
פרויקט גמר

כיתוח תעבורת בפרוטוקול TCP/IP

מוגש על ידי: לירן דגן

תעודת זהות: 215609397

שם מרצה: ד"ר קוֹזֶכְבָּ אַנְדָּרִי

קבוצת הרצאה: 61305-3

תוכן עניינים

3 חלק ראשון

| | | |
|---------|---------------------------------------|----|
| 3 | מבוא | .1 |
| 3 | יצירת קובץ ה-CSV | .2 |
| 6 | תיאור והסבר אരיזת הפאקטות | .3 |
| 9 | ניתוח התעבורה באמצעות WIRESHARK | .4 |

14 חלק שני

| | | |
|----------|---|----|
| 14 | מבוא | .5 |
| 15 | ארכיטקטורת הפרויקט | .6 |
| 16 | – אפליקציית השרת Server_interface .6.1 | |
| 21 | – אפליקציית הלקוח Client_interface .6.2 | |
| 24 | הוראות התקנה והרצה | .7 |
| 31 | דוגמאות קלט ופלט | .8 |

48ביבליוגרפיה ומקורות

חלק ראשון

אריזת נתונים ולבידת מנוט ב-Wireshark

מבוא

בחלק הראשון של הפרויקט נדרשים אנו להכין קובץ CSV עם עמודות שונות לניטוח בסיסי של מידע ברמת היישום. נשתמש בקובץ בתוך מחרבת ג'יפוי כדי להציג תעבורת שמקבל המחשב בהתאם לפרטוקול שנבחר. (כדוגמת SMTP, DNS, HTTP, DNS, PPP)

ההודעות שייכנסו לקובץ מתכילות על גבי פרוטוקול בשכבה האפליקציה/יישום, ולשם הפרויקט בחרתי ליבא הודעות שהועברו בפרטוקול DNS (Domain Name System DNS).

יצירת קובץ ה-CSV

לקובץ ה-CSV שמצויר לו"ח הוסף שדות שמכילות הפקודות, בין היתר סוג החיבור (UDP), Request/Reply, גודל הפקטה וכו'.

במובן שהקובץ מכיל גם את השדות להם נדרש בתיאור הפרויקט:

- msg_id - מספר סידורי המיחד כל הודעה
- app_protocol - פרוטוקול ברמת האפליקציה אשר בחרנו, שהוא DNS
- src_port – מספר הפורט ממנו יצאתה הפקטה מצד השולח
- dst_port – מספר הפורט אליו הגיעו הפקטה מצד המקלט
- time_stamp – הרגע שבו הגיעו הפקטה
- message – הודעה שנרצה להפיק מהפקטה

הקובץ נוצר אך ורק באמצעות לבידוד תעבורת תוכנת Wireshark (אפשר למשל למצאו בקובץ dns_input_capture.pcap) והודעות משרת DNS של Google ושל Discord, שרצו על המחשב שלו, בעת ההקלטה).

כדי לקבל את התעבורת השתמשתי בфиילטר 'dns' כדי לסוג את הפקודות שרציתי ליצא. עברו כל שדה שרציתי להוסיף ב العمודה Wireshark סימנתי "apply as column". ברגע קיברתי את כל השדות שרציתי ליצא עברו הפקות. לאחר שאספתי מספיק תעבורת עצרתית את ההקלטה, לחצתי על תפריט File ועל Export packet dissections CSV. ברגע ייצאתי את הטבלה שנוצרה ב-Wireshark אל תוכנת Excel ומשם שמרתי את הטבלה בתור קובץ CSV על המחשב.

השדה message – הרכבתי אותו על ידי שרשור שדות:

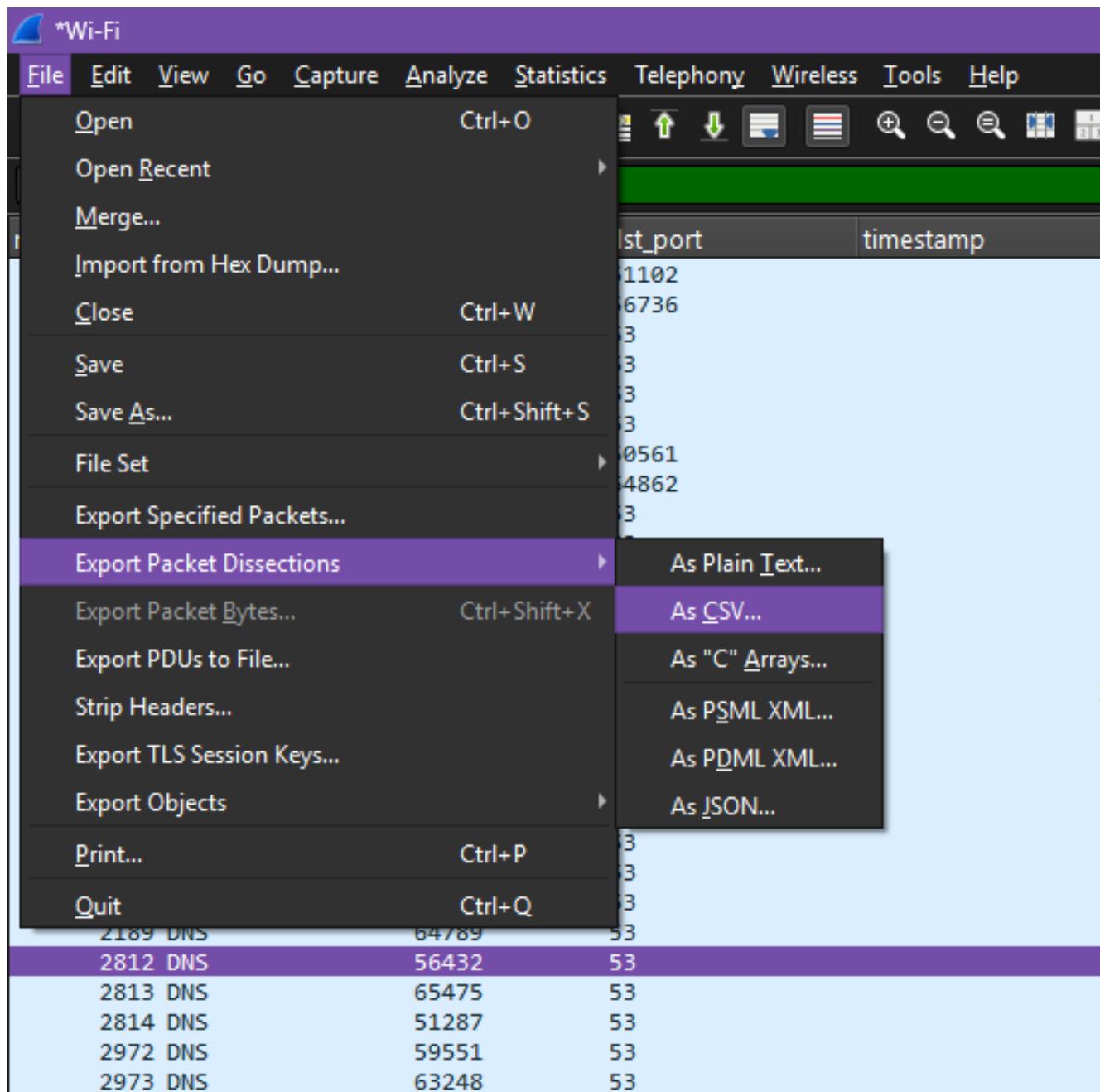
(ID | | domain name | | is Response | | Transaction)

כל אלו הם מאפיינים של פרוטוקול DNS. Domain name משמעו שם הכתובת של האפליקציה, is Response משמעו האם הפקטה היא בקשה לשרת (אפס) או תגובה של השרת (אחד). Transaction ID משמעו מספר סידורי של פרוטוקול DNS עבור השאלה.

יצירת השדה message

| Fields: dns.qry.name dns.flags.response dns.id | | |
|--|-----------|---------------------------|
| destination Address | timestamp | message |
| 1.1.1 | 0.000108 | dns.google,0,0x03d8 |
| 1.1.1 | 0.000140 | dns.google,0,0x0957 |
| 1.1.1 | 0.000336 | dns.google,0,0xb42 |
| 1.1.1 | 0.000071 | dns.google,0,0x162f |
| 1.1.1 | 16.693004 | dns.google,0,0x18d5 |
| 1.1.1 | 0.000361 | dns.google,0,0x2100 |
| 1.1.1 | 0.895141 | dns.google,0,0x3072 |
| 1.1.1 | 6.888908 | dns.google,0,0x368f |
| 1.1.1 | 29.761779 | dns.google,0,0x3abc |
| 1.1.1 | 1.014845 | dns.google,0,0x41f8 |
| 1.1.1 | 2.948365 | dns.google,0,0x4261 |
| 1.1.1 | 0.000235 | dns.google,0,0x5954 |
| 1.1.1 | 0.000210 | dns.google,0,0x60af |
| 1.1.1 | 0.000120 | dns.google,0,0x6d94 |
| 1.1.1 | 0.000145 | dns.google,0,0x6f9b |
| 1.1.1 | 0.000129 | dns.google,0,0x86f9 |
| 1.1.1 | 0.000216 | dns.google,0,0x940d |
| 1.1.1 | 0.000334 | dns.google,0,0x9473 |
| 1.1.1 | 0.000388 | dns.google,0,0x9cb5 |
| 1.1.1 | 0.000256 | dns.google,0,0xa3fa |
| 1.1.1 | 1.012652 | dns.google,0,0xa7a7 |
| 1.1.1 | 0.000329 | dns.google,0,0xe57d |
| 1.1.1 | 0.882902 | dns.google,0,0xf405 |
| 2.168.1.18 | 0.008372 | dns.google,1,0x0957 |
| 2.168.1.18 | 0.000000 | dns.google,1,0x162f |
| 2.168.1.18 | 0.006692 | dns.google,1,0x2100 |
| 2.168.1.18 | 0.001023 | dns.google,1,0x368f |
| 2.168.1.18 | 0.029216 | dns.google,1,0x3abc |
| 2.168.1.18 | 0.001168 | dns.google,1,0x41f8 |
| 2.168.1.18 | 0.026362 | dns.google,1,0x60af |
| 2.168.1.18 | 0.066347 | dns.google,1,0x6d94 |
| 2.168.1.18 | 0.006016 | dns.google,1,0x6f9b |
| 2.168.1.18 | 0.011840 | dns.google,1,0x940d |
| 2.168.1.18 | 0.001820 | dns.google,1,0xe57d |
| 2.168.1.18 | 0.020594 | dns.google,1,0xf405 |
| 1.1.1 | 14.580840 | dns.msftncsi.com,0,0x3fc4 |
| 1.1.1 | 0.100803 | dns.msftncsi.com,0,0x3fc4 |
| 1.1.1 | 1.011470 | dns.msftncsi.com,0,0x3fc4 |
| | 8.962245 | |

יצוא הפקטוט אל קובץ CSV



תיאור והסבר אריזת הפאקטות

השלב השני בחלק זה הוא טיענת קובץ ה-CSV אל מחברת הג'ופיטר שמצורפת, בה קוד פיתון לייצור תעבורות של הודעות שנלקחו מה-.CSV.

המטרה: להשתמש בכימוס (Encapsulation) במודול ISO כדי לאזרז פאקטות לפי השכבות ולהעבירן. האריזות מרכיבות את הפאקטה מהשכבה העליונה (האפליקציהקציה) עד לשכבות התחתונות. בהתאם בפועל ככל' את תהליך האריזה ולאחר מכן נפרט.

תחילת ההודעה נוצרת בשכבת האפליקציה (לחברת הג'ופיטר טוענת את ההודעה שתישלח - payload) ולאחר מכן מועברת אל שכבת התעבורה. במחברת - שלוחים את הודעות ה-CSV שלנו או הודעות דמה. בلومר המחברת שלוחת כל payload משורות ה-CSV אל שכבת התעבורה.

שכבת התעבורה עוטפת את ה-payload בתוך SEGMENT שיפוענה על ידי שכבת התעבורה בצד המקלט. בשכבת התעבורה משייכים לפאקטה בותרת TCP Header שמכילה:

Checksum source port, destination port, flags שצל שדות נוספים. בין הערכים נוסף שדה checksum אשר משמש את הצד המקלט לפענו שגיאות. לאחר מכן שכבת התעבורה מעבירה את הפאקטות לשכבת הרשות. עד בה הودעת ה-CSV שלנו נעטפה בסגמנט של שכבת התעבורה.

שכבת הרשות עוטפת את הסגמנט בתוך Datagram IP. בلومר היא מוסיפה בותרת IP Header IP, source IP, destination IP, version, TTL, flags ועוד. שכבת הרשות מעבירה את ה-Datagram לשכבת הקו.

ריעונית - שכבת הקו (הליינק) מוסיפה מסגרת Frame והוא השכבה האخונה שעוטפת את ה-Datagram כדי להפוך אותה לפאקטה שתשודר להלה. היא מצינית בתובות MAC (מקור ויעד) ומעבירה את הפאקטה לשכבה הפייזית, בلومר משדרת אותה מהממשק החוצה.

בפועל, במקרה שלנו התהילכים שהמחברת עושה הם לוקאלים ולכן כשננתח את הפאקטות לא נראה בתובות MAC אמיתיות.

תפקידה העיקרי של שכבה זו בהקשר של המחברת הוא לשדר את הפאקטות אל לולאת ה-loopback, בلومר מקומית בתוך המחשב.

ניתן לראות איך המחברת ממליצה את האזיה לsegment ב프וטוקול TCP בהתאם לפורטט:

TCP Header Format

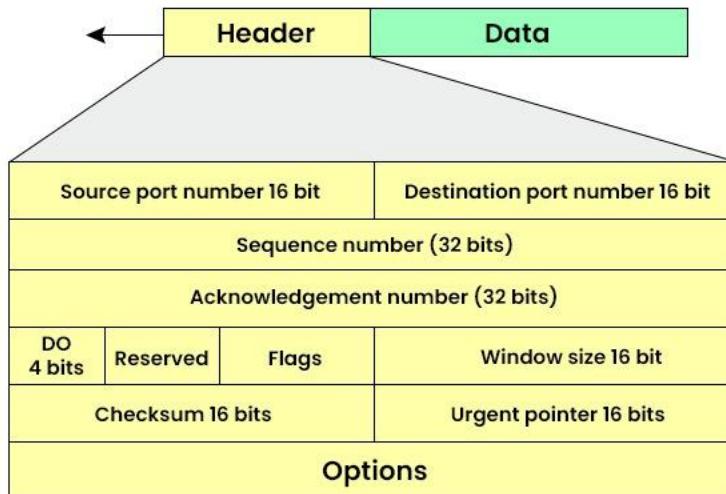
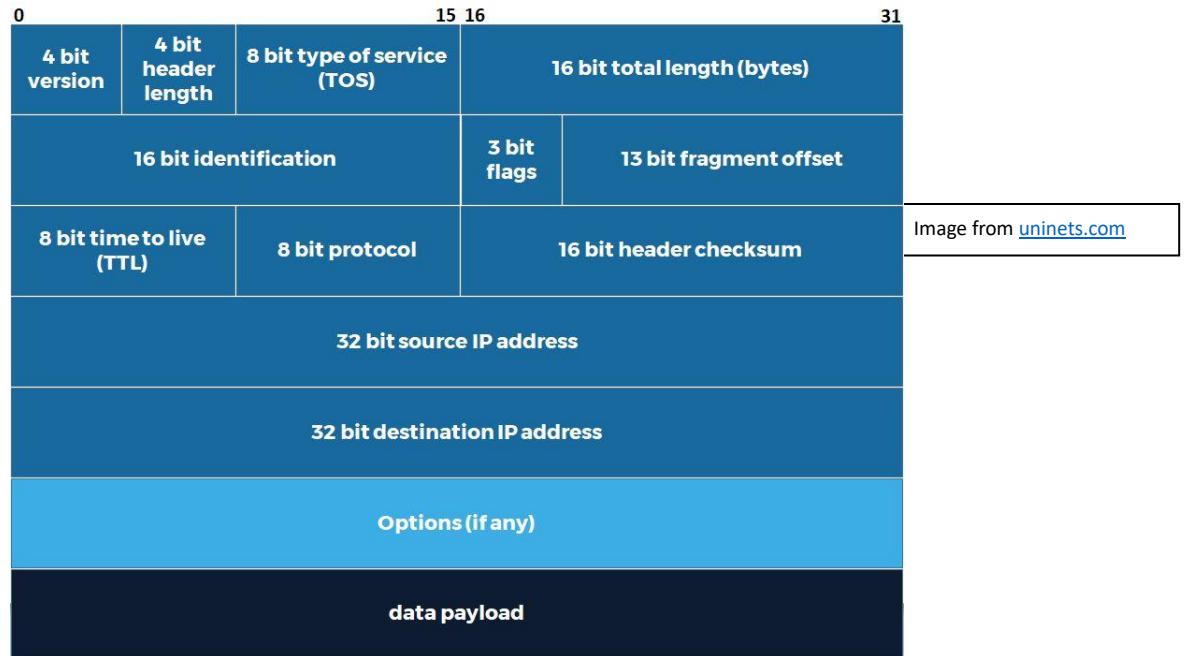


