



פרויקט גמר

קורס רשתות תקשורת מחשבים



ניתוח תעבורה בפרוטוקול TCP/IP

מגישות:

מאור סבטני - 213119282

מאי-סוליי סמכה - 212408025

אופל אסרף - 212713390

מרצה:

רויטל מרבב

דוח מסכםחלק 1 - אריזת נתונים ולכידת מנות בעזרת Wireshark1. יצירת קובץ CSV - שכבת היישום:

בשלב הראשון בחרנו להשתמש בפרוטוקול DNS בפרויקט. יצרנו קובץ CSV המכיל שורות של הודעות מפרוטוקול DNS, כאשר כל שורה מייצגת הודעה אחת ברצף התקשורת:

msg_id	app_protocol	src_port	dst_port	message	timestamp
1	DNS	55001	53	Query www.server	0.012
2	DNS	55001	53	Response client	0.024
3	DNS	55002	53	Query www.server	0.035
4	DNS	55002	53	Response client	0.048
5	DNS	55003	53	Query www.server	0.059
6	DNS	55003	53	Response client	0.073
7	DNS	55004	53	Query www.server	0.082
8	DNS	55004	53	Response client	0.096
9	DNS	55005	53	Query www.server	0.108

קובץ ה-CSV כולל את השדות הבאים:

- msg_id - מזהה הודעה
- app_protocol - פרוטוקול היישום (DNS)
- src_port - פורט מקור
- dst_port - פורט יעד
- message - תוכן ההודעה (Query / Response)
- timestamp - זמן יחסי לשליחת ההודעה

הקובץ משמש כקלט למחברת Jupyter ומדמה תקשורת בין לקוח DNS לשרת DNS. השדות src_port ו-dst_port נכללו בקובץ ה-CSV לצורך התאמה לשימוש במחברת Jupyter, אשר עושה שימוש בפורטים כחלק מתהליך יצירת חבילות התעבורה. בנוסף, נעזרנו במעט בבינה המלאכותית.

2. עבודה במחברת Jupyter:

את קובץ ה-CSV טענו למחברת Jupyter. המחברת מריצה כל שורה בקובץ ומדגימה את תהליך האריזה של המידע בין שכבות הרשת:

- שכבת היישום (Application) - הודעת DNS מתוך ה-CSV
- שכבת התעבורה (Transport) - עטיפה בפרוטוקול TCP
- שכבת הרשת (Network) - עטיפה בפרוטוקול IP
- שכבת הקישור (Data Link) - שליחה דרך ממשק Loopback

יצירת תעבורת רשת:

באמצעות Python וספריית Scrapy נוצרו חבילות TCP אמיתיות, כאשר תוכן ההודעות מתוך קובץ ה-CSV משמש כדאטה של החבילות. החבילות נשלחו דרך ממשק ה-Loopback (127.0.0.1) אל פורט יעד 53, לצורך הדמיית תקשורת DNS.

3. לכידת התעבורה באמצעות Wireshark:

לכידת התעבורה בוצעה באמצעות Wireshark על ממשק Npcap Loopback Adapter. הגדרנו פילטר תצוגה: `tcp.port == 53` כך שיהיה לנו קל יותר לגשת לפרוטוקול הרלוונטי לנו שהוא DNS. במהלך הרצת המחברת נלכדו חבילות TCP שנשלחו מקוד המחברת. באמצעות Wireshark לכדנו ושמרנו את התעבורה.

ניתוח התעבורה שנלכדה:

בלכידה ראינו מספר דברים חשובים בניהם: חבילות בפרוטוקול TCP, כתובת מקור ויעד: 127.0.0.1, פורט המקור ופורט יעד 53, החבילות כוללות תגי TCP מסוג PSH ו-ACK, המציגים את שליחת נתונים, ובנוסף, חבילות עם התג RST, חבילות אלה נשלחות כתוצאה מהיעדר שרת DNS פעיל המאזין ב-TCP על פורט זה. נתונים אשר מייצגים את הודעות DNS (Query / Response).

