

פרויקט ברשותה תקשורת

Project Source Code: [Link to GitHub Repository](#)(<https://github.com/LiroyKai/Computer-Networks-Final-Project>)

מגיסטים:

ליורי לייבל (212289987)

מייכאל וקנין (207973363)

רועי מימון (207972290)

חלק 1: אריזה ולכידת מנוט (Encapsulation & Capture)

1. ייצרת קובץ הנתונים (CSV)

בשלב הראשון של הפרויקט, נדרשנו ליצור תעבורת מדומינית של אפליקציה. לצורך כך, יצרנו קובץ CSV (בשם group212289987_http_input.csv) המכיל רשימה של הודעות ב프וטוקול HTTP, הקובץ נוצר בסיוו AI כדי להבטיח פורמט תקין נתונים מגוונים, והוא כולל את השדות הבאים: מזהה הודעה, פרוטוקול, אפליקציית מקור, אפליקצייתיעד תוכן ההודעה וחותמת זמן.

תהליך אריזת המנות(Encapsulation)

באמצעות מחברת Jupyter ביצענו סימולציה ידנית של תהליך הרכimos (Encapsulation) המתרחש במערכת הפעלה:

1. **שכבת היישום (Application):** חילצנו את המידע הגלומי (ההודעה) מתוך ה-CSV.
2. **שכבת התעבורה (Transport):** לכל הודעה הוספנו כותרת (Header) של פרוטוקול TCP/UDP/HTTP, הכוללת את פורט המקור ופורט היעד.
3. **שכבת הרשת (Network):** עטפנו את המזרב בכתובת IP, הכוללת את כתובות ה-IP-של המקור והיעד.
4. **שידור:** שלחנו את המנות המוכנות דרך ממשק הרשת, כך שתוכנת Wireshark תוכל לצלות אותן.

(Wireshark) ניתוח לכידת התעבורה

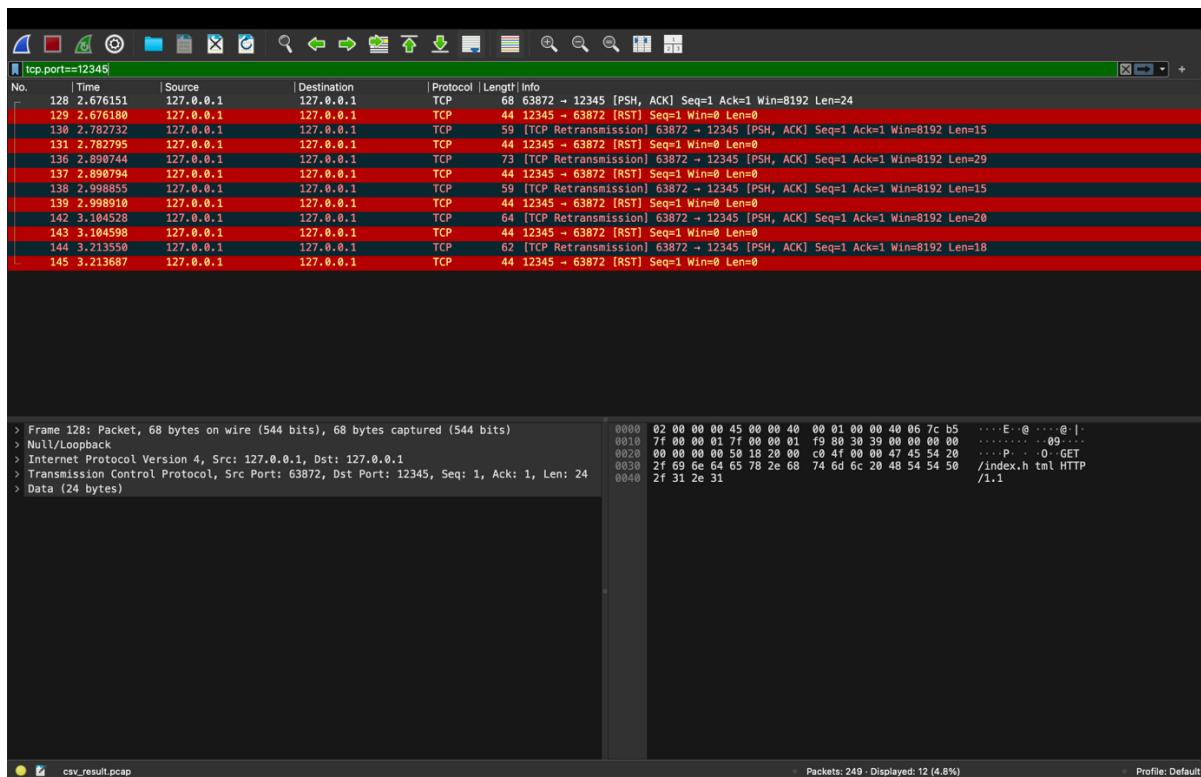
לאחר הרצת המחברת, ניתחנו את קובץ הלכידה csv_result.pcap.

