מבוא לרשתות תקשורת – תרגיל מספר 2

חלק א:

לקוח:

IP: 10.0.2.5

```python
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('10.0.2.4', 12345))
s.send(b'yarin')
data = s.recv(100)
print("Server sent: ", data.decode())
s.send(b'318229143')
data = s.recv(100)
print("Server sent: ", data.decode())
s.close()
```

שרת:

IP: 10.0.2.4

```python
import socket
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(('', 12345))
server.listen(5)

while True:
    client_socket, client_address = server.accept()
    print('Connection from: ', client_address)
    data = client_socket.recv(100)
    print('Received: ', data.decode())
    client_socket.send(data.upper())
    data = client_socket.recv(100)
    print('Received: ', data.decode())
    client_socket.send(data.upper())
    client_socket.close()
    print('Client disconnected')
```

פלט לקוח

```
Server sent:  YARIN
Server sent:  318229143
```

פלט שרת :

```
Connection from:  ('10.0.2.5', 50802)
Received:  yarin
Received:  318229143
Client disconnected
```

(השרת נמצא על גבי פורט 12344)

	Time	Source	Destination	Protoco	Length	Info
509	215.820001398	10.0.2.5	10.0.2.4	TCP	74	50802 → 12344 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=438381722 TSecr=0 WS=128
510	215.820090245	10.0.2.4	10.0.2.5	TCP	74	12344 → 50802 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2053375273 TSecr=438381722 WS=128
511	215.820614201	10.0.2.5	10.0.2.4	TCP	66	50802 → 12344 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=438381723 TSecr=2053375273

בשלב הראשון של ההתקשרות הלקוח והשרת מבצעים " לחיצת ידים" ,בה הם מעבירים אחד לשני מידע רלוונטי לגבי ההתקשרות.

1. בשורה הראשונה (509) הלקוח (שורה שלוש בקוד המקור של הלקוח, בה הוא מבצע connect) הלקוח שולח SYN ,(בקשת התחברות)
   בתחילית הTCP יידלק דגל SYN.

```
Sequence Number: 0    (relative sequence number)
Sequence Number (raw): 614446167
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
```

כפי שניצן לראות ה sequence number בשלב זה הוא 614446167 ,ה acknowledgement number מסומן באפס (אין מידע שאמור היה להתקבל לכן לא רלוונטי בשלב זה).

2. בשורה השנייה (510) השרת מאשר את בקשת הסנכרון ושולח בעצמו ACK נשים לב שלמרות שבשלב זה לא הועבר כלל מידע בשכבת האפליקציה עדיין נוסף בית לACK דבר הנקרא ביית פנטום מסמן לצד השני שבקשת הSYN בוצעה בהצלחה.
   בתחילית הTCP יידלקו דגלי SYN וACK השרת מאשר את הSYN של הלקוח ושולח SYN שלו.

```
Sequence Number: 0    (relative sequence number)
Sequence Number (raw): 120054468
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 1    (relative ack number)
Acknowledgment number (raw): 614446168
```

ה sequence number בשלב זה הוא 120054468 ,ה acknowledgement number הוא 614446168.

3. בשורה השלישית הלקוח מחזיר לשרת אישור על בקשת ה SYN שלו .
   בתחילית הTCP ACK יידלק דגל מהלקוח שמאשר את ה SYN של השרת.

```
Sequence Number: 1    (relative sequence number)
Sequence Number (raw): 614446168
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 1    (relative ack number)
Acknowledgment number (raw): 120054469
```

ה sequence number בשלב זה הוא 614446168 ,ה acknowledgement number הוא 120054469. (שוב Byte פנטום).

4. 512 215.820841173 10.0.2.5 10.0.2.4 TCP 71 50802 → 12344 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=5 TSval=438381723 TSecr=2053375273

בשורה זו הלקוח שולח לשרת הודעה עם הטקסט yarin , בנוסף שולח ACK עם המידע שהתקבל עד כה.
בתחילית הTCP יידלקו ACK וכן PSH שמבקש ממערכת ההפעלה לשלוח את ההודעה ללא דיחוי .

```
Sequence Number: 1    (relative sequence number)
Sequence Number (raw): 614446168
[Next Sequence Number: 6    (relative sequence number)]
Acknowledgment Number: 1    (relative ack number)
Acknowledgment number (raw): 120054469
```

ה sequence number בשלב זה הוא 614446168, ה acknowledgement number הוא 120054469.(כמו מקודם).

5. 513 215.820846595 10.0.2.4 10.0.2.5 TCP 66 12344 → 50802 [ACK] Seq=1 Ack=6 Win=65280 Len=0 TSval=2053375274 TSecr=438381723

בשורה הזאת השרת מחזיר ללקוח אישור על קבלת ההודעה ובהתאם מעדכן את הACK (הגדלה של 5 בתים בהתאם גודל ה Data).
בתחילית הTCP יידלק דגל SYN.

```
Sequence Number: 1    (relative sequence number)
Sequence Number (raw): 120054469
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 6    (relative ack number)
Acknowledgment number (raw): 614446173
```

ה sequence number בשלב זה הוא 120054469, ה acknowledgement number הוא 614446173 .

6. 514 215.821252529 10.0.2.4 10.0.2.5 TCP 71 12344 → 50802 [PSH, ACK] Seq=1 Ack=6 Win=65280 Len=5 TSval=2053375274 TSecr=438381723

לאחר קבלת המידע מהלקוח השרת מעבד אותו (הופך את הטקסט לאותיות גדולות) ושולח ללקוח בנוסף מעדכן ACK נוכחי.
בתחילית הTCP יידלקו ACK וכן PSH שמבקש ממערכת ההפעלה לשלוח את ההודעה ללא דיחוי .

```
Sequence Number: 1    (relative sequence number)
Sequence Number (raw): 120054469
[Next Sequence Number: 6    (relative sequence number)]
Acknowledgment Number: 6    (relative ack number)
Acknowledgment number (raw): 614446173
```

ה sequence number בשלב זה הוא 120054469, ה acknowledgement number הוא 614446173.

7. 515 215.821824675 10.0.2.5 10.0.2.4 TCP 66 50802 → 12344 [ACK] Seq=6 Ack=6 Win=64256 Len=0 TSval=438381724 TSecr=2053375274

הלקוח מחזיר לשרת אישור על קבלת ההודעה .
בתחילית הTCP יידלק דגל SYN.

```
Sequence Number: 6    (relative sequence number)
Sequence Number (raw): 614446173
[Next Sequence Number: 6    (relative sequence number)]
Acknowledgment Number: 6    (relative ack number)
Acknowledgment number (raw): 120054474
```

ה sequence number בשלב זה הוא 614446173 ה acknowledgement number הוא 120054474 (עלייה של 5 כאורך הטקסט שהתקבל אצל הלקוח).

8. 516 215.822245598 10.0.2.5 10.0.2.4 TCP 75 50802 → 12344 [PSH, ACK] Seq=6 Ack=6 Win=64256 Len=9 TSval=438381724 TSecr=2053375274

הלקוח שולח לשרת את מס תעודת הזהות.

בתחילית הTCP יידלקו ACK וכן PSH שמבקש ממערכת ההפעלה לשלוח את ההודעה ללא דיחוי .

```
Sequence Number: 6    (relative sequence number)
Sequence Number (raw): 614446173
[Next Sequence Number: 15    (relative sequence number)]
Acknowledgment Number: 6    (relative ack number)
Acknowledgment number (raw): 120054474
```

ה sequence number בשלב זה הוא 614446173 ה acknowledgement number הוא 120054474 .

9. 517 215.822251952 10.0.2.4 10.0.2.5 TCP 66 12344 → 50802 [ACK] Seq=6 Ack=15 Win=65280 Len=0 TSval=2053375275 TSecr=438381724

השרת מחזיר אישור על קבלת ההודעה
בתחילית הTCP יידלק דגל ACK.

```
Sequence Number: 6    (relative sequence number)
Sequence Number (raw): 120054474
[Next Sequence Number: 6    (relative sequence number)]
Acknowledgment Number: 15    (relative ack number)
Acknowledgment number (raw): 614446182
```

ה sequence number בשלב זה הוא 120054469, ה acknowledgement number הוא 614446182 (עלייה של 9 בתים של המידע שהועבר (הת"ז)).

.10 518 215.822564769 10.0.2.4 10.0.2.5 TCP 75 12344 → 50802 [PSH, ACK] Seq=6 Ack=15 Win=65280 Len=9 TSval=2053375275 TSecr=438381724

השרת שולח חזרה את תעודת הזהות ללקוח .

בתחילית הTCP יידלקו ACK וכן PSH שמבקש ממערכת ההפעלה לשלוח את ההודעה ללא דיחוי .

```
Sequence Number: 6    (relative sequence number)
Sequence Number (raw): 120054474
[Next Sequence Number: 15    (relative sequence number)]
Acknowledgment Number: 15    (relative ack number)
Acknowledgment number (raw): 614446182
```

ה sequence number בשלב זה הוא 120054474, ה acknowledgement number הוא 614446182 .

.11 519 215.822743666 10.0.2.4 10.0.2.5 TCP 66 12344 → 50802 [FIN, ACK] Seq=15 Ack=15 Win=65280 Len=0 TSval=2053375276 TSecr=438381724

מיד בסיום שליחת ההודעה הקודמת השרת מנתק את הסוקט של הלקוח לכן שולח (בנוסף ל ACK) הודעת FIN, הודעה שמצהירה מבחינת הצד ששלך אותה שהוא סיים "לומר" את מה שהוא רצה ומבחינתו אפשר לסיים את השיחה (בפועל בשלב זה הקשר עוד לא נותק).

```
Sequence Number: 15    (relative sequence number)
Sequence Number (raw): 120054483
[Next Sequence Number: 16    (relative sequence number)]
Acknowledgment Number: 15    (relative ack number)
Acknowledgment number (raw): 614446182
```

בתחילית ה TCP יידלק דגל ACK ודגל FIN .

ה sequence number בשלב זה הוא 120054474, ה acknowledgement number הוא 614446182.

.12 520 215.823014216 10.0.2.5 10.0.2.4 TCP 66 50802 → 12344 [ACK] Seq=15 Ack=15 Win=64256 Len=0 TSval=438381725 TSecr=2053375275

הלקוח החזיר לשרת ACK על תעודת הזהות שנשלחה לו (נשים לב שלא מדובר באישור על ה FIN)

בתחילית הTCP יידלק דגל ACK .

```
Sequence Number: 15    (relative sequence number)
Sequence Number (raw): 614446182
[Next Sequence Number: 15    (relative sequence number)]
Acknowledgment Number: 15    (relative ack number)
Acknowledgment number (raw): 120054483
```

ה sequence number בשלב זה הוא 614446182 ה acknowledgement number הוא 120054483 .