מצורפים צילומי מסך לניתוח התעבורה שנלכדה:
חבילות TCP -> לפורט 53 (DNS):

group99_dns_capture.pcap

tcp.port == 53

No.	Time	Source	Destination	Protocol	Length	Info
6256	75.681792	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Ack=1 Win=8192 Len=16 [PSH, ACK] 53 → 37586
6257	75.681898	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Win=0 Len=0 [RST] 37586 → 53 44
6258	75.785065	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Ack=1 Win=8192 Len=15 [TCP Retransmission] 59
6259	75.785174	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Win=0 Len=0 [RST] 37586 → 53 44
6288	75.888232	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Ack=1 Win=8192 Len=16 [TCP Retransmission] 60
6289	75.888303	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Win=0 Len=0 [RST] 37586 → 53 44
6290	75.991824	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Ack=1 Win=8192 Len=15 [TCP Retransmission] 59
6291	75.991938	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Win=0 Len=0 [RST] 37586 → 53 44
6292	76.094846	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Ack=1 Win=8192 Len=16 [TCP Retransmission] 60
6293	76.094947	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Win=0 Len=0 [RST] 37586 → 53 44
6294	76.196766	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Ack=1 Win=8192 Len=15 [TCP Retransmission] 59
6295	76.196819	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Win=0 Len=0 [RST] 37586 → 53 44
6296	76.299942	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Ack=1 Win=8192 Len=16 [TCP Retransmission] 60
6297	76.300032	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Win=0 Len=0 [RST] 37586 → 53 44
6338	76.404002	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Ack=1 Win=8192 Len=15 [TCP Retransmission] 59
6339	76.404113	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Win=0 Len=0 [RST] 37586 → 53 44
6340	76.509504	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Ack=1 Win=8192 Len=16 [TCP Retransmission] 60
6341	76.509710	127.0.0.1	127.0.0.1	TCP	60	Seq=1 Win=0 Len=0 [RST] 37586 → 53 44

Frame 6256: Packet, 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on Null/Loopback
 Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
 Transmission Control Protocol, Src Port: 37586, Dst Port: 53, Seq: 1, Ack: 1, Len: 16

החבילות המסומנות באדום מעידות על חבילות הכוללות דגלי RST (אין שרת DNS שמאזין ב-TCP על ה- Loopback) או על תעבורה חריגה/כישלון חיבור. סימון זה מאפשר זיהוי מהיר של אופי התקשורת והבנת מצב החיבור.

ניתוח Frame ומעטפת הלכידה:

Frame 6256: Packet, 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on Null/Loopback

Encapsulation type: NULL/Loopback (15)

Arrival Time: Dec 17, 2025 11:22:07.247889000
 UTC Arrival Time: Dec 17, 2025 09:22:07.247889000 UTC
 Epoch Arrival Time: 1765963327.247889000
 [Time shift for this packet: 0.000000000 seconds]
 [Time delta from previous captured frame: 1.358000 milliseconds]
 [Time since reference or first frame: 1 minute, 15.681792000 seconds]

Frame Number: 6256
 Frame Length: 60 bytes (480 bits)
 Capture Length: 60 bytes (480 bits)
 [Frame is marked: False]
 [Frame is ignored: False]
 [Protocols in frame: null:ip:tcp]
 Character encoding: ASCII (0)
 [Coloring Rule Name: TCP]
 [Coloring Rule String: tcp]

צילום המסך מוצגת חבילת רשת (Frame) שנלכדה באמצעות Wireshark. הלכידה בוצעה על ממשק Loopback, כפי שניתן לראות בשדה Encapsulation type: NULL/Loopback. אורך החבילה הוא 60 בתים, והחבילה נלכדה במלואה. ברשימת הפרוטוקולים ניתן לראות את תהליך האריזה (Encapsulation): Loopback-> IP -> TCP, המדגים את מעבר המידע משכבת הקישור ועד שכבת התעבורה.

ניתוח שכבת הרשת (IP):

```
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
  Version: 4 = .... 0100
  Header Length: 20 bytes (5) = 0101 ....
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT) <
    Total Length: 56
    Identification: 0x0001 (1)
    Flags: 0x0 = .... 0000 <
    Fragment Offset: 0 = 0000 0000 0000 0...
    Time to Live: 64
    Protocol: TCP (6)
    Header Checksum: 0x7cbd [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 127.0.0.1
    Destination Address: 127.0.0.1
    [Stream index: 0]
```

בצילום המסך הבא מוצגת שכבת הרשת של החבילה בפרוטוקול. כתובת המקור וכתובת היעד זהות והן 127.0.0.1, דבר היכול להעיד על תעבורה פנימית. אורך כותרת ה-IP הוא 20 בתים, ללא פרגמנטציה של החבילות (Fragment Offset = 0). ערך ה time to live הוא 64 ומצביע על חבילה תקינה שלא עברה דרך נתבים. השדה Protocol מצביע על TCP, ולכן הנתונים מועברים לשכבת התעבורה.

