Project: Messaging Application

Python

Study Course

B205 Computer Networks

By

Chehab Hany Mohamed Elsayed Elsayed Elmenoufi

Student Number: GH1034223

Under the Guidance of

Prof. Sami Alsalamin

GitHub URL: https://github.com/Chippo90/Computer-Networks/tree/main

Video Recording URL: https://youtu.be/icIrDzlxHTc

Gisma University of Applied Sciences

Berlin, Germany

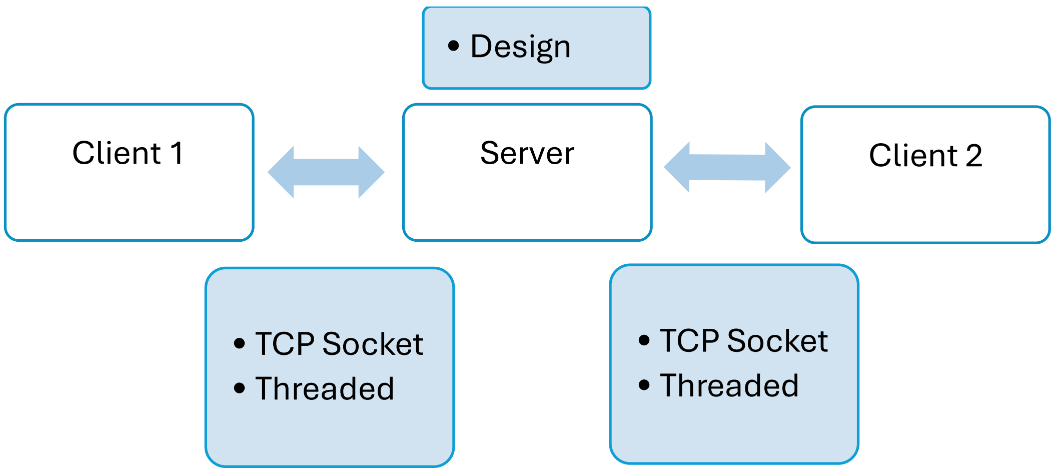
June 2025

Table of Contents

1. Task 1 – Messaging Application

1.1 System Architecture Design

The system is based on a Client-Server architecture using Python. The server accepts many client connections. Each client communicates with the server over a TCP connection. Messages and files are routed through the server.(*Client-Server Model*, 00:30:23+00:00)



1.2 Protocol Specifications

The application protocol contains the below layers:-

Layer	Protocol	Description
Transport Layer	TCP	Reliable messaging and file transferring
Application Layer	Custom text for header	UTF-8 message for files

Message Design:

- Text: <username>: <message>
- File Transfer: [FILE]:<filename>

1.3 Network Communication Flow

1.3.1 Client

- Connect to server.
- Enter username.

1.3.2 Server

- Add user to the client list.
- Send “user joined” notification.

1.3.3 Messaging

- Client send a message.
- Server tag it with username and send it to all clients.

1.3.4 File Transfer

- Client send a file.
- Header [FILE]:<filename> is sent
- End of file marked with <END>
- Other clients receive and save as received_<filename>

1.3.5 Exit

- Client send exit or closes window.
- Server notify others.

1.4 Protocol Selection Rationale

This project uses TCP for all communication between clients and the server.(*TCP/IP Model*, 13:33:57+00:00)

TCP was selected because:

- It guarantees reliable delivery of messages.
- It supports stable connections.
- Built-in error checking.

