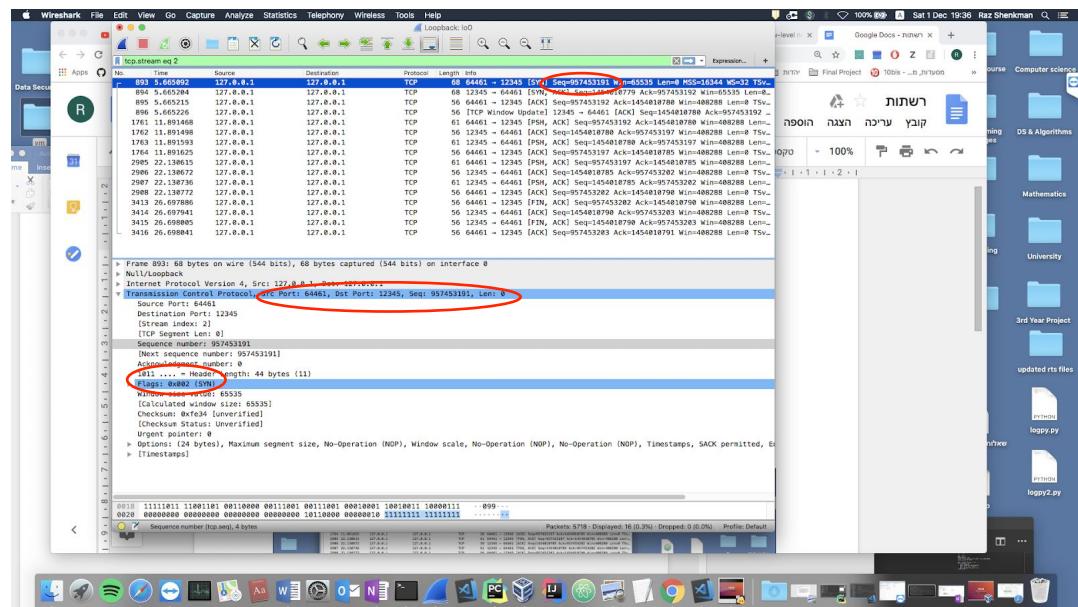


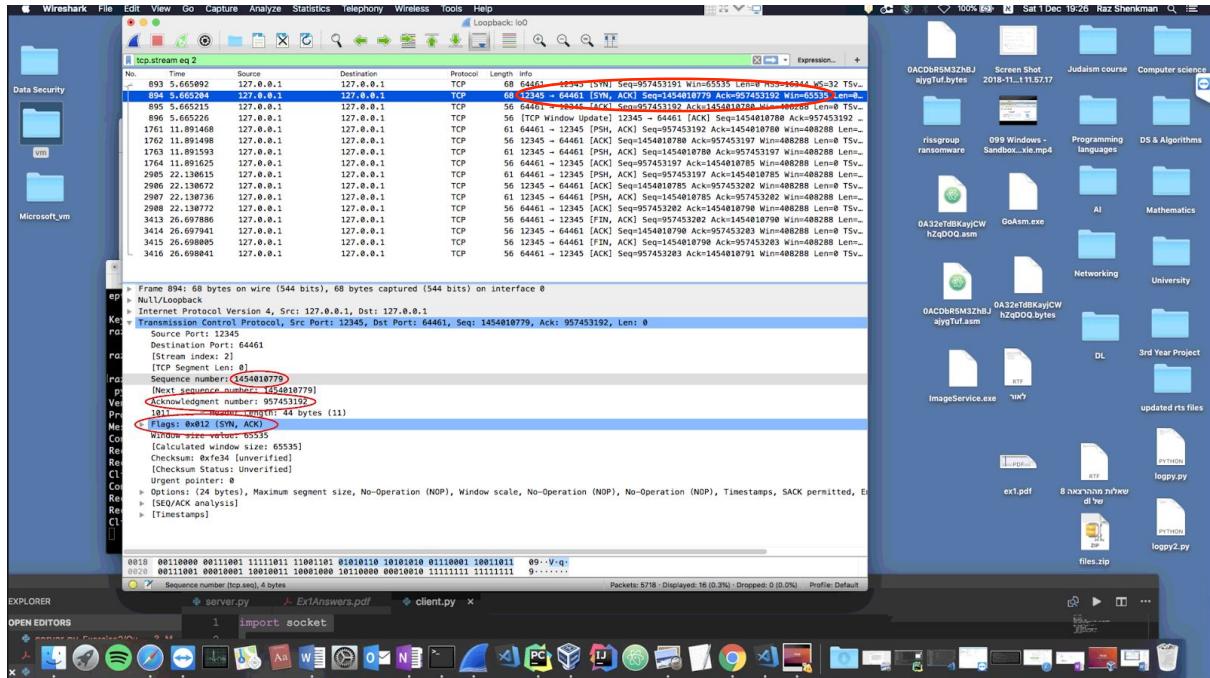
## רשותת תרגיל 2

(1) א'

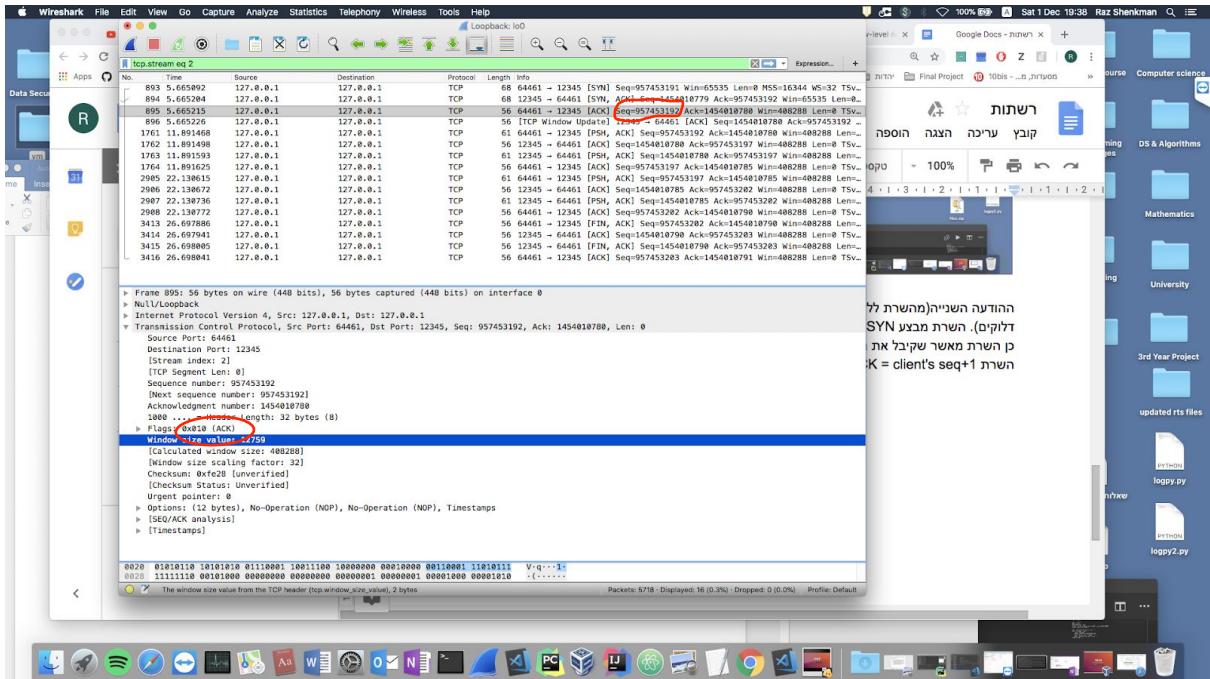
### הלקוח הראשון:



ההודעה הראשונה (הלקוח לשרת) היא בקשת chs, נשים לב שהיא מלאה ב chs flag (מוקף באדום), שבעצם מציין שהלקוח רוצה ליצור קשר עם השרת. נשים לב שללווה sequence number ראשוןי שנקבע ע"י הלקוח (מוקף באדום).

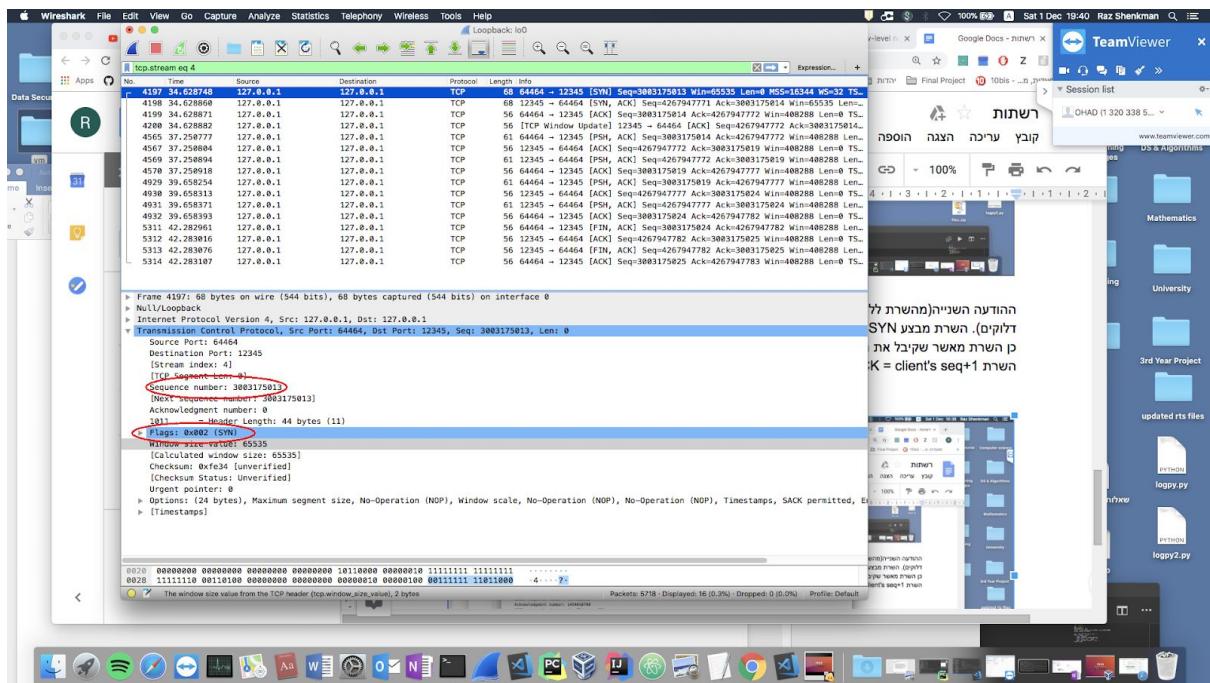


**ההודעה השנייה(המשרת ללקוח)** היא גם הודעת SYN וגם הودעת ACK(כמו שניתן לראות שני הדגמים האלו דלוקים). הרשת מבצע SYN - הוא קובע seq number משלו (שינויו של הלוקח) ושלח ללקוח. כמו כן הרשת מאשר שקיבל את ה seq number הראשון של הלוקח באמצעות ACK. ניתן לראות שבהודעת הרשת, ACK = client's seq+1, כלומר הרשת מאשר שקיבל ערך זה.

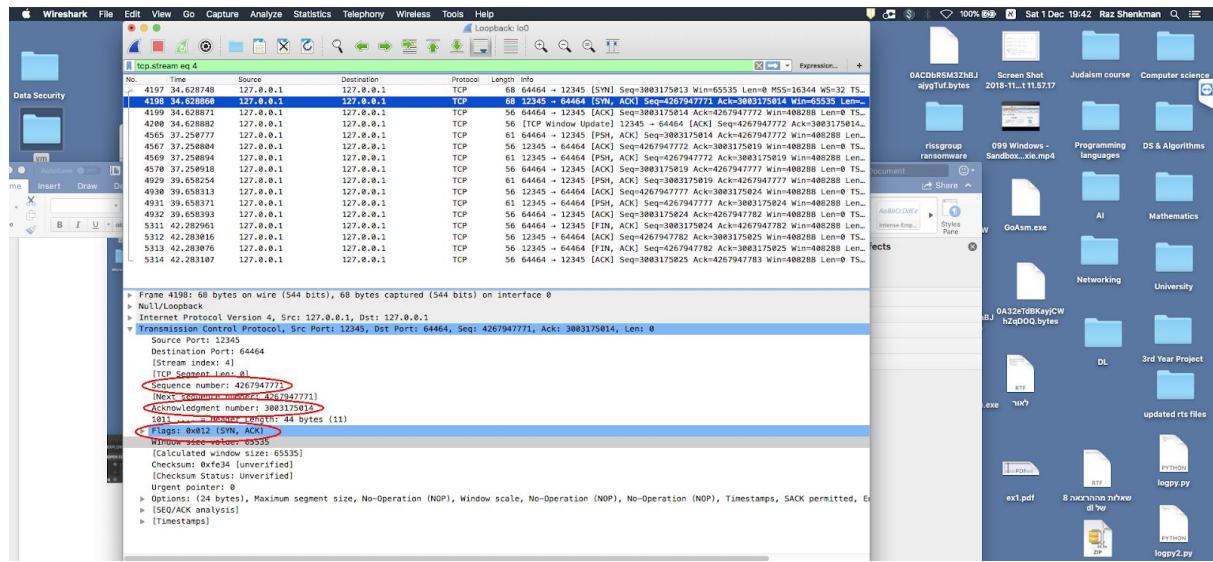


כאן זו הودעה מהלкова לשרת שהיא הודעת ack (רואים שהוא הדגל היחיד שדלוק), כאן ה seq הוא אותו seq של הלקוח ממוקדם (+1 עבור החעט הראשון) והו ackseq של השרת (עבור ה syn של השרת).  
כאן בעצם מסת'ים תהליך handshake, הלקוח אישר את ackseq של syn של השרת.

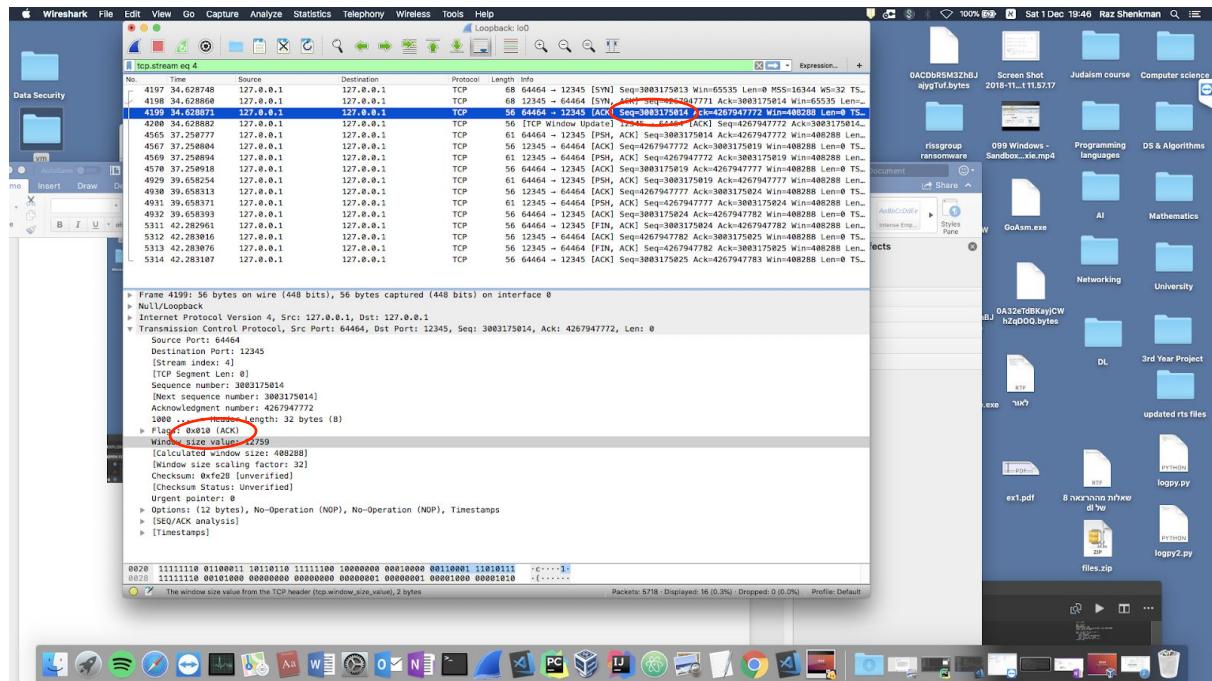
## הלקוח השני:



ההודעה הראשונה(הלקוח השני לשרת) היא כמו שניתן לראות בדגלים הודיעת SYN. הלוקו קבוע בseq number התחלתי(משמעותו גם הוא) ושלח לשרת.