דיאורי הпротокол Wireshark: זיהה את המנות TCP/HTTP-בהתבסס על הפורטים שהוגדרו.

תוכן המנה (Payload): בטור גוף המנה, ניתן לראות בבירור את הטקסט שנשלח מוקבץ ה- CSV.

תובנה: התרגיל המחייב כיצד כל שכבה מוסיפה את המידע המקורי לה (Headers) ללא תלות בתוכן המידע עצמו.

לכידת מנות ה- CSV ב Wireshark



No.	Time	Source	Destination	Protocol	Length	Info
128	2.676151	127.0.0.1	127.0.0.1	TCP	68	63872 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=24
129	2.676186	127.0.0.1	127.0.0.1	TCP	44	12345 → 63872 [RST] Seq=1 Win=0 Len=0
130	2.782732	127.0.0.1	127.0.0.1	TCP	59	[TCP Retransmission] 63872 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=15
131	2.782795	127.0.0.1	127.0.0.1	TCP	44	12345 → 63872 [RST] Seq=1 Win=0 Len=0
132	2.889744	127.0.0.1	127.0.0.1	TCP	73	[TCP Retransmission] 63872 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=29
133	2.889794	127.0.0.1	127.0.0.1	TCP	50	[TCP Retransmission] 63872 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=15
134	2.989315	127.0.0.1	127.0.0.1	TCP	44	12345 → 63872 [RST] Seq=1 Win=0 Len=0
135	3.108459	127.0.0.1	127.0.0.1	TCP	64	[TCP Retransmission] 63872 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=20
136	3.1084598	127.0.0.1	127.0.0.1	TCP	44	12345 → 63872 [RST] Seq=1 Win=0 Len=0
137	3.213550	127.0.0.1	127.0.0.1	TCP	62	[TCP Retransmission] 63872 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=8192 Len=18
145	3.213587	127.0.0.1	127.0.0.1	TCP	44	12345 → 63872 [RST] Seq=1 Win=0 Len=0

> Frame 130: Packet, 59 bytes on wire (472 bits), 59 bytes captured (472 bits)
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 63872, Dst Port: 12345, Seq: 1, Ack: 1, Len: 15

0009 02 00 00 00 45 00 00 37 00 01 00 00 48 06 7c be ..E..7 ..@ |
0010 7f 00 00 01 71 00 00 01 f9 80 30 39 00 00 00 ..09 ..
0020 00 00 00 50 18 28 00 b2 48 00 48 54 54 50 ..P...H..HTTP
0030 2f 31 2e 31 20 32 38 30 20 4f ab /1.1 200 OK

Packets: 249 - Displayed: 12 (4.8%) Profile: Default

חלק 2: יישום צ'אט וניתוח תယורה

מבנה המערכת והקוד

פיתחנו מערכת צ'אט מבוססת Sockets ב Python-הפעלת בארכיטקטורת שרת-לקוח-Server. המערכת אינה משתמשת בספריות ניהול שרתיים חיצונית, אלא ממשת את התקשרות באמצעות שיר.

רכיבי המערכת:

1. השרת (server.py) :

מנהל את כל התယורה

ריבוי תהליכיים (Multi-threading) עברו כל לקוח שמתחבר, השרת פותח Thread נפרד. מנגנון זה מאפשר לשרת לטפל במספר רב של לקוחות במקביל (מעל 5 כונדראש) מבלי לחסום את התקשרות עם לקוחות אחרים.

השרת מנהל רשימה משתמשים מחוברים וביצע ניתוב (Routing) של הודעות מלוקוט אחד לאחר.