At the application layer, a custom protocol was designed using UTF-8 encoded text messages and simple header tags for control messages. (*Unicode HOWTO*, no date)

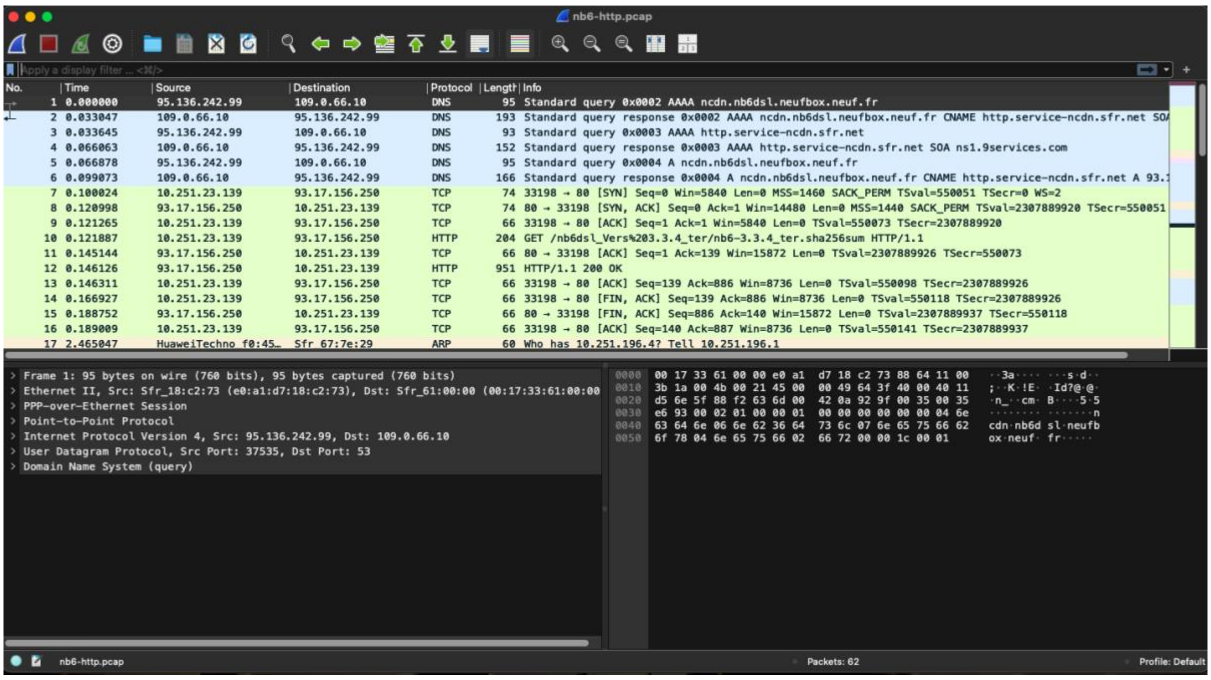
1.5 Pros and Cons for the TCP Protocol

Protocol	Pros	Cons
TCP	Reliable delivery, connection-oriented, handles congestion and retransmission	Slightly more overhead than UDP, not ideal for real-time video/audio
Custom Protocol using UTF-8	Easy to implement and debug, human-readable	Not encrypted, no authentication or compression built-in

(‘computer-networking-a-top-down-approach-8th-edition.pdf’, no date)

2. Task 2 - Wireshark

In this task, I have downloaded a file from the internet related to HTTP traffic.('nb6-http.pcap', no date)



