

## דוח - אלעד ישראל 313448888 ונקיס קرمיזי שלו 205832447

### חלק א'

שינינו את קוד השרת ואת קוד הלקוח כפי שנדרש בתרגיל.

להלן קוד השרת:

```
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)

    client_socket.send(data.upper())

    data = client_socket.recv(100)
    print('Received: ', data)

    client_socket.send(data.upper())

    client_socket.close()
    print('Client disconnected')
```

להלן קוד הלקוח:

```
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('10.0.2.15', 12345))

s.send(b'Narkis Shallev kremizi, Elad Israel')

data = s.recv(100)
print("Server sent: ", data)

s.send(b'205832447, 313448888')

data = s.recv(100)
print("Server sent: ", data)

s.close()
```

הסביר התעבורה בwireshark

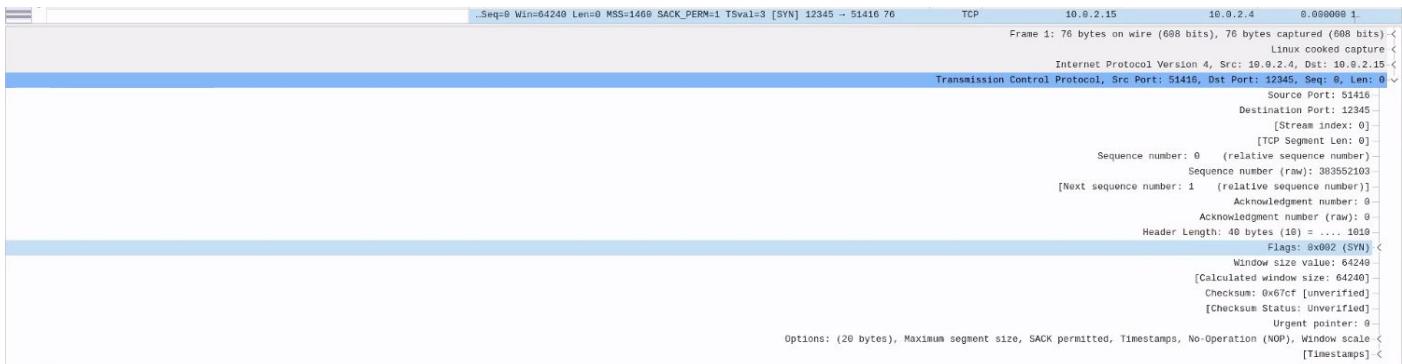
No	Time	Source	Destination	Protocol	Info.length
1	10.0.2.4 ..0.000000 1	10.0.2.15 ..0.000000 1		TCP	10.0.2.15 ..0.000000 1
2	10.0.2.15 ..0.000200 3		10.0.2.4 ..0.000200 3	TCP	10.0.2.15 ..0.000200 3
3	10.0.2.15 ..0.000223 9		10.0.2.4 ..0.000223 9	TCP	10.0.2.15 ..0.000223 9
4	10.0.2.4 ..0.000375 6		10.0.2.15 ..0.000375 6	TCP	10.0.2.15 ..0.000375 6
5	10.0.2.15 ..0.000486 7		10.0.2.4 ..0.000486 7	TCP	10.0.2.15 ..0.000486 7
6	10.0.2.4 ..0.000648 8		10.0.2.15 ..0.000648 8	TCP	10.0.2.15 ..0.000648 8
7	10.0.2.15 ..0.000653 2		10.0.2.4 ..0.000653 2	TCP	10.0.2.15 ..0.000653 2
8	10.0.2.15 ..0.000709 10		10.0.2.4 ..0.000709 10	TCP	10.0.2.15 ..0.000709 10
9	10.0.2.4 ..0.000725 11		10.0.2.15 ..0.000725 11	TCP	10.0.2.15 ..0.000725 11
10	10.0.2.4 ..0.000883 12		10.0.2.15 ..0.000883 12	TCP	10.0.2.15 ..0.000883 12
11	10.0.2.4 ..0.000984 13		10.0.2.15 ..0.000984 13	TCP	10.0.2.15 ..0.000984 13
12	10.0.2.15 ..0.000988 14		10.0.2.4 ..0.000988 14	TCP	10.0.2.15 ..0.000988 14

שיננו את החבילות כך שנראה רק חבילות שנשלחו ב프וטוקול TCP.

נשים לב שהIP של השרת הוא 10.0.2.15 והPORT שלו 12345. כמו כן, נשים לב שהIP של הלקוח הוא 10.0.2.4 והPORT שלו הוא 51416.

ראשית, מתבצע חיבור בין השרת ללקוח בשיטת "3-way handshake".

### בחבילה 1 הלוקו שולח בקשת הסתמכנות לשרת -

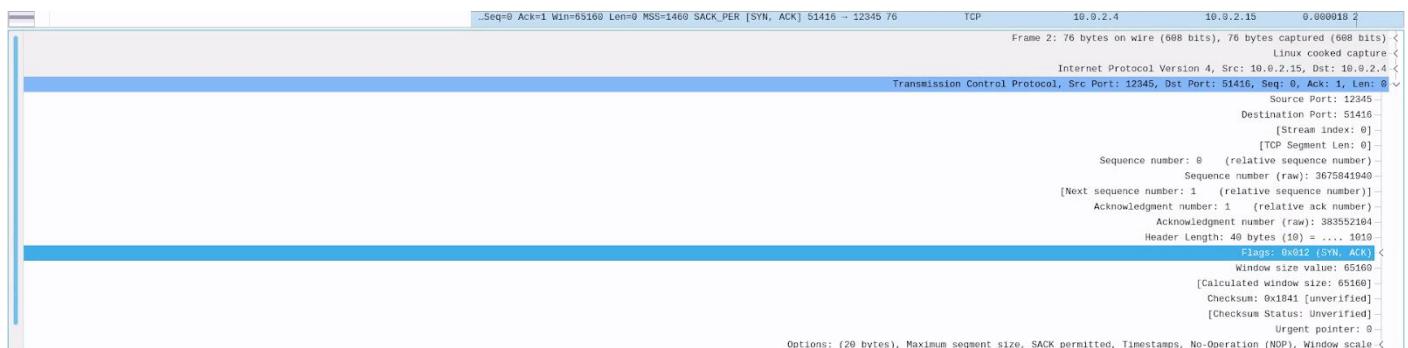


seq = 383552103 -->

אפשר לראות שבחבילה זו אין שום data אלא מדובר בheader TCP של שכבת התעבורה בלבד. כמו כן, ניתן לראות שהdag SYN בheader זה דלוק, כי הלוקו רוצה שהשרת יסתמך עליו.

הלקוח בוחר באקראי את המספר 383552103 כמספר סידורי וمعدכן את השרת שהוא מתחיל לשלוח מספר סידורי זה.

### בחבילה 2 השרת מאשר את בקשת SYN של הלקוח ושולח לו בקשת הסתמכנות משלו -



--> seq = 3675841940, ack = 383552104

שוב אפשר לראות שבחבילה זו אין שום data אלא מדובר בheader TCP של שכבת התעבורה בלבד. כמו כן, ניתן לראות שני דגלים דלוקים -

הראשון הוא דגל ACK כי השרת רוצה לעדכן את הלקוח שקיבל את בקשת SYN שלו. השרת מחזיר ACKnum שווה 383552103+1 (383552104). כדי לעדכן את הלקוח שהוא אכן קיבל את 383552103 ממנו והוא מzystה לקבל החל מ 383552104.

השני הוא דגל SYN כי השרת רוצה שהלקוח יסתמך גם כן עליו.

השרת בוחר באקראי את המספר 3675841940 כמספר סידורי וمعدכן את הלקוח שהוא מתחיל לשלוח מספר סידורי זה.

### בחבילה 3 הלקוח מאשר את בקשת SYN של השרת -

...Seq=1 Ack=1 Win=64256 Len=0 TSval=3709359551 TSecr [ACK] 12345 -> 51416 68	TCP	10.0.2.15	10.0.2.4	0.000290 3
		Frame 3: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) <		
		Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15 <		
		Transmission Control Protocol, Src Port: 51416, Dst Port: 12345, Seq: 1, Ack: 1, Len: 0 <		
		Source Port: 51416		
		Destination Port: 12345		
		[Stream Index: 0]		
		[TCP Segment Len: 0]		
		Sequence number: 1 (relative sequence number)		
		Sequence number (raw): 383552104		
		[Next sequence number: 1 (relative sequence number)]		
		Acknowledgment number: 1 (relative ack number)		
		Acknowledgment number (raw): 3675841941		
		Header Length: 32 bytes (8) = .... 1000		
		Flags: 0x010 (ACK) <		
		Window size value: 502		
		[Calculated window size: 64256]		
		[Window size scaling factor: 128]		
		Checksum: 0x6255 [unverified]		
		[Checksum Status: Unverified]		
		Urgent pointer: 0		

seq = 383552104, ack = 3675841941 -->

שוב אפשר לראות שבחייבה זו אין שום data אלא מדובר בheader של שכבת התעבורה בלבד. כמו כן, ניתן לראות שהdagל ACK בheader זה דлок כי הלקוח רוחה לעדכן את השרת שקיבל את בקשת SYN שלו. הלקוח מדווח ACKnum 3675841940 (3675841941) כדי לעדכן את השרת שהוא אכן קיבל 3675841940 (3675841941+1). מכך ניתן להבין שseq = 383552104, ack = 3675841941.

מעשיים הם יכולים להתחיל לדבר וממש לשЛОח הודעה אחת לשני.

נשים לב שהמספר הסידורי של הלקוח (המספר ממנו הוא מתחילה לשולח) הוא 383552104 שהוא תואם את ACKnum שדווח מהשרת (המספר ממנו השרת מצפה לקבל).

#### בחביבה 4 הלקוח שלוח לשרת חביבה עם הבקשה" Narkis Shallev Kremizi, Elad Israel "

...Seq=1 Ack=1 Win=64256 Len=35 TSval=3709359551 [PSH, ACK] 12345 -> 51416 163	TCP	10.0.2.15	10.0.2.4	0.000290 4
		Frame 4: 35 bytes on wire (280 bits), 35 bytes captured (280 bits) <		
		Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15 <		
		Transmission Control Protocol, Src Port: 12345, Seq: 1, Ack: 1, Len: 35 <		
		Source Port: 12345		
		Destination Port: 51416		
		[Stream Index: 0]		
		[TCP Segment Len: 35]		
		Sequence number: 1 (relative sequence number)		
		Sequence number (raw): 383552104		
		[Next sequence number: 36 (relative sequence number)]		
		Acknowledgment number: 1 (relative ack number)		
		Acknowledgment number (raw): 3675841941		
		Header Length: 32 bytes (8) = .... 1000		
		Flags: 0x010 (PSH, ACK) <		
		Window size value: 502		
		[Calculated window size: 64256]		
		[Window size scaling factor: 128]		
		Checksum: 0x2e9d [unverified]		
		[Checksum Status: Unverified]		
		Urgent pointer: 0		
		Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps <		
		[SEQ/ACK analysis] <		
		[Timestamps] <		
		TCP payload (35 bytes) <		
		Data (35 bytes) <		

seq = 383552104, ack = 3675841941, data = 35 -->

הפעם אפשר לראות בשכבת האפליקציה שיש data בגודל 35 בתים (כנראה שבחייבה הקודמת השרת טרם הספיק ליצור ורץ לעדכן שקיבל את הבקשה). כמו כן, בheader TCP של שכבת התעבורה ניתן לראות שני דגלים דלוקים - data ורץ.

הראשון הוא דגל PSH שלא הרחכנו עליו בשיעור.

השני הוא DAGL ACK כי בפרוטוקול TCP בכל חביבה מלבד החביבה הראשונה של SYN מעדכנים את הצד השני ב-ACKnum המציגו, והומרנו ACKnum היכי עדכני. במקרה שלנו ניתן לראות שערך של ACKnum זה הוא 3675841941 כי הלקוח לא קיבל שום דבר מאז חביבה 3.

נשים לב שהמספר הסידורי גם כן לא השתנה כי הלקוח לא שלח שום דבר לשרת.

### בחבילה 5 השרת מאשר ללקוח שקיבל את החבילה -

...Seq=1 Ack=36 Win=65152 Len=0 TSval=386487673 TSecr [ACK] 51416 → 12345 68	TCP	10.0.2.4	10.0.2.15	0.000223 5
				Frame 5: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) <
				Linux cooked capture <
				Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4 <
				Transmission Control Protocol, Src Port: 12345, Dst Port: 51416, Seq: 1, Ack: 36, Len: 0 <
				Source Port: 12345 Destination Port: 51416 [Stream index: 0] [TCP Segment Len: 0] Sequence number: 1 (relative sequence number) Sequence number (raw): 3675841941 [Next sequence number: 1 (relative sequence number)] Acknowledgment number: 36 (relative ack number) Acknowledgment number (raw): 383552139 Header Length: 32 bytes (8) = .... 1000 Flags: 0x010 (ACK) < Window size value: 509 [Calculated window size: 65152] [Window size scaling factor: 128] Checksum: 0xd839 [unverified] [Checksum Status: Unverified] Urgent pointer: 0 Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps < [SEQ/ACK analysis] < [Timestamps] <

<-- seq = 3675841941, ack = 383552139

אפשר לראות שבחבילה זו אין שום data אלא מדובר בheader TCP של שכבת התעבורה בלבד. כמו כן, ניתן לראות שהdag header ACK זה דלוק כי השרת רוצה לעדכן את הלקוח שקיבל את הבקשה שלו. השרת מוחזיר ACKnum 383552104+35 (383552139) כדי לעדכן את הלקוח שהוא אכן קיבל ממנו מידע בגודל 35 בתים החל מ-383552139. והוא מנסה לקבל מעכשו הצל מ-383552139.

