מבוא לרשתות תקשורת – תרגיל 2

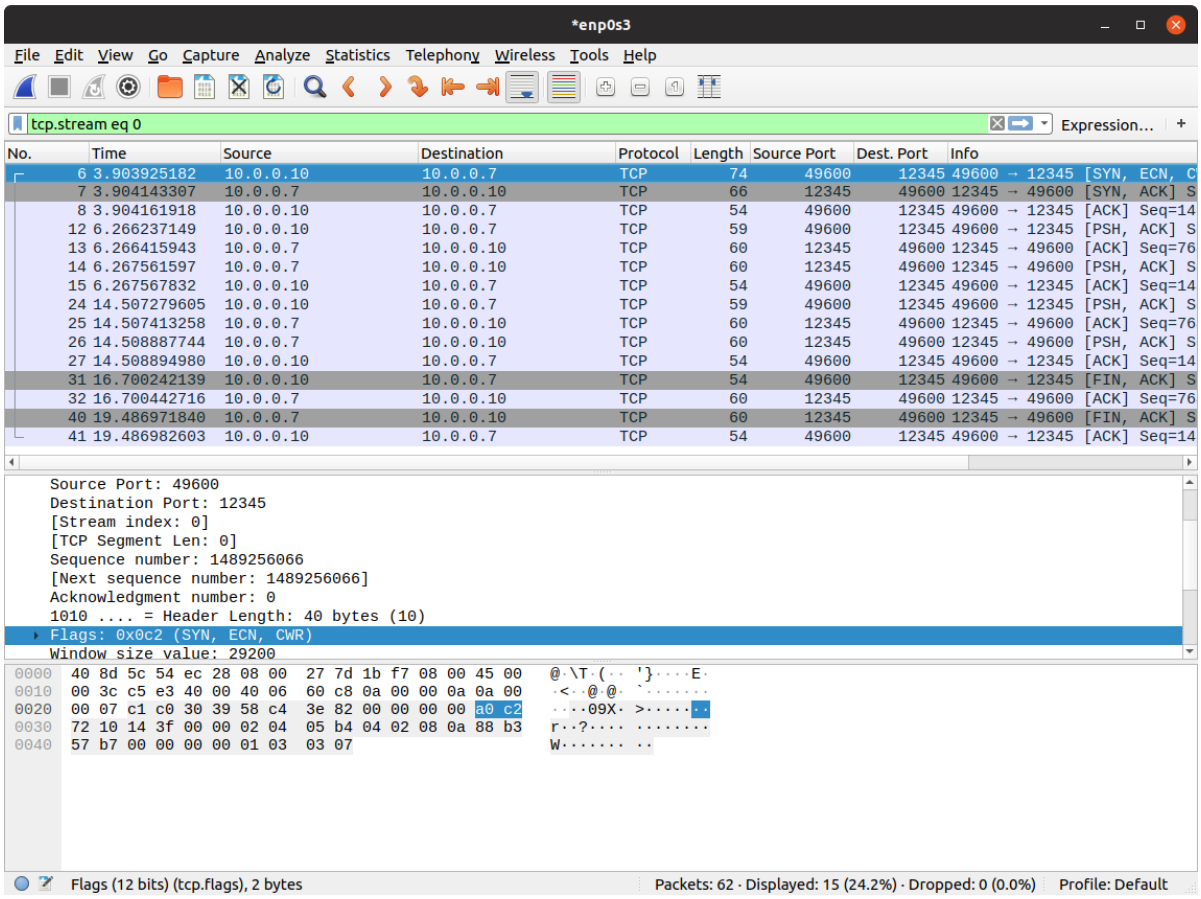
סעיף א

תהליך הhandshake מתחיל עם העברת פקטת SYN – Synchronize מהקליינט המעוניין בחיבור, העברת Syn-Ack מהשרת על הפקטה ולבסוף העברת Ack מהקליינט שמכריע שהחיבור בוצע בהצלחה.

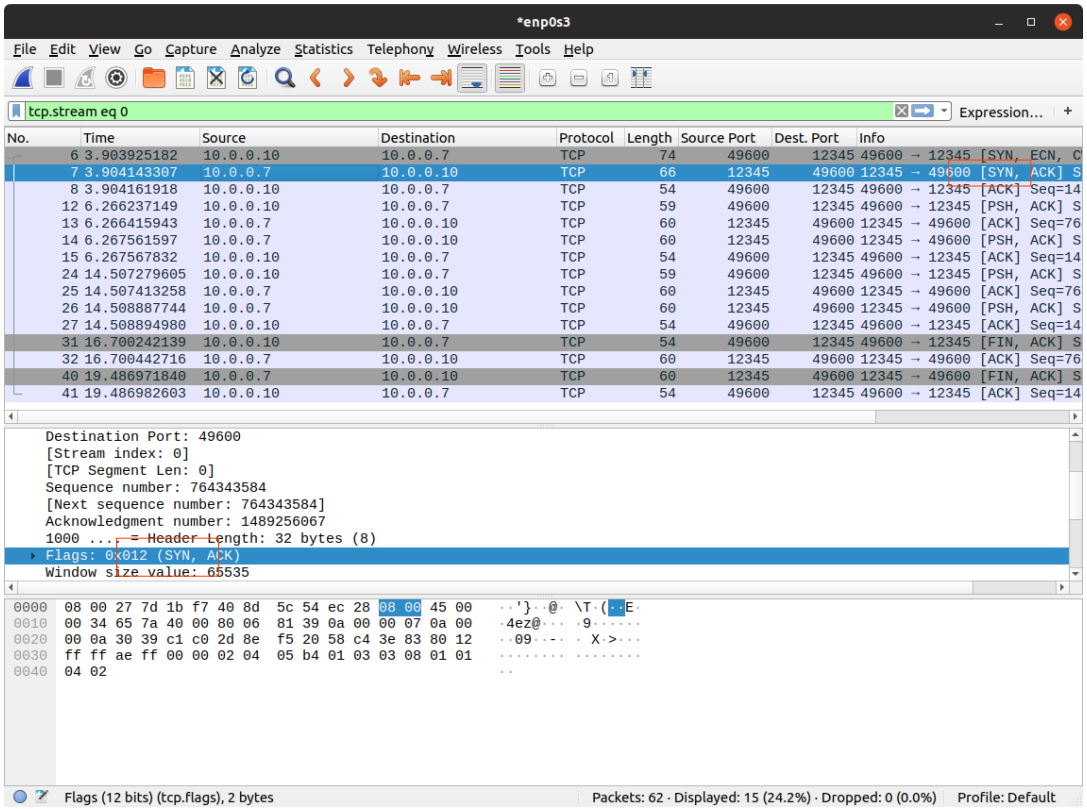
נראה זאת בהסנפת התעבורה בWireshark:

נתחיל עם הלקוח הראשון – מאזין לפורט 49600

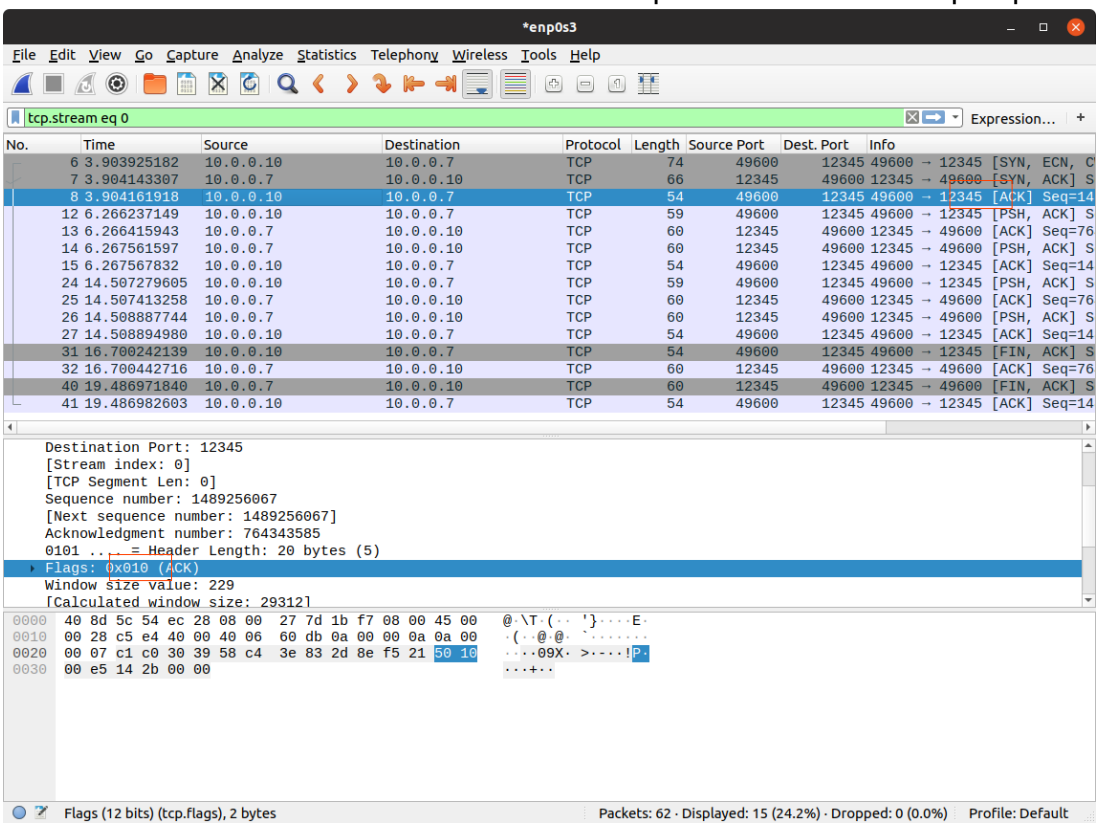
שליחת הבקשה לסנכרון – Syn (כאשר 10.0.0.10 הוא אייפי הלקוח, 10.0.0.7 אייפי השרת)