ניתוח שכבת התעבורה (TCP):

```
Transmission Control Protocol, Src Port: 37586, Dst Port: 53, Seq: 1, Ack: 1, Len: 16
  Source Port: 37586
  Destination Port: 53
  [Stream index: 54]
  [Stream Packet Number: 1]
  [Conversation completeness: Incomplete (40)] <
  [TCP Segment Len: 16]
  Sequence Number: 1 (relative sequence number)
  Sequence Number (raw): 0
  [Next Sequence Number: 17 (relative sequence number)]
  Acknowledgment Number: 1 (relative ack number)
  Acknowledgment number (raw): 0
  Header Length: 20 bytes (5) = .... 0101
  Flags: 0x018 (PSH, ACK) <
  Window: 8192
  [Calculated window size: 8192]
  [Window size scaling factor: -1 (unknown)]
  Checksum: 0x94b6 [unverified]
  [Checksum Status: Unverified]
```

בצילום המסך מוצגת שכבת התעבורה בפרוטוקול TCP. החבילה נשלחת מפורט מקור 37586 (פורט זמני של הלקוח) אל פורט יעד 53, המשמש לתקשורת DNS. מספר הרצף (Sequence Number) הוא 1, ואורך מקטע ה-TCP הוא 16 בתים, המעיד על שליחת נתונים בפועל. דגלי ה-TCP המסומנים הם PSH ו-ACK, המעידים על העברת מידע מיידית ואישור קבלה של נתונים קודמים. נתונים אלו מצביעים על תקשורת TCP פעילה ותקינה כחלק מתהליך העברת הודעת DNS.

קישור בין CSV ללכידת Wireshark:

כל חבילת TCP שנלכדה מייצגת הודעת DNS אחת מתוך קובץ ה־CSV. ניתן לקשר בין שורות כגון msg_id, תוכן ההודעה (message) והזמן היחסי (timestamp) לבין החבילות שנצפו ב־Wireshark, ובכך לעקוב אחר תהליך העברת המידע משכבת היישום ועד לרשת.

חלק 2 - כתיבת יישום רשת וניתוח תעבורה של אותו יישום

1. מבוא - תיאור כללי של המערכת

בחלק זה התבקשנו לתכנן מערכת צ'אט מבוססת Sockets תוך שימוש במטרה להבין את עקרונות התקשורת ברשת במודל של Client-Server. המערכת שיצרנו נכתבה בשפת פייתון אשר היא שפה קלה להבנה ומאפשרת לבצע את תכנות המודל בצורה נוחה לכל מתכנת. המערכת שיצרנו מאפשרת למספר לקוחות להתחבר לשרת בזמן-אמת, להזדהות באמצעות שם משתמש ייחודי, ולשלוח הודעות טקסט ללקוחות אחרים בזמן אמת. השרת אחראי לניהול החיבורים, לניתוב ההודעות בין הלקוחות.

2. מבנה המערכת

2.1 מבנה כללי

התקשורת מתבצעת בפרוטוקול TCP, על מנת להבטיח אמינות, סדר והגעה מלאה של ההודעות. המערכת בנויה משני רכיבים עיקריים:

- **Server** - השרת מאזין לחיבורים נכנסים, מנהל לקוחות ומעביר את הודעות בין הלקוחות.
- **Client** - הלקוח אשר מתחבר לשרת, שולח הודעות ומקבל הודעות מלקוחות אחרים.

2.2 הסבר על קובץ השרת (Server)

