

## חלק א

הIP של השרת: 192.168.1.30  
הIP של הקליינט: 192.168.1.44

נשאל בפורום: "האם מותר לשנות גם את קוד השרת כדי שהיא אפשר לשמר על החיבור פתוח בזמן שליחת שתי הודעהות?"

נענה: "שלום, זה מה שתתרגש מבקש. תודה"  
על כן, שינוינו את קוד השרת נוספת על קוד הלקוח, ככללו.  
קוד השרת:

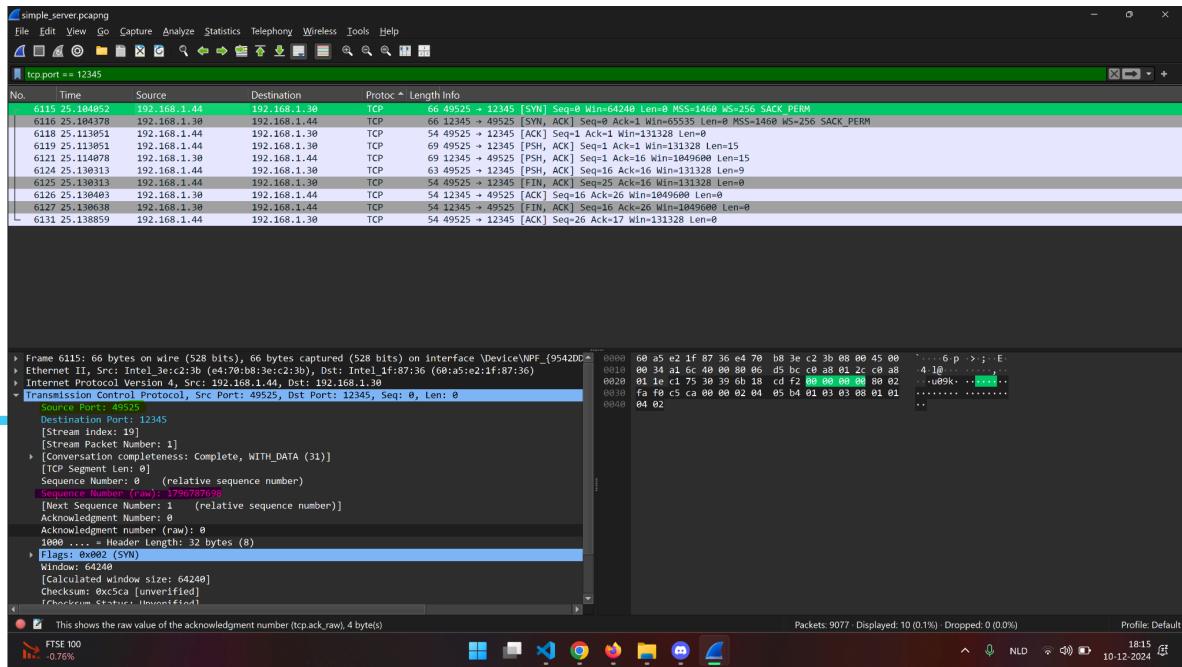
```
tcp_server.py > ...
import socket
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(('', 12345))
server.listen(5)

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

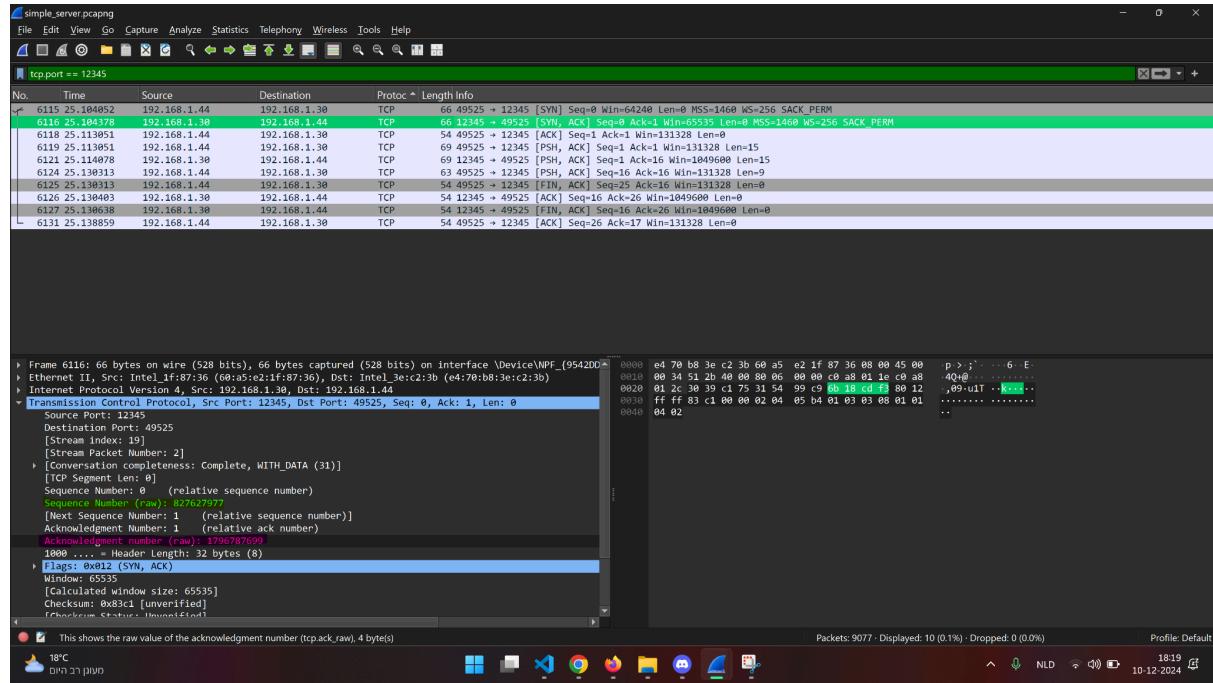
קוד הלקוח:

```
tcp_client.py > ...