.13 521 215.823221406 10.0.2.5 10.0.2.4 TCP 66 50802 → 12344 [FIN, ACK] Seq=15 Ack=15 Win=64256 Len=0 TSval=438381726 TSecr=2053375275

הלקוח מאשר את הודעת הFIN ושולח הודעת FIN גם.

בתחילית ה TCP יידלק דגל ACK ודגל FIN .

```
Sequence Number: 16    (relative sequence number)
Sequence Number (raw): 614446183
[Next Sequence Number: 16    (relative sequence number)]
Acknowledgment Number: 16    (relative ack number)
Acknowledgment number (raw): 120054484
```

ה sequence number בשלב זה הוא 614446183 ה acknowledgement number הוא120054484 (בית פאנטום).

.14 522 215.823221463 10.0.2.5 10.0.2.4 TCP 66 50802 → 12344 [ACK] Seq=16 Ack=16 Win=64256 Len=0 TSval=438381726 TSecr=2053375276

הלקוח שולח לשרת ACK נוכחי.

בתחילית ה TCP יידלק דגל ACK.

```
Sequence Number: 16    (relative sequence number)
Sequence Number (raw): 614446183
[Next Sequence Number: 16    (relative sequence number)]
Acknowledgment Number: 16    (relative ack number)
Acknowledgment number (raw): 120054484
```

ה sequence number בשלב זה הוא 614446183 ה acknowledgement number הוא120054484.

.15 523 215.823412371 10.0.2.4 10.0.2.5 TCP 66 12344 → 50802 [ACK] Seq=16 Ack=16 Win=65280 Len=0 TSval=2053375276 TSecr=438381726

לסיום השרת שולח אישור על בקשת הFIN של הלקוח ומסיים את ההתקשרות עם הלקוח (סוגר את הסוקט ממש).

בתחילית ה TCP יידלק דגל ACK.

```
Sequence Number: 16    (relative sequence number)
Sequence Number (raw): 120054484
[Next Sequence Number: 16    (relative sequence number)]
Acknowledgment Number: 16    (relative ack number)
Acknowledgment number (raw): 614446183
```

ה sequence number בשלב זה הוא 120054484, ה acknowledgement number הוא 614446183.

כול ההתקשרות באופן מרוכז:

tcp && tcp.port == 12344

No.	Time	Source	Destination	Protoco	Length	Info
509	215.820001398	10.0.2.5	10.0.2.4	TCP	74	50802 → 12344 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=438381722 TSecr=0 WS=128
510	215.820090245	10.0.2.4	10.0.2.5	TCP	74	12344 → 50802 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2053375273 TSecr=438381722 WS=128
511	215.820614201	10.0.2.5	10.0.2.4	TCP	66	50802 → 12344 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=438381723 TSecr=2053375273
512	215.820841173	10.0.2.5	10.0.2.4	TCP	71	50802 → 12344 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=5 TSval=438381723 TSecr=2053375273
513	215.820846595	10.0.2.4	10.0.2.5	TCP	66	12344 → 50802 [ACK] Seq=1 Ack=6 Win=65280 Len=0 TSval=2053375274 TSecr=438381723
514	215.821252529	10.0.2.4	10.0.2.5	TCP	71	12344 → 50802 [PSH, ACK] Seq=1 Ack=6 Win=65280 Len=5 TSval=2053375274 TSecr=438381723
515	215.821824675	10.0.2.5	10.0.2.4	TCP	66	50802 → 12344 [ACK] Seq=6 Ack=6 Win=64256 Len=0 TSval=438381724 TSecr=2053375274
516	215.822245598	10.0.2.5	10.0.2.4	TCP	75	50802 → 12344 [PSH, ACK] Seq=6 Ack=6 Win=64256 Len=9 TSval=438381724 TSecr=2053375274
517	215.822251952	10.0.2.4	10.0.2.5	TCP	66	12344 → 50802 [ACK] Seq=6 Ack=15 Win=65280 Len=0 TSval=2053375275 TSecr=438381724
518	215.822564769	10.0.2.4	10.0.2.5	TCP	75	12344 → 50802 [PSH, ACK] Seq=6 Ack=15 Win=65280 Len=9 TSval=2053375275 TSecr=438381724
519	215.822743666	10.0.2.4	10.0.2.5	TCP	66	12344 → 50802 [FIN, ACK] Seq=15 Ack=15 Win=65280 Len=0 TSval=2053375276 TSecr=438381724
520	215.823014216	10.0.2.5	10.0.2.4	TCP	66	50802 → 12344 [ACK] Seq=15 Ack=15 Win=64256 Len=0 TSval=438381725 TSecr=2053375275
521	215.823221406	10.0.2.5	10.0.2.4	TCP	66	50802 → 12344 [FIN, ACK] Seq=15 Ack=15 Win=64256 Len=0 TSval=438381726 TSecr=2053375275
522	215.823221463	10.0.2.5	10.0.2.4	TCP	66	50802 → 12344 [ACK] Seq=16 Ack=16 Win=64256 Len=0 TSval=438381726 TSecr=2053375276
523	215.823412371	10.0.2.4	10.0.2.5	TCP	66	12344 → 50802 [ACK] Seq=16 Ack=16 Win=65280 Len=0 TSval=2053375276 TSecr=438381726

גרסאות:

Version 1

קוד השרת:

מספר הפורט מתקבל כארגומנט, וגודל הבאפר הינו 1024 בתים.

ניתן לראות כי תחילה נעשה $bind$ לכתובת ip שהיא 0.0.0.0 ולמספר פורט (לדוגמא 12345)

המשמעות של זה היא שבעצם קולטים תעבורה מכל כרטיס רשת שנמצא במחשב כאשר הפורט הוא הפורט שמקושר לסוקט הנ"ל (שזה 12345 לדוגמא)

לאחר מכן ניתן לראות שמתבצעת הפקודה $s.listen(1)$, כאשר פקודה זו בעצם מסמנת שהסרבר מוכן לקבל פניות של לקוחות להתחברות. נשים לב שהעברנו לפונקציה פרמטר, זה בעצם מסמן את מספר הלקוחות שיכולים לחכות להתחברות מצד השרת עד שהשרת יסרב להתחברויות חדשות.

לאחר מכן רצים בלולאה אינסופית כאשר בתוכה מתבצע:

השרת מקבל חיבור על ידי הפקודה $accept$, נשים לב שמהפקודה $accept$ חוזרים 2 ערכים – ערך ראשון הוא ההתחברות החדשה, שזה בעצם סוקט ייעודי לתקשורת של השרת מול הלקוח הנ"ל, ובנוסף חוזרת הכתובת של הלקוח הנ"ל.

ואז מתבצעת הדפסה ולאחר מכן נכנסים לעוד לולאה אינסופית שבה מתבצע:

השרת מנסה לקרוא מהבאפר של הסוקט הייעודי מידע (בכל פעם צ'אנקים של 1024), אם הוא לא הצליח לקרוא הוא יוצא מהלולאה על מנת לקבל חיבור מלקוח חדש, אחרת הוא הצליח והוא שולח ללקוח את מה שהוא קיבל באותיות גדולות.

קוד הלקוח:

הלקוח תחילה מקבל כארגומנט את הפורט שבו השרת מאזין ואת כתובת הip של השרת, לאחר מכן הוא יוצר סוקט tcp חדש, ומתחבר לשרת לפי מה שהוא קיבל כארגומנט.

לאחר מכן הוא שולח לשרת את ההודעה, ואז הוא מחכה לתשובה מהשרת על ידי הפקודה $recv$ (קורא בצ'אנקים של 1024 בתים לפי מה שהוגדר בקוד), ואז הוא סוגר את הסוקט ובכך מסיים את התקשורת מהצד שלו ואז הוא מדפיס את המידע שהשרת החזיר לו.

tcp && tcp.port == 12345

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.4	10.0.2.15	TCP	74	48272 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2862273862 TSecr=0 WS=128
2	0.000037199	10.0.2.15	10.0.2.4	TCP	74	12345 → 48272 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=1147971160 TSecr=2862273862 WS=128
3	0.000523029	10.0.2.4	10.0.2.15	TCP	66	48272 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2862273863 TSecr=1147971160
4	0.001732164	10.0.2.4	10.0.2.15	TCP	79	48272 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=13 TSval=2862273863 TSecr=1147971160
5	0.001772528	10.0.2.15	10.0.2.4	TCP	66	12345 → 48272 [ACK] Seq=1 Ack=14 Win=65152 Len=0 TSval=1147971162 TSecr=2862273863
6	0.001936675	10.0.2.15	10.0.2.4	TCP	79	12345 → 48272 [PSH, ACK] Seq=1 Ack=14 Win=65152 Len=13 TSval=1147971162 TSecr=2862273863
7	0.002848580	10.0.2.4	10.0.2.15	TCP	66	48272 → 12345 [ACK] Seq=14 Ack=14 Win=64256 Len=0 TSval=2862273865 TSecr=1147971162
8	0.002848828	10.0.2.4	10.0.2.15	TCP	66	48272 → 12345 [FIN, ACK] Seq=14 Ack=14 Win=64256 Len=0 TSval=2862273865 TSecr=1147971162
9	0.003108945	10.0.2.15	10.0.2.4	TCP	66	12345 → 48272 [FIN, ACK] Seq=14 Ack=15 Win=65152 Len=0 TSval=1147971163 TSecr=2862273865
10	0.003637913	10.0.2.4	10.0.2.15	TCP	66	48272 → 12345 [ACK] Seq=15 Ack=15 Win=64256 Len=0 TSval=2862273866 TSecr=1147971163

תחילה ניתן לראות כי מתקיימת הקמת התקשורת בין השרת ללקוח. הלקוח שולח לשרת חבילה עם הדגל של SYN ואז השרת שולח ללקוח חבילה שהוא קיבל את המידע שנשלח, גם עם דגל SYN, ואז הלקוח שולח חבילה בחזרה לשרת שהוא קיבל את המידע וברגע זה התקשורת הוקמה והיא מסונכרנת.

