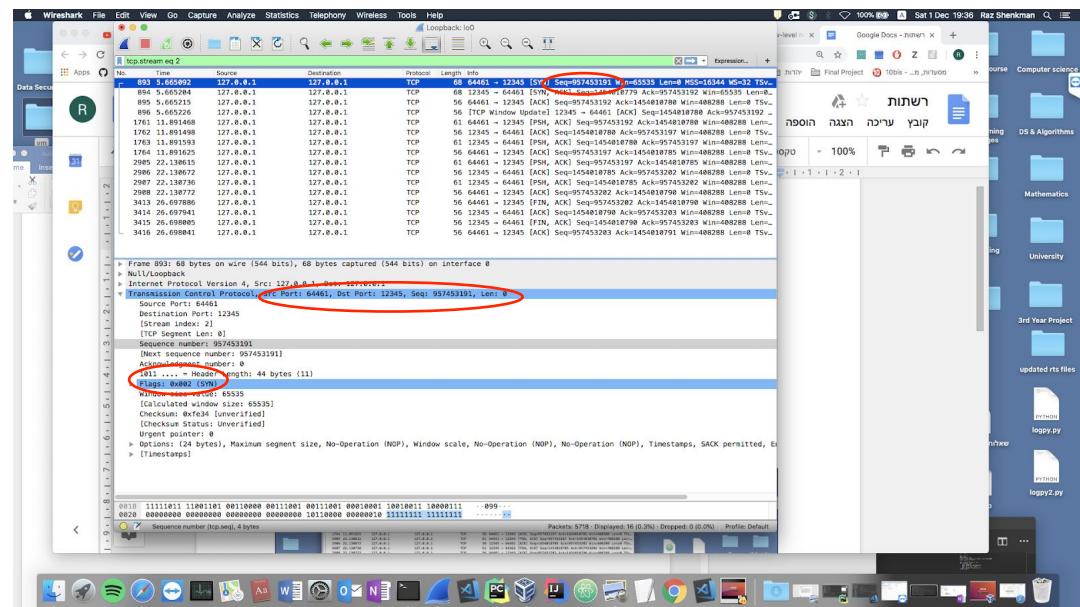


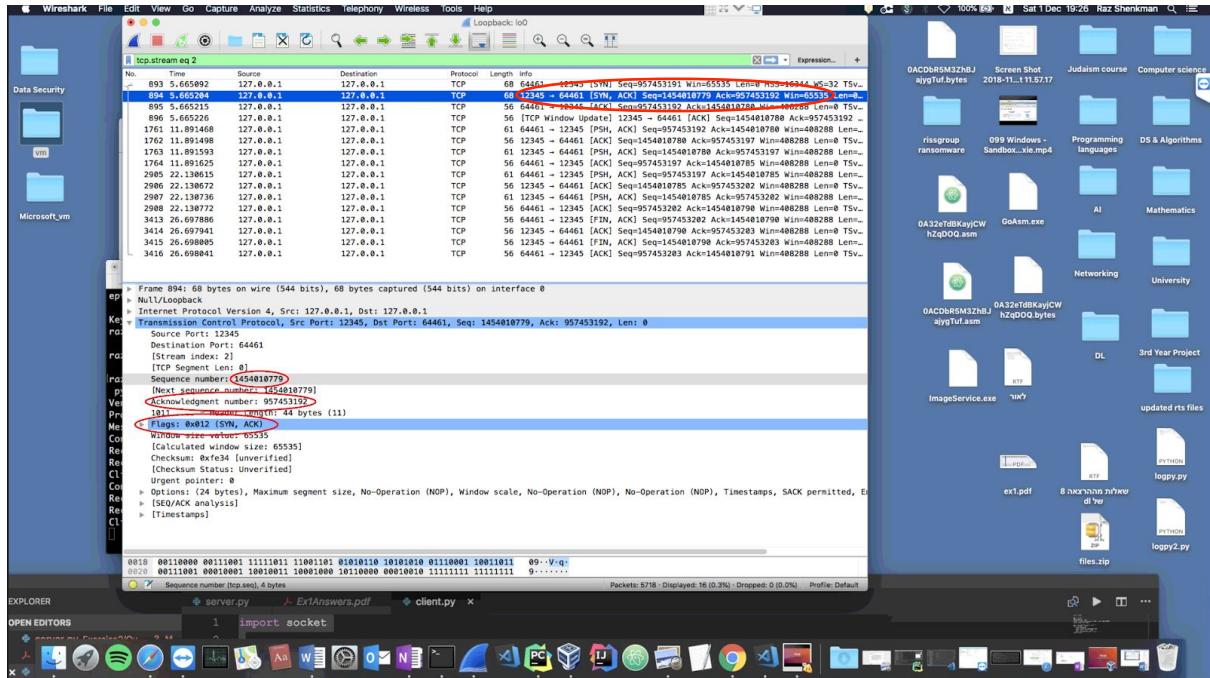
רשותת תרגיל 2

(1) א'

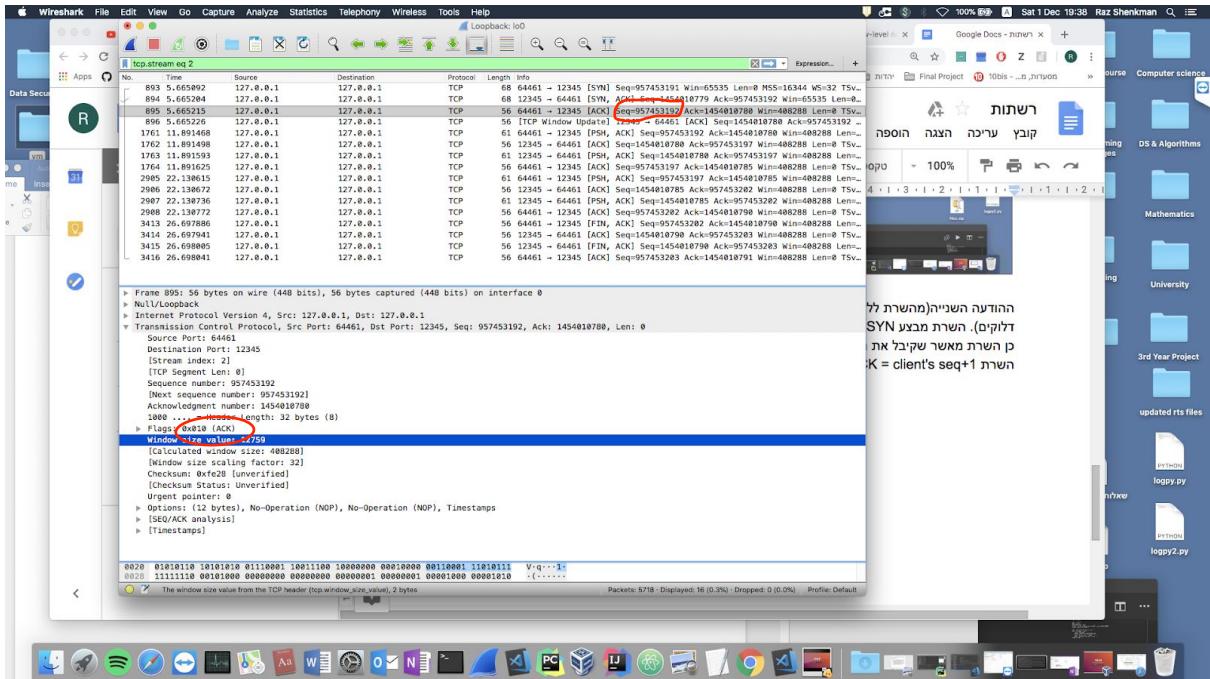
הלקוח הראשון:



ההודעה הראשונה (הלקוח לשרת) היא בקשה chs, נשים לב שהיא מלאה ב chs flag (מוקף באדום), שבעצם מציין שהלקוח רוצה ליזור קשר עם השרת. נשים לב שלລוּהsequence number ראשון שנקבע ע"י הלקוח (מוקף באדום).

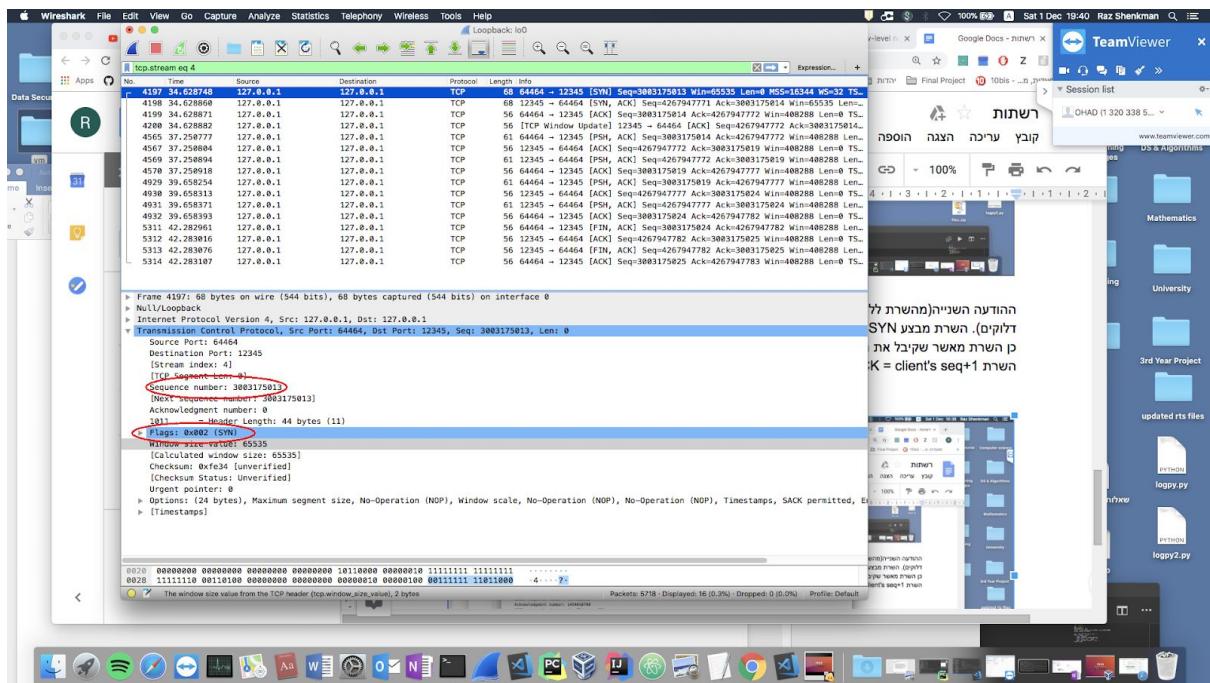


ההודעה השנייה(המשרת ללקוח) היא גם הודעת SYN וגם הودעת ACK(כמו שניתן לראות שני הדגמים האלו דלוקים). הרשת מבצע SYN - הוא קובע seq number - השונה משל הלוקח ושלח ללקוח. כמו כן הרשת מאשר שקיבל את ה seq number הראשון של הלוקח באמצעות ACK. ניתן לראות שבהודעת הרשת מאשר שקיבל את ה `seq number + 1`, כלומר הרשת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

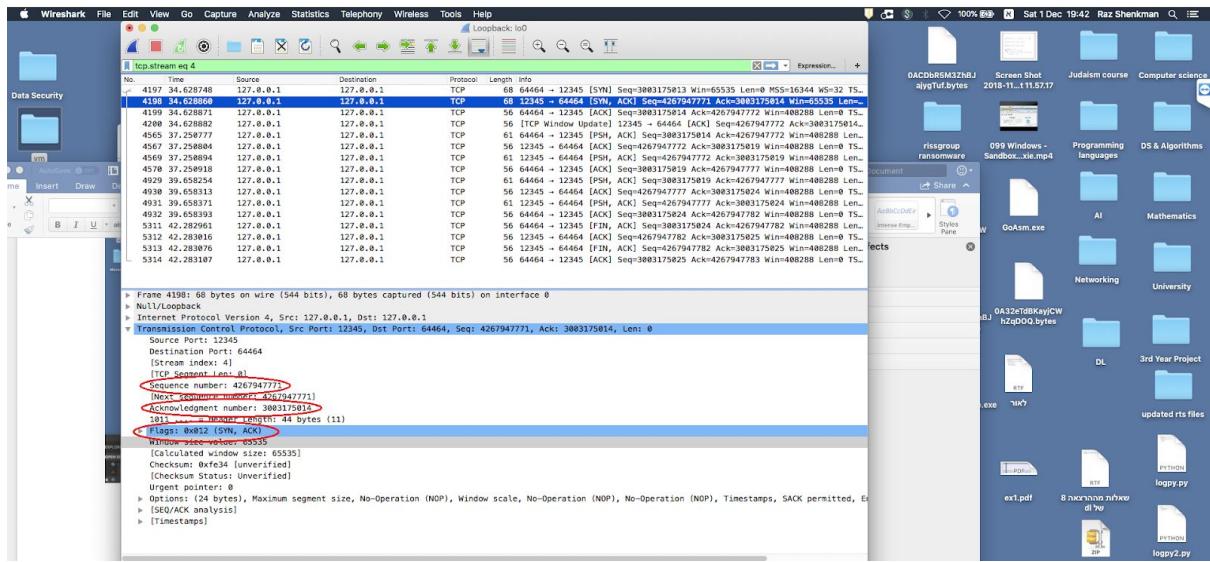


כאן זו הودעה מהלкова לשרת שהיא הודעת ack (רואים שהוא הדגל היחיד שדלוק), כאן ה seq הוא אותו seq של הלקוח ממוקדם (+1 עבור החעט הראשון) והו ackseq של השרת (עבור ה synseq של השרת).
כאן בעצם מסת'ים תהליך handshake, הלקוח אישר את ackseq את synseq של השרת.

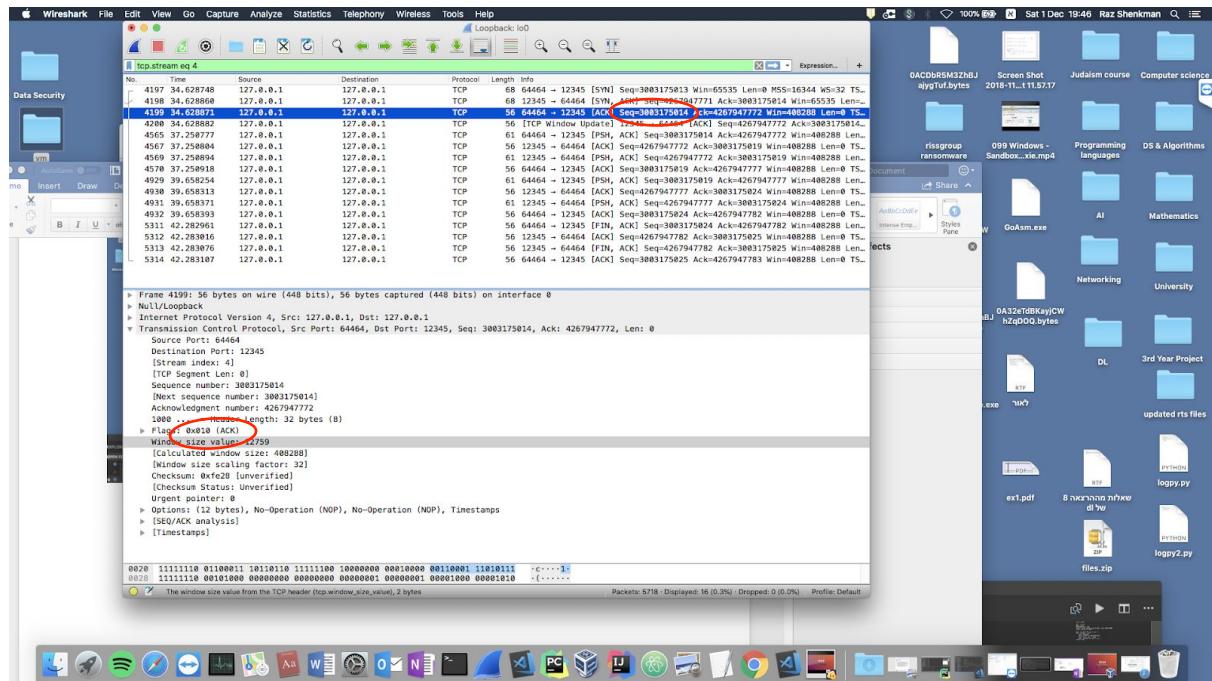
הלקוח השני:



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



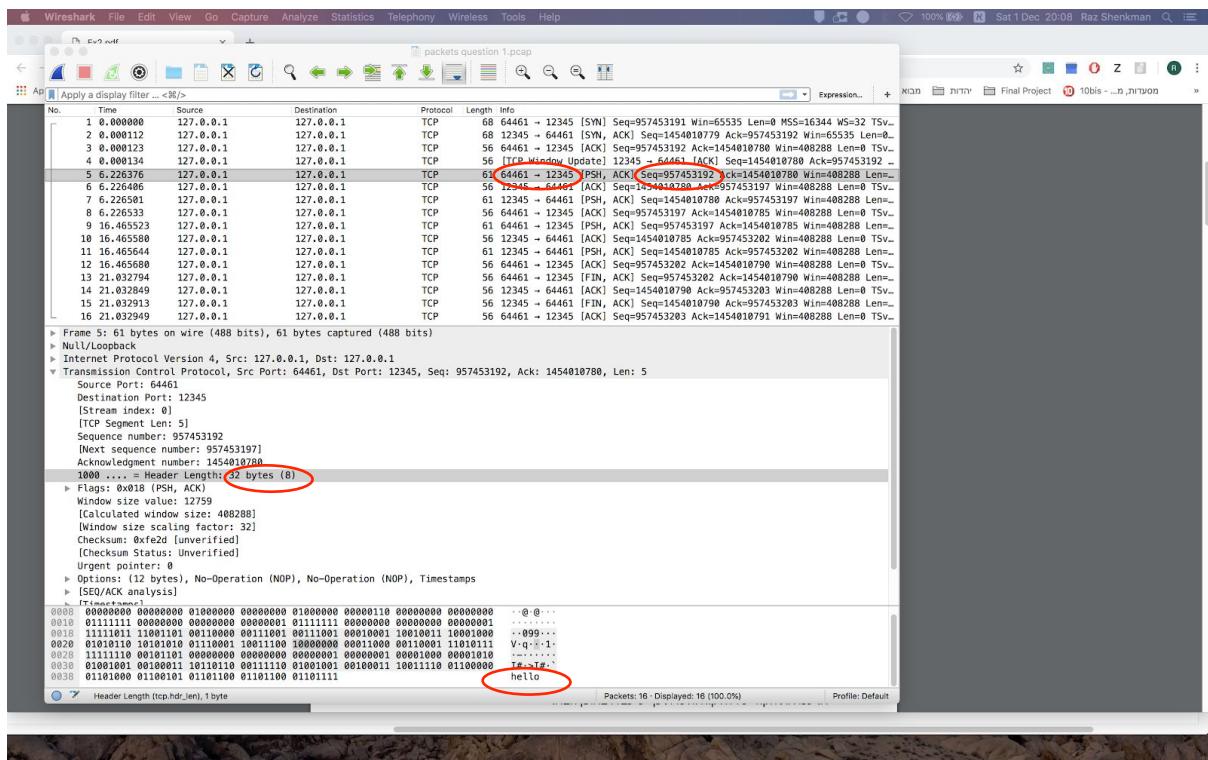
ההודעה השנייה בתהליך מהשרת ללקוח השני היא גם הודעת SYN וגם הודעת ACK (כמו שניתן לראות שני הדגמים האלו דלוקים). השרת מבצע SYN - הוא קובע num seq number משלו (שינויו של הלוקוט) ושולח בהודעה זו ללקוח השני. כמו כן השרת מאשר שקיבל את num seq number הראשון של הלוקוט השני באמצעות ACK. ניתן לראות שבה Hodutut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



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 מסתים תהליך handshake, הלוקון אישר את acksyn של השרת.

