

## רשתות תרגיל 2 חלק 1

מצורף כאן צילום מסך של תקשורת בין שני מחשבים שונים. הלקוח (10.0.0.16) מריץ את הקובץ tcp\_client.py אשר פונה לשרת (10.0.0.32) המריץ את הקובץ tcp\_server.py.

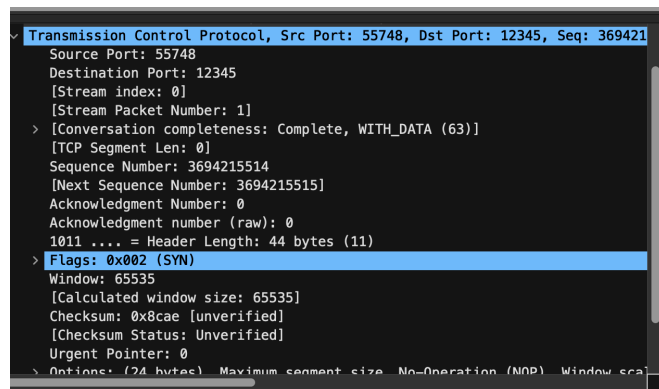
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.0.16	10.0.0.32	TCP	78	55748 → 12345 [SYN] Seq=3694215514 Win=65535 Len=0 MSS=1460 WS=64 TSval=3613366
2	0.040079	10.0.0.32	10.0.0.16	TCP	74	12345 → 55748 [SYN, ACK] Seq=1667296290 Ack=3694215515 Win=65160 Len=0 MSS=1460
3	0.040246	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [ACK] Seq=3694215515 Ack=1667296291 Win=131776 Len=0 TSval=3613366
4	0.040296	10.0.0.16	10.0.0.32	TCP	78	55748 → 12345 [PSH, ACK] Seq=3694215515 Ack=1667296291 Win=131776 Len=12 TSval=3613366
5	0.044234	10.0.0.32	10.0.0.16	TCP	66	12345 → 55748 [ACK] Seq=1667296291 Ack=3694215527 Win=65152 Len=0 TSval=3259653
6	0.045122	10.0.0.32	10.0.0.16	TCP	78	12345 → 55748 [PSH, ACK] Seq=1667296291 Ack=3694215527 Win=65152 Len=12 TSval=3259653
7	0.045124	10.0.0.32	10.0.0.16	TCP	66	12345 → 55748 [FIN, ACK] Seq=1667296303 Ack=3694215527 Win=65152 Len=0 TSval=3259653
8	0.045223	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [ACK] Seq=3694215527 Ack=1667296303 Win=131776 Len=0 TSval=3613366
9	0.045270	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [ACK] Seq=3694215527 Ack=1667296304 Win=131776 Len=0 TSval=3613366
10	0.045349	10.0.0.16	10.0.0.32	TCP	75	55748 → 12345 [PSH, ACK] Seq=3694215527 Ack=1667296304 Win=131776 Len=9 TSval=3613366
11	0.045388	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [FIN, ACK] Seq=3694215536 Ack=1667296304 Win=131776 Len=0 TSval=3613366
12	0.050230	10.0.0.32	10.0.0.16	TCP	54	12345 → 55748 [RST] Seq=1667296304 Win=0 Len=0
13	0.051229	10.0.0.32	10.0.0.16	TCP	54	12345 → 55748 [RST] Seq=1667296304 Win=0 Len=0

### תהליך הקמת החיבור (Three-Way Handshake)

תהליך זה מתרחש בחבילות 1 עד 3. זהו תהליך לחיצת היד המשולשת.

1. שלב ראשון (Packet 1) - שליחת SYN:

- הלקוח (10.0.0.16) יוזם את החיבור ושולח הודעת SYN לשרת (10.0.0.32).
- מספרים סידוריים: המקור מגריל מספר סידורי התחלתי.
- Seq = 3694215514
- Ack = 0 (כי עדיין לא התקבל כלום מהצד השני).
- משמעות: אני רוצה לפתוח איתך שיחה, המספר הסידורי שלי מתחיל ב-"3694215514".



2. שלב שני (Packet 2) - שליחת SYN, ACK:

- תיאור: השרת מקבל את הבקשה, ומחזיר הודעה המכילה גם SYN (כדי לפתוח ערוץ משלו) וגם ACK (אישור הבקשה של הלקוח).
- מספרים סידוריים:
- Seq = 1667296290 (המספר הסידורי ההתחלתי שהשרת הגריל).
- Ack = 3694215515 (החישוב הוא: Seq של הלקוח + 1).