2. הלקוח(client.py / Gui_Client.py) :

פותח חיבור TCP לשרת ומפריד בין תהליכי השידור והקליטה באמצעות Threads, מה שמאפשר קבלת הודעות בזמן אמת.

ממשק גרפי (GUI) כחלק מבונוס, פיתחנו ממשק משתמש באמצעות ספריית Tkinter. המשמש, מציג את היסטוריה הצ'אט ומפריד ויזואלית בין הודעות כניסה ליצאות.

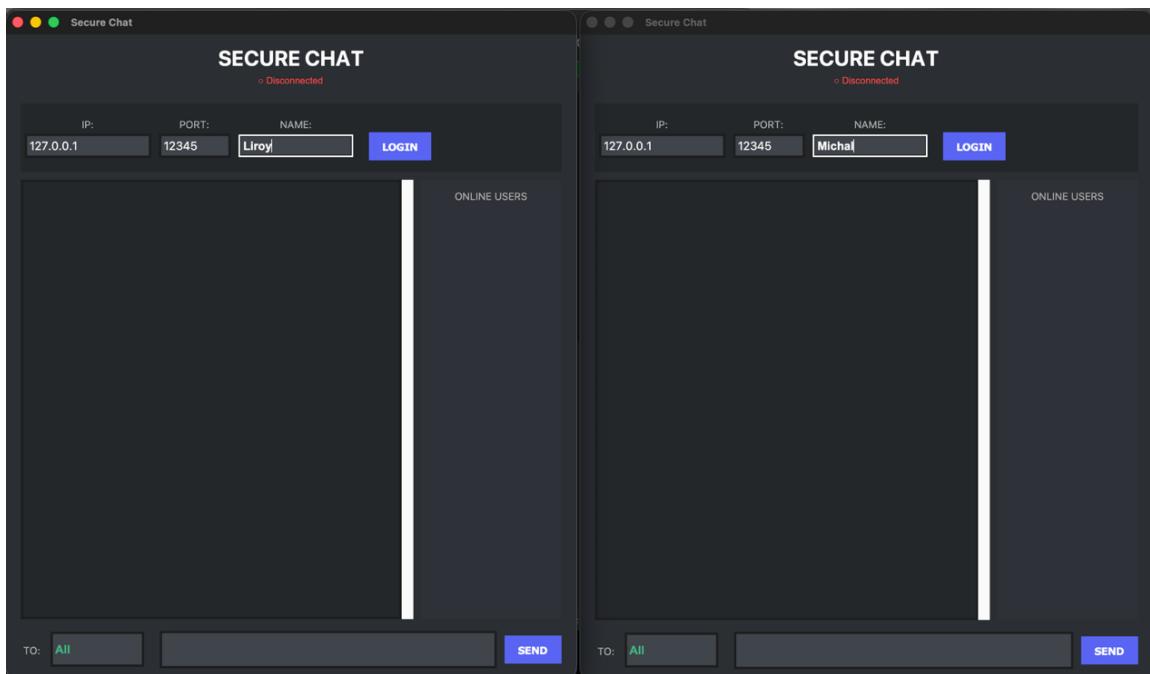
הוראות התקנה והרצה:

המערכת מtabסת על ספריות סטנדרטיות של Python ואינה דורשת התקנות מיוחדות.