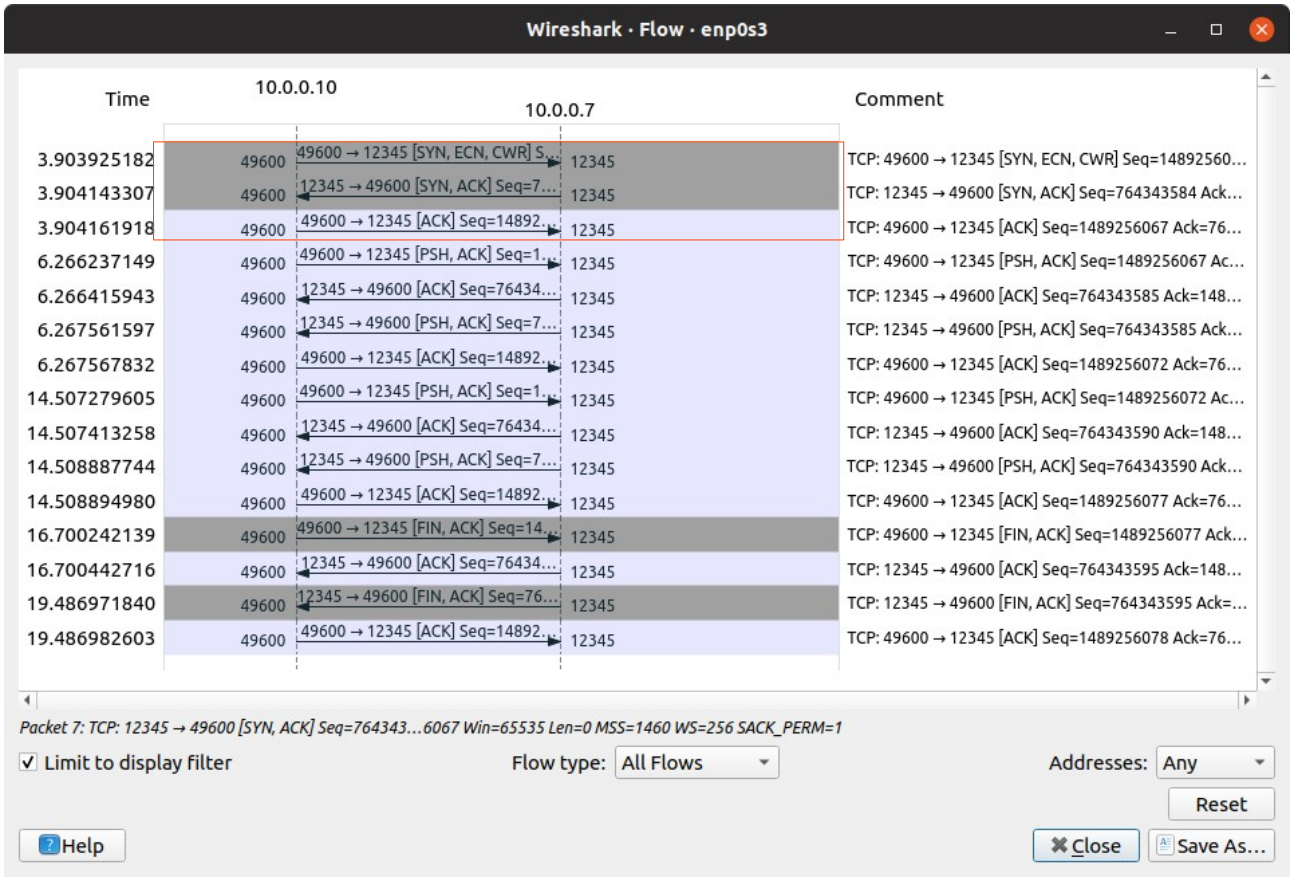
לאחר מכן ניתן לראות שהשרת שולח בחזרה Ack על הSyn



ולבסוף הקליינט שולח Ack ותהליך לחיצת הידיים מסתיים



אפשר גם כן לראות את תהליך לחיצת הידיים מתבצע בתוך הFlow Graph של התקשורת:



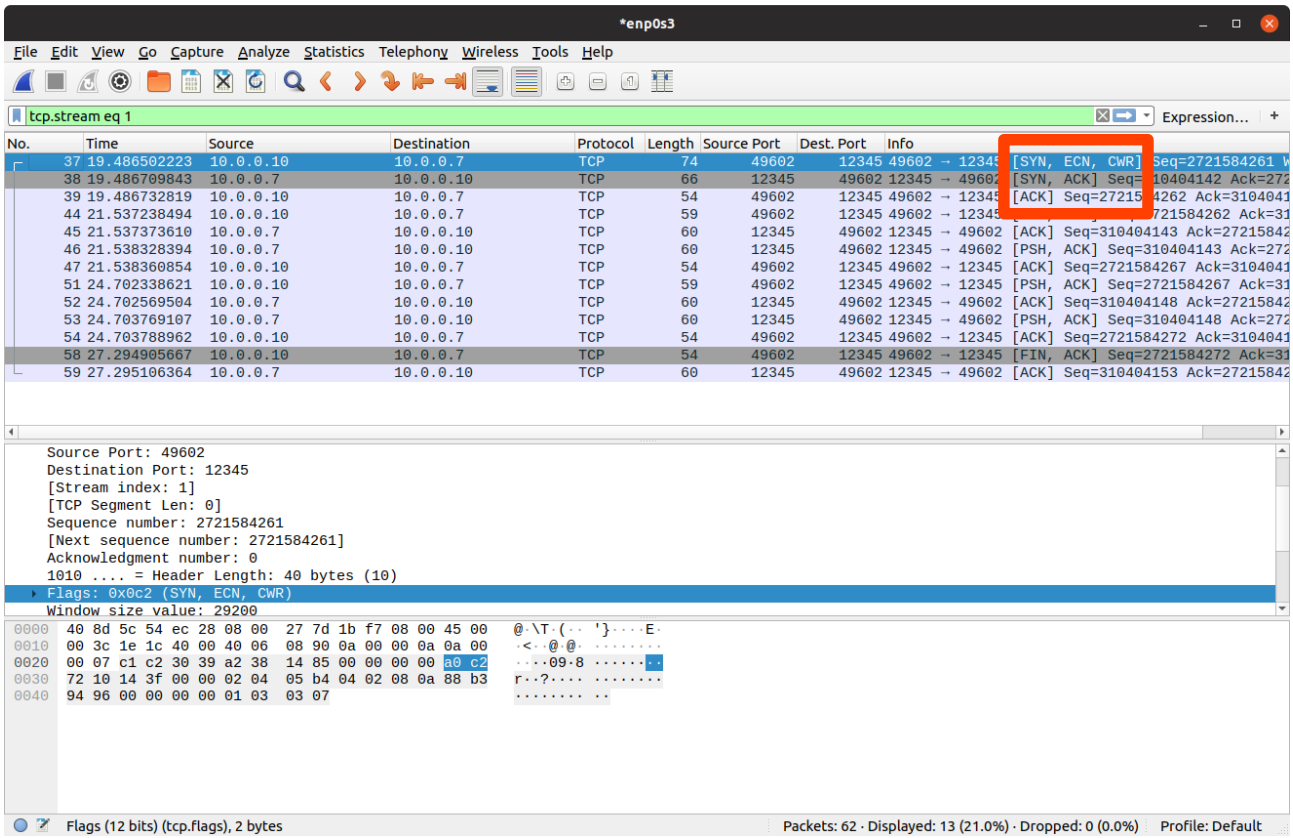
כעת נעבור ללקוח השני – מאזין לפורט 49602

ניתן לראות את שלושת שלבי לחיצת הידיים, רק שכעת הפורט של הקליינט השתנה (הפורט של ה שרת כמובן נשאר קבוע)