נשים לב שהמספר הסידורי של השרת (המספר ממנו הוא מתחילה לשולח) הוא 3675841941 שזה תואם את הלקוח (המספר ממנו הלקוח מנסה לקבל).

### בחבילה 6 השרת שלוח ללקוח חבילה עם התשובה - "NARKIS SHALLEV KREMIZI, ELAD ISRAEL"

...Seq=1 Ack=36 Win=65152 Len=35 TSval=386487673 [PSH, ACK] 51416 → 12345 103	TCP	10.0.2.4	10.0.2.15	0.000376 6
				Frame 6: 103 bytes on wire (824 bits), 103 bytes captured (824 bits) <
				Linux cooked capture <
				Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4 <
				Transmission Control Protocol, Src Port: 12345, Dst Port: 51416, Seq: 1, Ack: 36, Len: 35 <
				Source Port: 12345 Destination Port: 51416 [Stream index: 0] [TCP Segment Len: 35] Sequence number: 1 (relative sequence number) Sequence number (raw): 3675841941 [Next sequence number: 36 (relative sequence number)] Acknowledgment number: 36 (relative ack number) Acknowledgment number (raw): 383552139 Header Length: 32 bytes (8) = .... 1000 Flags: 0x010 (PSH, ACK) < Window size value: 509 [Calculated window size: 65152] [Window size scaling factor: 128] Checksum: 0xd85c [unverified] [Checksum Status: Unverified] Urgent pointer: 0 Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps < [SEQ/ACK analysis] < [Timestamps] < TCP payload (35 bytes) < Data (35 bytes) <

<-- seq = 3675841941, ack = 383552139, data = 35

הפעם אפשר לראות בשכבת האפליקציה שיש data בגודל 35 בתים (כנראה שבחבילה הקודמת השרת טרם הספיק ליצור data ורץ לעדכן שקיבל את הבקשה). כמו כן, בheader TCP של שכבת התעבורה ניתן לראות שני דגלים דלוקים -

הראשון הוא דגל PSH שלא הרחמנו עליו בשיעור.

השני הוא דגל ACK כי בפרוטוקול TCP בכל חבילה הראשונה של SYN מעדכנים את הצד השני בsumACKnum המצביע, והוACKnum הכי עדכני. במקרה שלנו ניתן לראות שערך של sumACKnum זה הוא 383552139 כי השרת לא קיבל שום דבר מאז חבילה 5.

נשים לב שהמספר הסידורי גם כן לא השתנה כי השרת לאשלח שום דבר ללקוח.

### - בחבילה 7 הלקוח מאשר לשרת שקיבל את החבילה -

Seq=36 Ack=36 Win=64256 Len=0 TSval=3709359551 TSe [ACK] 12345 → 51416 68	TCP	10.0.2.15	10.0.2.4	0.000466 8
Frame 7: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) <--> Linux cooked capture <				
Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15 <--> Transmission Control Protocol, Src Port: 51416, Dst Port: 12345, Seq: 36, Ack: 36, Len: 0				
Source Port: 51416 Destination Port: 12345 [Stream index: 0] [TCP Segment Len: 0]				
Sequence number: 36 (relative sequence number) Sequence number (raw): 383552139 [Next sequence number: 36 (relative sequence number)] Acknowledgment number: 36 (relative ack number) Acknowledgment number (raw): 3675841976				
Header Length: 32 bytes (8) = .... 1000 Flags: 0x010 (ACK) Window size value: 502 [Calculated window size: 64256] [Window size scaling factor: 128] Checksum: 0x620F [unverified] [Checksum Status: Unverified] Urgent pointer: 0				
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps < [SEQ/ACK analysis] < [Timestamps] <				

seq = 383552139, ack = 3675841976 -->

אפשר לראות שבחבילה זו אין שום data אלא מדובר בheader של שכבת התעבורה בלבד. כמו כן, ניתן לראות שהדגל ACK בheader זה דלוק כי הלקוח רוצה לעדכן את השרת שקיבל את התשובה שלו. הלקוח מוחזיר sumACKnum שווה 3675841976 (3675841941+35) כדי לעדכן את השרת שהוא אכן קיבל ממנו מידע בגודל 35 בתים החל מ1941 ועד 3675841976. ושהוא מցפה לקבל מעכשו החול מ3675841976.

נשים לב שהמספר הסידורי השטנה כי כל עוד הלקוח לא קיבל timeout על חבילה 5 (ACK) ששלח לו השרת הוא מניח שהכל בסדר ושולח כבר את החבילה הבאה. הלקוח שלח לשרת בחבילה 4 מידע בגודל 35 בתים החל מ3675841941 ולקן החבילה הבאה אמורה להישלח החול מ35+35 (383552139) בדיקן כפי שכתוב במספר הסידורי של החבילה הנוכחית.

### - בחבילה 8 הלקוח שלוח לשרת חבילה עם הבקשה "205832447 ,313448888"

Seq=36 Ack=36 Win=64256 Len=26 TSval=3709359551 [PSH, ACK] 12345 → 51416 88	TCP	10.0.2.15	10.0.2.4	0.099648 8
Frame 8: 88 bytes on wire (704 bits), 88 bytes captured (704 bits) <--> Linux cooked capture <				
Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15 <--> Transmission Control Protocol, Src Port: 51416, Dst Port: 12345, Seq: 36, Ack: 36, Len: 20				
Source Port: 51416 Destination Port: 12345 [Stream index: 0] [TCP Segment Len: 20]				
Sequence number: 36 (relative sequence number) Sequence number (raw): 383552139 [Next sequence number: 56 (relative sequence number)] Acknowledgment number: 36 (relative ack number) Acknowledgment number (raw): 3675841976				
Header Length: 32 bytes (8) = .... 1000 Flags: 0x010 (PSH, ACK) Window size value: 502 [Calculated window size: 64256] [Window size scaling factor: 128] Checksum: 0x65ed [unverified] [Checksum Status: Unverified] Urgent pointer: 0				
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps < [SEQ/ACK analysis] < [Timestamps] < TCP payload (20 bytes) Data (20 bytes) <				

seq = 383552139, ack = 3675841976, data = 20 -->

הסביר כאן דומה להסביר על חבילה 4 (רק המספרים שונים).

### בחבילה 9 השרת מאשר ללקוח שקיבל את החבילה -

...Seq=36 Ack=56 Win=65152 Len=0 TStamp=386487673 TSec [ACK] 51416 → 12345 68	TCP	10.0.2.4	10.0.2.15	0.000653 9
Frame 9: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) ↴ Linux cooked capture ↴ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4 ↴ Transmission Control Protocol, Src Port: 12345, Dst Port: 51416, Seq: 36, Ack: 56, Len: 0 ↴ Source Port: 12345 Destination Port: 51416 [Stream Index: 0] [TCP Segment Len: 0] Sequence number: 36 (relative sequence number) Sequence number (raw): 3675841976 [Next sequence number: 36 (relative sequence number)] Acknowledgment number: 56 (relative ack number) Acknowledgment number (raw): 383552159 Header Length: 32 bytes (8) = ..., 1000 Flags: 0x010 (PSH, ACK) ↴ Window size value: 569 [Calculated window size: 65152] [Window size scaling factor: 128] Checksum: 0x1839 [Unverified] [Checksum Status: Unverified] [Checksum: 383552159] Urgent pointer: 0 Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps ↴ [SEQ/ACK analysis] ↴ [Timestamps] ↴				

<-- seq = 3675841976, ack = 383552159

הסביר כאן דומה להסביר על חבילה 5 (רק המספרים שונים).

### - "205832447 ,313448888" השרת שולח ללקוח חבילה עם התשובה

...Seq=36 Ack=56 Win=65152 Len=20 TStamp=386487677 [PSH, ACK] 51416 → 12345 88	TCP	10.0.2.4	10.0.2.15	0.000709 10
Frame 10: 88 bytes on wire (764 bits), 88 bytes captured (764 bits) ↴ Linux cooked capture ↴ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4 ↴ Transmission Control Protocol, Src Port: 12345, Dst Port: 51416, Seq: 36, Ack: 56, Len: 20 ↴ Source Port: 12345 Destination Port: 51416 [Stream Index: 20] [TCP Segment Len: 20] Sequence number: 36 (relative sequence number) Sequence number (raw): 3675841976 [Next sequence number: 56 (relative sequence number)] Acknowledgment number: 56 (relative ack number) Acknowledgment number (raw): 383552159 Header Length: 32 bytes (8) = ..., 1000 Flags: 0x010 (PSH, ACK) ↴ Window size value: 569 [Calculated window size: 65152] [Window size scaling factor: 128] Checksum: 0x1840 [Unverified] [Checksum Status: Unverified] [Checksum: 383552159] Urgent pointer: 0 Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps ↴ [SEQ/ACK analysis] ↴ [Timestamps] ↴ TCP payload (20 bytes) Data (20 bytes) ↴				

<-- seq = 3675841976, ack = 383552159, data = 20

הסביר כאן דומה להסביר על חבילה 6 (רק המספרים שונים).

### בחבילה 11 השרת שולח בקשה התנטקות ללקוח -

...Seq=56 Ack=56 Win=65152 Len=0 TSval=386487673 [FIN, ACK] 51416 → 12345 68	TCP	10.0.2.4	10.0.2.15	0.090725 11
	Frame 11: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) <			
	Linux cooked capture <-			
	Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4 <			
	Transmission Control Protocol, Src Port: 12345, Dst Port: 51416, Seq: 56, Ack: 56, Len: 0			
	Source Port: 12345			
	Destination Port: 51416			
	[Stream index: 0]			
	Sequence number: 56 (relative sequence number)			
	Sequence number (raw): 3675841996			
	[Next sequence number: 57 (relative sequence number)]			
	Acknowledgment number: 56 (relative ack number)			
	Acknowledgment number (raw): 383552159			
	Header Length: 32 bytes (8) = ... 1000			
	Flags: 0x011 (FIN, ACK) <			
	Window size value: 509			
	[Calculated window size: 65152]			
	[Window size scaling factor: 128]			
	Checksum: 0x1839 [unverified]			
	[Checksum Status: Unverified]			
	Urgent pointer: 0			
	Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps <			
	[Timestamps] <			

<-- seq = 3675841996, ack = 383552159

אפשר לראות שבחביבה זו אין שום data מדווח בheader של שכבת התעבורה בלבד. כמו כן, ניתן לראות שני דגלים דולקים -

הראשון הוא דגל FIN כי השרת רוצה לסגור את החיבור.

השני הוא דגל ACK כי בפרוטוקול TCP בכל חבילה מלבד החביבה הראשונה של SYN מעדכנים את הצד השני בsumACK המוצבר, והו sumACK הכי עדכני. במקרה שלנו ניתן לראות שערך של sumACK זה הוא 383552159 כי השרת לא קיבל שום דבר מאז חבילה 9.

נשים לב שהמספר הסידורי כה שתגנה למרתת שהשרת לא קיבל ACK על החביבה הקודמת ששלחה ללקוח (כנראה החביבה או מהו אובל כל עוד השרת לא קיבל timeout על החביבה הוא מניח שהכל בסדר ושולח כבר את החביבה הבאה). השרת שלח ללקוח מידע בגין 20 בתים החל מ60+20 ולקמן החביבה הבאה אמורה להשליח החל מ60+20+20 (3675841976) בדיק כי כתוב במספר הסידורי של החביבה הנוכחית.

## בחביבה 12 הלקוח מאשר לשרת לקבל את החביבה (10) -

...Seq=56 Ack=56 Win=64256 Len=0 TSval=3709359552 TSe [ACK] 12345 → 51416 68	TCP	10.0.2.15	10.0.2.4	0.090883 12
	Frame 12: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) <			
	Linux cooked capture <-			
	Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4 <			
	Transmission Control Protocol, Src Port: 12345, Dst Port: 51416, Seq: 56, Ack: 56, Len: 0			
	Source Port: 12345			
	Destination Port: 51416			
	[Stream index: 0]			
	Sequence number: 56 (relative sequence number)			
	Sequence number (raw): 383552159			
	[Next sequence number: 56 (relative sequence number)]			
	Acknowledgment number: 56 (relative ack number)			
	Acknowledgment number (raw): 3675841996			
	Header Length: 32 bytes (8) = ... 1000			
	Flags: 0x010 (ACK) <			
	Window size value: 502			
	[Calculated window size: 64256]			
	[Window size scaling factor: 128]			
	Checksum: 0xd1ed [unverified]			
	[Checksum Status: Unverified]			
	Urgent pointer: 0			
	Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps <			
	[SEQ/ACK analysis] <			
	[Timestamps] <			

seq = 383552159, ack = 3675841996 -->

הסביר כאן דומה להסביר על חבילה 7 ( רק המספרים שונים ).