Image from pynetlabs.com

```
def build_tcp_header(src_ip: str, dst_ip: str, src_port: int, dst_port: int, payload: bytes=b'',  
                     seq: Optional[int]=None, ack_seq: int=0, flags: int=0x02, window: int=65535) -> bytes:  
    if seq is None: # if sequence number is not initialized  
        seq = random.randint(0, 0xFFFFFFFF)  
    doff_reserved = (5 << 4)  
    checksum_tcp = 0 #initialize checksum  
    urg_ptr = 0 # We don't have urgent data  
    tcp_header = struct.pack('!HLLBBHHH', # pack an initial header before evaluating checksum  
                            src_port, dst_port, seq, ack_seq,  
                            doff_reserved, flags, window,  
                            checksum_tcp, urg_ptr)  
    placeholder = 0  
    protocol = socket.IPPROTO_TCP  
    tcp_length = len(tcp_header) + len(payload)  
    pseudo_header = struct.pack('!4s4sBBH', # pseudo_header is only used to calculate checksum, using some fields included in IP header  
                               socket.inet_aton(src_ip), socket.inet_aton(dst_ip),  
                               placeholder, protocol, tcp_length)  
    checksum = checksum(pseudo_header + tcp_header + payload) #calculate checksum  
    tcp_header = struct.pack('!HLLBBB H H', # build the complete TCP header containing all fields  
                            src_port, dst_port, seq, ack_seq,  
                            doff_reserved, flags, window,  
                            checksum, urg_ptr)  
    return tcp_header
```

ניתן לראות איך המחברת ממליצה את אריית שכבת הרשת בפורמט IPv4 בהתאם לשדות:



```
def build_ip_header(src_ip: str, dst_ip: str, payload_len: int, proto: int=socket.IPPROTO_TCP) -> bytes:
    version_ihl = (4 << 4) + 5 # Version = 4, Header Length = 5 words (= 20 bytes)
    tos = 0 # Type of service, not used
    total_length = 20 + payload_len # Total Length = IP header (20 bytes) + payload length
    identification = random.randint(0, 65535) # This field is used for fragmentation
    flags_fragment = 0 # don't use fragmentation
    ttl = 64 # The Maximum amount of hops the packet can traverse
    header_checksum = 0 # initialize checksum
    src = socket.inet_aton(src_ip)
    dst = socket.inet_aton(dst_ip)
    ip_header = struct.pack('!BBHHHBBH4s4s', # pack an initial header before evaluating checksum
                           version_ihl, tos, total_length, identification,
                           flags_fragment, ttl, proto, header_checksum,
                           src, dst)
    chksum = checksum(ip_header) # calculate checksum
    ip_header = struct.pack('!BBHHHBBH4s4s', # build the complete IP header containing all fields
                           version_ihl, tos, total_length, identification,
                           flags_fragment, ttl, proto, chksum,
                           src, dst)
    return ip_header
```

בכך השלמנו את הארייה של שלושת השכבות, אפליקציה (הודעות CSV) שהמחברת טענתה ושולחת, התעבורה (הסגןט, tcp_header) והרשות (ה-IP). כל הودעה שנשלחה מהמחברת תיעטף עם עיקנון **option Encapsulation** בשכבות האלו.

ניתוח התעבורה באמצעות WIRESHARK

בשלב הראשון ננתה תעבורה שהתקבלה מהפעלת הפונקציה:

```
[30]: def demo_send(num_packets: int=3, delay_sec: float=1.0, flags: int=0x02):
    for i in range(num_packets):
        payload = f'Hello Packet {i}'.encode()
        transport.send(payload, flags=flags)
        time.sleep(delay_sec)
```

עבור כל אחת מהפקודות:

```
demo_send(num_packets=3, delay_sec=1.0, flags=0x02)
demo_send(num_packets=3, flags=0x18)
demo_send(num_packets=3, flags=0x10)
demo_send(num_packets=3, flags=0x01)
demo_send(num_packets=3, flags=0x04)
```

בתמונה שבמישר, מספר הפורט 25215 הוא פורט שהוקצה למשתמש (אני, שהריץ את Wireshark) על ידי מערכת הפעלה ומספר הפורט 12345 הוא מספר הפורט של האפליקציה (מחברת Jupyter-h).

• `demo_send(num_packets = 3, delay_sec=1.0, flags=0x02)`

זו הגדרה ל-TCP סגמנט שבה רק הדגל SYN פועל. במיילים אחרים זו פאקטה שמהווה את השלב 3-way-handshake.

אבל נשים לב שלפакטה אין פורט מאזין, במילאים אחרות אין כאן באמת חיבור TCP בין שני צדדים. אין תקשורת sockets אלא העברת סטמיה של פאקטות TCP גולמיות בין בתובת פורט כלשהו שהוקצתה על ידי מערכת הפעלה ל-loopback.

לכן נצפה שלאחר הפאקטה זהו, תישלח פאקטה שתבהיר על Reset, כי אין אפשרות להתחיל חיבור. ובאמת ניתן לראות שלאחר כל פאקטה כזו שנשלחת למחברת (ומזהוות על ידי 0 Hello Packet או 1 Hello Packet 2 Hello Packet 1 Ack=1, Reset=1 TCP

| No. | Source Port | Destination Port | Time | Protocol | Syn | Acknowledgment | Push | Reset | Fin | Flags | TCP payload |
|-----|-------------|------------------|-----------|----------|-----|----------------|------|-------|-----|--------|------------------------------|
| 349 | 25215 | 12345 | 11.084745 | TCP | 1 | 0 | 0 | 0 | 0 | 0x0002 | 48656c6c6f205061636b65742030 |
| 350 | 12345 | 25215 | 11.084795 | TCP | 0 | 1 | 0 | 1 | 0 | 0x0014 | |
| 355 | 25215 | 12345 | 12.088255 | TCP | 1 | 0 | 0 | 0 | 0 | 0x0002 | 48656c6c6f205061636b65742031 |
| 356 | 12345 | 25215 | 12.088322 | TCP | 0 | 1 | 0 | 1 | 0 | 0x0014 | |
| 374 | 25215 | 12345 | 13.092120 | TCP | 1 | 0 | 0 | 0 | 0 | 0x0002 | 48656c6c6f205061636b65742032 |
| 375 | 12345 | 25215 | 13.092247 | TCP | 0 | 1 | 0 | 1 | 0 | 0x0014 | |

| | | | |
|------|-------------------------|-------------------------|-------------------|
| 0000 | 02 00 00 00 45 00 00 36 | 00 01 00 00 40 06 7c bf | ... E 6 ... @ . |
| 0010 | 7f 00 00 01 7f 00 00 01 | 62 7f 30 39 00 00 00 00 | b 09 .. |
| 0020 | 00 00 00 00 50 02 20 00 | a1 b6 00 00 48 65 6c 6c | ... P ... Hell |
| 0030 | 6f 20 50 61 63 6b 65 74 | 20 30 | o Packet 0 |

demo_send(num_packets=3, flags=0x18) •

זו הגדירה לפאקטת TCP שבה רק הדגלים ack+push פועלים. כמו מוקודם אין באמת תקשורת TCP בין הצדדים ולבן נצפה לקבל חזרה 1. Reset=1. הפעם נשים לב שנקלט 0.ack=0. הסיבה היא שהמחברת לא קיבלה לפני פניה פקעת SYN מהמשתמש ולבן היא לא מודעת או לא חוזדת בקיים קשר.

| | | | | | | | | | | |
|---------------|-------|-------------|-----|---|---|---|---|---|--------|------------------------------|
| 20313 25215 | 12345 | 2116.209752 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 48656c6c6f205061636b65742030 |
| 20314 25215 | 12345 | 2116.209836 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 20325 25215 | 12345 | 2117.214729 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 48656c6c6f205061636b65742031 |
| 20326 25215 | 12345 | 2117.214837 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 20331 25215 | 12345 | 2118.217473 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 48656c6c6f205061636b65742032 |
| - 20332 12345 | 25215 | 2118.217552 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |

שאר הפקודות מתנהגות באופן דומה למעט האחרון:

demo_send(num_packets=3, flags=0x04) •

זו פקעתה שבה רק הדגל Reset פועל. בשפaketת זו נשלחת אפיו לא נקלט פאקטות בתגובה מהמחברת. הסיבה היא שפaketת מהסוג זהה לא מצפה בתגובה בחזרה אלא מבקשת מהמחברת לסגור את החיבור הקיים. מכיוון שלמעשה אין חיבור קיים, לא נקלט תגובה מהמחברת.

| | | | | | | | | | | |
|---------------|-------|-------------|-----|---|---|---|---|---|--------|--|
| 33219 25215 | 12345 | 3169.293865 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 33432 25215 | 12345 | 3170.296178 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| - 33440 25215 | 12345 | 3171.299444 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |

Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 0
[Next Sequence Number: 14
Acknowledgment Number: 0
Acknowledgment number (raw): 0

```

0000 02 00 00 00 45 00 00 36 00 01 00 00 40 06 7c bf .E..@|.  

0010 7f 00 00 01 7f 00 00 01 62 7f 30 39 00 00 00 00 .....b.09...  

0020 00 00 00 00 50 04 20 00 a1 b4 00 00 48 65 6c 6c ....P...Hello  

0030 6f 20 50 61 63 6b 65 74 20 30 o Packet 0

```

התמונה הבאה מדגימה כיצד נראה ה-TCP סגמנט של הפאקטות כפי שמוצגים ב-WIRESHARK.

Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1

Transmission Control Protocol, Src Port: 25215, Dst Port: 12345, Seq: 0, Len: 14

Source Port: 25215
Destination Port: 12345 ← PORTS

[Stream index: 21]
[Stream Packet Number: 1]

[Conversation completeness: Incomplete (45)]
[TCP Segment Len: 14]

Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 0
[Next Sequence Number: 15 (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
0101 = Header Length: 20 bytes (5)

Flags: 0x002 (SYN)
000. = Reserved: Not set
...0 = Accurate ECN: Not set
.... 0... = Congestion Window Reduced: Not set
.... .0.. = ECN-Echo: Not set
.... .0. = Urgent: Not set
.... .0.... = Acknowledgment: Not set
.... .0... = Push: Not set
.... .0.. = Reset: Not set
....1. = Syn: Set
....0 = Fin: Not set
[TCP Flags:S.]

Window: 8192
[Calculated window size: 8192]

ניתן לראות גם את ה-TCP Payload (ההודעה) של הפאקטות (בתמונה – ”Hello World 0”)

| | | | |
|------|-------------------------|-------------------------|------------|
| 0000 | 02 00 00 00 45 00 00 36 | 00 01 00 00 40 06 7c bf | E 6 @ |
| 0010 | 7f 00 00 01 7f 00 00 01 | 62 7f 30 39 00 00 00 00 | b 09 .. |
| 0020 | 00 00 00 00 50 02 20 00 | a1 b6 00 00 48 65 6c 6c | P . Hell |
| 0030 | 6f 20 50 61 63 6b 65 74 | 20 30 | o Packet 0 |

בשלב השני ננתח תובורה ממחברת ה-PCI'ויפיר שהתקבלה כתוצאה מהעברת הודעות ה-CSV:

Send Messages from CSV file

Iterate over the rows and send message by message

```
[43]: #Send messages from CSV file
for index, row in messages_df.iterrows():
    # Extract message details from the DataFrame row
    message = row['message']
    message = f"test message {index}" if not message else message
    # Send the message using the RawTcpTransport class
    # (You may need to adjust flags and other parameters as needed)

    #TODO: uncomment the line below to send the messages
    transport.send(message.encode(), flags=0x18) # Example with PSH+ACK flags

    time.sleep(0.1) # Optional delay between messages
```

למעשה שולחים פאקטות TCP שמכילות את הודעות ה-C-עם הדגלים .ack=1,push=1

אך ניתן לראות שככל הפאקטות שמתקבלות בעלות TCP סגנון עם אותם דגלים:

Transmission Control Protocol, Src Port: 25215, Dst Port: 12345, Seq: 1, Ack: 1, Len: 32
Source Port: 25215
Destination Port: 12345
[Stream index: 0]
[Stream Packet Number: 17]
[Conversation completeness: Incomplete (40)]
[TCP Segment Len: 32]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 0
[Next Sequence Number: 33 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 0
0101 = Header Length: 20 bytes (5)
Flags: 0x018 (PSH, ACK)
000. = Reserved: Not set
...0 = Accurate ECN: Not set
.... 0... = Congestion Window Reduced: Not set
.... .0.. = ECN-Echo: Not set
.... ..0. = Urgent: Not set
.... ...1 = Acknowledgment: Set
.... 1... = Push: Set
....0.. = Reset: Not set
....0. = Syn: Not set
....0 = Fin: Not set
[TCP Flags:AP....]
Window: 8192

קובץ הלכידה (מצורף בGITהאב: jupyter_send_csv_capture.pcap) לאחר שליחת הודעות ה-CSV

| No. | Source Port | Destination Port | Time | Protocol | Syn | Acknowledgment | Push | Reset | Fin | Flags | TCP payload |
|-----|-------------|------------------|----------|----------|-----|----------------|------|-------|-----|--------|--|
| 1 | 25215 | 12345 | 0.000000 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 697076362e6d736674636f6e6e656374746573742e636f6d2c302c307862626430 |
| 2 | 12345 | 25215 | 0.000051 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 3 | 25215 | 12345 | 0.104457 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 697076362e6d736674636f6e6e656374746573742e636f6d2c302c307865346462 |
| 4 | 12345 | 25215 | 0.104562 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 5 | 25215 | 12345 | 0.209886 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 697076362e6d736674636f6e6e656374746573742e636f6d2c302c307862626430 |
| 6 | 12345 | 25215 | 0.209951 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 7 | 25215 | 12345 | 0.315308 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 697076362e6d736674636f6e6e656374746573742e636f6d2c302c307865346462 |
| 8 | 12345 | 25215 | 0.315410 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 9 | 25215 | 12345 | 0.419121 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 697076362e6d736674636f6e6e656374746573742e636f6d2c302c307862626430 |
| 10 | 12345 | 25215 | 0.419220 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 11 | 25215 | 12345 | 0.524244 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 697076362e6d736674636f6e6e656374746573742e636f6d2c302c307865346462 |
| 12 | 12345 | 25215 | 0.524342 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 13 | 25215 | 12345 | 0.631054 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 697076362e6d736674636f6e6e656374746573742e636f6d2c312c307865346462 |
| 14 | 12345 | 25215 | 0.631209 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 15 | 25215 | 12345 | 0.734618 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 697076362e6d736674636f6e6e656374746573742e636f6d2c312c307862626430 |
| 16 | 12345 | 25215 | 0.734695 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 17 | 25215 | 12345 | 0.840747 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 7777772e6d736674636f6e6e656374746573742e636f6d2c302c307835633234 |
| 18 | 12345 | 25215 | 0.840873 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 19 | 25215 | 12345 | 0.945010 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 7777772e6d736674636f6e6e656374746573742e636f6d2c302c307866636531 |
| 20 | 12345 | 25215 | 0.945091 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 21 | 25215 | 12345 | 1.048312 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 7777772e6d736674636f6e6e656374746573742e636f6d2c302c307835633234 |
| 22 | 12345 | 25215 | 1.048476 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 23 | 25215 | 12345 | 1.154639 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 7777772e6d736674636f6e6e656374746573742e636f6d2c302c307866636531 |
| 24 | 12345 | 25215 | 1.154743 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 25 | 25215 | 12345 | 1.260281 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 7777772e6d736674636f6e6e656374746573742e636f6d2c302c307866636531 |
| 26 | 12345 | 25215 | 1.260364 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 27 | 25215 | 12345 | 1.365673 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 7777772e6d736674636f6e6e656374746573742e636f6d2c302c307835633234 |
| 28 | 12345 | 25215 | 1.365783 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 29 | 25215 | 12345 | 1.470111 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 7777772e6d736674636f6e6e656374746573742e636f6d2c312c307866636531 |
| 30 | 12345 | 25215 | 1.470203 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 31 | 25215 | 12345 | 1.575103 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 7777772e6d736674636f6e6e656374746573742e636f6d2c312c307835633234 |
| 32 | 12345 | 25215 | 1.575235 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |
| 33 | 25215 | 12345 | 1.680408 | TCP | 0 | 1 | 1 | 0 | 0 | 0x0018 | 646973636f72642e636f6d2c302c307838306634 |
| 34 | 12345 | 25215 | 1.680446 | TCP | 0 | 0 | 0 | 1 | 0 | 0x0004 | |

בתוך המחברת נראה כך:

וניתן לראות שכל פאקטה בעלת TCP Payload נשאת את שדה message של השורה המתאימה בקובץ ה-CSV. למשל, הפאקטה הראשונה נשאת את ההודעה:

| | | | |
|------|-------------------------|-------------------------|-------------------|
| 0000 | 02 00 00 00 45 00 00 49 | 00 01 00 00 40 06 7c ac | E I ..@ |
| 0010 | 7f 00 00 01 7f 00 00 01 | 62 7f 30 39 00 00 00 00 |b 09 .. |
| 0020 | 00 00 00 00 50 18 20 00 | de f8 00 00 69 70 76 36 | ...P ...ipv6 |
| 0030 | 2e 6d 73 66 74 63 6f 6e | 6e 65 63 74 74 65 73 74 | .msftcon necttest |
| 0040 | 2e 63 6f 6d 2c 30 2c 30 | 78 62 62 64 30 | .com,0,0 xbbd0 |

נחזיר לקובץ ה-CSV (מצורף בGITהאב: liran_dns_input.csv) ונשים לב שגם השורה הראשונה בקובץ נושא את שדה message:

message
ipv6.msftconnecttest.com,0,0xbbd0

הפאקטה ושורת ה-CSV שתיהן נשאות את אותה ההודעה:

ipv6.msftconnecttest.com,0,0xbbd0

ולכן שורה מס' 1 בקובץ ה-CSV מתאימה לפאקטה מס' 1 בקובץ הלכידה.