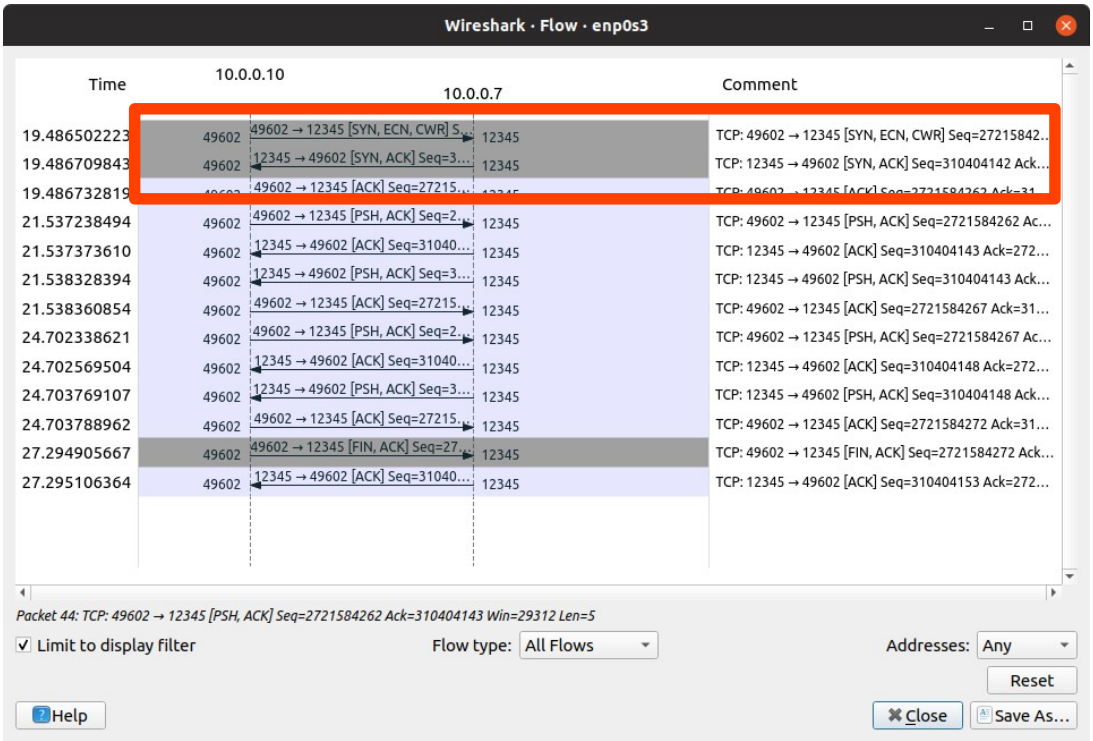
Client -> Syn -> Server

Client <- Syn,Ack <- Server

Client -> Ack -> Server

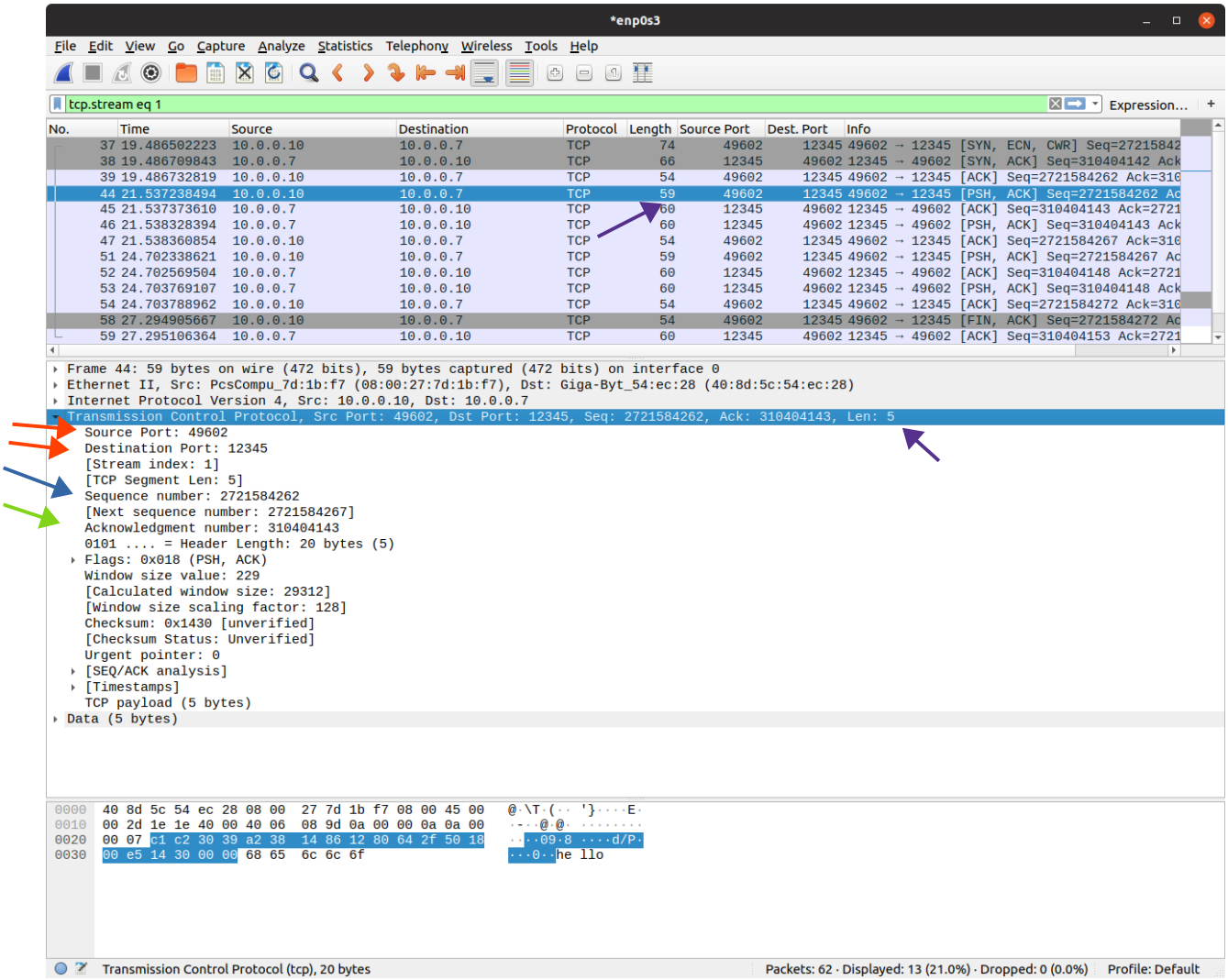


וב Flow Graph:



סעיף ב

כעת נעבור על החיבלות ששלחו את ההדעות עצמן (Worldו Hello) מצד הלקוחות:
החבילה שנשלחה לאחר חבילתה Ack על Syn,Ack היא החבילה שהעבירה את hello:



נעבור על כל רכיב בהדר של החבילה:

Source Port – כמובן שפורט המקור שלנו הוא הפורט שהקליינט מאזין לו – 49602

Destination Port – ופורט היעד הוא הפורט שהשרת מאזין לו – 12345

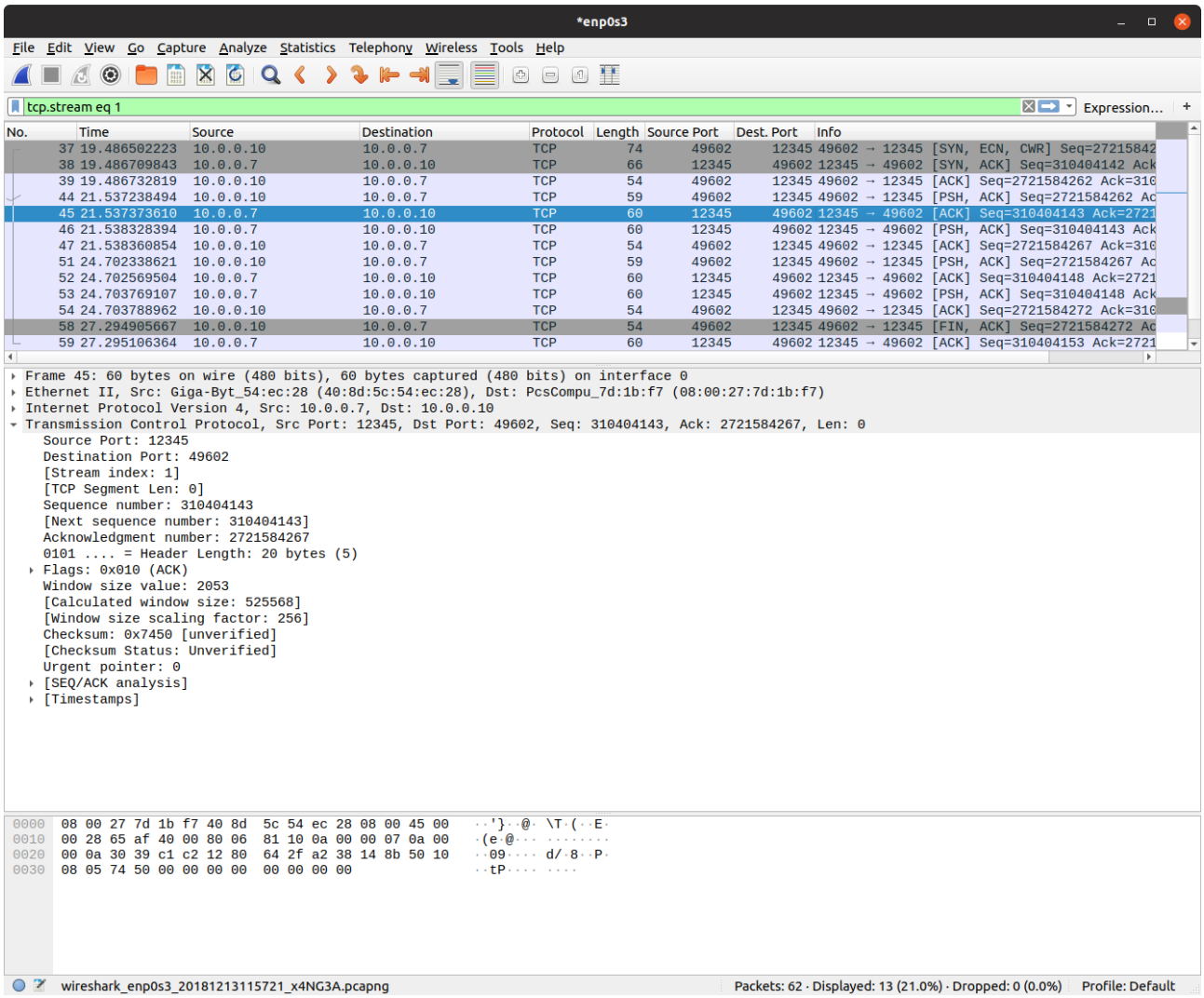
Sequence Number – הSequence Number שנשלח גדול ב1 מהמספר (שנבחר באופן רנדומלי) כאשר נשלחה חבילת הSyn, כעת הוא 2721584262, והוא היה 2721584261. הוא עלה ב1 כיוון שהשרת קיבל בAck והעלה את הAck Number ב1. (פקטות SYN מעלות ב1).