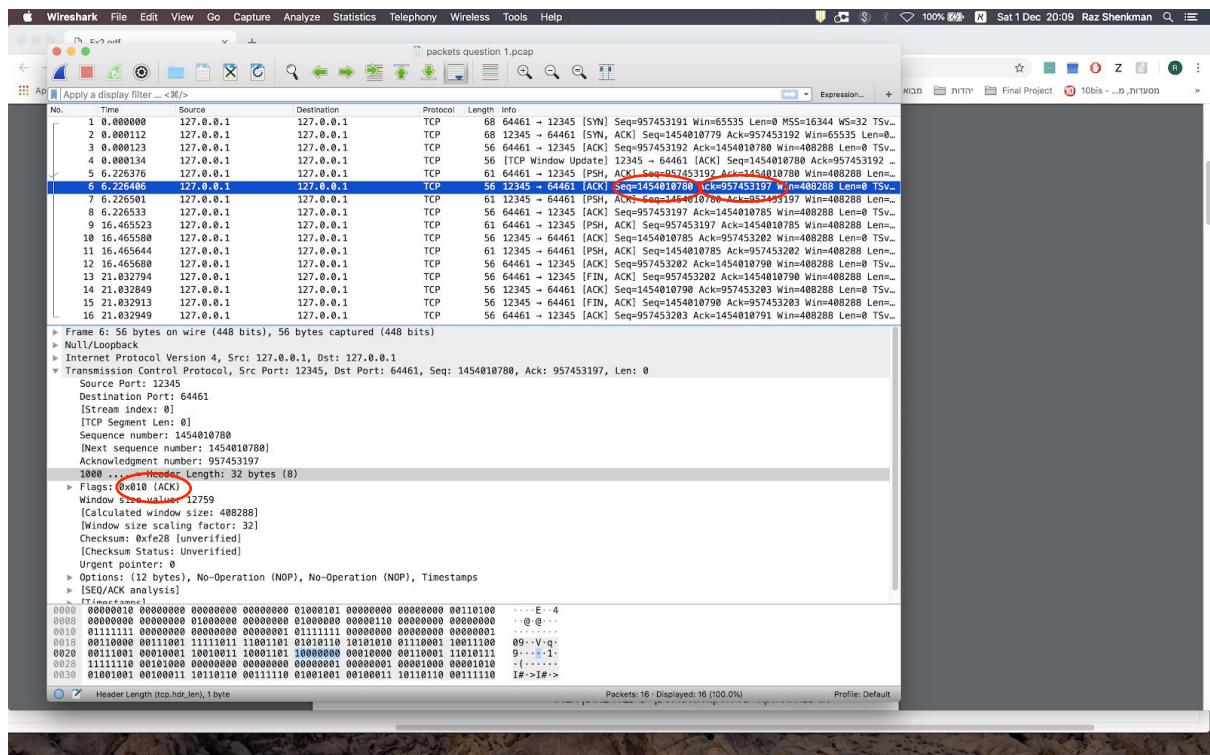
(1) ב'

בלוקות הראשונים
הודעת hello מהליקות לשרת:



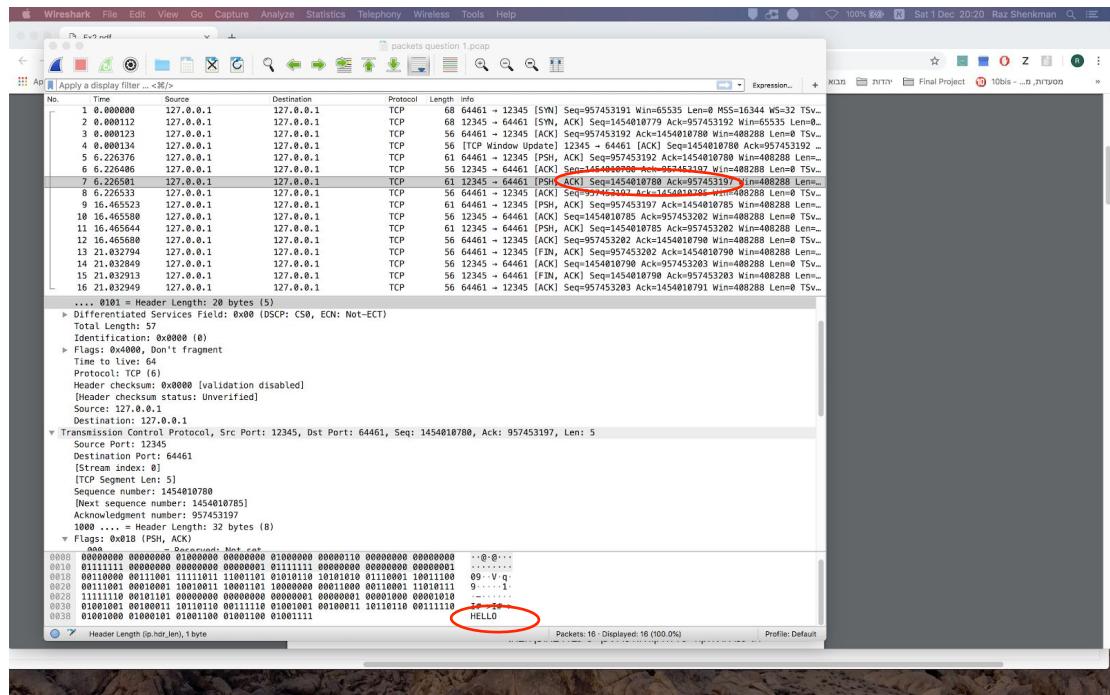
נשים לב לפורט של הליקות source port (64461) ופורט השרת destination port (12345) והוא 12345. נקבע ע"י מ"ה (sequence number) הוא 957453191 והאקרולוגים acknowledgement number (ההודעה הקודמת שלו) הוא 957453192. נשים לב שגודלו של הערךsequence number הוא 32 bytes (header length). אורך ההודעה הוא 5 bytes (המילה hello).

כאן השרת שולח ללקוח ack:



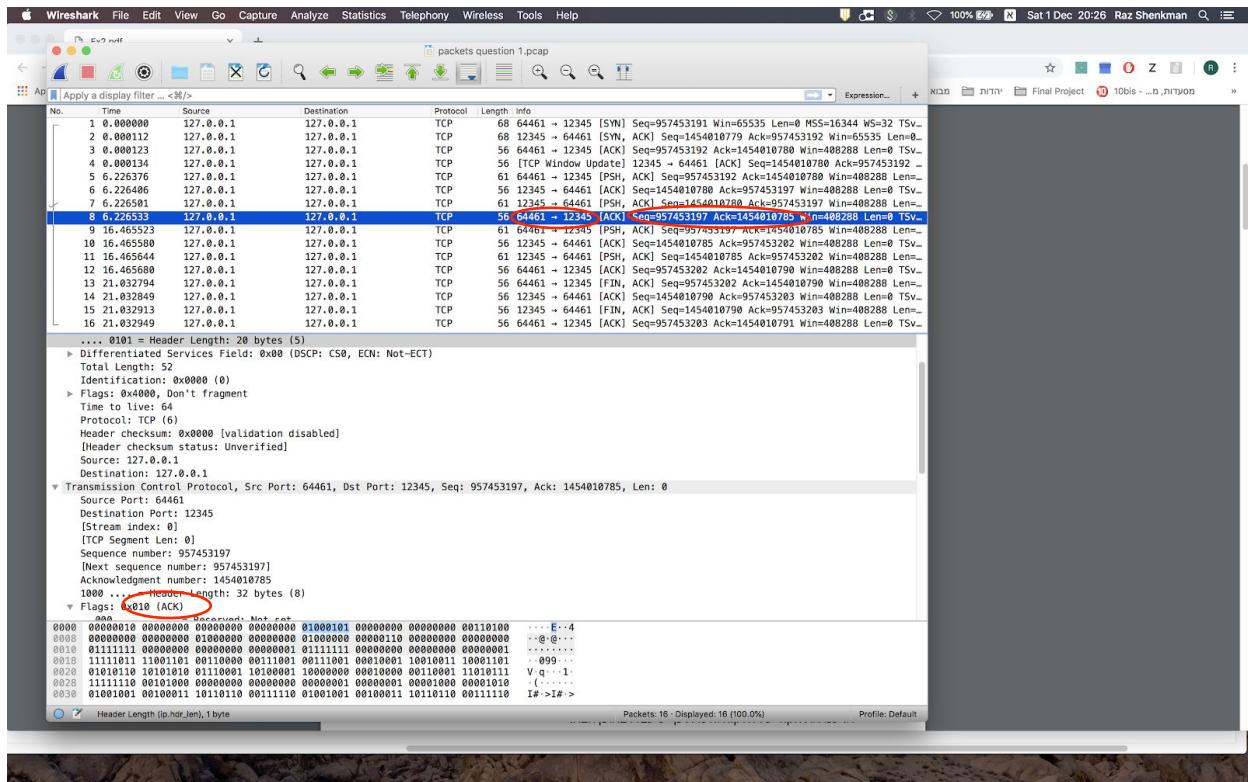
נשים לב שכן השרת (פורט 12345) שולח ללקוח (פורט 64461) הודעת ack (כיוון שדגל ack דלוק), גודל הheader tcp הוא 32 bytes והו seq number 0. הודהה השםsequence number הוא ack number ב**תוספת 5** (כמה השרת מאשר ללקוח שהוא 5 bytes ממנו).

השרת שולח ללקוח HELLO



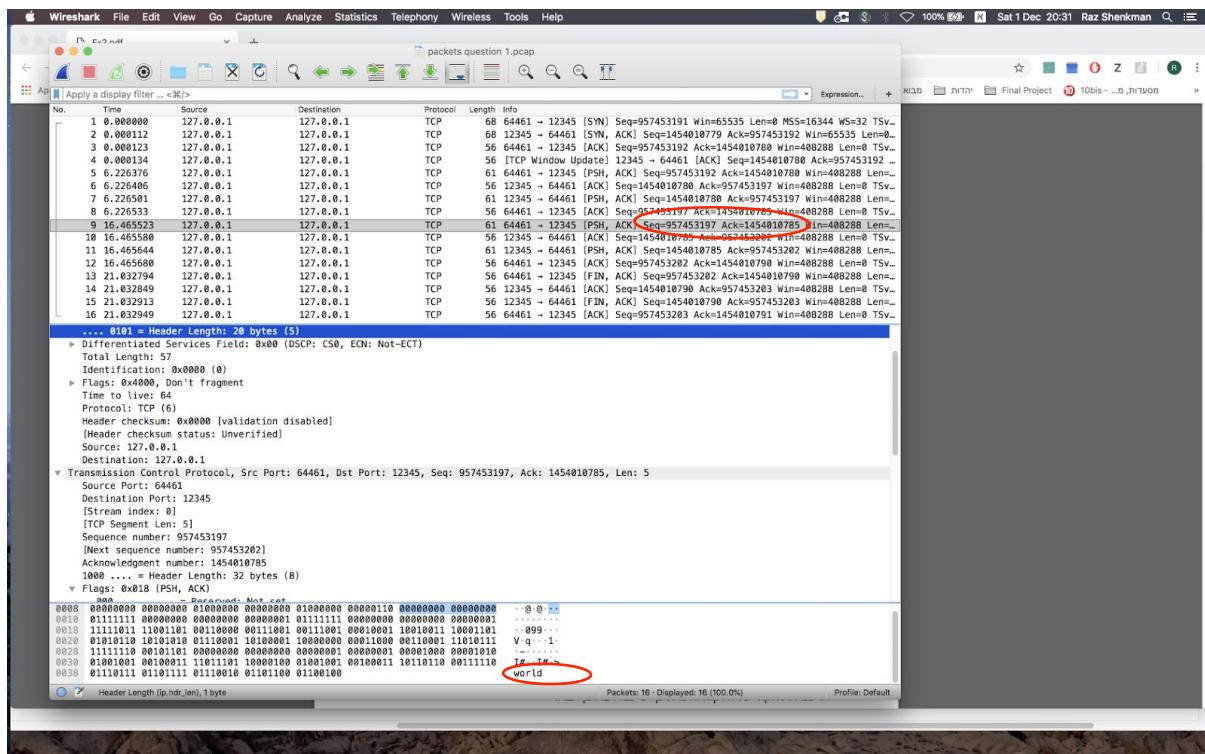
נשים לב שכן השרת (פורט 12345) שולח ללקוח (פורט 64461) הודעה שהມידעה בה הוא HELLO, גודל ה sequences הוא 5 bytes (HELLO) והוא(seq number) הולח הנושא(len) של ההודעה הוא 32 bytestcp header number מההודעות הקודמות (כי ההודעה הקודמת של ack לא משנה את seq number) והו seq number של הלקוח שנשלח מקודם .5+.

הלקוח שלוח ack לשרת:



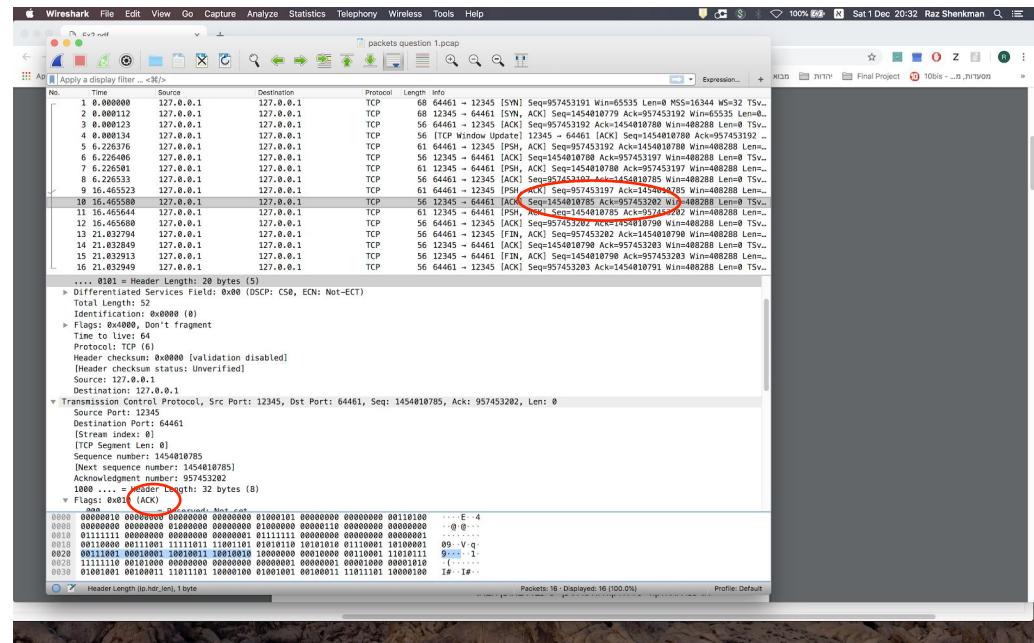
כאן הלקוח (פורט 64461) שלוח לשרת (פורט 12345) הודעת ack (12345) נשים לב שהlength של ההודעה הוא 0, והseq number הוא 12345. מתקיים שהגודל header bytes הוא 32 tcp header bytes, והseq number הוא 12345. ack flag של ack + 5 (ci) הוא קיביל אישור על השליחה שלו, אך העלה את seq number ב-5, בנוסף הקודם של הלקוח + 5 (ci) הוא קיביל אישור על השליחה שלו, אך העלה את seq number ב-5, בנוסף הקודם של הלקוח הוא seq number של הדרישה של HELLO של השרת, ולכן מוסיף 5.

