Network Traffic Analysis Using Wireshark: Capture, Inspect, and Interpret Data Packets

Sandra Sree Babu

Cyber Security Intern

Elevate Labs

September 2025

Contents

1 Task 5: Capture and Analyze Network Traffic Using Wireshark

1.1 Introduction

Network traffic analysis is an essential part of cybersecurity and digital forensics. It involves capturing, inspecting, and interpreting data packets transmitted over a network to understand communication patterns, detect anomalies, and identify potential security threats. Wireshark, a widely used open-source network protocol analyzer, allows users to capture live network traffic and analyze it in real time to study various protocols and packet structures.

1.2 Task Description

The purpose of this task is to capture and analyze live network packets using Wireshark on the Kali Linux environment. During the capture process, different types of traffic such as HTTP, DNS, TCP, and ICMP packets will be monitored. The captured data will then be saved as a packet capture file (.pcap) and analyzed to identify basic network protocols and traffic types present in the communication.

1.3 Objectives

- To capture live network traffic using Wireshark.
- To identify and analyze basic network protocols such as TCP, UDP, ICMP, DNS, and HTTP.
- To understand the structure of captured packets and their role in network communication.
- To generate and save a packet capture file for documentation and reporting purposes.

1.4 Tool Used

Wireshark — an open-source network packet analyzer that enables real-time capture and in-depth inspection of network traffic across multiple protocols. It provides a graphical interface for analyzing captured packets, making it a powerful tool for network troubleshooting, protocol analysis, and cybersecurity investigations.

2 Methodology

The following methodology was followed to capture and analyze live network traffic using Wireshark in the Kali Linux environment:

2.1 Packet Capture and Analysis using Wireshark

Steps Performed

1. Installed Wireshark on Kali Linux.
2. Started packet capture on the active network interface.
3. Browsed multiple websites to generate traffic.
4. Stopped the capture after one minute.
5. Filtered packets by protocol (e.g., HTTP, DNS, TCP).
6. Identified at least three different protocols observed in the capture.
7. Exported the captured packets as a `.pcap` file.