Ack Number – מתחיל ב0, ואז מתחיל Offset ממספר רנדומלי גם כן, במקרה שלנו 310404143.

Length – גודל החבילה, במקרה שלנו 5 בתים (hello) + ההדר

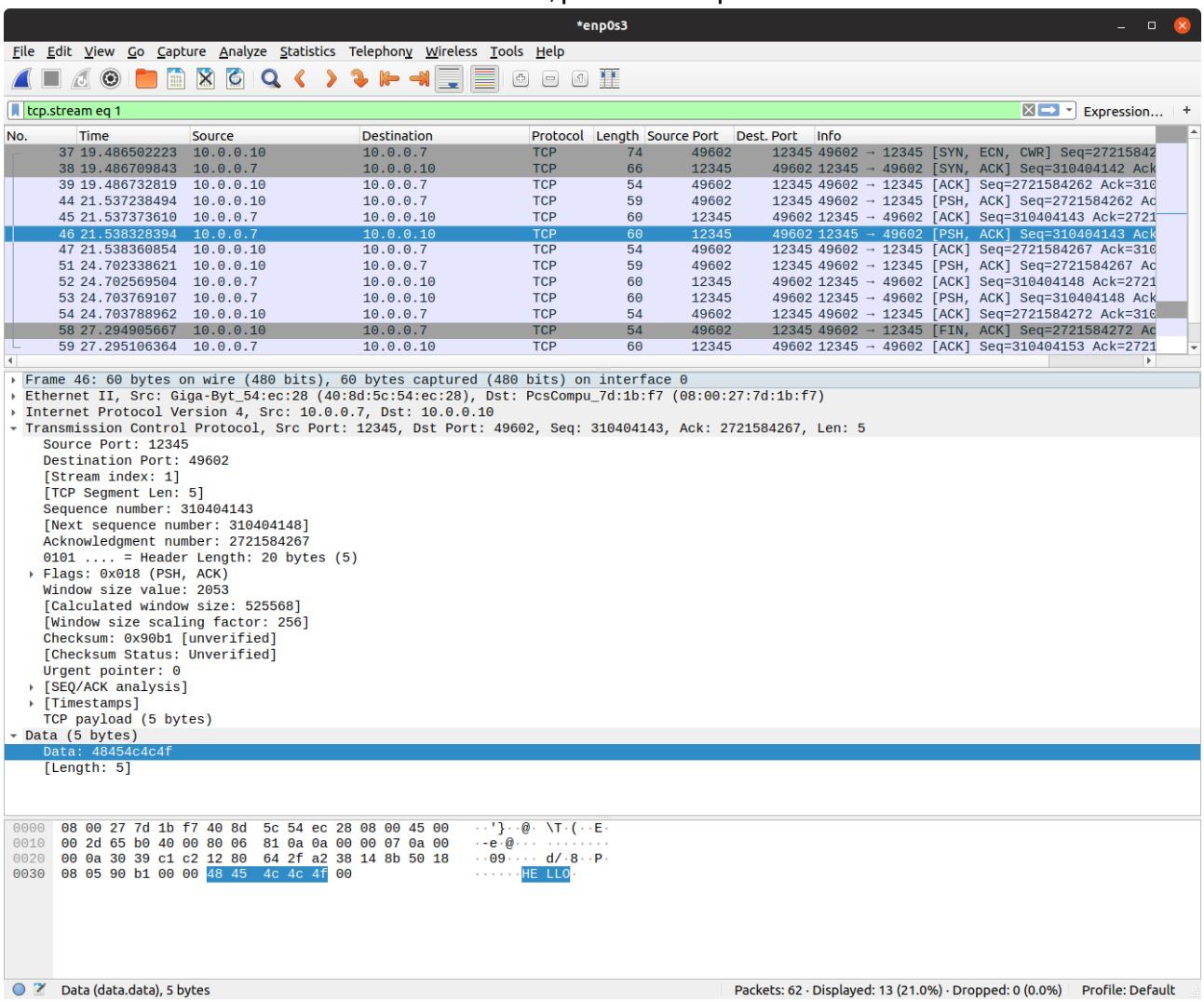
מוטב לציין כי בכל חבילה אחרי הSyn, פרוטוקול הTCP שולח את דגל הAck דלוק כיוון שכל חבילה שולחת את מספר הAck העדכני ביותר.

כעת נוכל לראות את הAck של השרת להודעת הhello



הפורטים כמובן הפוכים, היעד הוא 49602 (הלקוח) והמקור הוא 12345 (השרת)
הוא עושה Ack על הSeq Number שהתקבל מהלקוח + 5 הבתים של גודל החבילה ושולח את ה Seq Num שעליו הלקוח עשה Ack (כיוון שהוא לא שלח הודעה לפני כן)
גודל החבילה הוא 0 ללא ההדר כיוון שאין בעצם payload, החבילה היא רק Ack על חבילת הhello.

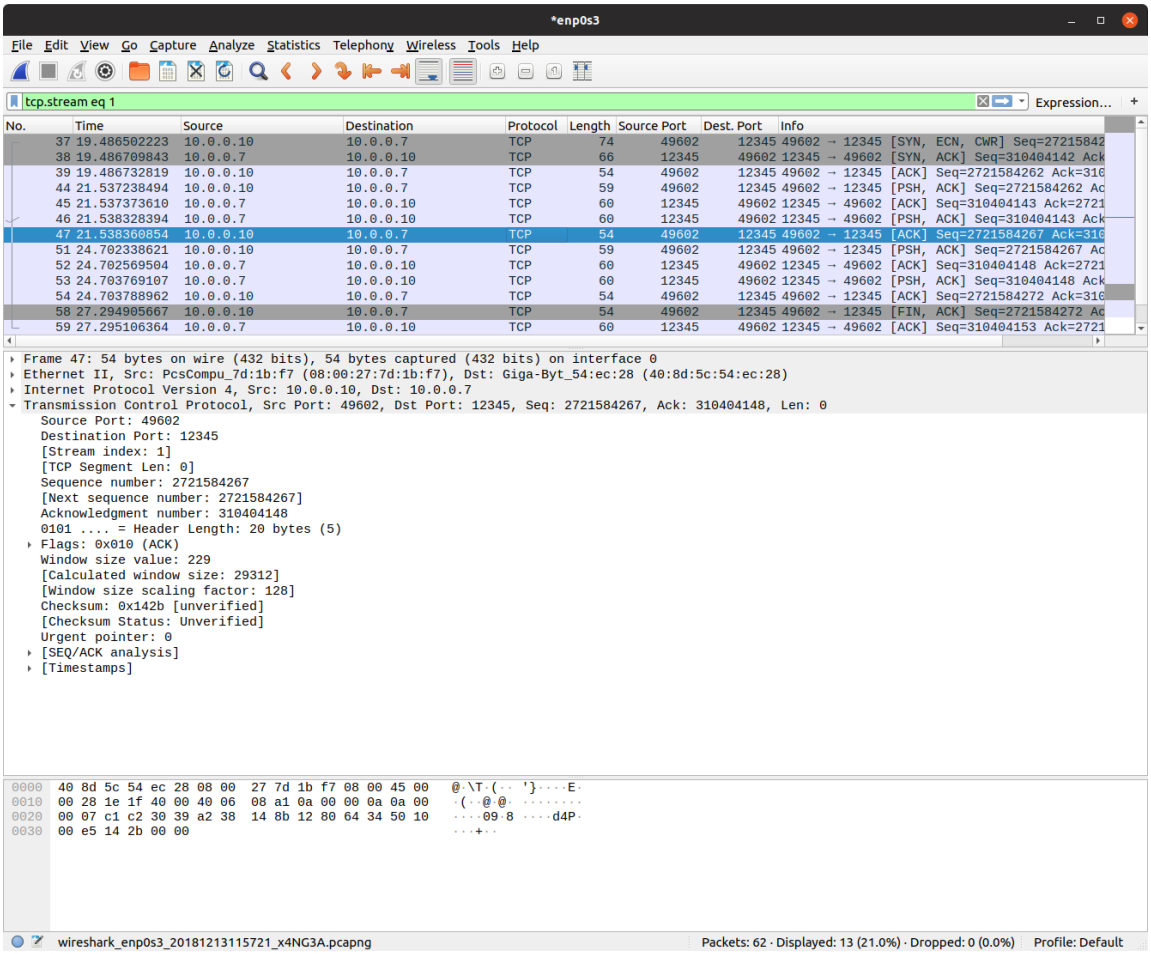
נסתכל על החבילה שהשרת שולח ללקוח לאחר מכן, החבילה עם HELLO