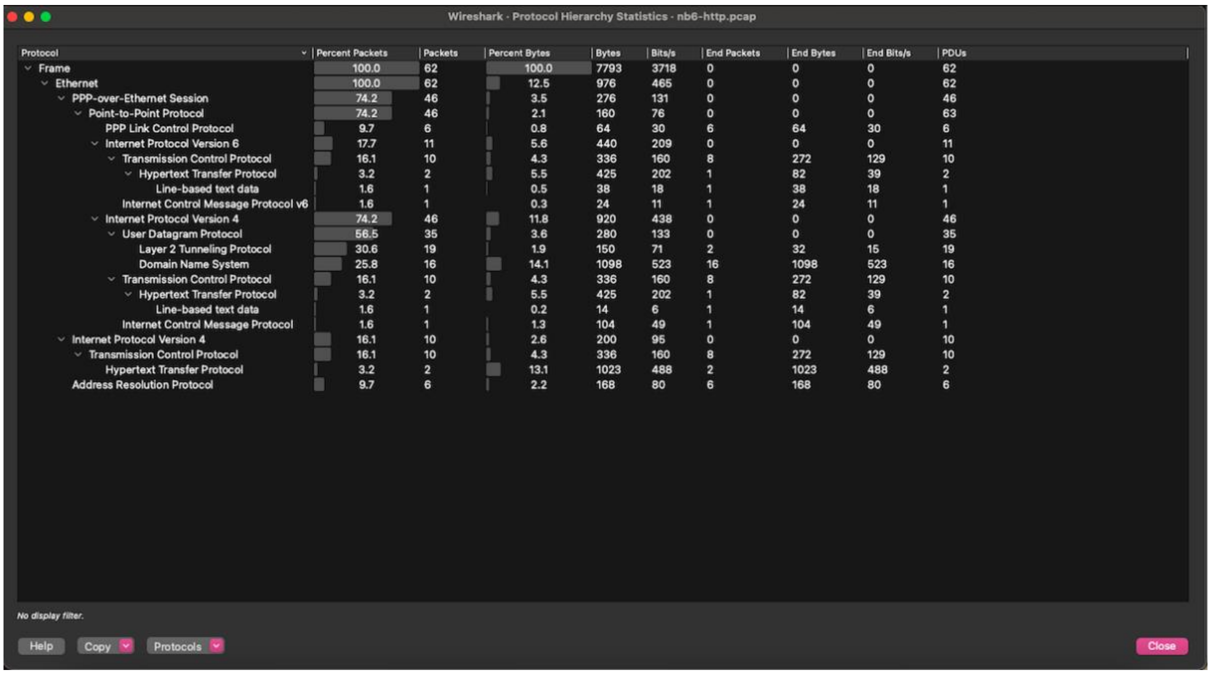
```
No.  |Time      |Source              |Destination     |Protocol |Length|Info
   1 0.000000   95.136.242.99        109.0.66.10      DNS        95 Standard query 0x0002 AAAA ncdn.nb6dsl.neufbox.neuf.fr
   2 0.033047   109.0.66.10          95.136.242.99    DNS       193 Standard query response 0x0002 AAAA ncdn.nb6dsl.neufbox.neuf.fr CNAME http.service-ncdn.sfr.net SOA
   3 0.033645   95.136.242.99        109.0.66.10      DNS        93 Standard query 0x0003 AAAA http.service-ncdn.sfr.net
   4 0.066063   109.0.66.10          95.136.242.99    DNS       152 Standard query response 0x0003 AAAA http.service-ncdn.sfr.net SOA ns1.9services.com
   5 0.066878   95.136.242.99        109.0.66.10      DNS        95 Standard query 0x0004 A ncdn.nb6dsl.neufbox.neuf.fr
   6 0.099073   109.0.66.10          95.136.242.99    DNS       166 Standard query response 0x0004 A ncdn.nb6dsl.neufbox.neuf.fr CNAME http.service-ncdn.sfr.net A 93.1
   7 0.100024   10.251.23.139        93.17.156.250    TCP        74 33198 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=550051 TSecr=0 WS=2
   8 0.120998   93.17.156.250        10.251.23.139    TCP        74 80 → 33198 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1440 SACK_PERM TSval=2307889920 TSecr=550051
   9 0.121265   10.251.23.139        93.17.156.250    TCP        66 33198 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0 TSval=550073 TSecr=2307889920
  10 0.121887   10.251.23.139        93.17.156.250    HTTP      204 GET /nb6dsl_Vers%203.3.4_ter/nb6-3.3.4_ter.sha256sum HTTP/1.1
  11 0.145144   93.17.156.250        10.251.23.139    TCP        66 80 → 33198 [ACK] Seq=1 Ack=139 Win=15872 Len=0 TSval=2307889926 TSecr=550073