- משמעות: קיבלתי את ההודעה שלך והוספתי לה 1, ואני גם רוצה לפתוח איתך שיחה במספר הסידורי שלי.

```

Transmission Control Protocol, Src Port: 12345, Dst Port: 55748, Seq: 166729
  Source Port: 12345
  Destination Port: 55748
  [Stream index: 0]
  [Stream Packet Number: 2]
  > [Conversation completeness: Complete, WITH_DATA (63)]
  [TCP Segment Len: 0]
  Sequence Number: 1667296290
  [Next Sequence Number: 1667296291]
  Acknowledgment Number: 3694215515
  1010 .... = Header Length: 40 bytes (10)
  > Flags: 0x012 (SYN, ACK)
  Window: 65160
  [Calculated window size: 65160]
  Checksum: 0x2eaf [unverified]
  [Checksum Status: Unverified]

```

### 3. שלב שלישי (Packet 3) - שליחת ACK:

- הלקוח מאשר את קבלת ה-SYN של השרת ע"י שליחת ACK. החיבור כעת הוקם.
- מספרים סידוריים:

- Seq = 3694215515 (התקדם ב-1 לעומת חבילה 1, כי דגל ה-SYN צורך מספר סידורי אחד).
- Ack = 1667296291 (החישוב הוא: ה-Seq של השרת + 1).

```

> Frame 3: Packet, 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on
> Ethernet II, Src: de:6e:31:1c:f5:de (de:6e:31:1c:f5:de), Dst: Intel_46:ba:9a
> Internet Protocol Version 4, Src: 10.0.0.16, Dst: 10.0.0.32
Transmission Control Protocol, Src Port: 55748, Dst Port: 12345, Seq: 369421
  Source Port: 55748
  Destination Port: 12345
  [Stream index: 0]
  [Stream Packet Number: 3]
  > [Conversation completeness: Complete, WITH_DATA (63)]
  [TCP Segment Len: 0]
  Sequence Number: 3694215515
  [Next Sequence Number: 3694215515]
  Acknowledgment Number: 1667296291
  1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x010 (ACK)
  Window: 2059
  [Calculated window size: 131776]
  [Window size scaling factor: 64]
  Checksum: 0x53d0 [unverified]

```

## העברת נתונים וניתוח המספרים

### 1. שליחת מידע (Packet 4):

- תיאור: הלקוח (10.0.0.16) שולח מידע לשרת. ניתן לראות את הדגל שמבקש להעביר את המידע לאפליקציה מיד.
- גודל המידע: Len = 12 (רואים זאת בעמודת Length או Info).
- מספר סידורי: Seq = 3694215515.

```
Sequence Number: 3694215515
[Next Sequence Number: 3694215527]
Acknowledgment Number: 1667296291
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x018 (PSH, ACK)
Window: 2059
[Calculated window size: 131776]
[Window size scaling factor: 64]
Checksum: 0x4960 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
> [Timestamps]
> [SEQ/ACK analysis]
[Client Contiguous Streams: 1]
[Server Contiguous Streams: 1]
TCP payload (12 bytes)
Data (12 bytes)
Data: 596f617620456c6861646164
```

0000	f4 06 69 46 ba 9a de 6e 31 1c f5 de 08 00 45 00	..iF...n 1.....E
0010	00 40 00 00 40 00 40 06 26 89 0a 00 00 10 0a 00	@...@.@ &.....
0020	00 20 d9 c4 30 39 dc 31 41 5b 63 60 ec 23 80 18	...09.1 A[c`.#...
0030	08 0b 49 60 00 00 01 01 08 0a d7 5f 97 5d c2 4a	..I'...._..].J
0040	5e 9b 59 6f 61 76 20 45 6c 68 61 64 61 64	^..Yoav E lhada

### 2. אישור קבלת מידע (Packet 5):

- תיאור: השרת (10.0.0.32) מאשר את קבלת המידע.
- חישוב ה-Ack: השרת מבצע חישוב פשוט: המספר הסידורי הקודם שקיבל + גודל המידע שקיבל.
- החישוב במספרים:
  - 3694215515 (ה-Seq של חבילה 4)
  - + 12 (ה-Len של חבילה 4)
  - = 3694215527
- לכן בחבילה 5 אנו רואים Ack = 3694215527. זהו האישור שהמידע התקבל בשלמותו והשרת מצפה לבית הבא במספר זה.

```

[Stream Packet Number: 5]
> [Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 0]
Sequence Number: 1667296291
[Next Sequence Number: 1667296291]
Acknowledgment Number: 3694215527
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x010 (ACK)
Window: 509
[Calculated window size: 65152]
[Window size scaling factor: 128]
Checksum: 0x59cd [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
> [Timestamps]
> [SEQ/ACK analysis]
[Client Contiguous Streams: 1]
[Server Contiguous Streams: 1]