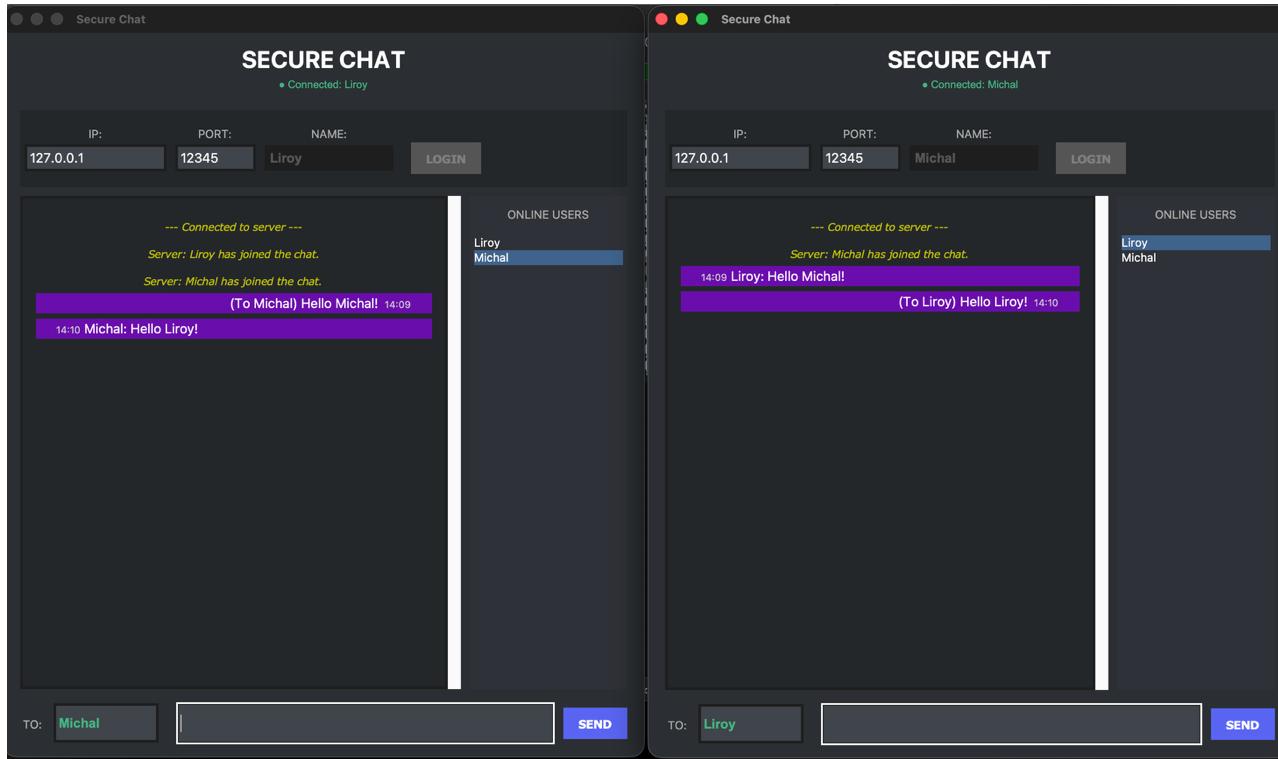
- יש להפעיל את השרת באמצעות הפקודה: `python server.py`: השרת יופיע בשם `python server.py`.
- עבור כל משתמש, יש להפעיל את הליקוב `python Gui_Client.py`.
- בחלון שנפתח, יש להזין שם משתמש ולהתחל בשייחת.

דוגמאות קלט ופלט:

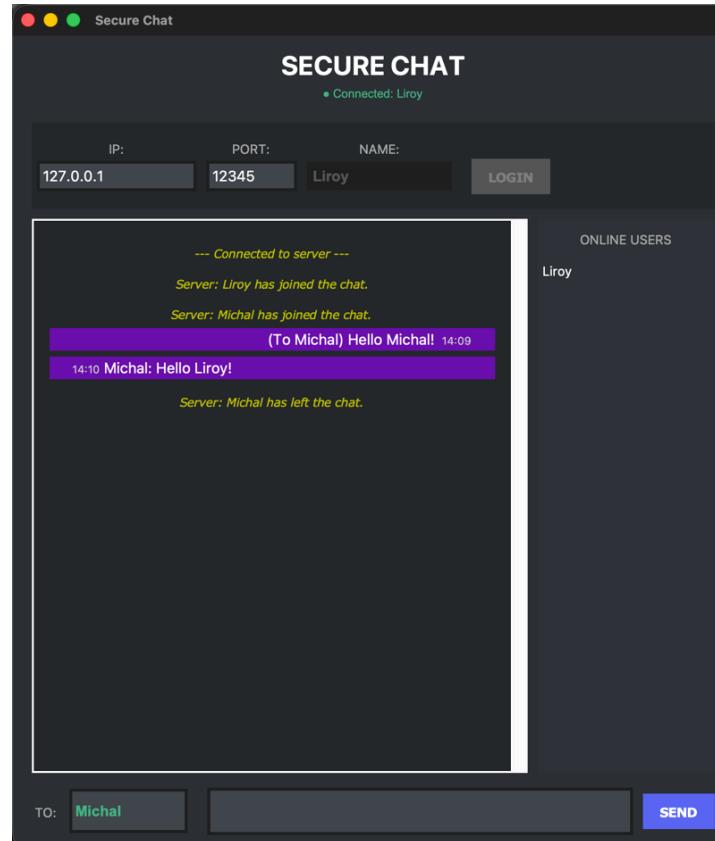
א. ממשק המשתמש (GUI) בתמונה הבאה ניתן לראות את חלון הצ'אט שפיתחנו. הממשק מאפשר חיבור נוח וצפיה בהודעות בזמן אמת.