  12 0.146126   93.17.156.250        10.251.23.139    HTTP      951 HTTP/1.1 200 OK
  13 0.146311   10.251.23.139        93.17.156.250    TCP        66 33198 → 80 [ACK] Seq=139 Ack=886 Win=8736 Len=0 TSval=550098 TSecr=2307889926
  14 0.166927   10.251.23.139        93.17.156.250    TCP        66 33198 → 80 [FIN, ACK] Seq=139 Ack=886 Win=8736 Len=0 TSval=550118 TSecr=2307889926
  15 0.188752   93.17.156.250        10.251.23.139    TCP        66 80 → 33198 [FIN, ACK] Seq=886 Ack=140 Win=15872 Len=0 TSval=2307889937 TSecr=550118
  16 0.189009   10.251.23.139        93.17.156.250    TCP        66 33198 → 80 [ACK] Seq=140 Ack=887 Win=8736 Len=0 TSval=550141 TSecr=2307889937
  17 2.465047   HuaweiTechno_f0:45…  Sfr_67:7e:29     ARP        60 Who has 10.251.196.4? Tell 10.251.196.1
```

```
> Frame 1: 95 bytes on wire (760 bits), 95 bytes captured (760 bits)
> Ethernet II, Src: Sfr_18:c2:73 (e0:a1:d7:18:c2:73), Dst: Sfr_61:00:00 (00:17:33:61:00:00
> PPP-over-Ethernet Session
> Point-to-Point Protocol
> Internet Protocol Version 4, Src: 95.136.242.99, Dst: 109.0.66.10
> User Datagram Protocol, Src Port: 37535, Dst Port: 53
> Domain Name System (query)
```

```
0000  00 17 33 61 00 00 e0 a1  d7 18 c2 73 88 64 11 00   ··3a····  ···s·d··
0010  3b 1a 00 4b 00 21 45 00  00 49 64 3f 40 00 40 11   ;··K·!E·  ·Id?@·@·
0020  d5 6e 5f 88 f2 63 6d 00  42 0a 92 9f 00 35 00 35   ·n_··cm·  B····5·5
0030  e6 93 00 02 01 00 00 01  00 00 00 00 00 00 04 6e   ········  ·······n
0040  63 64 6e 06 6e 62 36 64  73 6c 07 6e 65 75 66 62   cdn·nb6d  sl·neufb
0050  6f 78 04 6e 65 75 66 02  66 72 00 00 1c 00 01      ox·neuf·  fr·····
```