דוגמה נוספת, בקובץ הלכידה (jupyter_send_csv_capture.pcap), פאקטה מס' 63 נושא את Payload:

| | | |
|------|---|-------------------|
| 0000 | 02 00 00 00 45 00 00 3b 00 01 00 00 40 06 7c ba | ן ; ש |
| 0010 | 7f 00 00 01 7f 00 00 01 62 7f 30 39 00 00 00 00 | b..09.... |
| 0020 | 00 00 00 00 50 18 20 00 7d 2c 00 00 64 6e 73 2e |p.. },...dns. |
| 0030 | 67 6f 6f 67 6c 65 2c 31 2c 30 78 36 30 61 66 | google,1 ,0x60af |

נחזיר לקובץ CSV ונראה שורה מס' 32 נושא את שדה message:

| Protocol | Type | Message is a response | Protocol Version | Port | Source IP | Destination IP | Port | Protocol | Length | Timestamp |
|----------|------|-----------------------|------------------|------|-----------|---------------------|------|----------|--------------|-------------------------|
| UDP | | Message is a response | AAAA | 130 | | dns.google,1,0x60af | | 0.026362 | 192.168.1.18 | 1.1.1.1 59365 53 DNS 32 |

הפקטה ושורת CSV שתיהן נשואות את ההודעה:

dns.google,1,0x60af

ולכן שורה מס' 32 בקובץ CSV מתאימה לפאקטה מס' 63 בקובץ הלכידה.

באופן דומה ניתן לזרות ולשייר כל הودעה בקובץ CSV גם בקובץ הלכידה.

חלק שני

פיתוח יישום רשות ללקוח/שרות ב프וטוקול TCP

מבוא

נדרכנו לבתוב פרויקט המימוש צ'אט רב משתמשים שבו קיימים לקוחות שמתקשרים אחד עם השני ושרת שמנהל את החלפת המסרים ביניהם.

יש לציין: דרישות הפרויקט מבקשות מהמשתמש לציין לקוח אליו רוצה לפנות כדי לפתח אותו צ'אט: "באשר השירות קיבל פניה לקוח השירות צריך לפתח צ'אט בין הלקוח ללקוח אחר. הפניה צריכה להכיל את שם הלקוח המבוקש. ניתן להניח שלכל לקוח יש שם ייחודי".

בולם, משתמש יוכל לשולח הודעה רק למשתמש אחד שהוא מצין במפורש את שמו. ניתן היה לשנות את השירות בדריכים אחרים כך שיעביר למשל הודעה לכל המשתמשים, או יעביר רק ללקוחות מסוימים. ניתן להפוך את הציג את למשל לקובוצתי, "כולם עם כולם". הנΚודה היא שהציג את עלול להיראות שונה בהתאם להעדרה. (צ'אט אחד על אחד, צ'אט "כולם על כולם", וכו'). באנו הדרישה באהה לידי ביתוי - הפרויקט שיתווסף מכאן והילך, השירות והAPPLICATION שיתווסף בהמשך עוקבות בדיקות אחרי הדרישה - להעביר הודעה מלוקח אחד ורק ללקוח אחר עם שם שיניינן מראש.

לדו"ח זה מצורפים שתי APPLICATION שמרכיבות את הפרויקט, האחת היא עבר הלקוח ששולח בקשה לשרת ומתקבל מסרים מהשרת. השנייה היא עבר השירות. השירות מקבל קלטים ממספר לקוחות ויודע לנוהל אותם, יודע להעביר הודעה מלוקח פונה א' אל לקוח נמען ב' ומתקשר הוא עצמו עם הלקוחות. את היישום בחרתי לבתוב בשפת C++ ובסביבת Visual Studio 2022.

באמור בនוסף לדוח מצורפים האPLICATION בתייקות:

- Server_interface
- Client_interface

באשר server_interface היא APPLICATION השירות, client_interface היא APPLICATION הלקוח, וכל השאר הם עותקים של APPLICATION הלקוח client_interface. כדי להריץ שיחה בין לקוחות ניצר מספר מופעים של client_interface (נפתח את האPLICATION כמה פעמים בחלונות שונים).

כל קבץ הקוד בשתי APPLICATION server_interface, client_interface מלאוים בתיעודים ובהסבירם.

ארכיטקטורת הפרויקט

כדי ליצור תקשורת בין יישומים בשפת C++ אנחנו משתמשים בספרייה WinSock2 שמאפשרת לנו גישה לכל פועלות ה-Sockets להן נידרש כדי לאפשר תקשורת. כאשר מדובר בתקשורת מבוססת sockets אנחנו נדרשים לביצוע התהליכים טכניים לפני שנתחיל להעביר הודעות. התהליכים משתמשים כמעט בין הלקוח לשרת, ואלו הם:

עבור הלקוח נבצע:

- אתחול ספריית Winsock
- יצרת socket של הלקוח (פונקציית יצירת socket)
- ציון בתובת השרת וחיבור בין socket הלקוח לבין השרת (פונקציית connect)
- ניהול תקשורת עם השרת באמצעות שליחת וקבלת הודעות (recv, send)
- בסוף השימוש, קרי התنتקות של הלקוח מהשרת או יציאה מהצ'אט, נסגור את ה-socket (פונקציית closesocket)
- קרייה לפונקציית cleanup של ספריית Winsock

עבור השרת נבצע:

- אתחול ספריית Winsock
- יצרת socketamazon של השרת (socket)
- קשיית ה-socketamazon אל כתובות שבה נציב את השרת, כלומר אל כתובות ה-IP ואל מספר הפורט שנקבע לו (Bind)
- רק לאחר שייצרנו את ה-socket וקשרנו אותו לכתובת שלו, נתחליל בהאזנה ללקוחות נכנים.
- (פונקציית listen)
- בעת האזנה, קיבל לckoות חדשים ונשמר אותם בסיס נתונים שיפורט בהמשך (פונקציית accept)
- לאחר קבלת לckoות ניתן באמצעות קבלה ושליחת הודעות (recv, send).
- **חשוב לציין** שבכל לckoות פועל בצורה א-סינכרונית. כלומר הפעולות של לckoות אחד אין תלויות בזמן הפעולה של לckoות שני. התהליכים של כל לckoות קורים במקביל.
- באופן עקרוני השרת אמר לעובד תמיד (ריצה אינסופית) כי בלבדו הצ'אט לא קיים, בשונה מהלקוח שכן אמר להתנתק מהצ'אט בשלב מסוים (לא רץ אינסופית). עם זאת, אנחנו מתעסקים עם sockets ולכן למען הסדר הטוב נציין שיש לסגור את ה-socket בסוף השימוש.
- (פונקציית closesocket)
- בהמשך לסעיף הקודם, קרייה לפונקציית cleanup של ספריית Winsock

כאמור הפרויקט מורכב משתי אפליקציות, אלו הן, נפרט על אפליקציית השרת ולאחר מכן על אפליקציית הלקוח:

Server interface – אפליקציית השרת

תיקיה זו מורכבת מ-4 קבצי header שם מורכדים הפונקציות עם תיעודים כלליים ומ-4 קבצי קפ' גם הם עם תיעודים מפורטים. פירוט רוחב נמצא בתיקו שבספרוקט, כאן נתאר את ה-headers:

Client_info.h – בקובץ זה מוגדר class עם פרטיים על הלוקה, שבו אנחנו שומרים את השם של הלוקה (השם ייחודי לכל לוקה – דרישות הפרויקט אומרות זאת), ואת ה-socket של הלוקה. נגידר גם פונקציה למציאת לוקה בודד בתוך רשימת לוקות בהתבסס על השם שלו. כפי שניתן להבין, בסיס הנתונים שבו נשמר את הלוקות יהיה רשימה.

נגידר גם פונקציה למחיקת הלוקה מהרשימה (במקרה של יציאה מהצ'אט).

ונגידר פונקציה לקבלת הודעות מהשרת, בלוmr השרת יפנה ל-Socket של הלוקה באמצעות הפונקציה הזו.

```
1  /* Made by Liran Dagan 215609397 */
2  #pragma once
3  #include <iostream>
4  #include <WinSock2.h>
5  #include <string>
6  #include <list>
7
8  using namespace std;
9
10 // Each client has a name and a socket to which the server sends info to and receives info from
11 class ClientInfo {
12 public:
13     string name;
14     SOCKET socket;
15
16 public:
17     ClientInfo();
18     ClientInfo(string& name, SOCKET socket);
19     bool operator==(const ClientInfo&);
20
21     // A method to find a specific client in a list, We are given that all client names are unique
22     // This is used when a client specifies the name of the target they wish to message.
23     static ClientInfo* findByName(const string&, list<ClientInfo>&);
24
25     // Given a message, deliver it to this client via the socket with "send" function. true for a successful send, false for a failed send
26     bool getMessage(const string& message);
27
28     // Given a list containing this client, we remove the client from the list.
29     void deleteFromList(list<ClientInfo>& all_clients);
30 };
31
```

socket – בקובץ זה מוגדר class עם פעולות מקידימות שהשרת מבצע לפני שהוא מתחילה לקבל לקוחות.

כולם נשתמש בפונקציות שבו כדי לאתחל את ספריית Winsock, לאתחל socket עבור השרת ולקשרו אותו לבתוות שנציב.

```
1  /* Made by Liran Dagan 215609397 */
2  #pragma once
3  #include <WinSock2.h>
4  #include <WS2tcpip.h>
5  #include <string>
6  #pragma comment(lib, "ws2_32.lib")
7
8  using namespace std;
9
10 /*
11 These are the steps the server handles to perform socket interactions:
12 1. Initialize winsock lib
13 2. Create the server socket
14 3. Bind ip (Which is 0.0.0.0) and port (which is 12345) to the socket
15 -----
16 4. Listen on the socket
17 5. Accept clients
18 -----
19 6. 'recv' for receiving messages from clients and 'send' for forwarding them to other clients
20 -----
21 7. Close the server socket when finished
22 8. Cleanup
23
24 The goal: Use 'recv' to get inputs from the users, Use 'send' to instruct users and
25 forward their messages onwards
26 */
27 class SocketSetups {
28     public:
29         // step 1: Initialize WinSock version 2.2
30         static bool initialize();
31
32         // step 2: Create a listening TCP socket for the server to accept clients
33         static SOCKET createTCPsocket();
34
35         // step 3.1: Create the server's Address details, ip and port
36         static bool createAddress(sockaddr_in& serverAddr, const string& ip, const int portNumber);
37
38         // step 3.2: Bind our TCP listening socket to the address details we created
39         static bool assignAddress(const SOCKET& listenSocket, sockaddr_in& serverAddr);
40     };
41
```

בקובץ זה הfonקציה interact – נפעיל את הfonקציה זו באמצעות thread כדי לנהל את התקשרות עם הלוקה ותוך כדי א-סינכרוניות עם לקוחות אחרים. הfonקציה מעזרת בfonקציות אחרות וממשת לוגיקה והעברת הודעות בין השרת ללוקה.

```

1  /* Made by Liran Dagan 215609397 */
2  #pragma once
3  #include <iostream>
4  #include <string>
5  #include <list>
6  #include "client_info.h"
7  using namespace std;
8
9  enum Session {
10     REGISTER,
11     SEARCH_TARGET,
12     DELIVER_TO_TARGET
13 };
14 /* Given 2 strings, compare them case insensitively(for example "liRaN" == "LIRan") */
15 bool equalStrings(const string& str1, const string& str2);
16
17 /* Given any client's message to the server we first have to check if he entered 'quit'
18   If he did, then we notify the client on his disconnection and return true
19   If he didn't, we return false */
20 bool isQuit(string& message, ClientInfo& client);
21
22 /* Given a client's input of a target he wants to message, the server checks if the target
23   is valid and sends feedback to the client. Errors to be detected are:
24   1. target is the client himself (logical error, client is trying to message himself).
25   2. target doesn't exist (technical error, the target can not be found)
26   If there is no error we notify the client that we found the target, else we
27   send the error to the client. */
28 bool validateTarget(string& targetName, ClientInfo& client, list<ClientInfo>& all_clients);
29
30 /* Once the target is validated, we need to make sure they're still active on send.
31   If the target has disconnected mid conversation we notify the client that the connection is closed
32   and ask him to pick a different target */
33 ClientInfo* getTarget(string& targetName, ClientInfo& client, list<ClientInfo>& all_clients);
34
35 /* 'Interact' is the main function that manages the interactions between the server and the client.
36   It Handles the server's actions with a single client and forwards messages from the client
37   to a target client.
38   If needed, the server sends feedback messages regarding errors.
39   Interact utilizes all the functions from above.
40
41   In short, we do the following:
42   0. We got a message from the client
43   1. Check for 'quit' (isQuit), if the client didn't send quit we proceed to 2., else we stop running
44   and remove him (deleteClient).
45   2. Use a 'mode' variable that gets 0, 1 or 2 to distinguish between actions as followed:
46     mode = REGISTER -> We receive client's name and save his details (createClient)
47     mode = SEARCH_TARGET -> we receive the target client and check for errors (isError)
48     mode = DELIVER_TO_TARGET -> We got a message to deliver and we send it to the target client by finding his socket in the clients list
49   3. return to 0.
50 */
51 void Interact(SOCKET clientSocket, list<ClientInfo>& all_clients);
52
53

```

h.main – בקובץ זה פונקציית ה-main, בה אנחנו מקימים את השירות עם כל פעולות ה-socket המקידימות. כמו כן אנחנו משתמשים בקלאס socketSetups כדי להקים את השירות ובפונקציות accept, listen כדי לקבל לקוחות. כל לקוח שמתאפשר ינהל בנפרד באמצעות thread ובסוף הפונקציה .interact

כדי לנהל את כל הלקוחות במקביל אנחנו צריכים לשמר את כולם. כמו כן אנחנו צריכים לבני נתונים כלשהו כדי לנהל את הלקוחות, ובאמת כפי שנאמר אנחנו משתמשים ברשימה. בפרויקט נזדקק לבנייה נתונים שיאפשר להוסיף, למחוק ולהפוך ללקוחות בתוך מאגר.

אפשר לבצע זאת עם המון שיטות וסוגים של מבנים. למשל אפשר לשמר עץ AVL מואוזן שמסדר את הלקוחות לפי סדר האלף-בית, או להשתמש בדרכים מתוחכמות יותר. עם זאת, זו לא מהות הפרויקט זהה, שכן נפשט מעט את השימוש והסבירים. נשמר את הלקוחות בתוך רשימה מקוشرת (list) מסוג .ClientInfo.