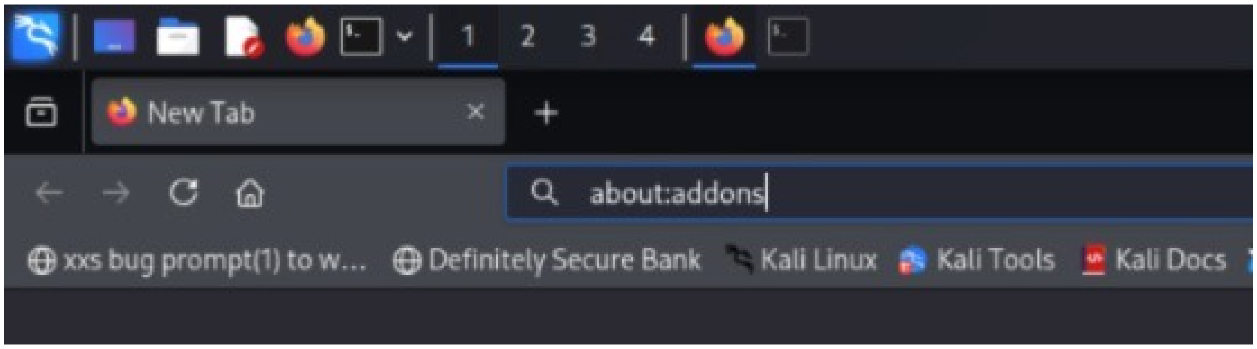
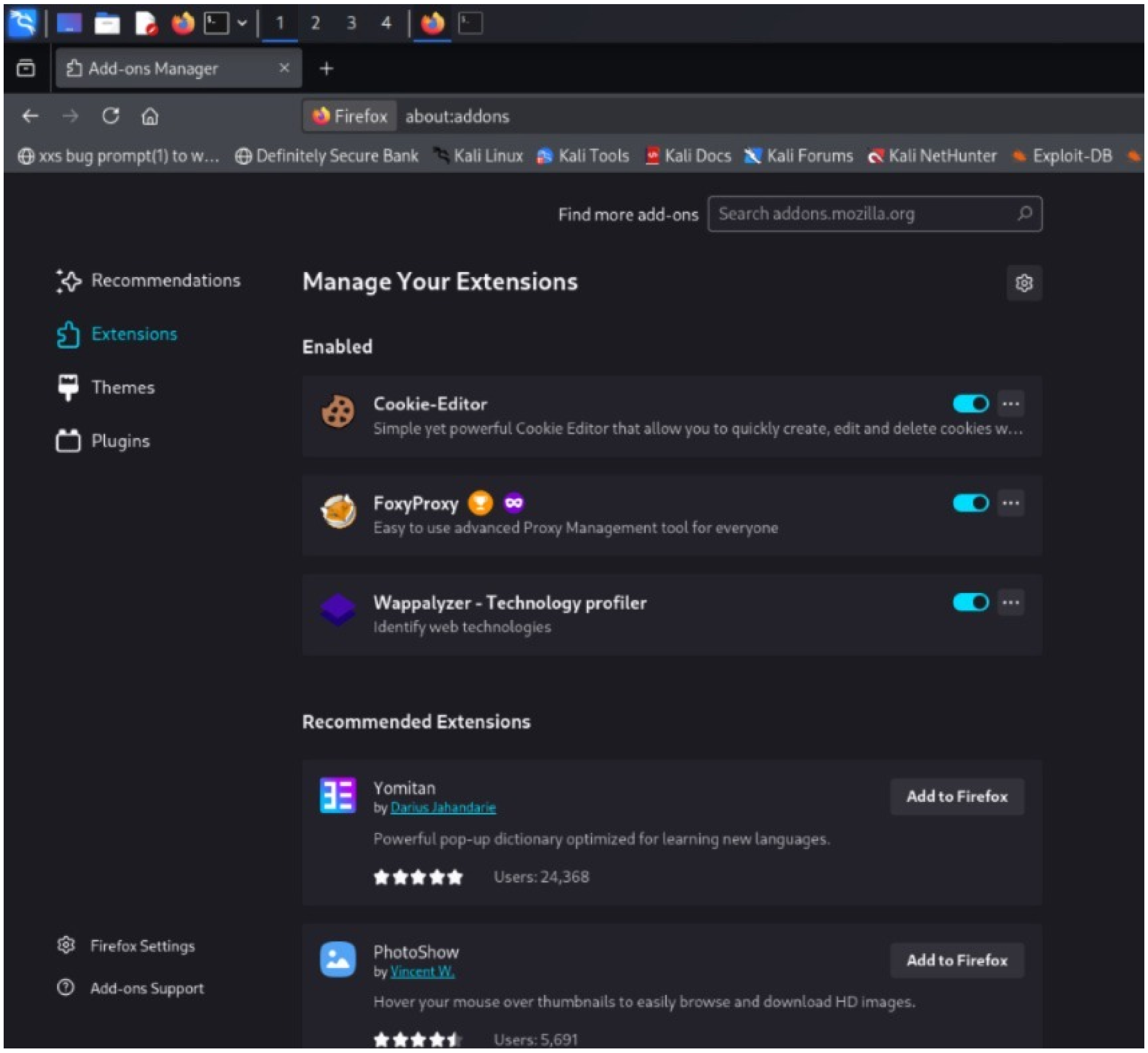


Figure 1: Opening the Firefox Add-ons page using `about:addons` to check installed extensions like Cookie Editor, FoxyProxy, and Wappalyzer.



Manage Your Extensions

Enabled

Cookie-Editor
Simple yet powerful Cookie Editor that allow you to quickly create, edit and delete cookies w...

FoxyProxy
Easy to use advanced Proxy Management tool for everyone

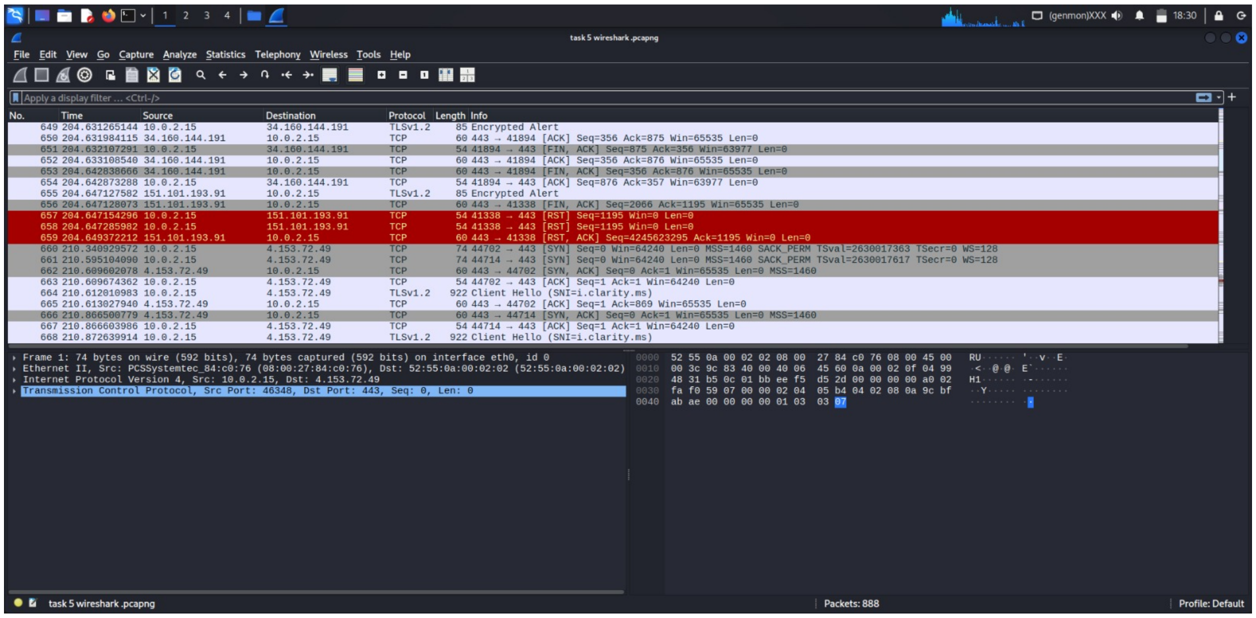
Wappalyzer - Technology profiler
Identify web technologies