## בחביבה 13 הלקוח מאשר את בקשת FIN של השרת ושולח לו בקשה התנטקיות משלו -

..Seq=56 Ack=57 Win=64256 Len=0 TSval=370935955 [FIN, ACK] 12345 -- 51416 88	TCP	10.0.2.15	10.0.2.4	0.000984 13
		Frame 13: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) <		
		Linux cooked capture <		
		Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15 <		
		Transmission Control Protocol, Src Port: 51416, Dst Port: 12345, Seq: 56, Ack: 57, Len: 8 <		
		Source Port: 51416		
		Destination Port: 12345		
		[Stream index: 0]		
		[TCP Segment Len: 0]		
		Sequence number: 56 (relative sequence number)		
		Sequence number (raw): 383552159		
		[Next sequence number: 57 (relative sequence number)]		
		Acknowledgment number: 57 (relative ack number)		
		Acknowledgment number (raw): 3675841997		
		Header Length: 32 bytes (8) = ... 1060		
		Flags: 0x0111 (FIN, ACK) <		
		Window size value: 502		
		[Calculated window size: 64256]		
		[Window size scaling factor: 128]		
		Checksum: 0x61e4 [unverified]		
		[Checksum Status: Unverified]		
		Urgent pointer: 0		
		Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps <		
		[SEQ/ACK analysis] <		
		[Timestamps] <		

seq = 383552159, ack = 3675841997 -->

שוב אפשר לראות שבכילה זו אין שום data אלא מדובר בheader TCP של שכבת התעבורה בלבד. כמו כן, ניתן לראות שני דגלים דולקים -

הראשון הוא דגל ACK כי הלוקו רוצה לעדכן את השרת שקיבל את בקשת FIN שלו. הלוקו מחזיר ACKnum שווה 3675841997 (3675841996+1).  
השני הוא דגל FIN כי הלוקו רוצה לסגור את החיבור גם כן.

נשים לב שהמספר הסידורי לא השתנה כי הלוקו לא שלח שום דבר לשרת.

#### בכילה 14 השרת מאשר את בקשת FIN של הלוקו -

..Seq=57 Ack=57 Win=65152 Len=0 TSval=386487673 TSec [ACK] 51416 -- 12345 68	TCP	10.0.2.15	10.0.2.4	0.000989 14
		Frame 14: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) <		
		Linux cooked capture <		
		Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4 <		
		Transmission Control Protocol, Src Port: 12345, Dst Port: 51416, Seq: 57, Ack: 57, Len: 0 <		
		Source Port: 12345		
		Destination Port: 51416		
		[Stream index: 0]		
		[TCP Segment Len: 0]		
		Sequence number: 57 (relative sequence number)		
		Sequence number (raw): 3675841997		
		[Next sequence number: 57 (relative sequence number)]		
		Acknowledgment number: 57 (relative ack number)		
		Acknowledgment number (raw): 383552160		
		Header Length: 32 bytes (8) = ... 1060		
		Flags: 0x0110 (ACK) <		
		Window size value: 569		
		[Calculated window size: 65152]		
		[Window size scaling factor: 128]		
		Checksum: 0x1839 [unverified]		
		[Checksum Status: Unverified]		
		Urgent pointer: 0		
		Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps <		
		[SEQ/ACK analysis] <		
		[Timestamps] <		

<-- seq = 3675841997, ack = 383552160

שוב אפשר לראות שבכילה זו דלק כי השרת רוצה לעדכן את הלוקו שקיבל את בקשת FIN שלו. השרת מחזיר ACKnum שווה 383552160 (383552159+1).  
נשים לב שהמספר הסידורי של השרת (המספר ממנו הוא מתחילה לשלח) הוא 3675841997 זהה תואם את הomega של FIN.

שחזור מהлокו (המספר ממנו הלוקו מצפה לקבל).

---

### קוד השרת V1:

```
1 import socket,sys
2
3 TCP_IP = '0.0.0.0'
4 TCP_PORT = int(sys.argv[1])
5 BUFFER_SIZE = 1024
6
7 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8 s.bind((TCP_IP, TCP_PORT))
9 s.listen(1)
10
11 while True:
12     conn, addr = s.accept()
13     print 'New connection from:', addr
14     while True:
15         data = conn.recv(BUFFER_SIZE)
16         if not data: break
17         print "received:", data
18         conn.send(data.upper())
19 conn.close()
```

הסבר מילוי -

השרת יוצר socket שמאגדר לפי פרוטוקול TCP וכתובות מסווג IPv4. לאחר מכן, השרת מבצע bind (קשירה) בין ה-socket למשהו שמתאפשר לקבלן. נשים לב שקשירה ל-IP=0.0.0.0 ל-PORT=5050 אומرت שהשרת יוכל לקבל הודעות מכל כתובת IP שנשלחת למכוונה. השרת מתחילה להאזין ללקוחות ומאפשר לך אחד בלבד בטור בכל רגע נתון.

בכל מקרה, השרת מקבל בבקשת התקשרות של לקוח ונוצר חיבור ייעודי לצורך ההתקשרות ביניהם. השרת מדפס "New connection from" + ה-IP וה-PORT של הלקוח שהתחבר אליו.

+ בנסיבות הפנימית, השרת מקבל בקשה מהלקוח שגודלה לכל היתר 1024 בתים (גודל הבапр) ומדפס "received" + "received" בבקשתו. לאחר מכן, הוא שולח את אותה בקשה באזיות גודלות אל הלקוח. השרת ממשיר לקבל עוד ועוד בקשות מהלקוח ולטפל בהן באותו זמן עד שהבקשת המתאפשרת ריקה. במקרה זה, הוא יצא מהולאה, מסיים את ההתקשרות עם הלקוח הספציפי וועבר ללקוח הבא.

פלט השרת -

```
elad@elad-virtualbox:~/Desktop/versions/v1$ python3 server.py 5050
New connection from: ('10.0.2.4', 40614)
received: b'Hello, World!'
```

---

### קוד הלוקה V1:

```

1 #!/usr/bin/env python
2
3 import socket,sys
4
5 TCP_IP = sys.argv[1]
6 TCP_PORT = int(sys.argv[2])
7 BUFFER_SIZE = 1024
8 MESSAGE = "Hello, World!"
9
10 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
11 s.connect((TCP_IP, TCP_PORT))
12 s.send(MESSAGE)
13 data = s.recv(BUFFER_SIZE)
14 s.close()
15
16 print "received data:", data

```

הסביר מילולי -

הלקוקה פותח socket שmagder לפי פרוטוקול TCP וכתובות מסוג IPv4.

לאחר מכן הוא מתחבר לכתובת הIP וה포ט של השירות שהוא קיבל כารוגומנט, ושולח את ההודעה "Hello, World".

אך הוא מקבל את התשובה שהשרת שולח בחזרה לתוכה הבאר שhogder בהתחלה (1024 בתים).

לבסוף, הוא סוגר את הsocket ומדפיס את data שהתקבל.

פלט הליקוח -

```
elad@elad-virtualbox:~/Desktop/versions/v1$ python3 client.py 10.0.2.15 5050
received data: b'HELLO, WORLD!'
```

-wireshark הסבר התעבורה ב-

נשים לב, הIp של הלוקה הוא 10.0.2.4, ושל השירות 10.0.2.15.

No	Time	Source	Destination	Protocol	Length	Info
3	0.000254779	10.0.2.4	10.0.2.15	TCP	76	5050 → 40614 [SYN]
4	0.000224792	10.0.2.15	10.0.2.4	TCP	76	40614 → 5050 [SYN, ACK]
5	0.000331245	10.0.2.4	10.0.2.15	TCP	68	40614 → 40614 [ACK]
6	0.000391068	10.0.2.4	10.0.2.15	TCP	81	3168075131 → 3284581703 [PSH, ACK]
7	0.000396754	10.0.2.4	10.0.2.15	TCP	68	3284581703 → 3168075132 [ACK]
8	0.000539611	10.0.2.4	10.0.2.15	TCP	81	40614 → 40614 [ACK]
9	0.000624536	10.0.2.4	10.0.2.15	TCP	68	3168075132 → 3284581703 [ACK]
10	0.000673164	10.0.2.4	10.0.2.15	TCP	68	3284581703 → 3168075132 [FIN, ACK]
11	0.000760292	10.0.2.4	10.0.2.15	TCP	68	40614 → 40614 [ACK]
12	0.000841686	10.0.2.4	10.0.2.15	TCP	68	3168075132 → 3284581703 [ACK]

בחבילה 3 הלוקה שולח בקשה הסתנכרנות לשרת.

בחבילה 4 השירות מאשר את בקשה SYN של הלוקה ושולח לו בקשה הסתנכרנות משלו.

בחבילה 5 הלוקה מאשר את בקשה SYN של השירות.

בחבילה 6 הלקוח שולח לשרת חvíלה עם הבקשה "Hello, World"

Frame 6: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface any, id 0 Linux cooked capture Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15 Transmission Control Protocol, Src Port: 40614, Dst Port: 5050, Seq: 1, Ack: 1, Len: 13 Data (13 bytes)  
0000 00 00 00 01 00 06 08 00 27 6e 78 c7 00 00 08 00 ..... 'nx.....  
0010 45 00 00 41 39 57 40 00 40 06 e9 4d 0a 00 02 04 E..A9W@. @..M...  
0020 0a 00 02 0f 9e a6 13 ba 97 1a 50 d6 81 74 47 71 .....P..tGq  
0030 80 18 01 f6 7c dd 00 00 01 01 08 0a c3 c6 bd 47 .....|.....G  
0040 bc d4 fd 7b 48 65 6c 6c 6f 2c 20 57 6f 72 6c 64 ....{Hell o, World  
0050 21 !

בחבילה 7 השרת מאשר ללקוח שקיבל את החvíלה.

- "!HELLO, WORLD"

Frame 7: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface any, id 0 Linux cooked capture Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4 Transmission Control Protocol, Src Port: 5050, Dst Port: 40614, Seq: 1, Ack: 14, Len: 13 Data (13 bytes)  
0000 00 04 00 01 00 06 08 00 27 ee ac 9d 00 00 08 00 ..... '.....  
0010 45 00 00 41 40 4b 40 00 40 06 e2 59 0a 00 02 0f E..A@K@. @..Y...  
0020 0a 00 02 04 13 ba 9e a6 81 74 47 71 97 1a 50 e3 .....tGq..P..  
0030 80 18 01 fd 18 46 00 00 01 01 08 0a bc d4 fd 7c .....F.....|  
0040 c3 c6 bd 47 48 45 4c 4c 4f 2c 20 57 4f 52 4c 44 ....GHELL O, WORLD  
0050 21 !

בחבילה 9 הלקוח מאשר לשרת שקיבל את החvíלה.

בחבילה 10 הלקוח שולח בקשה התנטקות לשרת.

בחבילה 11 השרת מאשר את בקשה FIN של הלקוח ושולח לו בקשה התנטקות משלו.

בחבילה 12 הלקוח מאשר את בקשה FIN של השרת.

---

קוד השרת 2:

```

1 import socket,sys
2
3 TCP_IP = '0.0.0.0'
4 TCP_PORT = int(sys.argv[1])
5 BUFFER_SIZE = 5
6
7 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8 s.bind((TCP_IP, TCP_PORT))
9 s.listen(1)
10
11 while True:
12     conn, addr = s.accept()
13     print 'New connection from:', addr
14     while True:
15         data = conn.recv(BUFFER_SIZE)
16         if not data: break
17         print "received:", data
18         conn.send(data.upper())
19 conn.close()

```

הסבר מילולי -

השרת זהה לשרת V1 מלבד זה שבולולאה הפנימית הוא מקבל בקשה מהלוקו שגודלה לכל היותר 5 בתים ולא 1024 בתים.

פלט השירות -

```

elad@elad-virtualbox:~/Desktop/versions/v2$ python3 server.py 5050
New connection from: ('10.0.2.4', 40618)
received: b'World'
received: b'! Hel'
received: b'lo, W'
received: b'orld!'

```

קוד הלוקו V2:

```

1 import socket,sys
2
3 TCP_IP = sys.argv[1]
4 TCP_PORT = int(sys.argv[2])
5 BUFFER_SIZE = 1024
6 MESSAGE = "World! Hello, World!"
7
8 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
9 s.connect((TCP_IP, TCP_PORT))
10 s.send(MESSAGE)
11 data = s.recv(BUFFER_SIZE)
12 s.close()
13
14 print "received data:", data

```

הסבר מילולי -

ההבדל היחיד בין הקליינט של V1 לV2 הוא שההודעה הנשלחת בV2 יותר ארוכה- "Hello, World! World!"

פלט הלוקו -

```
elad@elad-virtualbox:~/Desktop/versions/v2$ python3 client.py 10.0.2.15 5050
received data: b'WORLD'
```

- wireshark הסבר התעבורה ב-

נשים לב, הIP של הלוקו הוא 10.0.2.4, ושל השרת 10.0.2.15.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.0000000000 1	10.0.2.4	10.0.2.15	TCP	76	5050 → 40618 [SYN]
2	0.000020818 2	10.0.2.15	10.0.2.4	TCP	76	40618 → 5050 [ACK]
3	0.000176452 3	10.0.2.15	10.0.2.4	TCP	68	5050 → 40618 [ACK]
4	0.000426604 4	10.0.2.15	10.0.2.4	TCP	88	40618 → 5050 [ACK]
5	0.000434790 5	10.0.2.15	10.0.2.4	TCP	68	5050 → 40618 [ACK]
6	0.000495775 6	10.0.2.15	10.0.2.4	TCP	73	40618 → 5050 [ACK]
7	0.000633115 7	10.0.2.15	10.0.2.4	TCP	68	5050 → 40618 [ACK]
8	0.000633181 8	10.0.2.15	10.0.2.4	TCP	68	3287217421 → 40618 [ACK]
9	0.000638893 9	10.0.2.15	10.0.2.4	TCP	83	40618 → 5050 [ACK]
10	0.000667920 10	10.0.2.15	10.0.2.4	TCP	68	3170710850 → 40618 [ACK]
11	0.000802474 11	10.0.2.15	10.0.2.4	TCP	62	40618 → 5050 [RST]

בחבילה 1 הלוקו שולח בקשה הסתנכרנות לשרת.