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('192.168.1.30', 12345))
s.send(b'Gideon and Ayal')
data = s.recv(100)
print("Server sent: ", data)
s.send(b'329924567')
s.close()
```

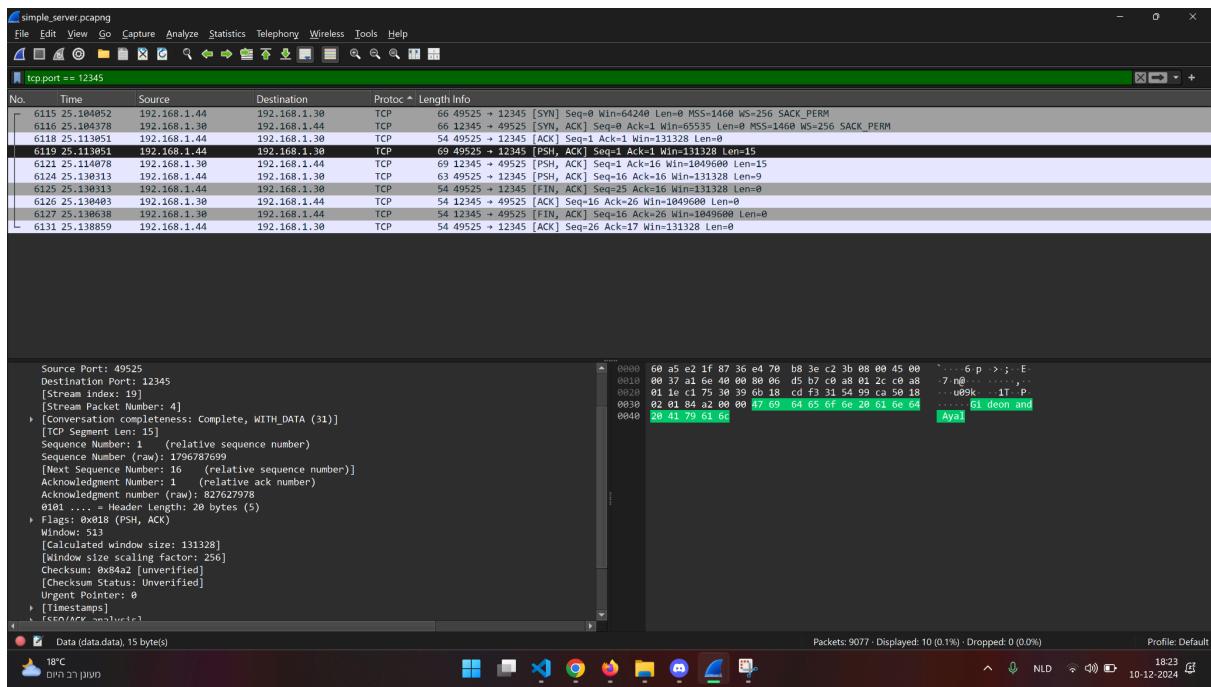
כפי שניתן לראות, בתחילת החיבור הלקוח פונה לשרת בבקשת SYN, המספר הסידורי המקורי הוא הדבר הנחמד הבא: 1796787698. והמספר של ACKNN הוא 0.