לאחר מכן ניתן לראות שהלקוח שולח חבילה עם שכבת האפליקציה לשרת המכילה הודעה באורך של 13 בתים (שזוהי ההודעה של $Hello, World!$)

ואז השרת שולח חבילה ללקוח שהוא קיבל עד 14 בתים (כולל) כי הוא קיבל בעצם בית אחד של ה SYN הראשוני ולאחר מכן עוד 13 בתים של ההודעה.

לאחר מכן ניתן לראות שהשרת שולח ללקוח חבילה עם שכבת האפליקציה שמכילה הודעה באורך של 13 בתים (שזוהי ההודעה שקיבל מהלקוח רק באותיות גדולות), ואז לאחר מכן הלקוח בעצם שולח חבילה שהוא קיבל את ההודעה.

כעת מתרחשת סגירת החיבור לפי מנגנון ה $four\ way\ handshake$

הלקוח שולח חבילה לשרת עם הדגל של FIN , בעצם הדגל הזה מסמן שהלקוח רוצה לסיים את התקשורת עם השרת ושאין לו עוד מה לשלוח. לאחר מכן השרת מחזיר הודעה ללקוח שיש בה 2 דברים רלוונטיים – דבר ראשון שהוא קיבל את הודעת הFIN שהלקוח שלח, ודבר שני הוא גם שדגל הFIN דולק, כלומר השרת מוכן גם לסיים את התקשורת.

נשים לב שבעצם התאחדו פה 2 פעולות – גם ה ACK על ה FIN וגם נשלח FIN מצד השרת, וזה מפני שברגע של קבלת הFIN מאת הלקוח השרת ידע שגם לו אין עוד מה לשלוח והוא מבחינתו גם יכול לסגור את החיבור.

ואז לאחר מכן הלקוח שולח הודעה לשרת שהוא קיבל את החבילה שמכילה את הדגל FIN ובכך התקשורת מסתיימת בין השניים.

<u>Version 2</u>

קוד השרת:

הקוד דומה מאוד לקוד בגרסא הקודמת רק נשים לב שכאן גודל הבאפר המוגדר הוא 5 בניגוד למקודם שהוא היה 1024.

כלומר כעת השרת קורא מהסוקט שנפתח עם התקשורת של לקוח ספציפי בכל פעם 5 בתים.

קוד הלקוח:

גם כאן הקוד מאוד דומה לקוד בגרסא הקודמת, אך נשים לב שכעת ההודעה שנשלחת הינה בגודל של 20 בתים, כאשר מקודם ההודעה שנשלחה הייתה בגודל של 13 בתים.

tcp && tcp.port == 12345

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.4	10.0.2.15	TCP	74	55556 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2865210778 TSecr=0 WS=128
2	0.000076192	10.0.2.15	10.0.2.4	TCP	74	12345 → 55556 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=1150908071 TSecr=2865210778 WS=128
3	0.000735329	10.0.2.4	10.0.2.15	TCP	66	55556 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2865210779 TSecr=1150908071
4	0.001580756	10.0.2.4	10.0.2.15	TCP	86	55556 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=20 TSval=2865210780 TSecr=1150908071
5	0.001608424	10.0.2.15	10.0.2.4	TCP	66	12345 → 55556 [ACK] Seq=1 Ack=21 Win=65152 Len=0 TSval=1150908072 TSecr=2865210780
6	0.001782338	10.0.2.15	10.0.2.4	TCP	71	12345 → 55556 [PSH, ACK] Seq=1 Ack=21 Win=65152 Len=5 TSval=1150908072 TSecr=2865210780
7	0.002162047	10.0.2.4	10.0.2.15	TCP	66	55556 → 12345 [ACK] Seq=21 Ack=6 Win=64256 Len=0 TSval=2865210780 TSecr=1150908072
8	0.002177558	10.0.2.15	10.0.2.4	TCP	81	12345 → 55556 [PSH, ACK] Seq=6 Ack=21 Win=65152 Len=15 TSval=1150908073 TSecr=2865210780
9	0.002474401	10.0.2.4	10.0.2.15	TCP	66	55556 → 12345 [ACK] Seq=21 Ack=21 Win=64256 Len=0 TSval=2865210781 TSecr=1150908073
10	0.003729447	10.0.2.4	10.0.2.15	TCP	66	55556 → 12345 [FIN, ACK] Seq=21 Ack=21 Win=64256 Len=0 TSval=2865210782 TSecr=1150908073
11	0.003929257	10.0.2.15	10.0.2.4	TCP	66	12345 → 55556 [FIN, ACK] Seq=21 Ack=22 Win=65152 Len=0 TSval=1150908075 TSecr=2865210782
12	0.004313493	10.0.2.4	10.0.2.15	TCP	66	55556 → 12345 [ACK] Seq=22 Ack=22 Win=64256 Len=0 TSval=2865210783 TSecr=1150908075

גם כאן כמו מקודם נשים לב ששלושת החבילות הראשונות קשורות להקמת החיבור בין השרת ללקוח ולכן לא אפרט עליהן כי הן זהות למה שהוסבר בגרסא הקודמת.

כעת נשים לב שחבילה 4 נשלחת מהלקוח אל השרת ומכילה את שכבת האפליקציה עם הודעה באורך של 20 בתים, לאחר מכן חבילה 5 היא מהשרת אל הלקוח כך שהוא קיבל את החבילה בהצלחה.

כעת כאן נשים לב להבדל מהגרסא הקודמת, כעת באפליקציה השרת בעצם קורא מהבאפר 5 בתים (כי ככה הוגדר בקוד) למרות שיש עוד בתים לקריאה בבאפר, ואז הוא שולח את הבתים הנ"ל שנקראו (רק באותיות גדולות) בחזרה אל הלקוח ולכן ניתן לראות שבחבילה 6 שהשרת שלח יש את שכבת האפליקציה עם הודעה באורך של 5 בתים.

נשים לב שההודעה נשלחת ישר מפני ש TCP מזהה שאין חבילות שמחכות לACK ולכן כאשר נכנס מידע לבאפר הוא נשלח ישר.

לאחר מכן בחבילה 7 הלקוח שולח חבילה שבה הוא מאשר את קבלת ההודעה שהשרת שלח.

נשים לב שבחבילה 8 ניתן לראות כי השרת שולח בעצם 15 בתים !!! (ולא 5 בתים כמו מקודם) למרות שבעצם בקוד שרשום בעצם מבצעים $send$ להודעה באורך של 5 בתים! הסיבה לכך היא שככל הנראה TCP בחר לא לשלוח את ההודעות באותו הרגע שנכנסו לבאפר השליחה אלא חיכה להצטברות ורק אז שלח.

ההבדל ממקודם שההודעה ישר נשלחה כנראה נובע מהסיבה שהחבילות נכנסו לבאפר השליחה ועדיין לא התקבל ack על החבילה הראשונה שנשלחה ובנוסף גודל המידע שהיה בבאפר גם קטן מהMSS ולכן הוא לא נשלח ישר.

לאחר מכן ניתן לראות שהתקשורת נסגרת כפי שהראנו בגרסא הקודמת.

<u>Version 3</u>

קוד שרת:

קוד השרת דומה לגרסה הראשונה רק ההבדל היחידי הוא שכעת השרת שולח ללקוח בחזרה את ההודעה שקיבל ממנו באותיות גדולות כפול 1000

קוד לקוח:

קוד הלקוח דוגמא לגרסה הראשונה רק ההבדל הוא שכעת הלקוח קורא פעמיים מהבאפר ולא פעם אחת באמצעות הפקודה $recv$ ומדפיס לאחר כל קבלה של מידע מהבאפר.

tcp && tcp.port == 12345

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.4	10.0.2.15	TCP	74	52460 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2869510663 TSecr=0 WS=128
2	0.000061501	10.0.2.15	10.0.2.4	TCP	74	12345 → 52460 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=1155207927 TSecr=2869510663 WS=128
3	0.000623423	10.0.2.4	10.0.2.15	TCP	66	52460 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2869510663 TSecr=1155207927
4	0.000623558	10.0.2.4	10.0.2.15	TCP	79	52460 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=13 TSval=2869510663 TSecr=1155207927
5	0.000693748	10.0.2.15	10.0.2.4	TCP	66	12345 → 52460 [ACK] Seq=1 Ack=14 Win=65152 Len=0 TSval=1155207928 TSecr=2869510663
6	0.000972860	10.0.2.15	10.0.2.4	TCP	7306	12345 → 52460 [PSH, ACK] Seq=1 Ack=14 Win=65152 Len=7240 TSval=1155207928 TSecr=2869510663
7	0.001071065	10.0.2.15	10.0.2.4	TCP	5826	12345 → 52460 [PSH, ACK] Seq=7241 Ack=14 Win=65152 Len=5760 TSval=1155207928 TSecr=2869510663
8	0.011173473	10.0.2.4	10.0.2.15	TCP	66	52460 → 12345 [ACK] Seq=14 Ack=7241 Win=60672 Len=0 TSval=2869510674 TSecr=1155207928
9	0.011173840	10.0.2.4	10.0.2.15	TCP	66	52460 → 12345 [ACK] Seq=14 Ack=13001 Win=56832 Len=0 TSval=2869510674 TSecr=1155207928
10	0.011732540	10.0.2.4	10.0.2.15	TCP	66	52460 → 12345 [RST, ACK] Seq=14 Ack=13001 Win=64128 Len=0 TSval=2869510674 TSecr=1155207928

כמו בגרסאות הקודמות שלושת החבילות הראשונות הן להקמת התקשורת בין הלקוח לשרת ולכן לא אפרט עליהן עוד הפעם כי הן זהות לחלוטין.

ניתן לראות שבחבילה 4 הלקוח שולח לשרת הודעה באורך של 13 בתים (שהיא $Hello, World!$) כלומר חבילה זו כוללת את שכבת האפליקציה, ולאחר מכן בחבילה 5 ניתן לראות שהשרת שולח חבילה בחזרה ללקוח שהוא קיבל בהצלחה.

לאחר מכן בחבילה 6 ניתן לראות כי השרת שולח ללקוח הודעה באורך של 7240 בתים (בגלל הMSS), כלומר חבילה זו גם כוללת את שכבת האפליקציה, ולאחר מכן השרת שולח עוד הפעם חבילה 7 שגם מכילה את שכבת האפליקציה עם הודעה באורך של 5760 בתים (את שאר הבתים שנותרו לשליחה)

לאחר מכן בחבילות 8 ו9 אלו חבילות שבהן הלקוח שולח שהוא קיבל את המידע שהשרת שלח לו (בהתאמה).