בחבילה 2 השרת מאשר את בקשה SYN של הלוקו ושולח לו בקשה הסתנכרנות משלו.

בחבילה 3 הלוקו מאשר את בקשה SYN של השרת.

בחבילה 4 הלוקו שולח לשרת חבילה עם הבקשה - "Hello, World!"

Frame 4: 88 bytes on wire (704 bits), 88 bytes captured (704 bits) on interface any, id 0	
Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15	
Transmission Control Protocol, Src Port: 40618, Dst Port: 5050, Seq: 1, Ack: 1, Len: 20	
Data (20 bytes)	
0000	00 00 00 01 00 06 08 00 27 6e 78 c7 00 00 08 00
0010	45 00 00 48 c8 04 40 00 40 06 5a 99 0a 00 02 04
0020	0a 00 02 0f 9e aa 13 ba 06 31 a0 59 45 4b 96 61
0030	80 18 01 f6 02 83 00 00 01 01 08 0a c3 ee f5 0d
0040	bc fd 35 41 57 6f 72 6c 64 21 20 48 65 6c 6c 6f
0050	2c 20 57 6f 72 6c 64 21

בחבילה 5 השרת מאשר לлокו שקיבל את החבילה.

בחבילה 6 השרת שולח לлокו חבילה עם התשובה - "WORLD"

Frame 6: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface any, id 0	
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4	
Transmission Control Protocol, Src Port: 5050, Dst Port: 40618, Seq: 1, Ack: 21, Len: 5	
	Data (5 bytes)
0000	00 04 00 01 00 06 08 00 27 ee ac 9d 00 00 08 00
0010	45 00 00 39 5a 1f 40 00 40 06 c8 8d 0a 00 02 0f
0020	0a 00 02 04 13 ba 9e aa 45 4b 96 61 06 31 a0 6d
0030	80 18 01 fd 18 3e 00 00 01 01 08 0a bc fd 35 42
0040	c3 ee f5 0d 57 4f 52 4c 44

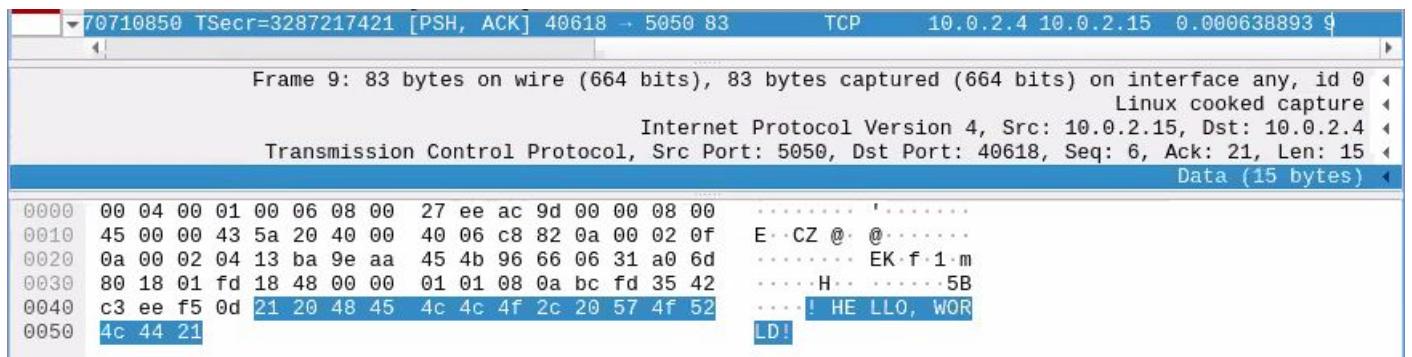
אומנם הלוקו שולח בחבילה 4 את כל 15 הבטים אר שכבת האפליקציה של השרת קוראת מהבאפר של שכבת התעבורה של השרת בכל פעם רק 5 בתים ומיד מטפלת בהם ושולחת אותם. אך, רק WORLD (חמשת הבטים הראשונים) נשלח בחרזה.

בנוסף, בזמן שהשרת מתחילה למסר ACK מהלוקו (כלומר מתחילה לחייבה 7) – בimentiים שכבת האפליקציה של השרת ממשיכה לרצות על לולאת `while` הפנימית ולעשות `recv` מהບאפר של שכבת התעבורה ומיד לאחר מכן `send`. כתוצאה לכך כשכבת התעבורה של השרת שולחת את המידע הבא (חייבה 9) – כל המידע משכבות האפליקציה כבר נשלח ביחד (כי יכולו נעשה `send` עד שכבת התעבורה המשיכה לשלחו).

בחבילה 7 הלקוח מאשר לשרת שקיבל את החבילה.

בחבילה 8 הלקוח שולח בבקשת התנטקיות לשרת.

בחבילה 9 השרת שולח ללקוח חבילה עם התשובה "! HELLO, WORLD!!!" (כלומר כל המידע שנותר לשילחה) –



בחבילה 10 השרת מאשר את בקשת FIN של הלקוח ושולח לו בבקשת התנטקיות משלו.

בחבילה 11 נשלח RST מהלוקו לשרת מכיוון שהגיעה אל הלקוח חבילה שהוא לא ציפה לה – חבילה 9, חבילה עם מידע שנשלחה מהשרת לאחר שהקלינט כבר ביקש מבחינותו מהשרת לסגור את החיבור(מכיוון המידע שהוזר חזר ב-2 פעימות' בכלל הקיראה האיטית עקב הבאפר המצומצם של השרת). RST זה קיזור של reset – חבילה זו אומרת למקבל (השרת) לעליון להפסיק להשתמש בחיבור זהה באופן מיידי. זו בעצם סגירה מיידית של החיבור ע"י צד אחד (לא נוהל לחיצת ידיהם).

קוד השרת 3:

```

1 import socket,sys
2
3 TCP_IP = '0.0.0.0'
4 TCP_PORT = int(sys.argv[1])
5 BUFFER_SIZE = 1024
6
7 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8 s.bind((TCP_IP, TCP_PORT))
9 s.listen(1)
10
11 while True:
12     conn, addr = s.accept()
13     print 'New connection from:', addr
14     while True:
15         data = conn.recv(BUFFER_SIZE)
16         if not data: break
17         print "received:", data
18         conn.send(data.upper()*1000)
19 conn.close()

```

הסביר מילולי –

השרת זהה לשרת 1 ובדומה להפעלתה הפנימית הוא שולח את אותה בקשה שקיבל מהלוקו לא רק באוטוות גדולות אלא גם משורשת 1000 פעמים.

פלט השירות - מתחבר לקלינט, מדפיס את הבקשה שהוא קיבל מהקלינט, ולאחר מכן מדפיס שגיאת שאומרת שהקלינט שלח RST בזמן שהשרת המתין וניסה לקבל ממנו מידע (עבור הבקשה הבאה- מידע עם תוכן או מידע רק לשימוש החיבור).

```
elad@elad-virtualbox:~/Desktop/versions/v3$ python3 server.py 5050
New connection from: ('10.0.2.4', 40648)
received: b'Hello, World!'
Traceback (most recent call last):
  File "server.py", line 16, in <module>
    data = conn.recv(BUFFER_SIZE)
ConnectionResetError: [Errno 104] Connection reset by peer
```

קוד הלesson 3:V3

```
1 import socket,sys
2
3 TCP_IP = sys.argv[1]
4 TCP_PORT = int(sys.argv[2])
5 BUFFER_SIZE = 1024
6 MESSAGE = "Hello, World!"
7
8 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
9 s.connect((TCP_IP, TCP_PORT))
10 s.send(MESSAGE)
11 data = s.recv(BUFFER_SIZE)
12 print "received data:", data
13 data = s.recv(BUFFER_SIZE)
14 print "received data:", data
15 s.close()
```

הסביר מילולי -

הקלינט זהה לו V7 חוץ מקבלת התשובה מהשרת. בגרסאות זו הקבלה מתבצעת פעמיים. כלומר הקלינט מקבל פעם אחת עד 1024 תווים לתוך הבאר, ולאחר מכן מקבל שוב עד 1024 תווים (מדפיס ואז דורס את מה שהיא במאפר).

**פלט הלוקו - קריאה ופלט של 1024 בתים (כגדל הבארף שלו) ואז שוב קריאה ופלט של 1024 הבטים הבאים.**

- wireshark ב

.10.0.2.15 נשים לב, הIp של הלקוח הוא 10.0.2.4, ושל השרת

בחבילה 3 הלקוח שלוח בקשה הסתמכנות לשרת.

בחבילה 4 השרת מאשר את בקשה ה-NYS של הלוקוח ושולח לו בקשה הסתנכרנות משלו.

בחבילה 5 הלקוח מאשר את בקשת NYIS של הרשות.

בחבילה 6 הלקוח שלח לשרת חיבור עם הבקשה "Hello, World" -

Frame 6: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface any, id 0  
Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15  
Transmission Control Protocol, Src Port: 40648, Dst Port: 5050, Seq: 1, Ack: 1, Len: 13  
Data (13 bytes)  
0000 00 00 00 01 00 06 08 00 27 6e 78 c7 00 00 08 00 .nx...  
0010 45 00 00 41 65 19 40 00 40 06 bd 8b 0a 00 02 04 E·Ae·@·@.....  
0020 0a 00 02 0f 9e c8 13 ba 2f 38 6f 80 f3 06 93 75 ...../8o....n  
0030 80 18 01 f6 03 ba 00 00 01 01 08 0a c4 f4 17 b5 .....  
0040 bd ce a5 89 48 65 6c 6c 6f 2c 20 57 6f 72 6c 64 .....Hello, World  
0050 21 !

בחבילה 7 השרת מאשר ללקוח שקיבל את החיבור.

בחבילה 8 ו-9 הלקוח שלח ללקוח חיבור עם התשובה "HELLO, WORLD!" משורשת 1000 פעמים (13 בתים \* 1000 = 13000 בתים בסה"כ).

התשובה נשלחת בתווך 2 חבילות. כל אחת מהן גודלה מ-MSS.

איך זה יכול להיות? מכיוון שמערכת הפעלה והכרטיסים תומכים בoffloading. במקרה זה TCP עושה offload ומוריד מעצמו את האחריות לזה - במקרה שהסגנטציה תיעשה ע"י מערכת הפעלה (על המעבד של המחשב) היא מתבצעת ע"י כרטיס הרשות עצמה (על המעבד שלו). מה שמצוג במקרה זה לפניו שCARTRIDGE הרשות עשו את הסגנטציה, ככלומר אנחנו רואים את החיבור לפניו שהוא מועברת לחיבור בCARTRIDGE ולכן היא גודלה מ-MSS, אבל בפועל זה יחולק (אבל זה יקרה אחרי ההסנה של wireshark).

למה 2 חבילות ולא 1? מכיוון שהגודל המקורי של חבילה אחת שהוא יודע לחלק בעצמו הוא 7240 בתים.

Frame 8: 7308 bytes on wire (58464 bits), 7308 bytes captured (58464 bits) on interface any, id 0  
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4  
Transmission Control Protocol, Src Port: 5050, Dst Port: 40648, Seq: 1, Ack: 14, Len: 7240  
Data (7240 bytes)  
0040 c4 f4 17 b5 48 45 4c 4c 4f 2c 20 57 4f 52 4c 44 ....HELL O, WORLD  
0050 21 48 45 4c 4c 4f 2c 20 57 4f 52 4c 44 21 48 45 !HELLO, WORLD!HE  
0060 4c 4c 4f 2c 20 57 4f 52 4c 44 21 48 45 4c 4c 4f LLO, WOR LD!HELLO  
0070 2c 20 57 4f 52 4c 44 21 48 45 4c 4c 4f 2c 20 57 57 !WORLD!HELLO\_W

Frame 9: 5828 bytes on wire (46624 bits), 5828 bytes captured (46624 bits) on interface any, id 0  
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4  
Transmission Control Protocol, Src Port: 5050, Dst Port: 40648, Seq: 7241, Ack: 14, Len: 5760  
Data (5760 bytes)  
0040 c4 f4 17 b5 21 48 45 4c 4c 4f 2c 20 57 4f 52 4c ....!HEL LO, WORL  
0050 44 21 48 45 4c 4c 4f 2c 20 57 4f 52 4c 44 21 48 D!HELLO, WORLD!HE  
0060 45 4c 4c 4f 2c 20 57 4f 52 4c 44 21 48 45 4c 4c ELLO, WO RLD!HELL  
0070 4f 2c 20 57 4f 52 4c 44 21 48 45 4c 4c 4f 2c 20 57 O\_WORLD!HELLO\_W

בחבילה 10 הלקוח מאשר לשרת שקיבל את חבילה 8.

בחבילה 11 הלקוח מאשר לשרת שקיבל את חבילה 9.