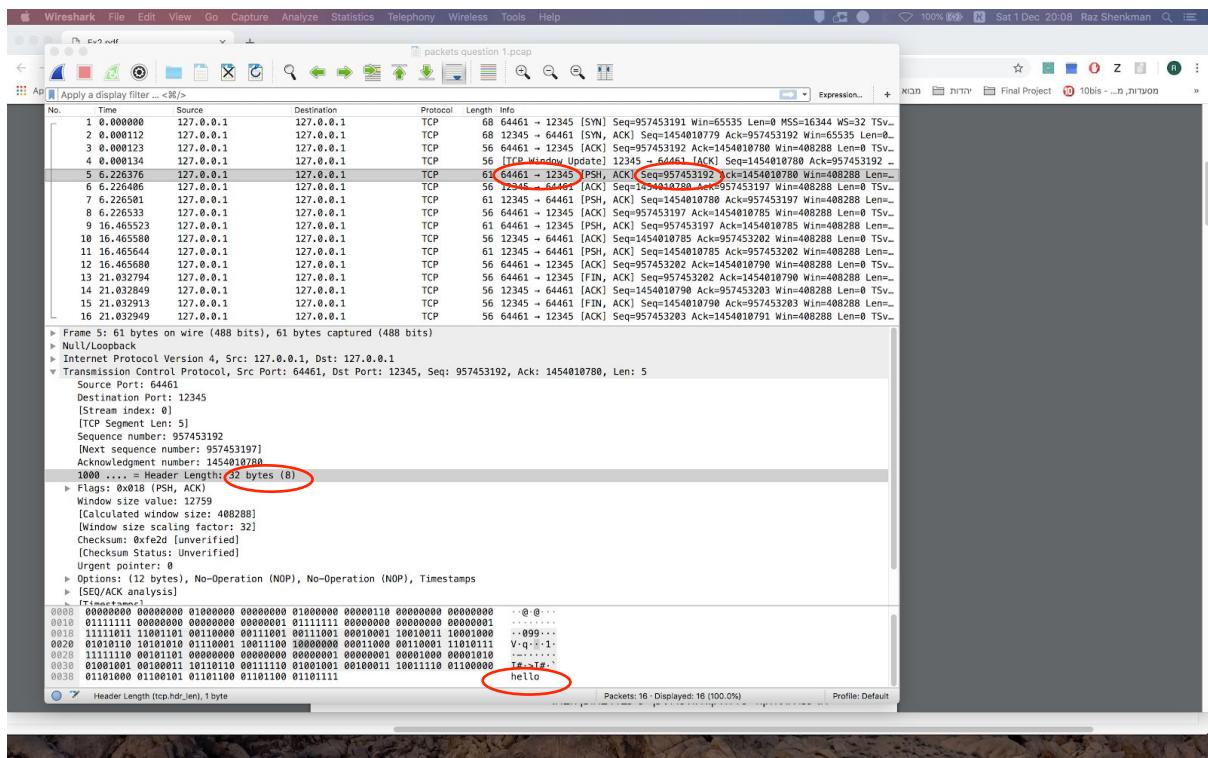
**ההודעה השנייה בתהליך מהשרת ללקוח השני** היא גם הודעת SYN וגם הודעת ACK (כמו שניתן לראות שני הדגמים האלו דלוקים). השרת מבצע SYN - הוא קובע num seq number משלו (שינויו של הלוקוט) ושולח בהודעה זו ללקוח השני. כמו כן השרת מאשר שקיבל את num seq number הראשון של הלוקוט השני באמצעות ACK. ניתן לראות שבה Hodutut השרת מאשר ACK = second client's seq+1 num seq+1 ACK = second client's ACK.



כאן זו הودעה מהל��ון לשרת שהוא הדגל היחיד שדלוק, כאן ה seq הוא אותו seq של הלוקון מוקדם (+1 עבור החען הראשון) והכו ackseq של השרת (+1 עBOR החען syn של השרת).  
כאן בעצם מסתiem תהליך handshake, הלוקון אישר את ack syn של השרת.

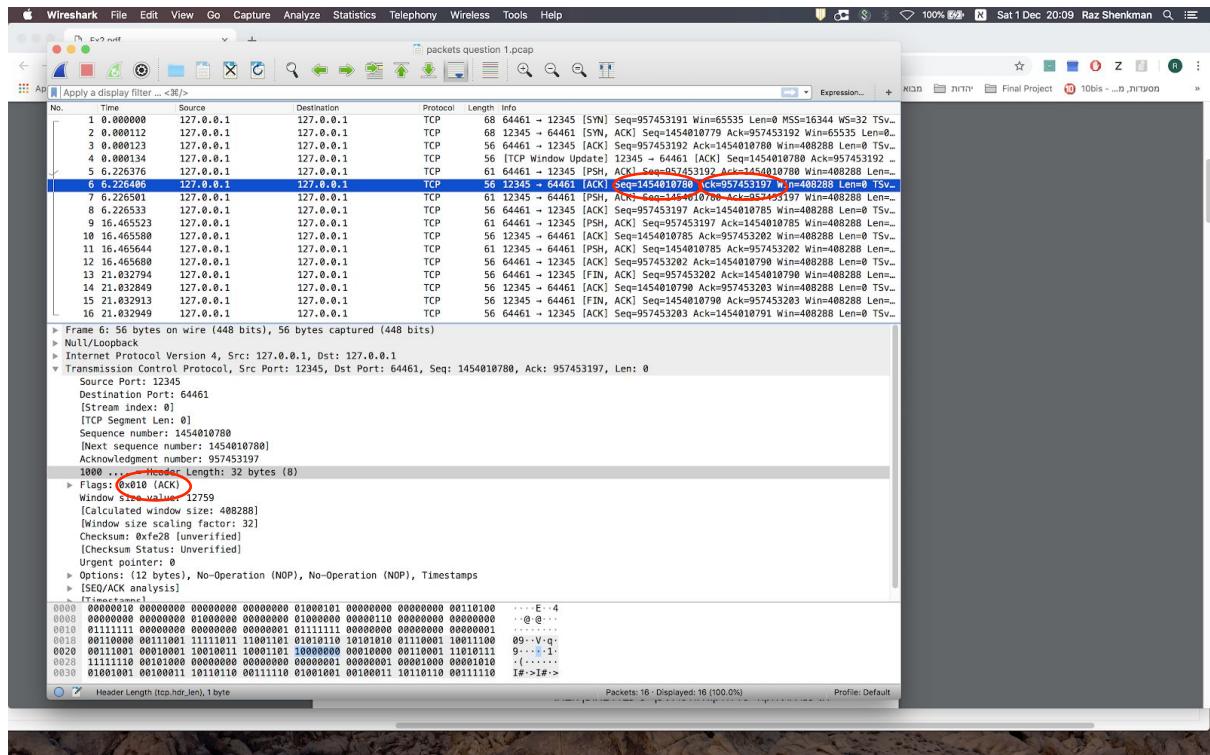
(1) ב'

**בלוקות הראשונים**  
**הודעת hello מהליקות לשרת:**



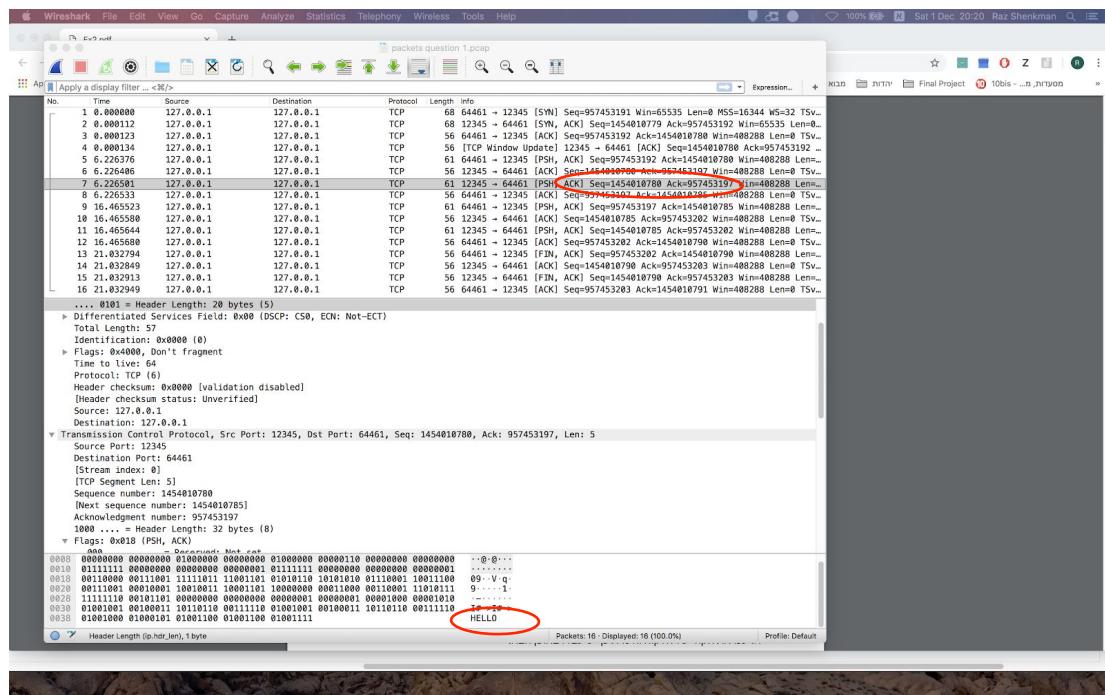
נשים לב לפורט של הליקות source port (64461) ופורט השרת destination port (12345) והוא 12345. נקבע ע"י מ"ה (sequence number) הוא 957453191 והאך (acknowledgment number) הוא 957453192. נשים לב שגודלו של הSEQUENCE NUMBER הוא 32 bytes (hello).

### כאן השרת שולח ללקוח ack:



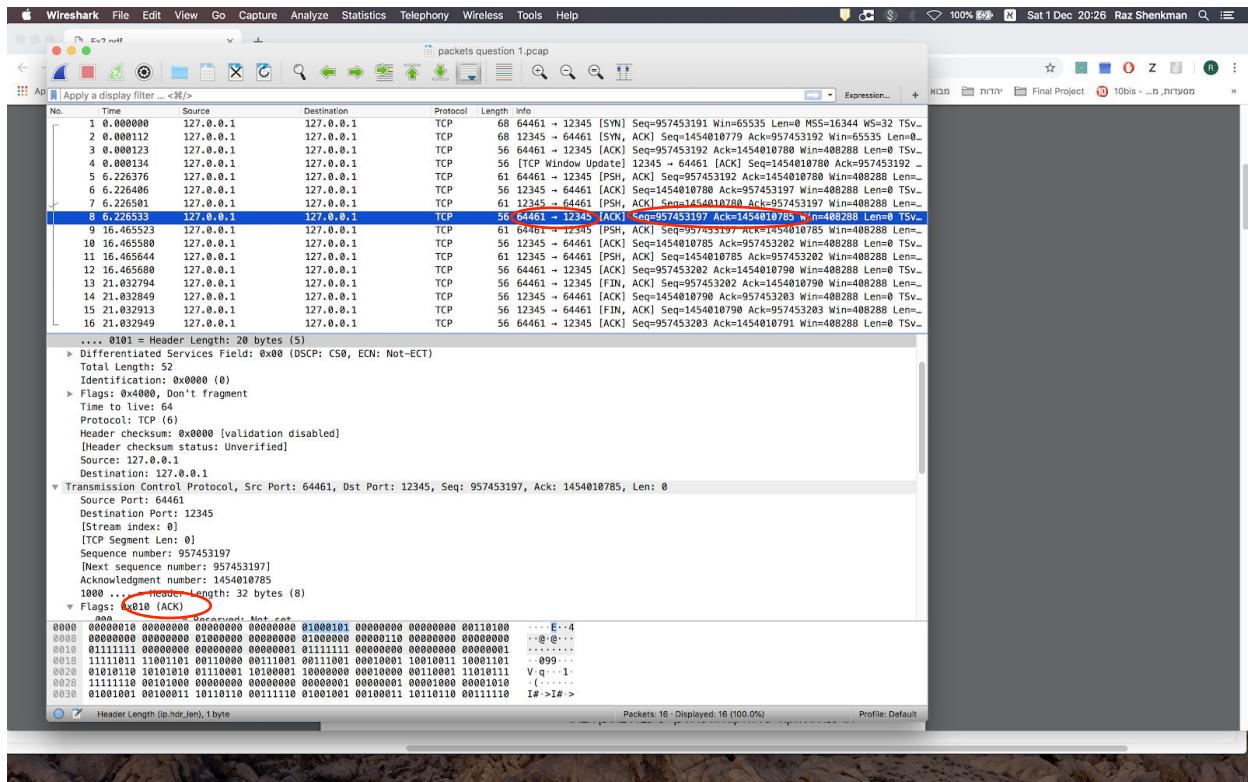
נשים לב שכן השרת (פורט 12345) שולח ללקוח (פורט 64461) הודעת ack (כיוון שדגל ack דלוק), גודל הheader tcp הוא 32 bytes והו seq number הוא 0. הודהה השםsequence number הוא ack number ב**תוספת 5** (כמה השרת מאשר ללקוח שהוא 5 bytes ממנו).

## השרת שולח ללקוח HELLO



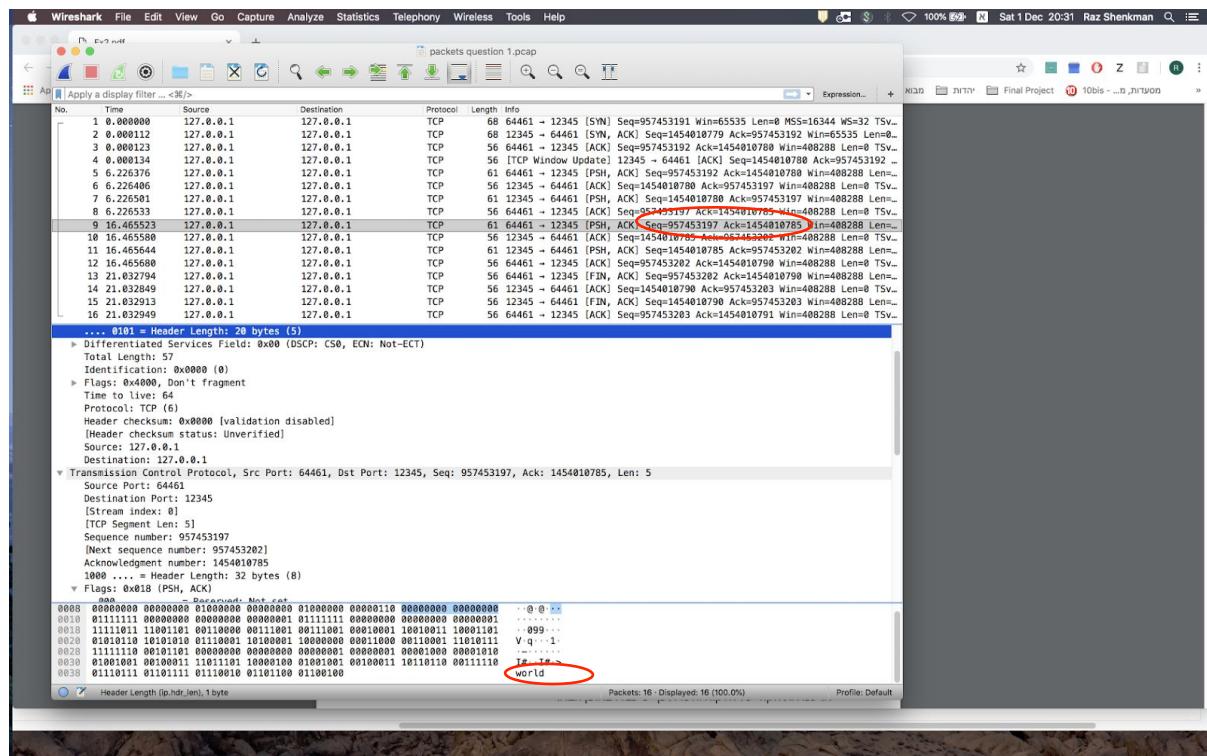
נשים לב שכן השרת (פורט 12345) שולח ללקוח (פורט 64461) הודעה שהມידעה בה הוא HELLO, גודל ה sequences הוא 5 bytes (HELLO) והוא seq number (HELLO) של ההודעה הוא 5 (נשלח bytes 32 tcp header number) והוא seq number של הלקוח הקודמת (כי ההודעה הקודמת של ack לא משנה את seq number) והו seq number של הלקוח שנשלח ממוקדם .5+.

### הלקוח שלוח ack לשרת:



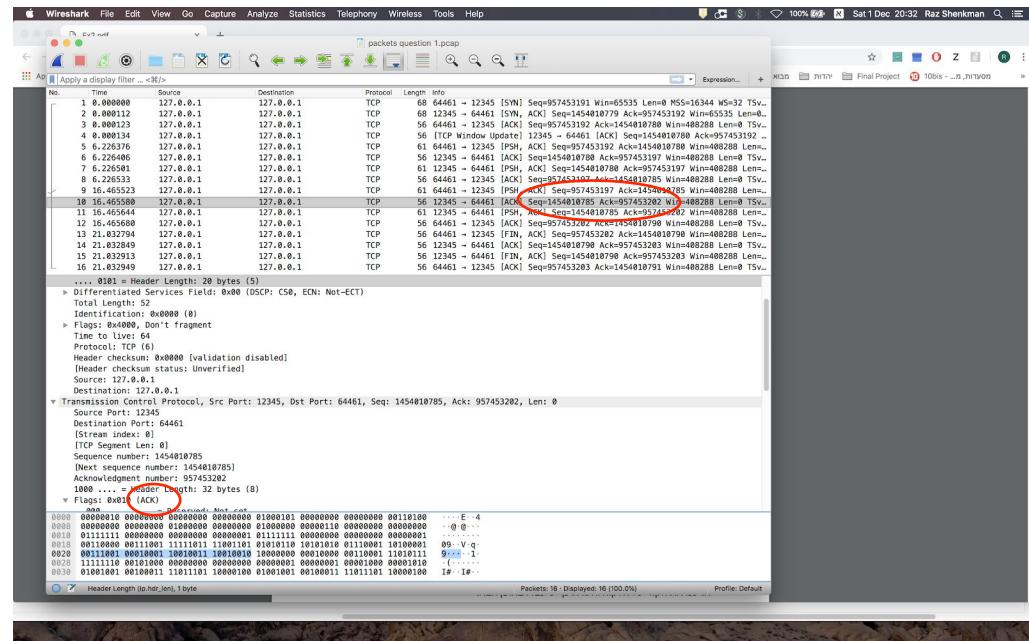
כאן הלקוח (פורט 64461) שלוח לשרת (פורט 12345) הודעת ack (12345) נשים לב שההוּהא של ההודעה הוא 0, והseq number הוא 12345. מתקיים שהഗודל header bytes הוא 32 tcp header bytes, והseq number הוא 12345. ack flag של ack + 5 (כי הוא קיבל אישור על השילחה שלו, אך העלה את seq number ב-5), בנוסף הנקודם של הלקוח + 5 (כי הוא קיבל אישור על השילחה שלו, אך העלה את seq number ב-5), בנוסף הנקודם של הלקוח + 5 (כי הוא קיבל אישור על השילחה שלו, אך העלה את seq number ב-5), בנוסף הנקודם של הלקוח + 5 (כי הוא קיבל אישור על השילחה שלו, אך העלה את seq number ב-5), בנוסף הנקודם של הלקוח + 5 (כי הוא קיבל אישור על השילחה שלו, אך העלה את seq number ב-5), מוסיף (5).