כעת נביט בחבילה 10, נשים לב שנשלחה החבילה ביחד עם הדגל RST. נצטרך להביט בקוד של הלקוח כעת כדי להבין למה זה קרה.

נשים לב שהלקוח קרא פעמיים 1024 בתים, כלומר בסך הכל קרא 2048 בתים, כלומר יש עוד מידע בבאפר שעוד לא נקרא בשכבת האפליקציה, והלקוח גם לא יקרא אותו ולכן נשלחת חבילה עם הדגל של RST.

<u>Version 4</u>

קוד שרת:

בדומה לגרסה 1, רק שכעת לפני שהשרת מבצע את הפקודה $conn.recv()$ כלומר לפני שהאפליקציה של השרת מנסה לקרוא מידע מהבאפר של החיבור עם הלקוח, האפליקציה של השרת נכנסת למצב שינה במשך 5 שניות.

קוד לקוח:

בדומה לגרסה 1 בהתחלה, רק שכעת הלקוח שולח במשך 4 פעמים את ההודעה "$Hello, World!$" כפול 10, כלומר הלקוח שולח הודעה לשרת בגודל של 130 בתים במשך 4 פעמים, לאחר מכן הלקוח מבצע את הפקודה $conn.recv()$ ואז סוגר את החיבור, ואז מדפיס את מה שקיבל מהשרת.

tcp && tcp.port == 12345

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.4	10.0.2.15	TCP	74	33472 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2872319973 TSecr=0 WS=128
2	0.000040466	10.0.2.15	10.0.2.4	TCP	74	12345 → 33472 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=1158017216 TSecr=2872319973 WS=128
3	0.000303746	10.0.2.4	10.0.2.15	TCP	66	33472 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2872319974 TSecr=1158017216
4	0.000561130	10.0.2.4	10.0.2.15	TCP	196	33472 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=130 TSval=2872319974 TSecr=1158017216
5	0.000604762	10.0.2.15	10.0.2.4	TCP	66	12345 → 33472 [ACK] Seq=1 Ack=131 Win=65152 Len=0 TSval=1158017216 TSecr=2872319974
6	0.000858964	10.0.2.4	10.0.2.15	TCP	456	33472 → 12345 [PSH, ACK] Seq=131 Ack=1 Win=64256 Len=390 TSval=2872319974 TSecr=1158017216
7	0.000872844	10.0.2.15	10.0.2.4	TCP	66	12345 → 33472 [ACK] Seq=1 Ack=521 Win=64768 Len=0 TSval=1158017217 TSecr=2872319974
8	5.005957672	10.0.2.15	10.0.2.4	TCP	586	12345 → 33472 [PSH, ACK] Seq=1 Ack=521 Win=64768 Len=520 TSval=1158022222 TSecr=2872319974
9	5.006696928	10.0.2.4	10.0.2.15	TCP	66	33472 → 12345 [ACK] Seq=521 Ack=521 Win=64128 Len=0 TSval=2872324980 TSecr=1158022222
10	5.006949569	10.0.2.4	10.0.2.15	TCP	66	33472 → 12345 [FIN, ACK] Seq=521 Ack=521 Win=64128 Len=0 TSval=2872324980 TSecr=1158022222
11	5.050557780	10.0.2.15	10.0.2.4	TCP	66	12345 → 33472 [ACK] Seq=521 Ack=522 Win=64768 Len=0 TSval=1158022266 TSecr=2872324980
12	10.011932977	10.0.2.15	10.0.2.4	TCP	66	12345 → 33472 [FIN, ACK] Seq=521 Ack=522 Win=64768 Len=0 TSval=1158027228 TSecr=2872324980
13	10.012643315	10.0.2.4	10.0.2.15	TCP	66	33472 → 12345 [ACK] Seq=522 Ack=522 Win=64128 Len=0 TSval=2872329986 TSecr=1158027228

כמו בגרסאות הקודמות שלושת החבילות הראשונות הן להקמת התקשורת בין הלקוח לשרת ולכן לא אפרט עליהן עוד הפעם כי הן זהות לחלוטין.

נשים לב לחבילה 4, שבה הלקוח שולח חבילה לשרת שמכילה את שכבת האפליקציה, עם הודעה באורך של 130 בתים, נשים לב שההודעה הזו נשלחת ישר כשהלקוח מבצע את הפקודה $send$ מפני שאין חבילות שמחכות ל ACK ולכן TCP שולח ישר את החבילה. לאחר מכן בחבילה 5 השרת שולח חבילה ללקוח שהוא מאשר את קבלת המידע שהלקוח שלח לו בהצלחה.

בחבילה 6 ניתן לראות כי הלקוח שולח לשרת חבילה המכילה את שכבת האפליקציה ביחד עם הודעה באורך של 390 בתים (שזה 130 כפול 3) שזה בעצם שלושת הכתיבות לבאפר שהלקוח עשה לאחר הכתיבה הראשונה. למרות שהמידע נכתב בפעימות שונות אל הבאפר הוא לא נשלח ישר מפני שככל הנראה לא התקבל ACK על החבילה הקודמת, ובנוסף גודל המידע שצריך להישלח קטן מגודל הMSS ולכן עדיין לא נשלח.

ובחבילה 7 ניתן לראות שאכן השרת מאשר את קבלת החבילה שהלקוח שלח לו בהצלחה.

נשים לב שלאחר שהשרת ביצע $accept$ אל הלקוח, ולאחר כניסה ללולאה השרת נכנס לשינה של 5 שניות, אך נשים לב שהשינה הזו מתרחשת באפליקציה של השרת, אבל מאחורי הקלעים TCP עדיין ממשיך לתפקד כרגיל, ולכן ניתן לראות שבעמודה של הTIME

בעצם כל החבילות שתיארנו עד כה נשלחו כרגיל והמידע שהלקוח שלח לשרת בעצם מחכה לו בבאפר של הסוקט הייעודי בתקשורת ביניהם, אז בעצם השינה של השרת רק משפיעה על הקריאה של המידע שהאפליקציה מבצעת בעצם.

כעת אכן ניתן לראות לאחר שהשרת מתעורר כי הוא שולח את חבילה 8 ללקוח עם מידע בשכבת האפליקציה של הודעה באורך של 520 (שזה 130 כפול 4) שזה בעצם כל המידע שחיכה לשרת בבאפר (שהספיק להגיע עד שהשרת התעורר), ולאחר מכן הלקוח שולח חבילה 9 לשרת שבה הוא מאשר את קבלת המידע שהשרת שלח לו.

נשים לב שכעת מתקיים מנגון ה $four\ way\ handshake$ לסגירת התקשורת בין הלקוח לשרת עם הבדל מהגרסאות הקודמות שאפרט עליו כעת.

בחבילה 10 ניתן לראות שהלקוח יוזם את סגירת החיבור על ידי שליחת חבילה עם הדגל FIN ואז השרת שולח חבילה 11 שבה הוא מאשר את קבלת הFIN שהלקוח שלח לו.

נשים לב שבקוד של השרת לאחר שליחת ההודעה ללקוח הוא נכנס לתרדמת נוספת של 5 שניות, ורק כאשר הוא מתעורר ומגלה שאין לו עוד מידע לשלוח אל הלקוח, אז הוא שולח חבילה עם דגל FIN (חבילה 12) אל הלקוח, ובחבילה 13 הלקוח מאשר את קבלת הFIN שהשרת שלח לו ובכך התקשורת בין הלקוח לשרת נסגרת.

נשים לב שבשונה מגרסה 1, שבה ניתן לראות שהתקשורת נסגרת על ידי 3 שלבים, מפני שהשרת מאחד את הACK על קבלת הFIN שהלקוח שלח לו, ביחד עם הFIN שהשרת שולח אל הלקוח ביחד, מפני שבגרסה 1, השרת לא היה בתרדמת והוא ידע שאין לו עוד מידע לשלוח אל הלקוח ולכן מבחינתו הוא גם היה יכול להגיד ללקוח שמבחינתו אפשר לסגור את התקשורת ביניהם, אך פה זה לא המצב מפני שכאשר הלקוח שלח את הFIN אזי הלקוח בעצם הודיע שלו אין מה עוד לשלוח אבל זה לא אומר בהכרח שלשרת אין מה לשלוח עוד, ומפני שהאפליקציה של השרת הייתה בתרדמת של 5 שניות אז לא התאחדה החבילה של הACK על הFIN ביחד עם החבילה של הFIN והן נשלחו בנפרד.

חלק ב'

נדגים כעת הרצות של פקודות שונות של הלקוח. אנחנו מריצים את השרת ואת הלקוח על אותה המכונה כפי שנרשם שניתן לעשות בתרגיל

אם אתם מריצים את השרת על המחשב שלכם, אפשר במקום כתובת הIP המקומית (127.0.0.1) לכתוב localhost.

בנוסף הרצנו את השרת עם הארגומנט של מספר פורט "12345" (במקרים שהשתמשנו בפורט אחר הדבר צוין) ובנוסף על פי הוראות התרגיל אנחנו מניחים כי השרת מורץ בתוך תיקייה שבה יש את תיקיית $files$.