שליחת world מהלך לשרת:



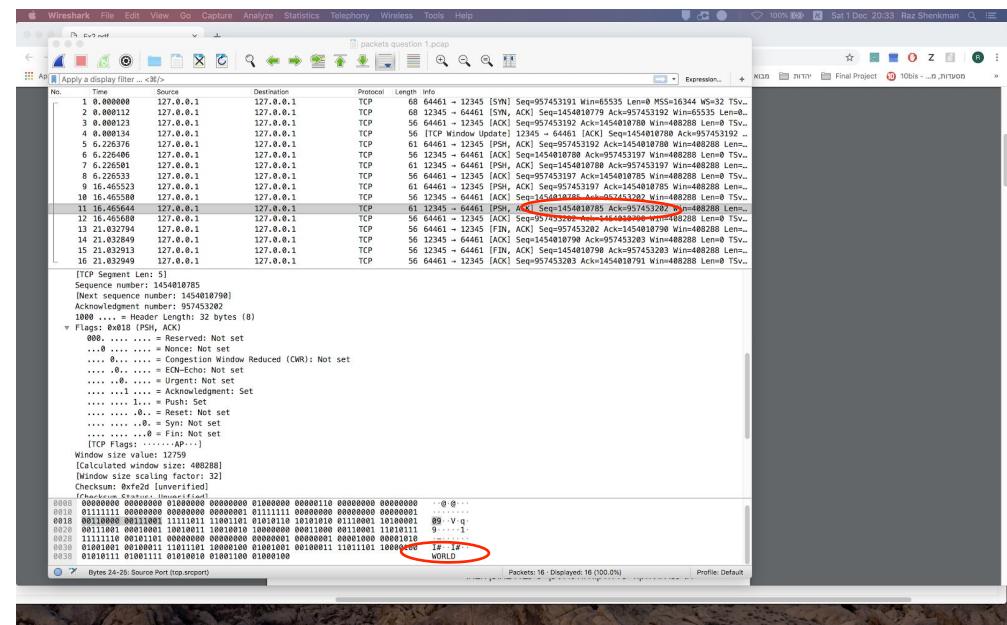
נשים לב לפורט של הלקוח source port (64461) ופורט השרת destination port (12345) נקבע ע"י מהה. וכך ב证实sequence number של הלקוח (מההודעה הקודמת שלו) והאקרוקודם (sequence number של השרת מההודעה הקודמת שלו). נשים לב שהגודל tcp header הוא 32 bytes והאsequence number הוא 5 bytes. כפי שכתוב. אורך ההודעה הוא world.

הודעת ack מהשרת ללקוח:



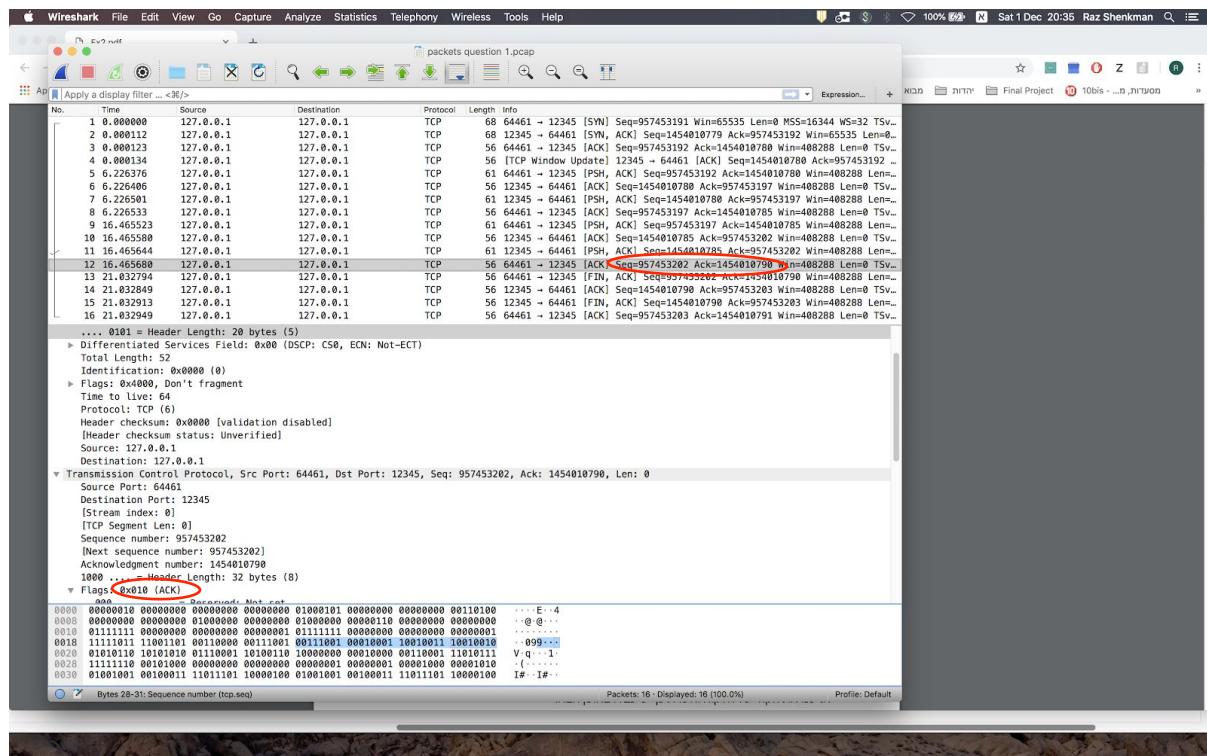
נשים לב שכן השרת (פורט 12345) שולח ללקוח (פורט ack (64461) הודעת ack (12345) של הלקוח הוא seq number 0. הוא הודהה שהוא הלקוח. השם המהוודעות הקודומות, והוא הום השם מארש ללקוח שהוא קרא 5 bytes ממנו.

הודעת WORLD מהשרת ללקוח:



נשים לב שכן השרת (פורט 12345) שולח ללקוח (פורט 64461) הודעה שהמیدע בה הוא WORLD, גודל הלקוח הוא 5 bytes tcp header + 32 bytes של ההודעה הוא 5 (נשלח WORLD (הו ה- seq number ה-sequence number של הלקוח מההודעות הקודמות (כי ההודעה הקודמת של ack לא משנה את seq number של הלקוח שנשלח ממקודם 5+), והו ack number של הלקוח שנשלח ממקודם).

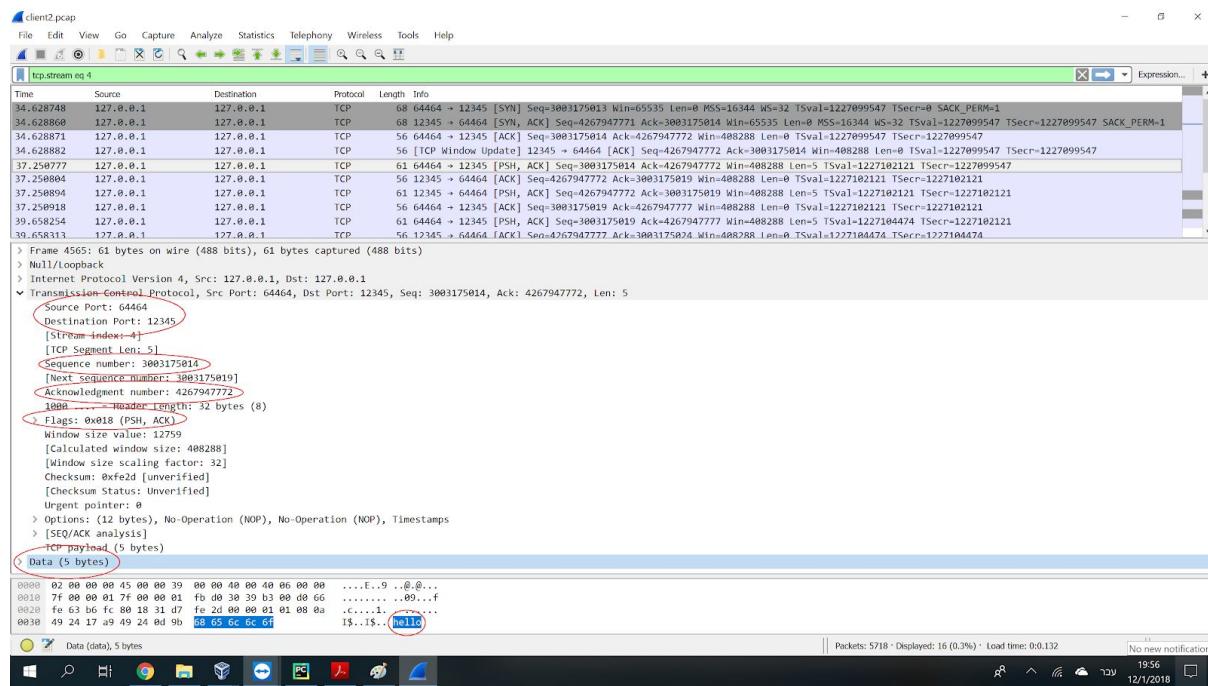
הודעת ack של הלקוח לשרת:



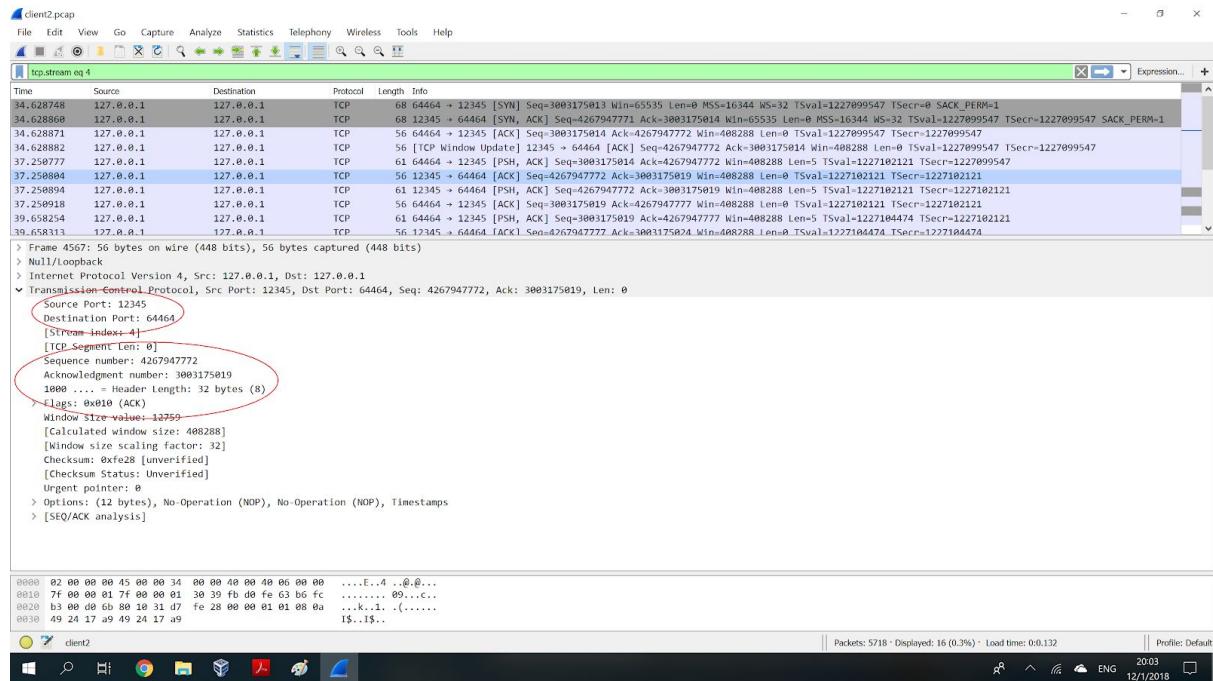
כאן הלקוח (פורט 64461) שולח לשרת (פורט 12345) הודעת ack (12345) (נשים לב שההוּדָה הוא 0, וזה seq של ack דילוק). מתקיים שהגודל header bytes, וההוּדָה הוא numbeRN, והוא numbeRN של tcp header + 5 (כי הוא קיבל אישור על השילחה שלו, לכן עלה את numbeRN ב-5), בנוסף לכך הקודם של הלקוח + 5 (כי הוא קיבל אישור על השילחה שלו, לכן עלה את numbeRN ב-5), בנוסף ה-1800 Header Length: 32 bytes (8).

ההוּדָה הוא numbeRN של השרת + (5 מוסיפים) (5).

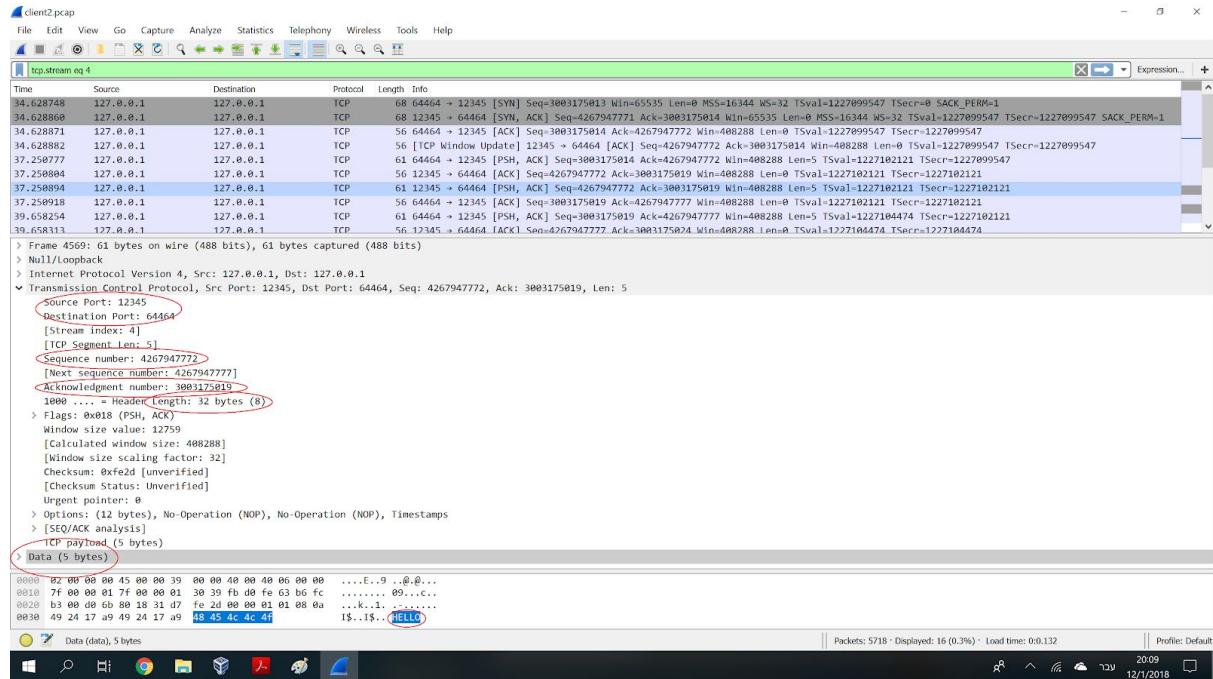
בלוקו השני:



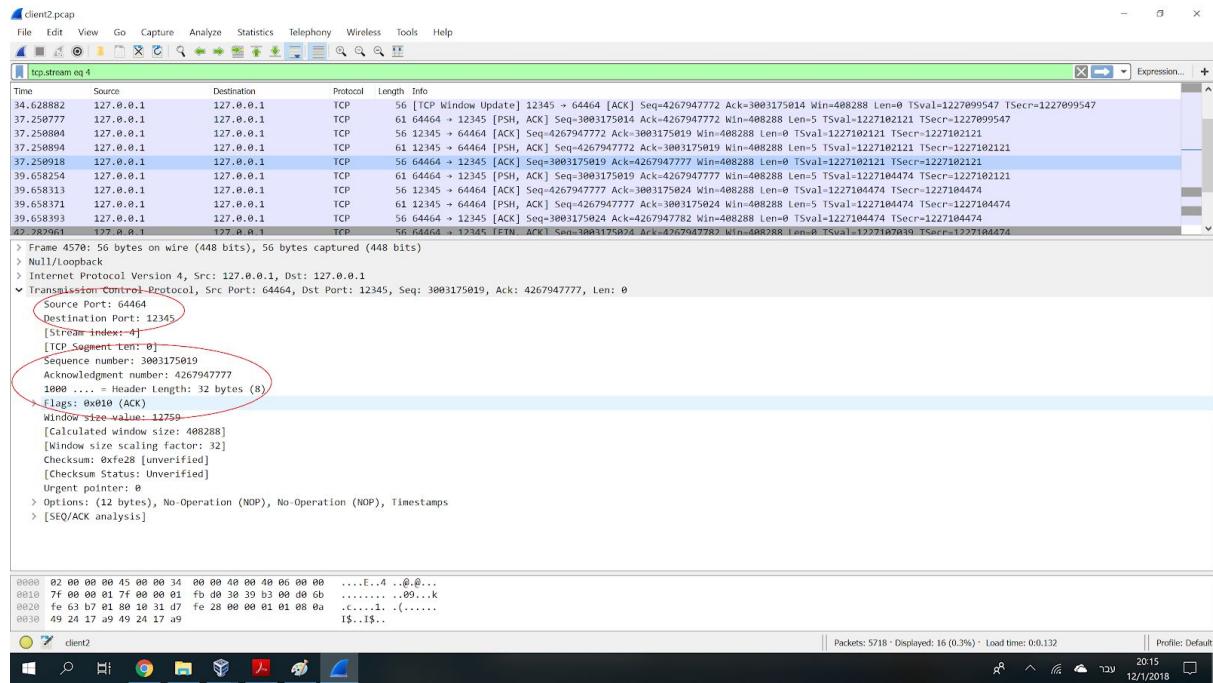
בהודעה זו הלקוח שלח לשרת hello (מוסומן למטה). נראה שהdagל של ה-ack דלוק והלקוח ביצע ACK לא seq ההתחלה(פלוא אחד) של השרת(הרי השרת לא שלח עוד מידע ולכן אצל הלקוח לא התקבל עוד מידע ממנה), ו-ה seq number זהה ל seq number ההתחלה של הלקוח(שעדין לא שלח מידע) פלאו אחד. ניתן לראות שה포רט של היעד(השרת) הוא 12345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