בקוד שיצרנו השתמשנו בספריות socket, threading שימוש בספריות תקשורת וניהול תהליכים מקבילים מאפשרים לשרת ליצור תקשורת עם כמה משתמשים במקביל ללא הפרעות. לאחר הגדרת הספריות לשימוש יצרנו מילון CLIENTS אשר באמצעות השרת שיצרנו שומר את הלקוחות המשתמשים כרגע ובאמצעותו ממפה את התעבורה בין הלקוחות. בנוסף, יצרנו את SENDER_LAST מטרת מנגנון זה נועד לעזור לשרת לזכור מי שלח הודעה למי. אם לקוח קיבל הודעה ואין לו יעד מוגדר באותו רגע, השרת יאפשר לו פשוט להקליד הודעה והיא תישלח אוטומטית לאדם האחרון שפנה אליו. מטרת מנגנון זה ליצור חווית משתמש רציפה. יתרה מזאת, הוספנו את מנגנון LOCK בגלל שהשרת מנהל משתמשים רבים במקביל, עלול להיווצר מצב שבו שני תהליכים (Threads) מנסים לעדכן את רשימת המשתמשים בדיוק באותו הזמן. ה Lock-מבטיח שרק תהליך אחד יתבצע ולא כמה במקביל, דבר המונע קריסות או שיבוש נתונים. לשרת יש כמה משימות חשובות בניהול התעבורה, שלב ראשון - הרצה והמתנה למשתמשים. השרת פותח "שער" בכתובת שהחלטנו (127.0.0.1) ופורט 8053, לאחר מכן השרת נכנס ללולאה אינסופית אשר הוא ממתין לחיבור של לקוחות. כאשר לקוח מתחבר השרת פותח עבורו THREAD וממשיך להמתין ללקוחות נוספים. בשלב הבא, השרת רושם את המשתמש - השרת מבקש מהלקוח להכניס שם משתמש בודק כי השם תקין או לא תפוס ולאחר מכן הוא מוסיף אותו למילון הלקוחות שיצרנו ומבקש ממנו לבחור לקוח לדבר איתו. לאחר שהתחברו מספר לקוחות זהו שלב בו מערכת הצ'אטים עובדת, השרת שואל את הלקוח

עם מי הוא מתכוון לדבר, אם הלקוח קיים השיח מתחיל. בנוסף בשלב זה, השרת לא רק מעביר טקסט אלא גם מנתח את הטקסט. אם הלקוח משתמש בסימן '@' לפני שם של משתמש, השרת מזהה שמדובר ב'תיוג' של משתמש אחר. תכונה זו מאפשרת לשלוח הודעה דחופה למשתמש אחר מבלי להפסיק את השיחה הנוכחית או להחליף את היעד הקבוע. במידה והלקוח אינו קיים, המשתמש נשאר בלולאה ומבקש שוב שם לשיחה, ובמידה והלקוח מקליד exit הלקוח מתנתק מהשרת. ברגע שנבחר יעד השרת מעביר את מה שלקוח א' כתב ללקוח ב'. בצד המקבל מופיע שם הלקוח השולח לפני ההודעה. בנוסף, במקרה ולקוח רוצה להחליף לקוח איתו הוא מעוניין לדבר הוא יכול לעשות זאת על ידי הקלדת 'change' אשר מחזיר את הלקוח לבחור לקוח לדבר איתו.

2.3 הסבר על קובץ הלקוח (Client)

תפקיד קובץ הלקוח הוא לאפשר למשתמש קצה להתחבר לשרת, לקבל ולשלוח הודעות בזמן אמת לאנשים אחרים המחוברים לשרת. כמו בקובץ השרת יש שימוש בספריות socket, threading. הלקוח משתמש בכתובת IP המקומית 127.0.0.1 ובפורט 8053 שבאמצעות מתחבר לשרת. השורה client.connect מטרתה ליצור חיבור בין הלקוח לשרת. Thread-נפתח ומתחיל להמתין להודעות. המשתמש מקליד שם/הודעה ב, input-send מעביר את זה לשרת, והשרת מחזיר תשובה שמודפסת על ידי ה-Thread. בתוכנת צ'יט יש בעיה-אם הלקוח מחכה שהמשתמש יקליד הודעה, (input) הוא לא יכול באותו זמן להקשיב לשרת. כדי לפתור את בעיה זאת פיצלנו את הלולאות לשניים. receive_messages לולאה המחכה להודעות נכנסות מהשרת, וברגע שמגיעה הודעה היא מודפסת מיד. ובנוסף, מריצים את הלולאה האחרונה. אשר מחכה שהמשתמש יכתוב משהו כדי לשלוח אותו לשרת. כדי לאפשר זרימה של ההודעות השתמשנו ב-daemon=True והגדרנו שהוא תלוי בתוכנית של שליחת ההודעות. משמעות הדבר - שברגע שנסגור את התוכנית הראשית (למשל על ידי סגירת החלון), ההאזנה להודעות ייסגר אוטומטית. בלי זה, התוכנית עלולה להישאר "תקועה" ברקע. אם השרת קורס או הקשר מתנתק, ה-try-except בתוך פונקציית הקבלה תופס את השגיאה ועוצר את הלולאה בצורה נקייה.