Recommended Extensions

Yomitan
by Darius Jahandarie
Powerful pop-up dictionary optimized for learning new languages.
Users: 24,368
Add to Firefox

PhotoShow
by Vincent W.
Hover your mouse over thumbnails to easily browse and download HD images.
Users: 5,691
Add to Firefox

Figure 2: Enabled browser extensions used during traffic generation for Wireshark capture.


task 5 wireshark .pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Packets: 888
Profile: Default

```
No.   Time            Source           Destination      Protocol  Length Info
  649 204.631265144 10.0.2.15        34.160.144.191   TLSv1.2     85 Encrypted Alert
  650 204.631984115 34.160.144.191   10.0.2.15        TCP         60 443 → 41894 [ACK] Seq=356 Ack=875 Win=65535 Len=0
  651 204.632107291 10.0.2.15        34.160.144.191   TCP         54 41894 → 443 [FIN, ACK] Seq=875 Ack=356 Win=63977 Len=0
  652 204.633108540 34.160.144.191   10.0.2.15        TCP         60 443 → 41894 [ACK] Seq=356 Ack=876 Win=65535 Len=0
  653 204.642838666 34.160.144.191   10.0.2.15        TCP         60 443 → 41894 [FIN, ACK] Seq=356 Ack=876 Win=65535 Len=0
  654 204.642873288 10.0.2.15        34.160.144.191   TCP         54 41894 → 443 [ACK] Seq=876 Ack=357 Win=63977 Len=0
  655 204.647127582 151.101.193.91   10.0.2.15        TLSv1.2     85 Encrypted Alert
  656 204.647128073 151.101.193.91   10.0.2.15        TCP         60 443 → 41338 [FIN, ACK] Seq=2066 Ack=1195 Win=65535 Len=0
  657 204.647154296 10.0.2.15        151.101.193.91   TCP         54 41338 → 443 [RST] Seq=1195 Win=0 Len=0
  658 204.647285982 10.0.2.15        151.101.193.91   TCP         54 41338 → 443 [RST] Seq=1195 Win=0 Len=0
  659 204.649372212 151.101.193.91   10.0.2.15        TCP         60 443 → 41338 [RST, ACK] Seq=4245623295 Ack=1195 Win=0 Len=0
  660 210.340929572 10.0.2.15        4.153.72.49      TCP         74 44702 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2630017363 TSecr=0 WS=128
  661 210.595104090 10.0.2.15        4.153.72.49      TCP         74 44714 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2630017617 TSecr=0 WS=128
  662 210.609602078 4.153.72.49      10.0.2.15        TCP         60 443 → 44702 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
  663 210.609674362 10.0.2.15        4.153.72.49      TCP         54 44702 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
  664 210.612010983 10.0.2.15        4.153.72.49      TLSv1.2    922 Client Hello (SNI=i.clarity.ms)
  665 210.613027940 4.153.72.49      10.0.2.15        TCP         60 443 → 44702 [ACK] Seq=1 Ack=869 Win=65535 Len=0
  666 210.866500779 4.153.72.49      10.0.2.15        TCP         60 443 → 44714 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
  667 210.866603986 10.0.2.15        4.153.72.49      TCP         54 44714 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
  668 210.872639914 10.0.2.15        4.153.72.49      TLSv1.2    922 Client Hello (SNI=i.clarity.ms)
```

