<u>ש.ב. 1</u>

איתמר גפן ורועי כהן

<u>חלק 1</u>

1 + תחילת 2) סיננו את התעבורה הרצויה בין הרשת והלקוח ויצאנו את התעבורה בקובץ תחת השם "trace1.pcapng" שמצורף במודל, לאחר פתיחת הקובץ דרך

0.	Time	Sour	ce	Destination	Protocol	Length Info		
2	1 8.000000	100 127	.0.0.1	127.0.0.1	UDP	103 37371	- 12345	Len=59
	2 8.090861	325 127	.0.0.1	127.0.0.1	UDP	103 12345	→ 37371	Len=59
Lin Int	ux cooked ca ernet Protoc r Datagram P	pture ol Versio	n 4, Src:	ts), 103 bytes captu 127.0.0.1, Dst: 127. 37371, Dst Port: 123	0.0.1	on interface a	any, id 6)
Lin Int	ux cooked ca ernet Protoc	pture ol Versio	n 4, Src:	127.0.0.1, Dst: 127.	0.0.1	on interface a	any, id 6)
Lint Use Dat	ux cooked ca ernet Protoc r Datagram P a (59 bytes)	oture ol Versio rotocol,	n 4, Src: Src Port:	127.0.0.1, Dst: 127. 37371, Dst Port: 123	0.0.1 345		any, id 6)
Lint Use Dat	ux cooked ca ernet Protoc Datagram P a (59 bytes)	oture ol Versio rotocol,	n 4, Src: Src Port:	127.0.0.1, Dst: 127.	0.0.1	***	umy, id 6	
Lint Use Dat	ux cooked ca ernet Protoc r Datagram P a (59 bytes)	oture ol Versio rotocol,	n 4, Src: Src Port:	127.0.0.1, Dst: 127. 37371, Dst Port: 123	0.0.1 445	*** **********************************	any, id 6)
Been Been Been Been Been Been Been Been	ux cooked ca ernet Protoc Datagram P a (59 bytes) 80 00 03 04 45 00 00 57 7f 00 00 05 7f 00 00 57	08 06 06 88 12 46 91 fb 36 65 66 26	n 4, Src: Src Port: 0 80 08 00 0 80 40 1: 0 39 08 4:	127.0.0.1, Dst: 127. 37371, Dst Port: 123 37371, Dst Port: 123 3 60 60 69 d4 08 00 1 bc 81 7f 00 00 01 3 fe 56 72 6f 65 69 3 33 32 35 37 31 34	0.0.1 145 16 W @ @ 17 Cohen = > 325	oei 714	any, id 6	1
0008 0018 0028 0038 0049	ux cooked ca ernet Protoc r Datagram P a (59 bytes) 90 90 93 94 45 90 90 57 7f 90 90 91 20 63 6f 68 31 35 32 28	08 06 06 80 12 40 91 fb 36 65 6e 26 2a 2a 2a	n 4, Src: Src Port: Src Port: 0 80 08 00 0 00 40 1: 0 39 08 4: 0 39 08 4: 0 34 3e 2:	127.0.0.1, Dst: 127. 37371, Dst Port: 123 8 88 88 69 d4 88 88 1 bc 81 7f 80 80 91 3 fe 56 72 6f 65 69 9 33 32 35 37 31 34 2 28 28 28 28 69 74	0.0.1 145 E. W. @. @	oei 714 *it	umy, id 6	
8008 8018 8028 8028 8038 8049	ux cooked ca ernet Protoc r Datagram P a (59 bytes) 80 00 03 04 45 00 00 57 7f 00 00 01 20 63 6f 68 31 35 32 72	08 06 06 06 08 12 46 91 fb 36 66 22 28 28 28 28 67 65	n 4, Src: Src Port: 0 80 08 00 0 80 40 1: 0 39 00 4: 0 3d 3e 2: 0 24 2a 2: 0 66 65 66	127.0.0.1, Dst: 127. 37371, Dst Port: 123 37371, Dst Port: 123 3 60 60 69 d4 08 00 1 bc 81 7f 00 00 01 3 fe 56 72 6f 65 69 3 33 32 35 37 31 34	0.0.1 145 16 W @ @ 17 Cohen = > 325	oei 714 *it	any, id 6)