2.1 Protocol Hierarchy Analysis

(*3.11. The "Statistics" Menu*, no date)



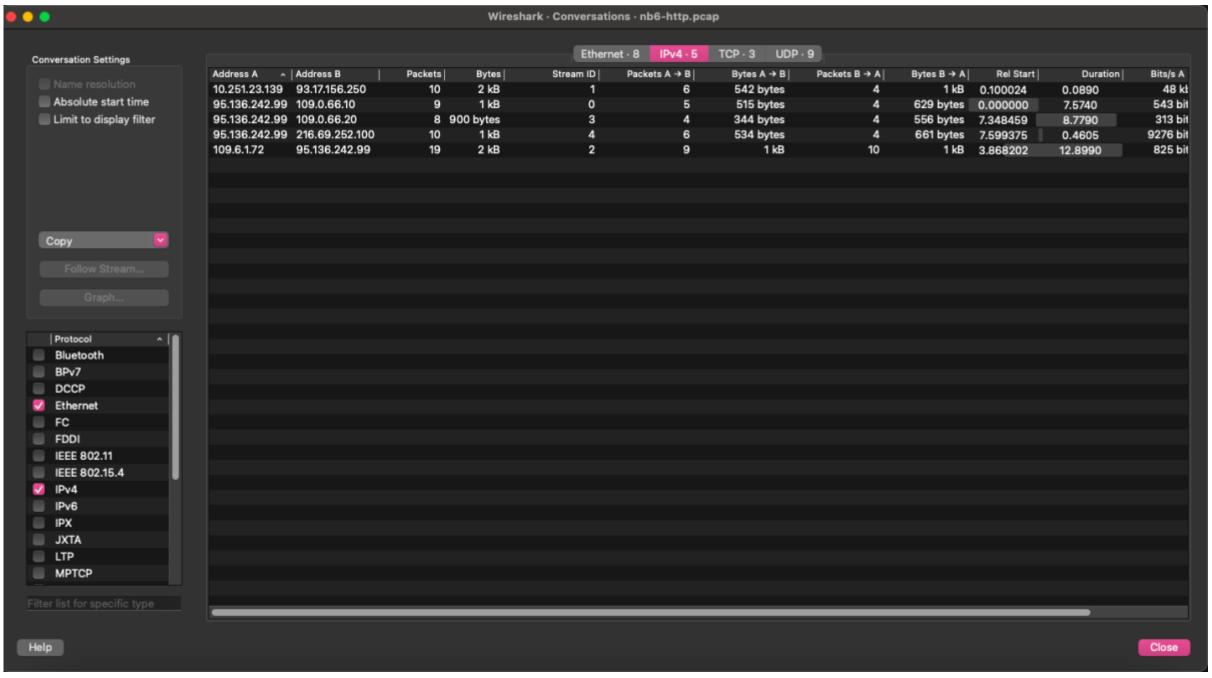
Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	62	100.0	7793	3718	0	0	0	62
Ethernet	100.0	62	12.5	976	465	0	0	0	62
PPP-over-Ethernet Session	74.2	46	3.5	276	131	0	0	0	46
Point-to-Point Protocol	74.2	46	2.1	160	76	0	0	0	63
PPP Link Control Protocol	9.7	6	0.8	64	30	6	64	30	6
Internet Protocol Version 6	17.7	11	5.6	440	209	0	0	0	11
Transmission Control Protocol	16.1	10	4.3	336	160	8	272	129	10
Hypertext Transfer Protocol	3.2	2	5.5	425	202	1	82	39	2
Line-based text data	1.6	1	0.5	38	18	1	38	18	1
Internet Control Message Protocol v6	1.6	1	0.3	24	11	1	24	11	1
Internet Protocol Version 4	74.2	46	11.8	920	438	0	0	0	46
User Datagram Protocol	56.5	35	3.6	280	133	0	0	0	35
Layer 2 Tunneling Protocol	30.6	19	1.9	150	71	2	32	15	19
Domain Name System	25.8	16	14.1	1098	523	16	1098	523	16
Transmission Control Protocol	16.1	10	4.3	336	160	8	272	129	10
Hypertext Transfer Protocol	3.2	2	5.5	425	202	1	82	39	2
Line-based text data	1.6	1	0.2	14	6	1	14	6	1
Internet Control Message Protocol	1.6	1	1.3	104	49	1	104	49	1
Internet Protocol Version 4	16.1	10	2.6	200	95	0	0	0	10
Transmission Control Protocol	16.1	10	4.3	336	160	8	272	129	10
Hypertext Transfer Protocol	3.2	2	13.1	1023	488	2	1023	488	2
Address Resolution Protocol	9.7	6	2.2	168	80	6	168	80	6

Observations:-

- IPv4 and IPv6 traffic present.
- Most traffic is TCP (16.1%) and UDP (56.5%).
- Small percentage of HTTP (3.2%) but not HTTPS.