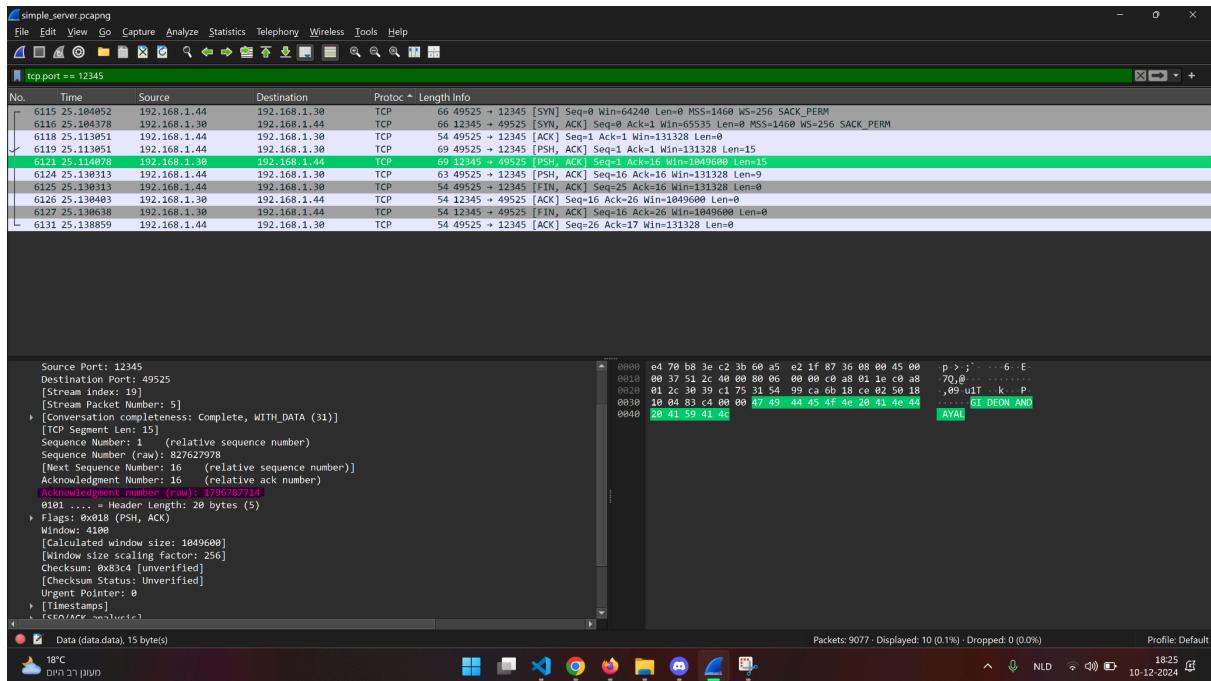
השרת מניב ב-ACK SYN ומבקש להסתنصرן. אם כך, המספר של ה-ACKNN CUT נדב ב-1 מהSEQ. השהיה מקודם, שחריו הוא מחזיר ACK. וכך המספר הוא 1796787699. המספר של SEQ של ה-ACK הוא .827627977.



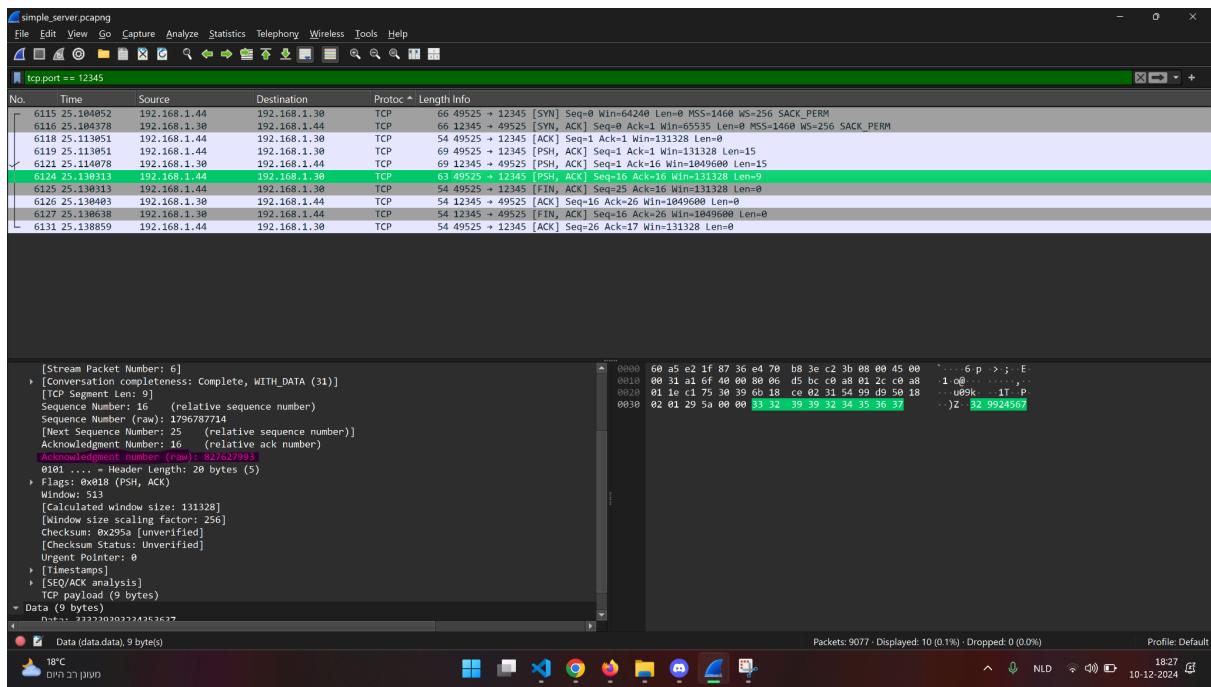
CUT לא נשים צילום מסך בכל רגע כדי לא להזכיר. הלקוח מחזיר ACK על ה-SYN של השרת, מעלה את ACKNN ב-1 (של השרת) ולאחר מכן הוא .827627978. לאחר מכן, הלקוח שלוח את השם (במקרה שלנו, השמות שלנו). לא התקדמנו עדין בSEQ, ומספר ACKNN עלה ב-1.