נתחיל את הרשימה בריקה, משם כל לקוח שמחבר לשרת יתווסף לרשימה, כל לקוח שייתנתק מהשרת יימחק מהרשימה ואם נרצה להעיבר הودעה אל לקוח מסוים יוכל פשוט לחפש אותו ברשימה ולקבל פוינטර לאובייקט שבו מאוחסנים הפרטים שלו.

זכור שככל הפעולות קוראות במקביל בין השירות לכל הלקוחות באמצעות פיצולים של thread.

```

1  /* Made by Liran Dagan 215609397 */
2  #pragma once
3  include <iostream>
4  include <tchar.h>
5  include <thread>
6  include <list>
7  include "client_info.h"
8  include "socket_setups.h"
9  define MAX_USERS 100
10 using namespace std;
11
12 /* In main we do all socket procedures from socket_setups.h along with listening and
13 accepting clients. We check for errors in each procedure. We then create an empty
14 list of type ClientInfo in which we will store all the active clients.
15 Once a client comes in and we accept him, we send his socket to the 'Interact'
16 function from interact_with_client.h by a thread to isolate the server and him, along with a
17 reference to the list of clients we created.
18 By using threads we split the server's attention to each client in particular creating desynchronization
19 which is essential for chatting between one and another */
20 int main();
21

```

מתוך main.cpp - קבלת הלקוחות ופיזול הקשբ של השרת לבולם במקביל:

```
list<ClientInfo> clients; // Stores the names & sockets of all clients participating in chat
while (true) {
    // step 5: Accept and start communicating with clients
    SOCKET clientSocket = accept(listenSocket, NULL, NULL); // A client tries to access the server
    if (clientSocket == INVALID_SOCKET)
    {
        cout << "Client socket is invalid" << endl;
    }
    else // We split the server's attention to each reaching client, so we get a multi client system
    {
        thread manageClient(Interact, clientSocket, ref(clients)); // Each client gets his own treatment simultaneously with threads
        manageClient.detach();
    }
}
closesocket(listenSocket);
WSACleanup();
return 0;
```

אפליקציית הלקוח – Client interface

תיקיה זו מורכבת מ-3 קבצי header שם מוחכדים הפונקציות עם תיעודים כלליים ומ-3 קבצי cpp גם הם עם תיעודים מפורטים וביהם בפועל מושגים התהילכים. פירוט רחב נמצא בתיעוד שבספרוקט, באן נציג את קבצי header-ים.

socket_setups.h – בדומה לאפליקציית השרת, גם באפליקציית הלקוח נגידר קובץ עם פעולות מקידימות שאנו צריכים לקיים כדי לאפשר תקשורת מבוססת sockets. בקובץ מופיע מימוש בסיסי של אתחול ספריית Winsock, יצירה socket התחברות לשרת והתחברות לשרת.

```
1  /* Made by Liran Dagan 215609397 */
2  #pragma once
3  include <WinSock2.h>
4  include <WS2tcpip.h>
5  include <string>
6  pragma comment(lib, "ws2_32.lib")
7  using namespace std;
8
9  /*
10 These are the steps the client handles to perform socket interactions:
11 1. Initialize winsock
12 2. Create communicating socket
13 3. Connect to server
14 4. 'send' for delivering messages to server and 'recv' for receiving messages from server
15 5. Close the socket when finished
16 6. Cleanup
17
18 The goal: Use 'send' and 'recv' methods with the server in order to
19     1. forward messages to a specified client
20     2. get messages from other clients
21
22 */
23 class SocketSetups
24 {
25     public:
26         // step 1: Initialize WinSock version 2.2
27         static bool initialize();
28
29         // step 2: Create a TCP socket to contact the server with
30         static SOCKET createTCPsocket();
31
32         // step 3.1: Create the address details of the server the client connects - ip and port
33         static bool createAddress(sockaddr_in& serverAddr, const string& ip, const int portNumber);
34
35         // step 3.2: Connect to the server using the 'connect' function
36         static bool connectToServer(SOCKET& serverSocket, sockaddr_in& serverAddr);
37 }
```

.sendMessage, receiveMessage interact_with_server.h

פונקציות אלה מרכזות את הלוגיקה של הלקוח בתקשורת עם השרת. פונקציה אחת אחראית על קבלת הודעות מהשרת והשניה על שליחת הודעות לשרת. הפעולות של הפונקציות משפיעות זו על זו והן מתואימות ביניהן. הן משתמשות בפונקציות אחרות כדי לנהל את התהליכים מול השרת.

```
1  /* Made by Liran Dagan 215609397 */
2  #pragma once
3
4  #include <WinSock2.h>
5  #include <iostream>
6  #include <thread>
7  #include <string>
8
9  using namespace std;
10
11 // return str1==str2 case insensitive ('a'=='A' for example)
12 bool equalStrings(const string&, const string&);
13
14 // Send client's name to the server to be known. Return success (true) or fail (false)
15 bool registerClient(SOCKET& server);
16
17 /* Send name of another client we want to message. Server will respond with a feedback message.
18 | Return success (true) or fail (false) */
19 bool searchTarget(SOCKET& server);
20
21
22 /* A thread function. Here we handle all of client's messages to the server in correlation to 'mode'
23 'sendMessage' utilizes all the functions above.
24 In short we do the following:
25 1. Use the 'mode' variable that gets 0, 1 or 2 to distinguish between actions as followed:
26    mode = 0 -> Send client's name to server (registerClient)
27    mode = 1 -> Specify target (searchTarget)
28    mode = 2 -> Send message to server to deliver to target
29 2. If connected==true meaning client didn't quit and no connection error occurred then go to 1.
30    else quit function */
31 void sendMessage(SOCKET server);
32
33
34 /* A thread function. Here we handle all of server's messages to the client including error feedbacks
35 and messages from other clients.
36 In short we do the following:
37 0. Get message/prompt from server
38 1. if got sameUser or UserNotFound prompts then stay in mode = 1 for sendMessage
39 2. if got userFound prompt then proceed to mode = 2 for sendMessage
40 3. if got Quit prompt or a disconnection error then connected=false and 6.
41 4. if none of the above then we actually got a message from another client and not a prompt, so we display it
42 5. Go to 0.
43 6. quit both thread functions sendMessage, receiveMessage and return to main to finish program */
44 void receiveMessage(SOCKET server);
```

main – בקובץ זה פונקציית ה-main, בה אנחנו מבצעים את כל פעולות ה-socket המקדיםות. לעומת זאת אנחנו משתמשים בפונקציות של קלאס SocketSetups. חשוב לציין – אנחנו יוצרים שני thread כאשר פונקציות ה-thread הן:

.interact_with_server.h, sendMessage receiveMessage
שילוב שני threads מאפשר לנו לשלוח הודעות ולקבל הודעות בו זמנית.

```

1  /* Made by Liran Dagan 215609397 */
2  #include "socket_setups.h"
3  #include "interact_with_server.h"
4  using namespace std;
5
6  /* In main we do all socket procedures from socket_setups.h.
7   * We check for errors in each procedure.
8   * We then create 2 threads, one for each function: sendMessage, receiveMessage
9   * and we give both of them the socket with which we communicate with the server.
10  By joining the threads we communicate with the server while having the two functions enable
11  each other. receiveMessage reacts to the server's feedback to sendMessage, and
12  sendMessage adjusts its actions by reacting to receiveMessage's response to the server's
13  feedback. As a result we get one coherent application that sends and receives messages
14  at the same time. */
15  int main();
16

```

מתוך main.cpp – חיבור שני הֆונקציות כדי לתקשר עם השירות:

```

// Connection is created, we're ready to communicate with the server
cout << "Connection to server established" << endl;

// We split the application's attention to both send messages to the server and receive messages from the server, simultaneously
thread senderthread(sendMessage, clientSocket);
thread receiverthread(receiveMessage, clientSocket); }
senderthread.join();
receiverthread.join();

// When both threads finish their run (by typing 'Quit' or by server connection error)
// in other words when the connection to the server stops, the program ends.
closesocket(clientSocket);
WSACleanup();
return 0;

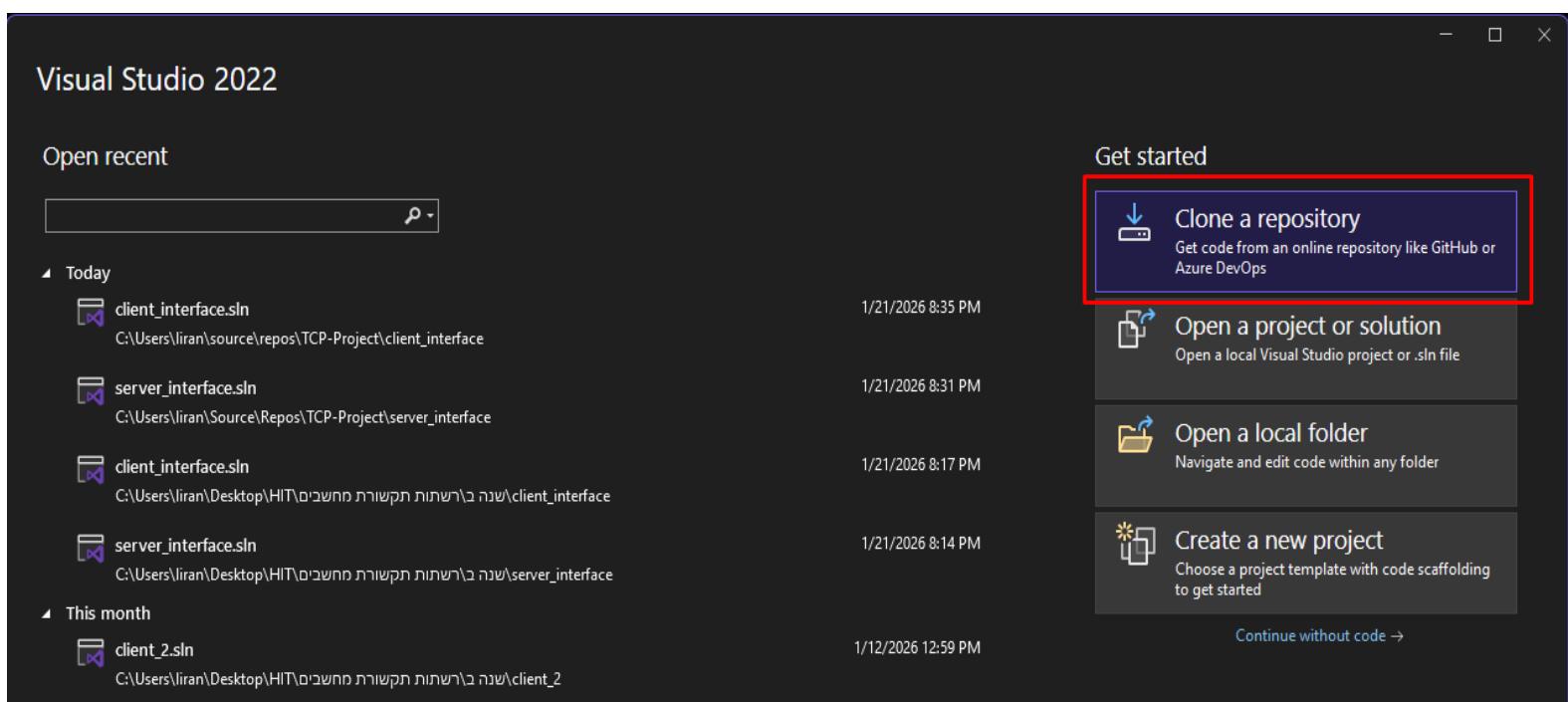
```

הוראות התקנה והרכבה

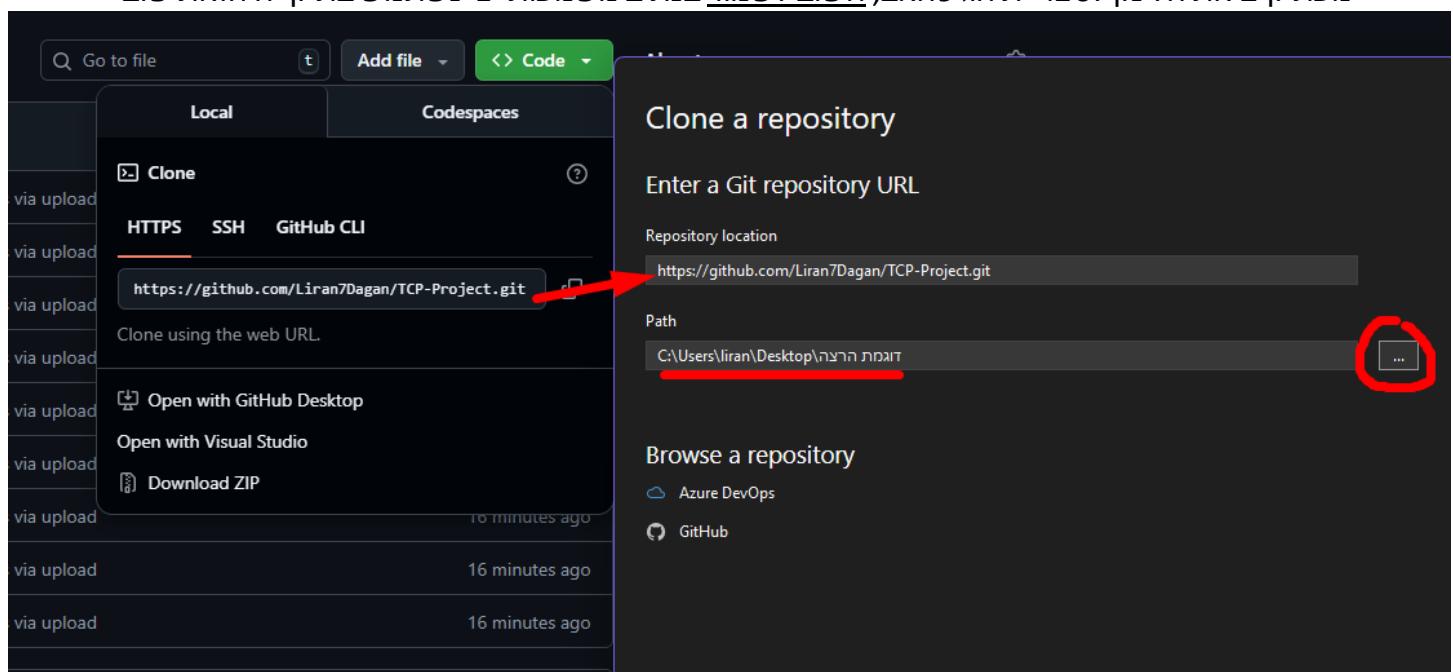
LAGITHEAB מצורפים התייחסות client_interface, server_interface שצינו קודם לכן. כאמור שתי האפליקציות נכתבו בשפת C++ ולכן להריץ אותן על המחשב יש לפתח את קבצי solution שליהן ביזואל סטודיו.

יש לפתח חלון ויזואל סטודיו אחד שירץ את השירות וחולנות נוספים שייריצו את אפליקציית הלקוח. ניתן להשתמש בשני חלונות, שלושה חלונות או יותר, בהתאם ליקוווטים שידרשו.