3. הוראות התקנה והרצה

3.1 דרישות מוקדמות

- Python 3.x
- מערכת הפעלה Windows / Linux / macOS

3.2 הרצת השרת

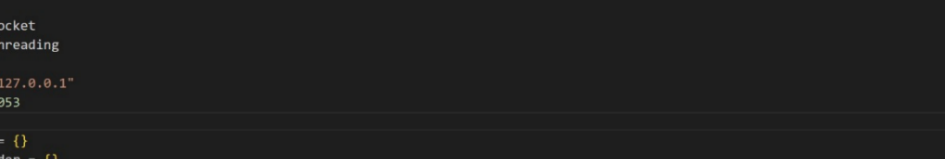
1. פתיחת חלון Terminal / CMD
2. ניווט לתיקיית הפרויקט
3. להריץ: `python server.py`
4. לאחר ההרצה תופיע הודעה שהשרת מאזין לפורט.

3.3 הרצת לקוחות

1. לפתוח חלון Terminal נוסף (לכל לקוח)
2. להריץ: `python client.py`
3. להזין שם משתמש ייחודי
4. להתחיל התכתבות עם לקוחות נוספים

4. דוגמאות קלט-פלט (Screenshots)

צילום 1 - שרת ה-TCP פועל ומאזין לחיבורים נכנסים, לאחר התחברות של מספר לקוחות.



The screenshot shows a VS Code editor with two tabs: 'server.py' and 'client.py'. The 'server.py' tab is active, displaying a Python script for a multi-threaded server. The script imports 'socket' and 'threading', sets 'HOST = "127.0.0.1"' and 'PORT = 8053', and uses a dictionary 'clients' to track connections. A 'handle_client' function is defined to process incoming requests. The terminal at the bottom shows the command 'python server.py' being executed, with output indicating the server is listening on port 8053 and successfully connecting to five clients: maor, opal, may, bar, and revival.

```
server.py > ...
1 import socket
2 import threading
3
4 HOST = "127.0.0.1"
5 PORT = 8053
6
7 clients = {}
8 last_sender = {}
9 lock = threading.Lock()
10
11 def handle_client(client_socket, addr):
12     username = ""
13     try:
14
15         while True:
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

PS C:\Users\Maor\OneDrive - Holon Institute of Technology\מסלול\סקרופ\הדוגמה תוחסר - עסיבשחמ תרושקת תוחסר python server.py
SERVER LISTENING on port 8053...
[CONNECTED] maor
[CONNECTED] opal
[CONNECTED] may
[CONNECTED] bar
[CONNECTED] revival
}

צילום 2 - התחברות מספר לקוחות לשרת באמצעות היישום Client, הזדהות עם שם משתמש והמתנה לבחירת יעד לשיחה:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
```

```
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\מחלקת תרופות> python client.py
Enter your username:
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\מחלקת תרופות> python client.py
Enter your username:
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\מחלקת תרופות> python client.py
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\מחלקת תרופות> python client.py
Enter your username:
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\מחלקת תרופות> python client.py
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\מחלקת תרופות> python client.py
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\מחלקת תרופות> python client.py
Enter your username:
Maor

--> Enter username to chat with (or type 'exit' to quit):
[ ]
```

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
Enter your username:
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
Enter your username:
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
Enter your username:
Opal
--> Enter username to chat with (or type 'exit' to quit):

```

צילום 3 - התכתבות דו-כיוונית בין שני לקוחות דרך השרת עם מנגנון תגובה אוטומטית:

מאור בוחרת לשלוח למאי הודעה:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
Enter your username:
Maor

--> Enter username to chat with (or type 'exit' to quit):
May
--- Chat started with may. Type '/change' to switch user ---
Tip: send to a specific user using @username (example: hi @may)

Hey May:)

```

נפתח צ'אט - ההודעה מגיעה למאי, והיא עונה לה באופן אוטומטי ללא צורך בבחירה:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
Enter your username:
May

--> Enter username to chat with (or type 'exit' to quit):

[maor]: Hey May:)

Hey Maor, how are you today?

```

התשובה של מאי מגיעה חזרה למאור- וגם מאור עונה למאי באופן אוטומטי:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקרין\הדוגמה\תלוי\python client.py
Enter your username:
Maor

--> Enter username to chat with (or type 'exit' to quit):
May
--- Chat started with may. Type '/change' to switch user ---
Tip: send to a specific user using @username (example: hi @may)

Hey May:)

[may]: Hey Maor, how are you today?

I'm fine, working on a special project. how about you?

