.0.2.15         HTTP     14454 Continuation
12977 11.207590873  118.189.200.147   10.0.2.15         TCP       4374 80 → 37926 [PSH, ACK] Seq=4914721 Ack=1 Win=65535 Len=4320
12978 11.208354956  10.0.2.15         118.189.200.147   TCP         54 37926 → 80 [ACK] Seq=1 Ack=4919041 Win=65535 Len=0
12979 11.208840390  104.17.253.239    10.0.2.15         HTTP     18774 Continuation
12980 11.210204491  10.0.2.15         104.17.253.239    TCP         54 39532 → 80 [ACK] Seq=5176 Ack=20502721 Win=65535 Len=0
12981 11.216275149  163.47.176.15     10.0.2.15         TCP       8694 80 → 59040 [PSH, ACK] Seq=5708161 Ack=1 Win=65535 Len=8640
12982 11.217981216  10.0.2.15         163.47.176.15     TCP         54 59040 → 80 [ACK] Seq=1 Ack=5716801 Win=65535 Len=0
12983 11.218348507  219.216.128.25    10.0.2.15         TCP       5814 80 → 43800 [ACK] Seq=312032 Ack=144 Win=65535 Len=5760
12984 11.218388914  10.0.2.15         219.216.128.25    TCP         54 43800 → 80 [ACK] Seq=144 Ack=317792 Win=65535 Len=0
12985 11.219626512  219.216.128.25    10.0.2.15         TCP       4374 80 → 43800 [PSH, ACK] Seq=317792 Ack=144 Win=65535 Len=4320
12986 11.219627003  104.17.253.239    10.0.2.15         HTTP      2934 Continuation
12987 11.219656815  10.0.2.15         219.216.128.25    TCP         54 43800 → 80 [ACK] Seq=144 Ack=322112 Win=65535 Len=0
12988 11.220461083  219.216.128.25    10.0.2.15         TCP      11574 80 → 43800 [PSH, ACK] Seq=322112 Ack=144 Win=65535 Len=11520
```

```
› Frame 1: 5814 bytes on wire (46512 bits), 5814 bytes captured (46512 bits) on interface eth0, id 0
› Ethernet II, Src: 52:55:0a:00:02:02 (52:55:0a:00:02:02), Dst: PCSSystemtec_be:ee:04 (08:00:27:be:ee:04)
› Internet Protocol Version 4, Src: 104.17.253.239, Dst: 10.0.2.15
› Transmission Control Protocol, Src Port: 80, Dst Port: 39532, Seq: 1, Ack: 1, Len: 5760
```

```
0000  08 00 27 be ee 04 52 55  0a 00 02 02 08 00 45 08   ··'···RU ······E·
0010  16 a8 82 21 00 00 40 06  70 17 68 11 fd ef 0a 00   ···!··@· p·h·····
0020  02 0f 00 50 9a 6c 18 d6  a9 11 ed e8 85 73 50 18   ···P·l·· ·····sP·
0030  ff ff 88 aa 00 00 63 44  89 cd 8d bd ea b5 7b 27   ······cD ······{'
0040  0d d5 61 65 fb 97 f0 00  b8 99 a3 68 06 7b f9 83   ··ae···· ···h·{··
0050  d1 e3 d0 d7 4d 55 57 53  44 22 cd 88 81 19 fe 98   ····MUWS D"······
0060  25 d4 76 96 ba 7d 5e bf  28 02 24 89 2b a6 a9 19   %·v··}^· (·$·+···
0070  87 21 4d fa c8 73 11 96  7e 74 a3 b6 42 b7 93 db   ·!M··s·· ~t··B···
0080  bd e3 b9 b0 a4 af 33 19  99 7f aa fa e7 f1 c4 7f   ······3· ········
0090  0f 15 7b 43 fd 12 c5 15  19 78 2a 14 55 ab d8 4a   ··{C···· ·x*·U··J
00a0  84 db 5d 90 e9 6a b4 e1  54 7a cb 02 df 5c 8e 89   ··]··j·· Tz···\··
00b0  b1 49 55 64 dc 4a 1d 0e  ab ea 86 3a 59 a8 03 d9   ·IUd·J·· ···:Y···
00c0  e9 7b 61 4b c0 a7 a3 87  33 6b dc 48 c9 9f a8 60   ·{aK···· 3k·H···`
00d0  45 42 a4 6e 2d aa cf 54  f9 d2 31 85 3c f0 10 6f   EB·n-··T ··1·<··o
00e0  ca cf 59 d8 78 c0 fc a8  f1 56 6d 0c f3 6f aa a7   ··Y·x··· ·Vm··o··
00f0  7b 2e 2d 45 6a bf f9 d5  87 fe 26 c7 91 4d 71 73   {.-Ej··· ··&··Mqs
0100  61 01 93 a6 ea 27 39 0c  5a 67 c6 f7 51 76 d5 1f   a····'9· Zg··Qv··
0110  ac 1a fd f2 62 ec ad d4  08 cd 7e 70 4d e4 99 03   ····b··· ··~pM···
0120  af 24 c4 9c 97 1f dd e0  74 fc 68 02 8c 58 ec f3   ·$······ t·h··X··
0130  b7 e7 42 1b 49 4c 9e e4  5d 2d 2c b5 52 2c bc 99   ··B·IL·· ]-,·R,··
0140  a9 a8 97 67 d4 a8 f1 fa  fa 77 54 52 c6 a7 3d 43   ···g···· ·wTR··=C
```