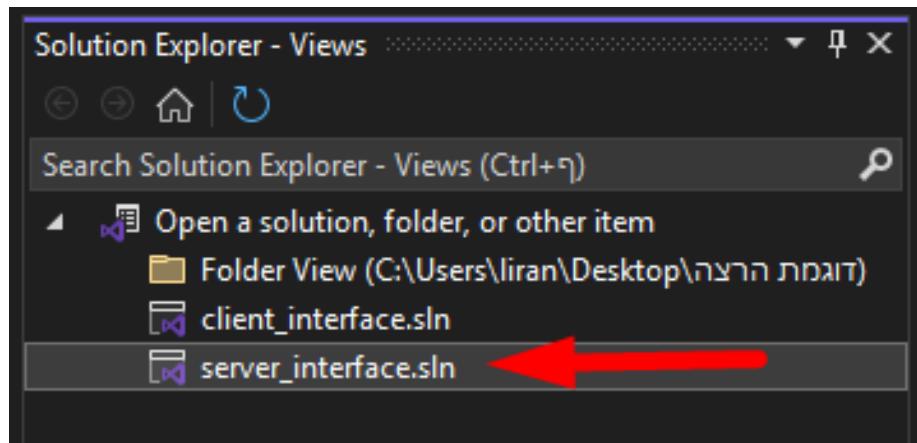
:clone a repository פותחים את ויזואל סטודיו ולחוצים



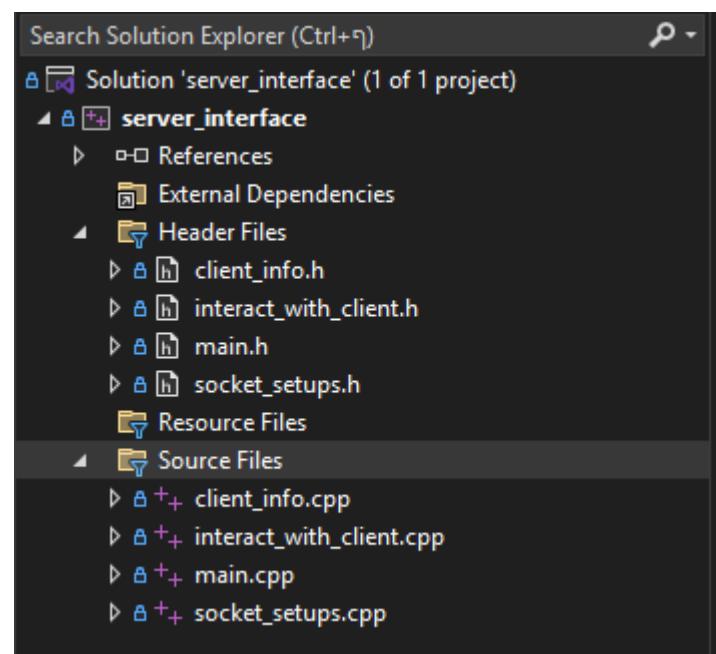
מעתיקם את הלינק לספריית הגיטהאב, חשוב לשמור בנתיב שימושי כי נשתמש בתיקייה הזאת שוב



נלחץ clone ומכאן אפשר לפתוח את אפליקציית השרת:



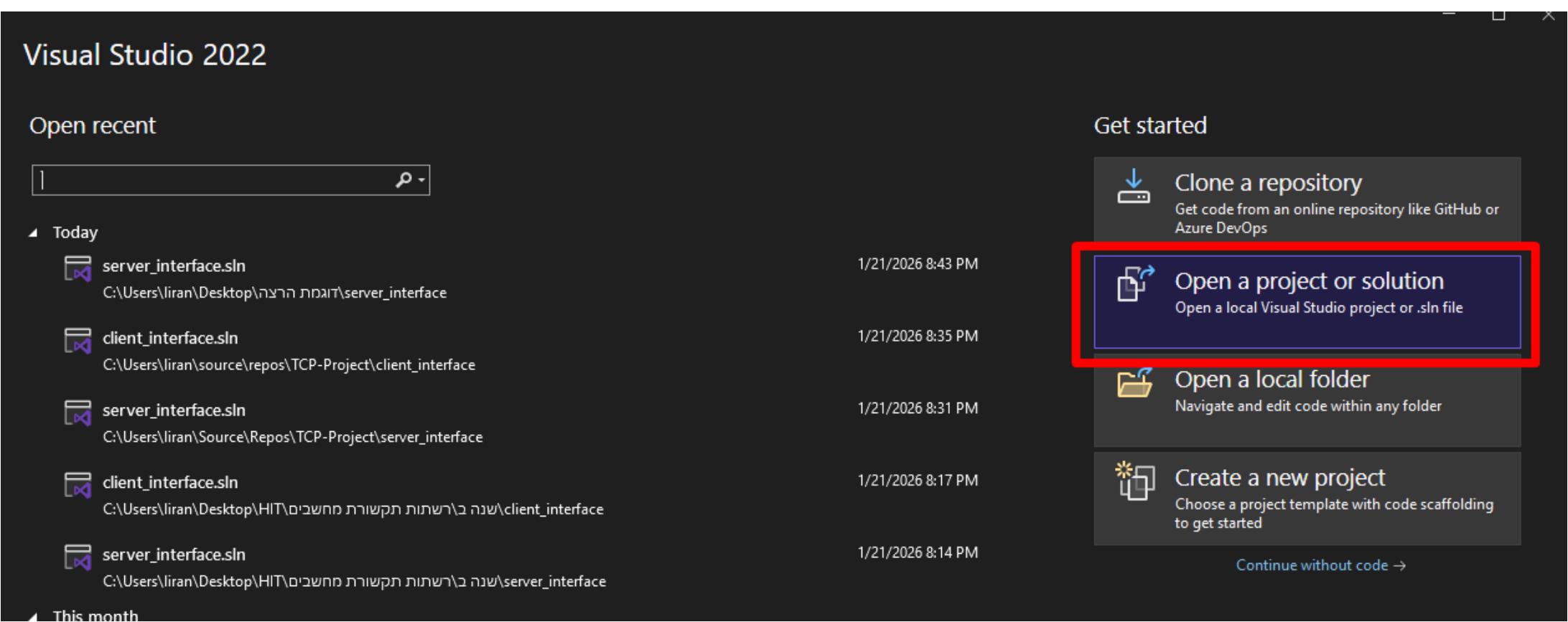
בשלב זה ניתן לפתוח ולראות את כל קבצי הקוד: ה-.h ו-ה-.cpp של השרת בתוך ויוזאל סטודיו.



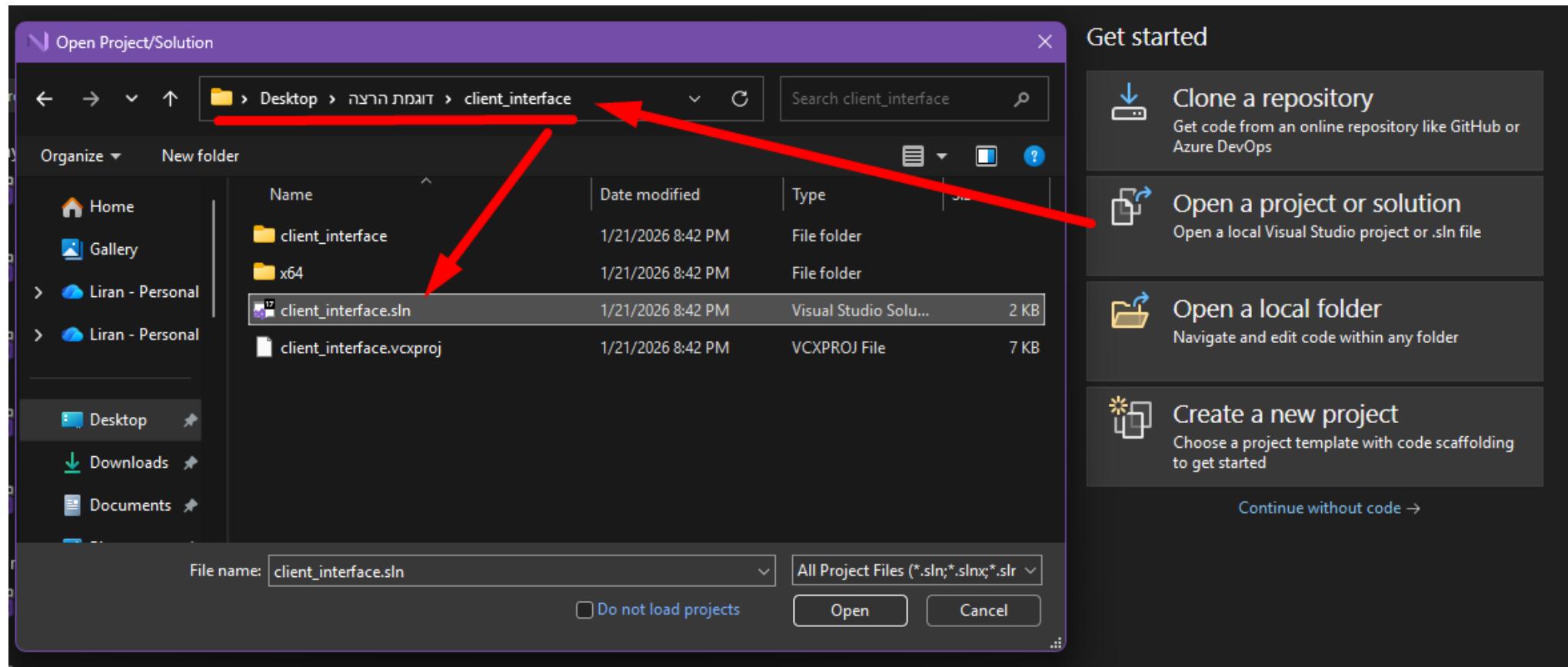
עבשו ציר לפתיחת אפליקציית הלוקוח, בשבייל כך צריך לפתוח חלון נוסף של ויוזאל סטודיו:



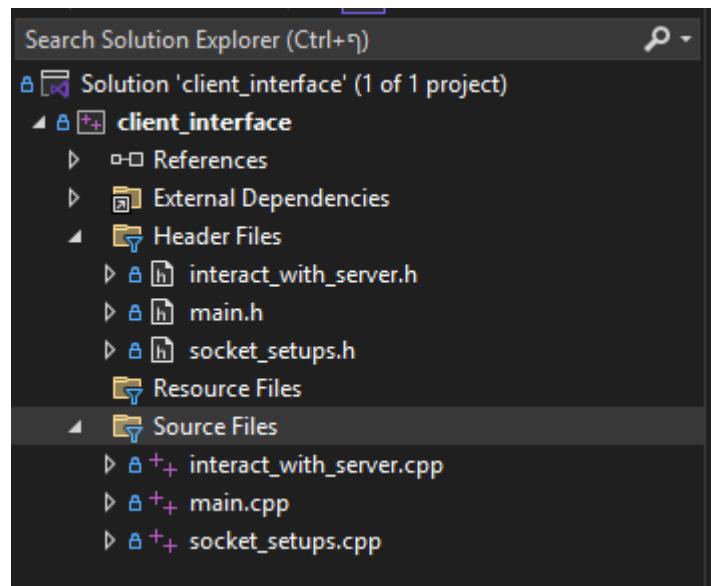
נפתח את קובץ ה-solution של הלקוט בנתיב ששמרנו.



פתיחת אפליקציית הלוקוט:

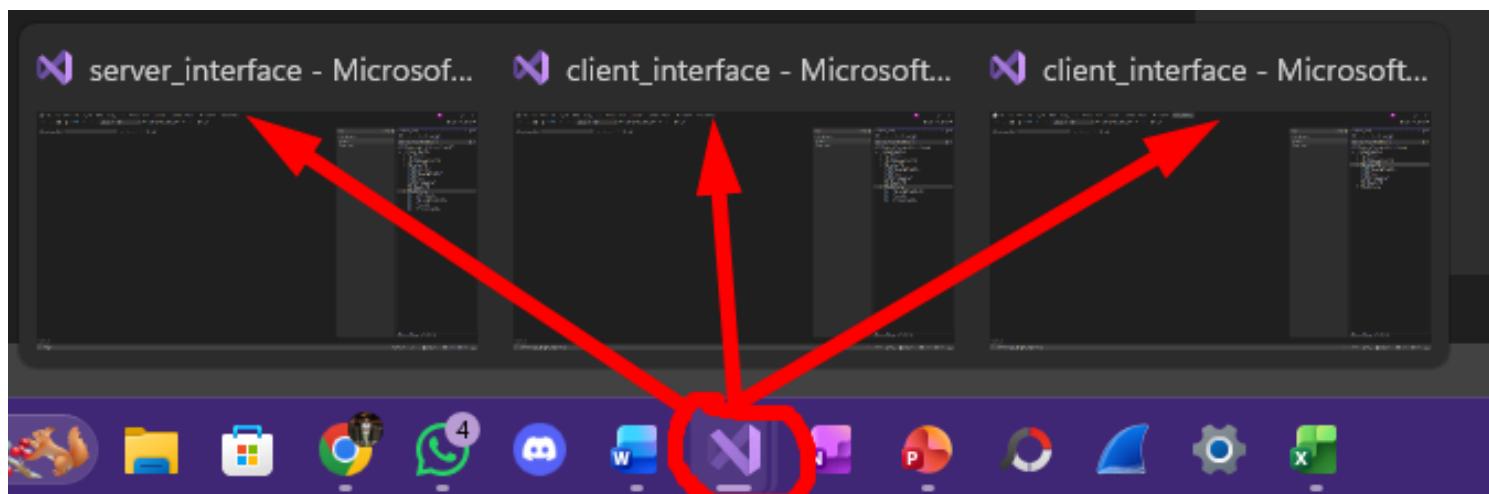


בשלב זה ניתן לפתח ולראות את כל קבצי הקוד: ה-.h ו-ה-.cpp של הלוקוט בתוך ויזואל סטודיו.



יש לפתח עוד לפחות ל��וח אחד באוטו אופן שתואר בשני העמודים האחרונים.

עד לקבלתו של שרת אחד ומספר ל��וחות, ככה:



באן מסתיים שלב ההתקנה, יש לנו שרת אחד ולפחות שני ל��וחות שמוכנים להרצה. ניתן לעיין בקוד.

שგיאה צפוייה היא שגיאה מהסוג:

```
C:\Users\liran\Desktop\HIT> Server is now listening on port 12345
```

```
Microsoft Visual Studio Debug> Connection failed to establish: 10060
```

C:\Users\liran\Desktop\HIT\??? ?\????? ??
ted with code -1 (0xffffffff).
To automatically close the console when d
le when debugging stops.
Press any key to close this window . . .

הסיבה לשגיאה: באפליקציית הלוקוח, בקובץ main.cpp, שורה 23:

```
1  /* Made by Liran Dagan 215609397 */
2  #include "main.h"
3
4  int main()
5  {
6      // step 1: Start winsock
7      if (!SocketSets::initialize())
8      {
9          cout << "Failed to start the application." << WSAGetLastError() << endl;
10         return -1;
11     }
12
13     // step 2: Create TCP type server socket
14     SOCKET clientSocket = SocketSets::createTCPsocket();
15     if (clientSocket == INVALID_SOCKET)
16     {
17         cout << "Failed to create socket " << WSAGetLastError() << endl;
18         return -1;
19     }
20
21     // step 3.1: Assign the server address details we wish to connect to
22     sockaddr_in serverAddr;
23
24     //const string ip = "127.0.0.1"; // UNCOMMENT FOR LOOPBACK
25     const string ip = "192.168.1.15"; // local IP address, comment or change if needed
26
27     const int port = 12345; // Our assigned port number
28     if (!SocketSets::createAddress(serverAddr, ip, port))
29     {
30         cout << "Setting address failed " << WSAGetLastError() << endl;
31         closesocket(clientSocket);
32         WSACleanup();
33         return -1;
34     }
35
36     // step 3.2: Connect client to the server socket
```

לפני ההרצה יש להתאים את כתובת ה-IP באפליקציית הלוקוח לו של מחשב הבודק. ישנה גם אפשרות להשתמש בכתובת ה-loopback במקומם, אם רצים. אני השתמשתי בכתובת IP אמיתית, מומלץ להשתמש בכתובת אמיתית אבל הוסףתי גם את הלופבאק בשבייל הנוחות. הסיבה שעדיין בכתובת אמיתית היא שכך תהיה באמת תקשורת בין האפליקציות דרך הרשת. בכלל זאת, בשתי הדריכים (כתובת אמיתית או לופבאק) הcz'אט עובד והכל בסדה. יש להריץ את אפליקציית השרת ראשונה, ורק לאחר מכן את אפליקציית הלוקוח. אם ההרצה תקורה בסדר הפוך (קודם לлокוח ואז שרת), השרת יעבד כרגע אבל הלוקוח יקבל את ההדפסה (שהיא חלק מהקод): Connection failed to establish: 10061 (השרת הורץ רק אחרי הלוקוח ולבסוף הוא עדין לא קיים בזמן הרצת הלוקוח). וכן לא ניתן להתחבר אליו. (השרת הורץ רק אחרי הלוקוח ולבסוף הוא עדין לא קיים בזמן הרצת הלוקוח).

דוגממה:

The screenshot shows two terminal windows side-by-side. The left window, titled 'C:\Users\liran\Desktop\' (with a red arrow pointing to it), displays the message 'Server is now listening on port 12345'. The right window, titled 'Microsoft Visual Studio' (with a red arrow pointing to it), displays an error message: 'Connection failed to establish: 10061' followed by the command-line output of a process termination: 'C:\Users\liran\Desktop\client_interface\x64\Debug\client_interface.exe (process 50404) exited with code -1 (0xffffffff). To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops. Press any key to close this window . . .' A red arrow points from the text 'To automatically close the console when debugging stops...' towards the right edge of the image.