## **שליחת world מהלך לשרת:**



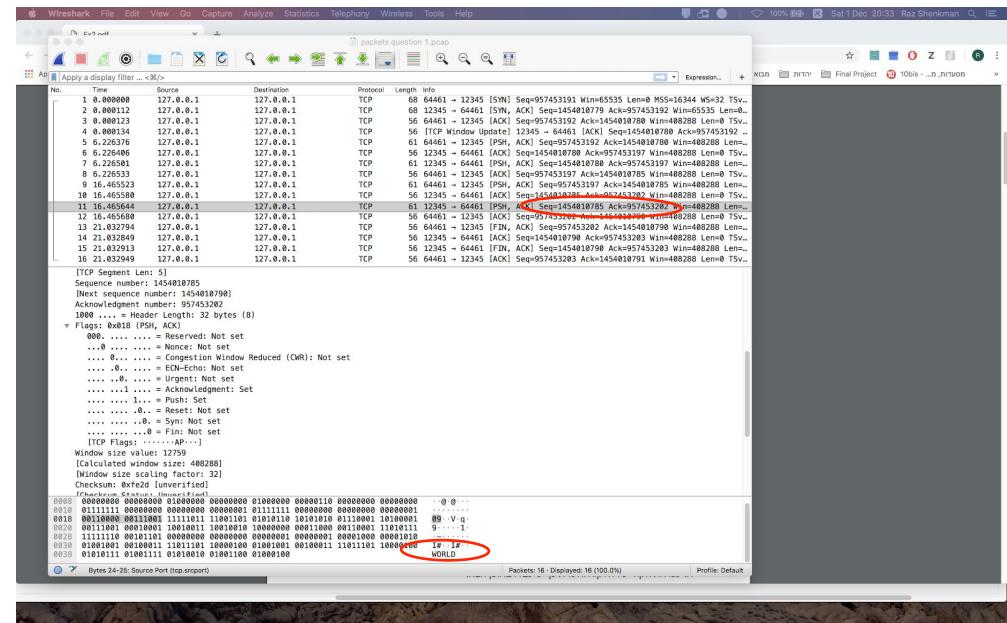
נשים לב לפורט של הלקוח source port (נקבע ע"י מ"ה) ופורט השרת destination port (12345).sequence number הוא בעצם הערך של הלקוח מההודעה הקודמת שלו. נשים לב שגודל tcp header הוא 32 bytessequence number של השרת (מההודעה הקודמת שלו). נשים לב שגודל header הוא 32 bytes. כפי שכתבו. אורך ההודעה הוא 5 bytes (המילה world).

## הודעת ack מהשרת ללקוח:



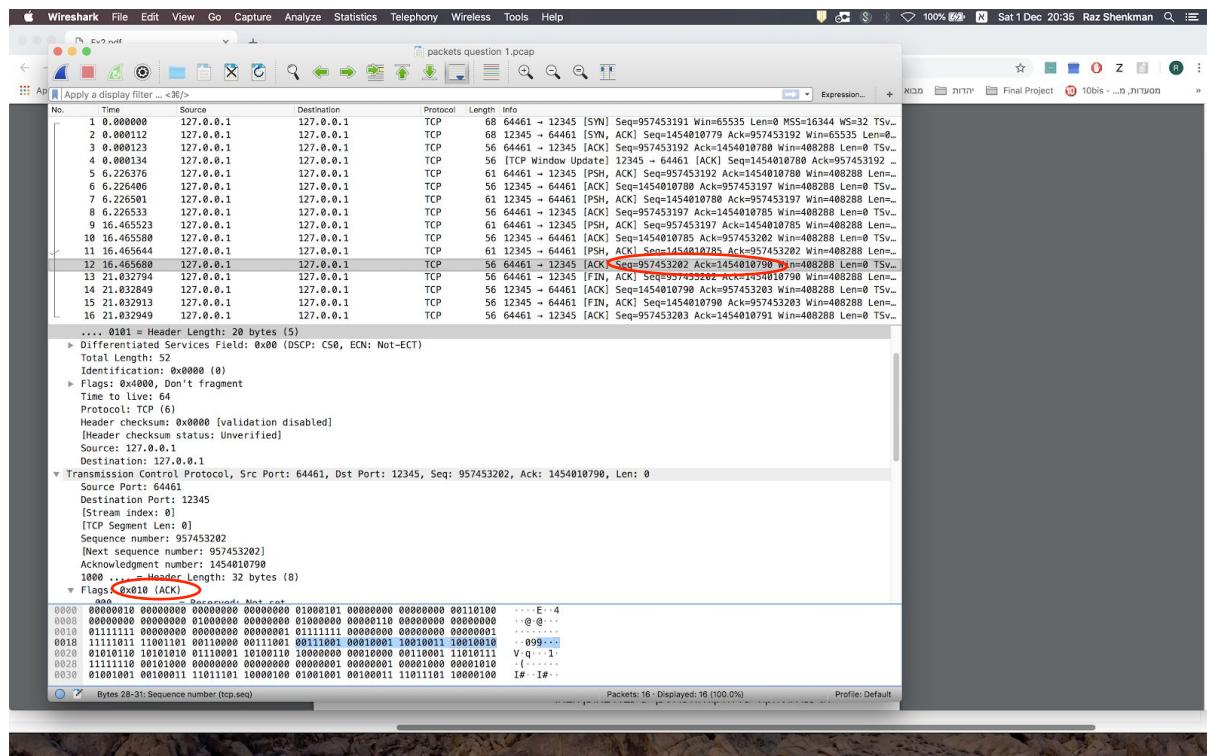
נשים לב לכך השרת (פורט 12345) שולח ללקוח (פורט 64461) הודעת ack (כיוון שדגל ack דלוק), גודל header הוא 32 bytes והlength של ההודעה הוא 0. הערך seq number הוא הsequence number של השרת מההודעות הקודומות, והack number הוא seq number של הליקו **בתוספת 5** (ככה השרת מאשר ללקוח שהוא קרא 5 bytes ממנו).

## הודעת WORLD מהשרת ללקוח:



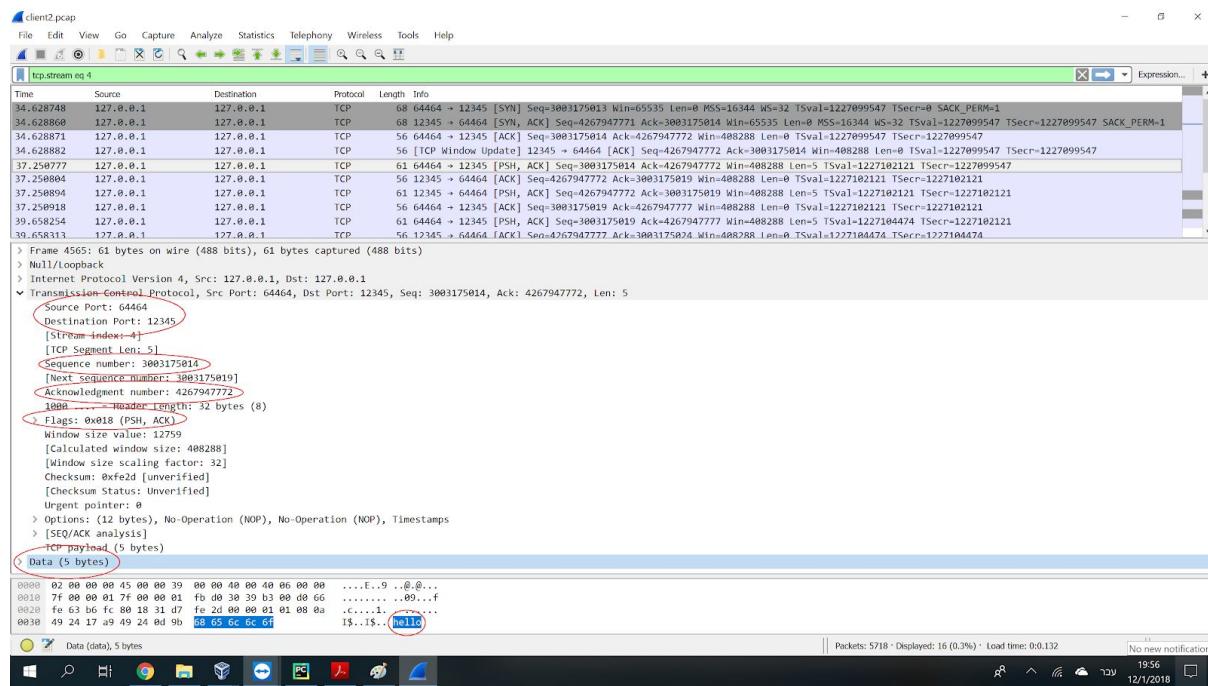
נשים לב שכן השרת (פורט 12345) שולח ללקוח (פורט 64461) הודעה שהມידע בה הוא WORLD, גודל הלקוח הוא 5 bytes (WINDOW\_SCALING), והוא מציין שהחלון הוא 5 (WINDOW\_SIZE\_SCALING). הלקוח י��ה מה הודעה הוא WORLD, כי בSEQUENCE NUMBER הוא מוסיף 5.5.

## הודעת ack של הלקוח לשרת:

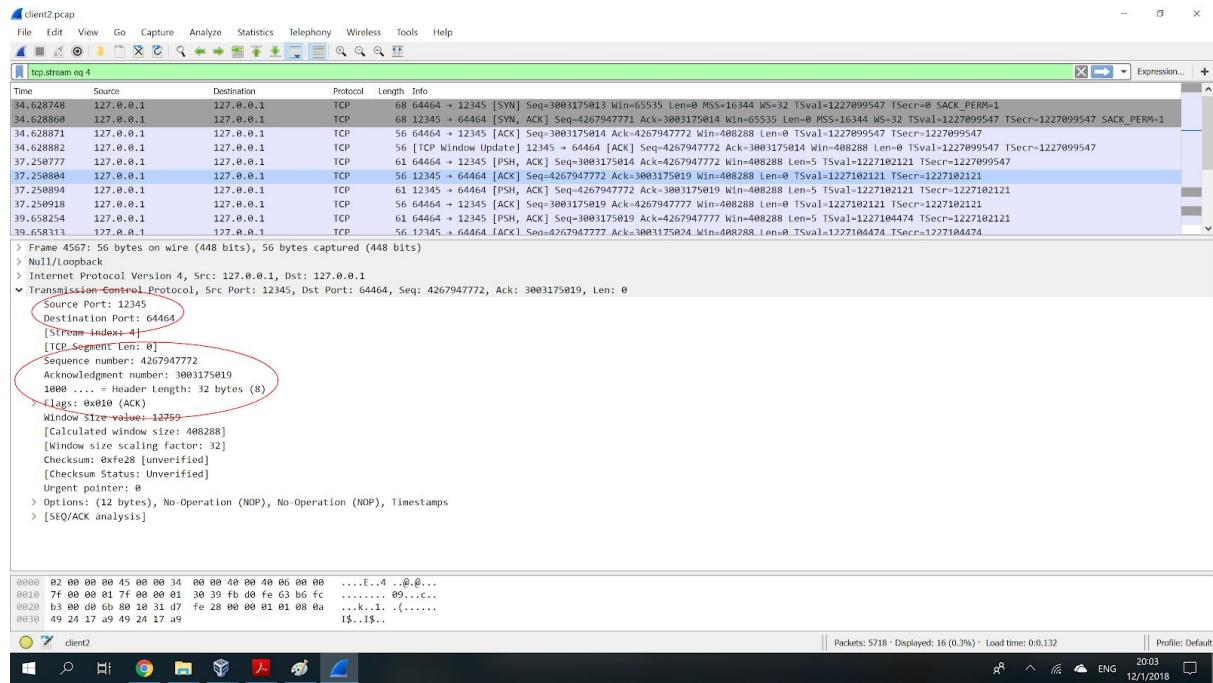


כאן הלקוח (פורט 64461) שולח לשרת (פורט 12345) הודעת ack (12345) (נשים לב שהlength של ההודעה הוא 0, וזה seq של ack דילוק). מתקיים שהגודל header bytes, וההוגדר numbeRNumbR הוא numbeRNumbRseq. הflag ack מוקדם של הלקוח + 5 כי הוא קיבל אישור על השילחה שלו, וכן עלה את numbeRNumbRseq ב-5, בנוסף numbeRNumbRseq הוא numbeRNumbRseq של השרת, ולכן מוסיף 5.

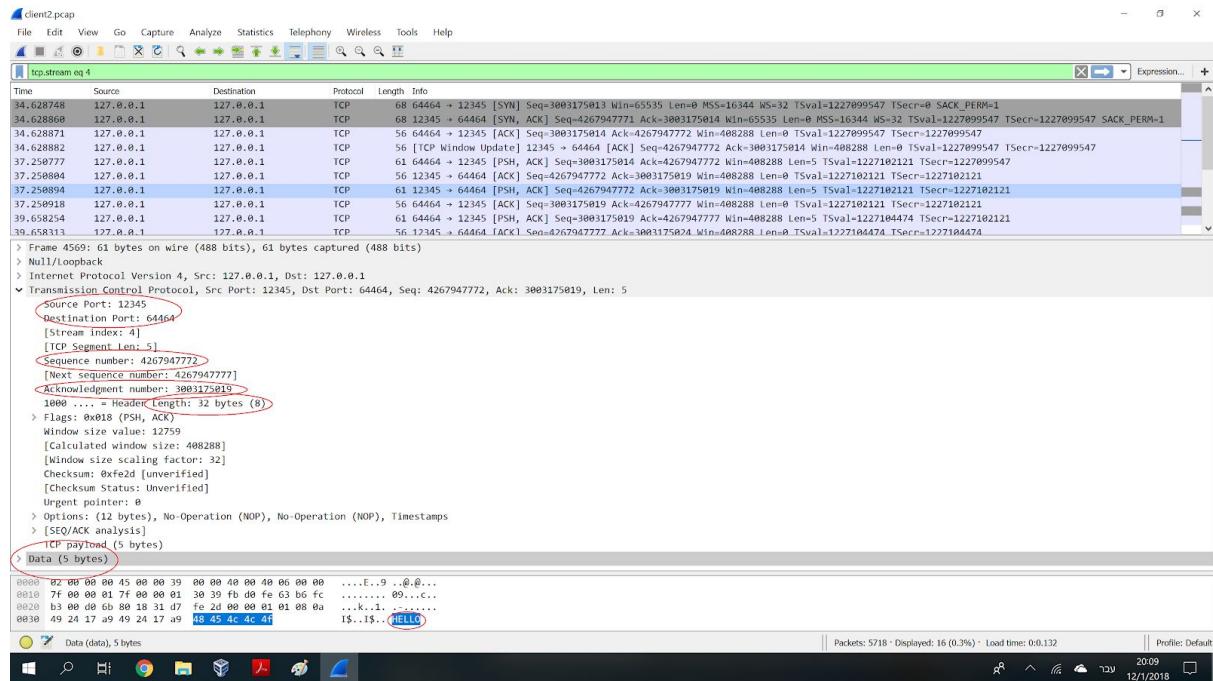
## בלוקו השני:



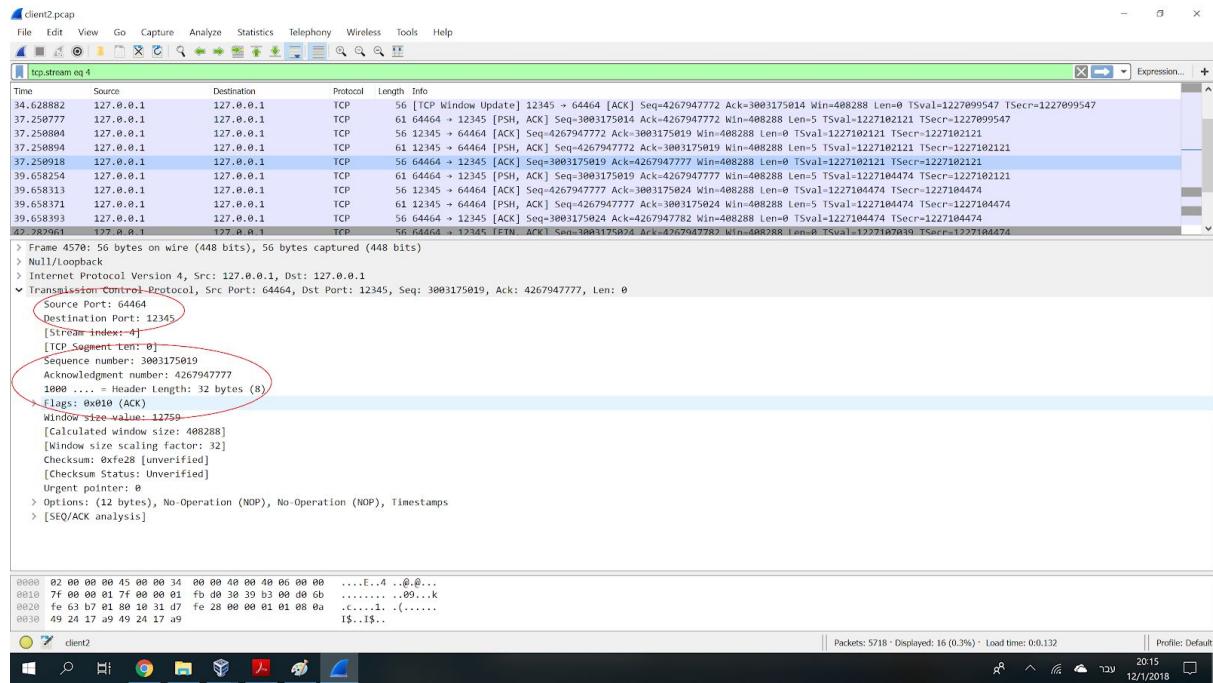
בהודעה זו הלקוח שלח לשרת hello (מוסומן למטה). נראה שהdagל של ה-ack דלוק והלקוח ביצע ACK לא seq ההתחלה(פלוא אחד) של השרת(הרי השרת לא שלח עוד מידע ולכן אצל הלקוח לא התקבל עוד מידע ממנה), ו-ה seq number זהה ל seq number ההתחלה של הלקוח(שעדין לא שלח מידע) פלאו אחד. ניתן לראות שה포רט של היעד(השרת) הוא 12345 כמו שבחרנו. הפורט של המקור(הלקוח) שנבחר לו ע' מערכת הפעלה הוא 64464.