```

המשך זרימת ההתכתבות בין 2 הקליינטים (מאור ומאי):

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology> python client.py
Enter your username:
May

--> Enter username to chat with (or type 'exit' to quit):

[maor]: Hey May:)

Hey Maor, how are you today?

[maor]: I'm fine, working on a special project. how about you?

Sounds interesting, actually I'm working on something too:)
```

צילום 4 - התכתבות בין מספר לקוחות במקביל, כולל ניתוב הודעות באמצעות @username והעברת הודעות דרך השרת:

שימוש ב-@ על מנת לפנות ספציפית ללקוח אחר בשרת:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology> python client.py
Enter your username:
Maor

--> Enter username to chat with (or type 'exit' to quit):
May
--- Chat started with may. Type '/change' to switch user ---
Tip: send to a specific user using @username (example: hi @may)

Hey May:)

[may]: Hey Maor, how are you today?

I'm fine, working on a special project. how about you?

[may]: Sounds interesting, actually I'm working on something too:)

Hey Opal, i miss you @opal
```

ניתן לראות שההודעה הגיעה ללקוח הספציפי @אופל (ולא למאי שנמצאת בצ'אט עם מאור)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology> python client.py
Enter your username:
Opal

--> Enter username to chat with (or type 'exit' to quit):

[maor]: Hey Opal, i miss you

Hi:) i miss you too
```

מאור מקבלת הודעות מ-2 משתמשים, ועונה לשניהם:

תגובה אוטומטית לאופל (הלקוח ששלח את ההודעה האחרונה בציאט של מאור)

ותגובה ספציפית עם @ למאי:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקיינר\הדיובה וזליו\python client.py
Enter your username:
Maor

--> Enter username to chat with (or type 'exit' to quit):
May
--- Chat started with may. Type '/change' to switch user ---
Tip: send to a specific user using @username (example: hi @may)

Hey May:)

[May]: Hey Maor, how are you today?

I'm fine, working on a special project. how about you?

[May]: Sounds interesting, actually I'm working on something too:)

Hey Opal, i miss you @opal

[Opal]: Hi:) i miss you too

Oh, let's meet this weekend!!
Good luck, sister. @may

```

ניתן לראות שכל הודעה הגיע ליעד המתאים לה:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקיינר\הדיובה וזליו\python client.py
Enter your username:
Opal

--> Enter username to chat with (or type 'exit' to quit):

[maor]: Hey Opal, i miss you

Hi:) i miss you too

[maor]: Oh, let's meet this weekend!!

[]

```

המערכת תומכת בחיבור של לפחות 5 לקוחות במקביל, וניתנת להרחבה למספר לקוחות נוסף ללא שינוי מהותי בקוד.

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Maor\OneDrive - Holon Institute of Technology\סקיינר\הדיובה וזליו\python client.py
Enter your username:
May

--> Enter username to chat with (or type 'exit' to quit):

[maor]: Hey May:)

Hey Maor, how are you today?

[maor]: I'm fine, working on a special project. how about you?

Sounds interesting, actually I'm working on something too:)

[maor]: Good luck, sister.

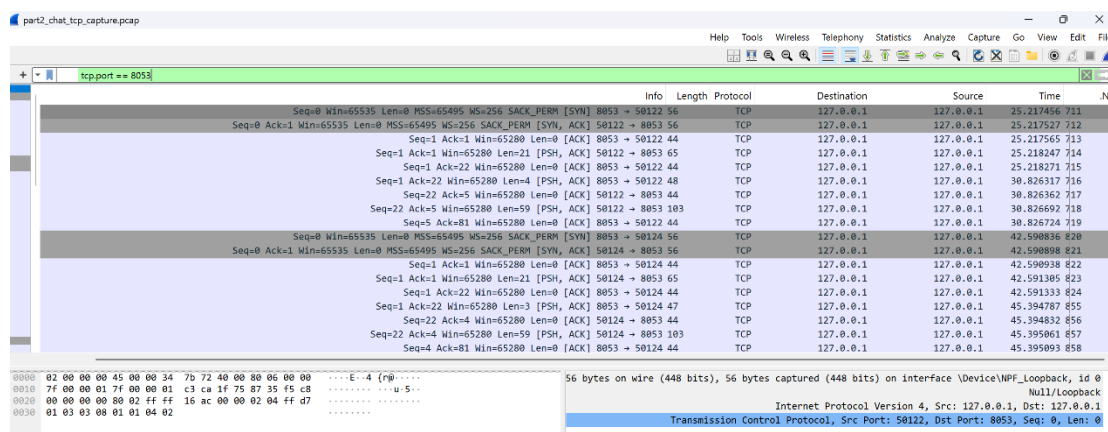
[]