בנוסף, האפליקציות מטפלות בשגיאות (כמו זו שבדוגמה) ולכן אמורים לקרות קריסות בלתי צפויות. אם קורות שגיאות כלשהן כמו התנטקיות למשל, האפליקציות ידפיסו על המסך הודעות מתאימות.

המסמך שמקבלים מאפליקציה השרת הוא נטו אינפורטטיבי, והוא לא מקבל קלט מהמשתמש.
המסמך שמקבלים מאפליקציה הלקוח לעומת זאת הוא מבון גם אינטראקטיבי.

דוגמאות קלט ופלט

דוגמיה לשיחה בין שני משתמשים:

The screenshot shows two Microsoft Visual Studio windows side-by-side, each displaying a terminal-like interface for a chat application. The left window is titled 'Microsoft Visual Studio Del' and the right window is titled 'Microsoft Visual Studio Debug'. Both windows show a conversation between two users, Liran and Maor.

Liran's Chat History:

- Connection to server established
- Welcome to the chat! What is your name?
- Liran
- Hello Liran! Have fun chatting.
- You can quit any time by typing 'QUIT'.
-
- Who do you want to message?
- itzik
- User Was Not Found.
-
- Who do you want to message?
- yaron
- User Was Not Found.
-
- Who do you want to message?
- Maor
- Connection created
-
- Hey Maor!!!
-
- Maor: Whats upppppppppp Lirannnn
- I'm doing great
- There's a big football game today
- Maor: I know today is Real Madrid vs Barcelona
- Right it will be a good game for sure!!!
- Do you wanna come over and watch it together? We c
an order pizza ;)
- Maor: YEAH For sure
- Great so be at my place in about 15 minutes
- Maor: I'm on my way, see you then!
- quit

Maor's Chat History:

- Connection to server established
- Welcome to the chat! What is your name?
- Maor
- Hello Maor! Have fun chatting.
- You can quit any time by typing 'QUIT'.
-
- Who do you want to message?
- Maor
- Please Message a Different User.
-
- Who do you want to message?
- Liran
- Connection created
-
- Liran: Hey Maor!!!
- Whats upppppppppp Lirannnn
- Liran: I'm doing great
- Liran: There's a big football game today
- I know today is Real Madrid vs Barcelona
- Liran: Right it will be a good game for sure!!!
- Liran: Do you wanna come over and watch it together?
We can order pizza ;)
- YEAH For sure
- Liran: Great so be at my place in about 15 minutes
- I'm on my way, see you then!
- quit

עבור השיחה שלמעלה, פלט השירות הוא:

```
C:\Users\liran\Desktop\HIT\o X + ▾  
Server is now listening on port 12345  
New client joined: 'Liran'  
New client joined: 'Maor'  
'Liran' Tried to reach a non existent client: 'itzik'  
Maor Tried to message himself  
'Liran' Tried to reach a non existent client: 'yaron'  
'Liran' Reaches 'Maor'. Server is ready to transmit messages!  
'Maor' Reaches 'Liran'. Server is ready to transmit messages!  
'Liran' TO 'Maor': Hey Maor!!!  
'Maor' TO 'Liran': Whats upppppppppp Lirannnn  
'Liran' TO 'Maor': I'm doing great  
'Liran' TO 'Maor': There's a big football game today  
'Maor' TO 'Liran': I know today is Real Madrid vs Barcelona  
'Liran' TO 'Maor': Right it will be a good game for sure!!!  
'Liran' TO 'Maor': Do you wanna come over and watch it together? We can order pizza ;)  
'Maor' TO 'Liran': YEAH For sure  
'Liran' TO 'Maor': Great so be at my place in about 15 minutes  
'Maor' TO 'Liran': I'm on my way, see you then!  
'Liran' Disconnected  
'Maor' Disconnected
```

מנקודת המבט שלי – הכותב:) למעלה פלט השרת, משמאלי הקלט/פלט של לiran ומימין הקלט/פלט של Maor בתמונה שני מחשבים – המסך העליון (השרת) והמסך הימני (הלקוח מאור) שייכים לאוטו מחשב. המסך השמאלי (לiran) שיר למחשב אחר באותו רשת.

```

Server is now listening on port 12345
New client joined: 'Liran'
New client joined: 'Maor'
'Liran' Tried to reach a non existent client: 'itzik'
'Maor' Tried to message himself
'Liran' Tried to reach a non existent client: 'Yaron'
'Liran' Reaches 'Maor'. Server is ready to transmit messages!
'Maor' Reaches 'Liran'. Server is ready to transmit messages!
'Liran' TO 'Maor': Hey Maor!!!
'Maor' TO 'Liran': Whats upppppppppp Lirannnn
'Liran' TO 'Maor': I'm doing great
'Liran' TO 'Maor': There's a big football game today
'Maor' TO 'Liran': I know today is Real Madrid vs Barcelona
'Liran' TO 'Maor': Right it will be a good game for sure!!!
'Liran' TO 'Maor': Do you wanna come over and watch it together? We can order pizza ;)
'Maor' TO 'Liran': YEAH For sure
'Liran' TO 'Maor': Great so be at my place in about 15 minutes
'Maor' TO 'Liran': I'm on my way, see you then!
'Liran' Disconnected
'Maor' Disconnected

```

Microsoft Visual Studio Debug

Welcome to the chat! What is your name?
Liran
Hello Liran! Have fun chatting.
You can quit any time by typing 'QUIT'.
Who do you want to message?
itzik
User Was Not Found.
Who do you want to message?
Yaron
User Was Not Found.
Who do you want to message?
Maor
Connection created. You can message a different user by typing 'BACK'.
Hey Maor!!!
Maor: Whats upppppppppp Lirannnn
I'm doing great
There's a big football game today
Maor: I know today is Real Madrid vs Barcelona
Right it will be a good game for sure!!!
Do you wanna come over and watch it together? We can order pizza ;)
Maor: YEAH For sure
Great so be at my place in about 15 minutes
Maor: I'm on my way, see you then!
quit

Microsoft Visual Studio Debug

Connection to server established
Welcome to the chat! What is your name?
Maor
Hello Maor! Have fun chatting.
You can quit any time by typing 'QUIT'.
Who do you want to message?
Maor
Please Message a Different User.
Who do you want to message?
Liran
Connection created. You can message a different user by typing 'BACK'.
Liran: Hey Maor!!!
Whats upppppppppp Lirannnn
Liran: I'm doing great
Liran: There's a big football game today
I know today is Real Madrid vs Barcelona
Liran: Right it will be a good game for sure!!!
Liran: Do you wanna come over and watch it together? We can order pizza ;)
YEAH For sure
Liran: Great so be at my place in about 15 minutes
I'm on my way, see you then!
quit

קצת על התקשרות בין השירות ללקוחות בשיחה זו (כנרת תעבורה בהמשך, כאן אסביר על האפליקציה):

המשתמשים שלחיהם לשרת את שמותיהם "לiran", "מאור", השירות זכר ושמור אותם. בכל פעם שהמשתמשים מנסים לפנות ללקוח שהשרת אינו מכיר, כמו "איציק" ו"ירון" הוא שולח להם פידבק "המשתמש לא נמצא". באופן דומה אם המשתמש פונה אל עצמו כמו שעשה מאור, השירות משיב: "בבקשה פנה למשתמש אחר". במקרה פונה אל מאור (שהשרת מכיר) השירות מודיע לו שנוצר חיבור וההודעות מועברות מכאן והילך אל מאור. באופן זהה השירות עשו עבור מאור כשהוא פונה אל לiran. מכאן המשתמשים משוחחים ביניהם. משתמש מתנקת מהשרת על ידי הקלט "quit", כמו שעושים מאור ולiran בהודעות האחרונות שלהם. שאלת שואלי נשאלת היא: מה היה קורה אם לiran היה שולח הודעה למאור אחרי שהוא התנקק? כמובן, מאור לא מזוהה יותר על ידי השירות ולiran בכל זאת מנסה לשלוח לו הודעות.

במקרה זהה השירות לא יזהה את מאור ולכן לiran על סיום החיבור ימחזר אותו למצב ההתחלתי, ככלומר יבקש ממנו לצין לךו חדש שהוא רוצה לתקשר אליו.

בתמונה הבאה ניתן לראות תהליך שבו לiran מנסה לשלוח הודעה למאור לאחר שכבר התנקק. השירות מספר לiran שאינו אפשר לשלוח את ההודעה כי השני התנקק. מיד לאחר מכן הוא ממחזר אותו לנקודת ההתחלת ומבקש ממנו שם של משתמש אחר.

The screenshot shows two Microsoft Visual Studio terminal windows side-by-side. Both windows have a dark theme and show a chat interface.

Left Terminal (Liran's Session):

- Line 1: Connection to server established
- Line 2: Welcome to the chat! What is your name?
- Line 3: Liran
- Line 4: Hello Liran! Have fun chatting.
- Line 5: You can quit any time by typing 'QUIT'.
- Line 6: Who do you want to message?
- Line 7: Maor **בקשה להתחבר למאור, בשלב זה הוא קיים**
- Line 8: Connection created
- Line 9: hey maor **לiran שלוח הודעה למאור, בשלב זה הוא מונתק**
- Line 10: The other party has disconnected.
- Line 11: Who do you want to message?
- Line 12: maor
- Line 13: User Was Not Found.
- Line 14: Who do you want to message?
- Line 15: Quit

Right Terminal (Maor's Session):

- Line 1: Connection to server established
- Line 2: Welcome to the chat! What is your name?
- Line 3: Maor
- Line 4: Hello Maor! Have fun chatting.
- Line 5: You can quit any time by typing 'QUIT'.
- Line 6: Who do you want to message?
- Line 7: quit **מאור מתנקק תור כד שיחה**
- Line 8: C:\Users\liran\Desktop\HIT\??? ?\????? ????.
code 0 (0x0).
- Line 9: To automatically close the console when debu
le when debugging stops.
- Line 10: Press any key to close this window . . .

בר השירות מתמודד עם ניתוקים יזומים תור כד שיחות.

יש לציין גם כי המשתמשים לא חייבים בהכרח להתהבר זה אל זה בו זמן, כלומר משתמש אחד יכול לקבל הודעה מממשמש שני ללא תלות בהתהבות שלו אליו. (איסינכרוניות). היתרון שהדבר נותן הוא יכול גמישה למשתמש לשלוח הודעות רק למי שהוא רוצה בכלל רגע. לדוגמה:

```
C:\Users\liran\Desktop\ X + v - □ X C:\Users\liran\Desktop\HIT\ D X + v
Connection to server established
Welcome to the chat! What is your name?
Liran
Hello Liran! Have fun chatting.
You can quit any time by typing 'QUIT'.
-----
Who do you want to message?
Maor
Connection created
-----
hello
Maor: HI! ההודעה של מאור
-----
```

```
Connection to server established
Welcome to the chat! What is your name?
Maor
Hello Maor! Have fun chatting.
You can quit any time by typing 'QUIT'.
-----
Who do you want to message?
Liran: hello ההודעה של לiran
Who do you want to message?
xxx
User Was Not Found.
-----
Who do you want to message?
la la la
User Was Not Found.
-----
Who do you want to message?
Liran רק בשלב הזה מאור מתהבר
Connection created
-----
HI!
```

גננת את השicha הראשונה בין לירן למארו ב-Wireshark. גננת את הצד של לירן (שרץ במחשב אחר ברשות מהשרת ומארו):

קובץ הלכידה גם מצורף בשם 2_clients_chat.pcap

| No. | Time | Source Address | Destination Address | Source Port | Destination Port | Protocol | TCP payload | Syn | Acknowledgment | Fin | Push | Sequence Number | Acknowledgment Number | ip version | Time to Live |
|------|------------|----------------|---------------------|-------------|------------------|----------|-----------------|-----|----------------|-----|------|-----------------|-----------------------|------------|--------------|
| 245 | 21.223307 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 1 | 0 | 0 | 0 | 0 | 0 | 4 | 128 |
| 249 | 21.341685 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 1 | 1 | 0 | 0 | 0 | 1 | 4 | 128 |
| 250 | 21.341881 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 1 | 1 | 4 | 128 |
| 561 | 43.966002 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 4c6972616e | 0 | 1 | 0 | 1 | 1 | 1 | 4 | 128 |
| 562 | 44.349885 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 4c6972616e | 0 | 1 | 0 | 1 | 1 | 1 | 4 | 128 |
| 563 | 44.387799 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 0 | 0 | 1 | 6 | 4 | 128 |
| 657 | 63.896643 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 69747a696b | 0 | 1 | 0 | 1 | 6 | 1 | 4 | 128 |
| 658 | 64.278017 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 69747a696b | 0 | 1 | 0 | 1 | 6 | 1 | 4 | 128 |
| 659 | 64.286456 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 533a55736572... | 0 | 1 | 0 | 1 | 1 | 11 | 4 | 128 |
| 660 | 64.339513 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 11 | 15 | 4 | 128 |
| 661 | 64.461313 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 0 | 0 | 15 | 11 | 4 | 128 |
| 738 | 78.110474 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 7961726f6e | 0 | 1 | 0 | 1 | 11 | 15 | 4 | 128 |
| 744 | 78.347603 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 533a55736572... | 0 | 1 | 0 | 1 | 15 | 16 | 4 | 128 |
| 745 | 78.390215 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 16 | 29 | 4 | 128 |
| 746 | 78.742892 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 533a55736572... | 0 | 1 | 0 | 1 | 15 | 16 | 4 | 128 |
| 747 | 78.742974 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 16 | 29 | 4 | 128 |
| 844 | 83.144811 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 4d616f72 | 0 | 1 | 0 | 1 | 16 | 29 | 4 | 128 |
| 845 | 83.188005 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 533a55736572... | 0 | 1 | 0 | 1 | 29 | 20 | 4 | 128 |
| 846 | 83.239717 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 20 | 40 | 4 | 128 |
| 918 | 108.988693 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 486579204d61... | 0 | 1 | 0 | 1 | 20 | 40 | 4 | 128 |
| 919 | 109.157486 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 0 | 0 | 40 | 31 | 4 | 128 |
| 1018 | 126.671707 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 433a4d616f72... | 0 | 1 | 0 | 1 | 40 | 31 | 4 | 128 |
| 1019 | 126.717885 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 31 | 74 | 4 | 128 |
| 1161 | 136.825199 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 49276d20646f... | 0 | 1 | 0 | 1 | 31 | 74 | 4 | 128 |
| 1163 | 137.108872 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 0 | 0 | 74 | 46 | 4 | 128 |
| 1266 | 157.734175 | 192.168.1.15 | 192.168.1.56 | 12345 | 57760 | TCP | 546865726527... | 0 | 1 | 0 | 1 | 46 | 74 | 4 | 128 |
| 1267 | 158.000740 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 0 | 0 | 74 | 79 | 4 | 128 |
| 1386 | 181.944050 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 433a4d616f72... | 0 | 1 | 0 | 1 | 74 | 79 | 4 | 128 |
| 1387 | 181.985822 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 79 | 122 | 4 | 128 |
| 1510 | 211.550325 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 526967687420... | 0 | 1 | 0 | 1 | 79 | 122 | 4 | 128 |
| 1511 | 211.757499 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 0 | 0 | 122 | 119 | 4 | 128 |
| 1674 | 235.291860 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 446f20796f75... | 0 | 1 | 0 | 1 | 119 | 122 | 4 | 128 |
| 1675 | 235.392300 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 0 | 0 | 122 | 186 | 4 | 128 |
| 1706 | 245.037435 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 433a4d616f72... | 0 | 1 | 0 | 1 | 122 | 186 | 4 | 128 |
| 1707 | 245.079946 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 186 | 143 | 4 | 128 |
| 1708 | 245.651650 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 433a4d616f72... | 0 | 1 | 0 | 1 | 122 | 186 | 4 | 128 |
| 1709 | 245.651729 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 186 | 143 | 4 | 128 |
| 1710 | 245.652483 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 433a4d616f72... | 0 | 1 | 0 | 1 | 122 | 186 | 4 | 128 |
| 1711 | 245.652526 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 186 | 143 | 4 | 128 |
| 1825 | 269.606406 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 477265617420... | 0 | 1 | 0 | 1 | 186 | 143 | 4 | 128 |
| 1826 | 269.819964 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 0 | 0 | 143 | 229 | 4 | 128 |
| 1950 | 287.024120 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 433a4d616f72... | 0 | 1 | 0 | 1 | 143 | 229 | 4 | 128 |
| 1951 | 287.069915 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 229 | 179 | 4 | 128 |
| 1977 | 291.217116 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 71756974 | 0 | 1 | 0 | 1 | 229 | 179 | 4 | 128 |
| 1979 | 291.260262 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 533a71756974 | 0 | 1 | 0 | 1 | 179 | 233 | 4 | 128 |
| 1981 | 291.285594 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 1 | 0 | 185 | 233 | 4 | 128 |
| 1982 | 291.285649 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 0 | 0 | 233 | 186 | 4 | 128 |
| 1984 | 291.727135 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 | 1 | 0 | 233 | 186 | 4 | 128 |
| 1986 | 291.793995 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 0 | 1 | 0 | 0 | 186 | 234 | 4 | 128 |

הדבר הראשון שנראה לעין הוא מבון סוג התעבורה – TCP בכל הפקטות. ובאמת זה המצב מכיוון שהגדנו את השרת והל孔ות ברמת האפליקציה לעבוד עם חיבור TCP.