הפורטים זהים לחבילה הקודמת, מהשרת ללקוח, כיוון שלא התקבלה שום חבילה מהלקוח ה Sequence Number והAck Number נשארים זהים.

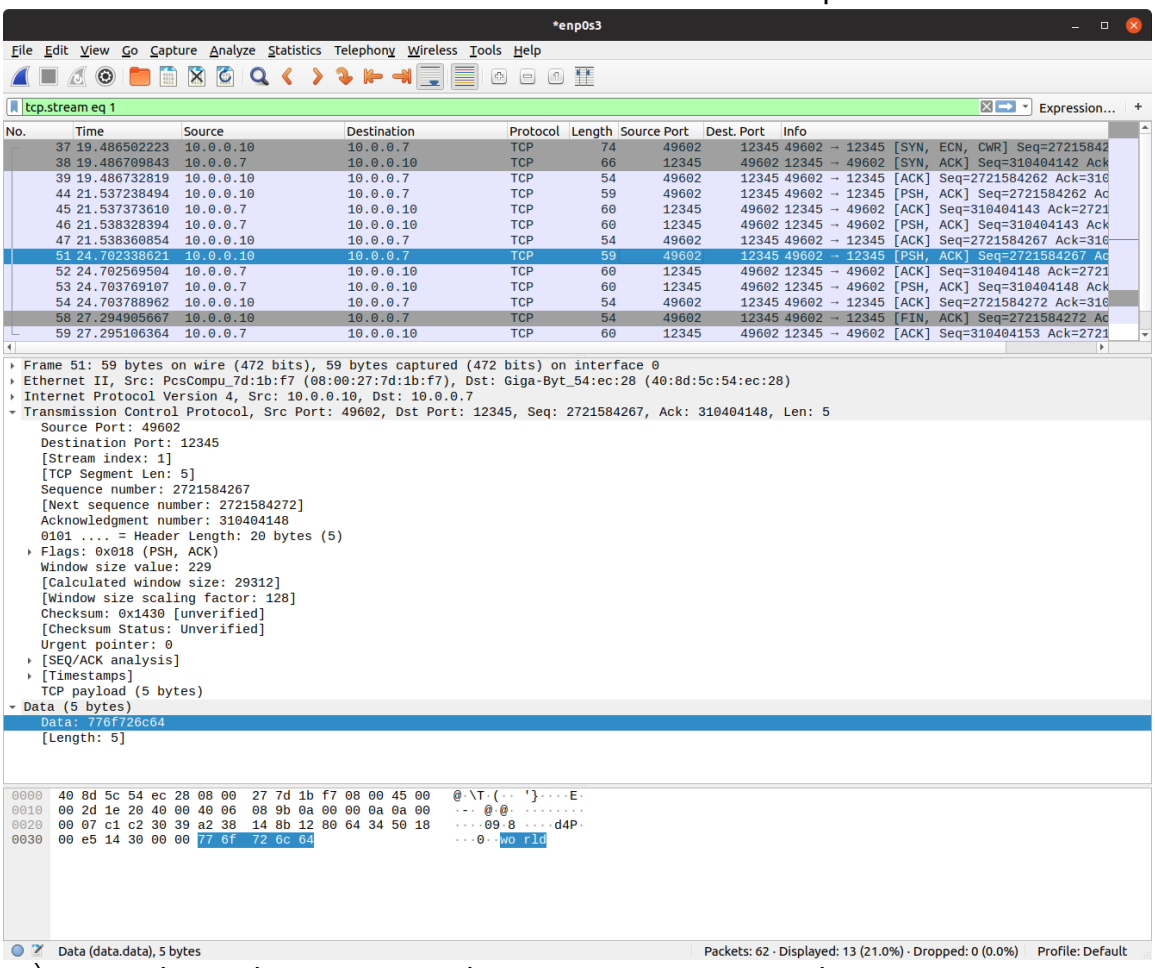
גודל החבילה יהיה 5 בתים בעקבות הודעת הHELLO לא כולל ההדר

הלקוח כעת שולח Ack על החבילה שהתקבלה.



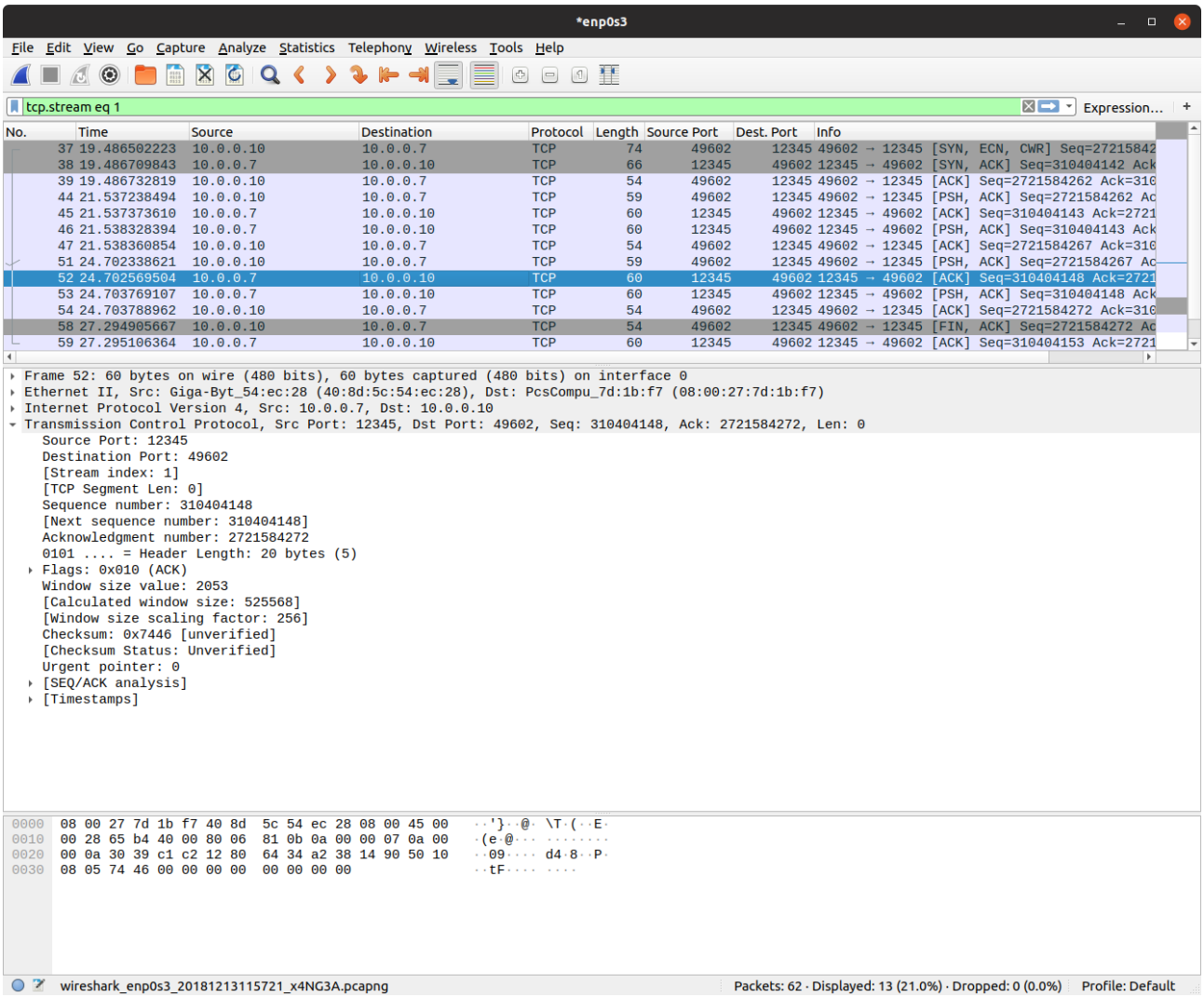
הלקוח מאשר את הSEQ שנשלח מהשרת + 5 (גודל החבילה)
שולח את הSeq שלו + 5 מההודעה הקודמת שהוא שלח (הhello הרגיל)