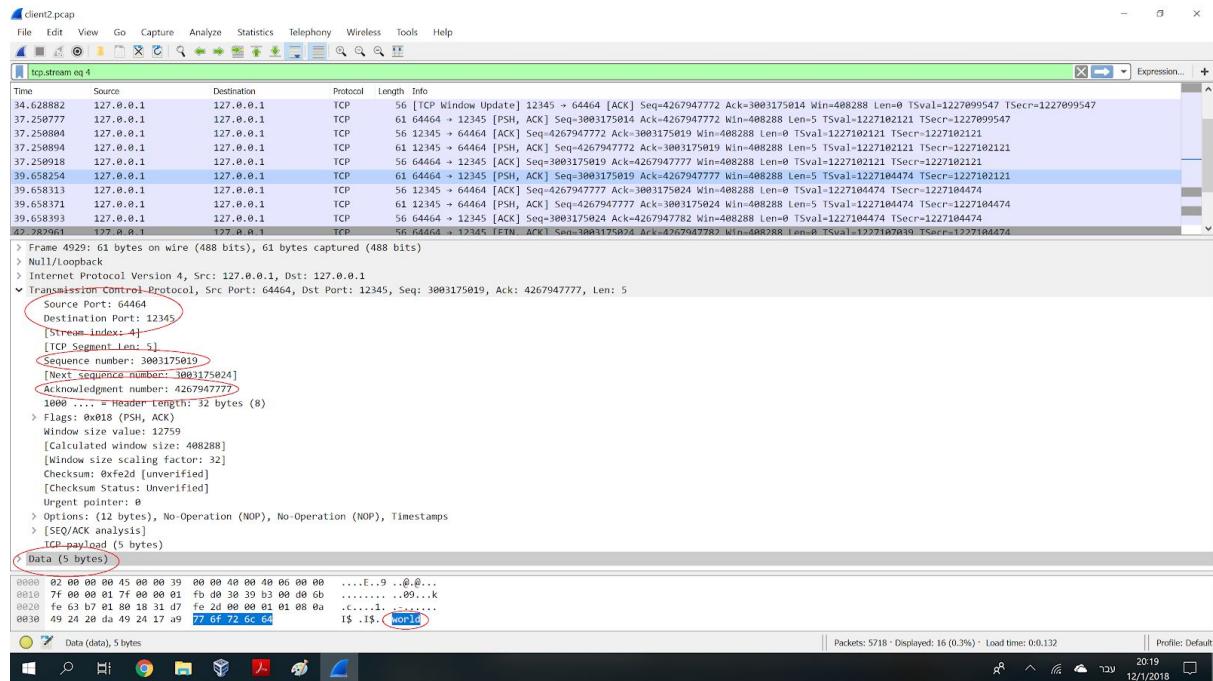
**ההודעה הבאה התקבלה מהשרת ללקוח**, ניתן לראות שפורט המקור שלה זהה לפорт של השרת ופורט היעד זהה לפорт של הלקוח. ההודעה היא הודעת ACK(הציג דלוק) על הודעת hello שנשלחה מהלקוח - קל לראותות ש ה-ACK number זהה ל- seq number+5 (הר היידעה Hello בגודל 5) כולם ההודעה התקבלה במלואה אצל השרת. הר היידעה עוד לא שלח מידע.



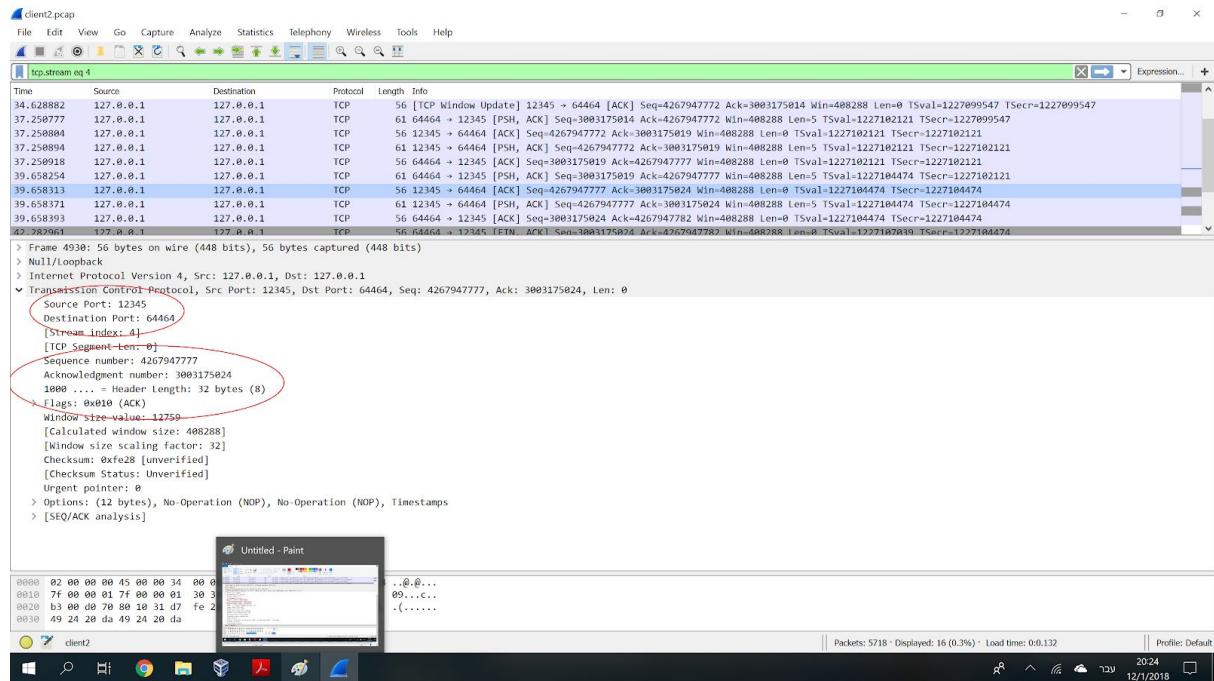
הודעה זו (לפי הפורטים מהשרת ללקוח) היא ההודעה בה השרת כותבת **HELLO** ללקוח. ה **seq number** נשאר זהה למה שהיota בזמן הקמת החיבור כי רק עכשו השרת מתחילה לשלחן מידע (וכשישלח המודיע הבא הוא יתחל מ ה **seq** הבא, שרשום מתחת ל **seq** הנוכחי). ה **ACK** זהה ל **ACK** מהחביבה הקודמת(למקרה שהלקוח לא קיבל את **ACK** הקודם, אז הוא יוכל לקבל אותו בעודם ולדעת שהשרת כן קיבל את המידע שלו עד ל **client initial seq number +5**)



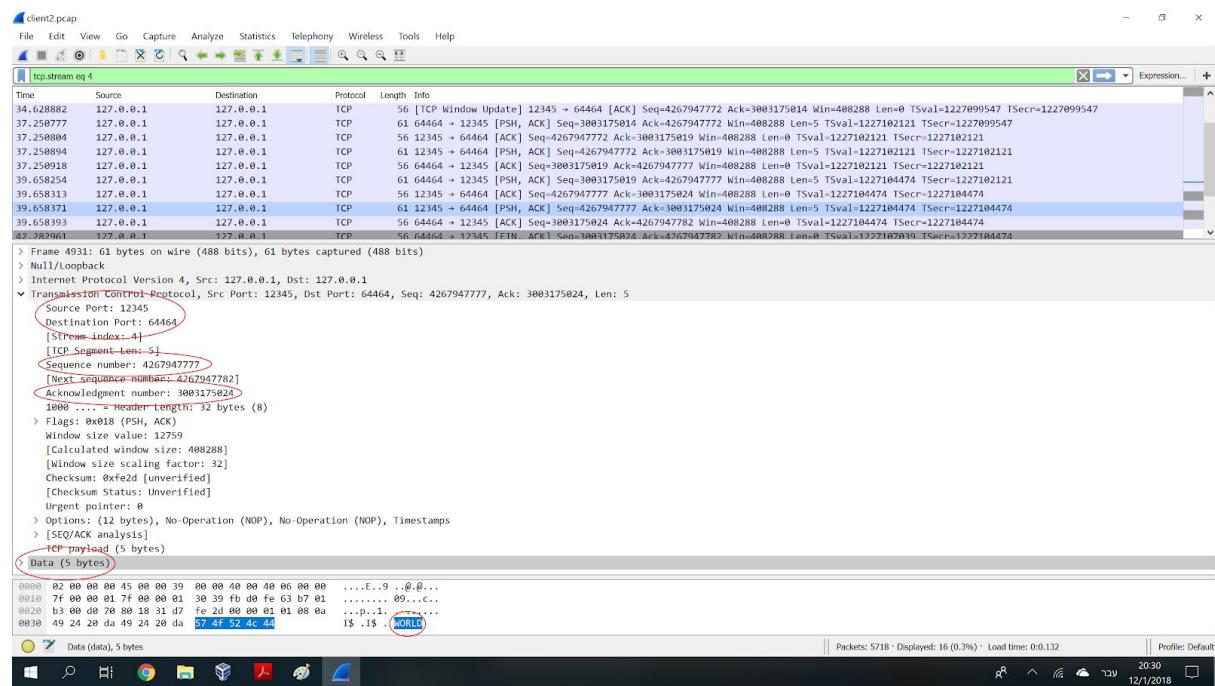
**בהתודעה זו (מהלך לשרת, כאמור לפי הפורטים), הלוקו שולח ACK לשרת על ההודעה Hello (התקבלה ניתן לראות שדגל ה ACK דלוק) ו ה ACK number שווה ל 5 + initial seq number של ה-server (וההודעה היא מוגדלת 5 שכן זה אומר שההודעה התקבלה במלואה אצל השרת). ה Seq number עלה בחמש + 5 (client's initial seq + 5) כי הלוקו מתחילה שישלח Hello.**



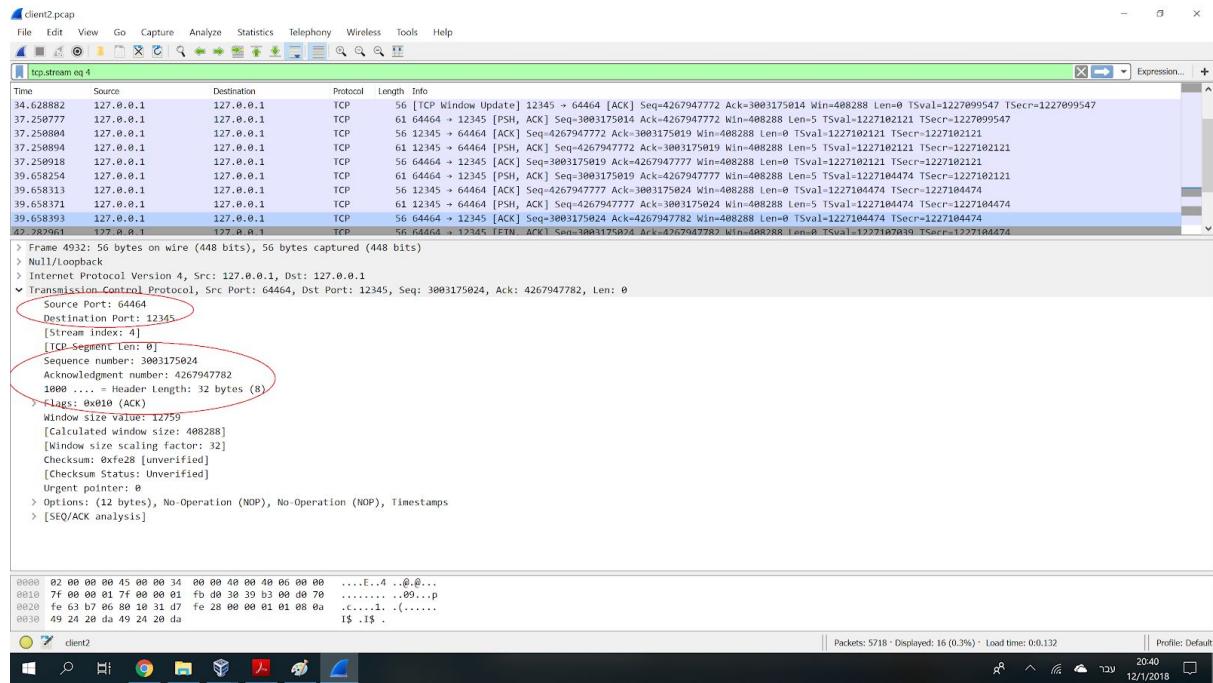
הודעה זו (**לפי הפורטים**) היא מהלך לשרת, ומכליה את המידע המוסף `world`. כמובן שאין שינוי הגיע בseq שווה ל 5 + initial client seq + 5 bytes (כלומר ההודעה הבאה אחראית חמשת התווים של `hello` שנשלחת כרגע) ואין שינוי בACK כי לא התקבל עוד מידע מהשרת.



הודעה זו היא מהשרת ללקוח (פורט המקור 12345 הוא של השרת). זהה הودעת ACK (הדגל דלוק כמובן) על הודעתה world של הלקוח - ניתן לראות שהACK גדול ב-5 מ-הseq number שנשלח בהודעתה world מהלקוח כלומר ההודעת התקבלה אצל השרת במלואה. הseq number של השרת עלה בחמש מאז החבילה הקודמת שלו כי מاز נשלח (והתקבל גם) הודעתה HELLO.



הודעה זו מהשרת ללקוח(ניתן לראות **לפי הפורטים**) מכילה את המילה WORLD כי שnitן לראות למטה. ה-ACK נשאר זהה כי לא התקבל עוד מידע מהלקוח, ו-ה SEQ נשאר זהה ל +5 server initial seq כי רק עכשין השרת שולח עוד מידע(מאז הودעת ה HELLO).

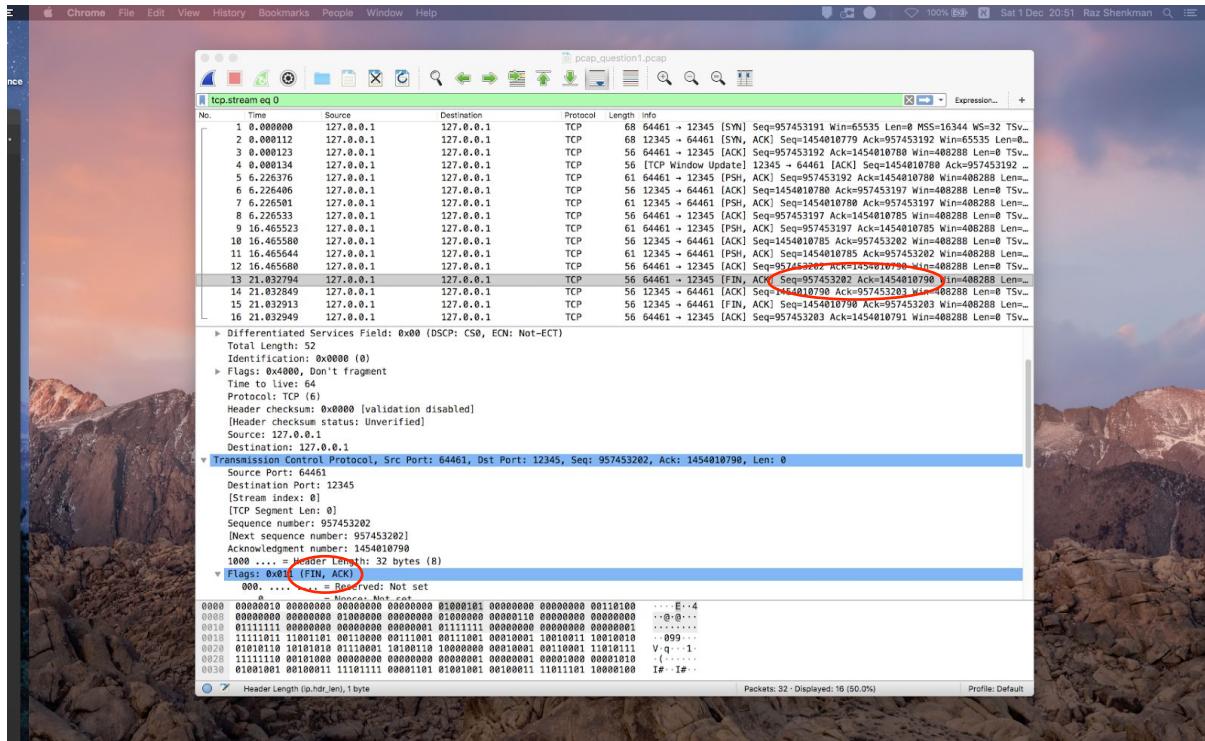


**ההודעה الأخيرة - מהלכות לשרת** (פורט המקור 64464 שייך ללקוח). הלקוח שלח ACK על ההודעת WORLD של השרת, קל לראות ש ה-field ACK number שווה ל-field seq number שהלכו שלח בהודעת הקודמת פלוס 5 (כלומר ההודעת WORLD שאורכה 5 התקבלה במלואה). ה-field SEQ number של הלקוח עלה ב 5 מאז הפעם האחרונה שראינו את הלקוח שלח הודעה, מכיוון שגם שלח הלקוח את ההודעת world(וקיבל עלייה גם ACK).