בחבילה 12 נשלח RST מהלקוח לשרת מכיוון שהלקוח מנסה לסגור את החיבור למטרות שיש בשכבה התעבורה שלו הרבעה מידע שמחכה להתקבל אצלו (מה שנשאר מהמידע של חבילה 8 ומהמידע של חבילה 9). RST זה קיזור של reset - חבילה זו אומerta למקבל (השרת) שעליו להפסיק להשתמש בחיבור זהה באופן מיידי. זו בעצם סגירה מיידית של החיבור ע"י צד אחד (לא נהיה לחיצת ידיים). ככלומר הקליינט "טרק את הטלפון בפנים" של השירות במקומות פשט להעתלים מזה שהשרת שחל לו עוד מידע שהוא לא קרא. זה נחשב יותר מנומס מאשר להעתלים ולהשאיר את השירות "באוויר" מבלי שידע מה קרה עם הלקוח.

## קוד השרת V4

```
1 import socket,sys,time
2
3 TCP_IP = '0.0.0.0'
4 TCP_PORT = int(sys.argv[1])
5 BUFFER_SIZE = 1024
6
7 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8 s.bind((TCP_IP, TCP_PORT))
9 s.listen(1)
10
11 while True:
12     conn, addr = s.accept()
13     print 'New connection from:', addr
14     while True:
15         time.sleep(5)
16         data = conn.recv(BUFFER_SIZE)
17         if not data: break
18         print "received:", data
19         conn.send(data.upper())
20     conn.close()
```

הסביר מילולי -

השתתfaction זה שבלולאה הפנימית הוא ישן 5 שניות לפני שהוא מקבל בקשה מהמלך.

**פלט השירות - חיבור חדש ולאחר מכן 5 שניות מודפס השאר.**

## קוד הלקוח V4:

```

1 import socket,sys
2
3 TCP_IP = sys.argv[1]
4 TCP_PORT = int(sys.argv[2])
5 BUFFER_SIZE = 1024
6 MESSAGE = "Hello, World!"
7
8 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
9 s.connect((TCP_IP, TCP_PORT))
10 s.send(MESSAGE*10)
11 s.send(MESSAGE*10)
12 s.send(MESSAGE*10)
13 s.send(MESSAGE*10)
14 data = s.recv(BUFFER_SIZE)
15 s.close()
16
17 print "received data:", data

```

הסביר מילולי -

הקלינט זהה ל17 חוץ משליחת ההודעה לשרת. הקלינט משליח את "Hello, World!" עשר פעמים, ושולח אותו כך ב-4 שlijות שונות אחת אחריו השניה.

פלט הלקוח - תגובה השרת לבקשה, מודפסת רק לאחר קצת יותר מ-5 שניות.

```

elad@elad-virtualbox:~/Desktop/versions/v4$ python3 client.py 10.0.2.15 5050
received data: b'HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, W
ORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO,
WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO,
WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO
, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELL
O, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HEL
LO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!HELLO, WORLD!'
```

- wiresharkה הסבר התעבורה ב-

נשים לב, ה-IP של הלקוח הוא 10.0.2.4, ושל השרת 10.0.2.15.

	Info	Length	Protocol	Destination	Source	Time	No.
Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3311383584 TSeср=0 WS=128 [SYN]	5050 → 40650 76	TCP	10.0.2.15	10.0.2.4	0.000000000 1		
Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=3191428534 TSeср=3311383584 WS=128 [SYN, ACK]	40650 → 5050 76	TCP	10.0.2.4	10.0.2.15	0.000308397 4		
Seq=1 Ack=1 Win=64256 Len=130 TSval=3311383584 TSeср=3191428534 [ACK]	5050 → 40650 68	TCP	10.0.2.15	10.0.2.4	0.000594146 5		
Seq=1 Ack=1 Win=64256 Len=130 TSval=3311383584 TSeср=3191428534 [PSH, ACK]	5050 → 40650 198	TCP	10.0.2.15	10.0.2.4	0.000594223 6		
Seq=1 Ack=1 Win=65152 Len=0 TSval=3191428534 TSeср=3311383584 [ACK]	40650 → 5050 68	TCP	10.0.2.4	10.0.2.15	0.000619386 7		
Seq=131 Ack=1 Win=64256 Len=390 TSval=3311383585 TSeср=3191428534 [PSH, ACK]	5050 → 40650 458	TCP	10.0.2.15	10.0.2.4	0.000870141 8		
Seq=1 Ack=521 Win=64768 Len=0 TSval=3191428535 TSeср=3311383585 [ACK]	40650 → 5050 68	TCP	10.0.2.4	10.0.2.15	0.000874501 9		
Seq=1 Ack=521 Win=64768 Len=520 TSval=3191433541 TSeср=3311383585 [PSH, ACK]	40650 → 5050 588	TCP	10.0.2.4	10.0.2.15	0.0008601573 10		
Seq=521 Ack=521 Win=64128 Len=0 TSval=3311388591 TSeср=3191433541 [ACK]	5050 → 40650 68	TCP	10.0.2.15	10.0.2.4	5.007228789 11		
Seq=521 Ack=521 Win=64128 Len=0 TSval=3311388591 TSeср=3191433541 [FIN, ACK]	5050 → 40650 68	TCP	10.0.2.15	10.0.2.4	5.007306554 12		
Seq=521 Ack=522 Win=64768 Len=0 TSval=3191433584 TSeср=3311388591 [ACK]	40650 → 5050 68	TCP	10.0.2.4	10.0.2.15	5.049891329 13		
Seq=521 Ack=522 Win=64768 Len=0 TSval=3191433584 TSeср=3311388591 [FIN, ACK]	40650 → 5050 68	TCP	10.0.2.4	10.0.2.15	10.0.12010309 14		
Seq=522 Ack=522 Win=64128 Len=0 TSval=3311393596 TSeср=3191438546 [ACK]	5050 → 40650 68	TCP	10.0.2.15	10.0.2.4	10.0.12456439 15		

בחבילה 1 הלקוח שולח בקשה הסתמכנות לשרת.

בחבילה 4 השרת מאשר את בקשה SYN של הלקוח ושולח לו בקשה הסתמכנות שלו.

בחבילה 5 הלקוח מאשר את בקשה SYN של השרת.

- בחבילה 6 הלקוח שלח לשרת חיבור עם הבקשה "Hello, World" משורשת 10 פעמים (13 בתים)

Secr=3191428534 [PSH, ACK] 5050 → 40650 198		TCP	10.0.2.15	10.0.2.4	0.000594223 6
Frame 6: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface any, id 0 1 Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15 1 Transmission Control Protocol, Src Port: 40650, Dst Port: 5050, Seq: 1, Ack: 1, Len: 130 1 Data (130 bytes) 1					
0000	00 00 00 01 00 06 08 00	27 6e 78 c7 00 00 08 00	.....'nx.....		
0010	45 00 00 b6 be 48 40 00	40 06 63 e7 0a 00 02 04	E.....H@...@.c.....		
0020	0a 00 02 0f 9e ca 13 ba	49 a6 b5 f5 40 a8 da 74	.....I...@..t		
0030	80 18 01 f6 de 59 00 00	01 01 08 0a c5 5f b4 20	.....Y.....		
0040	be 39 55 b6 48 65 6c 6c	6f 2c 20 57 6f 72 6c 64	.9U.Hell o, World		
0050	21 48 65 6c 6c 6f 2c 20	57 6f 72 6c 64 21 48 65	!Hello, World!He		
0060	6c 6c 6f 2c 20 57 6f 72	6c 64 21 48 65 6c 6c 6f	llo, Wor ld!Hello		
0070	2c 20 57 6f 72 6c 64 21	48 65 6c 6c 6f 2c 20 57	, World! Hello, W		
0080	6f 72 6c 64 21 48 65 6c	6c 6f 2c 20 57 6f 72 6c	orld!Hel lo, Wor		
0090	64 21 48 65 6c 6c 6f 2c	20 57 6f 72 6c 64 21 48	d!Hello, World!H		
00a0	65 6c 6c 6f 2c 20 57 6f	72 6c 64 21 48 65 6c 6c	ello, Wo rld!Hell		
00b0	6f 2c 20 57 6f 72 6c 64	21 48 65 6c 6c 6f 2c 20	o, World !Hello,		
00c0	57 6f 72 6c 64 21		World!		

בחבילה 7 השרת מאשר ללקוח שקיבל את החיבור.

בחבילה 8 הלקוח שלח לשרת חיבור עם הבקשה "Hello, World" משורשת 30 פעמים (390=3\*10\*13 בתים) - שהרי הלקוח עשה עוד 3 פעמים send (חזק מהמידע שנשלח בחבילה הקודמת) ובכל אחד כזה שלח את "Hello, World".

משורשת 10 פעמים TCP מחבר את החיבורות ושולח אותן ייחד.

Secr=3191428534 [PSH, ACK] 5050 → 40650 458		TCP	10.0.2.15	10.0.2.4	0.000870141 8
Linux cooked capture 1 Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.15 1 Transmission Control Protocol, Src Port: 40650, Dst Port: 5050, Seq: 131, Ack: 1, Len: 390 1 Data (390 bytes) 1					
0000	00 00 00 01 00 06 08 00	27 6e 78 c7 00 00 08 00	.....'nx.....		
0010	45 00 01 ba be 49 40 00	40 06 62 e2 0a 00 02 04	E.....I@...@.b.....		
0020	0a 00 02 0f 9e ca 13 ba	49 a6 b6 77 40 a8 da 74	.....I...w@..t		
0030	80 18 01 f6 96 8c 00 00	01 01 08 0a c5 5f b4 21	....._!.		
0040	be 39 55 b6 48 65 6c 6c	6f 2c 20 57 6f 72 6c 64	.9U.Hell o, World		
0050	21 48 65 6c 6c 6f 2c 20	57 6f 72 6c 64 21 48 65	!Hello, World!He		
0060	6c 6c 6f 2c 20 57 6f 72	6c 64 21 48 65 6c 6c 6f	llo, Wor ld!Hello		
0070	2c 20 57 6f 72 6c 64 21	48 65 6c 6c 6f 2c 20 57	, World! Hello, W		
0080	6f 72 6c 64 21 48 65 6c	6c 6f 2c 20 57 6f 72 6c	orld!Hel lo, Wor		
0090	64 21 48 65 6c 6c 6f 2c	20 57 6f 72 6c 64 21 48	d!Hello, World!H		
00a0	65 6c 6c 6f 2c 20 57 6f	72 6c 64 21 48 65 6c 6c	ello, Wo rld!Hell		
00b0	6f 2c 20 57 6f 72 6c 64	21 48 65 6c 6c 6f 2c 20	o, World !Hello,		
00c0	57 6f 72 6c 64 21 48 65	6c 6c 6f 2c 20 57 6f 72	World!He llo, Wor		
00d0	6c 64 21 48 65 6c 6c 6f	2c 20 57 6f 72 6c 64 21	ld!Hello , World!		
00e0	48 65 6c 6c 6f 2c 20 57	6f 72 6c 64 21 48 65 6c	Hello, W orld!Hel		
00f0	6c 6f 2c 20 57 6f 72 6c	64 21 48 65 6c 6c 6f 2c	lo, Wor ld!Hello,		
0100	20 57 6f 72 6c 64 21 48	65 6c 6c 6f 2c 20 57 6f	World!H ello, Wo		
0110	72 6c 64 21 48 65 6c 6c	6f 2c 20 57 6f 72 6c 64	rld!Hell o, World		
0120	21 48 65 6c 6c 6f 2c 20	57 6f 72 6c 64 21 48 65	!Hello, World!He		
0130	6c 6c 6f 2c 20 57 6f 72	6c 64 21 48 65 6c 6c 6f	llo, Wor ld!Hello		
0140	2c 20 57 6f 72 6c 64 21	48 65 6c 6c 6f 2c 20 57	, World! Hello, W		
0150	6f 72 6c 64 21 48 65 6c	6c 6f 2c 20 57 6f 72 6c	orld!Hel lo, Wor		
0160	64 21 48 65 6c 6c 6f 2c	20 57 6f 72 6c 64 21 48	d!Hello, World!H		
0170	65 6c 6c 6f 2c 20 57 6f	72 6c 64 21 48 65 6c 6c	ello, Wo rld!Hell		
0180	6f 2c 20 57 6f 72 6c 64	21 48 65 6c 6c 6f 2c 20	o, World !Hello,		
0190	57 6f 72 6c 64 21 48 65	6c 6c 6f 2c 20 57 6f 72	World!He llo, Wor		
01a0	6c 64 21 48 65 6c 6c 6f	2c 20 57 6f 72 6c 64 21	ld!Hello , World!		
01b0	48 65 6c 6c 6f 2c 20 57	6f 72 6c 64 21 48 65 6c	Hello, W orld!Hel		
01c0	6c 6f 2c 20 57 6f 72 6c	64 21	lo, Wor ld!		

בחבילה 9 השרת מאשר ללקוח שקיבל את החיבור האחרון.

עד רגע זה השרת לא שלח כלום מכיוון שהוא נמצא בקדים של 5 שניות מיד לאחר יצירת החיבור עם הקליינט. لكن בשלב זה לא יקרה כלום עד לשניה ה-5 לאחר הכניסה לsleep. בשניה ה-5 השרת יעורר לחימם, יקבל את המידע שסמתין לו בבאפר של שכבת התעבורה (בשלב זה זהו כל המידע שהקלייןיט שלח - כי הקלייןיט כבר הספיק מזמן לשולח הכל), ידפיס אותו, וילוח תשובה לכל 4 הבקשות שנשלחו מהקלייןיט יחד. וכך:

בחבילה 10 השרת שולח ללקוח חבילה עם התשובה "!"HELLO, WORLD!" משורשר 40 פעמים. ניתן לראות חבילה זו נשלחת מעט לאחר 5 שניות מתחילה ההתקשרות.