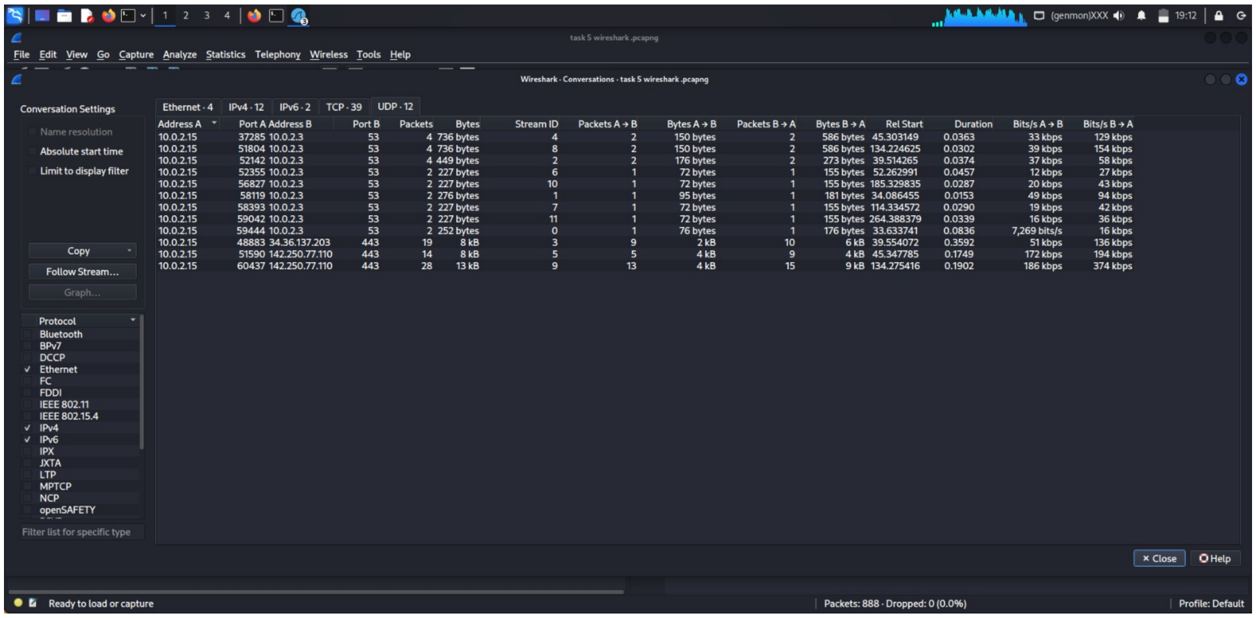
```
› Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface eth0, id 0
› Ethernet II, Src: PCSSystemtec_84:c0:76 (08:00:27:84:c0:76), Dst: 52:55:0a:00:02:02 (52:55:0a:00:02:02)
› Internet Protocol Version 4, Src: 10.0.2.15, Dst: 4.153.72.49
› Transmission Control Protocol, Src Port: 46348, Dst Port: 443, Seq: 0, Len: 0
```

```
0000  52 55 0a 00 02 02 08 00  27 84 c0 76 08 00 45 00   RU······'··v··E·
0010  00 3c 9c 83 40 00 40 06  45 60 0a 00 02 0f 04 99   ·<··@·@·E`······
0020  48 31 b5 0c 01 bb ee f5  d5 2d 00 00 00 00 a0 02   H1·······-······
0030  fa f0 59 07 00 00 02 04  05 b4 04 02 08 0a 9c bf   ··Y·············
0040  ab ae 00 00 00 00 01 03  03 07                     ··········
```

TCP Packet Capture in Wireshark showing source, destination, protocol type, and sequence details.

Observation

The capture highlights a series of TCP packets exchanged between IP addresses 10.0.2.15 and 151.101.193.91 over port 443 (HTTPS). Notably, there are reset (RST) flags observed in packets 657 and 658, indicating an abrupt termination or rejection of the connection, possibly due to an error or forced close from either endpoint.