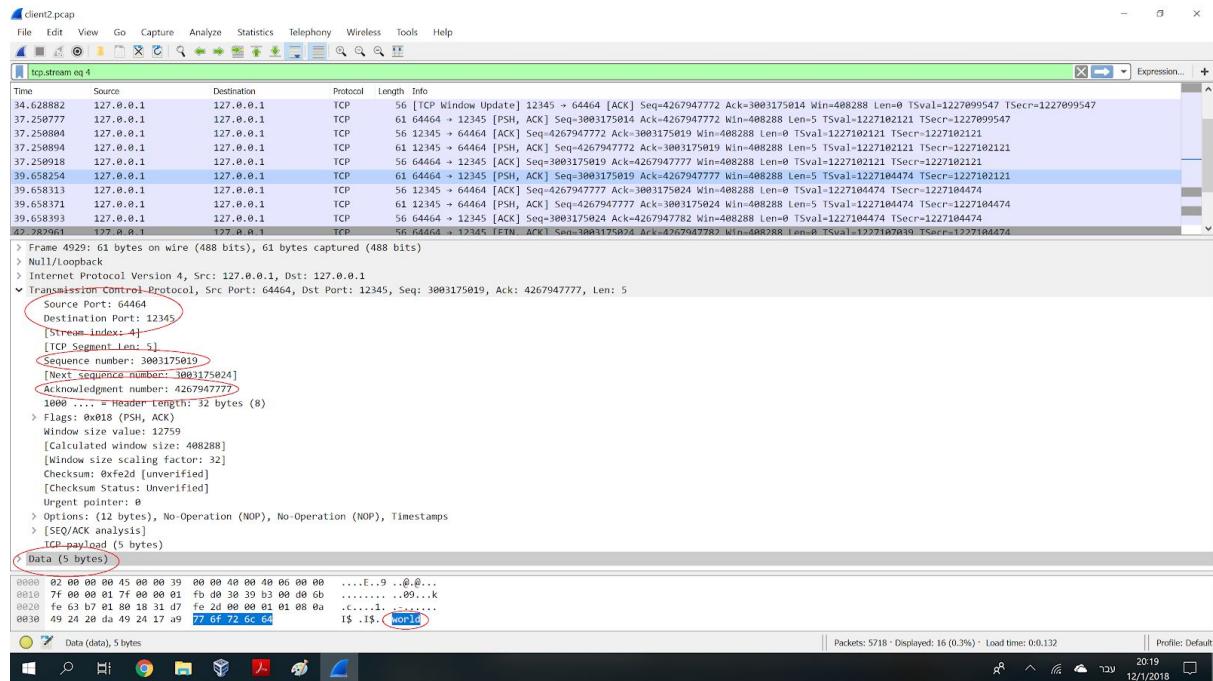
ההודעה הבאה התקבלה מהשרת ללקוח, ניתן לראות שפורט המקור שלזהה לפורט של השרת ופורט היעד זהה לפורט של הלוקו. ההודעה היא הודעת ACK(הציג דלוק) על הודעת hello שנשלחה מהלקוח - קל לראותות ש ה-ACK number זהה ל- seq number+5 (הריהודעה Hello בגודל 5) כולם ההודעה התקבלה במלואה אצל השרת. ה- seq number זהה להתחלתי של השרת, הריהוא עוד לא שלח מידע.



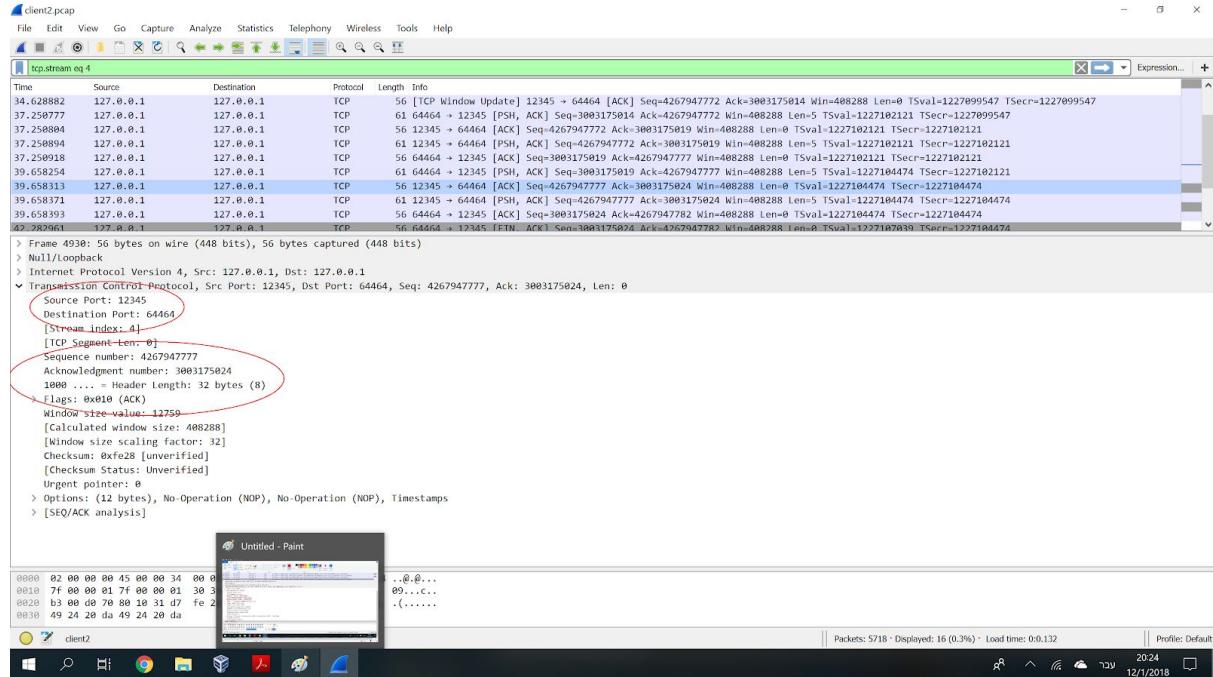
הודעה זו (לפי הפורטים מהשרת ללקוח) היא ההודעה בה השרת כותבת **HELLO** ללקוח. ה **seq number** נשאר זהה למה שהיota בזמן הקמת החיבור כי רק עכשו השרת מתחילה לשולח מידע (וכשישלח המודיע הבא הוא יתחל מ ה **seq** הבא, שרשום מתחת ל **seq** הנוכחי). ה **ACK** זהה ל **ACK** מהחבריה הקודמת(למקרה שהלקוח לא קיבל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



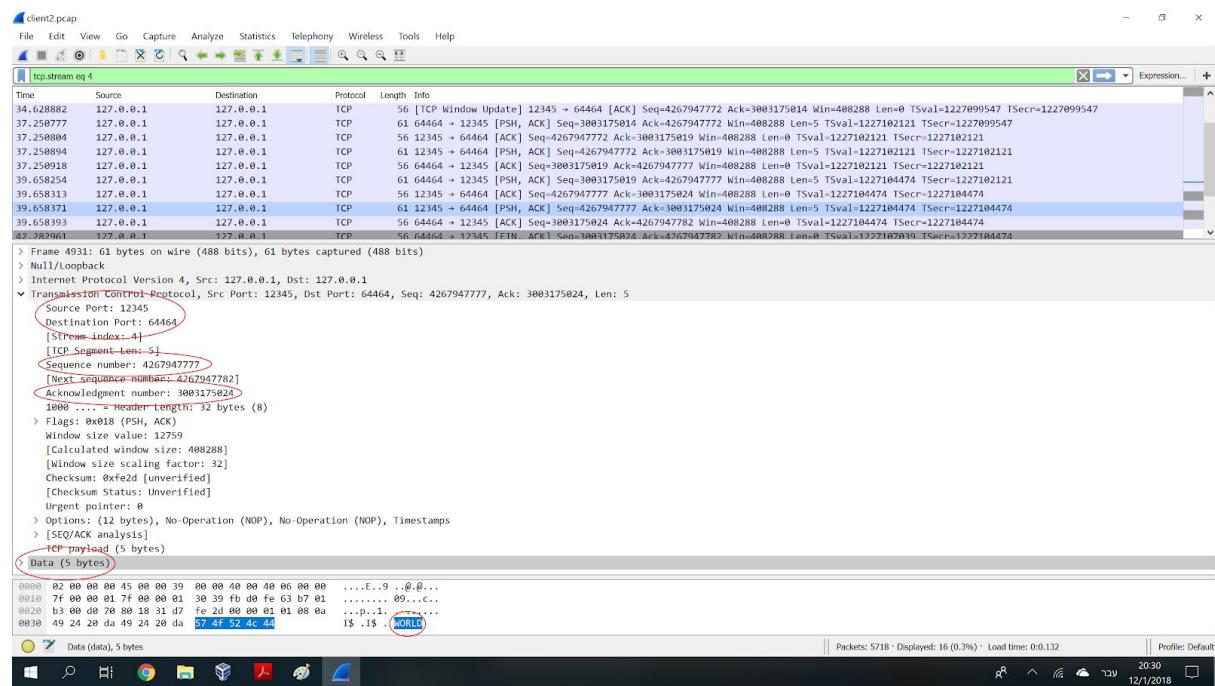
בהתודעה זו (מהלך לשרת, כאמור לפי הפורטים), הלוקו שולח ACK לשרת על ההודעה Hello (התקבלה ניתן לראות שדגל ה ACK דלוק) ו ה ACK number שווה ל 5 + initial seq number של ה-server (וההודעה היא מוגדלת 5 שכן זה אומר שההודעה התקבלה במלואה אצל השרת). ה Seq number עלה בחמש + 5 (client's initial seq + 5) כי הלוקו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



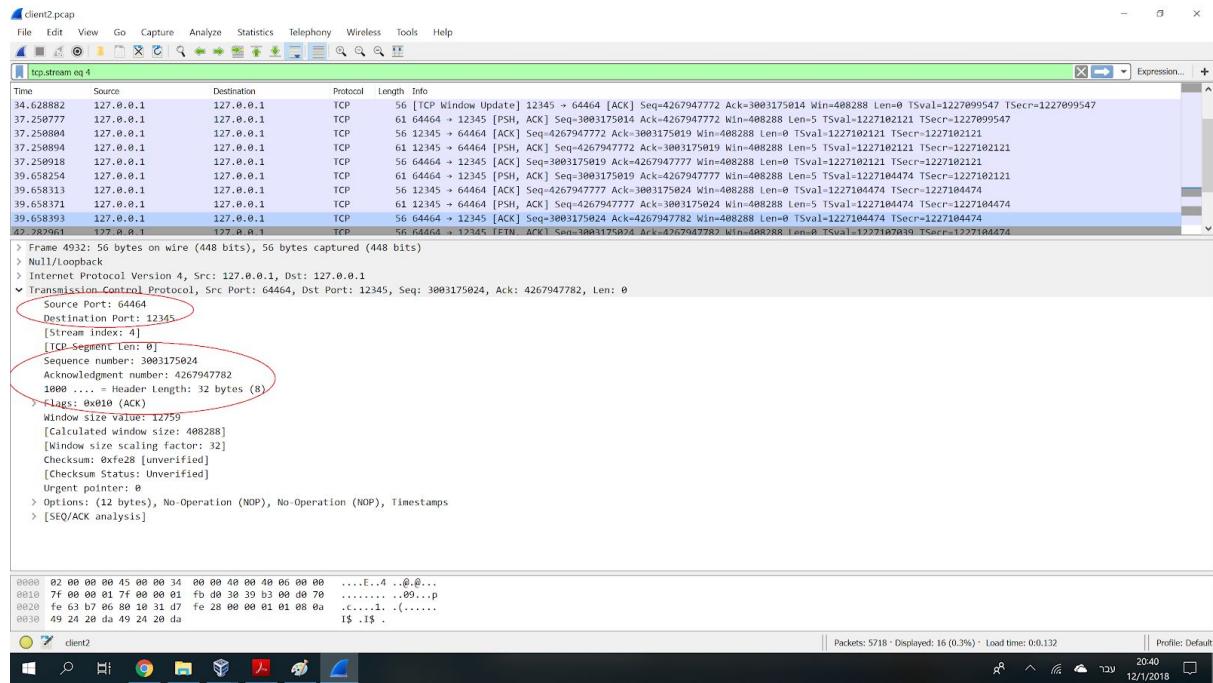
הודעה זו (**לפי הפורטים**) היא מהלך לשרת, ומכליה את המידע המוסף `world`. כמובן שאין שינוי הגיע בseq שווה ל 5 + initial client seq + 5 bytes (כלומר ההודעה הבאה אחראית חמשת התווים של `Hello` שנשלחת כרגע) ואין שינוי בACK כי לא התקבל עוד מידע מהשרת.



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 (ו�택בל גם הודעת ה-HELLO).



הודעה זו מהשרת ללקוח(ניתן לראות לפני הפורטיטם) מכילה את המילה WORLD כדי שניתן 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 הודעה ה-HELLO).

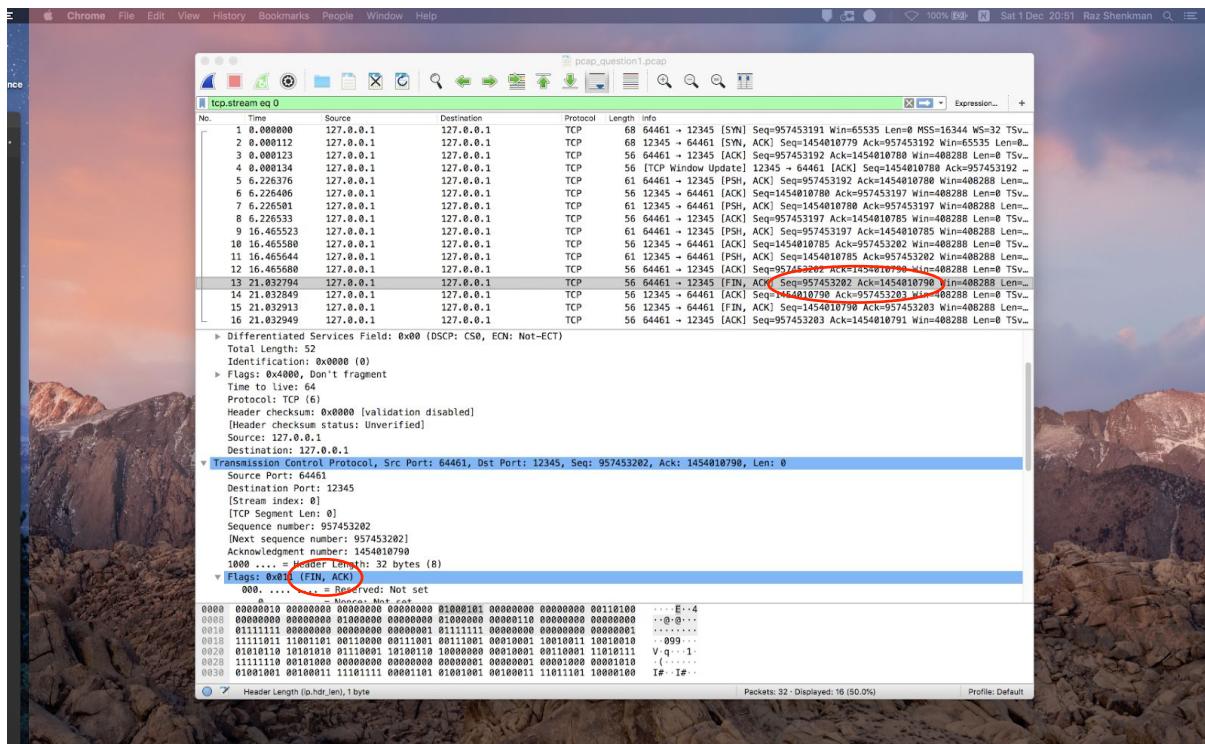


ההודעה الأخيرة - מהלכות לשרת (פורט המקור 64464 שייך ללקוח). הלקוח שלח ACK על ההודעת WORLD של השרת, קל לראות ש ה-field ACK number שווה ל-field seq number שהלכו שלח בהודעת הקודמת פלוס 5 (כלומר הודעת WORLD שאורכה 5 התקבלה במלואה). ה-field SEQ number של הלקוח עלה ב 5 מאז הפעם האחרונה שראינו את הלקוח שלח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

(ג) 1

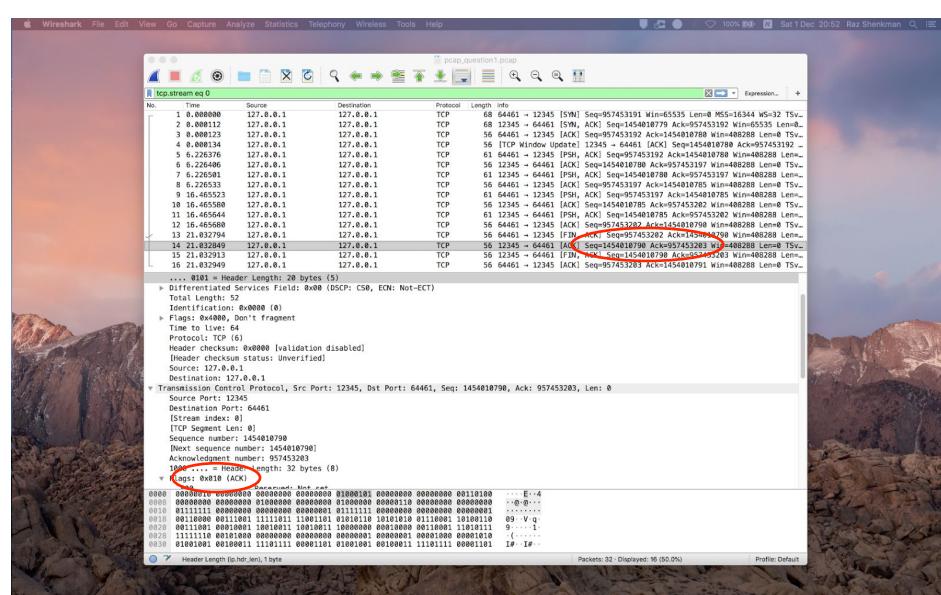
עבור הליקות הראשונות:

can the ACK be sent back to the connection?