Secr=3311383585 [PSH, ACK] 40650 → 5050 588		TCP	10.0.2.4	10.0.2.15	5.006801573	10
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 10.0.2.4						
Transmission Control Protocol, Src Port: 5050, Dst Port: 40650, Seq: 1, Ack: 521, Len: 520						
Data (520 bytes)						
0000	00 04 00 01 00 06 08 00	27 ee ac 9d 00 00 08 00				
0010	45 00 02 3c 08 ca 40 00	40 06 17 e0 0a 00 02 0f	E..<..@.. @.....			
0020	0a 00 02 04 13 ba 9e ca	40 a8 da 74 49 a6 b7 fd	.....@...tI...			
0030	80 18 01 fa 1a 41 00 00	01 01 08 0a be 39 69 45	....A.....9iE			
0040	c5 5f b4 21 48 45 4c 4c	4f 2c 20 57 4f 52 4c 44	._.!HELL O, WORL			
0050	21 48 45 4c 4c 4f 2c 20	57 4f 52 4c 44 21 48 45	!HELLO, WORL!HE			
0060	4c 4c 4f 2c 20 57 4f 52	4c 44 21 48 45 4c 4c 4f	LL, WORL!HELLO			
0070	2c 20 57 4f 52 4c 44 21	48 45 4c 4c 4f 2c 20 57	,WORLD! HELLO, W			
0080	4f 52 4c 44 21 48 45 4c	4c 4f 2c 20 57 4f 52 4c	ORLD!HEL LO, WORL			
0090	44 21 48 45 4c 4c 4f 2c	20 57 4f 52 4c 44 21 48	D!HELLO, WORL!H			
00a0	45 4c 4c 4f 2c 20 57 4f	52 4c 44 21 48 45 4c 4c	ELLO, WORL!HE			
00b0	4f 2c 20 57 4f 52 4c 44	21 48 45 4c 4c 4f 2c 20	O, WORLD !HELLO,			
00c0	57 4f 52 4c 44 21 48 45	4c 4c 4f 2c 20 57 4f 52	WORLD!HE LLO, WOR			
00d0	4c 44 21 48 45 4c 4c 4f	2c 20 57 4f 52 4c 44 21	LD!HELLO ,WORLD!			
00e0	48 45 4c 4c 4f 2c 20 57	4f 52 4c 44 21 48 45 4c	HELLO, WORL!HEL			
00f0	4c 4f 2c 20 57 4f 52 4c	44 21 48 45 4c 4c 4f 2c	LO, WORL D!HELLO,			
0100	20 57 4f 52 4c 44 21 48	45 4c 4c 4f 2c 20 57 4f	WORLD!H ELL, WO			
0110	52 4c 44 21 48 45 4c 4c	4f 2c 20 57 4f 52 4c 44	RLD!HELL O, WORLD			
0120	21 48 45 4c 4c 4f 2c 20	57 4f 52 4c 44 21 48 45	!HELLO, WORLD!HE			
0130	4c 4c 4f 2c 20 57 4f 52	4c 44 21 48 45 4c 4c 4f	LL, WORL!HELLO			
0140	2c 20 57 4f 52 4c 44 21	48 45 4c 4c 4f 2c 20 57	,WORLD! HELLO, W			
0150	4f 52 4c 44 21 48 45 4c	4c 4f 2c 20 57 4f 52 4c	ORLD!HEL LO, WORL			
0160	44 21 48 45 4c 4c 4f 2c	20 57 4f 52 4c 44 21 48	D!HELLO, WORLD!H			
0170	45 4c 4c 4f 2c 20 57 4f	52 4c 44 21 48 45 4c 4c	ELLO, WORL!HE			
0180	4f 2c 20 57 4f 52 4c 44	21 48 45 4c 4c 4f 2c 20	O, WORLD !HELLO,			
0190	57 4f 52 4c 44 21 48 45	4c 4c 4f 2c 20 57 4f 52	WORLD!HE LLO, WOR			
01a0	4c 44 21 48 45 4c 4c 4f	2c 20 57 4f 52 4c 44 21	LD!HELLO ,WORLD!			
01b0	48 45 4c 4c 4f 2c 20 57	4f 52 4c 44 21 48 45 4c	HELLO, WORL!HEL			
01c0	4c 4f 2c 20 57 4f 52 4c	44 21 48 45 4c 4c 4f 2c	LO, WORL D!HELLO,			
01d0	20 57 4f 52 4c 44 21 48	45 4c 4c 4f 2c 20 57 4f	WORLD!H ELL, WO			
01e0	52 4c 44 21 48 45 4c 4c	4f 2c 20 57 4f 52 4c 44	RLD!HELL O, WORLD			
01f0	21 48 45 4c 4c 4f 2c 20	57 4f 52 4c 44 21 48 45	!HELLO, WORLD!HE			
0200	4c 4c 4f 2c 20 57 4f 52	4c 44 21 48 45 4c 4c 4f	LL, WORL D!HELLO			
0210	2c 20 57 4f 52 4c 44 21	48 45 4c 4c 4f 2c 20 57	,WORLD! HELLO, W			
0220	4f 52 4c 44 21 48 45 4c	4c 4f 2c 20 57 4f 52 4c	ORLD!HEL LO, WORL			
0230	44 21 48 45 4c 4c 4f 2c	20 57 4f 52 4c 44 21 48	D!HELLO, WORLD!H			
0240	45 4c 4c 4f 2c 20 57 4f	52 4c 44 21	ELLO, WORL D!			

בחבילה 11 הלקוח מאשר לשרת לקבל את החבילה.

בחבילה 12 הלקוח שולח בקשה התנטקות לשרת.

בחבילה 13 השרת מאשר את בקשה FIN של הלקוח.

בחבילה 14 השרת שולח לו בקשה התנטקות משלו.

בחבילה 15 הלקוח מאשר את בקשה FIN של השרת.

## ניטוח חלק ב:

פניה לכתובת localhost:8080 (כליוםר ל"/" - הטענה)

הסבר התעבורה בwireshark

	Info	Length	Protocol	Dst Port	Src Port	Destination	Source	Time	No
Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=455818906 TSeср=0 WS=128 [SYN] 8080 → 37506 76	TCP 8080	37506	127.0.0.1	127.0.0.1	0.000000	1			
Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=455818906 TSeср=455818906 WS=128 [SYN, ACK] 37506 → 8080 76	TCP 37506	8080	127.0.0.1	127.0.0.1	0.000005	2			
Seq=1 Ack=1 Win=65536 Len=0 MSS=65495 [ACK] 8080 → 37506 68	TCP 8080	37506	127.0.0.1	127.0.0.1	0.000010	3			
Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=455818906 TSeср=455818906 WS=128 [SYN] 8080 → 37510 76	TCP 8080	37510	127.0.0.1	127.0.0.1	0.000013	4			
Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=455818906 TSeср=455818906 WS=128 [SYN, ACK] 37510 → 8080 76	TCP 37510	8080	127.0.0.1	127.0.0.1	0.000015	5			
Seq=1 Ack=1 Win=65536 Len=0 TSval=455818906 TSeср=455818906 [ACK] 8080 → 37510 68	TCP 8080	37510	127.0.0.1	127.0.0.1	0.000019	6			
GET / HTTP/1.1 635	HTTP 8080	37506	127.0.0.1	127.0.0.1	0.001082	7			
HTTP/1.1 200 OK Continuation 242	HTTP 37506	8080	127.0.0.1	127.0.0.1	0.001087	8			
Seq=568 Ack=175 Win=65408 Len=0 TSval=455818907 TSeср=455818907 [ACK] 8080 → 37506 68	TCP 8080	37506	127.0.0.1	127.0.0.1	0.001172	9			
Seq=568 Ack=175 Win=65536 Len=0 TSval=455818907 TSeср=455818907 [FIN, ACK] 8080 → 37506 68	TCP 8080	37506	127.0.0.1	127.0.0.1	0.001172	11			
Seq=175 Ack=569 Win=65536 Len=0 TSval=455818907 TSeср=455818907 [FIN, ACK] 37506 → 8080 68	TCP 37506	8080	127.0.0.1	127.0.0.1	0.001178	12			
Seq=569 Ack=176 Win=65536 Len=0 TSval=455818907 TSeср=455818907 [ACK] 8080 → 37506 68	TCP 8080	37506	127.0.0.1	127.0.0.1	0.0011741	13			
Seq=1 Ack=1 Win=65536 Len=0 TSval=455818906 TSeср=455818906 [FIN, ACK] 37510 → 8080 68	TCP 37510	8080	127.0.0.1	127.0.0.1	1.002426	14			
Seq=1 Ack=2 Win=65536 Len=0 TSval=455819909 TSeср=455819909 [ACK] 8080 → 37510 68	TCP 8080	37510	127.0.0.1	127.0.0.1	1.003604	15			

```
Connection from: ('127.0.0.1', 37290)
timeout!
Client disconnected
Connection from: ('127.0.0.1', 37294)
timeout!
Client disconnected
```

בכניסת הכתובת localhost:8080 לדפדפן הוא מתחבר לשרת שלנו (שנמצא בפורט 8080 ובכתובת IP של localhost, localhost:8080) בתור קליינט מפורט 37506, ולאחר מכןשוב בתור קליינט אחר מפורט

<sup>1</sup> 37510

שם 2 חיבורים(2) פרוצדורות handshake של חיבור 2 של ניתוק).

רק בחיבור של הקליינט מפורט 37506 מתבצעת בקשה- בחיבור 7 הוא שולח בקשה לשרת עבור הקובץ "/"

	Info	Length	Protocol	Dst Port	Src Port	Destination	Source	Time	No
<pre>GET / HTTP/1.1 635</pre>									
				HTTP 8080	37506	127.0.0.1	127.0.0.1	0.001082	7
								Hypertext Transfer Protocol <	
0030 80 18 02 09 09 60 00 00 01 01 01 08 00 01 01 02 03 9b	...> GET / HTTP/1	...>							
0049 1b 2b 3e 9b 17 45 54 20 2f 29 48 54 54 58 2f 31	...> .1...Host : localch								
0050 2e 31 0d 0a 48 6f 73 74 3a 29 6c 6f 63 61 66 68	ost:8080 ..Connec								
0056 6f 73 74 3a 38 30 38 30 0d 0a 43 0f 66 66 65 63	tion: ke ep-alive								
0070 74 69 6f 6e 3a 20 60 65 65 70 2d 61 6c 69 70 65	..Cache- Control:								
0080 0d 0a 43 61 63 68 68 2d 43 6f 6a 74 72 6f 6e 3a	max-age =6..Upgr								
0090 28 6a 61 78 2d 61 67 65 3d 39 0d 0a 55 78 67 72	adeInse cure-Red								
0100 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 6g 6h 6i 6j	igest: 6.6.6.6.6.6.								
009b 0d 65 66 74 73 3a 20 31 0d 0a 55 73 65 72 2d 41	0 Mozilla/5.								
009c 07 65 66 74 73 3a 20 4d 6f 7a 69 6c 6d 61 2f 35 26	0 (X11; Linux x8								
009d 30 29 28 59 31 31 31 3b 2d 4c 69 66 75 78 28 78 38	6 64) Ap plewebK1								
009e 36 5f 36 34 29 20 41 70 70 66 65 57 65 62 6b 69	t/537.36 (KHTML,								
00f0 74 2f 35 33 37 2e 33 36 26 28 4b 48 54 4d 4c 2c	like Gecko) Chr								
0100 28 6a 69 6b 05 20 47 65 63 6b 6f 29 29 43 6b 72	ome/97.0 .4280.66								
0120 20 53 61 66 61 72 69 2f 05 33 37 2e 33 36 0d 0a	Safari/ 537.36 .								
0129 0d 65 66 67 68 69 6a 6b 6c 6d 6e 6f 6g 6h 6i 6j	Accept: */*								
0140 6c 6c 61 70 70 66 69 63 61 74 69 6f 66 2f 78 68	Content-Type: ap								
0150 74 6d 6c 2b 70 66 6d 6c 62 01 79 70 66 69 63 61 74	plication/xml, applicat								
0169 69 6f 6e 2f 78 6d 6c 6b 71 3d 30 2e 39 2c 69 69 6d	ion/xml; q=0.9, im								
0170 61 67 65 2f 71 66 69 66 2c 69 6d 61 67 65 2f 77	age/avif ,image/w								
0188 65 62 78 26 69 6d 61 67 65 2f 61 78 66 67 2c 2a	ebo,imag e/png,								
0198 2f 2b 73 3d 30 2e 38 26 61 70 70 66 69 63 61	:q=0.8 , applica								
01a0 74 69 6f 66 2f 73 69 67 66 65 64 2d 65 78 63 68	tion/sig ned-exch								
01b0 61 66 67 68 70 66 6d 62 01 79 70 66 69 63 61	ange:y=b 3;q=0.9 .								
01c0 65 66 67 68 69 6a 6b 6c 6d 6e 6f 6g 6h 6i 6j	QSA=1.0 .								
01d9 20 6e 6f 66 65 0d 0a 53 05 63 2d 46 65 74 63 68	none; \$ et- fetch								
01e9 2d 4d 6f 64 65 2a 0e 61 76 69 67 61 74 65 6d	-Mode: n avigate								
01f0 8a 53 65 03 2d 46 66 74 68 69 65 73 65 72 3a	-Sec-Fet ch-user:								
0200 26 3f 31 80 0a 53 65 63 2d 46 65 74 63 68 20 4d	?1- Se -Fetch-D								
0210 65 73 74 20 64 6f 63 75 60 65 6e 74 9d 0a 41	est: do mument: A								
0220 63 63 65 76 74 2d 45 6e 63 6f 64 69 66 67 3e 60	ccept: En coding:								
0228 67 73 69 79 2c 28 65 66 6c 61 74 65 2c 26 2d	gzip, de flate, b								
0240 72 0d 0a 41 63 63 69 70 74 20 4d 61 66 67 61 68	r ,Accep t-Langua								
0250 65 66 67 68 69 6a 6b 6c 6d 6e 6f 6g 6h 6i 6j	ge;q=1.0 , he;q=0								
0260 2e 39 2c 65 66 2d 55 53 3b 71 3d 38 2e 38 2c 65	,9,en,es ;q=0.8 , e								
0270 0e 3b 71 3d 39 2e 37 0d 0a 0d 0b	n;q=0.7 .								