- הרצה ללא קלט (מעבר לאינדקס) על ידי הכתובת $localost$: 12345

tcp && tcp.port == 12345

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	::1	::1	TCP	94	54250 → 12345 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=936962444 TSecr=
2	0.000007570	::1	::1	TCP	74	12345 → 54250 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3	0.000052580	127.0.0.1	127.0.0.1	TCP	74	42126 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=3551127684 TSecr
4	0.000060434	127.0.0.1	127.0.0.1	TCP	74	12345 → 42126 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=35511
5	0.000068182	127.0.0.1	127.0.0.1	TCP	66	42126 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3551127684 TSecr=3551127684
6	0.000157527	127.0.0.1	127.0.0.1	HTTP	764	GET / HTTP/1.1
7	0.000163200	127.0.0.1	127.0.0.1	TCP	66	12345 → 42126 [ACK] Seq=1 Ack=699 Win=64896 Len=0 TSval=3551127684 TSecr=3551127684
8	0.000395209	127.0.0.1	127.0.0.1	HTTP	299	HTTP/1.1 200 OK
9	0.000407237	127.0.0.1	127.0.0.1	TCP	66	42126 → 12345 [ACK] Seq=699 Ack=234 Win=65408 Len=0 TSval=3551127684 TSecr=3551127684
10	0.265811803	127.0.0.1	127.0.0.1	HTTP	639	GET /favicon.ico HTTP/1.1
11	0.266095165	127.0.0.1	127.0.0.1	HTTP	3258	HTTP/1.1 200 OK
12	0.266106424	127.0.0.1	127.0.0.1	TCP	66	42126 → 12345 [ACK] Seq=1272 Ack=3426 Win=63360 Len=0 TSval=3551127950 TSecr=355112795
13	1.267054208	127.0.0.1	127.0.0.1	TCP	66	12345 → 42126 [FIN, ACK] Seq=3426 Ack=1272 Win=65536 Len=0 TSval=3551128951 TSecr=3551
14	1.312381486	127.0.0.1	127.0.0.1	TCP	66	42126 → 12345 [ACK] Seq=1272 Ack=3427 Win=64896 Len=0 TSval=3551128996 TSecr=355112895
15	6.497496542	127.0.0.1	127.0.0.1	TCP	66	42126 → 12345 [FIN, ACK] Seq=1272 Ack=3427 Win=65536 Len=0 TSval=3551134181 TSecr=3551
16	6.497512168	127.0.0.1	127.0.0.1	TCP	66	12345 → 42126 [ACK] Seq=3427 Ack=1273 Win=65536 Len=0 TSval=3551134181 TSecr=355113418

בשורות 1,2 ניתן לראות שהדפדפן מנסה להקים התחברות דרך הפורט 54250 , ההתחברות נכשלת, ניתן לראות לפי דגל ה RST שהשרת שלח בחזרה לדפדפן.
לאחר מכן בשורות 3-5 ניתן לראות כי מתקיים חיבור מוצלח עם הלקוח שמספר הפורט שלו הוא 42126 .
הלקוח שולח כעת חבילה עם בקשת $HTTP$ (שורה 6) כאשר ההודעה בעצם מכילה את הבקשה ל / (אינדקס)
כתוצאה מכך השרת שולח ACK שאכן החבילה התקבלה בהצלחה (חבילה 7)
לאחר מכן ניתן לראות כי נשלחת חבילה 8 מהשרת אל הלקוח הכתובה לפי פרוטוקול $HTTP$ המכיל בעצם את קובץ ה $index.html$ ולאחר מכן בחבילה הבאה (מספר 9) הלקוח מאשר את קבלת החבילה.
בחבילה 10 הלקוח מבקש את favicon.ico ולאחר מכן ניתן לראות כי בחבילה 11 השרת מחזיר ללקוח 2 דברים:
גם ACK שהוא קיבל את החבילה של הלקוח שמכילה את הבקשה הנ"ל ובנוסף הוא שלח את המידע שהלקוח ביקש, לאחר מכן בחבילה 12 ניתן לראות כי הלקוח מאשר את קבלת החבילה.
לאחר מכן בחבילות 13-16 מתקיים מנגנון ה $four\ way\ handshake$ לסיום התקשורת בין השרת ללקוח כאשר השרת הוא זה שיוזם את סיום התקשורת.
נשים לב שפה היה ניסיון לפתיחת חיבור אחד (בהתחלה) אשר נכשל, ולאחר מכן כל התקשורת התבצעה על פני חיבור יחידי, וכאמור בחיבור הזה נשלחה יותר מבקשה אחת מהלקוח.

- קובץ לא קיים על ידי הכתובת localhost:12345/a/7.jpg

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	127.0.0.1	127.0.0.1	TCP	74	54272 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=3552081427 TSecr
2	0.000010	127.0.0.1	127.0.0.1	TCP	74	12345 → 54272 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=35520
3	0.000018	127.0.0.1	127.0.0.1	TCP	66	54272 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3552081427 TSecr=3552081427
4	0.000146	127.0.0.1	127.0.0.1	HTTP	757	GET /a/7.jpg HTTP/1.1
5	0.000154	127.0.0.1	127.0.0.1	TCP	66	12345 → 54272 [ACK] Seq=1 Ack=692 Win=64896 Len=0 TSval=3552081427 TSecr=3552081427
6	0.000398	127.0.0.1	127.0.0.1	TCP	112	12345 → 54272 [PSH, ACK] Seq=1 Ack=692 Win=65536 Len=46 TSval=3552081427 TSecr=3552081
7	0.000413	127.0.0.1	127.0.0.1	TCP	66	54272 → 12345 [ACK] Seq=692 Ack=47 Win=65536 Len=0 TSval=3552081427 TSecr=3552081427
8	0.000429	127.0.0.1	127.0.0.1	HTTP	66	HTTP/1.1 404 Not Found
9	0.001342	127.0.0.1	127.0.0.1	TCP	66	54272 → 12345 [FIN, ACK] Seq=692 Ack=48 Win=65536 Len=0 TSval=3552081428 TSecr=3552081
10	0.001353	127.0.0.1	127.0.0.1	TCP	66	12345 → 54272 [ACK] Seq=48 Ack=693 Win=65536 Len=0 TSval=3552081428 TSecr=3552081428

כאן אנחנו נקבל שיש חיבור אחד בלבד (ניתן לראות את הקמת החיבור בחבילות 1-3)
לאחר מכן בחבילה 4 הלקוח בעצם מבקש את הקובץ (שלא קיים כאמור), ובחבילה 5 השרת מאשר את קבלת הבקשה של הלקוח, וניתן לראות כי בחבילה 6 בעצם השרת שולח ללקוח חבילה המכילה מידע בשכבת האפליקציה (עם הודעה באורך של 46 תווים) כאשר המידע הוא:

TCP segment data (46 bytes)

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ........ ......E.
0010  00 62 81 d3 40 00 40 06  ba c0 7f 00 00 01 7f 00   .b..@.@. ........
0020  00 01 30 39 d4 00 1a 25  21 34 86 e5 e1 8b 80 18   ..09...% !4......
0030  02 00 fe 56 00 00 01 01  08 0a d3 b8 76 13 d3 b8   ...V.... ....v...
0040  76 13 48 54 54 50 2f 31  2e 31 20 34 30 34 20 4e   v.HTTP/1 .1 404 N
0050  6f 74 20 46 6f 75 6e 64  20 0d 0a 43 6f 6e 6e 65   ot Found  ..Conne
0060  63 74 69 6f 6e 3a 20 63  6c 6f 73 65 0d 0a 0d 0a   ction: c lose....
```

כלומר השרת מודיע על סגירת החיבור, וכאמור בחבילה הבאה (חבילה מספר 7) הלקוח מאשר את קבלת החבילה (ניתן לראות לפי מספר ה ACK)
כעת נביט בחבילה 8, היא כאמור חבילה המכילה את שכבת האפליקציה (HTTP) ובעצם גם מכילה את הדגל FIN (מוסיף צילום מ WIRESHARK):