מצהים גם שקיימות שתי כתובות IP:

192.168.1.15 – כתובת המחשב שבו רץ המשתמש לירן

192.168.1.56 – כתובת המחשב שבו רץ השרת

מצהים גם שקיימים שני מספרי פורט:

57760 – מספר הפורט ממנו מדבר לירן (הוקצב על ידי מערכת הפעלה של המחשב)

12345 – מספר הפורט ממנו מדבר השרת (הגדרנו זאת בקוד)

נתיחס לכך והילך בניתוח למספר הפורט 57760 ו-12345 במאזינים, כשהראשון מזהה עבור לירן והשני מזהה עבור השרת. מבון שאפשר להתייחס גם כתובות ה-IP במאזינים, 192.168.1.15 עبور לירן ו-192.168.1.56 עبور השרת.

נשים לב לחיבור של המשתמש לירן לשרת. בחבילות הראשונות נוכל לבדוק לראות את ה-3 way handshake של פרוטוקול TCP בין המשתמש לשרת:

| No. | Time | Source Address | Destination Address | Source Port | Destination Port | Protocol | TCP payload | Syn | Acknowledgment |
|-----|-----------|----------------|---------------------|-------------|------------------|----------|-------------|-----|----------------|
| 245 | 21.223307 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 1 | 0 |
| 249 | 21.341685 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | | 1 | 1 |
| 250 | 21.341881 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | | 0 | 1 |

בחבילה מספר 245 קורה 245 קורה syn=1, ack=0, destination port = 12345, כלומר זו חvíלה שהשרת קיבל מהמשתמש "ליין" והוא מהוות בקשה של הלוקו להתחבר לשרת.

בחבילה מספר 249 קורה 249 קורה syn=1, ack=1, source port = 12345, כלומר זו חvíלה שהשרת שלח למשתמש "ליין" והוא מהוות אישור לבקשת החיבור של הלוקו

בחבילה מספר 250 קורה 250 קורה syn=0, ack=1, destination port = 12345, כלומר זו חvíלה שהשרת קיבל מהמשתמש "ליין" שמהוות את אישור ההתחברות של הלוקו, תהליך handshake של 3 הסטיים.

ניתן לראות את כל ההודעות שלירן שלוח לשרת ואת כל ההודעות שהשרת שלוח לירן, למשל:

ליין שלוח את שמו לשרת כדי להירשם לצ'אט, ההודעה הראשונה:

| | | | | | | | |
|-----|-----------|--------------|--------------|-------|-------|-----|------------|
| 561 | 43.966002 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 4c6972616e |
| 562 | 44.349885 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 4c6972616e |
| 563 | 44.387700 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | |

```

Frame 561: Packet, 59 bytes on wire (476 bits), 59 bytes captured (476 bits) on interface 
Ethernet II, Src: Intel_f4:fe:1 (00:0c:29:f4:fe:01), Dst: Liran (00:0c:29:00:00:0e)
Internet Protocol Version 4, Src: 192.168.1.15 (192.168.1.15), Dst: 192.168.1.56 (192.168.1.56)
    Identification: 0x2cff (11!>)
    Flags: 0x2, Don't Fragment
    Time to Live: 128
    Protocol: TCP (6)
    Header Checksum: 0x4a17 [verified]
    Source Address: 192.168.1.15 (192.168.1.15)
    Destination Address: 192.168.1.56 (192.168.1.56)
    Identification: 0x2cff (11!>)
    Flags: 0x2, Don't Fragment
    Time to Live: 128
    Protocol: TCP (6)
    Header Checksum: 0x4a17 [verified]
    Source Address: 192.168.1.15 (192.168.1.15)
    Destination Address: 192.168.1.56 (192.168.1.56)

0000  ea 94 f6 21 f7 c8 08 5b d6 f4 fe 1b 08 00 45 00
0010  00 2d e8 e2 40 00 80 06 00 00 c0 a8 01 0f c0 a8
0020  01 38 e1 a0 30 39 00 b7 64 f2 8a 31 d7 af 50 18
0030  00 ff 83 b7 00 00 4c 69 72 61 6e
0040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00a0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00b0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00c0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00e0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00f0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100  .... = Version: 4
0110  .... = Header Length: 20 (0x14)
0120  .... = Total Length: 59 (0x3b)
0130  .... = Identification: 0x2cff (11!>)
0140  .... = Flags: 0x2, Don't Fragment
0150  .... = Time to Live: 128
0160  .... = Protocol: TCP (6)
0170  .... = Header Checksum: 0x4a17 [verified]
0180  .... = Source Address: 192.168.1.15 (192.168.1.15)
0190  .... = Destination Address: 192.168.1.56 (192.168.1.56)
0200  .... = Options: 0x0000 0000 0000 0000
0210  .... = Options: 0x0000 0000 0000 0000
0220  .... = Options: 0x0000 0000 0000 0000
0230  .... = Options: 0x0000 0000 0000 0000
0240  .... = Options: 0x0000 0000 0000 0000
0250  .... = Options: 0x0000 0000 0000 0000
0260  .... = Options: 0x0000 0000 0000 0000
0270  .... = Options: 0x0000 0000 0000 0000
0280  .... = Options: 0x0000 0000 0000 0000
0290  .... = Options: 0x0000 0000 0000 0000
0300  .... = Options: 0x0000 0000 0000 0000
0310  .... = Options: 0x0000 0000 0000 0000
0320  .... = Options: 0x0000 0000 0000 0000
0330  .... = Options: 0x0000 0000 0000 0000
0340  .... = Options: 0x0000 0000 0000 0000
0350  .... = Options: 0x0000 0000 0000 0000
0360  .... = Options: 0x0000 0000 0000 0000
0370  .... = Options: 0x0000 0000 0000 0000
0380  .... = Options: 0x0000 0000 0000 0000
0390  .... = Options: 0x0000 0000 0000 0000
0400  .... = Options: 0x0000 0000 0000 0000
0410  .... = Options: 0x0000 0000 0000 0000
0420  .... = Options: 0x0000 0000 0000 0000
0430  .... = Options: 0x0000 0000 0000 0000
0440  .... = Options: 0x0000 0000 0000 0000
0450  .... = Options: 0x0000 0000 0000 0000
0460  .... = Options: 0x0000 0000 0000 0000
0470  .... = Options: 0x0000 0000 0000 0000
0480  .... = Options: 0x0000 0000 0000 0000
0490  .... = Options: 0x0000 0000 0000 0000
0500  .... = Options: 0x0000 0000 0000 0000
0510  .... = Options: 0x0000 0000 0000 0000
0520  .... = Options: 0x0000 0000 0000 0000
0530  .... = Options: 0x0000 0000 0000 0000
0540  .... = Options: 0x0000 0000 0000 0000
0550  .... = Options: 0x0000 0000 0000 0000
0560  .... = Options: 0x0000 0000 0000 0000
0570  .... = Options: 0x0000 0000 0000 0000
0580  .... = Options: 0x0000 0000 0000 0000
0590  .... = Options: 0x0000 0000 0000 0000
0600  .... = Options: 0x0000 0000 0000 0000
0610  .... = Options: 0x0000 0000 0000 0000
0620  .... = Options: 0x0000 0000 0000 0000
0630  .... = Options: 0x0000 0000 0000 0000
0640  .... = Options: 0x0000 0000 0000 0000
0650  .... = Options: 0x0000 0000 0000 0000
0660  .... = Options: 0x0000 0000 0000 0000
0670  .... = Options: 0x0000 0000 0000 0000
0680  .... = Options: 0x0000 0000 0000 0000
0690  .... = Options: 0x0000 0000 0000 0000
0700  .... = Options: 0x0000 0000 0000 0000
0710  .... = Options: 0x0000 0000 0000 0000
0720  .... = Options: 0x0000 0000 0000 0000
0730  .... = Options: 0x0000 0000 0000 0000
0740  .... = Options: 0x0000 0000 0000 0000
0750  .... = Options: 0x0000 0000 0000 0000
0760  .... = Options: 0x0000 0000 0000 0000
0770  .... = Options: 0x0000 0000 0000 0000
0780  .... = Options: 0x0000 0000 0000 0000
0790  .... = Options: 0x0000 0000 0000 0000
0800  .... = Options: 0x0000 0000 0000 0000
0810  .... = Options: 0x0000 0000 0000 0000
0820  .... = Options: 0x0000 0000 0000 0000
0830  .... = Options: 0x0000 0000 0000 0000
0840  .... = Options: 0x0000 0000 0000 0000
0850  .... = Options: 0x0000 0000 0000 0000
0860  .... = Options: 0x0000 0000 0000 0000
0870  .... = Options: 0x0000 0000 0000 0000
0880  .... = Options: 0x0000 0000 0000 0000
0890  .... = Options: 0x0000 0000 0000 0000
0900  .... = Options: 0x0000 0000 0000 0000
0910  .... = Options: 0x0000 0000 0000 0000
0920  .... = Options: 0x0000 0000 0000 0000
0930  .... = Options: 0x0000 0000 0000 0000
0940  .... = Options: 0x0000 0000 0000 0000
0950  .... = Options: 0x0000 0000 0000 0000
0960  .... = Options: 0x0000 0000 0000 0000
0970  .... = Options: 0x0000 0000 0000 0000
0980  .... = Options: 0x0000 0000 0000 0000
0990  .... = Options: 0x0000 0000 0000 0000
1000  .... = Options: 0x0000 0000 0000 0000
1010  .... = Options: 0x0000 0000 0000 0000
1020  .... = Options: 0x0000 0000 0000 0000
1030  .... = Options: 0x0000 0000 0000 0000
1040  .... = Options: 0x0000 0000 0000 0000
1050  .... = Options: 0x0000 0000 0000 0000
1060  .... = Options: 0x0000 0000 0000 0000
1070  .... = Options: 0x0000 0000 0000 0000
1080  .... = Options: 0x0000 0000 0000 0000
1090  .... = Options: 0x0000 0000 0000 0000
1100  .... = Options: 0x0000 0000 0000 0000
1110  .... = Options: 0x0000 0000 0000 0000
1120  .... = Options: 0x0000 0000 0000 0000
1130  .... = Options: 0x0000 0000 0000 0000
1140  .... = Options: 0x0000 0000 0000 0000
1150  .... = Options: 0x0000 0000 0000 0000
1160  .... = Options: 0x0000 0000 0000 0000
1170  .... = Options: 0x0000 0000 0000 0000
1180  .... = Options: 0x0000 0000 0000 0000
1190  .... = Options: 0x0000 0000 0000 0000
1200  .... = Options: 0x0000 0000 0000 0000
1210  .... = Options: 0x0000 0000 0000 0000
1220  .... = Options: 0x0000 0000 0000 0000
1230  .... = Options: 0x0000 0000 0000 0000
1240  .... = Options: 0x0000 0000 0000 0000
1250  .... = Options: 0x0000 0000 0000 0000
1260  .... = Options: 0x0000 0000 0000 0000
1270  .... = Options: 0x0000 0000 0000 0000
1280  .... = Options: 0x0000 0000 0000 0000
1290  .... = Options: 0x0000 0000 0000 0000
1300  .... = Options: 0x0000 0000 0000 0000
1310  .... = Options: 0x0000 0000 0000 0000
1320  .... = Options: 0x0000 0000 0000 0000
1330  .... = Options: 0x0000 0000 0000 0000
1340  .... = Options: 0x0000 0000 0000 0000
1350  .... = Options: 0x0000 0000 0000 0000
1360  .... = Options: 0x0000 0000 0000 0000
1370  .... = Options: 0x0000 0000 0000 0000
1380  .... = Options: 0x0000 0000 0000 0000
1390  .... = Options: 0x0000 0000 0000 0000
1400  .... = Options: 0x0000 0000 0000 0000
1410  .... = Options: 0x0000 0000 0000 0000
1420  .... = Options: 0x0000 0000 0000 0000
1430  .... = Options: 0x0000 0000 0000 0000
1440  .... = Options: 0x0000 0000 0000 0000
1450  .... = Options: 0x0000 0000 0000 0000
1460  .... = Options: 0x0000 0000 0000 0000
1470  .... = Options: 0x0000 0000 0000 0000
1480  .... = Options: 0x0000 0000 0000 0000
1490  .... = Options: 0x0000 0000 0000 0000
1500  .... = Options: 0x0000 0000 0000 0000
1510  .... = Options: 0x0000 0000 0000 0000
1520  .... = Options: 0x0000 0000 0000 0000
1530  .... = Options: 0x0000 0000 0000 0000
1540  .... = Options: 0x0000 0000 0000 0000
1550  .... = Options: 0x0000 0000 0000 0000
1560  .... = Options: 0x0000 0000 0000 0000
1570  .... = Options: 0x0000 0000 0000 0000
1580  .... = Options: 0x0000 0000 0000 0000
1590  .... = Options: 0x0000 0000 0000 0000
1600  .... = Options: 0x0000 0000 0000 0000
1610  .... = Options: 0x0000 0000 0000 0000
1620  .... = Options: 0x0000 0000 0000 0000
1630  .... = Options: 0x0000 0000 0000 0000
1640  .... = Options: 0x0000 0000 0000 0000
1650  .... = Options: 0x0000 0000 0000 0000
1660  .... = Options: 0x0000 0000 0000 0000
1670  .... = Options: 0x0000 0000 0000 0000
1680  .... = Options: 0x0000 0000 0000 0000
1690  .... = Options: 0x0000 0000 0000 0000
1700  .... = Options: 0x0000 0000 0000 0000
1710  .... = Options: 0x0000 0000 0000 0000
1720  .... = Options: 0x0000 0000 0000 0000
1730  .... = Options: 0x0000 0000 0000 0000
1740  .... = Options: 0x0000 0000 0000 0000
1750  .... = Options: 0x0000 0000 0000 0000
1760  .... = Options: 0x0000 0000 0000 0000
1770  .... = Options: 0x0000 0000 0000 0000
1780  .... = Options: 0x0000 0000 0000 0000
1790  .... = Options: 0x0000 0000 0000 0000
1800  .... = Options: 0x0000 0000 0000 0000
1810  .... = Options: 0x0000 0000 0000 0000
1820  .... = Options: 0x0000 0000 0000 0000
1830  .... = Options: 0x0000 0000 0000 0000
1840  .... = Options: 0x0000 0000 0000 0000
1850  .... = Options: 0x0000 0000 0000 0000
1860  .... = Options: 0x0000 0000 0000 0000
1870  .... = Options: 0x0000 0000 0000 0000
1880  .... = Options: 0x0000 0000 0000 0000
1890  .... = Options: 0x0000 0000 0000 0000
1900  .... = Options: 0x0000 0000 0000 0000
1910  .... = Options: 0x0000 0000 0000 0000
1920  .... = Options: 0x0000 0000 0000 0000
1930  .... = Options: 0x0000 0000 0000 0000
1940  .... = Options: 0x0000 0000 0000 0000
1950  .... = Options: 0x0000 0000 0000 0000
1960  .... = Options: 0x0000 0000 0000 0000
1970  .... = Options: 0x0000 0000 0000 0000
1980  .... = Options: 0x0000 0000 0000 0000
1990  .... = Options: 0x0000 0000 0000 0000
2000  .... = Options: 0x0000 0000 0000 0000
2010  .... = Options: 0x0000 0000 0000 0000
2020  .... = Options: 0x0000 0000 0000 0000
2030  .... = Options: 0x0000 0000 0000 0000
2040  .... = Options: 0x0000 0000 0000 0000
2050  .... = Options: 0x0000 0000 0000 0000
2060  .... = Options: 0x0000 0000 0000 0000
2070  .... = Options: 0x0000 0000 0000 0000
2080  .... = Options: 0x0000 0000 0000 0000
2090  .... = Options: 0x0000 0000 0000 0000
2100  .... = Options: 0x0000 0000 0000 0000
2110  .... = Options: 0x0000 0000 0000 0000
2120  .... = Options: 0x0000 0000 0000 0000
2130  .... = Options: 0x0000 0000 0000 0000
2140  .... = Options: 0x0000 0000 0000 0000
2150  .... = Options: 0x0000 0000 0000 0000
2160  .... = Options: 0x0000 0000 0000 0000
2170  .... = Options: 0x0000 0000 0000 0000
2180  .... = Options: 0x0000 0000 0000 0000
2190  .... = Options: 0x0000 0000 0000 0000
2200  .... = Options: 0x0000 0000 0000 0000
2210  .... = Options: 0x0000 0000 0000 0000
2220  .... = Options: 0x0000 0000 0000 0000
2230  .... = Options: 0x0000 0000 0000 0000
2240  .... = Options: 0x0000 0000 0000 0000
2250  .... = Options: 0x0000 0000 0000 0000
2260  .... = Options: 0x0000 0000 0000 0000
2270  .... = Options: 0x0000 0000 0000 0000
2280  .... = Options: 0x0000 0000 0000 0000
2290  .... = Options: 0x0000 0000 0000 0000
2300  .... = Options: 0x0000 0000 0000 0000
2310  .... = Options: 0x0000 0000 0000 0000
2320  .... = Options: 0x0000 0000 0000 0000
2330  .... = Options: 0x0000 0000 0000 0000
2340  .... = Options: 0x0000 0000 0000 0000
2350  .... = Options: 0x0000 0000 0000 0000
2360  .... = Options: 0x0000 0000 0000 0000
2370  .... = Options: 0x0000 0000 0000 0000
2380  .... = Options: 0x0000 0000 0000 0000
2390  .... = Options: 0x0000 0000 0000 0000
2400  .... = Options: 0x0000 0000 0000 0000
2410  .... = Options: 0x0000 0000 0000 0000
2420  .... = Options: 0x0000 0000 0000 0000
2430  .... = Options: 0x0000 0000 0000 0000
2440  .... = Options: 0x0000 0000 0000 0000
2450  .... = Options: 0x0000 0000 0000 0000
2460  .... = Options: 0x0000 0000 0000 0000
2470  .... = Options: 0x0000 0000 0000 0000
2480  .... = Options: 0x0000 0000 0000 0000
2490  .... = Options: 0x0000 0000 0000 0000
2500  .... = Options: 0x0000 0000 0000 0000
2510  .... = Options: 0x0000 0000 0000 0000
2520  .... = Options: 0x0000 0000 0000 0000
2530  .... = Options: 0x0000 0000 0000 0000
2540  .... =
```