ב. תרחיש שיחת בין לקוחות בתרחיש זה הפעלנו שני לקוחות במקביל. ההודעה שנשלחה מלקוח א' התקבלה מידית אצל לקוח ב'.



ג. לוג השרת השירות מציג בזמן אמת הסטטוס של המחברים והודעות מערכת כולל התראה על עציבה והצטרפות לצ'אט.



ניתוח תעבורת היישום

בניטוח קובץ הילכידה Chat_Analyze.pcap נניתן לראות את שלבי הփוטווקול:TCP

1. יצירת הקשר (Handshake): רואים את תהליך SYN, SYN-ACK, ACK.
2. העברת מידע (Data Transfer): הודעות הצ'אט (כגון שמות המשתמשים והתוכן) מועברות במנוגת עם דגל (PSH Push). מיד לאחר מכן מתקבל אישור ACK.

3. סיום קשר: בעת סגירת החלון, נשלחת מנת FIN/RST לסתירת Socket.

הודעת הטרפות:

הודעות בין משתמשים:

No.	Time	Source	Destination	Protocol	Length	Info
33	137.492006	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [ACK] Seq=7 Ack=61 Win=408320 Len=0 TSval=3654444784 TSecr=1036117591
34	150.007138	10.100.102.26	10.100.102.255	UDP	76	57621 → 57621 Len=44
35	160.409390	127.0.0.1	127.0.0.1	TCP	76	61018 → 12345 [PSH, ACK] Seq=9 Ack=113 Win=408320 Len=20 TSval=1373033355 TSecr=1464158800
36	160.409455	127.0.0.1	127.0.0.1	TCP	56	12345 → 61019 [ACK] Seq=113 Ack=26 Win=408320 Len=0 TSval=1464181717 TSecr=1373033354
37	160.409574	127.0.0.1	127.0.0.1	TCP	87	12345 → 61019 [PSH, ACK] Seq=61 Ack=7 Win=408320 Len=31 TSval=1036140508 TSecr=3654444784
38	160.409598	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [ACK] Seq=7 Ack=92 Win=408320 Len=0 TSval=36544467701 TSecr=1036140508
39	172.762773	127.0.0.1	127.0.0.1	TCP	74	61019 → 12345 [PSH, ACK] Seq=7 Ack=92 Win=408320 Len=18 TSval=3654480054 TSecr=1036140508
40	172.762840	127.0.0.1	127.0.0.1	TCP	56	12345 → 61019 [ACK] Seq=92 Ack=25 Win=408320 Len=0 TSval=1036152861 TSecr=3654480054
41	172.762941	127.0.0.1	127.0.0.1	TCP	87	12345 → 61018 [PSH, ACK] Seq=113 Ack=26 Win=408320 Len=31 TSval=1464194071 TSecr=1373033354
42	172.762962	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=26 Ack=144 Win=408320 Len=0 TSval=1373045708 TSecr=1464194071
43	180.007962	10.100.102.26	10.100.102.255	UDP	76	57621 → 57621 Len=44
44	186.297389	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [PSH, ACK] Seq=25 Ack=92 Win=408320 Len=0 TSval=3654493589 TSecr=1036152861
45	186.297373	127.0.0.1	127.0.0.1	TCP	56	12345 → 61019 [ACK] Seq=92 Ack=26 Win=408320 Len=0 TSval=1036166396 TSecr=3654493589
46	186.297587	127.0.0.1	127.0.0.1	TCP	98	12345 → 61018 [PSH, ACK] Seq=144 Ack=26 Win=408320 Len=34 TSval=1464207605 TSecr=1373045708
47	186.297617	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=26 Ack=178 Win=408320 Len=0 TSval=1373059242 TSecr=1464207605
48	186.297632	127.0.0.1	127.0.0.1	TCP	56	12345 → 61018 [FIN, ACK] Seq=92 Ack=26 Win=408320 Len=0 TSval=1036166396 TSecr=3654493589
49	186.297639	127.0.0.1	127.0.0.1	TCP	73	12345 → 61018 [PSH, ACK] Seq=178 Ack=26 Win=408320 Len=17 TSval=1464207605 TSecr=1373059242
50	186.297666	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [ACK] Seq=26 Ack=93 Win=408320 Len=0 TSval=3654493589 TSecr=1036166396
51	186.297670	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=26 Ack=195 Win=408320 Len=0 TSval=1373059242 TSecr=1464207605
52	210.008868	10.100.102.26	10.100.102.255	UDP	76	57621 → 57621 Len=44
53	215.846177	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [FIN, ACK] Seq=26 Ack=195 Win=408320 Len=0 TSval=1373088791 TSecr=1464207605
54	215.846246	127.0.0.1	127.0.0.1	TCP	56	12345 → 61018 [ACK] Seq=195 Ack=27 Win=408320 Len=0 TSval=1464237154 TSecr=1373088791
> Frame 41: Packet, 87 bytes on wire (696 bits), 87 bytes captured (696 bits)					0000 02 00 00 00 45 00 00 53 00 00 40 00 40 06 00 00	... E-S -@...
> Null/Loopback					0010 ff 00 00 01 7f 00 00 01 30 39 ee 50 8c 6b eb	... 9. k...
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1					0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	... R... G...
> Transmission Control Protocol, Src Port: 12345, Dst Port: 61018, Seq: 113, Ack: 26, Len: 3					0030 57 45 d4 17 51 d6 d3 8a 28 50 72 69 76 61 74 65	WE_Q... (Private
> Data (31 bytes)					0040 29 20 4d 69 63 68 61 6c 3a 20 48 65 6c 6c 6f 20) Michal: Hello
Data: 28507269766174652904d696368616c3a2048656c6c6f204c69726f79210a	[Length: 31]				0050 4c 69 72 fd 79 21 0a	Liroy!