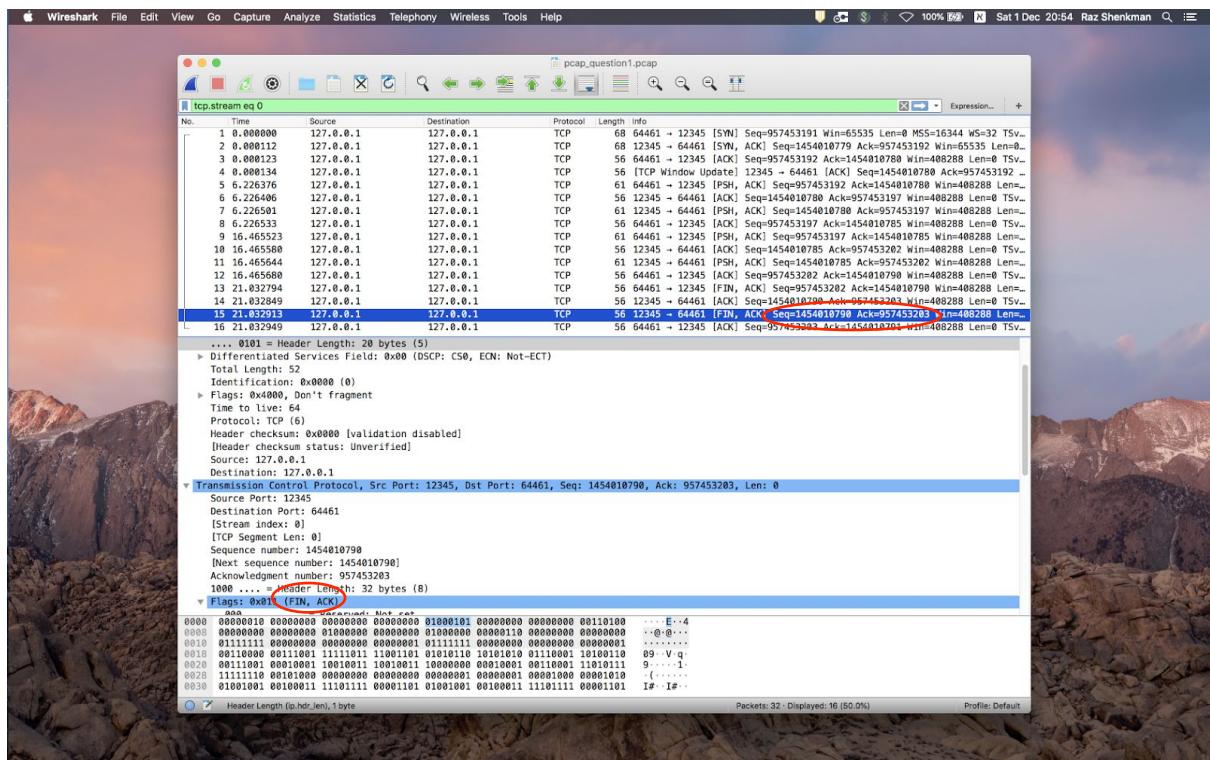
נשים לברר מה הערך של ה-ACK (12345) ושל השרת (64461), שגודל ההודעה הוא 0 וshallוק בה דגל fin.

the ACK sent back to the connection?



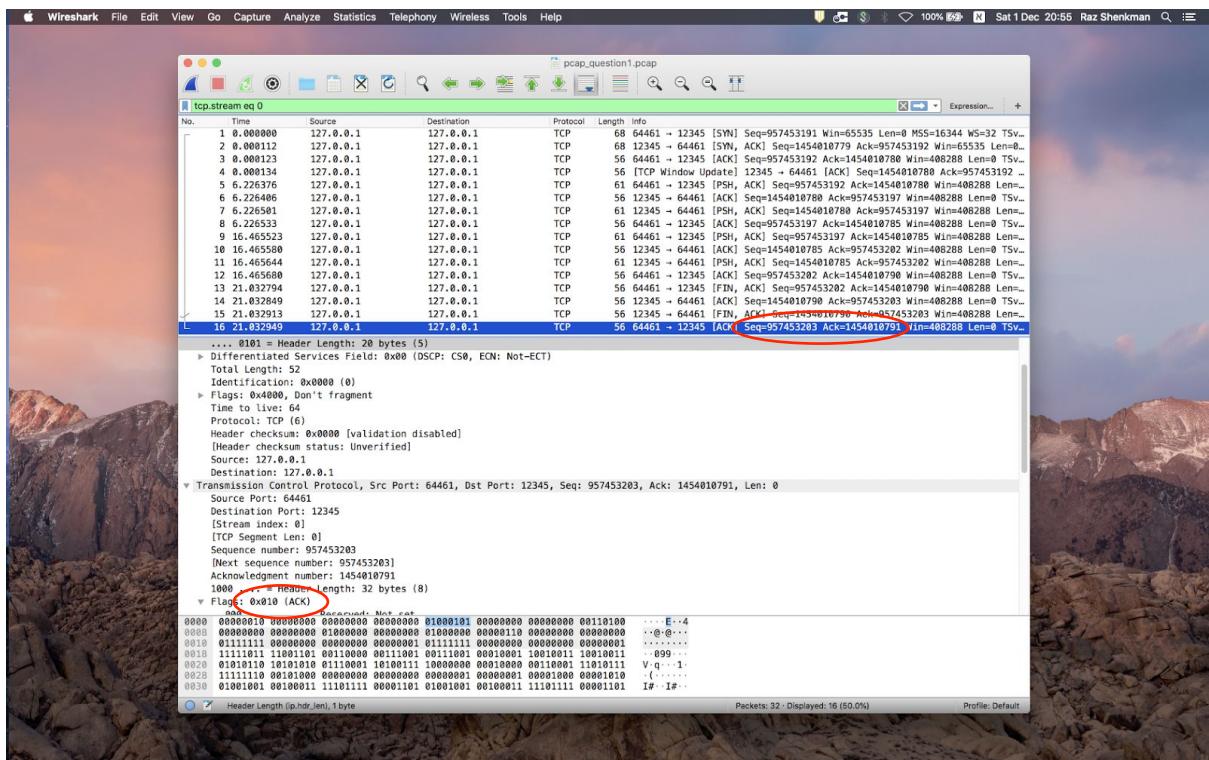
נשים לברר מה הערך של ה-ACK (12345) ושל השרת (64461), CAN THE ACK BE SENT BACK TO THE CONNECTION?

השרת שלוח ללקוח גם בקשה לסיום החיבור: connection close request



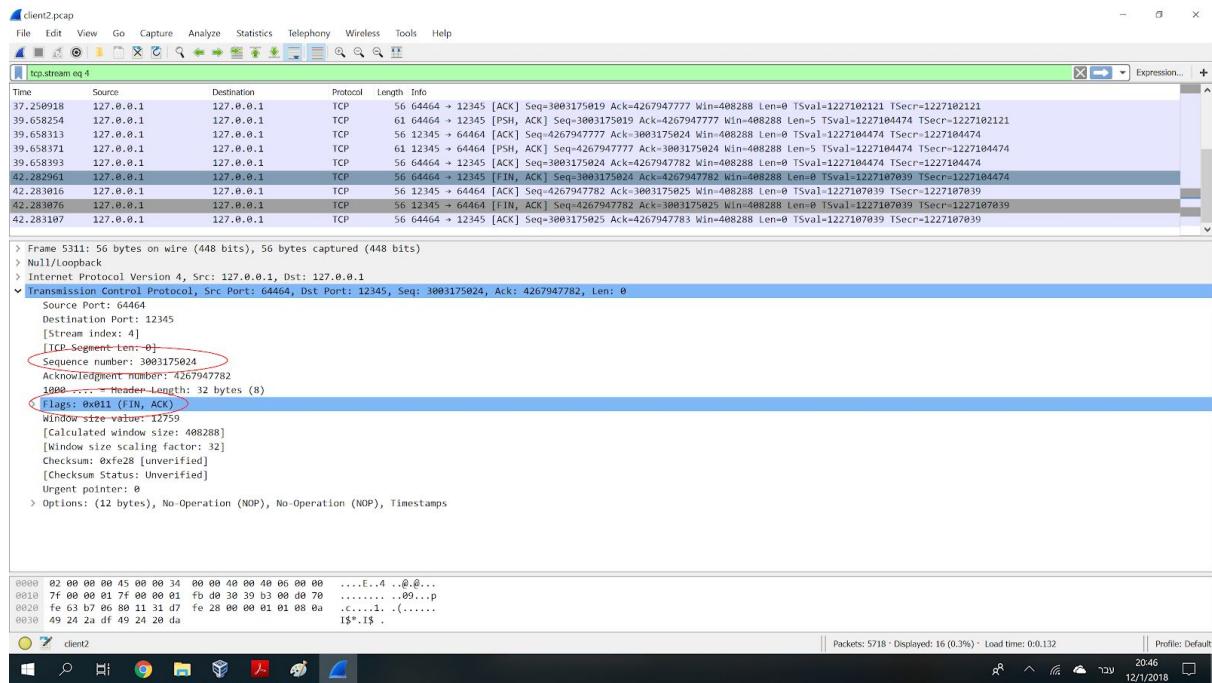
נשים לב לכך שהשרת שלוח ללקוח (זזהה לפני הפורטים) בקשת FIN (דגל החסן דלוק).

הלקוח שלוח לשרת ack:

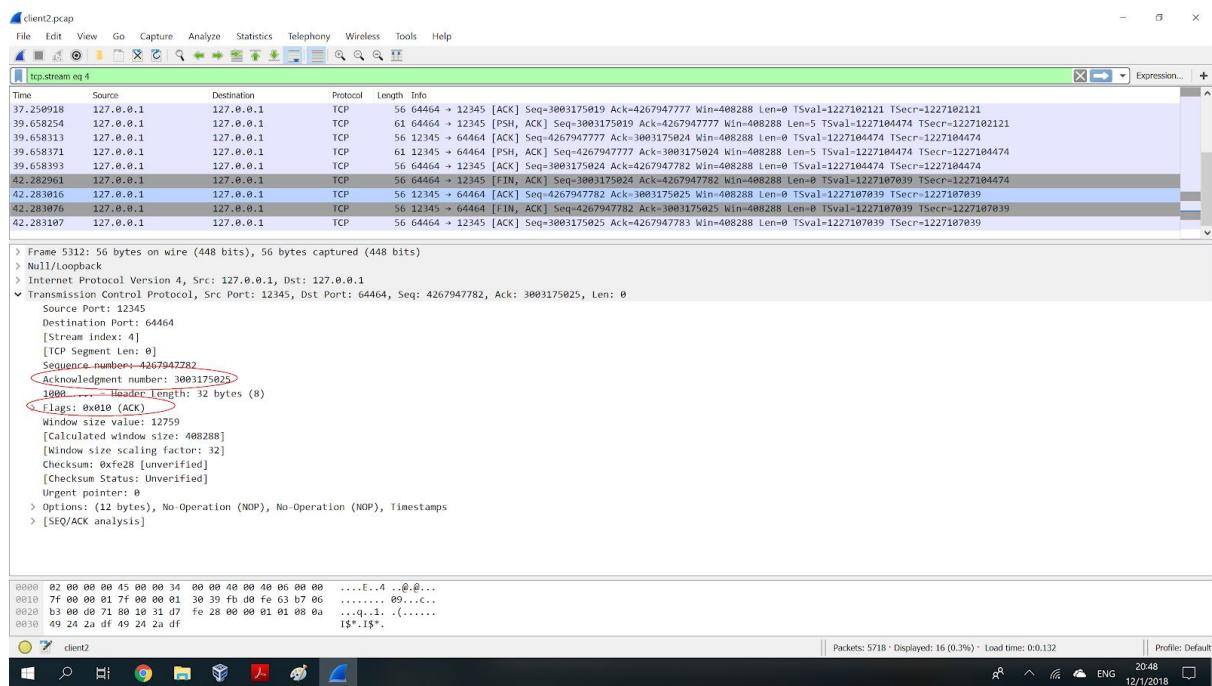


זהו בעצם ההודעה האחרונות בconnection, כאן הלקוח שלוח לשרת (נזהה לפי הפורטים) הודיעת ACK כשר האck num הוא seq number + 1 (כי נשלחה בקשה fin) והseq number של הלקוח הוא ה מההודעה האחרונות של הלקוח לשרת + 1 (כי הוא קיביל אישור על ההודעה שלו מוקדם).

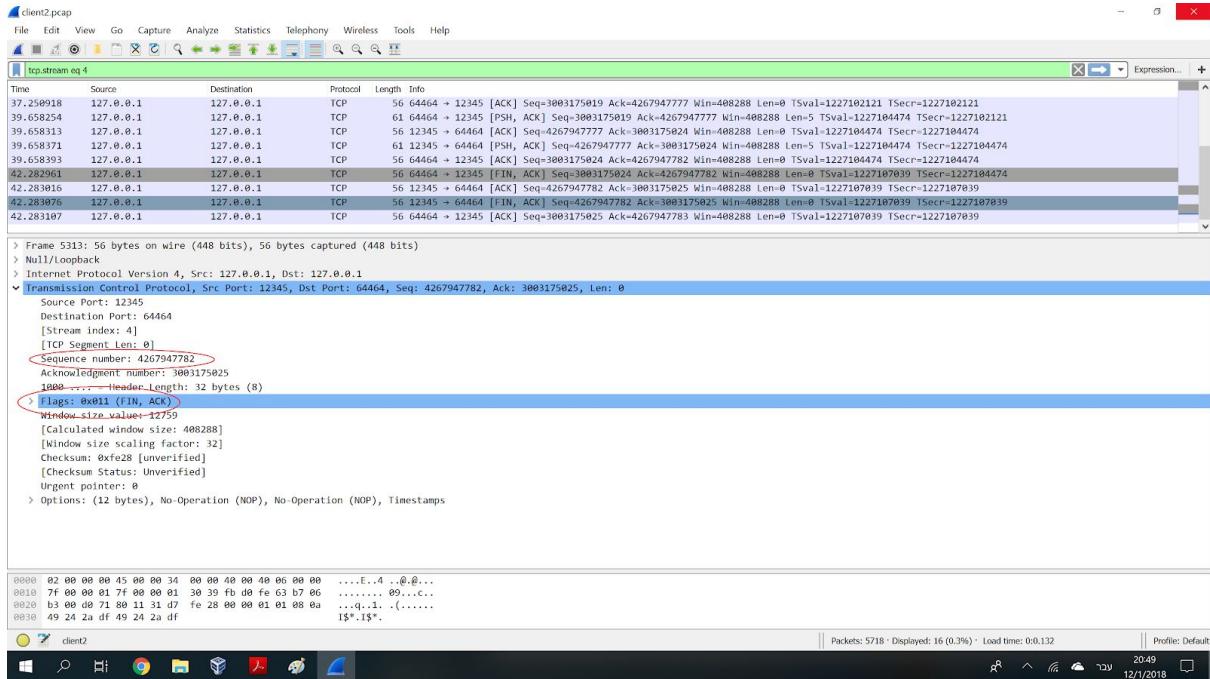
עבור הליקון השני:



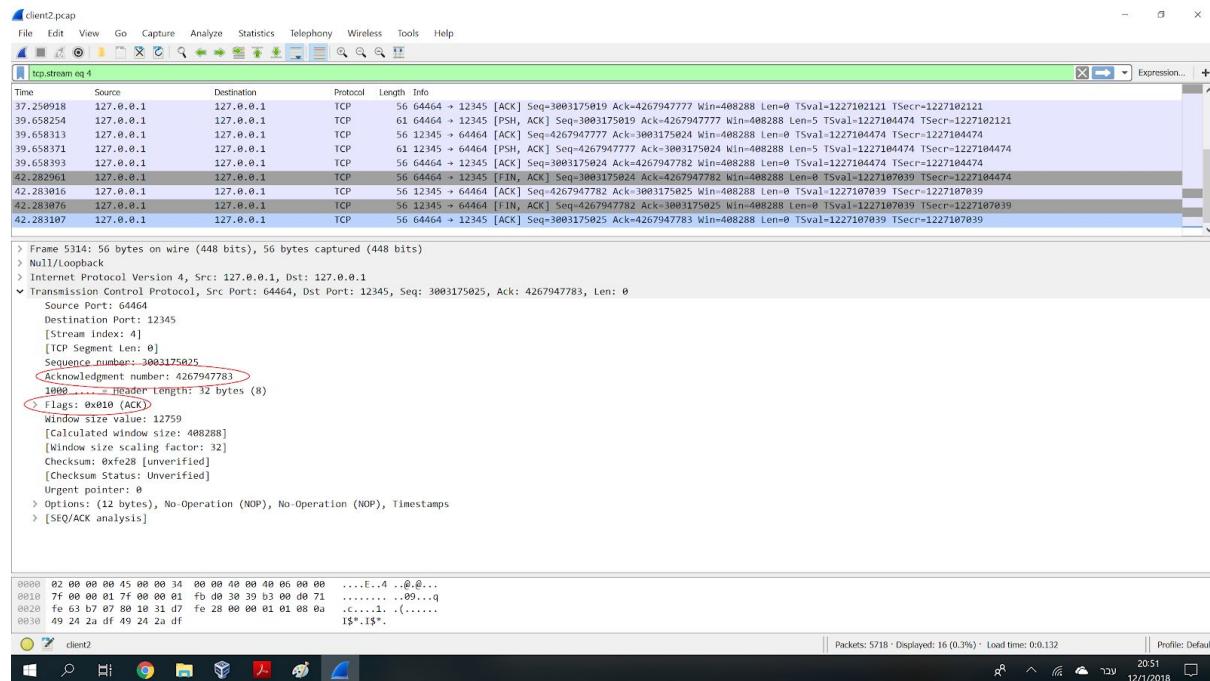
כאן ניתן לראות שדל ה FIN דולק. זהה הודעת על הריסת החיבור שהליקון שלוח לשרת. נשים לב ב seq number ונסתכל בהודעה הבאה:



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:

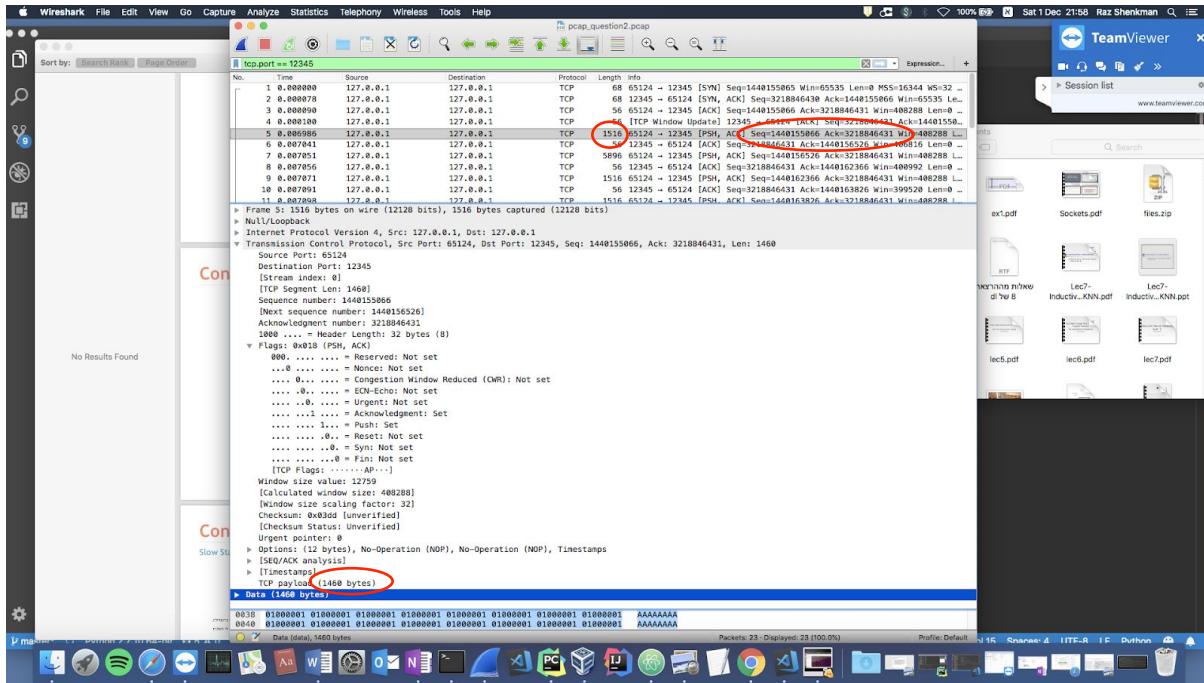


ההודעה הבאה היא הودעת ACK על ה FIN שהשרתשלח. ניתן לראות ש ה ACK number שווה ל seq number של FIN שהודעתה ב握手.

שאלה 2

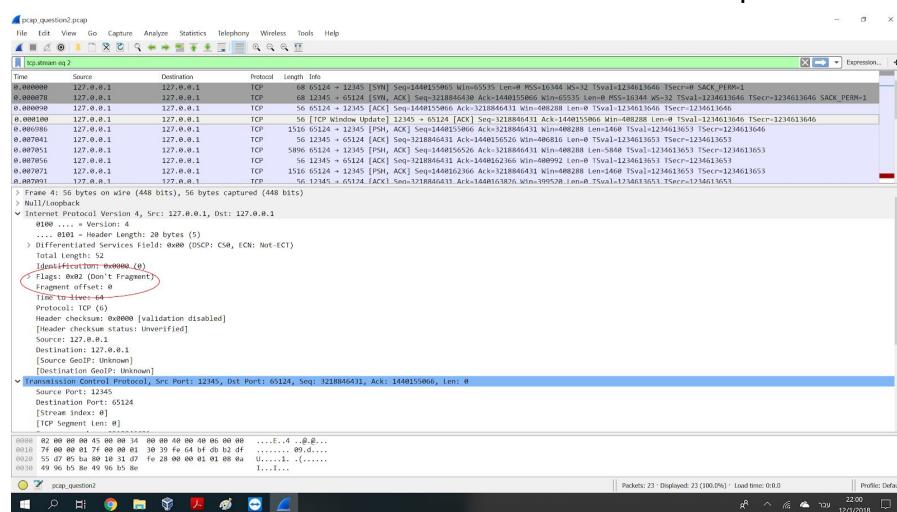
(א) 2

נתבון בחבילה שהל��ו שלח לשרת:



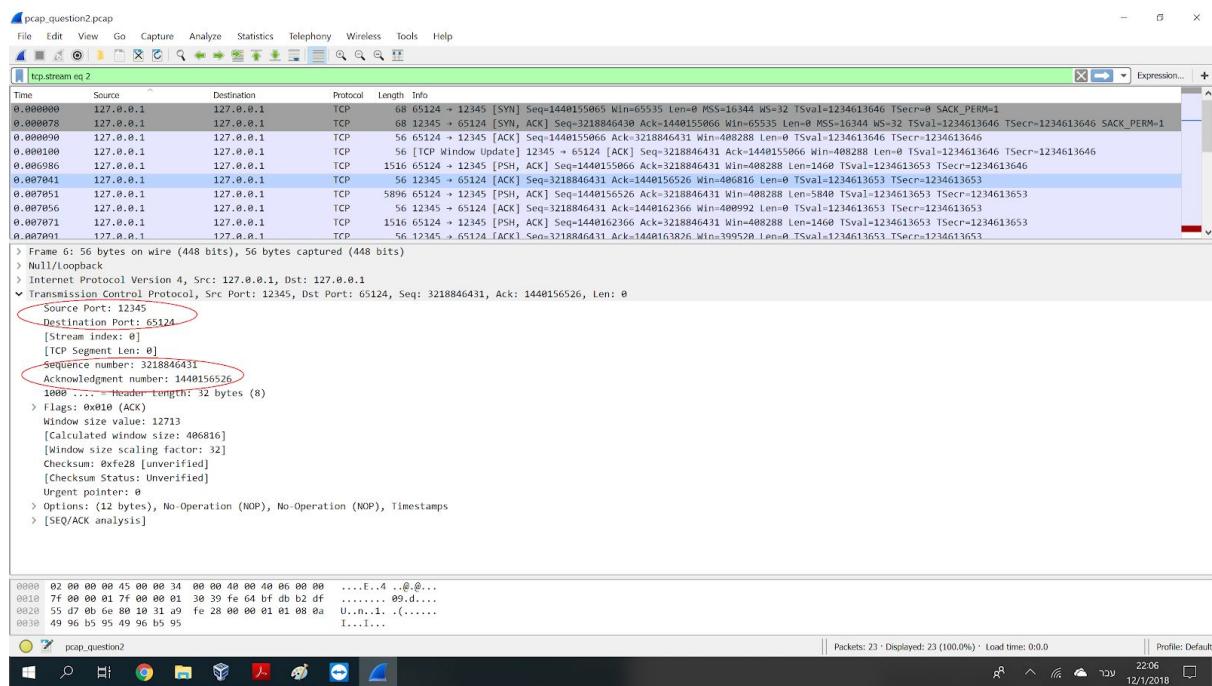
סה"כ הלוקו ציר לשלוח 15000 בטים. בהודעה הראשונה נשים לב שהלוקו שלח מיד של 1460 בתים (סיה"כ 1516 עם 1516 עט 1516 עם headers) כאשר כל המידע הזה הוא A, נשים לב שההבדל בין seq number לבין ההודעה זו (הודעה מס' 5) להודעה הבאה שהלוקו שלח (הודעה מס' 7) הוא 1460 (כי הלוקו הגדיל את השם בטעות הפקט ה-ack של הרשת), נשים לב שיש את הדגל dont fragment בשכבות ה-seq number .network

נשים לב שבאף אחת מההודעות לא הייתה פרוגמנטייה בשכבה הרטשת:



בכל ההודעות של רצפי A בשכבה הרטשת גם הדגל don't fragment וגם מומן ה-don't fragment offset הוא תמיד 0 כי אין פרוגמנטייה (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.

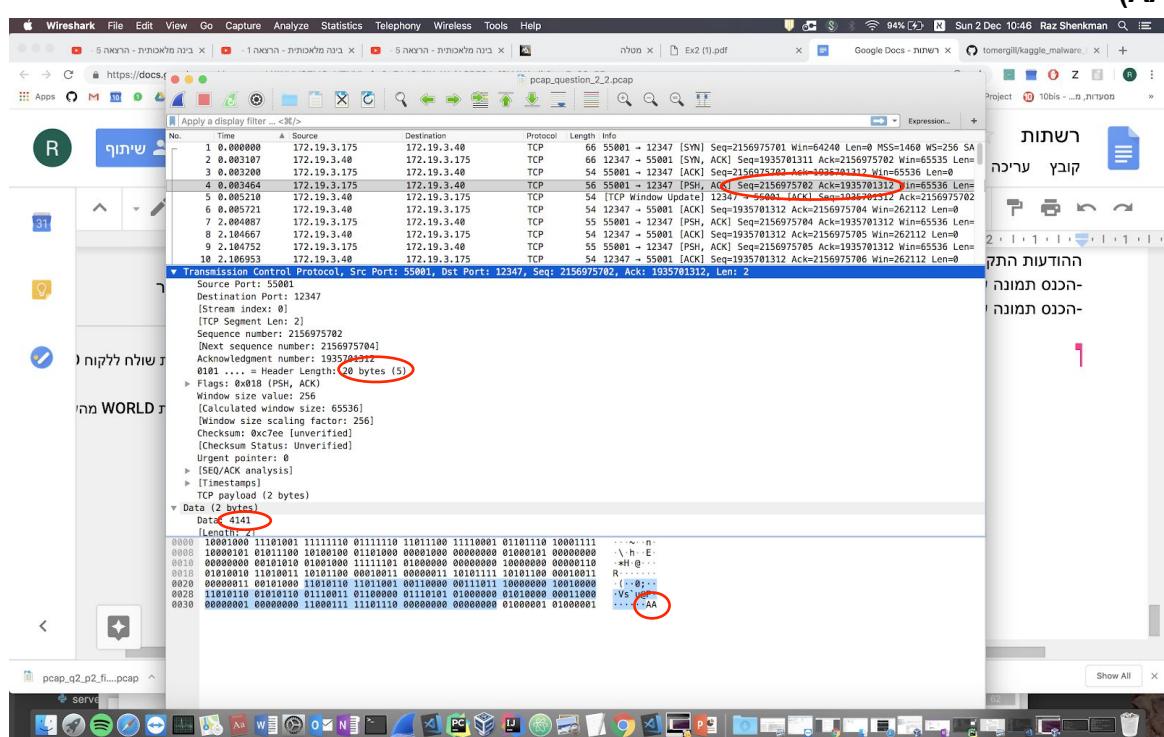
(ב)

תהליך handshake בדומה לסעיפים קודמים שהסבירנו. נשים לב שהקוו של הלקוח הוא 172.19.3.175 ושל השרת בפורט 12347, השרת בפורט 12347, הלקוח בפורט שירות של מ"ה שהוא 55001.

בתהליכי התקשרות בין הלקוח לשרת, השרת קיבל הודעה AA (שני A רצופים באותה הודעה) ובשאר ההודעות התקבל כל A בpared.

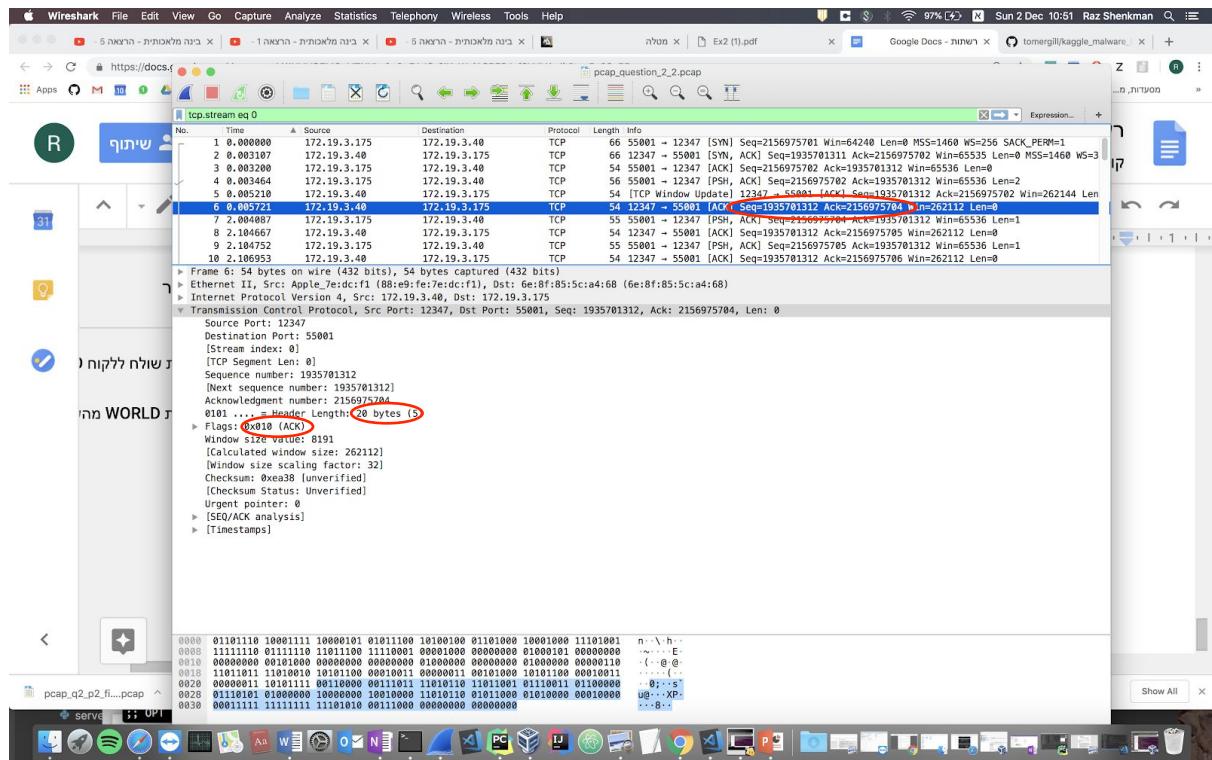
```
Connection from: ('172.19.3.175', 55001)
Received: AA
Received: A
Client disconnected
```

כאן רואים את החביליה הראשונה שנשלחה לשרת: AA (וакן data הוא 2 בתים- 4141 בזוויתasci שדה (AA)



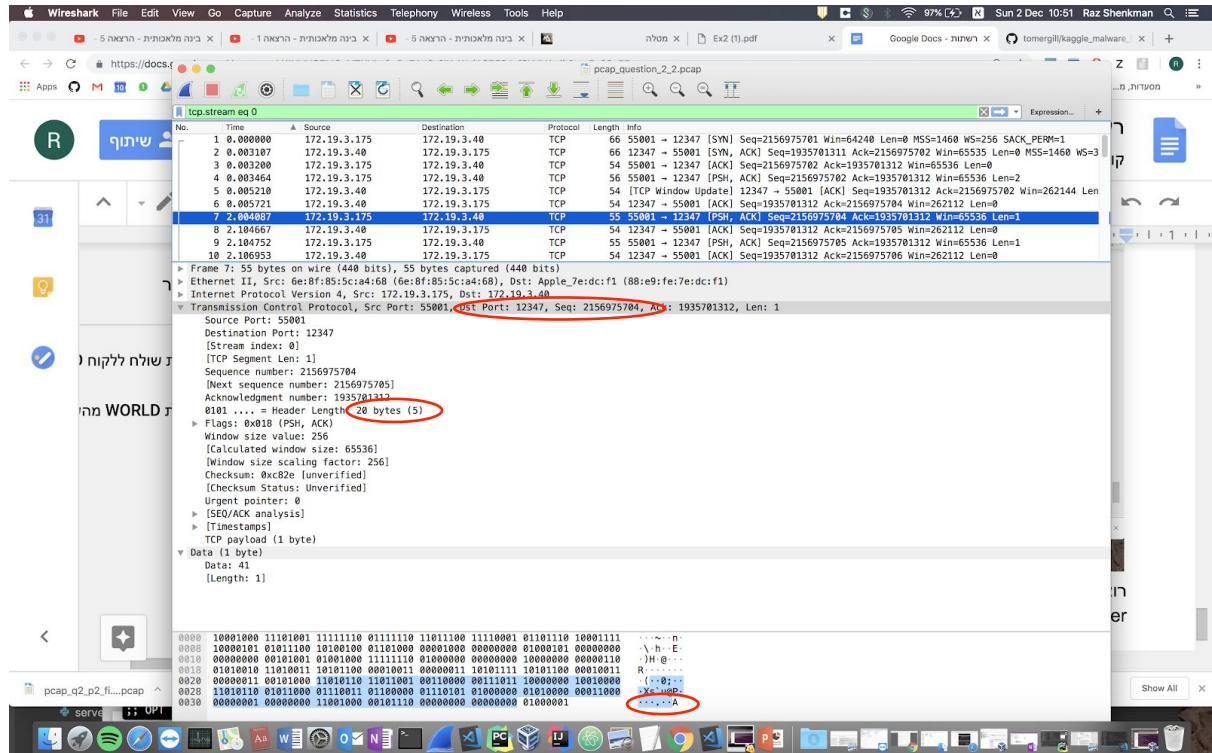
בהתודעה זו ניתן לראות את שליחת ההודעה AA מהלוקה לשרת. אנחנו יודעים שבקדוד הלקוקה ה send מבוצע פעמיים בנפרד, אך כמו שניתן לראות ב data של ההודעה השליחות "התלכדו" להודעה אחת והميدע נשלח הפעם נשלח בהודעה אחת.