Wireshark · Conversations · task 5 wireshark .pcapng

Ethernet · 4 | IPv4 · 12 | IPv6 · 2 | TCP · 39 | UDP · 12

Address A	Port A	Address B	Port B	Packets	Bytes	Stream ID	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.0.2.15	37285	10.0.2.3	53	4	736 bytes	4	2	150 bytes	2	586 bytes	45.303149	0.0363	33 kbps	129 kbps
10.0.2.15	51804	10.0.2.3	53	4	736 bytes	8	2	150 bytes	2	586 bytes	134.224625	0.0302	39 kbps	154 kbps
10.0.2.15	52142	10.0.2.3	53	4	449 bytes	2	2	176 bytes	2	273 bytes	39.514265	0.0374	37 kbps	58 kbps
10.0.2.15	52355	10.0.2.3	53	2	227 bytes	6	1	72 bytes	1	155 bytes	52.262991	0.0457	12 kbps	27 kbps
10.0.2.15	56827	10.0.2.3	53	2	227 bytes	10	1	72 bytes	1	155 bytes	185.329835	0.0287	20 kbps	43 kbps
10.0.2.15	58119	10.0.2.3	53	2	276 bytes	1	1	95 bytes	1	181 bytes	34.086455	0.0153	49 kbps	94 kbps
10.0.2.15	58393	10.0.2.3	53	2	227 bytes	7	1	72 bytes	1	155 bytes	114.334572	0.0290	19 kbps	42 kbps
10.0.2.15	59042	10.0.2.3	53	2	227 bytes	11	1	72 bytes	1	155 bytes	264.388379	0.0339	16 kbps	36 kbps
10.0.2.15	59444	10.0.2.3	53	2	252 bytes	0	1	76 bytes	1	176 bytes	33.633741	0.0836	7,269 bits/s	16 kbps
10.0.2.15	48883	34.36.137.203	443	19	8 kB	3	9	2 kB	10	6 kB	39.554072	0.3592	51 kbps	136 kbps
10.0.2.15	51590	142.250.77.110	443	14	8 kB	5	5	4 kB	9	4 kB	45.347785	0.1749	172 kbps	194 kbps
10.0.2.15	60437	142.250.77.110	443	28	13 kB	9	13	4 kB	15	9 kB	134.275416	0.1902	186 kbps	374 kbps

Conversation Settings: Absolute start time; Limit to display filter

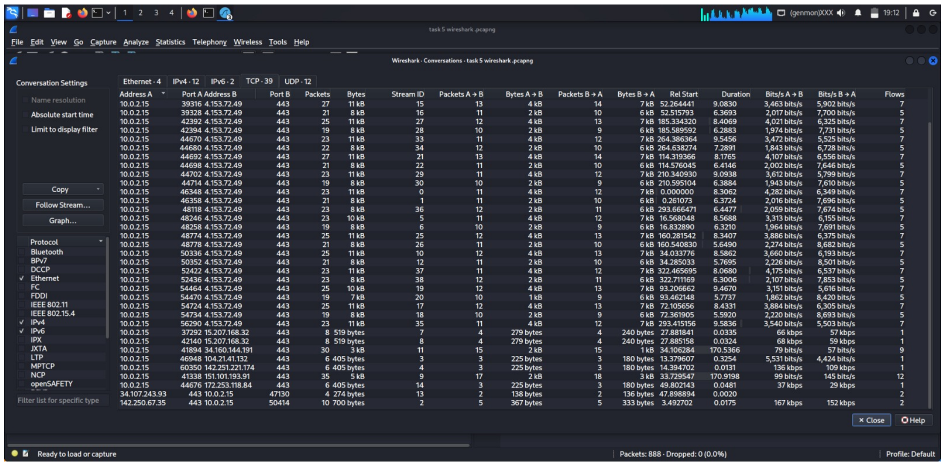
Protocol: ✓ Ethernet, ✓ IPv4, ✓ IPv6

Ready to load or capture | Packets: 888 · Dropped: 0 (0.0%) | Profile: Default

Figure 3: Packet Analysis

Observations

- Majority of traffic originates from IP communicating with internal () and external IPs.
- Port 53 (DNS) and 443 (HTTPS) are the primary ports used.
- HTTPS communication with and shows the highest data transfer (up to 13 kB).
- Data rates peaked at 374 kbps, with short session durations (below 0.5 sec).
- No suspicious activity was detected in this short capture.