No.	Time	Source	Destination	Protocol	Length	Info
29	137.491991	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [ACK] Seq=9 Ack=65 Win=408320 Len=0 TSval=1373010437 TSecr=1464158800
30	137.491997	127.0.0.1	127.0.0.1	TCP	80	12345 → 61018 [PSH, ACK] Seq=89 Ack=66 Win=408320 Len=24 TSval=1464158800 TSecr=1036117591
31	137.492001	127.0.0.1	127.0.0.1	TCP	80	12345 → 61019 [PSH, ACK] Seq=37 Ack=7 Win=408320 Len=24 TSval=1036117591 TSecr=3654444784
32	137.492006	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=6 Ack=113 Win=408320 Len=0 TSval=1373018437 TSecr=1464158800
33	137.492006	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [ACK] Seq=7 Ack=61 Win=408320 Len=0 TSval=3654444784 TSecr=1036117591
34	150.007138	10.100.102.26	10.100.102.255	UDP	76	57621 → 57621 Len=44
35	160.409390	127.0.0.1	127.0.0.1	TCP	76	61018 → 12345 [PSH, ACK] Seq=6 Ack=113 Win=408320 Len=20 TSval=1373033354 TSecr=1464158800
36	160.409455	127.0.0.1	127.0.0.1	TCP	56	12345 → 61018 [ACK] Seq=113 Ack=26 Win=408320 Len=0 TSval=1464181717 TSecr=1373033354
37	160.409574	127.0.0.1	127.0.0.1	TCP	87	12345 → 61019 [PSH, ACK] Seq=61 Ack=7 Win=408320 Len=31 TSval=1036140508 TSecr=3654444784
38	160.409598	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [ACK] Seq=92 Win=408320 Len=0 TSval=3654467701 TSecr=1036140508
39	160.409673	127.0.0.1	127.0.0.1	TCP	74	61019 → 12345 [PSH, ACK] Seq=92 Ack=69 Win=408320 Len=18 TSval=3654480054 TSecr=1036140508
40	172.762840	127.0.0.1	127.0.0.1	TCP	56	12345 → 61019 [ACK] Seq=92 Ack=25 Win=408320 Len=0 TSval=1036152861 TSecr=3654480054
41	172.762941	127.0.0.1	127.0.0.1	TCP	87	12345 → 61018 [PSH, ACK] Seq=113 Ack=26 Win=408320 Len=31 TSval=1464194071 TSecr=1373033354
42	172.762962	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=26 Ack=144 Win=408320 Len=0 TSval=1373045708 TSecr=1464194071
43	180.007962	10.100.102.26	10.100.102.255	UDP	76	57621 → 57621 Len=44
44	186.297389	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [FIN, ACK] Seq=25 Ack=92 Win=408320 Len=0 TSval=3654493589 TSecr=1036152861
45	186.297373	127.0.0.1	127.0.0.1	TCP	56	12345 → 61019 [ACK] Seq=92 Ack=26 Win=408320 Len=0 TSval=1373045708 TSecr=1464207605
46	186.297587	127.0.0.1	127.0.0.1	TCP	98	12345 → 61018 [PSH, ACK] Seq=144 Ack=26 Win=408320 Len=34 TSval=1464207605 TSecr=1373045708
47	186.297617	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=26 Ack=178 Win=408320 Len=0 TSval=1373059242 TSecr=1464207605
48	186.297632	127.0.0.1	127.0.0.1	TCP	56	12345 → 61018 [FIN, ACK] Seq=92 Ack=26 Win=408320 Len=0 TSval=1036166396 TSecr=3654493589
49	186.297639	127.0.0.1	127.0.0.1	TCP	73	12345 → 61018 [ACK] Seq=178 Ack=26 Win=408320 Len=17 TSval=1464207605 TSecr=1373059242
> Frame 37: Packet, 87 bytes on wire (696 bits), 87 bytes captured (696 bits)					0000 02 00 00 00 45 00 00 53 00 00 40 00 40 06 00 00	... E-S -@...
> Null/Loopback					0010 ff 00 00 01 7f 00 00 01 30 39 ee 50 8c 6b eb	... 9. k...
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1					0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	... R... G...
> Transmission Control Protocol, Src Port: 12345, Dst Port: 61019, Seq: 61, Ack: 7, Len: 31					0030 c4 b3 36 c4 80 18 18 ec fe 47 00 00 01 00 00	= 6... G...
> Data (31 bytes)					0040 3d c2 3f dc d9 62 66 f0 28 50 72 69 76 61 74 65	= ?-f. (Private
Data: 28507269766174652904d696368616c3a2048656c6c6f204c69726f79210a	[Length: 31]				0050 69 63 68 61 6c 21 0a) Liroy: Hello M
>						ichall!