2.2 Conversations Analysis

(*3.11. The "Statistics" Menu*, no date)



Wireshark · Conversations · nb6-http.pcap

Conversation Settings

- Name resolution
- Absolute start time
- Limit to display filter

Ethernet · 8 | IPv4 · 5 | TCP · 3 | UDP · 9

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
10.251.23.139	93.17.156.250	10	2 kB	1	6	542 bytes	4	1 kB	0.100024	0.0890	48 kl
95.136.242.99	109.0.66.10	9	1 kB	0	5	515 bytes	4	629 bytes	0.000000	7.5740	543 bit
95.136.242.99	109.0.66.20	8	900 bytes	3	4	344 bytes	4	556 bytes	7.348459	8.7790	313 bit
95.136.242.99	216.69.252.100	10	1 kB	4	6	534 bytes	4	661 bytes	7.599375	0.4605	9276 bit
109.6.1.72	95.136.242.99	19	2 kB	2	9	1 kB	10	1 kB	3.868202	12.8990	825 bit

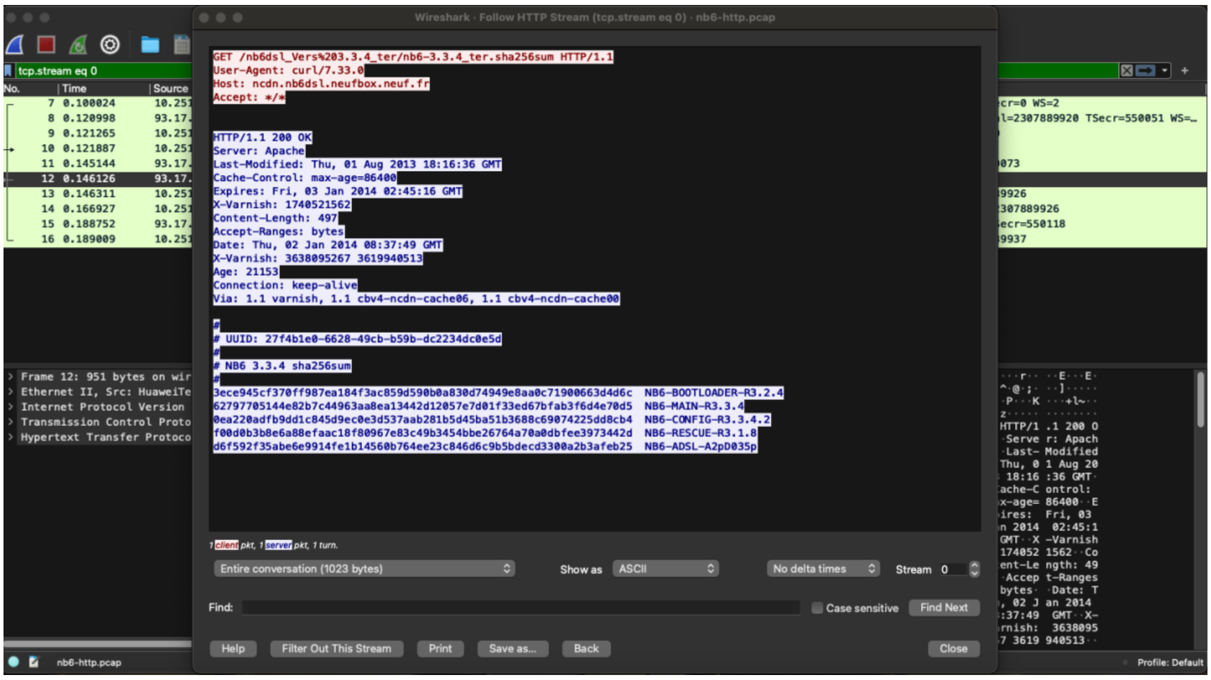
	Protocol
	Bluetooth
	BPv7
	DCCP
✓	Ethernet
	FC
	FDDI
	IEEE 802.11
	IEEE 802.15.4
✓	IPv4
	IPv6
	IPX
	JXTA
	LTP
	MPTCP

Observations:-

- IP 95.136.242.99 communicates with 109.0.66.20, 109.0.66.10, and 216.69.252.100
- Multiple connections with 8–10 packets per stream