(ג) 1

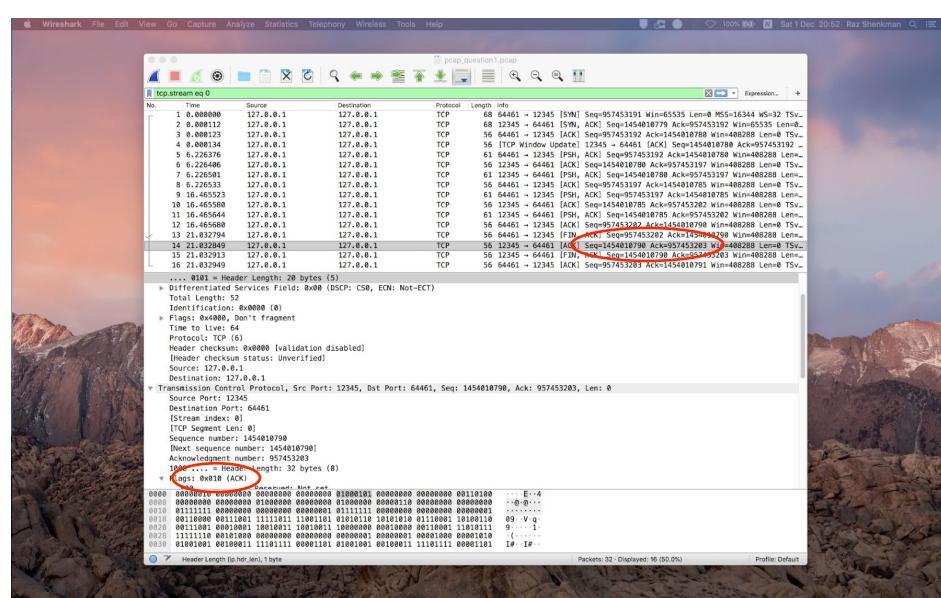
### עבור הליקוּת הראשון:

כאן הליקוּת שולח בקשה לסיום החיבור: connection



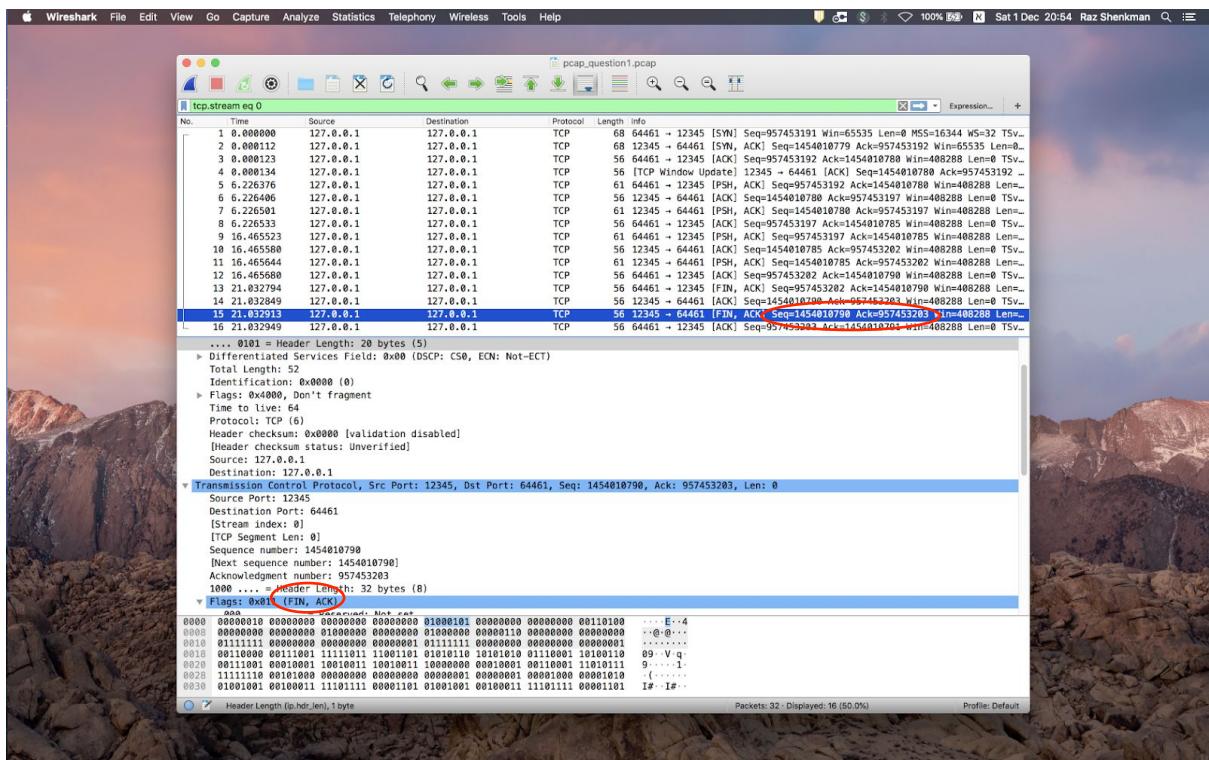
נשים לב לפורט של הליקוּת (12345), שגודל ההודעה הוא 0 ושל הרשות (64461) שולח בה דגל fin.

השרות שולח לליקוּת ack על בקשה סיום החיבור: connection



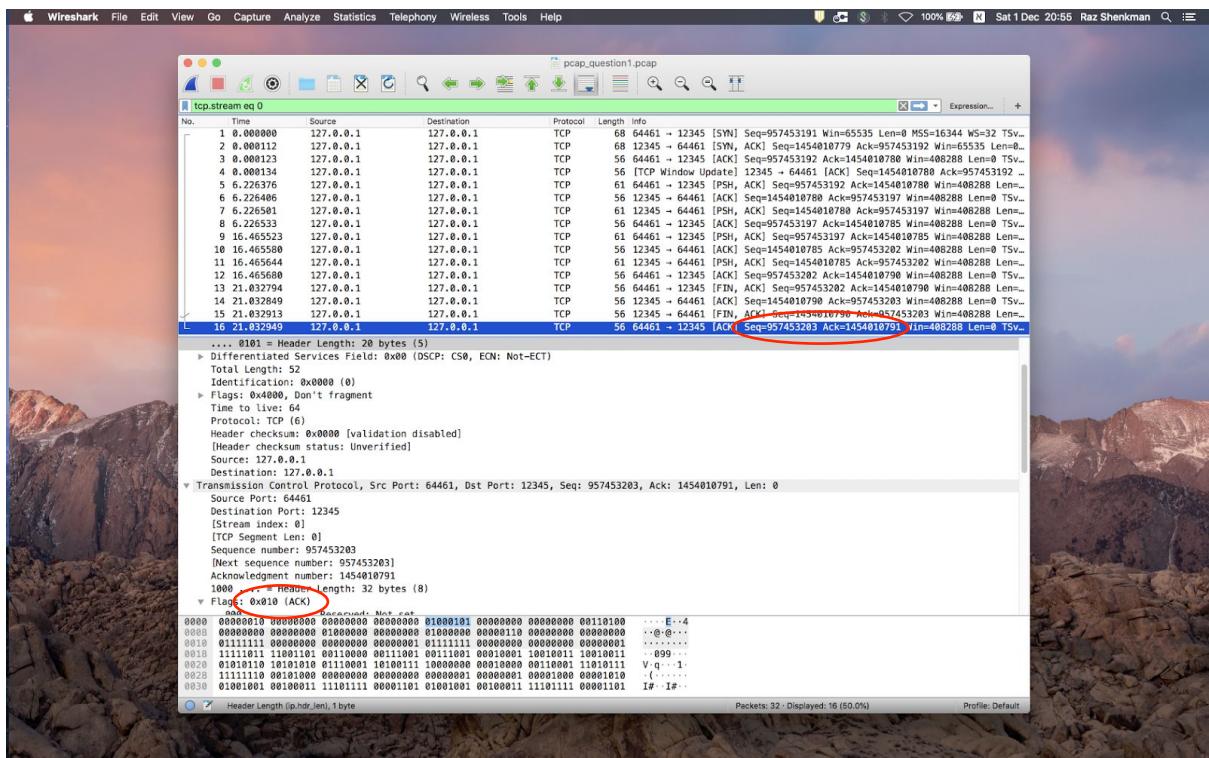
נשים לב לאck שנשלח מהשרות לליקוּת (נזהה לפני הפורטים), כאן ack שנשלח הוא seq number של הליקוּת +1 (כי נשלחה בבקשת fin).

### השרת שלוח ללקוח גם בקשה לסיום הconnection



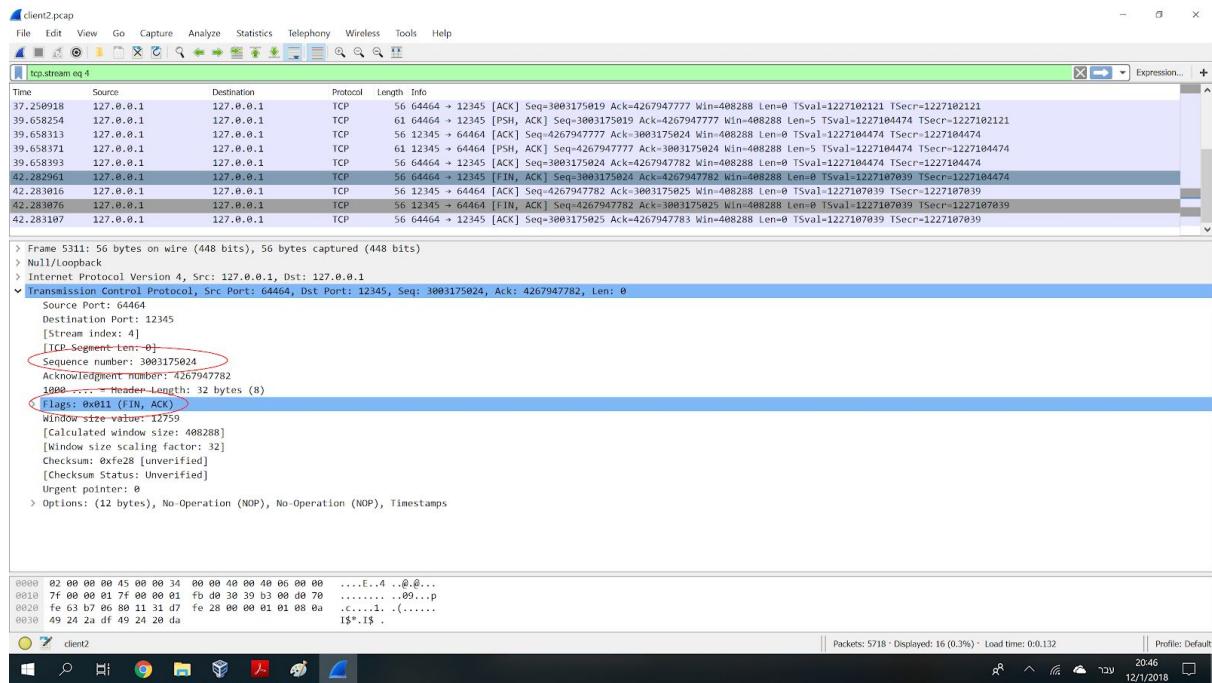
נשים לב לכך שהשרת שלוח ללקוח (זהה לפי הפורטים) בקשת FIN (דגל החסן דלוק).

## הלקוח שלוח לשרת ack:

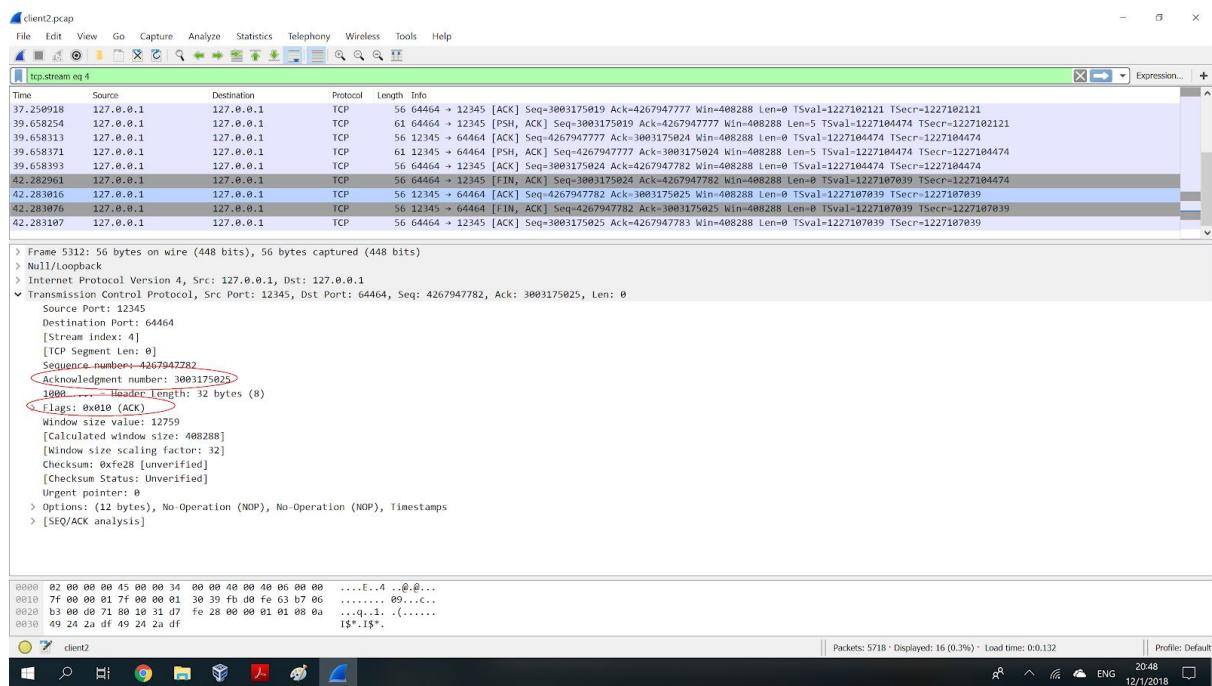


זהו בעצם ההודעה האחורונה בחס桐ן, connection, כאן הלקוח שלוח לשרת (נזהה לפי ה포רטים) הודיעת מאשר ack הוא numbeר ack הוא numbeר seq + 1 (כי נשלחה בקשה fin) והן numbeר seq של הלקוח הוא ה מההודעה האחורונה של הלקוח לשרת + 1 (כי הוא קיביל אישור על ההודעה שלו מוקדם).

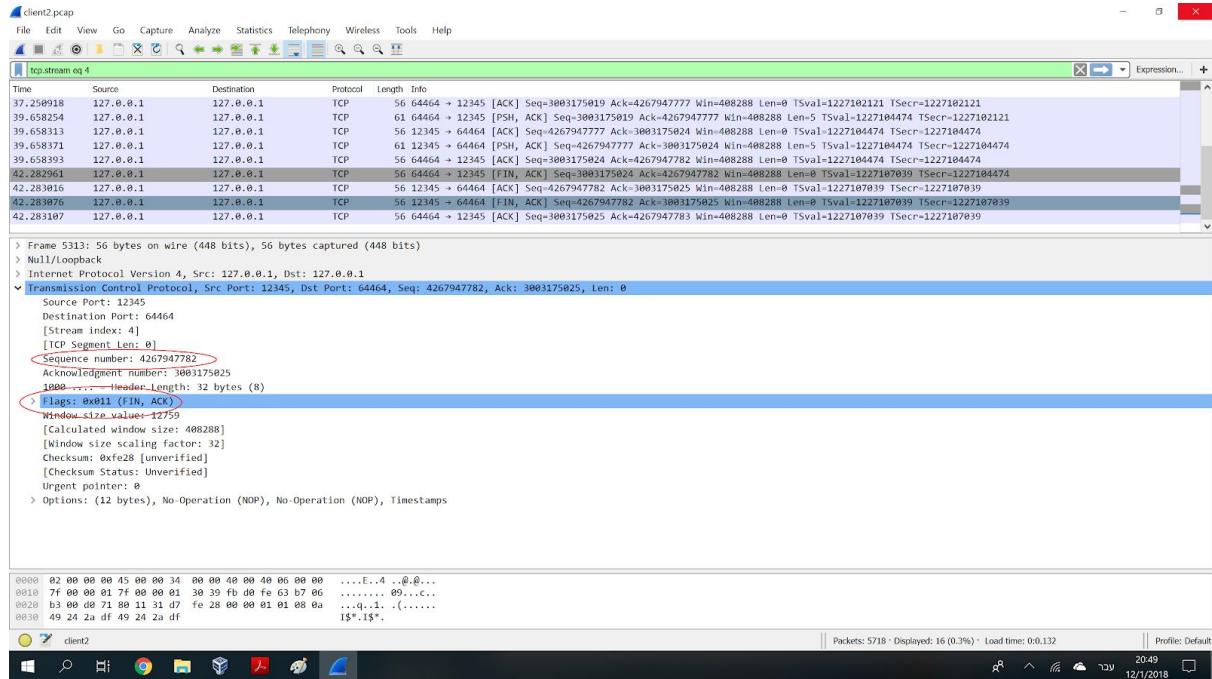
## עבור הליקון השני:



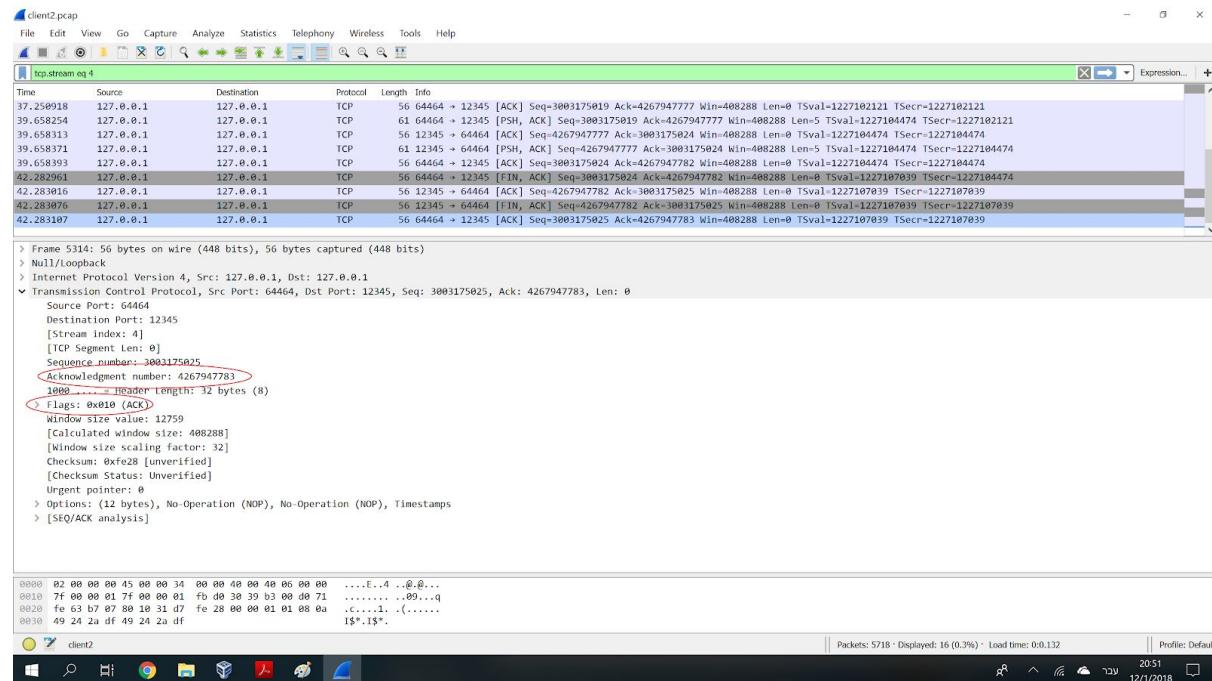
כאן ניתן לראות שדל ה FIN דולק. זהה הודעת על הריסת החיבור שהליקון שלוח לשרת. נשים לב ב seq number ונסתכל בהודעה הבאה:



זהה הודעת ACK שהשרת שלוח להליקון על ה FIN שקיבל. ניתן לראות ש ה ACK number בהודעה זו שונה ל seq number של הליקון מהודעת ה FIN פלוס אחד, כלומר הודעת ה FIN התקבלה כנדרש.