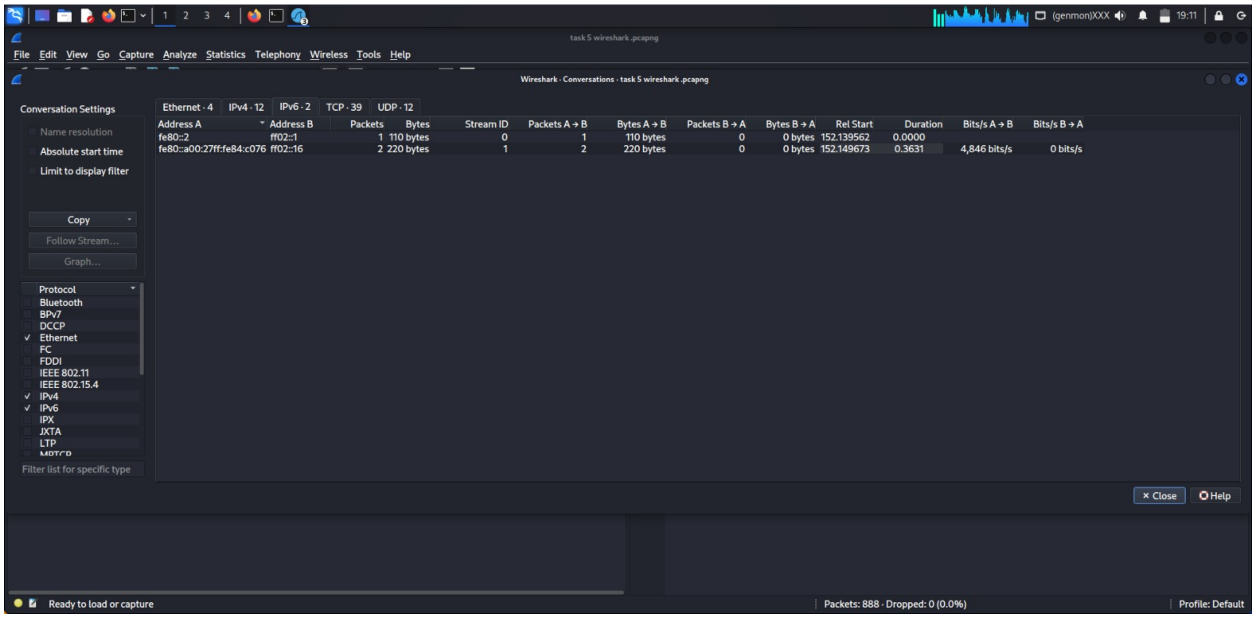


Address A	Port A	Address B	Port B	Packets	Bytes	Stream ID	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A	Flows
10.0.2.15	39316	4.153.72.49	443	27	11 kB	15	13	4 kB	14	7 kB	52.264441	9.0830	3,463 bits/s	5,902 bits/s	7
10.0.2.15	39328	4.153.72.49	443	21	8 kB	16	11	2 kB	10	6 kB	52.515793	6.3693	2,017 bits/s	7,700 bits/s	5
10.0.2.15	42392	4.153.72.49	443	25	11 kB	27	12	4 kB	13	7 kB	185.334320	8.4069	4,021 bits/s	6,325 bits/s	7
10.0.2.15	42394	4.153.72.49	443	19	8 kB	28	10	2 kB	9	6 kB	185.589592	6.2883	1,974 bits/s	7,731 bits/s	5
10.0.2.15	44670	4.153.72.49	443	23	11 kB	33	11	4 kB	12	7 kB	264.386364	9.5456	3,472 bits/s	5,525 bits/s	7
10.0.2.15	44680	4.153.72.49	443	22	8 kB	34	12	2 kB	10	6 kB	264.638274	7.2891	1,843 bits/s	6,728 bits/s	5
10.0.2.15	44692	4.153.72.49	443	27	11 kB	21	13	4 kB	14	7 kB	114.319366	8.1765	4,107 bits/s	6,556 bits/s	7
10.0.2.15	44698	4.153.72.49	443	21	8 kB	22	11	2 kB	10	6 kB	114.576045	6.4146	2,002 bits/s	7,646 bits/s	5
10.0.2.15	44702	4.153.72.49	443	23	11 kB	29	11	4 kB	12	7 kB	210.340930	9.0938	3,612 bits/s	5,799 bits/s	7
10.0.2.15	44714	4.153.72.49	443	19	8 kB	30	10	2 kB	9	6 kB	210.595104	6.3884	1,943 bits/s	7,610 bits/s	5
10.0.2.15	46348	4.153.72.49	443	23	11 kB	0	11	4 kB	12	7 kB	0.000000	8.3062	4,282 bits/s	6,349 bits/s	7
10.0.2.15	46358	4.153.72.49	443	21	8 kB	1	11	2 kB	10	6 kB	0.261073	6.3724	2,016 bits/s	7,696 bits/s	5
10.0.2.15	48118	4.153.72.49	443	23	8 kB	36	12	2 kB	11	6 kB	293.666471	6.4477	2,059 bits/s	7,674 bits/s	5
10.0.2.15	48246	4.153.72.49	443	23	10 kB	5	11	4 kB	12	7 kB	16.568048	8.5688	3,313 bits/s	6,155 bits/s	7
10.0.2.15	48258	4.153.72.49	443	19	8 kB	6	10	2 kB	9	6 kB	16.832890	6.3210	1,964 bits/s	7,691 bits/s	5
10.0.2.15	48774	4.153.72.49	443	25	11 kB	25	12	4 kB	13	7 kB	160.281542	8.3407	3,886 bits/s	6,375 bits/s	7
10.0.2.15	48778	4.153.72.49	443	21	8 kB	26	11	2 kB	10	6 kB	160.540830	5.6490	2,274 bits/s	8,682 bits/s	5