```
Flags: 0x011 (FIN, ACK)
```

ואז בעצם בחבילה 9 מתקיים כי הלקוח מחזיר ACK על מה שנשלח מהשרת ביחד עם דגל FIN ובחבילה 10 השרת מאשר את קבלת הFIN מאת הלקוח ובכך התקשורת מסתיימת בין השניים.

- הרצה של $redirect$ על ידי הכתובת localhost:12346/redirect

	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	::1	::1	TCP	94	34244 → 12346 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=2714202630 TSecr=0 WS=128
2	0.000005822	::1	::1	TCP	74	12346 → 34244 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3	0.000058175	127.0.0.1	127.0.0.1	TCP	74	60826 → 12346 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2910178746 TSecr=0 WS=128
4	0.000066322	127.0.0.1	127.0.0.1	TCP	74	12346 → 60826 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2910178746 TSecr
5	0.000072671	127.0.0.1	127.0.0.1	TCP	66	60826 → 12346 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2910178747 TSecr=2910178746
6	0.000226887	127.0.0.1	127.0.0.1	HTTP	720	GET /redirect HTTP/1.1
7	0.000231809	127.0.0.1	127.0.0.1	TCP	66	12346 → 60826 [ACK] Seq=1 Ack=655 Win=64896 Len=0 TSval=2910178747 TSecr=2910178747
8	0.000467360	127.0.0.1	127.0.0.1	TCP	144	12346 → 60826 [PSH, ACK] Seq=1 Ack=655 Win=65536 Len=78 TSval=2910178747 TSecr=2910178747 [TCP se
9	0.000474804	127.0.0.1	127.0.0.1	HTTP	66	HTTP/1.1 301 Moved Permanently
10	0.000488706	127.0.0.1	127.0.0.1	TCP	66	60826 → 12346 [ACK] Seq=655 Ack=79 Win=65536 Len=0 TSval=2910178747 TSecr=2910178747
11	0.000543914	127.0.0.1	127.0.0.1	TCP	66	60826 → 12346 [FIN, ACK] Seq=655 Ack=80 Win=65536 Len=0 TSval=2910178747 TSecr=2910178747
12	0.000545691	127.0.0.1	127.0.0.1	TCP	66	12346 → 60826 [ACK] Seq=80 Ack=656 Win=65536 Len=0 TSval=2910178747 TSecr=2910178747
13	0.004803798	::1	::1	TCP	94	34246 → 12346 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=2714202635 TSecr=0 WS=128
14	0.004809375	::1	::1	TCP	74	12346 → 34246 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
15	0.004847004	127.0.0.1	127.0.0.1	TCP	74	60828 → 12346 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2910178751 TSecr=0 WS=128
16	0.004852776	127.0.0.1	127.0.0.1	TCP	74	12346 → 60828 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2910178751 TSecr
17	0.004857617	127.0.0.1	127.0.0.1	TCP	66	60828 → 12346 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2910178751 TSecr=2910178751
18	0.004997024	127.0.0.1	127.0.0.1	HTTP	723	GET /result.html HTTP/1.1
19	0.005001241	127.0.0.1	127.0.0.1	TCP	66	12346 → 60828 [ACK] Seq=1 Ack=658 Win=64896 Len=0 TSval=2910178751 TSecr=2910178751
20	0.005230784	127.0.0.1	127.0.0.1	HTTP	253	HTTP/1.1 200 OK
21	0.005241551	127.0.0.1	127.0.0.1	TCP	66	60828 → 12346 [ACK] Seq=658 Ack=188 Win=65408 Len=0 TSval=2910178752 TSecr=2910178752
22	0.080048628	127.0.0.1	127.0.0.1	HTTP	650	GET /favicon.ico HTTP/1.1
23	0.080063088	127.0.0.1	127.0.0.1	TCP	66	12346 → 60828 [ACK] Seq=188 Ack=1242 Win=65024 Len=0 TSval=2910178826 TSecr=2910178826
24	0.080245710	127.0.0.1	127.0.0.1	HTTP	3258	HTTP/1.1 200 OK
25	0.080269843	127.0.0.1	127.0.0.1	TCP	66	60828 → 12346 [ACK] Seq=1242 Ack=3380 Win=62464 Len=0 TSval=2910178827 TSecr=2910178827
26	1.088118207	127.0.0.1	127.0.0.1	TCP	66	12346 → 60828 [FIN, ACK] Seq=3380 Ack=1242 Win=65536 Len=0 TSval=2910179835 TSecr=2910178827
27	1.135144669	127.0.0.1	127.0.0.1	TCP	66	60828 → 12346 [ACK] Seq=1242 Ack=3381 Win=64896 Len=0 TSval=2910179882 TSecr=2910179835
28	4.324350688	127.0.0.1	127.0.0.1	TCP	66	60828 → 12346 [FIN, ACK] Seq=1242 Ack=3381 Win=65536 Len=0 TSval=2910183071 TSecr=2910179835
29	4.324363445	127.0.0.1	127.0.0.1	TCP	66	12346 → 60828 [ACK] Seq=3381 Ack=1243 Win=65536 Len=0 TSval=2910183071 TSecr=2910183071

(השרת בפורט 12346)

בשורות 1-2 הלקוח מנסה להקים חיבור אל מול השרת , בכך שהוא (דגל SYN) הדבר לא צלח כאשר בשורה 2 השרת מחזיר RST והחיבור מתנתק.

לאחר מכן מפורט 60826 מוקם חיבור מוצלח מול השרת , לחיצת הידיים מתרחשת ב שורות 3-5 ,שלאחריה נשלח לשרת בקשת redirect. השרת מאשר את הבקשה , ושולח חזרה ללקוח בקשת Moved Permanently, בשלב זה כנדרש במשימה החיבור נסגר , הלקוח שולח לשרת הודעת FIN ,בשורת 13-14 הדפדפן מנסה להקים חיבור חדש (כדי לטפל בהודעת movedה שהתקבלה קודם לכן, חיבור זה נכשל , ובשורות 15-17 מוקם חיבור מוצלח דרך פורט 60828 ,שבסופו החיבור שולח בקשה ל result (הכתובת שהושמה בהודעת הmoved).השרת מאשר את הבקשה ובשורה 20 שולח את קובץ הhtml הדרוש , בנוסף ניתן לראות בשורה 22 בקשה נוספת מהלקוח , הפעם ל icon של העמוד ,אשר נשלח בשורה 24.

ולבסוף בשורות 26-29 השרת ראשית שולח הודעת סיום התקשרות FIN אותה החיבור מאשר ומחזיר הודעת סיום התקשרות משלו.

סה"כ היו שני חיבורים מוצלחים (בכול רגע נתון היה אחד פעיל),הראשון שלח את הודעת ה redirect וקיבל על כך תגובה מהשרת , השני שלח את הודעת ה result וקיבל את הדרוש מהשרת (קובץ html ו icon).

- הרצה של הכתובת localhost:12347/a/2.jpg

	Time	Source	Destination	Protocol	Length	Info
1	0.000000	::1	::1	TCP	94	59668 → 12347 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=67612243 TSecr=0 WS=128
2	0.000010	::1	::1	TCP	74	12347 → 59668 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3	0.000064	127.0.0.1	127.0.0.1	TCP	74	47122 → 12347 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2423929579 TSecr=0 WS=128
4	0.000073	127.0.0.1	127.0.0.1	TCP	74	12347 → 47122 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2423929579 TSecr=2423929579 WS=128
5	0.000081	127.0.0.1	127.0.0.1	TCP	66	47122 → 12347 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2423929579 TSecr=2423929579
6	0.000298	127.0.0.1	127.0.0.1	HTTP	740	GET /a/2.jpg HTTP/1.1
7	0.000303	127.0.0.1	127.0.0.1	TCP	66	12347 → 47122 [ACK] Seq=1 Ack=675 Win=64896 Len=0 TSval=2423929579 TSecr=2423929579
8	0.000774	127.0.0.1	127.0.0.1	TCP	32834	12347 → 47122 [ACK] Seq=1 Ack=675 Win=65536 Len=32768 TSval=2423929580 TSecr=2423929579 [TCP segment of a reassembled PDU]
9	0.000831	127.0.0.1	127.0.0.1	TCP	66	47122 → 12347 [ACK] Seq=675 Ack=32769 Win=48512 Len=0 TSval=2423929580 TSecr=2423929580
10	0.000842	127.0.0.1	127.0.0.1	TCP	32834	12347 → 47122 [PSH, ACK] Seq=32769 Ack=675 Win=65536 Len=32768 TSval=2423929580 TSecr=2423929579 [TCP segment of a reassembled PDU]
11	0.002032	127.0.0.1	127.0.0.1	TCP	66	47122 → 12347 [ACK] Seq=675 Ack=65537 Win=65536 Len=0 TSval=2423929581 TSecr=2423929580
12	0.002040	127.0.0.1	127.0.0.1	TCP	32834	12347 → 47122 [ACK] Seq=65537 Ack=675 Win=65536 Len=32768 TSval=2423929581 TSecr=2423929581 [TCP segment of a reassembled PDU]
13	0.002043	127.0.0.1	127.0.0.1	TCP	32834	[TCP Window Full] 12347 → 47122 [PSH, ACK] Seq=98305 Ack=675 Win=65536 Len=32768 TSval=2423929581 TSecr=2423929581 [TCP segment of a
14	0.002489	127.0.0.1	127.0.0.1	TCP	66	47122 → 12347 [ACK] Seq=675 Ack=131073 Win=65536 Len=0 TSval=2423929582 TSecr=2423929581
15	0.002496	127.0.0.1	127.0.0.1	TCP	32834	12347 → 47122 [ACK] Seq=131073 Ack=675 Win=65536 Len=32768 TSval=2423929582 TSecr=2423929582 [TCP segment of a reassembled PDU]
16	0.002499	127.0.0.1	127.0.0.1	TCP	32834	[TCP Window Full] 12347 → 47122 [PSH, ACK] Seq=163841 Ack=675 Win=65536 Len=32768 TSval=2423929582 TSecr=2423929582 [TCP segment of
17	0.002504	127.0.0.1	127.0.0.1	TCP	66	47122 → 12347 [ACK] Seq=675 Ack=163841 Win=196480 Len=0 TSval=2423929582 TSecr=2423929582
18	0.002507	127.0.0.1	127.0.0.1	HTTP	24276	HTTP/1.1 200 OK (JPEG JFIF image)
19	0.002510	127.0.0.1	127.0.0.1	TCP	66	47122 → 12347 [ACK] Seq=675 Ack=196609 Win=327552 Len=0 TSval=2423929582 TSecr=2423929582
20	0.002512	127.0.0.1	127.0.0.1	TCP	66	47122 → 12347 [ACK] Seq=675 Ack=220819 Win=458496 Len=0 TSval=2423929582 TSecr=2423929582
21	0.011651	::1	::1	TCP	94	59670 → 12347 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=67612255 TSecr=0 WS=128
22	0.011657	::1	::1	TCP	74	12347 → 59670 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
23	0.011698	127.0.0.1	127.0.0.1	TCP	74	47124 → 12347 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2423929591 TSecr=0 WS=128
24	0.011705	127.0.0.1	127.0.0.1	TCP	74	12347 → 47124 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2423929591 TSecr=2423929591 WS=128
25	0.011710	127.0.0.1	127.0.0.1	TCP	66	47124 → 12347 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2423929591 TSecr=2423929591
26	1.001365	127.0.0.1	127.0.0.1	TCP	66	12347 → 47122 [FIN, ACK] Seq=220819 Ack=675 Win=65536 Len=0 TSval=2423930580 TSecr=2423929582
27	1.043841	127.0.0.1	127.0.0.1	TCP	66	47122 → 12347 [ACK] Seq=675 Ack=220820 Win=458496 Len=0 TSval=2423930623 TSecr=2423930580
28	2.004190	127.0.0.1	127.0.0.1	TCP	66	12347 → 47124 [FIN, ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2423931583 TSecr=2423929591
29	2.006888	127.0.0.1	127.0.0.1	TCP	66	47124 → 12347 [ACK] Seq=1 Ack=2 Win=65536 Len=0 TSval=2423931586 TSecr=2423931583
30	3.045926	127.0.0.1	127.0.0.1	TCP	66	47122 → 12347 [FIN, ACK] Seq=675 Ack=220820 Win=458496 Len=0 TSval=2423932625 TSecr=2423930580
31	3.045934	127.0.0.1	127.0.0.1	TCP	66	12347 → 47122 [ACK] Seq=220820 Ack=676 Win=65536 Len=0 TSval=2423932625 TSecr=2423932625
32	3.045944	127.0.0.1	127.0.0.1	TCP	66	47124 → 12347 [FIN, ACK] Seq=1 Ack=2 Win=65536 Len=0 TSval=2423932625 TSecr=2423931583
33	3.045946	127.0.0.1	127.0.0.1	TCP	66	12347 → 47124 [ACK] Seq=2 Ack=2 Win=65536 Len=0 TSval=2423932625 TSecr=2423932625

(השרת בפורט 12347)

בשורות 1-2 הלקוח מנסה להקים חיבור אל מול השרת , בכך שהוא (ניתן לראות דגל SYN) ניתן לראות כי הדבר לא צלח כאשר בשורה 2 השרת מחזיר RST והחיבור מתנתק.