```

○

### 3. תשובת השרת (Packet 6):

- תיאור: השרת (10.0.0.32) שולח חזרה ללקוח את המחרוזת "YOAV ELHADAD".
- ה Seq הוא 1667296291 (זהה ל-ACK הקודם, כי השרת עדיין לא שלח מידע, רק אישר).
- ה Ack הוא 3694215527 (השרת עדיין מצפה לבית ה-27, כלומר מאשר שקיבל את השם).
- האורך הוא 12 בתים

```

Transmission Control Protocol, Src Port: 12345, Dst Port: 55748, Seq: 1, Ack: 13, Len: 12
Source Port: 12345
Destination Port: 55748
[Stream index: 0]
[Stream Packet Number: 6]
> [Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 12]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 1667296291
[Next Sequence Number: 13 (relative sequence number)]
Acknowledgment Number: 13 (relative ack number)
Acknowledgment number (raw): 3694215527
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x018 (PSH, ACK)
Window: 509
[Calculated window size: 65152]
[Window size scaling factor: 128]
Checksum: 0xcffd [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
> [Timestamps]
> [SEQ/ACK analysis]
[Client Contiguous Streams: 1]
[Server Contiguous Streams: 1]
TCP payload (12 bytes)

```

### 4. שליחת תעודת הזהות (Packet 8)

- תיאור: הלקוח (10.0.0.16) שולח את ההודעה השנייה המכילה את תעודת הזהות: "315853793"
- החישוב במספרים:
  - $Seq=3694215527$ , המספר הסידורי הקודם של הלקוח בחבילה 4 היה 3694215515, שלחנו 12 בתים ולכן  $3694215515 + 12 = 3694215527$ .
  - $Ack=1667296303$ , הלקוח קיבל מהשרת את תשובת ה Echo בחבילה 6 שהייתה באורך 12 בתים והתחילה ב 1667296291 לכן הלקוח מאשר קבלה ומצפה ל:  $1667296303 = 12 + 1667296291$ .

## ■ האורך הוא 9 בתים

```

Transmission Control Protocol, Src Port: 55748, Dst Port: 12345, Seq: 13, Ack: 13, Len: 0
  Source Port: 55748
  Destination Port: 12345
  [Stream index: 0]
  [Stream Packet Number: 8]
  [Conversation completeness: Complete, WITH_DATA (63)]
  [TCP Segment Len: 0]
  Sequence Number: 13 (relative sequence number)
  Sequence Number (raw): 3694215527
  [Next Sequence Number: 13 (relative sequence number)]
  Acknowledgment Number: 13 (relative ack number)
  Acknowledgment number (raw): 1667296303
  1000 .... = Header Length: 32 bytes (8)
  [Flags: 0x010 (ACK)]
  Window: 2059
  [Calculated window size: 131776]
  [Window size scaling factor: 64]
  Checksum: 0x53af [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  [Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps]
  [Timestamps]
  [SEQ/ACK analysis]
  [Client Contiguous Streams: 1]
  [Server Contiguous Streams: 1]

```

## חלק 3: סיום החיבור

1. ניסיון סיום (Packet 7):
  - השרת (10.0.0.32) שולח ACK, FIN.
  - משמעות: השרת מודיע "סיימתי לשלוח מידע, אני רוצה לסגור את החיבור מצדי".
2. סגירה מיידיה / תקלה (Packet 12-13):
  - רואים הודעות של RST Reset.
  - הסבר: במקום תהליך סגירה מסודר (שבו הצד השני שולח FIN משלו ו-ACK), נשלחה פקודת "Reset". זה קורה בדרך כלל כאשר צד אחד סוגר את התוכנה בכוח, או שהפורט כבר לא מאזין, והוא "שובר" את החיבור מיד במקום לסגור אותו בעדינות.

No.	Time	Source	Destination	Protocol	Length	Info
11	0.045388	10.0.0.16	10.0.0.32	TCP	66	55748 → 12345 [FIN, ACK] Seq=3694215536 Ack=1667296304 Win=131776 Len=0 TSval=3
12	0.050230	10.0.0.32	10.0.0.16	TCP	54	12345 → 55748 [RST] Seq=1667296304 Win=0 Len=0
13	0.051229	10.0.0.32	10.0.0.16	TCP	54	12345 → 55748 [RST] Seq=1667296304 Win=0 Len=0