**ההודעה היא FIN מהשרת ללקוח כפי שניתן לראות בדgelim. נשים לב ל seq number של השרת בשלב זה ונראה בהודעה הבאה מהלקוח:**

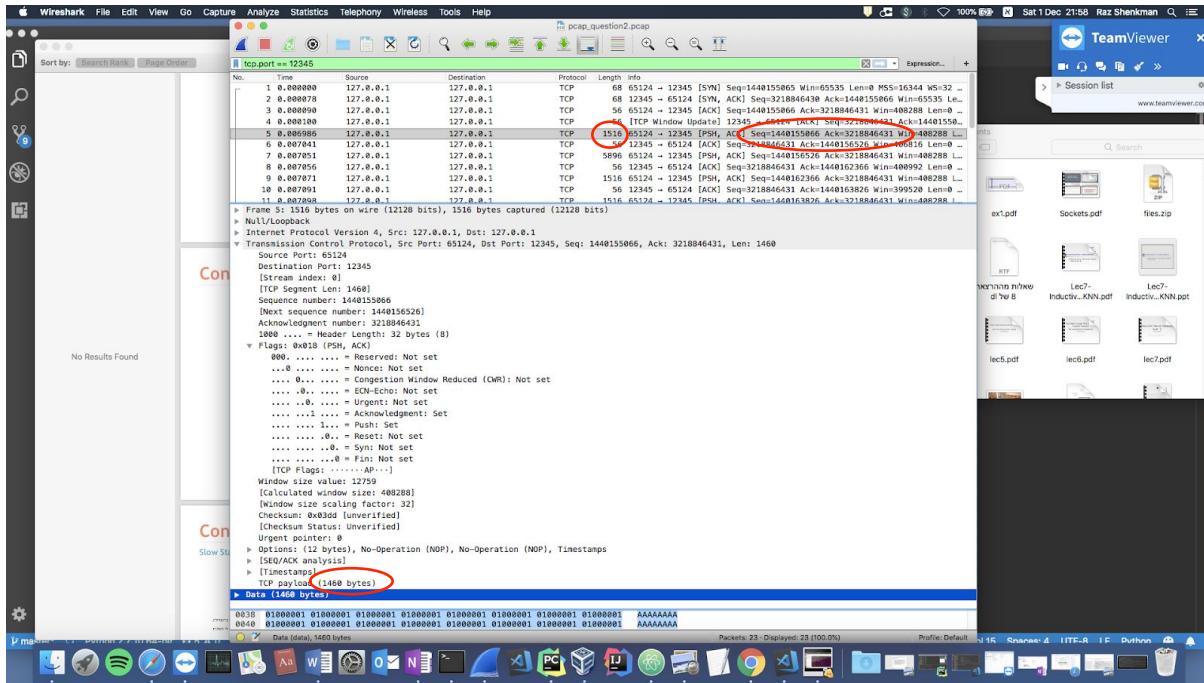


**ההודעה הבאה היא הودעת ACK על ה FIN שהשרת שלח. ניתן לראות ש ה ACK number שווה ל seq number של FIN מהודעתה.**

## שאלה 2

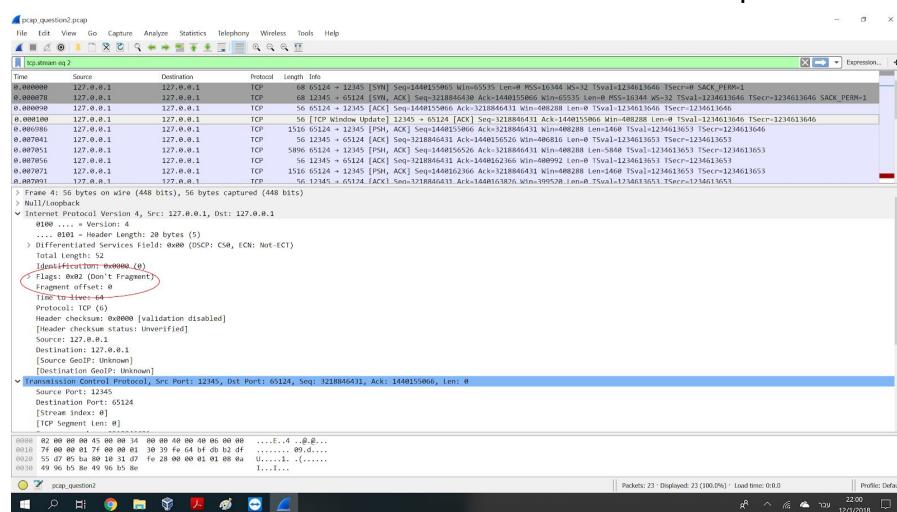
(א) 2

נתבון בחבילה שהל��ו שלח לשרת:



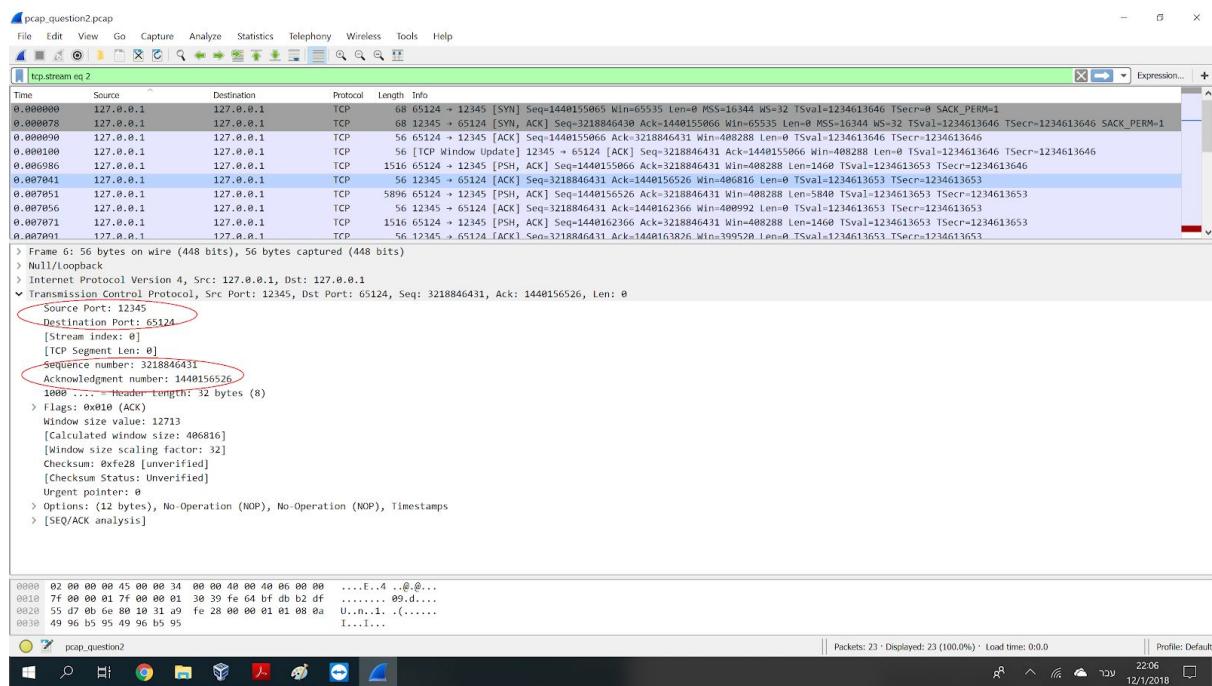
סה"כ הלוקו ציר לשלוח 15000 בטים. בהודעה הראשונה נשים לב שהלוקו שלוח לשרת מיד של 1460 בתים (זה"כ 1516 עם headers) כאשר כל המידע הזה הוא A, נשים לב שההבדל בין seq number לבין ההודעה זו (הודעה מס' 5) להודעה הבאה שהלוקו שלוח (הודעה מס' 7) הוא 1460 (כי הלוקו הגדיל את ההודעה זו) בעקבות הדגל dont fragment הנקוב באקסט ה-ack של השרת, נשים לב שיש את הדגלdont fragment(seq number)network.

נשים לב שבאופן אcht מההודעות לא הייתה פרוגמנטציה בשכבות הרשות:



בכל ההודעות של רצפי A בשכבות הרשות גם הדגל don't fragment ו-fragment offset הוא תמיד 0 כי אין פרוגמנטציה (או תמיד פרוגמנט אחד נשלח).

### התשובה שהתקבל מהשרת היא:



השרת שלח ACK על ההודעה שקיבל. ניתן לראות שnumACK השערת שלח גדול ב 1460 מ 1460 numseq שלוח לו הלקוח עם ההודעה, כלומר השרת קיבל את כל ההודעה (שארכיה 1460). באופן זה ניתן לראות שהלקוח שלוח לשרת מספר הודעות בהן רצפים של 1460 איטם, ולא הודעה אחת של 15,000 איטם. כולם בוצע תהליך סגמנטציה ע"י שכבת התעבורה.

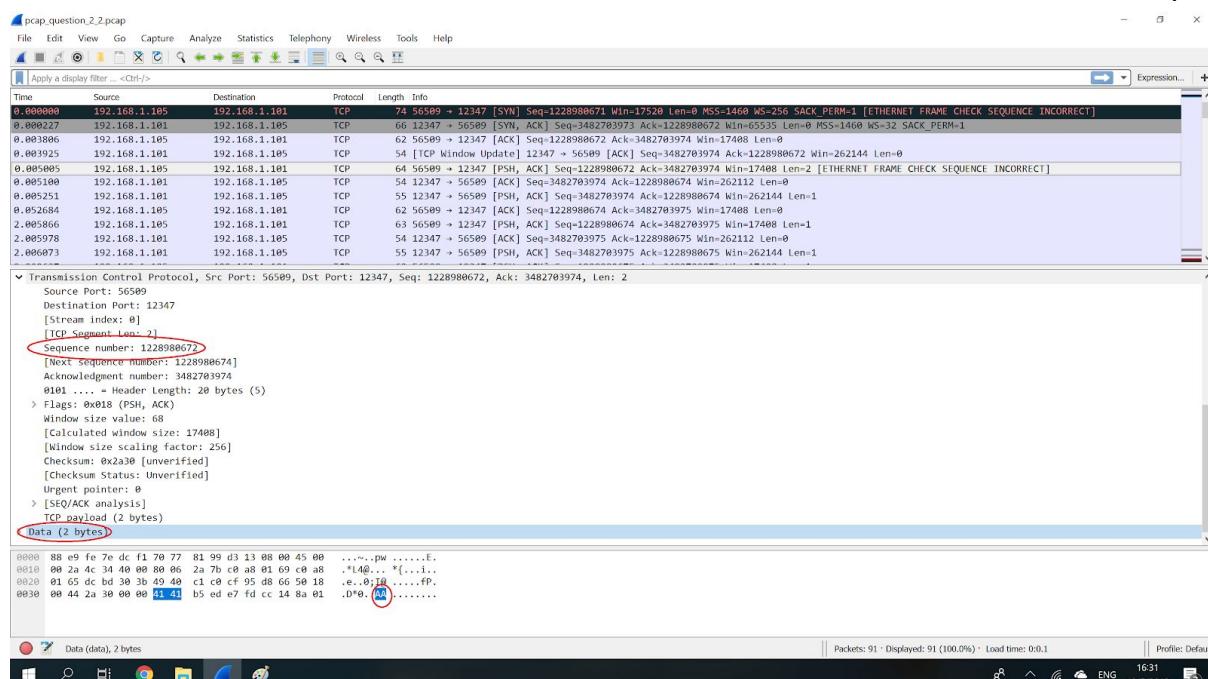
(ב)

תהליך handshake בדומה לסייעים קודמים שהסבירנו. נשים לב שהקן של הלקוח הוא 105.1.168.192, והשרת בפורט 12347, הלקוח בפורט שרירוט של מ"ה שהוא 56509.101.168.202.

בתהליכי התקשרות בין הלקוח לשרת, השרת קיבל הודעה AA (שני A צופים באותה הודעה) ובשאר ההודעות קיבל כל A בנפרד.

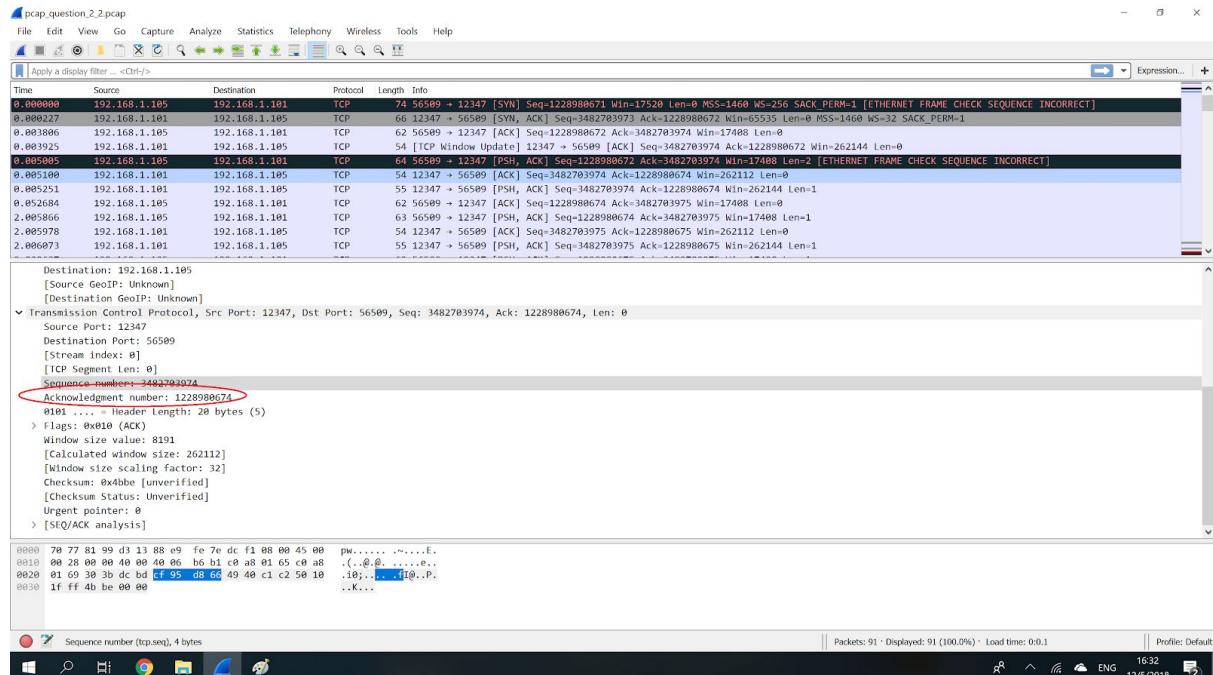
```
Received: AA
Received: A
Client disconnected
```

כאן רואים את החבילה הראשונה שנשלחה לשרת: AA (וין ה-2 בתים- data הוא AA) בזוויתascii שדה (AA)



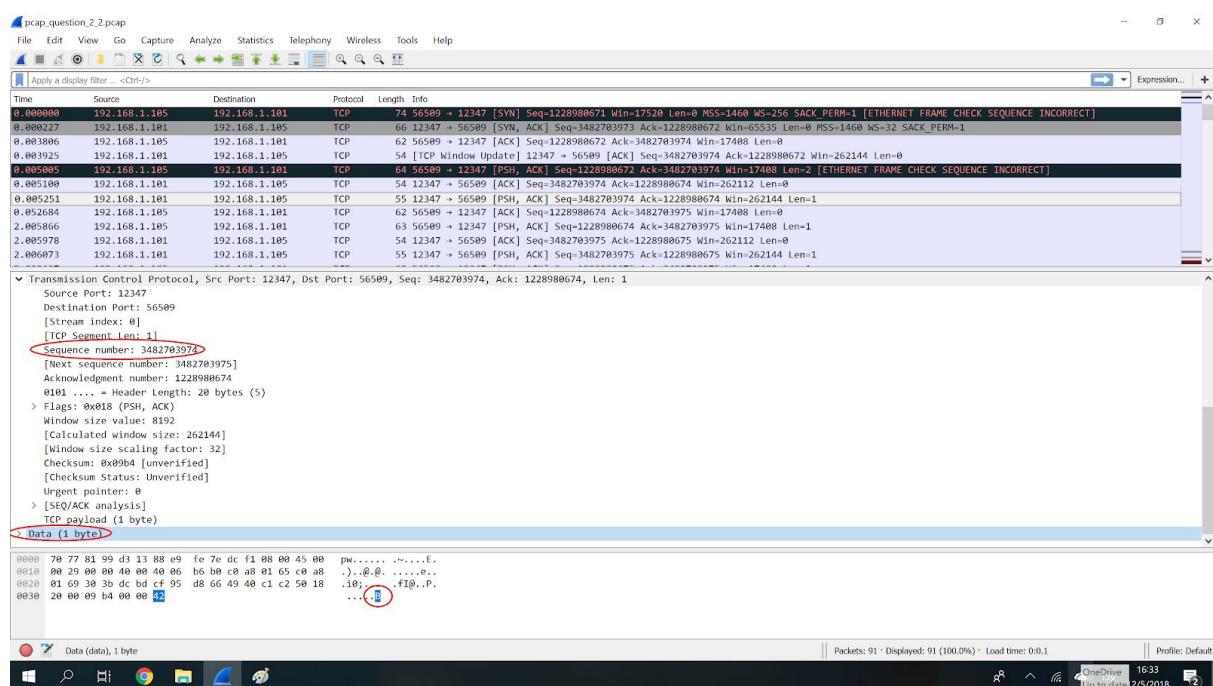
בהתודעה זו ניתן לראות את שליחת ההודעה AA מהלוקה לשרת. אנחנו יודעים שבקדוד הלקוקה ה send מבוצע פעמיים בנפרד, אך כמו שניתן לראות ב data של ההודעה השליחות "התלכדו" להודעה אחת והميدע נשלח הפעם נשלח בהודעה אחת.