נשים לב ש ה seq הוא 2156975702, ונוטכל בצלום הבא:



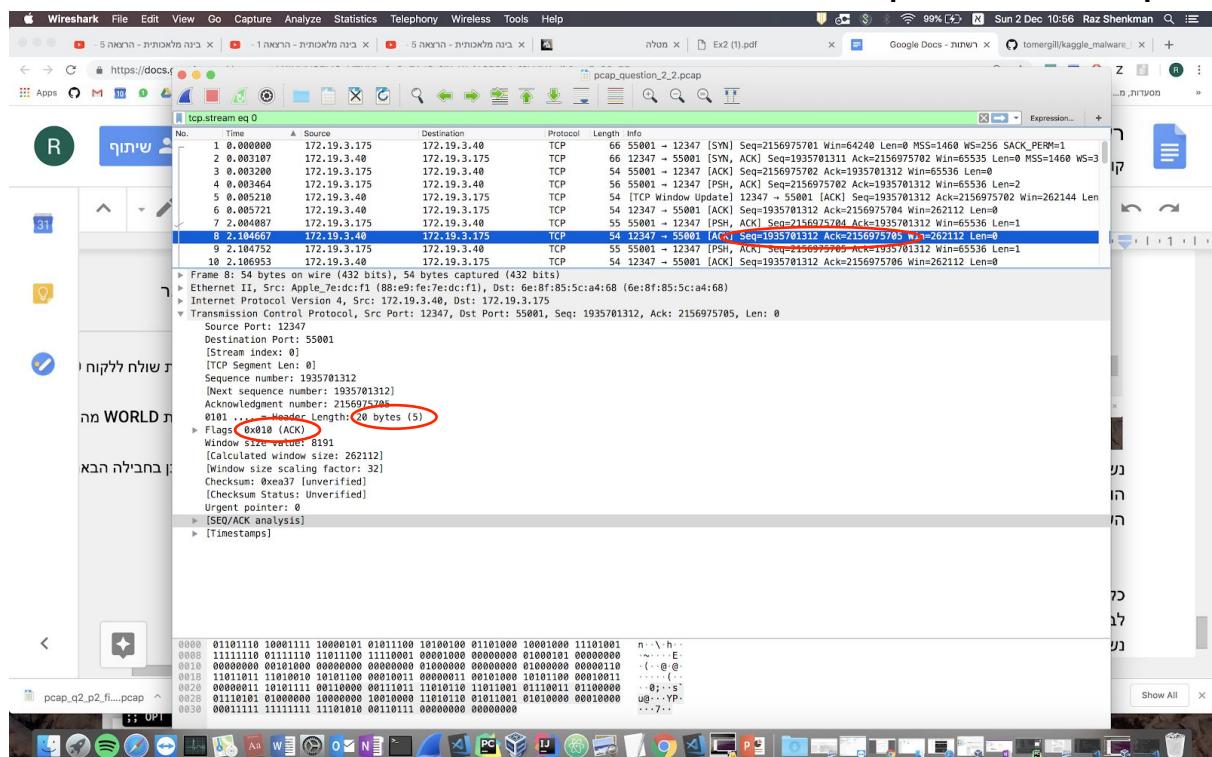
ראים שהשרת שולח ללקוקה ACK עם seq number 2156975704 (כולם גודל ב 2 מ ה seq number של הודעת AA שאורכה 2 - כולם ההודעה התקבלה במלואה בשרת, והתקבלה כהודעה אחת כנראה הרי נשלח עליה ACK נפרד).

נתבון בחבילה הבאה, נשלח A מהלך לשרת:



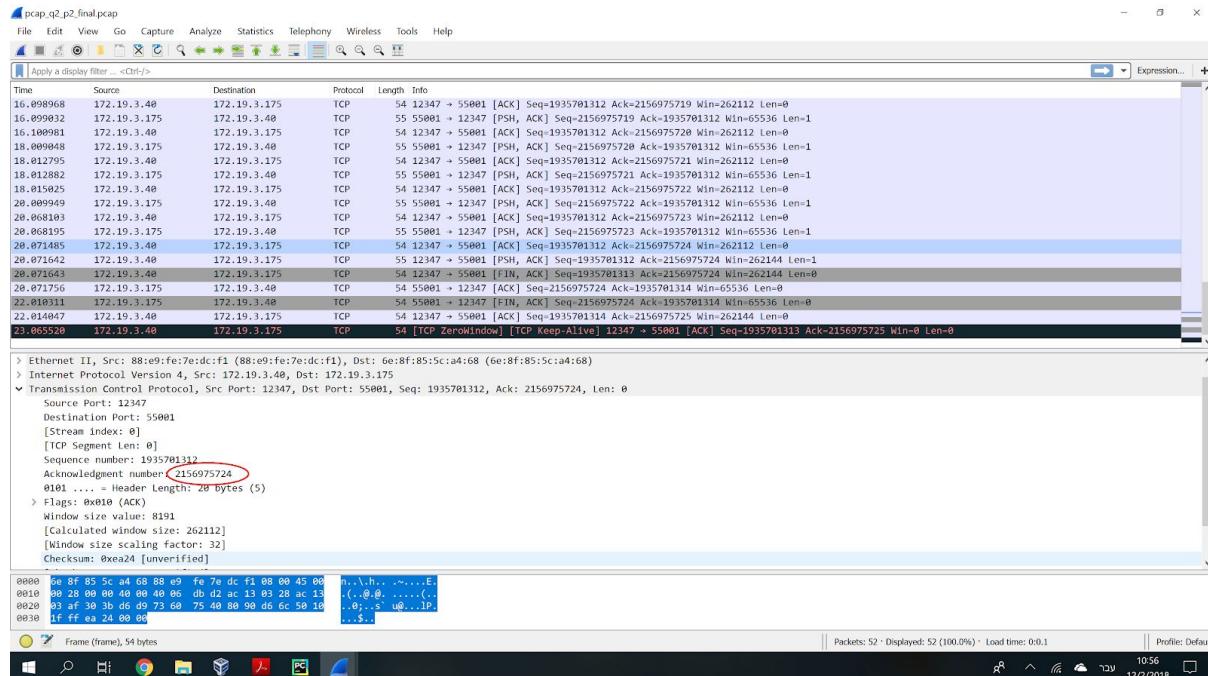
נשים לב שהגודל ההודעה הוא 1, מתקיים שנסלח A מהלך לשרת (גודל הdata הוא 1), גודל header הוא 20, הseq number הוא 2156975704 והאck הוא ack number 2156975705. השתנה מהשליחה הקודמת של הרשת (כי נשלח רק ack).

נתבון בacky של הרשת ללקוט:

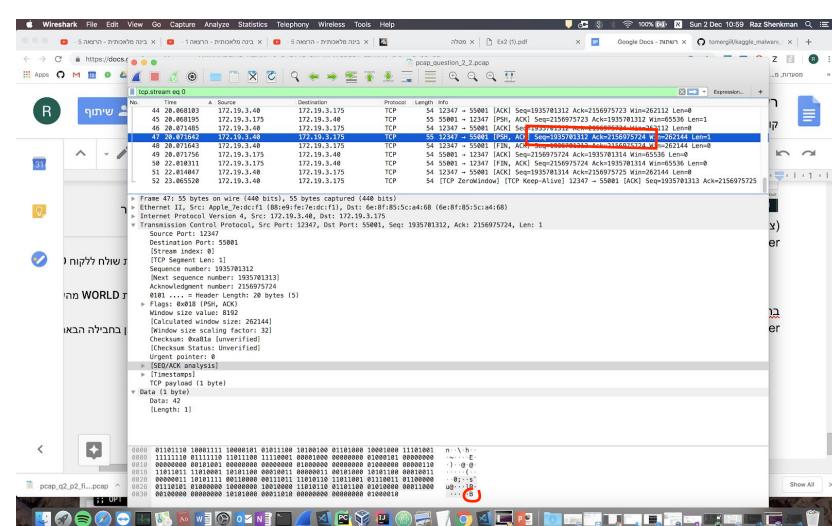


נשים לב שהacky הוא 2156975705, לעומת הseq number 2156975704 של הלוכ'h מקודם +1, כך הרשת מאשר ללקוט שהוא קרא מידע באורך בית אחד. נשים לב שהגודל של header tcp הוא 20, דגל ack דלוק.

כל הabiliaות הבאות גם זיהות ב data שליהן(כלומר שלוחות רק A אחד) וכך נחסוך מלהראות את כלן. נשים לב שבהודעה האחורונה(ACK מהשרת על ה A האחרון), ה number seq של הлокון גדול ב 22 ממה שהיא בהתחלה, מה שמשמעותו שakan נשלחו בכל ההודעות ביחד סה"כ 22 תווים. תוצאה זו(numbert)seq שגדול ב 22(22) צפוייה לתקבל לא משנה אם שתי הודעות send מתלכדות או נשלחות בנפרד שחרי בכל מקרה יישלחו סה"כ 22 בתים.



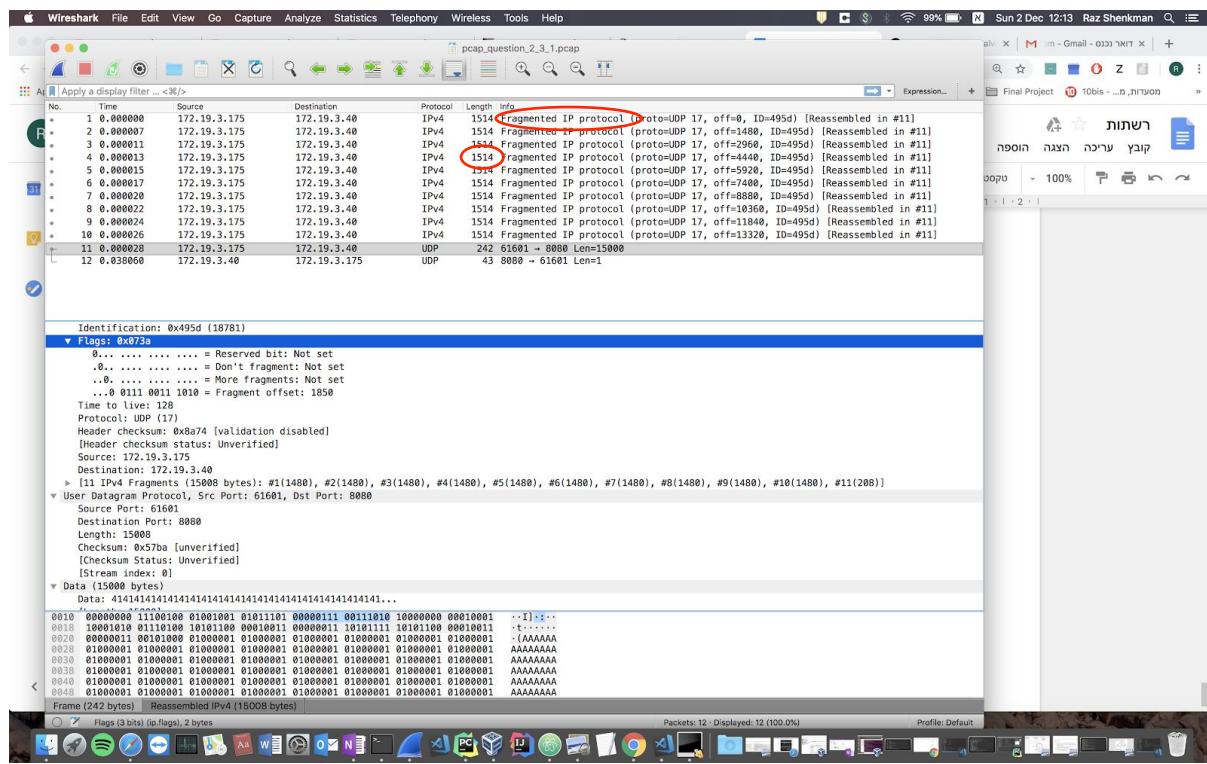
(א) אם ACK number של השרת על ההודעה האחורונה של הлокון, כלומר להיות שווה בערך לseq number של הлокון בחלק זה.



(ב) אם ACK number של השרת על ההודעה הבאה יחזיר לו הлокון ack שעריך numbert seq של גודל ב 1 מערך ה 1 מערך הлокון.

(1)(ג)

שליחת A 15000 מהלך לשרת בקען:



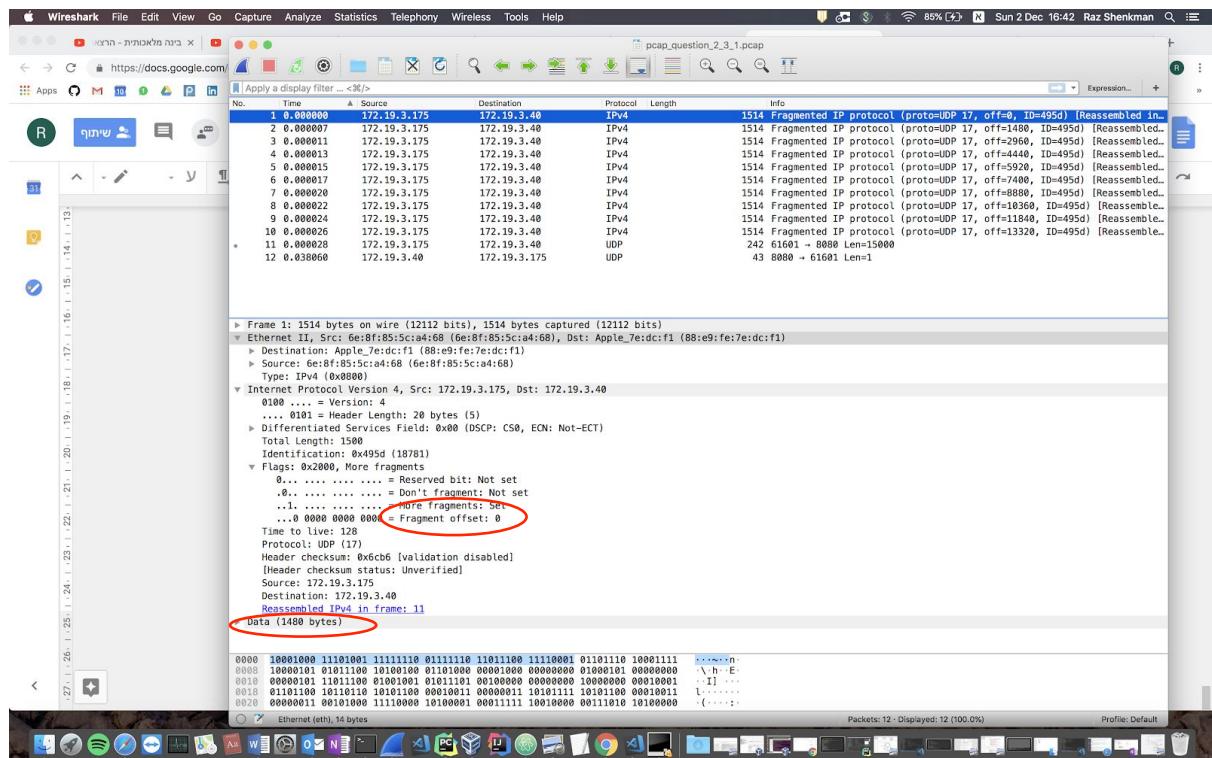
כאן אנו מתריצים בחבילה המורכבת (שהורכבה אחרי הפקטה الأخيرة).

נשים לב לתהילך: הקען מורד לשכבות הרשת בקשה לשולח 15000 בתים, כאן שכבת הרשת עשו

פרגמנטציה, כלומר מפרקת את החבילה לפני השולח, שהוא 1514 (הסקטרי לפ' מה שנשלח).

כיון שהקען לא פרוטוקול אחראי כמו tcp הוא לא מבצע חלוקה של החבילה (tcp מבצע segmentation segments וכך הוא מעביר דגל don't fragment ואין צורך לשכבות הרשת לטפל בחלוקת הפקטוות לחליים שונים).

החבילה הראשונה שנשלחה:



נשים לב שה[mess](#) (גודל המידע המקורי) יכול להשתלח לא כולל [headers](#) הוא 1480 (רואים זאת לפי השדה `data`, עבוריו מתווסף `header` של שכבת הרשות (20 בתים) ו-`header` של שכבת הקו (14 בתים) ושה"כ גודל של 1514.