ובהביבה 9 הוא מקבל ממנו תשובה הכוללת את הדף הנדרש(index.html) קובץ(/index.html) שההתאמנו לבקשת "/"

	Info	Length	Protocol	Dst Port	Src Port	Destination	Source	Time	No
<pre>HTTP/1.1 200 OK Continuation 242</pre>									
				HTTP 37506	8080	127.0.0.1	127.0.0.1	0.001172	9
								Hypertext Transfer Protocol <	
0030 80 03 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	E.....@. . . . *								
0455 00 00 c2 0b 6e 49 69 49 65 11 05 7f 00 00 01	...> GET / HTTP/1.1 200	...>							
7f 00 00 01 1f 00 02 02 dd a4 8b 00 d2 2a 03 03	OK- Con nection:								
0088 18 02 00 fe d6 09 00 01 00 09 00 1b 2b 03 9e	keep-alive-Con								
01b1 2b 3e 9b 40 54 54 59 2f 31 26 31 28 32 38 30	tent-Len gth: 108								
024f 4b 0d 0a 43 6f 6e 66 65 63 74 69 6f 6e 3a	...> ...> titleMy								
28 6b 65 65 70 61 66 65 3d 2f 74 69 74 6c 65	web pag e</title>								
3d 0d 0a 09 3c 2f 68 65 61 64 3e 0d 0a 09 3c 62	</>...</head>...<b>								
0f 64 79 3d 0d 0a 09 3c 2f 62 3e 0d 0a 09 3c 2f	ody>...<b><u>he								
6c 6f 64 79 3e 0d 0a 3c 2f 68 74 6d 6c 3e 0d 0a	file</u></b>...</>								
09 0d	body></html>								

<sup>1</sup> מספר הפורט בתמונה מההארט מושובשים(עקב החלפת הדוגמא). המספרים Wireshark ובהסבירם נוכנים.

לא נשלחות בקשה נוספת לאחר מילוי timeout והוא עובר לחכotta לקליניטים הבאים.

פניה לכתובת localhost:8080/a/oh\_no.jpg

הסביר התגובה בwireshark

	Info	Length	Protocol	Dst Port	Src Port	Destination	Source	Time	No
Seq=0 Ack=1 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=454737567 TSeср=454737567 WS=128 [SYN] 8080 → 37432 76	TCP 8080 37432 127.0.0.1 127.0.0.1 0.000000 1								
Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=454737567 TSeср=454737567 WS=128 [SYN, ACK] 37432 → 8080 76	TCP 37432 8080 127.0.0.1 127.0.0.1 0.000005 2								
Seq=1 Ack=1 Win=65536 Len=0 TSval=454737567 TSeср=454737567 [ACK] 8080 → 37432 68	TCP 8080 37432 127.0.0.1 127.0.0.1 0.000013 3								
Seq=0 Ack=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=454737567 TSeср=454737567 WS=128 [SYN] 8080 → 37436 76	TCP 8080 37436 127.0.0.1 127.0.0.1 0.000152 4								
Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=454737567 TSeср=454737567 WS=128 [SYN, ACK] 37436 → 8080 76	TCP 37436 8080 127.0.0.1 127.0.0.1 0.000155 5								
Seq=1 Ack=1 Win=65536 Len=0 TSval=454737567 TSeср=454737567 [ACK] 8080 → 37436 68	TCP 8080 37436 127.0.0.1 127.0.0.1 0.000158 6								
GET /a/oh_no.jpg HTTP/1.1 620	HTTP 8080 37432 127.0.0.1 127.0.0.1 0.004513 7								
Seq=1 Ack=553 Win=65024 Len=0 TSval=454737511 TSeср=454737511 [ACK] 37432 → 8080 68	TCP 37432 8080 127.0.0.1 127.0.0.1 0.004521 8								
Seq=1 Ack=553 Win=65536 Len=32768 TSval=454737511 TSeср=454737511 [TCP segment of a reassembled PDU] [ACK] 37432 → 8080 32836	TCP 37432 8080 127.0.0.1 127.0.0.1 0.011503 9								
Seq=553 Ack=32769 Win=48512 Len=0 TSval=454737518 TSeср=454737518 [ACK] 8080 → 37432 68	TCP 8080 37432 127.0.0.1 127.0.0.1 0.011511 10								
Seq=32769 Ack=553 Win=65536 Len=32768 TSval=454737518 TSeср=454737511 [TCP segment of a reassembled PDU] [PSH, ACK] 37432 → 8080 32836	TCP 37432 8080 127.0.0.1 127.0.0.1 0.011519 11								
HTTP/1.1 200 OK (JPEG JFIF image)Continuation 3379	HTTP 37432 8080 127.0.0.1 127.0.0.1 0.011528 12								
Seq=553 Ack=68848 Win=12544 Len=0 TSval=454737528 TSeср=454737511 [ACK] 8080 → 37432 68	TCP 8080 37432 127.0.0.1 127.0.0.1 0.021654 13								
37432 → 8080 [ACK] Seq=553 Ack=68848 Win=65536 Len=0 TSval=454737529 TSeср=454737518 [TCP Window Update] 68	TCP 8080 37432 127.0.0.1 127.0.0.1 0.022368 14								
Seq=553 Ack=68848 Win=65536 Len=0 TSval=454737536 TSeср=454737518 [FIN, ACK] 8080 → 37432 68	TCP 8080 37432 127.0.0.1 127.0.0.1 0.028912 15								
Seq=68848 Ack=554 Win=65536 Len=0 TSval=454737536 TSeср=454737536 [FIN, ACK] 37432 → 8080 68	TCP 37432 8080 127.0.0.1 127.0.0.1 0.028959 16								
Seq=554 Ack=68849 Win=65536 Len=0 TSval=454737536 TSeср=454737536 [ACK] 8080 → 37432 68	TCP 8080 37432 127.0.0.1 127.0.0.1 0.028962 17								
Seq=1 Ack=1 Win=0 TSval=454738537 TSeср=454737507 [FIN, ACK] 37432 → 8080 68	TCP 37436 8080 127.0.0.1 127.0.0.1 1.030202 18								
Seq=1 Ack=2 Win=65536 Len=0 TSval=454738541 TSeср=454738537 [ACK] 8080 → 37436 68	TCP 8080 37436 127.0.0.1 127.0.0.1 1.034233 19								

```
Connection from: ('127.0.0.1', 37432)
GET /a/oh_no.jpg HTTP/1.1
Host: localhost:8080
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.66 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: he-IL,he;q=0.9,en-US;q=0.8,en;q=0.7

data is empty
Client disconnected
Connection from: ('127.0.0.1', 37436)
timeout!
Client disconnected
```

בהכנסת הכתובת localhost:8080/a/oh\_no.jpg הוא מתחבר לשרת שלו (שנמצא בפורט 8080 ובכתובת IP של localhost, כיוון מר 127.0.0.1) בתור קלינט מפורט 37432, ולאחר מכן שוב בתור קלינט אחר מפורט 37436.

ישנו 2 חיבורים(2) פרוצדורות של חיבור ו2 של ניתוק).

רק בחיבור של הקלינט מפורט 37432 מתבצעת בקשה- בחייבה 7 הוא שולח בקשה לשרת עבור הקובץ "a/oh\_no.jpg/"

GET /a/oh_no.jpg HTTP/1.1 620	HTTP 8080 37432 127.0.0.1 127.0.0.1 0.004513 7
Frame 7: 620 bytes on wire (4960 bits), 620 bytes captured (4960 bits) < Linux cooked capture <	
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 <	
Transmission Control Protocol, Src Port: 37432, Dst Port: 8080, Seq: 1, Ack: 1, Len: 552 <	
Hypertext Transfer Protocol <	
GET /a/oh_no.jpg HTTP/1.1\r\n	
Host: localhost:8080\r\n	
Connection: keep-alive\r\n	
Upgrade-Insecure-Requests: 1\r\n	
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.66 Safari/537.36\r\n	
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n	
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, br\r\n	
Accept-Language: he-IL,he;q=0.9,en-US;q=0.8,en;q=0.7\r\n	
[Full request URI: http://localhost:8080/a/oh_no.jpg]	
[HTTP request 1/1]	
[Response in frame: 12]	

בחבילות 12-8 התמונה נשלחת במספר סגמנטים(הבקשה נועית).

### פניה לכתובת localhost:8080/a/b/ref.html

מפתח קצר היריעה רק בקשות HTTP מצורפות. התקשרות המלאה מפורטת בקובץ PCAP.

No	Time	Source	Destination	Src Port	Dst Port	Protocol	Length	Info
15	20.882371290	127.0.0.1 20.882371290	127.0.0.1 127.0.0.1 20.882896909	8080	37724	HTTP	621	GET /a/b/ref.html HTTP/1.1 621
17	20.882896909	127.0.0.1 20.882896909	127.0.0.1 127.0.0.1 20.915899007	8080	37724	HTTP	533	HTTP/1.1 200 OK Continuation 795
22	20.915899007	127.0.0.1 20.915899007	127.0.0.1 127.0.0.1 20.917584237	8080	37728	HTTP	3379	GET /a/oh_no.jpg HTTP/1.1 533
28	20.917584237	127.0.0.1 20.917584237	127.0.0.1 127.0.0.1 20.919502718	8080	37728	HTTP	238	(JPEG JFIF image)Continuation 3379
38	20.919502718	127.0.0.1 20.919502718	127.0.0.1 127.0.0.1 20.922468256	8080	37732	HTTP	533	GET /a/b/oh_no.jpg HTTP/1.1 535
49	20.922468256	127.0.0.1 20.922468256	127.0.0.1 127.0.0.1 20.923001674	8080	37732	HTTP	534	(JPEG JFIF image)Continuation ...238
51	20.923001674	127.0.0.1 20.923001674	127.0.0.1 127.0.0.1 20.923485079	8080	37736	HTTP	534	GET /a/oh_no2.jpg HTTP/1.1 534
59	20.923485079	127.0.0.1 20.923485079	127.0.0.1 127.0.0.1 20.950593930	8080	37736	HTTP	3379	(JPEG JFIF image)Continuation 3379
95	20.950593930	127.0.0.1 20.950593930	127.0.0.1 127.0.0.1 20.950761111	8080	37740	HTTP	536	GET /a/b/oh_no2.jpg HTTP/1.1 536
97	20.950761111	127.0.0.1 20.950761111	127.0.0.1 127.0.0.1 20.950782694	8080	37744	HTTP	534	GET /a/oh_no3.jpg HTTP/1.1 534
101	20.950782694	127.0.0.1 20.950782694	127.0.0.1 127.0.0.1 20.950801565	8080	37748	HTTP	536	GET /a/b/oh_no3.jpg HTTP/1.1 536
103	20.950801565	127.0.0.1 20.950801565	127.0.0.1 127.0.0.1 20.950823478	8080	37752	HTTP	534	GET /a/oh_no4.jpg HTTP/1.1 534
105	20.950823478	127.0.0.1 20.950823478	127.0.0.1 127.0.0.1 20.950839701	8080	37756	HTTP	536	GET /a/b/oh_no4.jpg HTTP/1.1 536
105	20.950839701	127.0.0.1 20.950839701	127.0.0.1 127.0.0.1 20.950839701	8080	37760	HTTP	534	GET /a/oh_no5.jpg HTTP/1.1 534
111	20.953184789	127.0.0.1 20.953184789	127.0.0.1 127.0.0.1 20.954604665	8080	377740	HTTP	238	(JPEG JFIF image)Continuation ...238
119	20.954604665	127.0.0.1 20.954604665	127.0.0.1 127.0.0.1 20.959168778	8080	377744	HTTP	3379	(JPEG JFIF image)Continuation 3379
127	20.959168778	127.0.0.1 20.959168778	127.0.0.1 127.0.0.1 20.964234515	8080	37764	HTTP	536	GET /a/b/oh_no5.jpg HTTP/1.1 536
141	20.964234515	127.0.0.1 20.964234515	127.0.0.1 127.0.0.1 20.964667562	8080	377748	HTTP	238	(JPEG JFIF image)Continuation ...238
142	20.964667562	127.0.0.1 20.964667562	127.0.0.1 127.0.0.1 20.965650705	8080	37768	HTTP	534	GET /a/oh_no6.jpg HTTP/1.1 534
151	20.965650705	127.0.0.1 20.965650705	127.0.0.1 127.0.0.1 20.982292030	8080	377752	HTTP	3379	(JPEG JFIF image)Continuation 3379
159	20.982292030	127.0.0.1 20.982292030	127.0.0.1 127.0.0.1 20.986992982	8080	377772	HTTP	536	GET /a/b/oh_no6.jpg HTTP/1.1 536
168	20.986992982	127.0.0.1 20.986992982	127.0.0.1 127.0.0.1 20.995554210	8080	377756	HTTP	238	(JPEG JFIF image)Continuation ...238
176	20.995554210	127.0.0.1 20.995554210	127.0.0.1 127.0.0.1 21.014583731	8080	377760	HTTP	3379	(JPEG JFIF image)Continuation 3379
186	21.014583731	127.0.0.1 21.014583731	127.0.0.1 127.0.0.1 21.018096416	8080	377764	HTTP	536	GET /a/oh_no6.jpg HTTP/1.1 536
194	21.018096416	127.0.0.1 21.018096416	127.0.0.1 127.0.0.1 21.026968450	8080	377772	HTTP	238	(JPEG JFIF image)Continuation ...238
204	21.026968450	127.0.0.1 21.026968450						

בכניסת הכתובת localhost:8080/a/b/ref.html הוא מתחבר לשרת שלנו (שנמצא ב포רט 8080 ובכתובת IP של localhost, כתובת קליינט, 127.0.0.1) בתור קליינט מפורט 37724. הבקשה נועית והוא מקבל את הקובץ מהשרת. בקבלת הקובץ ref.html רואה שיש עוד הרבה אלמנטים של img, וכך שולח בקשות נוספות (כל בקשה מקליינט נפרד) לשרת שלנו לקבלת תמונות אלו.