```

5. ניתוח תעבורת רשת (Wireshark) :

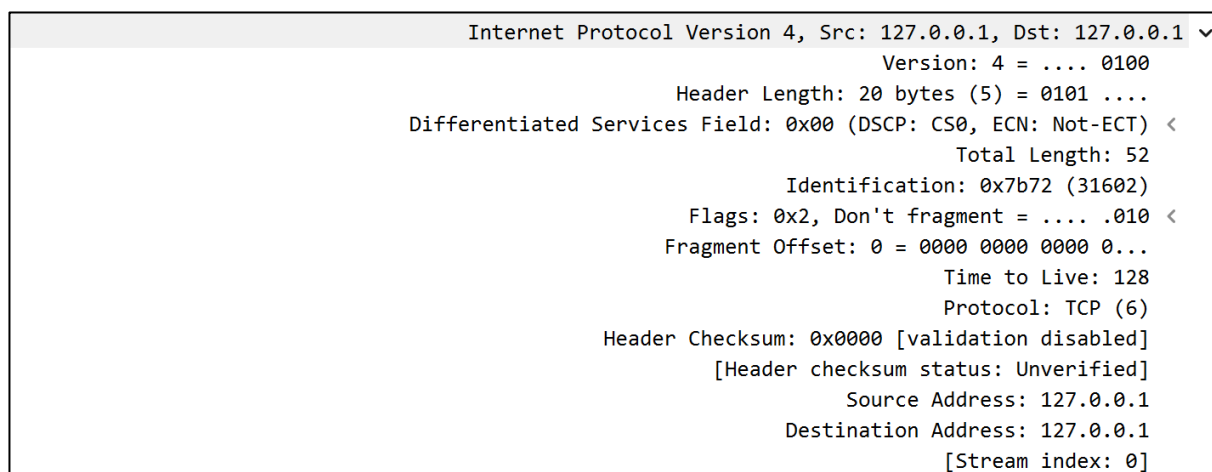
לצורך ניתוח תעבורת הרשת של מערכת הצ'אט, בוצעה לכידת תעבורה באמצעות תוכנת Wireshark במהלך הרצת השרת והקליינטיים. הלכידה בוצעה על ממשק loopback, מאחר והשרת והלקוחות פועלים על אותו מחשב. לאחר סיום הלכידה בוצע סינון לפי הפורט של השרת (tcp.port == 8053).

צילום 1- תעבורת TCP של יישום הצ'אט לאחר סינון לפי פורט השרת 8053:



בצילום זה ניתן לראות את כלל חבילות ה-TCP השייכות ליישום הצ'אט. התעבורה מסוננת לפי הפורט 8053, עליו מאזין השרת. ניתן להבחין בחבילות מסוג SYN, SYN-ACK ו-ACK המעידות על הקמת חיבור TCP, וכן חבילות PSH/ACK המעידות על העברת נתונים (הודעות צ'אט) בין הלקוחות לשרת.