10.0.2.15	50336	4.153.72.49	443	25	11 kB	10	12	4 kB	13	7 kB	34.033776	8.5862	3,660 bits/s	6,193 bits/s	7
10.0.2.15	50352	4.153.72.49	443	21	8 kB	12	11	2 kB	10	6 kB	34.285033	5.7695	2,226 bits/s	8,501 bits/s	5
10.0.2.15	52422	4.153.72.49	443	23	11 kB	37	11	4 kB	12	7 kB	322.465695	8.0680	4,175 bits/s	6,537 bits/s	7
10.0.2.15	52436	4.153.72.49	443	23	8 kB	38	12	2 kB	11	6 kB	322.711169	6.3006	2,107 bits/s	7,853 bits/s	5
10.0.2.15	54464	4.153.72.49	443	25	10 kB	19	12	4 kB	13	7 kB	93.206662	9.4670	3,151 bits/s	5,616 bits/s	7
10.0.2.15	54470	4.153.72.49	443	19	7 kB	20	10	1 kB	9	6 kB	93.462148	5.7737	1,862 bits/s	8,420 bits/s	5
10.0.2.15	54724	4.153.72.49	443	25	11 kB	17	12	4 kB	13	7 kB	72.105656	8.4331	3,884 bits/s	6,305 bits/s	7
10.0.2.15	54734	4.153.72.49	443	19	8 kB	18	10	2 kB	9	6 kB	72.361905	5.5920	2,220 bits/s	8,693 bits/s	5
10.0.2.15	56290	4.153.72.49	443	23	11 kB	35	11	4 kB	12	7 kB	293.415156	9.5836	3,540 bits/s	5,503 bits/s	7
10.0.2.15	37292	15.207.168.32	443	8	519 bytes	7	4	279 bytes	4	240 bytes	27.881841	0.0335	66 kbps	57 kbps	1
10.0.2.15	42140	15.207.168.32	443	8	519 bytes	8	4	279 bytes	4	240 bytes	27.885158	0.0324	68 kbps	59 kbps	1
10.0.2.15	41894	34.160.144.191	443	30	3 kB	11	15	2 kB	15	1 kB	34.106284	170.5366	79 bits/s	57 bits/s	9
10.0.2.15	46948	104.21.41.132	443	6	405 bytes	3	3	225 bytes	3	180 bytes	13.379607	0.3254	5,531 bits/s	4,424 bits/s	1
10.0.2.15	60350	142.251.221.174	443	6	405 bytes	4	3	225 bytes	3	180 bytes	14.394702	0.0131	136 kbps	109 kbps	1
10.0.2.15	41338	151.101.193.91	443	35	5 kB	9	17	2 kB	18	3 kB	33.729547	170.9198	99 bits/s	145 bits/s	12
10.0.2.15	44676	172.253.118.84	443	6	405 bytes	14	3	225 bytes	3	180 bytes	49.802143	0.0481	37 kbps	29 kbps	1
34.107.243.93	443	10.0.2.15	47130	4	274 bytes	13	2	138 bytes	2	136 bytes	47.898894	0.0020			2
142.250.67.35	443	10.0.2.15	50414	10	700 bytes	2	5	367 bytes	5	333 bytes	3.492702	0.0175	167 kbps	152 kbps	2

Figure 4: Wireshark conversation analysis showing IPv4, TCP, and Ethernet protocols with details like packets, bytes, and bit rates.