נשים לב ש ה seq number הוא 1228980672, **ונסכל בצלום הבא:**



ראים שהשרת שולח ללקוקה ACK עם seq number 1228980674 שהוא גדול ב 2 מ ה seq number של הודעת AA שאורכה 2 --column ההודעה התקבלה במלואה בשרת, ואפיו הצבירה כהודעה אחת נראה הר' נשלח עליה ACK מצטבר ייחיד.

בחבילה הבאה נשלח B בודד מהשרת:



311130777 רץ שינקמן  
322952433 אוד שחר

## עליו הוחזר ACK מהלוקות

Wireshark screenshot showing a sequence of TCP segments between two hosts. The first host (192.168.1.105) sends SYN, ACK, and Window Update segments. The second host (192.168.1.181) responds with ACKs and SYN-ACKs. The fifth segment from the first host is circled in red, indicating it is being analyzed.

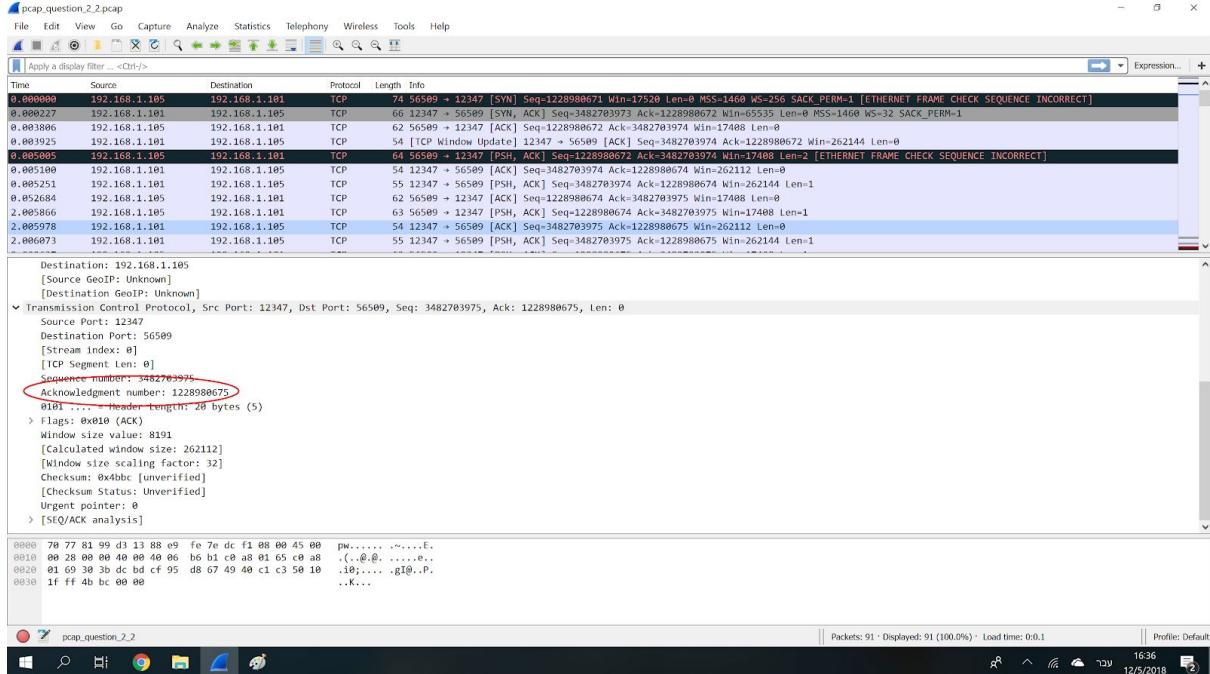
ואכן קל לראות ש ה-ACK בחבילה ה-**Z** (מהלוקה לשרת) גדול באחד מ ה-**seq** שהוא בחבילה עם המידע **B** (מהשרת ל-**локות**) שצולמה קודם.

## נתבונן בחבילה הבאה, נשלח A מהלוקות לשרת:

Wireshark screenshot showing a sequence of TCP segments between two hosts. The first host (192.168.1.105) sends SYN, ACK, and Window Update segments. The second host (192.168.1.181) responds with ACKs and SYN-ACKs. The fifth segment from the first host is circled in red, indicating it is being analyzed. A red circle also highlights the 'TCP payload (1 byte)' field in the packet details pane.

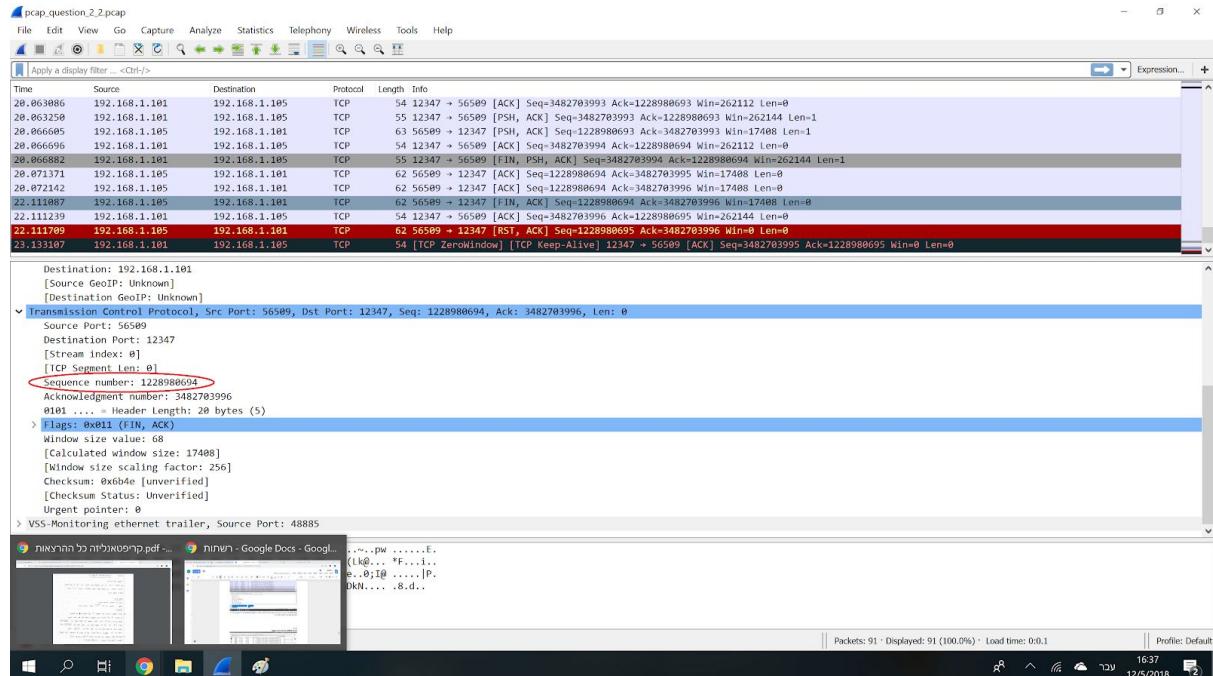
נשים לב שגודל ההודעה הוא 1, מתקיימם שנשלח A מהלך 0 לשרת (גודל הdata הוא 1), גודל header הוא 20, ה number seq של הלך הוא 1228980674 וה number ack הוא 1228980675 של השרת שהוא לא השתנה מהשליחה הקודמת של השרת (כי נשלח רק ack).  
**נתבונן ב ack של השרת ללקוח:**

**נתבונן בack של השרת ללקות:**



נשים לב שהערך `ack` הוא 1228980675, כלומר הערך `seq` של הילוקה ממקודם +1, כך שהרטת מאשר לילוקה שהוא קרא מיד באורך בית אחד. נשים לב שהגודל של הערך `tcp header` הוא 20, דגל `ack` דלוק.

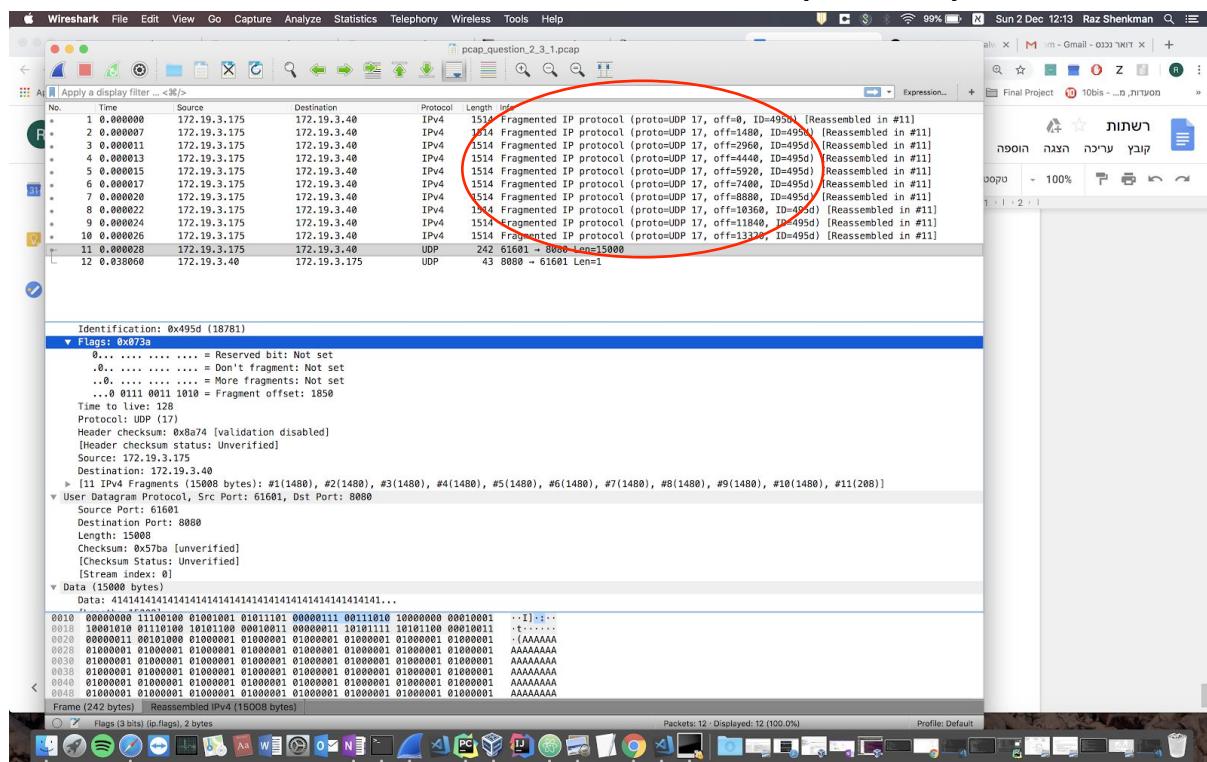
כל הבעיות הבאות גם זהות ב data שהה[ן] (כלומר שלוחות רק A אחד, B אחד) ולכן נחשוך מהראות את כלן. נשים לב שב הודעה האחורונה(ACK מהשרת על ה A האחרון), ה number seq של היקף גדול ב 22 ממה שהוא בהתחלה, מה שמשמעותו נשלחו בכל ההודעות ביחד סה"כ 22 תווים. תוצאה זו (seq number שגדול ב 22) צפיה להתקבל לא משנה אם שתי הודעות send מתלכדות או נשלחות בנפרד שהרי בכל מקרה "ישלו" סה"כ 22 בתים.



צולם ה FIN של הלקוח שבו ה seq number הוא 1228980694, לעומת גודל ב 22 מ ה initial sequence שהוא 1228980672 (כמו שציינו).

(1(λ(2

**שליחת A 15000 מהלך לשרת בקס:**



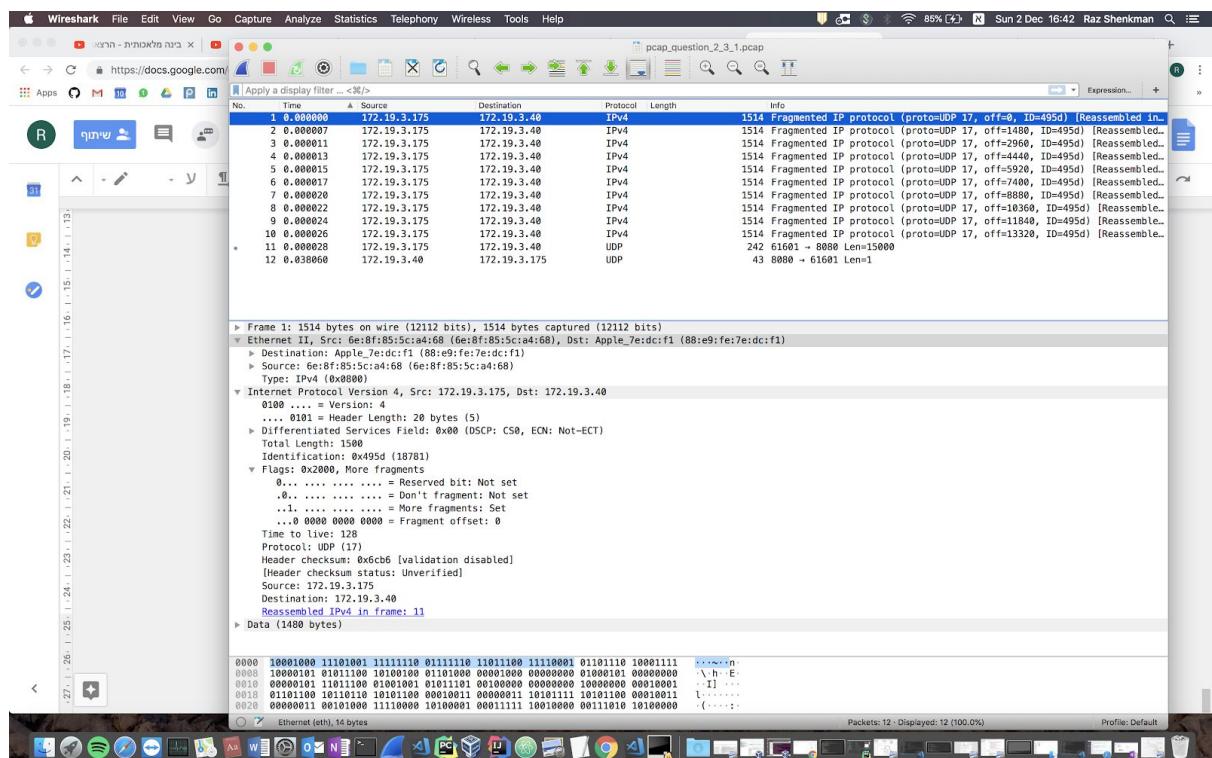
can also mean that the complex (the company that issued the final certificate of arrival).

נשים לב לתהילר: הקדש מורה לשכבת הרשות בקשה לשלוות 15000 בתים, כאן שכבת הרשות עשו  
ברגמאנישוב, לטעמך אפרת אט חביבלה לטעמך ברענין, שבועות 1514 (בכבודם לטעמך ברענין).

פרגמנטציה, כלומר מפרקת את החבילה לפי הטעות, שהוא 1514 (הסקתי לפי מה שנשלח).

כיוון שהפונקציית tcp לא מבצע חלוקה של החבילות (tcp מבצע segmentation ו-tcp משביר דגל fragment don't), אין צורך לשכבות הרשות לטפל בחלוקת הפקודות לחלקיים שונים).

### החבילה הראשונה שנשלחה:



נשים לב שה**mss** (גודל המידע המקורי) יכול להשתנות לא כולל **headers** הוא 1480 (רואים זאת לפי השדה **data**, עבוריו מתווסף header של שכבת הרשות (20 בתים) ושל שכבת הקו (14 בתים) וזה"כ גודל של 1514).