```
Hypertext Transfer Protocol
+ GET /a/2.jpg HTTP/1.1\r\n
  Host: localhost:12347\r\n
  Connection: keep-alive\r\n
  sec-ch-ua: "Not?A_Brand";v="8", "
  sec-ch-ua-mobile: ?0\r\n
  sec-ch-ua-platform: "Linux"\r\n
  Upgrade-Insecure-Requests: 1\r\n
  User-Agent: Mozilla/5.0 (X11; Li
  Accept: text/html,application/xh
  Sec-Fetch-Site: none\r\n
  Sec-Fetch-Mode: navigate\r\n
  Sec-Fetch-User: ?1\r\n
  Sec-Fetch-Dest: document\r\n
  Accept-Encoding: gzip, deflate, 
  Accept-Language: he-IL,he;q=0.9,
  \r\n
```

לאחר מכן ניתן לראות בשורות 3-5 לחיצת ידיים מוצלחת של לקוח הפונה מ פורט 47122 , לאחריה בשורה 6 הלקוח שולח בקשת HTTP , בה הוא מבקש את /a/2.jpg

התמונה הנמצאת בתיקייה a .

השרת מאשר בשורה 7 את הבקשה , ובשורות 8-18 ניתן לראות כי השרת שולח ללקוח חבילות המכילות את חלקי התמונה, (בנוסף ניתן לראות כי הלקוח מחזיר Ack ים על קבלת החבילות)

בשורה 18 ניתן לראות את הודעת הHTTP המתאימה לשליחת התמונה .

(ניתן להבחין בהודעות TCP Window Full בשורות 13 16 עליהן הלקוח מחזיר ACK , התבקשנו לא להרחיב בנושא ע"י המרצה).

בשורה 20 הלקוח שולח לשרת הודעה על אישור קבלת המידע.

נשים לב כי כול השלבים הללו התנהלו בחיבור בודד מול הלקוח שבפורט 47122 .

בשלב זה הדפדפן מנסה להקים חיבור חדש (שורות 21 22) שוב גם כאן לחיצת הידיים נכשלת , והחיבור נסגר.

ובשורות 23-25 ניתן לראות לחיצת ידיים מוצלחת מול כאשר החיבור מתבצע דרך פורט 47124 ,

לאחר מכן בשורות 26-27 לאחר שעבר פרק זמן(time out) השרת שולח הודעת FIN, סיום התקשרות ,הלקוח מאשר זאת בשורה 27 , עוד לפני שהלקוח שבפורט 47122 מחזיר לשרת הודעת סיום התקשרות השרת מנתק גם את החיבור שבפורט 47124 (שולח לו הודעת FIN אשר מאושרת על ידו)

בשורות 30-31 הלקוח שבפורט 47122 שולח הודעת סיום התקשרות ,שמאושרת ע"י השרת.

בשורות 32-33 הלקוח שבפורט 47124 שולח הודעת סיום התקשרות ,שמאושרת ע"י השרת.

סה"כ היו שני חיבורים מצלחים כאשר רק בראשון הועבר כול המידע ובשני לא הועבר כלל מידע .

- הרצה של ref על ידי הכתובת localhost:12347/a/b/ref.html

בהרצת ref העברו מספר רב במיוחד של חבילות , זאת כיוון שקובץ זה מכיל תמונות רבות אשר נשלחות בנפרד אל הלקוח , אסביר את האירועים המרכזים בתהליל ההתקשרות של הדפדפן אל מול השרת.

	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	::1	::1	TCP	94	59672 → 12347 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=70635599 TSecr=0 WS=128
2	0.000016689	::1	::1	TCP	74	12347 → 59672 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3	0.000075812	127.0.0.1	127.0.0.1	TCP	74	47126 → 12347 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2426952935 TSecr=0 WS=128
4	0.000082384	127.0.0.1	127.0.0.1	TCP	74	12347 → 47126 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2426952935 TSecr=2426952935 WS=128
5	0.000087626	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2426952935 TSecr=2426952935
6	0.000216592	127.0.0.1	127.0.0.1	HTTP	745	GET /a/b/ref.html HTTP/1.1
7	0.000221756	127.0.0.1	127.0.0.1	TCP	66	12347 → 47126 [ACK] Seq=1 Ack=680 Win=64896 Len=0 TSval=2426952935 TSecr=2426952935
8	0.001089372	127.0.0.1	127.0.0.1	HTTP	734	HTTP/1.1 200 OK
9	0.001134807	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=680 Ack=669 Win=64896 Len=0 TSval=2426952936 TSecr=2426952936
10	0.002378767	::1	::1	TCP	94	59676 → 12347 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=70635601 TSecr=0 WS=128
11	0.002382997	::1	::1	TCP	74	12347 → 59676 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
12	0.002413556	127.0.0.1	127.0.0.1	TCP	74	47128 → 12347 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2426952937 TSecr=0 WS=128
13	0.002418407	127.0.0.1	127.0.0.1	TCP	74	12347 → 47128 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2426952937 TSecr=2426952937 WS=128
14	0.002422639	127.0.0.1	127.0.0.1	TCP	66	47128 → 12347 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2426952937 TSecr=2426952937
15	0.025416830	127.0.0.1	127.0.0.1	HTTP	668	GET /a/1.jpg HTTP/1.1
16	0.025563557	127.0.0.1	127.0.0.1	TCP	66	12347 → 47126 [ACK] Seq=669 Ack=1282 Win=65024 Len=0 TSval=2426952960 TSecr=2426952960
17	0.026470873	127.0.0.1	127.0.0.1	TCP	32834	12347 → 47126 [ACK] Seq=669 Ack=1282 Win=65536 Len=32768 TSval=2426952961 TSecr=2426952960 [TCP segment of a reassembled PDU]
18	0.026523041	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1282 Ack=33437 Win=48512 Len=0 TSval=2426952961 TSecr=2426952961
19	0.026534804	127.0.0.1	127.0.0.1	TCP	32834	12347 → 47126 [PSH, ACK] Seq=33437 Ack=1282 Win=65536 Len=32768 TSval=2426952961 TSecr=2426952960 [TCP segment of a reassembled PDU]
20	0.026538411	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1282 Ack=66205 Win=15744 Len=0 TSval=2426952961 TSecr=2426952961
21	0.028662071	127.0.0.1	127.0.0.1	TCP	66	[TCP Window Update] 47126 → 12347 [ACK] Seq=1282 Ack=66205 Win=65536 Len=0 TSval=2426952963 TSecr=2426952961
22	0.028669900	127.0.0.1	127.0.0.1	TCP	32834	12347 → 47126 [ACK] Seq=66205 Ack=1282 Win=65536 Len=32768 TSval=2426952963 TSecr=2426952963 [TCP segment of a reassembled PDU]

חיבור מוצלח ראשון בשורות 3-5 ,שבסופו מוקם חיבור בפורט 47126 מחיבור זה יוצאת בקשה לקבלת קובץ ref, השרת מאשר ושולח לדפדפן את קובץ ה html (שורה 8) ,בשורות 12-14 מוקם חיבור מוצלח נוסף פורט 47126 , דרכו הלקוח מבקש את התמונה הראשונה מתוך קובץ ה ref : /a/1.jpg, כאן מתרחשת התנהגות דומה לזאת שראינו כאשר הלקוח ביקש תמונה בודדת ובשורה 49 ניתן לראות את שליחת התמונה (על פי פרוטוקול HTTP)ץ

47	0.028843513	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1282 Ack=492189 Wi
48	0.028845811	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1282 Ack=524957 Wi
49	0.028847998	127.0.0.1	127.0.0.1	HTTP	7769	HTTP/1.1 200 OK (JPEG JFIF image)
50	0.028849880	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1282 Ack=532660 Wi
51	0.028885315	127.0.0.1	127.0.0.1	HTTP	670	GET /a/b/1.jpg HTTP/1.1
52	0.028887389	127.0.0.1	127.0.0.1	TCP	66	12347 → 47128 [ACK] Seq=1 Ack=605 Win=6489
53	0.168754593	127.0.0.1	127.0.0.1	HTTP	668	GET /a/2.jpg HTTP/1.1
54	0.169590637	127.0.0.1	127.0.0.1	TCP	65549	12347 → 47126 [ACK] Seq=532660 Ack=1884 Wi
55	0.169665508	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1884 Ack=598143 Wi
56	0.169680320	127.0.0.1	127.0.0.1	TCP	65549	12347 → 47126 [ACK] Seq=598143 Ack=1884 Wi
57	0.169687066	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1884 Ack=663626 Wi
58	0.169701524	127.0.0.1	127.0.0.1	TCP	65549	12347 → 47126 [ACK] Seq=663626 Ack=1884 Wi
59	0.169707845	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1884 Ack=729109 Wi
60	0.169714526	127.0.0.1	127.0.0.1	HTTP	24435	HTTP/1.1 200 OK (JPEG JFIF image)
61	0.169717746	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1884 Ack=753478 Wi

בשורה 51 הדפדפן מחיבור 47128 מבקש את התמונה הבאה מהקובץ .

ובשורה 53 הדפדפן מחיבור 47126 מבקש את התמונה שלאחריה.

השרת שולח את התמונה שביקש הדפדפן מחיבור שבפורט 47126, וסיום השליחה נראה בשורה 60.

61	0.169717746	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [ACK] Seq=1884 Ack=753478 Win=2179968 Len=0 TSval=2426953104 TSecr=2426953104
62	0.171563165	::1	::1	TCP	94	59678 → 12347 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=70635770 TSecr=0 WS=128
63	0.171568094	::1	::1	TCP	74	12347 → 59678 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
64	0.171606054	127.0.0.1	127.0.0.1	TCP	74	47132 → 12347 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2426953106 TSecr=0 WS=128
65	0.171612146	127.0.0.1	127.0.0.1	TCP	74	12347 → 47132 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2426953106 TSecr=2426953106 WS=128
66	0.171616825	127.0.0.1	127.0.0.1	TCP	66	47132 → 12347 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2426953106 TSecr=2426953106
67	0.172426166	::1	::1	TCP	94	59680 → 12347 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=70635771 TSecr=0 WS=128
68	0.172429125	::1	::1	TCP	74	12347 → 59680 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
69	0.172451442	127.0.0.1	127.0.0.1	TCP	74	47134 → 12347 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2426953107 TSecr=0 WS=128
70	0.172454615	127.0.0.1	127.0.0.1	TCP	74	12347 → 47134 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2426953107 TSecr=2426953107 WS=128
71	0.172457993	127.0.0.1	127.0.0.1	TCP	66	47134 → 12347 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2426953107 TSecr=2426953107
72	0.172499757	::1	::1	TCP	94	59682 → 12347 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=70635771 TSecr=0 WS=128
73	0.172501549	::1	::1	TCP	74	12347 → 59682 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
74	0.172514929	127.0.0.1	127.0.0.1	TCP	74	47136 → 12347 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2426953107 TSecr=0 WS=128
75	0.172517427	127.0.0.1	127.0.0.1	TCP	74	12347 → 47136 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2426953107 TSecr=2426953107 WS=128
76	0.172519929	127.0.0.1	127.0.0.1	TCP	66	47136 → 12347 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2426953107 TSecr=2426953107
77	0.172550579	::1	::1	TCP	94	59684 → 12347 [SYN] Seq=0 Win=65476 Len=0 MSS=65476 SACK_PERM=1 TSval=70635771 TSecr=0 WS=128
78	0.172552252	::1	::1	TCP	74	12347 → 59684 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
79	0.172564467	127.0.0.1	127.0.0.1	TCP	74	47138 → 12347 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2426953107 TSecr=0 WS=128
80	0.172566222	127.0.0.1	127.0.0.1	TCP	74	12347 → 47138 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2426953107 TSecr=2426953107 WS=128
81	0.172568624	127.0.0.1	127.0.0.1	TCP	66	47138 → 12347 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2426953107 TSecr=2426953107
82	0.172710811	127.0.0.1	127.0.0.1	HTTP	670	GET /a/b/2.jpg HTTP/1.1
83	0.172713781	127.0.0.1	127.0.0.1	TCP	66	12347 → 47132 [ACK] Seq=1 Ack=605 Win=64896 Len=0 TSval=2426953107 TSecr=2426953107

בשלב זה הדפדפן מנסה להקים חיבורים נוספים אל מול הלקוח שחלקם נכשלים ,ולאחר מכן שולח בקשות לתמונות (טרם קיבל את כול התמונות הקודמות שביקש) כך שכול חיבור שולח בקשה לתמונה אחרת.

שלב שליחת התמונות מתנהל באופן דומה למה שראינו בשליחת תמונה בודדת ,כלומר כול חבילות התמונה נשלחות אחת אחרי השנייה אל החיבור שביקש את התמונה .

נשים לב שכיוון שהשרת שלנו עובד מול חיבור בודד בכול פעם , בכול פעם השרת שולח לחיבור זה בלבד את החבילות של תמונה בודדת .

251	5.261028533	127.0.0.1	127.0.0.1	TCP	66	47138 → 12347 [ACK] Seq=603 Ack=524289 Win=1637120 Len=0 TSval=2426958196 TSecr=2426958196
252	5.261030158	127.0.0.1	127.0.0.1	HTTP	7769	HTTP/1.1 200 OK (JPEG JFIF image)
253	5.261031817	127.0.0.1	127.0.0.1	TCP	66	47138 → 12347 [ACK] Seq=603 Ack=531992 Win=1768064 Len=0 TSval=2426958196 TSecr=2426958196
254	5.303205049	127.0.0.1	127.0.0.1	TCP	66	47136 → 12347 [ACK] Seq=605 Ack=220820 Win=458496 Len=0 TSval=2426958238 TSecr=2426958193
255	6.262216661	127.0.0.1	127.0.0.1	TCP	66	12347 → 47138 [FIN, ACK] Seq=531992 Ack=603 Win=65536 Len=0 TSval=2426959197 TSecr=2426958196
256	6.303970722	127.0.0.1	127.0.0.1	TCP	66	47138 → 12347 [ACK] Seq=603 Ack=531993 Win=1768064 Len=0 TSval=2426959239 TSecr=2426959197
257	10.049917185	127.0.0.1	127.0.0.1	TCP	66	47126 → 12347 [FIN, ACK] Seq=4900 Ack=2791088 Win=2783616 Len=0 TSval=2426962985 TSecr=2426954184
258	10.049928447	127.0.0.1	127.0.0.1	TCP	66	12347 → 47126 [ACK] Seq=2791088 Ack=4901 Win=65536 Len=0 TSval=2426962985 TSecr=2426962985
259	10.049945714	127.0.0.1	127.0.0.1	TCP	66	47128 → 12347 [FIN, ACK] Seq=605 Ack=531993 Win=1768064 Len=0 TSval=2426962985 TSecr=2426955186
260	10.049955620	127.0.0.1	127.0.0.1	TCP	66	12347 → 47128 [ACK] Seq=531993 Ack=606 Win=65536 Len=0 TSval=2426962985 TSecr=2426962985
261	10.049967131	127.0.0.1	127.0.0.1	TCP	66	47132 → 12347 [FIN, ACK] Seq=605 Ack=220820 Win=458496 Len=0 TSval=2426962985 TSecr=2426956189
262	10.049968908	127.0.0.1	127.0.0.1	TCP	66	12347 → 47132 [ACK] Seq=220820 Ack=606 Win=65536 Len=0 TSval=2426962985 TSecr=2426962985
263	10.049976531	127.0.0.1	127.0.0.1	TCP	66	47134 → 12347 [FIN, ACK] Seq=603 Ack=220820 Win=458496 Len=0 TSval=2426962985 TSecr=2426957191
264	10.049977872	127.0.0.1	127.0.0.1	TCP	66	12347 → 47134 [ACK] Seq=220820 Ack=604 Win=65536 Len=0 TSval=2426962985 TSecr=2426962985
265	10.049991341	127.0.0.1	127.0.0.1	TCP	66	47136 → 12347 [FIN, ACK] Seq=605 Ack=220820 Win=458496 Len=0 TSval=2426962985 TSecr=2426958193
266	10.049992610	127.0.0.1	127.0.0.1	TCP	66	12347 → 47136 [ACK] Seq=220820 Ack=606 Win=65536 Len=0 TSval=2426962985 TSecr=2426962985
267	10.050014940	127.0.0.1	127.0.0.1	TCP	66	47138 → 12347 [FIN, ACK] Seq=603 Ack=531993 Win=1768064 Len=0 TSval=2426962985 TSecr=2426959197
268	10.050016694	127.0.0.1	127.0.0.1	TCP	66	12347 → 47138 [ACK] Seq=531993 Ack=604 Win=65536 Len=0 TSval=2426962985 TSecr=2426962985

בשורה 255 אפשר לראות את סיום שליחת התמונה האחרונה ולאחריה, ניתוק כול החיבורים שהיו מחוברים לשרת אחד אחרי השני.

סה"כ הדפדפן פתח אל מול השרת 6 חיבורים מוצלחים אשר דרכם נשלחו התמונות:

ניתן

פורט חיבור	מידע שהועבר			
47126	Ref.html	a/2.jpg	a/b/4.jpg	a/5.jpg
(47126)	/a/b/5.jpg	a/6.jpg	a/b/6.jpg	
47128	a/1.jpg	a/b/1.jpg		
47132	a/3.jpg			
47134	a/3.jpg			
47136	a/b/3.jpg			
47138	a/4.jpg			

לראות כי מרבית מהבקשות (7)נשלחו מפורט 27126 שזהו החיבור הראשון.

-קבלת connection: close מצד הלקוח ,סעיף זה לא קרה באופן טבעי בדפדפן שבדקנו שהשתמשנו בו ,לכן על פי הנחיית המרצה ביצענו סימולציה של התרחיש ע"י קוד לקוח מותאם (קוד שנשלח על ידי המרצה, ושונה על מנת לקבל את התוצאה הרצויה) השרת נמצא בפורט 8088.