ישנו 13 חיבורים(13 פרוצדורות handshake של חיבור ו13 של ניתוק).

כאמור ישנה בקשה של הקליינט הראשון לקובץ ref.html ולאחר מכן עד בקשה מכל קליענט נוספת(קליענט 13-2) לכל תמונה בקובץ.

### פניה לכתובת localhost:8080/c.html (קובץ שאינו קיים)

	Info	Length	Protocol	Dst Port	Src Port	Destination	Source	Time	No
Seq=0	Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=464651209 TSeср=0 WS=128 [SYN]	8080 - 37944 76	TCP	8080	37944	127.0.0.1 127.0.0.1	0.0000330972	3	
Ack=1	Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=464651209 TSeср=464651209 WS=128 [SYN, ACK]	37944 - 8080 76	TCP	37944	8080	127.0.0.1 127.0.0.1	0.000035569	4	
Seq=1	Ack=1 Win=65536 Len=0 TSval=464651209 TSeср=464651209 [ACK]	8080 - 37944 68	TCP	8080	37944	127.0.0.1 127.0.0.1	0.0000405457	5	
Seq=0	Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=464651209 TSeср=0 WS=128 [SYN]	8080 - 37948 76	TCP	8080	37948	127.0.0.1 127.0.0.1	0.000176674	8	
Ack=1	Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=464651209 TSeср=464651209 WS=128 [SYN, ACK]	37944 - 8080 76	TCP	37948	8080	127.0.0.1 127.0.0.1	0.000179401	9	
Seq=1	Ack=1 Win=65536 Len=0 TSval=464651209 TSeср=464651209 [ACK]	8080 - 37948 68	TCP	8080	37948	127.0.0.1 127.0.0.1	0.000182556	10	
Seq=1	Ack=580 Win=65024 Len=0 TSval=464651213 TSeср=464651213 [ACK]	37944 - 8080 113	HTTP	8080	37944	127.0.0.1 127.0.0.1	0.003842097	11	
..Seq=1	Ack=580 Win=65536 Len=45 TSval=464651213 TSeср=464651213 [TCP segment of a reasse [PSH, ACK]]	37944 - 8080 113	TCP	37944	8080	127.0.0.1 127.0.0.1	0.003921626	12	
Seq=580	Ack=46 Win=65536 Len=0 TSval=464651213 TSeср=464651213 [ACK]	8080 - 37944 68	HTTP	8080	37944	127.0.0.1 127.0.0.1	0.004251405	15	
Seq=588	Ack=47 Win=65536 Len=0 TSval=464651213 [FIN, ACK]	8080 - 37944 68	TCP	8080	37944	127.0.0.1 127.0.0.1	0.004689114	16	
Seq=47	Ack=581 Win=65536 Len=0 TSval=464651214 TSeср=464651214 [ACK]	37944 - 8080 68	TCP	37944	8080	127.0.0.1 127.0.0.1	0.004612712	17	
Seq=0	Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=464651243 TSeср=464651243 WS=128 [SYN, ACK]	37952 - 8080 76	TCP	8080	37952	127.0.0.1 127.0.0.1	0.033621992	26	
Ack=1	Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=464651243 TSeср=464651243 WS=128 [SYN, ACK]	37952 - 8080 76	TCP	37952	8080	127.0.0.1 127.0.0.1	0.033626394	21	
Seq=1	Ack=1 Win=65536 Len=0 TSval=464651243 TSeср=464651243 [ACK]	8080 - 37952 68	TCP	8080	37952	127.0.0.1 127.0.0.1	0.033631225	22	
Seq=1	Ack=1 Win=65536 Len=0 TSval=464652214 [FIN, ACK]	8080 - 37952 68	TCP	37948	8080	127.0.0.1 127.0.0.1	1.004648184	23	
Seq=1	Ack=2 Win=65536 Len=0 TSval=464652217 TSeср=464652214 [ACK]	8080 - 37948 68	TCP	8080	37948	127.0.0.1 127.0.0.1	1.008265282	24	
Seq=1	Ack=1 Win=65536 Len=0 TSval=464653215 TSeср=464651243 [FIN, ACK]	37952 - 8080 68	TCP	37952	8080	127.0.0.1 127.0.0.1	2.006262283	25	
Seq=1	Ack=2 Win=65536 Len=0 TSval=464653217 TSeср=464653215 [ACK]	8080 - 37952 68	TCP	8080	37952	127.0.0.1 127.0.0.1	2.008150252	26	
<pre>Connection from: ('127.0.0.1', 37944) GET /c.html HTTP/1.1 Host: localhost:8080 Connection: keep-alive Cache-Control: max-age=0 Upgrade-Insecure-Requests: 1 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.66 Safari/537.36 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9 Sec-Fetch-Site: cross-site Sec-Fetch-Mode: navigate Sec-Fetch-User: ? Sec-Fetch-Dest: document Accept-Encoding: gzip, deflate, br Accept-Language: he-IL,he;q=0.9,en-US;q=0.8,en;q=0.7  Client disconnected Connection from: ('127.0.0.1', 37948) timeout! Client disconnected Connection from: ('127.0.0.1', 37952) timeout! Client disconnected</pre>									

בהכנסת הכתובת localhost:8080/c.html הוא מתחבר לשרת שלו (שנמצא בפורט 8080 ובכתובת IP של localhost, כלומר 127.0.0.1) בתווך קליינט מפורט 37944, וклиינט נוסף מפורט 37948, ועוד אחד מפורט 37952. הקליינט מפורט 37944 שולח את בקשת get אל השרת, ומכיון שהקובץ זה אינו קיים אצל השרת הוא מחזיר בתגובה קוד שגיאה "404 . "Not Found

ישנו 2 חיבורים(2) פרוצדורות handshake של חיבור 2 של ניתוק).

כאמור ישנה בקשה של הקליינט הראשון לקובץ html.c.

## פניה לכתובת localhost:8080/redirect (result.html ליתובlocalhost:8080/redirect)

	Info	Length	Protocol	Dst Port	Src Port	Destination	Source	Time	No
Seq=0	Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=465409864 TSeср=0 WS=128 [SYN]	8080 - 37990 76	TCP	8080	37990	127.0.0.1 127.0.0.1	0.0000000	1	
Ack=1	Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=465409864 TSeср=465409864 WS=128 [SYN, ACK]	37990 - 8080 76	TCP	37990	8080	127.0.0.1 127.0.0.1	0.000005	2	
Seq=1	Ack=1 Win=65536 Len=0 TSval=465409864 TSeср=465409864 [ACK]	8080 - 37990 68	TCP	8080	37990	127.0.0.1 127.0.0.1	0.000010	3	
Seq=0	Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=465409864 TSeср=0 WS=128 [SYN]	8080 - 37994 76	TCP	8080	37994	127.0.0.1 127.0.0.1	0.0000154	4	
Ack=1	Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=465409864 TSeср=465409864 WS=128 [SYN, ACK]	37994 - 8080 76	TCP	37994	8080	127.0.0.1 127.0.0.1	0.0000157	5	
Seq=1	Ack=1 Win=65536 Len=0 TSval=465409864 TSeср=465409864 [ACK]	8080 - 37994 68	TCP	8080	37994	127.0.0.1 127.0.0.1	0.0000160	6	
Seq=1	Ack=1 Win=65536 Len=0 TSval=465409864 TSeср=465409864 GET / redirect HTTP/1.1 617	37990 - 8080 68	HTTP	8080	37990	127.0.0.1 127.0.0.1	0.0000128	7	
Seq=1	Ack=550 Win=65024 Len=0 TSval=465409870 TSeср=465409870 [ACK]	37990 - 8080 68	TCP	37990	8080	127.0.0.1 127.0.0.1	0.0000140	8	
..Seq=1	Ack=550 Win=65536 Len=77 TSval=465409870 TSeср=465409870 [TCP segment of a reasse [PSH, ACK]]	37990 - 8080 145	TCP	37990	8080	127.0.0.1 127.0.0.1	0.0000248	9	
Seq=550	Ack=78 Win=65536 Len=0 TSval=465409870 TSeср=465409870 [ACK]	37990 - 8080 68	HTTP	37990	8080	127.0.0.1 127.0.0.1	0.0000260	10	
Seq=550	Ack=79 Win=65536 Len=0 TSval=465409870 TSeср=465409870 [FIN, ACK]	8080 - 37990 68	TCP	8080	37990	127.0.0.1 127.0.0.1	0.0000450	11	
Seq=79	Ack=551 Win=65536 Len=0 TSval=465409871 TSeср=465409871 [ACK]	37990 - 8080 68	TCP	37990	8080	127.0.0.1 127.0.0.1	0.0000598	12	
Seq=1	Ack=553 Win=65024 Len=0 TSval=465409875 TSeср=465409875 GET / result.html HTTP/1.1 620	37990 - 8080 68	HTTP	8080	37994	127.0.0.1 127.0.0.1	0.011446	14	
Seq=1	Ack=553 Win=65024 Len=0 TSval=465409875 TSeср=465409875 [ACK]	37994 - 8080 68	TCP	37994	8080	127.0.0.1 127.0.0.1	0.011453	15	
Seq=553	Ack=189 Win=65536 Len=0 TSval=465409876 TSeср=465409876 HTTP/1.1 200 OK Continuation 256	37994 - 8080 68	HTTP	8080	37994	127.0.0.1 127.0.0.1	0.011814	16	
Seq=553	Ack=189 Win=65408 Len=0 TSval=465409876 TSeср=465409876 [ACK]	8080 - 37994 68	TCP	8080	37994	127.0.0.1 127.0.0.1	0.011818	17	
Seq=553	Ack=189 Win=65536 Len=0 TSval=465409876 TSeср=465409876 [FIN, ACK]	8080 - 37994 68	TCP	8080	37994	127.0.0.1 127.0.0.1	0.012173	18	
Seq=189	Ack=554 Win=65536 Len=0 TSval=465409876 TSeср=465409876 [FIN, ACK]	8080 - 37994 68	TCP	37994	8080	127.0.0.1 127.0.0.1	0.012196	19	
Seq=554	Ack=190 Win=65536 Len=0 TSval=465409876 TSeср=465409876 [ACK]	8080 - 37994 68	TCP	8080	37994	127.0.0.1 127.0.0.1	0.012199	20	

```

Connection from: ('127.0.0.1', 37990)
GET /redirect HTTP/1.1
Host: localhost:8080
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.66 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: he-IL,he;q=0.9,en-US;q=0.8,en;q=0.7

Client disconnected
Connection from: ('127.0.0.1', 37994)
GET /result.html HTTP/1.1
Host: localhost:8080
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.66 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: he-IL,he;q=0.9,en-US;q=0.8,en;q=0.7

data is empty
Client disconnected

```

בהນוסת הכתובת localhost:8080/redirect הוא מתחבר לשרת שלנו (שנמצא בפורט 8080 ובכתובת IP של localhost, כמו כן, 127.0.0.1) בתור קלינט מפורט 0, 37990, וקלינט נוסף מפורט 37994. הקלינט מפורט 37990 שולח את בקשה get, והשרת בתגובה מסיב שהקובץ הועבר והמיקום שלו כרגע נמצא ב"/result.html".



כעת לאחר שהדפדפן קיבל את ההודעה אל הקלינט הראשון - הקלינט השני מפורט 37994 שולח בקשה get לקובץ במיקום החדש - "/result.html", והשרת מחזיר לו את דף זה.

ישנו 2 חיבורים(2) פרוטוקול handshake של חיבור ו2 של ניתוק).

כאמור ישנה בקשה של הקלינט הראשון ל redirect ולאחר מכן בעקבות התשובה שהשרת החזיר - עוד בקשה מהקלינט השני לקובץ העדכני.