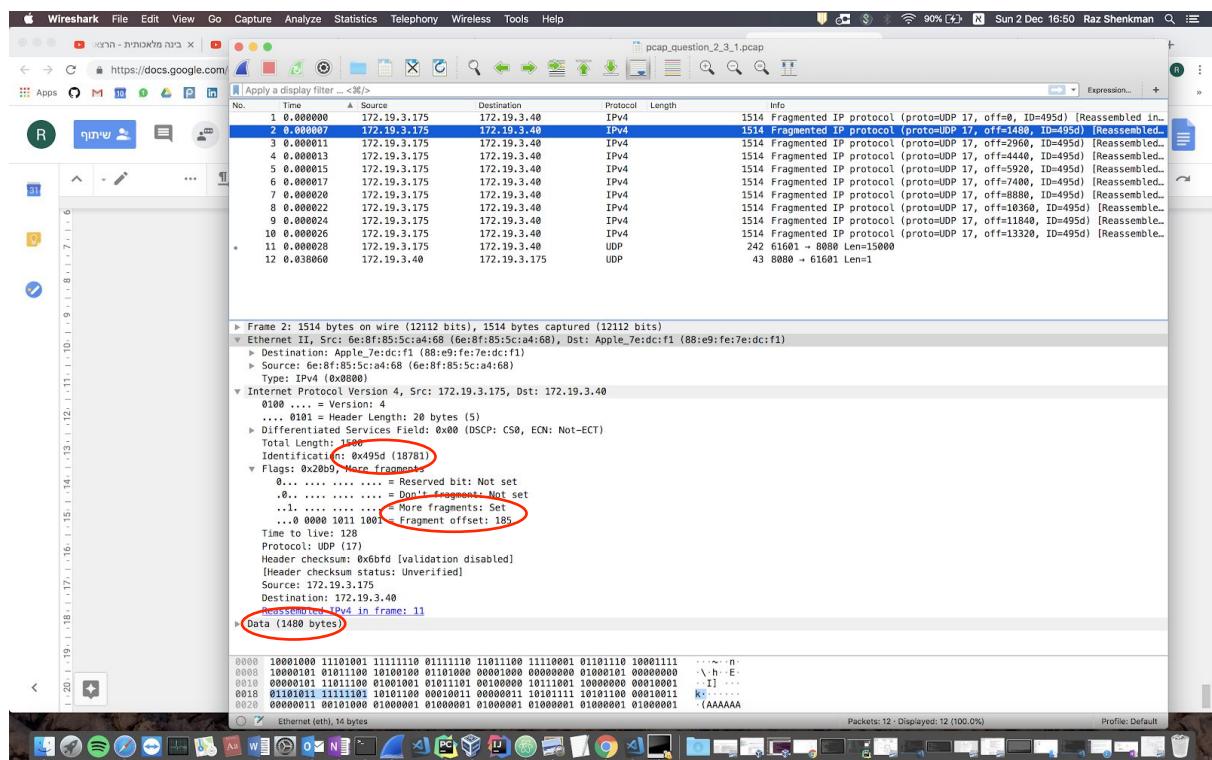
נתבונן בכל החבילות לא כולל החבילות האחרונות:

לכל חבילה הinfo כולל `ip protocol`, `fragmented ip`, כלומר שהיא חלק מחבילה שעברה פרגמנטציה. נשים לב שעבור החבילות הראשונות והחבילות האחרונות (לא כולל האחרונות) דלוק הדגל של `mf` (כלומר, בוצעה פרגמנטציה, ויש עוד חבילות אחרות החבילות הוזו). נשים לב שב`tcp` היה דלוק הדגל `don't fragment` בשכבת הרשות.

בכל החבילות חוץ מחבילה אחת לא יהיה `header` של `kdp`, כיוון שכבת הרשות מבצעת פרגמנטציה לא משנה לה איך המידע שהוא קיבל הגע, היא סה"כ מחלקת אותו לחבילות בגודלים המתאים ומוסיפה את `headers` הRELATIONALים שלה, וכן יהיה רק `kdp` אחד בכל החבילות שנשלחו.

נשים לב שה[identification](#) (שדה שחשוב בפרגמנטציה כדי לדעת שכל החבילות קשורות אחת לשניה) הוא זהה בכל החבילות והוא 0x495d. נשים לב שה[offset](#) של החבילות הראשונה הוא 0.

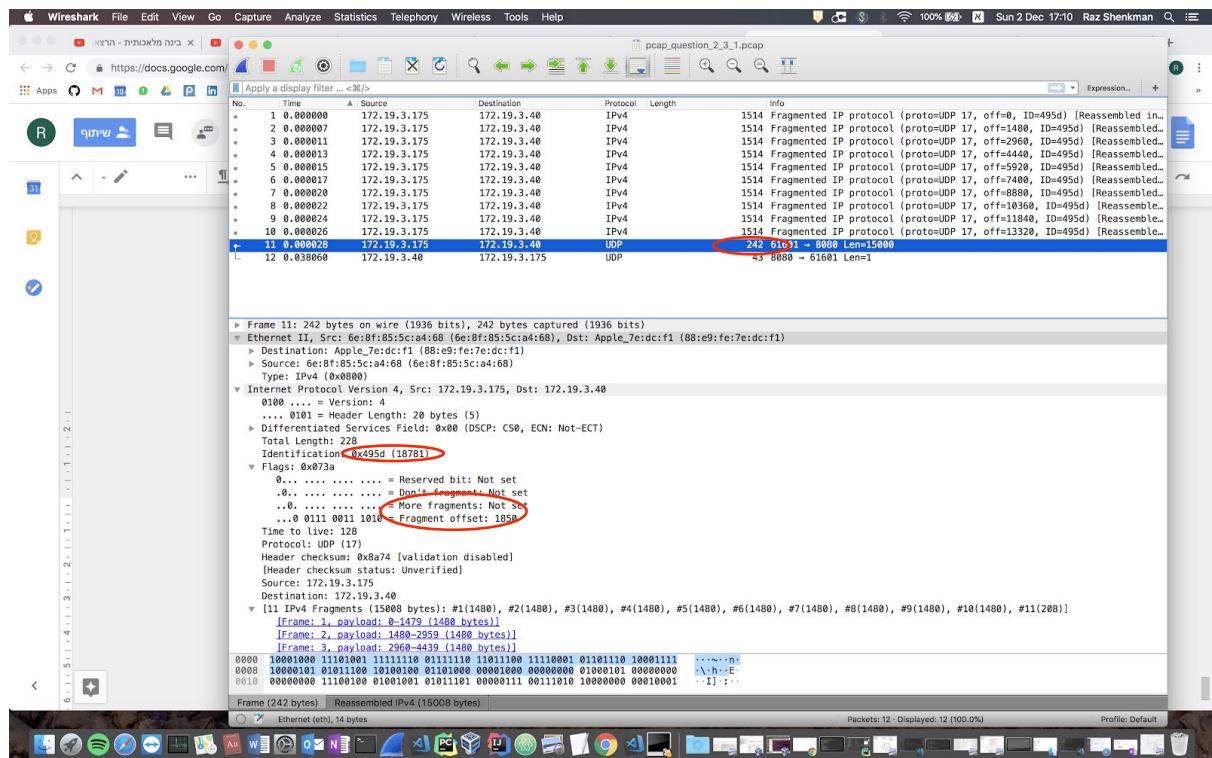
נתבון בחבילה השנייה שנשלחה:



כאן בחבילה השנייה נשים לב שהoffset הוא 185, המידע מוחלק ב8 ולק זה בעצם $185 * 8 = 1480$, שזה הגיוני שכן החבילה הראשונה שנשלחה כוללת 1480 bytes מהחבילה המקורית, והחבילה השנייה תתחיל עם המידע שנמצא אחרי אוטם בתים. נשים לב שההidentification הוא גם 0x495d. גם כאן דלוק דגל mf.

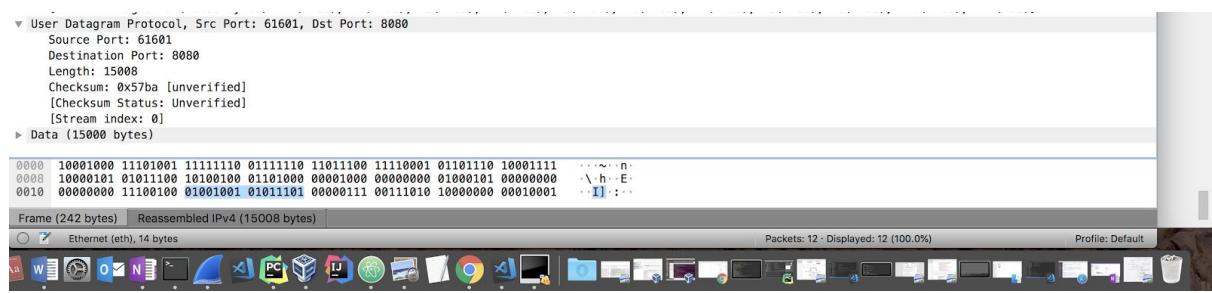
ושזה'כ נשלח גם כאן 1480 בתים.

נתובן בחבילה الأخيرة שנשלחה מהלוקות:



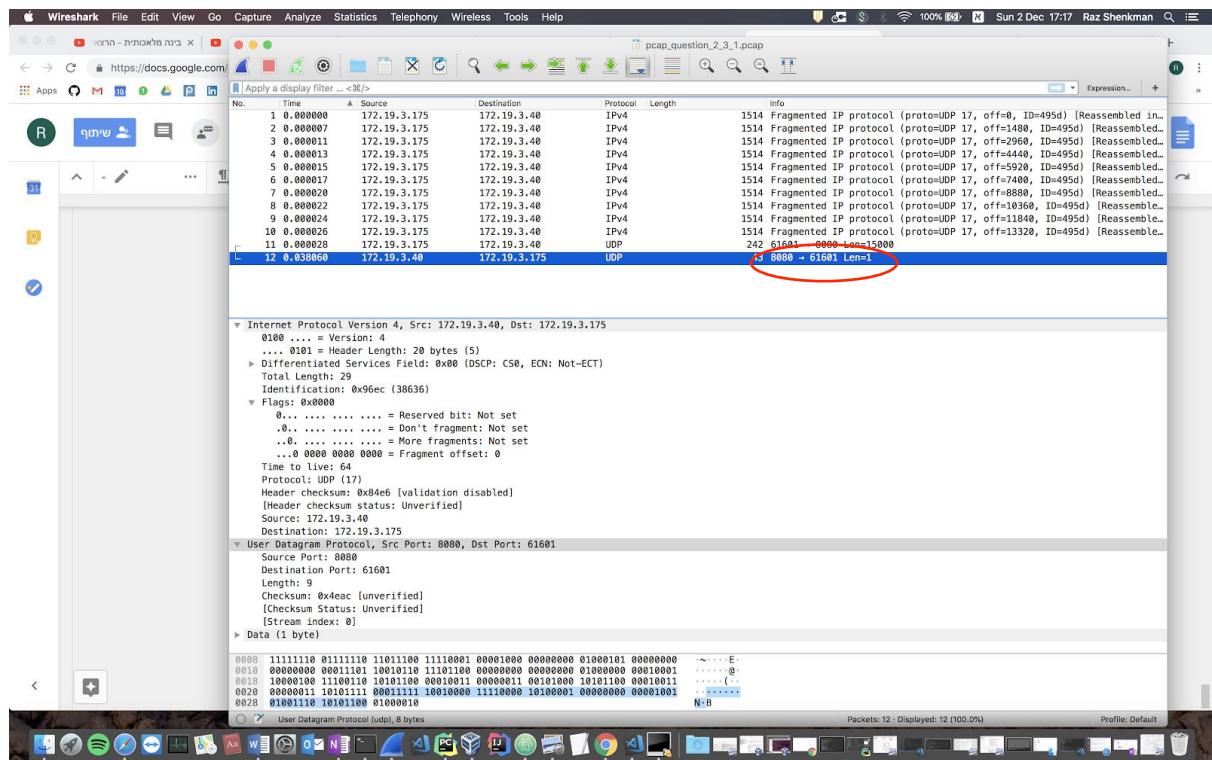
נשים לב שהגודל החבילה צריך להיות 15000 פוחות 1480 * כמספר החבילות השלומות שנשלחו = 200, וכיון שיש גם את header udp, ואת שאר headers שביחד הם 42 בתים נקבל שגודל החבילה הזו הוא 242 בתים.

נשים לב שהחומר id כאן הוא 0x495d (כי זו החבילה الأخيرة), נשים לב שבשכבה הקדם המידע הוא יחסית מצומצם:



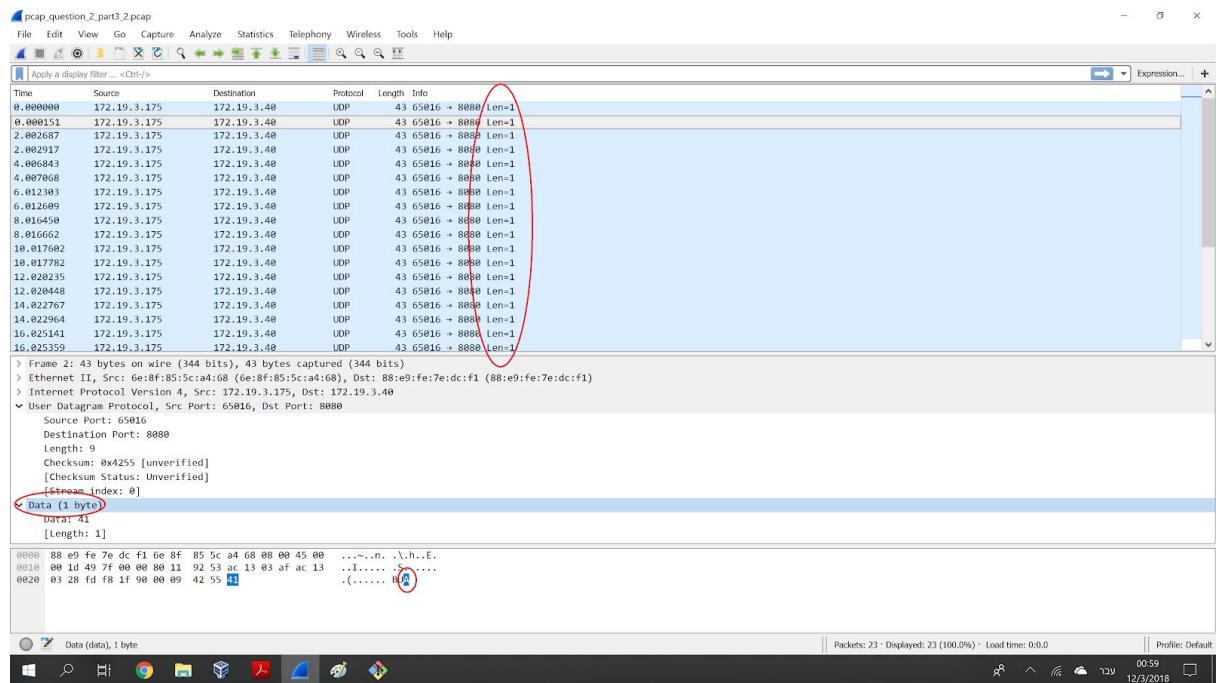
שכן לcdn לא משנה כמה בתים נשלחים, והיא פשוט מורידה לשכבה הרשות שתתפל בשילוחו שלhn.

נתבונן בחבילת שהורת שלח ללקוט:



נשים לב של חבילה זו אין דגלים בחייבת הרשות שקשורים ל프로그램נציה (כיוון שהוא עברה פרוגמנטציה) וכן נשלח רק ביט אחד של מידע, בתוספת headers משאר השכבות נגיעה ל-43 בתים של מידע סה"כ.

שאלה 2-2



נשים לב שבמקרה זה (שרות ולקוח UDP), הלוקו שולח כל פעם 2 תוו' 'A' רצופים) כל ההודעות יתתקבלו בונפרד (ניתן לראות בתמונה שכל ההודעות מוגדל 1 ולא 2, ושהתוכן הוא **A**). נזכיר ש בTCP ראיינו שלפעמים שת' הודעות 'A' רצופות יתמצאו להודעת 'AA' אחת. ננסה להסביר את התופעה.

ידוע לנו ש TCP מעביר את המידע כسطף של בתים (byte stream). ז"א, הוא לא מודיע לגבולות של הודעה או לא הודעה, הוא רק יודיע שיש לו רצף של בתים שעליו להעביר הלאה בסדר זה. כאשר בTCP עושים שינוי send רצופים, אז לברור הבטים שהוא צריך להעביר נכנסוividually ביחיד שני ה AIDS האלו TCP כמו שהוא יודיע העביר אותו כ byte stream. לעומת TCP לא הספיק לשולח את ה A הראשון לפני שהשני כבר נכנסו, TCP אמרו שולח את כל הבטים שיש לו ב buffer前に מודיעות ל "מי הגיע מאיזו הודעה" וכן לפעם (כשהר TCP לא הספיק לשולח את ה A הראשון לפני שהשני נכנס) הם יתמצאו להודעה אחת (הרי ב buffer ישם המידיע AA).

ב UDP לעומת זאת, אין את אותה ארכיטקטורה של שף בתים שמועברים מוקצה לבקשתם. כתובנו:

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

כלומר את הודעה הוא יהיה שולח מיד ליעד, ולא בהתבסס על החיבור ביניהם. לכן כל שליחת הודעה הייתה בלתי תליה נקודת בשליחות ההודעות האחרות, וכך כל הודעה A נשלחה בפני עצמה והתקבלת אצל הרשות בונפרד.