```python
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

s.connect(('127.0.0.1', 8088))

s.send(b'GET /a/b/ref.html HTTP/1.1\r\nHost: localhost:
8080\r\nConnection: close\r\nPragma: no-cache\r\nCache-
Control: no-cache\r\nUpgrade-Insecure-Requests: 1\r\nSec-
Fetch-Site: none\r\nSec-Fetch-Mode: navigate\r\nSec-Fetch-
User: ?1\r\nSec-Fetch-Dest: document\r\nAccept-Encoding:
gzip, deflate, br\r\nAccept-Language: en-
US,en;q=0.9\r\n\r\n')

data = s.recv(10000)

print("Server sent: ", data)


s.close()
```

בקוד הלקוח מבקש את קובץ ref ,אך בניגוד למה שקורה כשהבקשה מגיעה מהדפדפן פה שיניתי את ערך ה Connection להיות close כדי שהחיבור ייסגר עוד בבקשה הראשונה.

	Time	Source	Destination	Protocol	Length	Info
1	0.000000	127.0.0.1	127.0.0.1	TCP	76	47468 → 8088 [SYN] Seq=0 Win=65495 Len=0 MSS=
2	0.000024	127.0.0.1	127.0.0.1	TCP	76	8088 → 47468 [SYN, ACK] Seq=0 Ack=1 Win=65483
3	0.000034	127.0.0.1	127.0.0.1	TCP	68	47468 → 8088 [ACK] Seq=1 Ack=1 Win=65536 Len=
4	0.000121	127.0.0.1	127.0.0.1	HTTP	375	GET /a/b/ref.html HTTP/1.1
5	0.000125	127.0.0.1	127.0.0.1	TCP	68	8088 → 47468 [ACK] Seq=1 Ack=308 Win=65280 Le
6	0.001017	127.0.0.1	127.0.0.1	HTTP	731	HTTP/1.1 200 OK
7	0.001083	127.0.0.1	127.0.0.1	TCP	68	47468 → 8088 [ACK] Seq=308 Ack=664 Win=64896
8	0.001114	127.0.0.1	127.0.0.1	TCP	68	8088 → 47468 [FIN, ACK] Seq=664 Ack=308 Win=6
9	0.001251	127.0.0.1	127.0.0.1	TCP	68	47468 → 8088 [FIN, ACK] Seq=308 Ack=665 Win=6
10	0.001255	127.0.0.1	127.0.0.1	TCP	68	8088 → 47468 [ACK] Seq=665 Ack=309 Win=65536

ניתן לראות בשורות 1-3 לחיצת ידיים מוצלחת , לאחרי הלקוח בשורה 4 שולח בקשת HTTP לקובץ ref אלא שלא כמו במקרה הקודם הפעם ערך השדה Connection הוא close.

```
      HT TP/1.1··
Host: lo calhost:
8080··Co nnection
: close·  ·Pragma:
 no-cach e··Cache
-Control : no-cac
he··Upgr ade-Inse
cure-Req uests: 1
··Sec-Fe tch-Site
: none··  Sec-Fetc
h-Mode:  navigate
```

על כן כמו שנדרש בתרגיל השרת שולח ללקוח את קובץ הref המבוקש ובסיום השליחה סוגר את החיבור באופן מידי ולא מחכה לקבל תגובה מהלקוח ניתן לראות זאת בפרקי הזמן הקצרים שעוברים משליחת הקובץ ועד שליחת הודעת הFIN מצד השרת .(לא חיכה ל timeout).

6	0.001017	127.0.0.1	127.0.0.1	HTTP	731	HTTP/1.1 200 OK
7	0.001083	127.0.0.1	127.0.0.1	TCP	68	47468 → 8088 [ACK] Seq=308 Ac
8	0.001114	127.0.0.1	127.0.0.1	TCP	68	8088 → 47468 [FIN, ACK] Seq=6