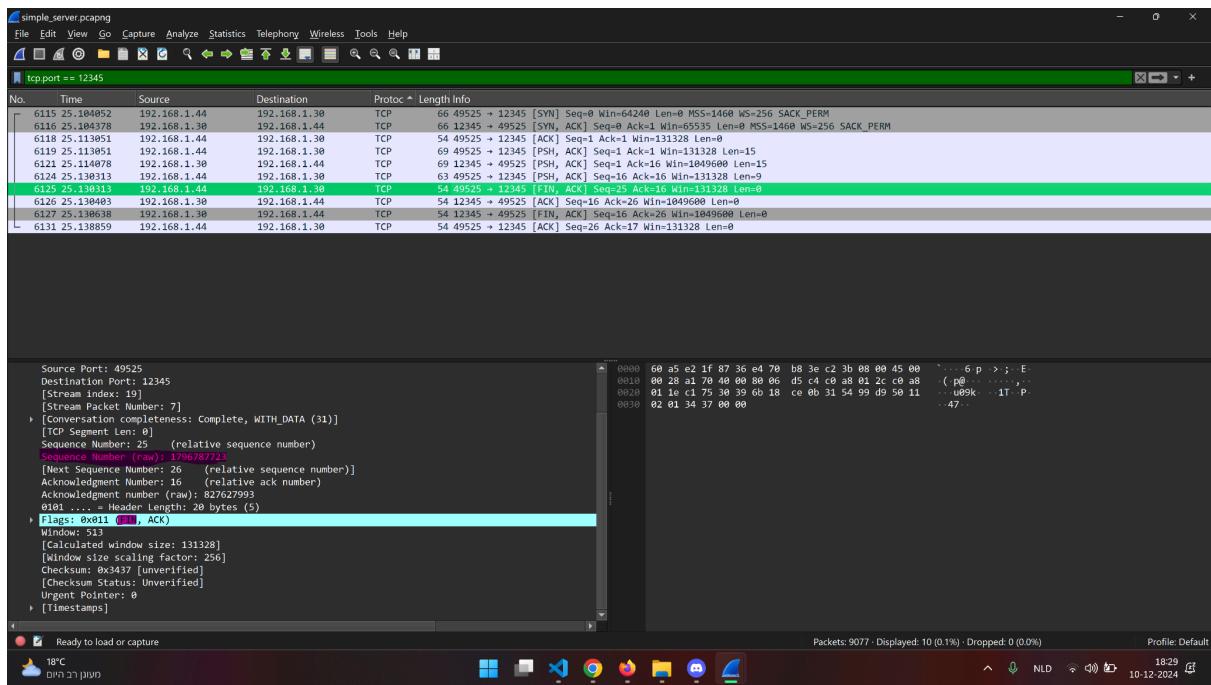
השרת מוחזר ACK כשהΝΑΚ נדל ב15 בתים, 1796787714 על כך שקיבל את השמות, ושולח אותם באמצעות גוזלות.



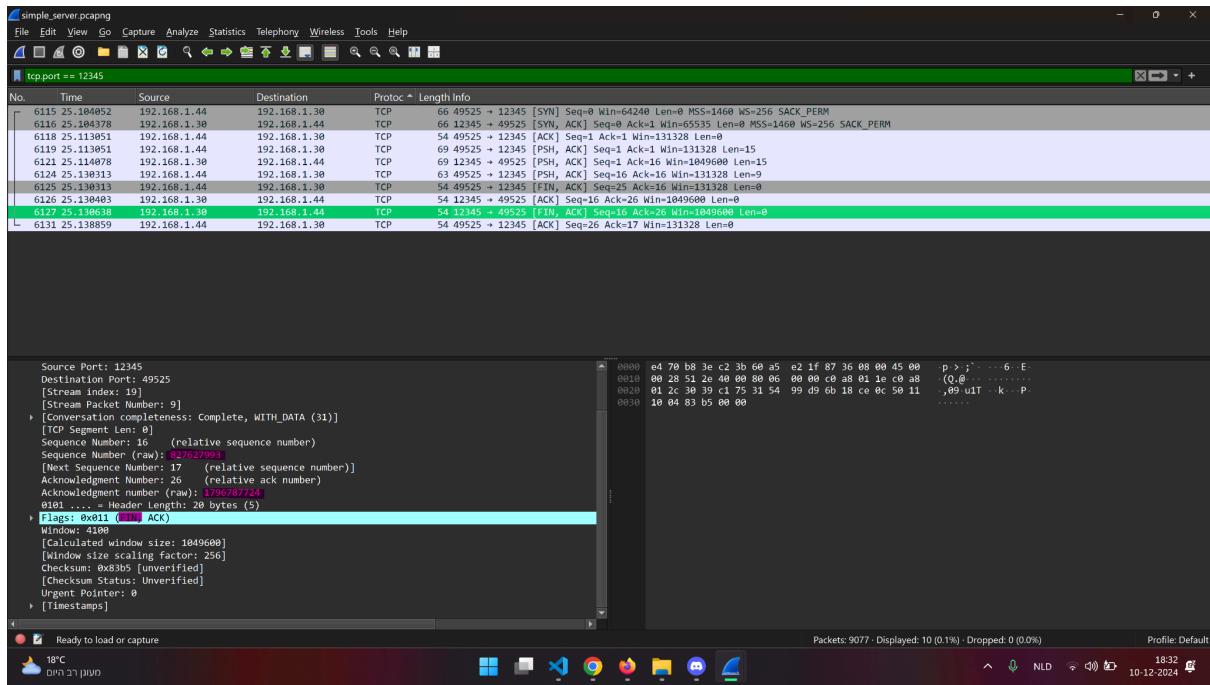
כעת הלוקו מסר שקיבל את השמות, ושולח את התז, ועל כן הNAK נדל ב15 ל-827627993.