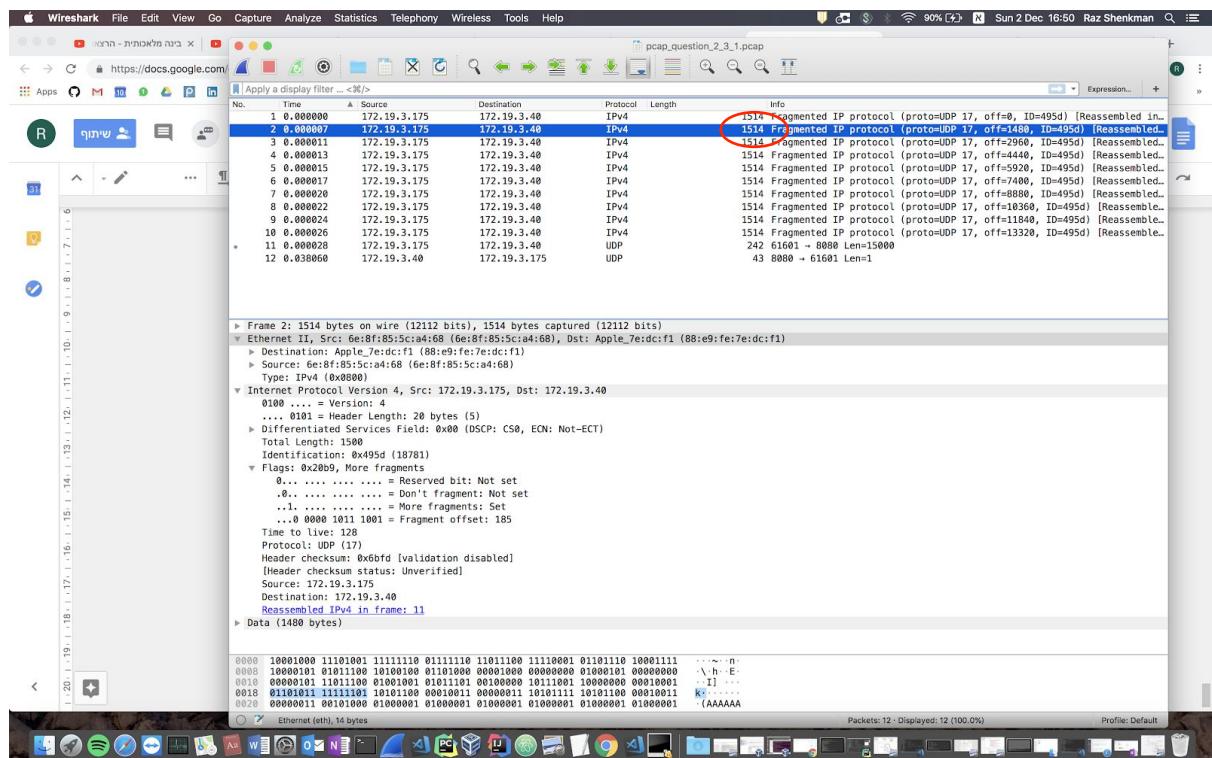
נתובן בכל החבילות לא כולל החבילות האחרונות:  
לכל חבילה המידע כולל protocol ip, fragmented ip, column שהוא חלק מהחבילות שעברית פרוגמנטציה.

נשים לב שעבור החבילה הראשונה והחbillות אחרת (לא כולל האחרונה) דלוק הדגל של `fm` (כלומר, בוצעה פרוגמנטציה, ויש עוד חbillות אחרית החבילה הזאת). נשים לב שבtcp היה דלוק הדגל `don't fragment` בשכבת הרשת.

בכל החbillות חוץ מחבילה אחת לא יהיה `header` של `dps`, כיון ששכבת הרשת מבצעת פרוגמנטציה לא משנה לה איך המידע שהוא קיבל הגיע, היא סה"כ מחלקת אותו לחbillות בגודלים המתאימים ומוסיפה את `header`'ים הרלוונטיים שלה, ולכן רק `header` אחד של `dps` בכל החbillות שנשלחו.

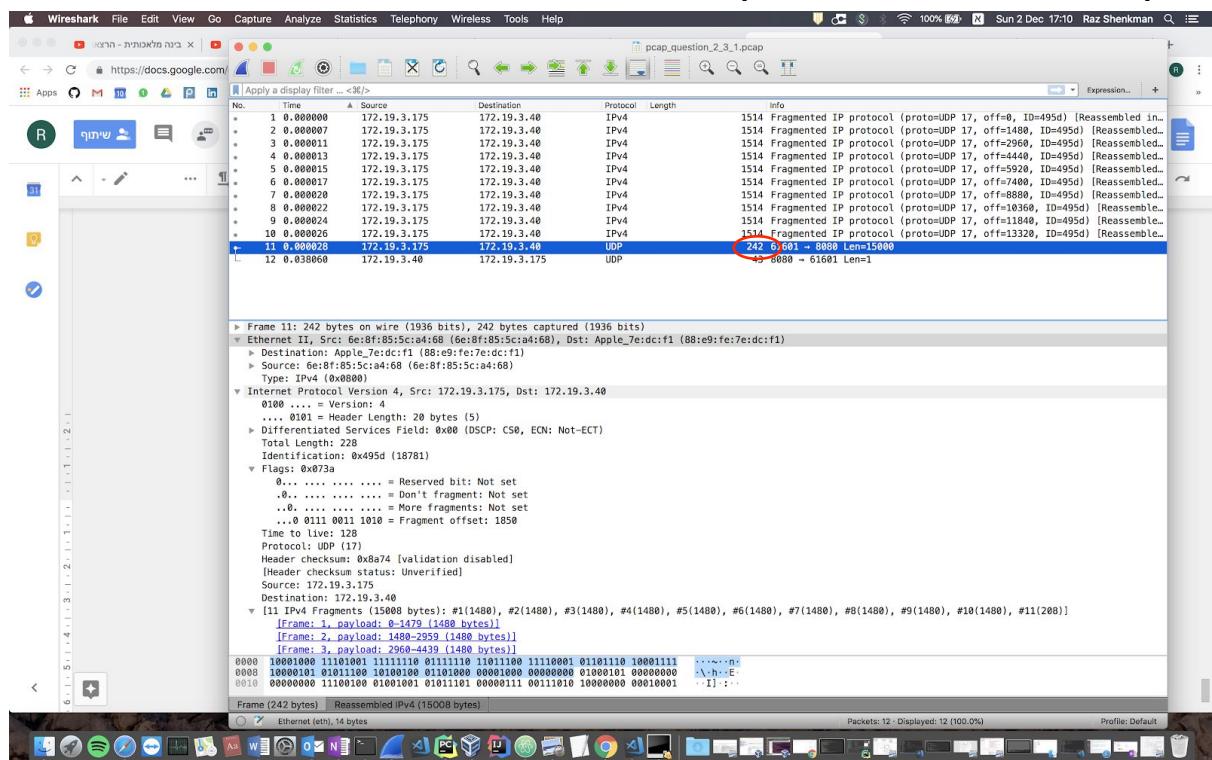
נשים לב שהchassisId (שדה שחשוב בפרוגמנטציה כדי לדעת שכל החbillות קשורות אחת לשניה) הוא זהה בכל החbillות והוא `0x495d0`.  
נשים לב שהoffset של החבילה הראשונה הוא 0.

**נתבונן בחבילה השנייה שנשלחה:**



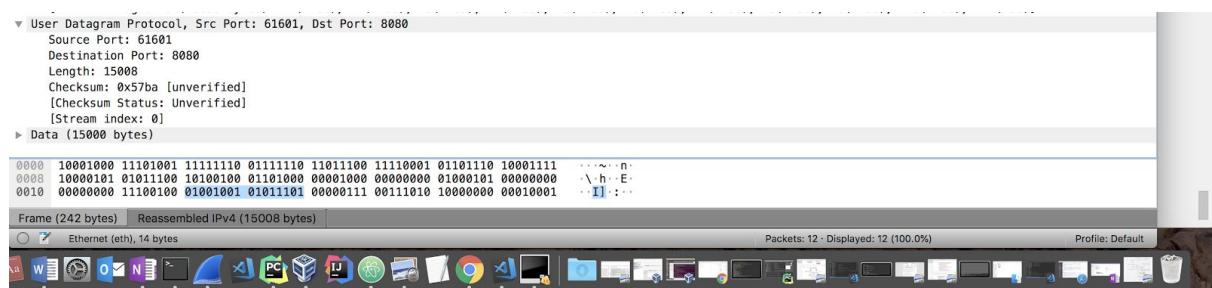
כאן בחבילה השנייה נשים לב שהoffset הוא 8 ונתנו מפה הבטים בחבילה המקורית לפני הפרגמנטציה הוא 185, המידע מוחלק ב8 ולכן זה בעצם  $185 * 8 = 1480$ , זהה הגיוני שכן החבילה הראשונה שנשלחה כוללת 1480 bytes מהחבילה המקורית, והחבילה השנייה תתחיל עם המידע שנמצא אחרי אוטם 1480 בתים. נשים לב שהidentification הוא גם 0x495d. גם כאן דלוק דגל `mf`.  
ושזה"כ נשלח גם כאן 1480 בתים.

### נתבן בחבילה האחורה שנשלחה מהלוקות:



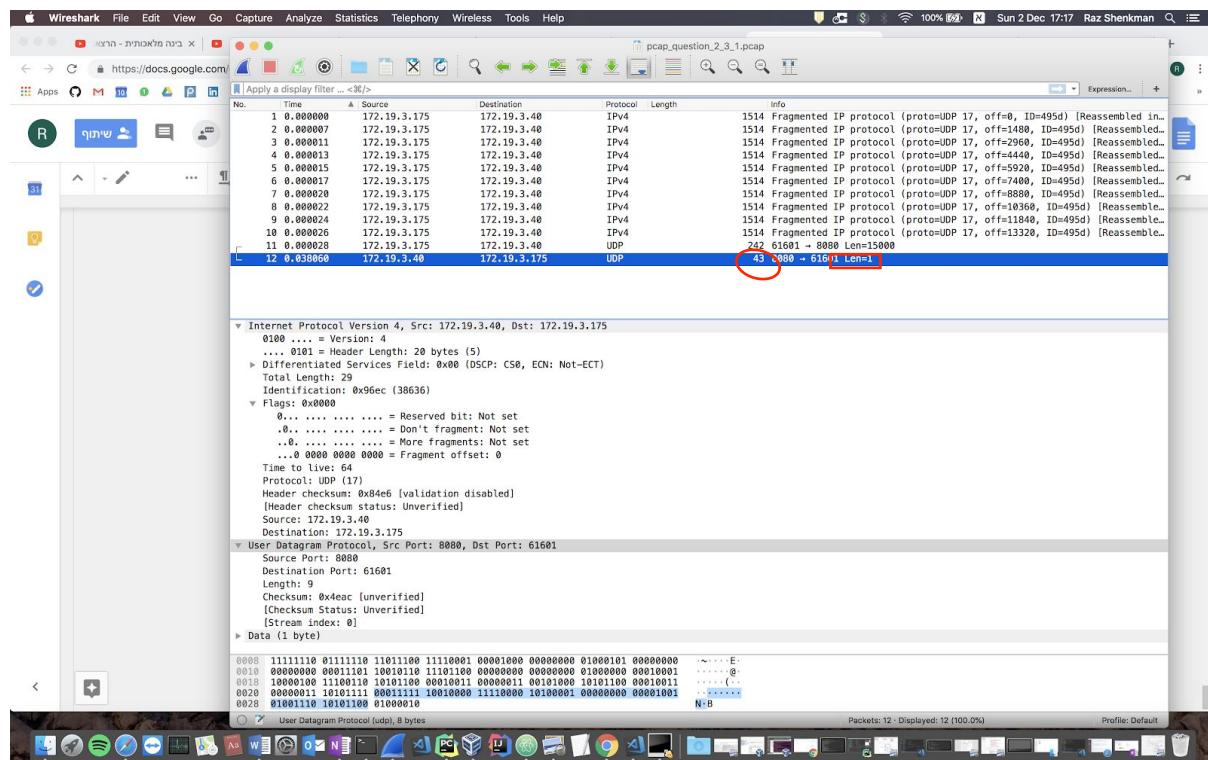
נשים לב שהגודל החבילה צריך להיות 15000 פחות 1480 \* כמספר החבילות השלומות שנשלחו = 200, וכיון שיש גם את header udp, ואת שאר headers שבידם הם 42 בתיים ניקבל שגודל החבילה הוזה הוא 242 בתים.

נשים לב שהheader identification הוא 0x495d, ודגל mf는 0, (כי זו החבילה האחורה), נשים לב שבשכבה הקפם המידע הוא יחסית מצומצם:



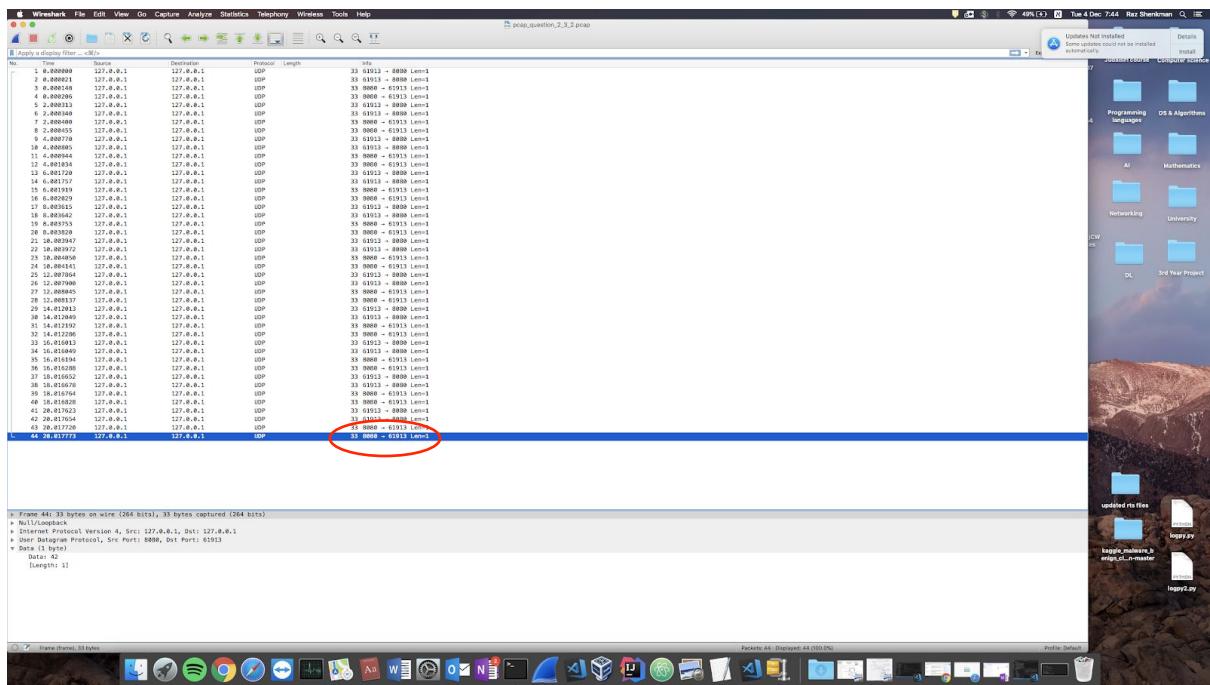
שכן לUDP לא משנה כמה בתים נשלחים, והוא פשוט מורידה לשכבה הרשות שתתפל בשילוחו שלהן.

**נתבונן בחבילה שהשרות שלח ללקוח:**



נשים לב שלחבריה זו אין דגלים בחבילת הרשות הקשורות ל프로그램נטציה (כיוון שהיא לא עברת פרוגמנטציה) וכן נשלח רק ביטחון אחד של מידע, בתוספת headers משאר השכבות נגי'ע ל-43 בתים של מידע סה"כ.

## שאלה 2ג



נשים לב שבמקרה זה (שרות ולקוח UDP, הלקוח שלוח כל פעם 2 תווים 'A' רצופים) כל ההודעות יתתקבלו בהפוך (ניתן לראות בתמונה ובקובץ pcap\_question\_2\_3\_2.hdr) שכל ההודעות מוגדל 1 ולא 2, וההТОן הוא A או B). נזכיר שTCP ראיינו שלפעמים שת' הודעות 'A' רצופות יתמזגו להודעת 'AA' אחת. ננסה להסביר את התופעה.

ידוע לנו שTCP מעביר את המידע כпотם של Bytes (byte stream). ז"א, הוא לא מודיע על גבולות של הודעה או לא הודעה, הוא רק יודיע שיש לו רצף של Bytes עליון להעיבר הלאה בסדר זה. כאשר בTCP עושים שני send רצופים, אז' לבסוף הבטים שהוא צריך להעביר נוכנסו ביחד ביחס שני הBytes האלו TCP כמו שהוא יודיע העיבר אותם כbyte stream לא הספיק לשלוח את ה A הראשון לפני שהשני כבר נוכנס, TCP אמרו שלוח את כל הבטים שיש לו ב buffer前に מודעות ל "מי הגיע מאיזו הודעה" ולכן פעמים (כאשר TCP לא הספיק לשלוח את ה A הראשון לפני שהשני נוכנס) הם יתמזגו להודעה אחת (הרי ב buffer ישם המידיע AA).

ב UDP לעומת זאת, אין את אותה ארכיטקטורה של שף בתים שMOVEMENTS מקופה לסתה. ב UDP תמיד כתובנו:

```
socket.sendto(msg,(ip,port))
```

כזכור את הודעה הוא יהיה שלוח מיד ליעד, ולא בהתבסס על החיבור ביניהם. לכן כל שליחת הודעה הייתה בלתי תליה נזומה בשליחות ההודעות האחרות, וכן כל הודעה A או B נשלחה בפני עצמה והתקבלה אצל השרת או הלקוח בהפוך/