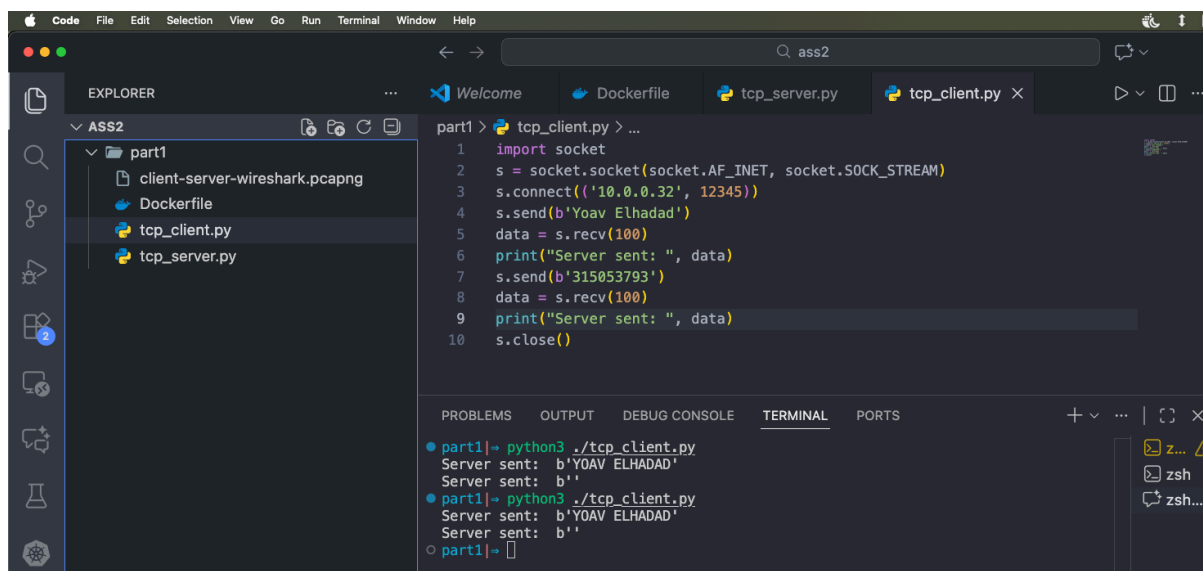
  

```

Frame 12: Packet, 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
  Ethernet II, Src: Intel_46:ba:9a (f4:06:09:46:ba:9a), Dst: de:6e:31:1c:f5:de
  Internet Protocol Version 4, Src: 10.0.0.32, Dst: 10.0.0.16
  Transmission Control Protocol, Src Port: 12345, Dst Port: 55748, Seq: 1667296304
    Source Port: 12345
    Destination Port: 55748
    [Stream index: 0]
    [Stream Packet Number: 12]
    [Conversation completeness: Complete, WITH_DATA (63)]
    [TCP Segment Len: 0]
    Sequence Number: 1667296304
    [Next Sequence Number: 1667296304]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    0101 .... = Header Length: 20 bytes (5)
    [Flags: 0x004 (RST)]
    Window: 0
    [Calculated window size: 0]
    [Window size scaling factor: 128]

```

נספחים:  
הקוד של הלקוח:



The screenshot shows a Visual Studio Code editor window with a project named 'ASS2'. The Explorer sidebar on the left shows a folder 'part1' containing files: 'client-server-wireshark.pcapng', 'Dockerfile', 'tcp\_client.py', and 'tcp\_server.py'. The main editor area displays the code for 'tcp\_client.py' in the 'part1' directory. The code is a Python script that uses the 'socket' module to establish a TCP connection to '10.0.0.32' on port 12345. It sends the string 'Yoav Elhadad' and receives a response, which it prints. It then sends the string '315053793' and receives another response, which it also prints. Finally, it closes the socket connection.

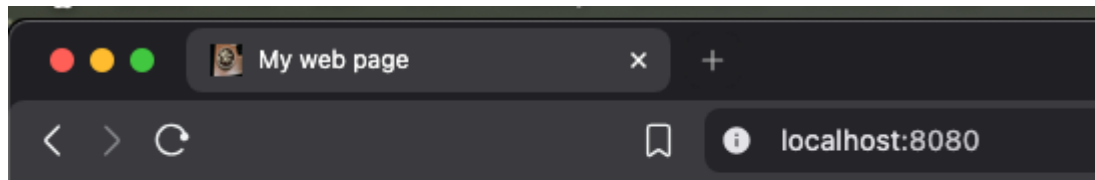
```
1 import socket
2 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 s.connect(('10.0.0.32', 12345))
4 s.send(b'Yoav Elhadad')
5 data = s.recv(100)
6 print("Server sent: ", data)
7 s.send(b'315053793')
8 data = s.recv(100)
9 print("Server sent: ", data)
10 s.close()
```

Below the code editor, the 'TERMINAL' tab is active, showing the output of running the script twice. The first run shows the server sending 'b'YOAV ELHADAD'' and then an empty string 'b''. The second run shows the server sending 'b'YOAV ELHADAD'' and then 'b'' again.

```
part1| python3 ./tcp_client.py
Server sent: b'YOAV ELHADAD'
Server sent: b''

part1| python3 ./tcp_client.py
Server sent: b'YOAV ELHADAD'
Server sent: b''

part1|
```



**hello**