כעת הלוקה מודיע שסימן לשלוח, ככלומר FIN, וمعدכן ב+9 עברו התז, וכן SEQn עלה .1796787723.



כעת שוב, השרת מודיע שקובל, הנקוד, נמדד ב1 בעבור ACKn, ולאחר מכן, מעדכן סיסים בעדרת FIN, הלוקה מעדכן שקיבל, ומעליה ב1 את ACKn אל 1796787724



וכמוון הלקוח מעדכן שקיבל את הסגירה.

cutת לחץ של ה17 וכוכו.

- בעבר 17:

- השרת

■ מקבל חיבור, מדפס שקיבל אותו, מקבל עוד חיבור וככל. עברו החיבור השני, מקבל 1024 בתים, מודיע שקיבל ומחזיר 7 בתים ראשונים. עברו החיבור הראשון, מקבל 1024 בתים, מודיע שקיבל, ומחזיר 5 בתים ראשונים. לאחר מכן, סוגרים את החיבור הראשון, ומנסים לבצע פעולה זו פעמיים.

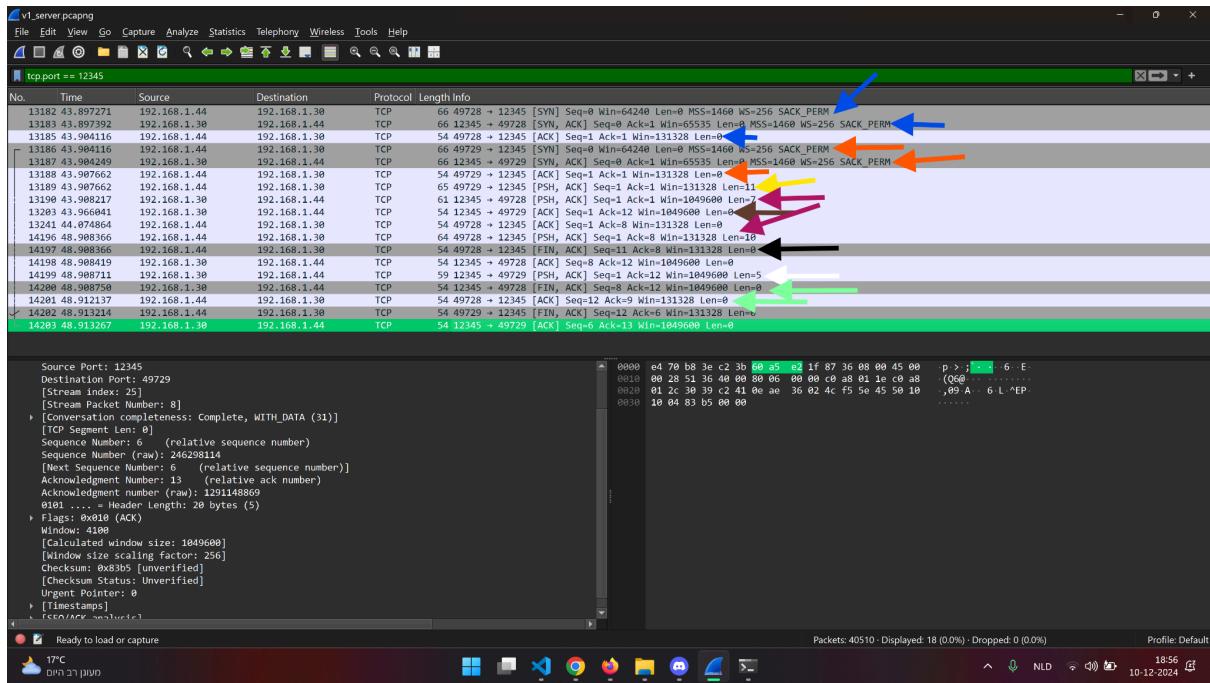
- הלקוח

יש לו שתי הודעות, נסמן M וM1. יוצר סוקט S1 ומתחבר אל השרת. יוצר סוקט נסף, S, ומחברו אל השרת. שלוח בזה האחרון את M. ישן 5 שניות, שלוח בS1 את M1, מקבל בחזרה מההשרת החזר באותו הסוקט. סוגר את S1. מקבל מההשרת שלח בS, וסגור את S.