2.3 HTTP Stream Analysis

(*3.11. The "Statistics" Menu*, no date)


Wireshark · Follow HTTP Stream (tcp.stream eq 0) · nb6-http.pcap

```
GET /nb6dsl_Vers%203.3.4_ter/nb6-3.3.4_ter.sha256sum HTTP/1.1
User-Agent: curl/7.33.0
Host: ncdn.nb6dsl.neufbox.neuf.fr
Accept: */*


HTTP/1.1 200 OK
Server: Apache
Last-Modified: Thu, 01 Aug 2013 18:16:36 GMT
Cache-Control: max-age=86400
Expires: Fri, 03 Jan 2014 02:45:16 GMT
X-Varnish: 1740521562
Content-Length: 497
Accept-Ranges: bytes
Date: Thu, 02 Jan 2014 08:37:49 GMT
X-Varnish: 3638095267 3619940513
Age: 21153
Connection: keep-alive
Via: 1.1 varnish, 1.1 cbv4-ncdn-cache06, 1.1 cbv4-ncdn-cache00

#
# UUID: 27f4b1e0-6628-49cb-b59b-dc2234dc0e5d
#
# NB6 3.3.4 sha256sum
#
3ece945cf370ff987ea184f3ac859d590b0a830d74949e8aa0c71900663d4d6c  NB6-BOOTLOADER-R3.2.4
62797705144e82b7c44963aa8ea13442d12057e7d01f33ed67bfab3f6d4e70d5  NB6-MAIN-R3.3.4
0ea220adfb9dd1c845d9ec0e3d537aab281b5d45ba51b3688c69074225dd8cb4  NB6-CONFIG-R3.3.4.2
f00d0b3b8e6a88efaac18f80967e83c49b3454bbe26764a70a0dbfee3973442d  NB6-RESCUE-R3.1.8
d6f592f35abe6e9914fe1b14560b764ee23c846d6c9b5bdecd3300a2b3afeb25  NB6-ADSL-A2pD035p
```

1 client pkt, 1 server pkt, 1 turn.

Entire conversation (1023 bytes) · Show as: ASCII · No delta times · Stream: 0

Observations:-

- This might be an update for network or computer (NB6-BOOTLOADER-R3.2.4), (NB6-MAIN-R3.3.4), (NB6-CONFIG-R3.4.2), (NB6-ADSL-A2p035p)

2.4 Firewall Rules

Rule	Action	Description
DROP TCP port 80	Block	Block unsecured HTTP files
PERMIT DNS from local network	Allow	Allow domain lookups
DROP TCP with user curl/7.33.0	Block	Block downloads
PERMIT TCP 443	Allow	Use HTTPS for updates

3. Conclusion and Future Work

This project shows the implementation of a client-server messaging application using Python. The application supports messaging, basic file transfer, and chat log.

Future Work:

- Implement user authentication and login system.
- Add encryption to secure communication.
- Create a user interface for better usability.

4. References

3.11. The "Statistics" Menu (no date). Available at: https://www.wireshark.org/docs/wsug_html_chunked/ChUseStatisticsMenuSection.html (Accessed: 24 June 2025).

Client-Server Model (00:30:23+00:00) *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/system-design/client-server-model/ (Accessed: 10 June 2025).

'computer-networking-a-top-down-approach-8th-edition.pdf' (no date). Available at: https://networking.harshkapadia.me/files/books/computer-networking-a-top-down-approach-8th-edition.pdf (Accessed: 17 May 2025).

'nb6-http.pcap' (no date).

TCP/IP Model (13:33:57+00:00) *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/computer-networks/tcp-ip-model/ (Accessed: 12 June 2025).

Unicode HOWTO (no date) *Python documentation*. Available at: https://docs.python.org/3/howto/unicode.html (Accessed: 16 June 2025).