נעבור להודעת הworld של הקליינט:



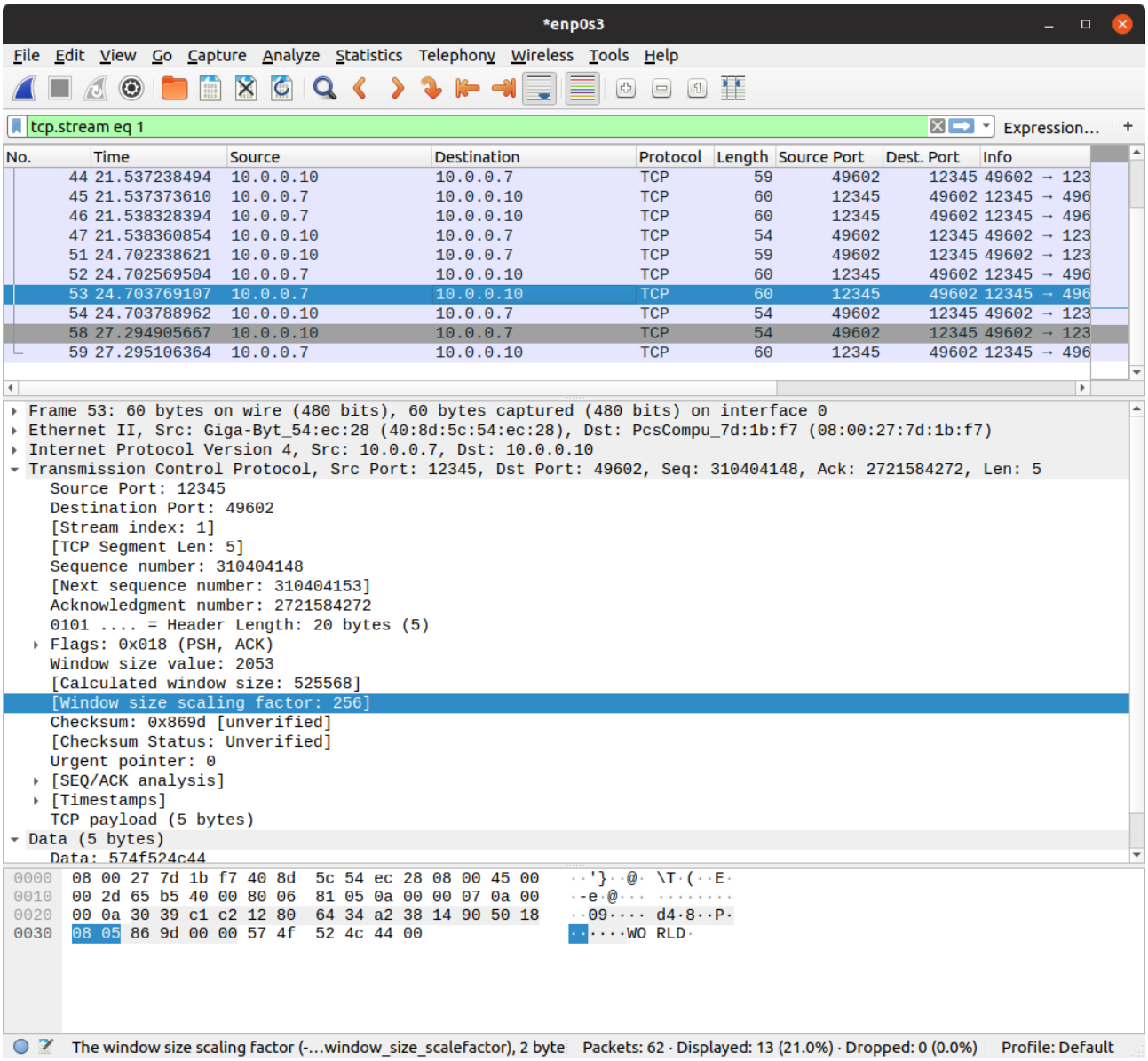
הAck והSeq נשארים זהים כי לא הייתה תעבורה שהתקבלה מהשרת. גודל החבילה הוא 5 (world) ללא ההדר הפורטים הם מהלקוח (49602) לשרת (12345)

והAck מהשרת על הודעת הworld:



הוא שולח Ack Num של הSeq שהלקוח שלח + 5 (גודל ההודעה), ושולח Seq של אותו המספר שהלקוח עשה לו Ack פעם שעברה כי הוא לא שלח שום הודעה מאז שהלקוח קיבל את ההודעה הקודמת

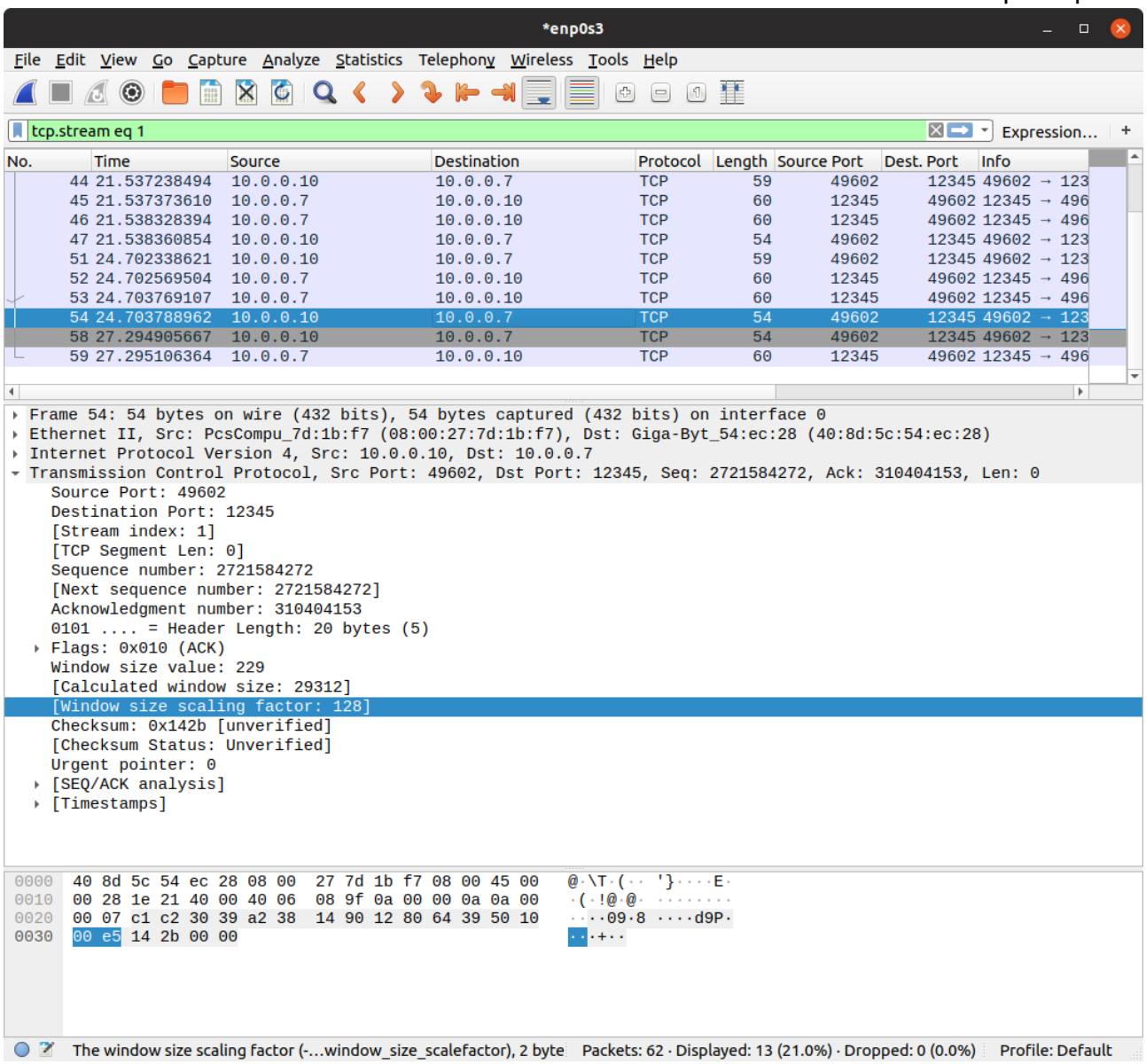
נעבור לשליחת ההודעה WORLD מצד השרת



השרת שולח אותו SEQ NUMBER כיוון ששליחת הACK לא העלתה אותו, וכן אותו ACK NUMBER כיוון שהלקוח לא שלח משהו חדש מאז.