- cutת לwireshark

■ הלקוח פותח סוקט (נסמן בקצרה) 8 בכחול, ומבקש להסתנכרן עם השרת. השרת מאשר את הקמת החיבור, ו8 מאשר שקיבל. cutת, הלקוח פותח סוקט נסף (נסמן בקצרה) 9 ובכחות, ושוב מבקש להסתנכרן, השרת מאשר ו9 מאשר שקיבל. cutת 9 שלוח לו את M בכחוב, השרת מחזיר אל 8 חלק מההודעה בוורוד, "so, Foo". מודיע 9 שקיבל ממנו את ההודעה. cutת 8 מודיע שקיבל את הורוד, ומבקש לשולח את M1. בנסף, הוא שלוח לשרת שישים בשחור. השרת מודיע שהוא קיבל, ושלוח אל 9 חלק מההודעה בלבד, (בר ורודות, כל מיני שיטויות של RTL אז לא יצא טוב

באנגלית) ומודיע ל8 שהוא סגור, 8 מאשר שקיבל (בירוק). 9 מבקש לסגור, והשרת מודיע שקיבל.



## בעבור 2/7:

השרות:

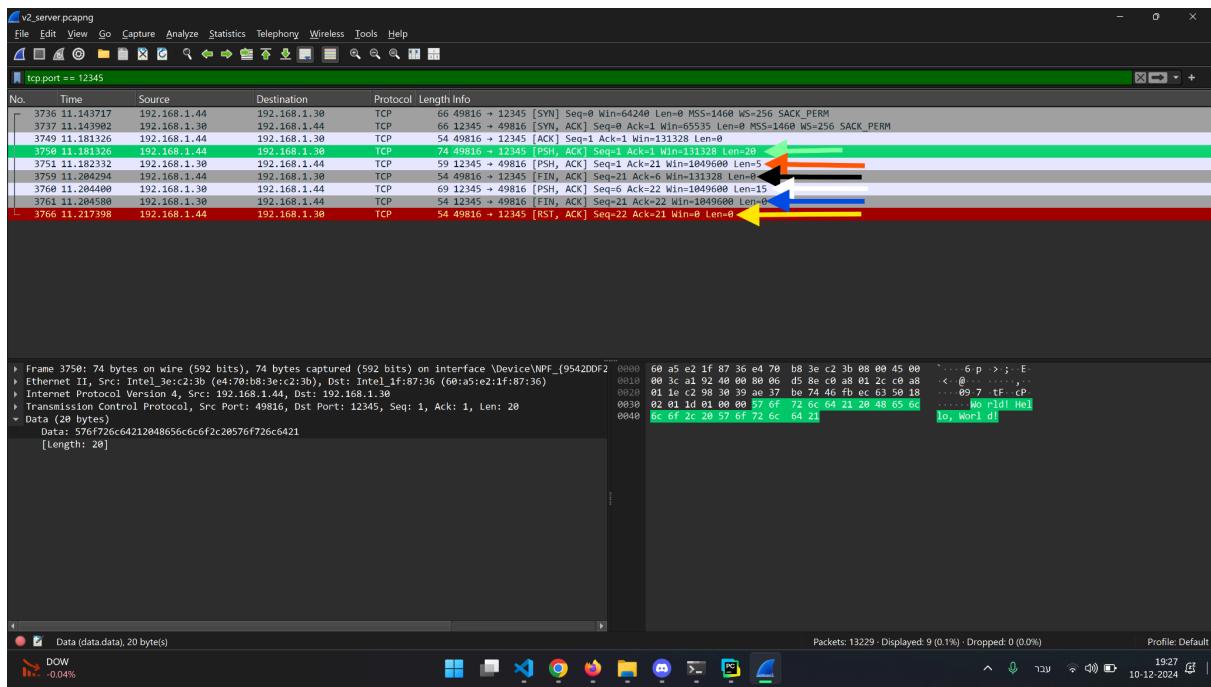
- השרות פותח סוקט ורץ בולאה אינסופית, מאשר חיבור חדש, וכל עוד יש בباءר מידע שהוא יכול לעלות לשכבות האפליקציה הוא מעלה 5 bytes
- ושולח אותם בחזרה ללקוח ב-**uppercase**. לאחר מכן עוד מידע שהוא מקבל, הוא סגור את החיבור וממשיר לחכotta להתחברויות נוספות.

הלקוח:

- הלקוח פותח סוקט ושולח הודעה 'Hello, World!' מעלה לשכבות האפליקציה את התשובה מהשרות, סגור את הסוקט ומדפיס את התשובה למסך.

**cut to wihark**

- שוב, הלקוח פותח סוקט, מבקש להתחבר, השרות מאשר והכל כרגע. הלקוח שולח את ההודעה (ירוק). השרות מחזיר ACK על הכל ושולח את ההודעה WORLD
- 5 בתים לפי האמור בקוד. הב哀ר בקוד בגודל 5 - لكن מועלה לשכבות האפליקציה הודעות בגודל 5 אר-TCP בשכבות הרשות מחזיר ACK על כל ההודעה שהתקבלה (כתום). הלקוח סגור ומודיע שישים לשולח (שחור).