באופן דומה להתחברות, אפשר לבדוק את ניתוק המשתמש מהשרת, נשים לב לפאקטות: Ack 1981, 1984, 1982. נזכיר שבתמונה הבאה, הבית הימני הוא השדה Fin, משמאלו מושםalo Syn שמאופס בכל השירות.

| | | | | | | | | | |
|------|------------|--------------|--------------|-------|-------|-----|---|---|---|
| 1981 | 291.285594 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 0 | 1 | 1 |
| 1982 | 291.285649 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 0 | 1 | 0 |
| 1984 | 291.727135 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 0 | 1 | 1 |
| 1986 | 291.793995 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 0 | 1 | 0 |

אלו הפאקטות האחרונות שהוקלטו עבור המשתמש "ליין" שבפורט 57760.
נשים לב לתהילן handshake 4 עבור הניתוק של ליין מהשרת:
בחבילה מס' 1981 קורה fin=1, ack=1, וזה חביבה שהשרת שולח לליין. כאן השרת שולח בקשה לניתוק החיבור. (בתגובה לפאקטה קודמת שבה ליין מבונן שלח הודעה quit לשרת).
בחבילה מס' 1982 קורה fin=0, ack=1, וזה חביבה ששולח ליין לשרת. כאן ליין מאשר את בקשת השרת לניתוק החיבור.

בחבילה מס' 1984 קורה fin=1, ack=1, וזה חביבה ששולח ליין לשרת. כאן ליין שולח בקשה לניתוק החיבור.

בחבילה מס' 1986 קורה fin=0, ack=1, וזה חביבה ששולח השרת לליין. כאן השרת מאשר את הבקשה של ליין והחיבור נסגר סופית. זו הפאקטה الأخيرة שעבירה.

בשכבות הרשת, כאמור כתובות ה-IP שונות עבור ליין והשרת כי הם רצים במכשירים שונים על אותה רשת.

יש לציין שבשונה מהריצה של כל האפליקציות על אותו המחשב ועל כתובות ה-loopback, השתמשנו כתובות IP אמיתיות של מחשבים שונים כדי לראות תגובה בשכבה הרשת. בתוצאה מכך מתקבלים באמצעות **Datagrams** Wireshark. ניתן לראות את האזיה של הפאקטות ולנתח אותן ב-

את השרת הגדרנו כתובות 0.0.0.0, ככלומר אפשרנו לו להאזין לכל כתובות ה-IP של השרת המקומיית ולקלבל ל��וחות לפי כתובות ה-IP שלהם. הכתובת 0.0.0.0 היא בתובת לוגית לצורך binding שאינה נשלחת ברשת, ולכן היא לא מופיעה בתוור כתובות ה-IP האמיתית של השרת בקובץ הלכידה. (כזכור כתובת השרת היא 192.168.1.56, הכתובת של המחשב).

בוחרות ה-IPv4 של כל הפקות הן מהצורה:

```

▼ Internet Protocol Version 4, Src: 192.168.1.15, Dst: 192.168.1.56
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
▼ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 40
    Identification: 0xe8f9 (59641)
▼ 010. .... = Flags: 0x2, Don't fragment
    0.... .... = Reserved bit: Not set
    .1... .... = Don't fragment: Set
    ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 128
    Protocol: TCP (6)
    Header Checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 192.168.1.15
    Destination Address: 192.168.1.56
    [Stream index: 11]

```

למעט שינויים מעט לנעט בשוחות ה-Header Total Length בהתאם לגודל ה-*Datagram*, ובשדה ה-Identification. בנוסף שמאפעם לפעם בתובות ה-IP מקור ויעד מתחלפים ביניהם (ליין והשרת מחליפים תפקדים, אחד שולח והשני מקבל). נשים לב לסוג הпрוטוקול, 6 עבור חיבור TCP שהגדנו. נשים לב גם לשדה Time to Live, שדה זה תמיד עם ערך 128, ערך שהוא קבוע וגודול יחסית.

הסיבה לשדה זה לא משתנה היא שחברו הרשות הוא בין מכשירים באותו רשת מקומית. על אף שהמכשירים נפרדים ובעלי בתובות IP שונות, אין רכיבי רשות אחרים בדרך, הפקות לא עוזבות את הרשות המקומית ולבן תמיד נשארות עם ערך קבוע וגודול.

בשימוש לתמונות האחוריות, מבון שבקובץ הלכידה שמצוורף (2_clients_chat.pcap) אפשר לעיין בכל ההודעות ברמת האפליקציה, כמו:

ליין שולח לשרת "itzik" בפקטה 658 (משתמש אליו רוצה לשלוח הודעה)

| | 657 | 63.896643 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 69747a696b |
|-------------------------------------|--|--|--------------|------------------|-------|-------|-----|------------|
| | 658 | 64.278017 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 69747a696b |
| ► | Ethernet II, Src: Intel_ | 0000 ea 94 f6 21 f7 c8 08 5b d6 f4 fe 1b 08 00 45 00 | | ...!...[.....E | | | | |
| ▼ | Internet Protocol Version 4 | 0010 00 2d e8 e5 40 00 80 06 00 00 c0 a8 01 0f c0 a8 | | ...@..... | | | | |
| 0100 = Version: | 0020 01 38 e1 a0 30 39 00 b7 64 f7 8a 31 d7 af 50 18 | | | ..8.09...d.1..P. | | | | |
| 0101 = Header Length: | 0030 00 ff 83 b7 00 00 69 74 7a 69 6b | | |it zik | | | | |
| 0000 00.. = Differentiated Services | | | | | | | | |
| Codepoint: | | | | | | | | |

השרות משיב לליין: UserNotFound: S בפקטה 659 כי איזיק הוא לא משתמש קיים:

| | | | | | | | |
|---|--|---|--------------|-------|-------|-----|-----------------|
| 659 | 64.286456 | 192.168.1.56 | 192.168.1.15 | 12345 | 57760 | TCP | 533a55736572... |
| ► Ethernet II, Src: ea:94: ▼ Internet Protocol Version 0100 = Version: 0101 = Header Le ▼ Differentiated Servic 0000 00.. = Differ | 0000 08 5b d6 f4 fe 1b ea 94 f6 21 f7 c8 08 00 45 00 0010 00 36 2c f9 40 00 80 06 4a 31 c0 a8 01 38 c0 a8 0020 01 0f 30 39 e1 a0 8a 31 d7 af 00 b7 64 fc 50 18 0030 10 04 94 04 00 00 53 3a 55 73 65 72 4e 6f 74 46 0040 6f 75 6e 64 | [..... ! .. E 6, @ .. J1 .. 8 .. 09 .. 1 .. d .. P S: UserNotF ound | | | | | |

הודעה שלירן שלח למאור דרך השירות בפקטה 1674 :
"Wanna come over? We can order pizza"

| | | | | | | | |
|--|---|---|--------------|-------|-------|-----|-----------------|
| 1674 | 235.291860 | 192.168.1.15 | 192.168.1.56 | 57760 | 12345 | TCP | 446f20796f75... |
|00 = Explicit Total Length: 107 Identification: 0xe8f ▼ 010. = Flags: 0x 0.... = Reserv .1... = Don't ..0. = More f ...0 0000 0000 0000 = Time to Live: 128 Protocol: TCP (6) | 0000 ea 94 f6 21 f7 c8 08 5b d6 f4 fe 1b 08 00 45 00 0010 00 6b e8 f2 40 00 80 06 00 00 c0 a8 01 0f c0 a8 0020 01 38 e1 a0 30 39 00 b7 65 68 8a 31 d8 28 50 18 0030 00 ff 83 f5 00 00 44 6f 20 79 6f 75 20 77 61 6e 0040 6e 61 20 63 6f 6d 65 20 6f 76 65 72 20 61 6e 64 0050 20 77 61 74 63 68 20 69 74 20 74 6f 67 65 74 68 0060 65 72 3f 20 57 65 20 63 61 6e 20 6f 72 64 65 72 0070 20 70 69 7a 7a 61 20 3b 29 | ! .. L .. E .. k .. @ .. 8 .. 09 .. eh .. 1 .. (P Do you wan na come over and watch i t togeth er? We c an order pizza ;) | | | | | |

מלבד הדוגמאות האלו ניתן למצוא את כל שאר ההודעות והפקטות בקובץ הלכידה המצורף.

נציג דוגמה לשיחה בין חמישה(!) משתמשים. נזכיר שהצ'אט לא קבוצתי; כל משתמש צריך לפנות למשתמש אחד. יש לו כמובן את היכולת לפנות למשתמש אחר ולהחליף בין המשתמשים בכל רגע, אבל בכל פעם ההודעה תישלח רק למשתמש שנבחרה. נציג את פלט השירות שמסכם את השיחה. הודעות בין המשתמשים "ליין", "אושר", "נעעה", "עדן" ו"גל" מסומנות בקו אדום, הפלט נראה כך:

הפלט של המשתמש "Liran":

```
Connection to server established
Welcome to the chat! What is your name?
Liran
Hello Liran! Have fun chatting.
You can quit any time by typing 'QUIT'.
```

Who do you want to message?

Osher

Connection created

Hey, want to go see a movie tonight?

Osher: I was just talking with Eden about it. join us

ok I can drive us

Osher: ok I'll tell eden

back

Who do you want to message?

Noa

Connection created

WANNA COME WITH US?

QUIT

הפלט של המשתמש "Osher":

```
Connection to server established
Welcome to the chat! What is your name?
Osher
Hello Osher! Have fun chatting.
You can quit any time by typing 'QUIT'.
```

Who do you want to message?

Eden

Connection created

Want to go to the cinema later?

Liran: Hey, want to go see a movie tonight?

Back

Who do you want to message?

Line
Liran

Elan
Connection created

I was just talking with Eden about it, join us.

Iran: ok I can drive us

ok I'll tell eden

back

Who do you want to message?

eden

User Was Not Found

Who do you want to message?

Eden

Connection created

Liran can pick us up

Eden: Ok, also Gal comes too

back

Who do you want to message?

Nra

New
Connection created

WANNA COME WTTH US?

quit

הפלט של המשתמשת "Eden":

```
Connection to server established
Welcome to the chat! What is your name?
Eden
Hello Eden! Have fun chatting.
You can quit any time by typing 'QUIT'.
```

Who do you want to message?
Osher: Want to go to the cinema later?

Who do you want to message?

Gal

Connection created

Are you free for a movie tonight?

Gal: Yeah sure

Osher: Liran can pick us up

back

Who do you want to message?

Osher

Connection created

Ok, also Gal comes too

back

1000

Liran

back

Noa

Who do you want to message?
None

Noa
6

Connection created

WANNA COME WITH US?

quit

הפלט של המשתמש "Gal":

```
Connection to server established
Welcome to the chat! What is your name?
Gal
Hello Gal! Have fun chatting.
You can quit any time by typing 'QUIT'.
```

Who do you want to message?
Eden: Are you free for a movie tonight?

Who do you want to message?
Eden
Connection created

Yeah sure

back

Who do you want to message?
Noa
Connection created

WANNA COME WITH US?

הפלט של המשתמשת "Noa":

```
Connection to server established
Welcome to the chat! What is your name?
Noa
Hello Noa! Have fun chatting.
You can quit any time by typing 'QUIT'.
```

Who do you want to message?
Liran: WANNA COME WITH US?

Who do you want to message?
Osher: WANNA COME WITH US?

Who do you want to message?
Eden: WANNA COME WITH US?

Who do you want to message?
Gal: WANNA COME WITH US?

Who do you want to message?
Liran
Connection created

back

Who do you want to message?
Osher
Connection created

back

Who do you want to message?
Eden
Connection created

back

Who do you want to message?
Gal
Connection created

quit

שיחה זו בין 5 משתמשים גם כן הוקלטה ב-Wireshark.
במקרה זה כל ששת האפליקציות (שרת + 5 משתמשים) הורצו על אותו המחשב.
קובץ הלכידה מצורף בשם 5_client_chat.pcap

ביבליוגרפיה ומקורות

- Geeks For Geeks on TCP analysis using wireshark –
<https://www.geeksforgeeks.org/ethical-hacking/tcp-analysis-using-wireshark>
- Jim Kurose on Wireshark –
<https://youtu.be/kCwd2YoJcvg?si=Tpu3bIBNL6yWyGfe>
- Nicolas Day on C++ socket programming (part 1) -
<https://youtu.be/gntyAFoZp-E?si=59aVk5LLMTQO1X15>
- Nicolas Day on C++ socket programming (part 2)
https://youtu.be/sXW_sNGvqcU?si=ZmPkmQ1Q2F5KMAow
- Jim Kurose on socket programming –
https://youtu.be/_iHMMo7SDfQ?si=q-hwG4Nf6ODydAAQ
- stack overflow Q&A's
<https://stackoverflow.com/questions>

as well as other public CS forums and youtube tutorials regarding c++ libraries and
wireshark

לא נעשה שימוש בכלל AI בשום שלב של הפרויקט. כל פסקה ותמונה בדו"ח זה וכל שאר הקבצים שמצורפים: כולל קבצי לכידה ב-Wireshark, כולל קבצי קוד בשפת C++, כולל הלוגיקה שביהם וכל תיעוד שביהם נכתבו על ידי. אני רأיתי ובדיעבד בסיסים רואה אם אולי יותר בפרויקט בהזדמנות גדולה להבין חלק גדול מהחומר ולראות אותו מנקודת מבט מעשית, פרקטית. מיידך הוא עוזר לי לחזק את ההבנה במושגים ונוסחים רבים בסילבוס.

Fin