הפורטים הם מצד השרת 12345 ללקוח 49602, גודל החבילה הוא 5 (WORLD)

ולבסוף הלקוח שולח ACK על הודעת הWORLD

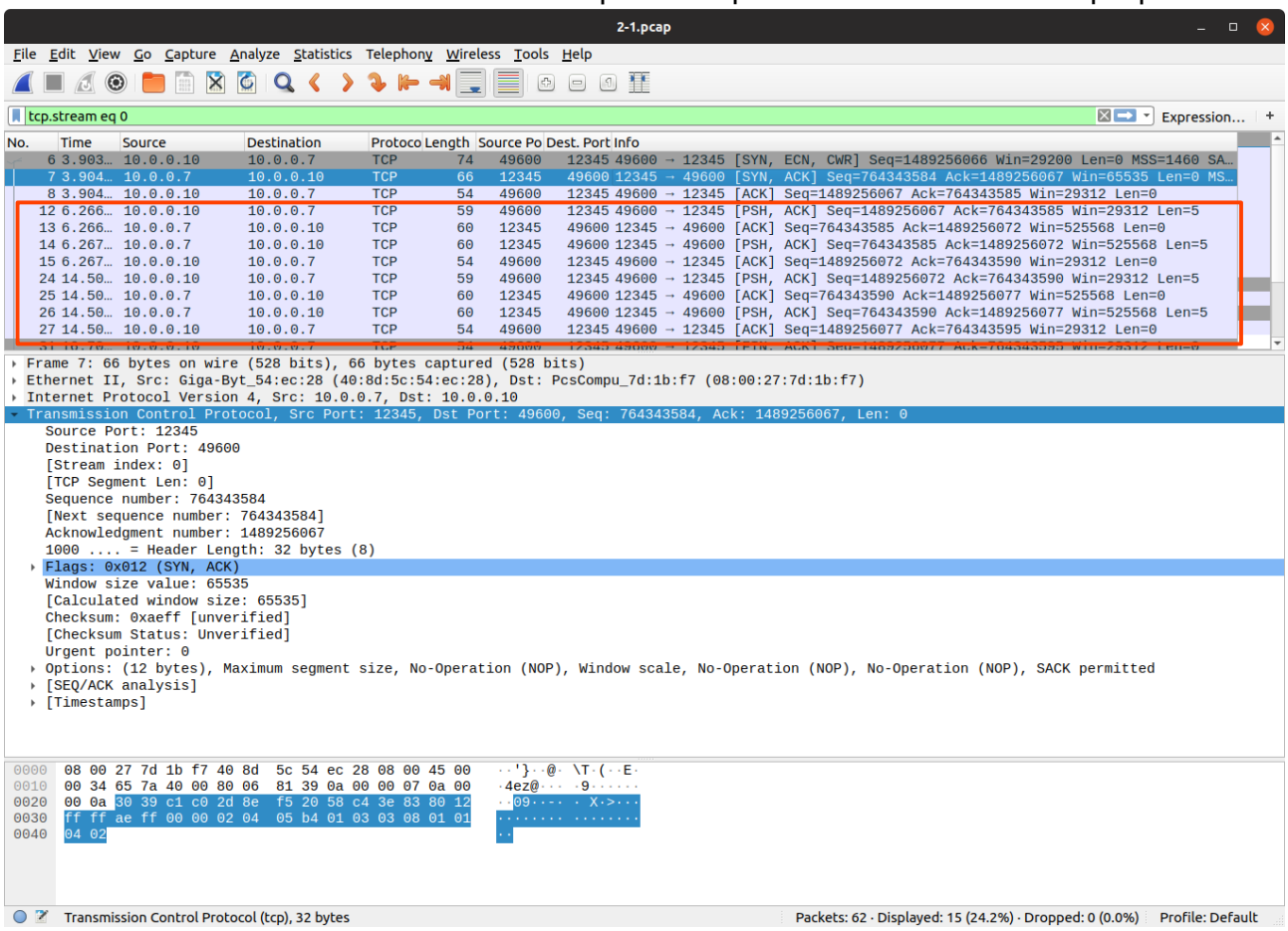


הלקוח שולחACK NUMBER עם הSEQ NUMBER שהוא קיבל מהשרת + 5 בתים של גודל החבילה

וכן הSEQ NUMBER שהוא שולח גדול ב5 מאז החבילה האחרונה שהוא שלח (הworld הקטן שגודלו גם היה 5)

הפורטים הם מהלקוח 49602 לשרת 12345 וגודל החבילה כאומר הוא 5 ללא ההדר.

כעת נתבונן בקצרה על התעבורה של הלקוח הראשון מול השרת:



הלקוח שולח את החבילה הראשונה אחרי הקמת החיבור, נבחר מספר סידורי 1489256067, שהתחיל ב1 פחות אבל עלה בעקבות הSYN לפני כן.

השרת עשה ACK לאותו הSYN בהתחלה ובחר מספר סידורי גם כן (764343585) שהקליינט עשה לה ACK.

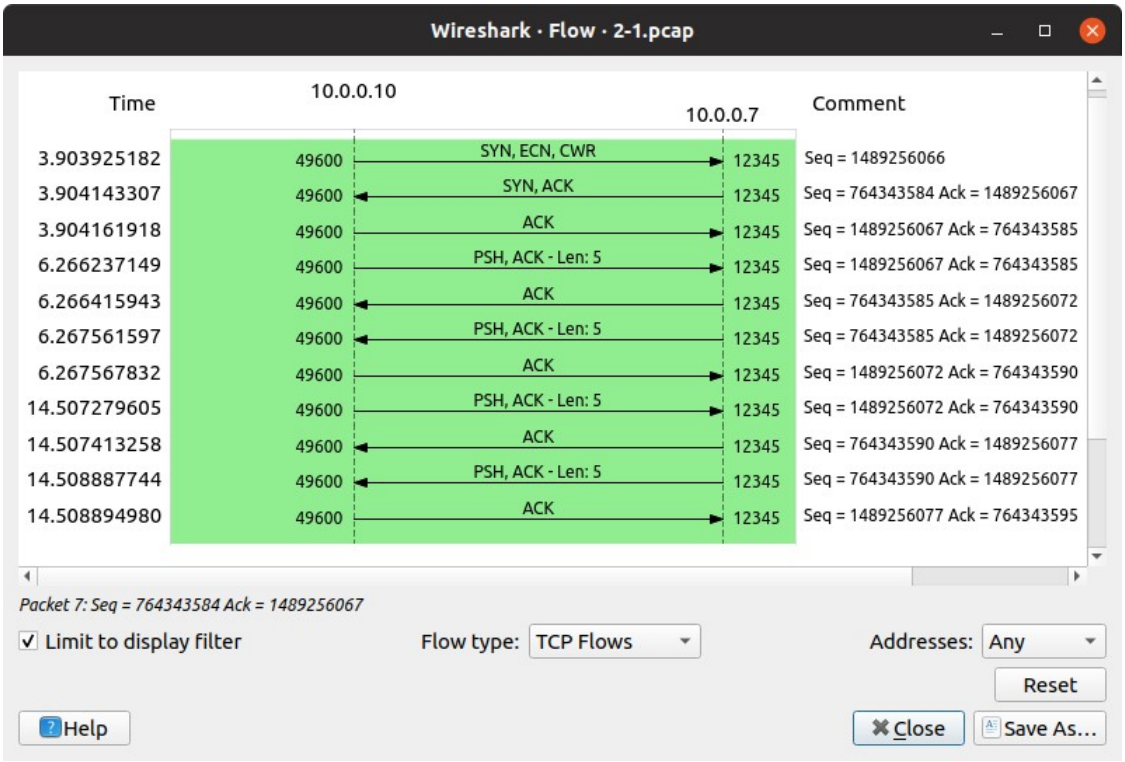
לאחר שליחת הודעת הhello (בגודל כולל הדר של 59) השרת עושה ACK לחבילה, כלומר ACK לSEQ שהקליינט שלח +5 בייטים כגודל התוכן.

כעת השרת שולח (עם אותו SEQ של הPACKET הקודמת ששלח ואותו ACK) את ההודעה שלו (HELLO)

הקליינט מאשר את ההודעה, עושה ACK לSEQ שהשרת שלח + 5, שולח את הSEQ שלו + 5 מההודעה הקודמת ששלח (hello)