- השרות מחזיר CUT בעפם אחת אבל שלוש הודעות שכל אחת בגודל 5, מודיע שישים (לבן+כחול). CUT מקבל RST כי הלקוח כבר סגר את עצמו (צהוב).



### • בעבר 37:

- השרת:

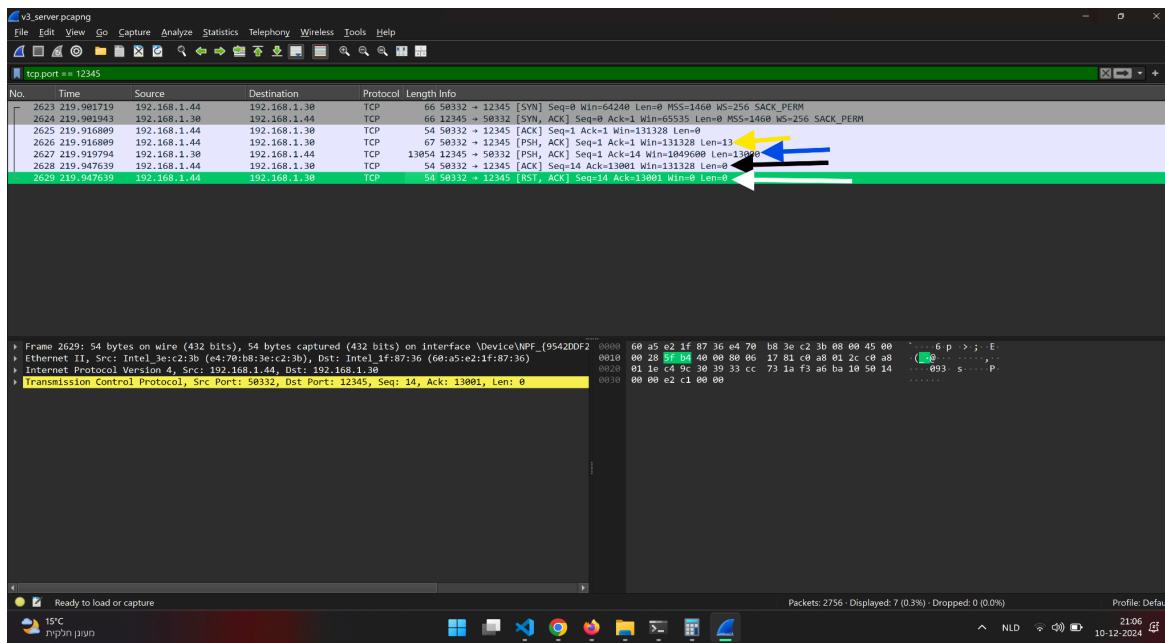
השרת פותח סוקט ומתחילה לróż בולולה אינסופית. הוא מאפשר חיבור של לקוחות וכל עוד יש הודעות שניתן להעלות לשכבות האפליקציה הוא מעלה עד 1024 bytes ומחזיר ללקוח את ההודעה כפول uppercase 1000 times. לאחר סיום שליחת ההודעות הוא סגור את הסוקט של הלוקו ומסHIR לחכות להתחברויות נוספות.

- הלוקו:

הלוקו פותח סוקט ושולח לשרת 'Hello world'! ולאחר מכן מבצע פעמיים receive של 1024 בתים לשכבות האפליקציה ומדפיס אותם ולאחר מכן סגור את הסוקט.

- CUTETTCP wireshark

מתחילה קרניל עם ack - syn-ack על מנת להקים חיבור TCP. בצד הלקוח ניתן לראות את ההודעה באורך 13 בתים שהלוקו שולח. לאחר מכן בצד השרת מוחזרה ACK 14 על ההודעה של הלוקו ומחזיר לו בונסף הודעה באורך 13000 זהה בדיקת WORLD HELLO!. לאחר מכן שוכתו בקוד השרת. בשחור ניתן לראות שהלוקו מוחזר ACK 10000 כמו שכתוב בקוד השרת. בצד הלקוח ניתן לראות שהלוקו קיבל את ההודעה בגודל 13000. לאחר מכן הלוקו נסגר בקוד ולכן שולח לשרת בשורה בלבד ACK RST. כמובן בלקוח קיבלו פעמיים 1024 בתים שהודפסו בגודל הביאר המוגדר בקוד ועל כן ישנו חלק גודל שהאפליקציה בחרה שלא לקרוא.