הודעת עזיבת משתמש מהצ'אט:

Apply a display filter ... <幫/>

No.	Time	Source	Destination	Protocol	Length	Info
35	160.409398	127.0.0.1	127.0.0.1	TCP	76	61018 → 12345 [PSH, ACK] Seq=6 Ack=113 Win=408320 Len=20 TSval=1373033354 TSecr=1464158800
36	160.409455	127.0.0.1	127.0.0.1	TCP	56	12345 → 61018 [ACK] Seq=113 Ack=26 Win=408320 Len=0 TSval=1464181717 TSecr=1373033354
37	160.409574	127.0.0.1	127.0.0.1	TCP	87	12345 → 61019 [PSH, ACK] Seq=61 Ack=7 Win=408320 Len=31 TSval=1036140508 TSecr=3654444784
38	160.409598	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [ACK] Seq=7 Ack=92 Win=408320 Len=0 TSval=3654467701 TSecr=1036140508
39	172.762773	127.0.0.1	127.0.0.1	TCP	74	61019 → 12345 [PSH, ACK] Seq=7 Ack=92 Win=408320 Len=18 TSval=3654480054 TSecr=1036140508
40	172.762840	127.0.0.1	127.0.0.1	TCP	56	12345 → 61019 [ACK] Seq=92 Ack=25 Win=408320 Len=0 TSval=1036152861 TSecr=3654480054
41	172.762941	127.0.0.1	127.0.0.1	TCP	87	12345 → 61018 [PSH, ACK] Seq=113 Ack=26 Win=408320 Len=31 TSval=1464194071 TSecr=1373033354
42	172.762963	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=26 Ack=144 Win=408320 Len=0 TSval=1373045708 TSecr=1464194071
43	180.007962	10.100.102.255	10.100.102.255	UDP	76	57621 → 57621 Len=44
44	186.297309	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [FIN, ACK] Seq=25 Ack=92 Win=408320 Len=0 TSval=3654493589 TSecr=1036152861
45	186.297373	127.0.0.1	127.0.0.1	TCP	56	12345 → 61019 [ACK] Seq=92 Ack=26 Win=408320 Len=0 TSval=1036166396 TSecr=3654493589
46	186.297587	127.0.0.1	127.0.0.1	TCP	90	12345 → 61018 [PSH, ACK] Seq=144 Ack=26 Win=408320 Len=34 TSval=1464207605 TSecr=1373045708
47	186.297617	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=26 Ack=178 Win=408320 Len=0 TSval=1373059242 TSecr=1464207605
48	186.297632	127.0.0.1	127.0.0.1	TCP	56	12345 → 61019 [FIN, ACK] Seq=92 Ack=26 Win=408320 Len=0 TSval=1036166396 TSecr=3654493589
49	186.297639	127.0.0.1	127.0.0.1	TCP	73	12345 → 61018 [PSH, ACK] Seq=178 Ack=26 Win=408320 Len=0 TSval=1464207605 TSecr=1373059242
50	186.297666	127.0.0.1	127.0.0.1	TCP	56	61019 → 12345 [ACK] Seq=26 Ack=93 Win=408320 Len=0 TSval=3654493589 TSecr=1036166396
51	186.297670	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=26 Ack=195 Win=408320 Len=0 TSval=1373059242 TSecr=1464207605