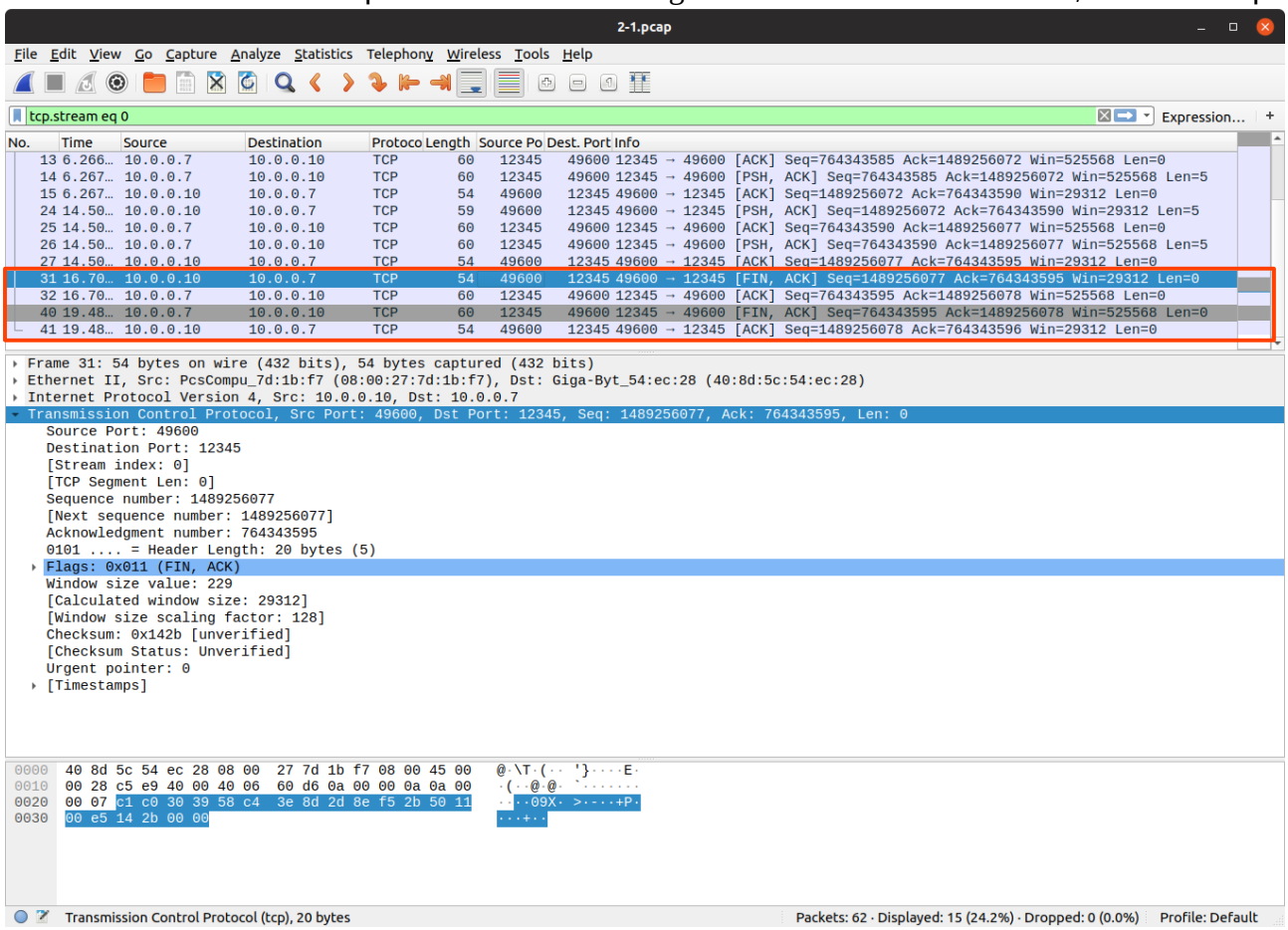
השרת מקבל עם ACK, והתהליך חוזר חלילה באותם Offsets לגבי world.

ניתן לראות את התהליך מתבצע טוב ע"י הFlow Chart

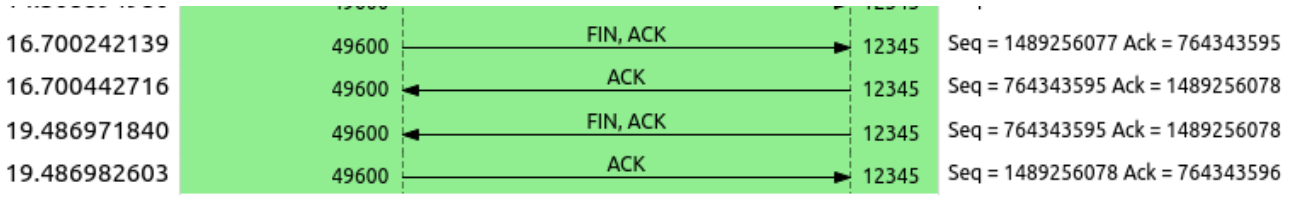


סעיף ג

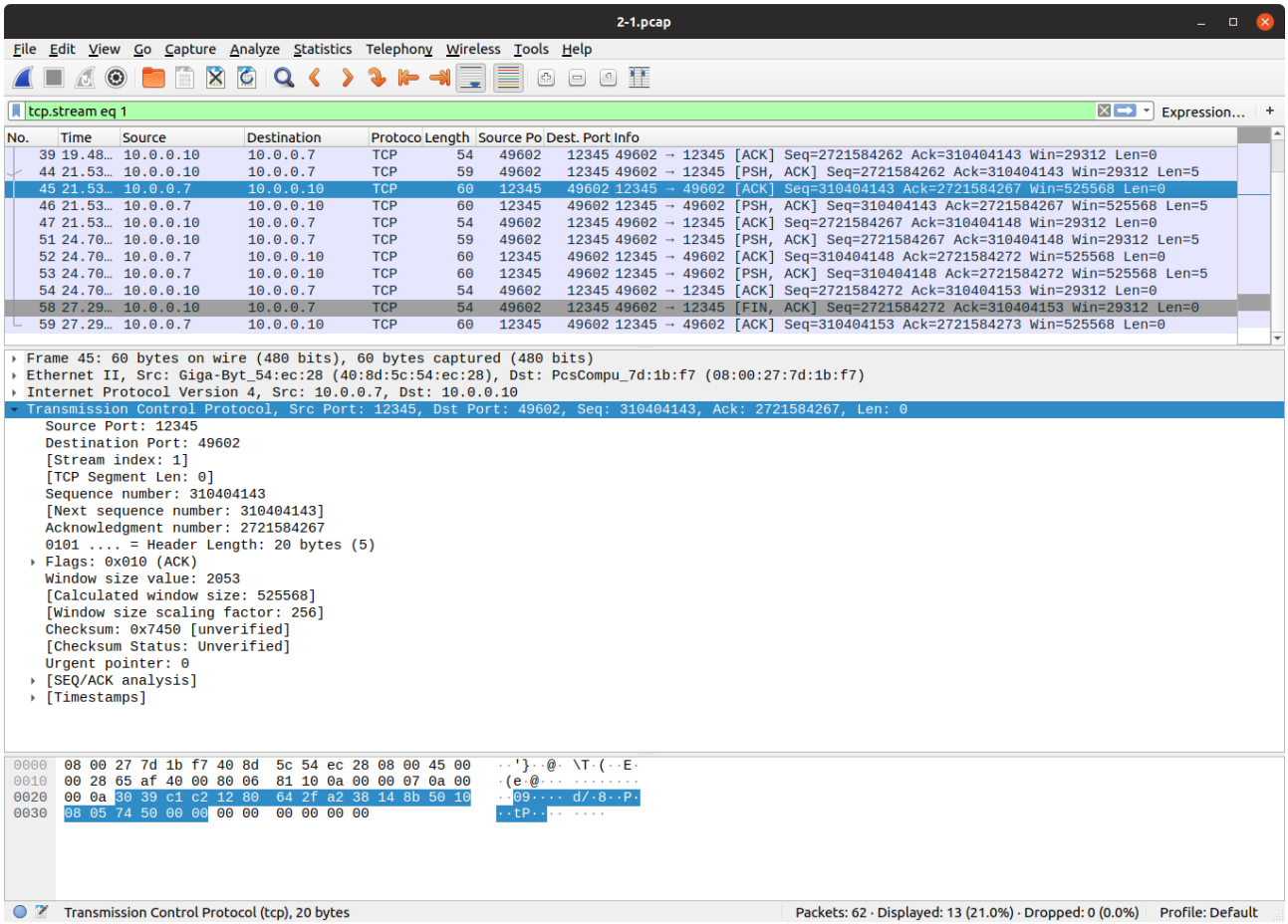
כעת נסביר את תהליך הTeardown ע"י החבילות שלנו. תהליך הteardown הוא תהליך ניתוק החיבור בין שני הצדדים, והוא מתבצע בעזרת שימוש הflag fin שמעיד על רצון לסגור את החיבור.



ברגע שהלקוח רושם quit, במקום לשלוח את ההודעה התכונה של הקליינט שולחת בקשה לסגור את החיבור. הקליינט שולח חבילה עם הדגל FIN שמעיד על רצון לסגור את החיבור.
לאחר מכן השרת מחזיר לו ACK על כך שהודעת הFIN התקבלה. אז השרת גם הוא יוזם את הליך הניתוק, ושולח הודעת FIN בעצמו אל הלקוח. הלקוח מקבל את ההודעה, שולח ACK וברגע שזה מתקבל השרת סוגר את החיבור וכך גם הלקוח.



נראה את תהליך הteardown בלקוח השני גם כן:



כאן לעומת זאת ניתן לראות שאומנם הקליינט שלח FIN, והשרת קיבל ושלח Ack על הFin הזה, התעבורה הפסיקה ישר. (כלומר השרת לא שלח בעצמו Fin)

לפי בידקה באינטרנט המצב הזה נקרא Half Closed Connection, שבעצם מתאר מצב בו אומנם הלקוח מוכן לסגירת החיבור, אבל לשרת עדיין יש מידע לשלוח ללקוח ולכן הוא אינו שולח FIN. עם זאת, הלקוח עדיין סוגר את החיבור (בעקבות הquit)