• בעבור 4:

- ## השראת:

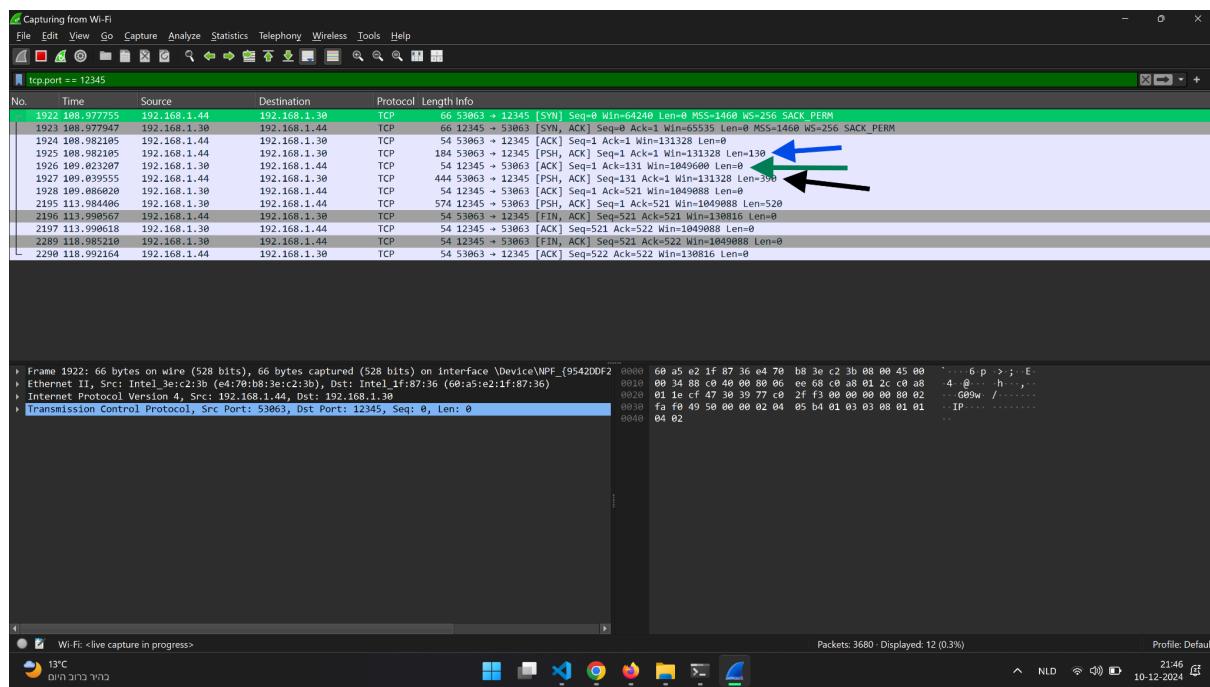
הشرط פותח סוקט ראשי ורץ בלולאה אינסופית. מאשר חיבור של לקוב  
ולאחר מכן ישן 5 שניות. מעלה לשכבת האפליקציה לכל היותר 1024  
בתים מדפיס אותם ומחייב ללקוח את ההודעה ב *upper case*. וחוזר לישון  
והעלות הודעות לשכבת האפליקציה. לאחר שאין יותר הודעות להעלות  
לשכבת האפליקציה השרת סוגר את החיבור עם הלקו ומחכה להתחברות  
של לקוב נספ.

## הלקוח:

הלקוח פותח סוקט כאשר ההודעה היא HELLO WORLD והוא מנדיר \*10>Hello World ושולח את ההודעה הנ"ל ב 4 SEND כלומר שהכ שולח לשרת 40 פעמים Hello World! לאחר מכן הוא מעלה לשכבה האפליקציה את התשובה של השרת וסגור את הסוקט.

## ויכוח wireshark

השורות הראשונות מייצגות את הריטוֹאַל הקבוּאַק - ack - syn על מנת ליצור חיבור TCP. הלוקה שולח את ההודעה כפול 10 בפעם הראשונה והיא נשלחת לשרת (כחול). השרת מחזיר ACK בירוק שקיבל את ההודעה. לאחר מכן נשלחים כל שאר ה-SEND בבת אחת לשרת (בשחור). בשורה הבאה השרת מחזיר ACK שקיבל גם את כל ההודעות הנ"ל. בשורה הבאה השרת שולח את ההודעות בבת אחת בחזרה על כולם בsuper case (לבן). ומסיים את החיבור ע"י כך שהלוקה שולח דגל FIN והשרת עונה לו ACK. והפוך ממשיכים על FIN של השרת. הסבר אפשרי הוא שבהתחלת אין פקטות YET THE FLY ON لكن השרת מיד שולח את ההודעה הראשונה. ולאחר מכן כיוון שיש פקטות YET THE FLY ON לא נשלח ישר כל הודעה ולכן רואים אותו בסוף נשלחים בבת אחת. השרת לא ישר עונה על ההודעה הראשונה שקיבול ונכנס ל-5 שניות של שינוי ולאחר מכן כבר כל שאר ההודעה הניעה וכן הוא שולח בבת אחת את התשובה שלו על כל מה שהגיעה.



הערה לקריאת הסוף: הרצינו את כל הקודים כמה פעמים אך לאחר הריצה אחת עיצרנו את hark.h wireshark שכן אחרת הינו הולכים לאיבוד בתוך כל ההודעות בין אותם מחשבים. חשוב לציין שלא היה אצלנו ספציפיות שינוי בהתנהנות בין הריצות של אותו קוד.