52	210.008866	10.100.102.255	10.100.102.255	UDP	76	57621 → 57621 Len=44
53	215.846177	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [FIN, ACK] Seq=26 Ack=195 Win=408320 Len=0 TSval=1373088791 TSecr=1464207605
54	215.846246	127.0.0.1	127.0.0.1	TCP	56	12345 → 61018 [ACK] Seq=195 Ack=27 Win=408320 Len=0 TSval=1464237154 TSecr=1373088791
55	215.846482	127.0.0.1	127.0.0.1	TCP	56	12345 → 61018 [FIN, ACK] Seq=195 Ack=27 Win=408320 Len=0 TSval=1464237154 TSecr=1373088791
56	215.846537	127.0.0.1	127.0.0.1	TCP	56	61018 → 12345 [ACK] Seq=27 Ack=196 Win=408320 Len=0 TSval=1373088791 TSecr=1464237154

```
> Frame 46: Packet, 90 bytes on wire (720 bits), 90 bytes captured (720 bits)
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 12345, Dst Port: 61018, Seq: 144, Ack: 26, Len: 3
Data: 5365727665723a204d696368616c20686173206c6566742074686520636861742e0a
[Length: 34]
```

0000 02 00 00 00 45 00 00 56 00 00 40 00 40 00 00 00 E· V ·@ @ ··
0010 7f 00 00 01 7f 00 00 01 30 39 ee 5a 8c 6b ec 05 ··· 09 Z k ··
0020 fb b3 03 48 80 18 18 ec fe 4a 00 00 01 01 08 0a ··· H ··· J ···
0030 57 46 08 f5 51 d7 03 cc 53 65 72 76 65 72 3a 20 WF · 0 ··· Server:
0040 4d 69 63 68 61 6c 20 68 61 73 20 6c 65 66 74 20 Michal h as left
0050 74 68 65 20 63 68 61 74 2e 0a the chat ..

Data (data.data), 34 bytes

Packets: 56

Profile: Default

חלק 3: שימוש בבינה מלאכותית (AI)

במסגרת העבודה נעשה שימוש בכלים הבינה המלאכותית Google Gemini ככלי עזר לפיתוח ולמידה. השימוש בוצע באופן מבוקר תוך בדיקת התוצאות.

מטרות השימוש ודוגמאות לפורומפטים:

1. **יצירת נתונים (חלק 1)**: (יצירת קובץ CSV-עם נתונים דמה ריאליים לשימולציה).

Prompt Example: "Create a CSV file with columns: msg_id, app_protocol, src_app, dst_app, message, timestamp. Generate 10 rows representing HTTP GET requests."

2. **פיתוח השירות (Server)**: סיעם כתיבת הקוד לטיפול בריבוי לקוחות.

Prompt Example: "Write a Python TCP server using 'socket' and 'threading' to handle multiple clients simultaneously."

3. **עיצוב הממשק (GUI)**: עזרה לבניית הממשק הגרפי ב Tkinter-וסידור האלמנטים בחלון.

Prompt Example: "Create a GUI chat client in Python using tkinter with a scrollable text area and an input field"