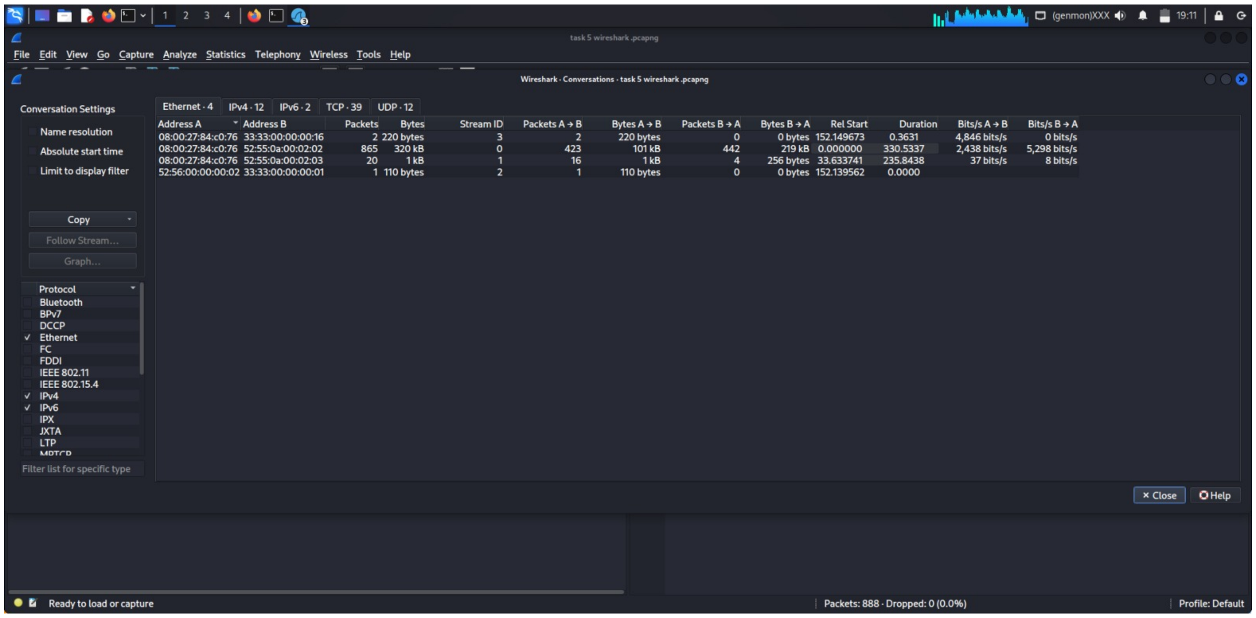
Observation:

This image shows the **Wireshark Conversation Statistics** window displaying multiple TCP and IPv4 communication sessions. The local IP `10.0.2.15` communicated primarily with remote IP `4.153.72.49` on port 443 (HTTPS), confirming encrypted web traffic. Data columns display packet count, bytes transferred, duration, and bit rates. Protocols observed include **Ethernet**, **IPv4**, and **TCP**, with stable throughput and no packet loss, validating successful capture and analysis of secure web communication.



Address A	Address B	Packets	Bytes	Stream ID	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
fe80::2	ff02::1	1	110 bytes	0	1	110 bytes	0	0 bytes	152.139562	0.0000		
fe80::a00:27ff:fe84:c076	ff02::16	2	220 bytes	1	2	220 bytes	0	0 bytes	152.149673	0.3631	4,846 bits/s	0 bits/s

Figure 5: Wireshark IPv6 Conversation Statistics, showing a minimal number of active flows (Stream 1) between two link-local addresses (fe80::) over a short duration.



Address A	Address B	Packets	Bytes	Stream ID	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
08:00:27:84:c0:76	33:33:00:00:00:16	2	220 bytes	3	2	220 bytes	0	0 bytes	152.149673	0.3631	4,846 bits/s	0 bits/s
08:00:27:84:c0:76	52:55:0a:00:02:02	865	320 kB	0	423	101 kB	442	219 kB	0.000000	330.5337	2,438 bits/s	5,298 bits/s
08:00:27:84:c0:76	52:55:0a:00:02:03	20	1 kB	1	16	1 kB	4	256 bytes	33.633741	235.8438	37 bits/s	8 bits/s
52:56:00:00:00:02	33:33:00:00:00:01	1	110 bytes	2	1	110 bytes	0	0 bytes	152.139562	0.0000		

Figure 6: Wireshark Ethernet Conversation Statistics, detailing Layer 2 traffic by MAC address, with the dominant flow (320 KB) occurring between the client's gateway and another MAC address (52:55:0a:00:02:02).

Observation:
The Firefox Add-ons Manager interface displays three enabled extensions — **Cookie-Editor**, **FoxyProxy**, and **Wappalyzer**.

- **Cookie-Editor:** Used to create, edit, or delete cookies.
- **FoxyProxy:** Simplifies proxy management for generating diverse traffic.
- **Wappalyzer:** Identifies web technologies of visited websites.

These add-ons were active during browsing, creating HTTP(S) and proxy-related packets visible in Wireshark analysis.

3 Result

The packet capture and analysis task using Wireshark was successfully completed. Network traffic was captured from the active interface while browsing various websites, and the resulting `.pcap` file was analyzed. The captured data revealed multiple communication sessions between the host system (`10.0.2.15`) and remote servers primarily over port 443 (HTTPS).

Different network protocols such as **Ethernet**, **IPv4**, and **TCP** were identified in the capture. Statistical analysis in Wireshark showed continuous encrypted web traffic with no packet loss, indicating a stable and active connection during the capture session.