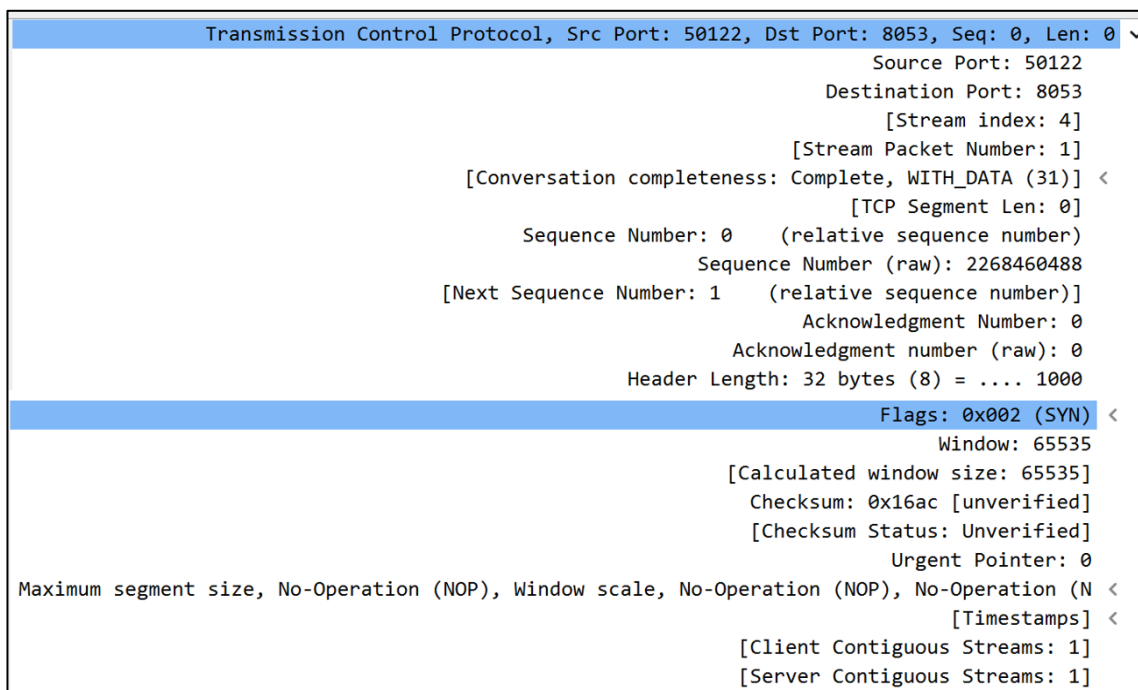
צילום 2- שכבת הרשת (IP):



בצילום זה מוצגת שכבת הרשת (IP) של אחת החבילות. ניתן לראות כי מדובר בפרוטוקול IPv4, כאשר כתובת המקור וכתובת היעד הן 127.0.0.1, דבר המעיד על תקשורת מקומית (loopback).

שדה ה-TTL מוגדר לערך 128, והחבילה אינה מפורקת (Fragment Offset == 0). בנוסף, שדה ה-Protocol מציין כי הנתונים מועברים לפרוטוקול TCP, בהתאם למימוש המערכת.

צילום 3- שכבת התעבורה (TCP):



בצילום זה מוצגת שכבת התעבורה (TCP) של החבילה. ניתן לראות את פורט המקור (פורט זמני של הלקוח) ואת פורט היעד 8053, עליו מאזין השרת. החבילה מכילה דגל SYN, המעיד על תחילת תהליך הקמת חיבור TCP בין הקליינט לשרת. כמו כן מוצגים מספרי הרצף (Sequence Number) והחלון (Window Size), אשר משמשים את מנגנוני האמינות ובקרת הזרימה של TCP. ניתוח התעבורה מראה כי מערכת הצ'אט עושה שימוש בפרוטוקול TCP מעל IPv4, ומקיימת חיבורים אמינים בין הלקוחות לשרת. תהליך הקמת החיבור, העברת הנתונים וניהול החיבורים תואמים את אופן הפעולה של הקוד שנכתב, ומדגימים תקשורת מלאה עד שכבת הרשת (IP כולל) ושכבת התעבורה (TCP).

חלק 3 - שימוש בבינה מלאכותית במהלך העבודה

במהלך העבודה על הפרויקט נעשה שימוש חלקי בבינה מלאכותית, בהתאם להנחיות הקורס.

השימוש בבינה מלאכותית נעשה לצרכים הבאים :

- יצירת נתוני דוגמה המדמים תעבורת רשת (DNS) לצורך בניית קובץ CSV .
- סיוע בניסוח הסברים לדוח (הבהרת מושגים, קיצור ניסוחים).
- בדיקה והבנה של פלטים מ-Wireshark (Frame, IP, TCP) לצורך ניתוח נכון.
- סיוע חלקי בהסברים על הספריות והפונקציות שניתן להשתמש לשם כתיבת הקוד. בנוסף, נעזרנו כדי לטפל בשגיאות בקוד.

דוגמאות לפרומפטים שנעשה בהם שימוש :

1. "תעזור לי ליצור קובץ CSV עם הודעות DNS "
2. "תסביר לי בקצרה מה המשמעות של PSH TCP ו-ACK שמופיעים ב-Wireshark."
3. "תעזור לי לנסח סיכום קצר וברור לניתוח שכבת TCP "
4. "תבדוק את תקינות הקוד שלי ותן לי הצעות לשיפור"
5. "איך אני יכולה לשפר את הפונקציה try, except בקובץ"
6. "תסביר לי מה ההבדל בין פורט 53 לפורט 8053"