2) **איך הסנפנו את התעבורה בין הלקוח לשרת?** פתחנו ווירשרק, בחרנו להאזין ל any, כלומר כל כרטיס רשת עם הפילטר

udp && udp.port == 12345

סינון זה מוודא שנסניף רק חבילות udp שעוברות דרך פורט 12345 שאנחנו עשינו לו

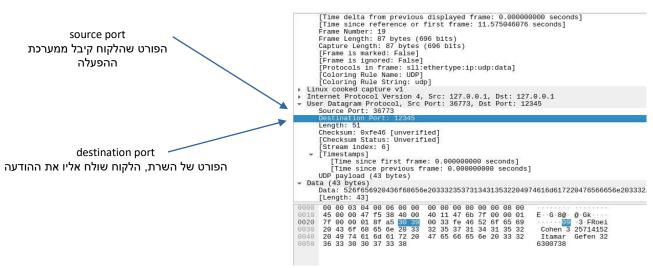
3) השרת השתמש בפורט 12345. זה בא לידי ביטוי בקוד בפונקציה bind שקושרת את הסוקט לפורט.

בקוד של הלקוח אפשר לראות ששולחים לפורט 12345.

הלקוח לא עושה bind לפורט מסוים – הוא מקבל פורט כלשהו ממערכת ההפעלה, והשרת יכול לענות לו כי הלקוח רושם את מספר הפורט שלו בהודעה שהוא שולח לשרת בשכבת התעבורה.

הפורט משמש בשכבת התעבורה. הוא משמש כמזהה יחודי לסוקט (האובייקט שדרכו האפליקציה מקבלת ומעבירה הודעות). בעזרתו שכבת התעבורה בצד המקבל יודעת לאיזה סוקט (אפליקציה) להעביר את ההודעה

החבילה שהלקוח שלח לשרת:



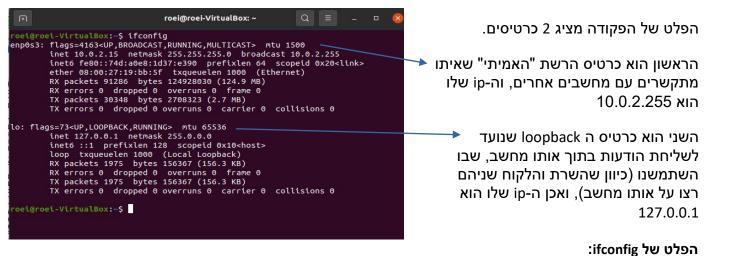
כמו שנאמר לעיל, ניתן לראות בתמונה שבתחילית ששייכת לשכבת התעבורה מצוין שהלקוח שולח את החבילה לפורט מספר 12345 (דסטינייטיון פורט), ורושם גם את הפורט שלו (סורס פורט).

החבילה שהשרת החזיר ללקוח:

ניתן לראות שבתחילית של שכבת התעבורה כתוב שהשרת שולח את החבילה לפורט מספר 36773 (דסטינייטיון פורט) – שזה מה שהיה הסורס פורט בהודעה הקודמת (שהלקוח שלח לשרת).

4) בגלל שהרצנו את הלקוח והשרת על אותו מחשב, הם שלחו אל ומכתובת ה-127.0.0.1 ip . זאת כתובת מיוחדת שמשמעותה "אני". משתמשים בה לתעבורה בין אפליקציות שונות באותו מחשב – כשאין צורך להוציא את המידע החוצה. אפשר לראות את זה בחבילות שהסנפנו בווירשרק:

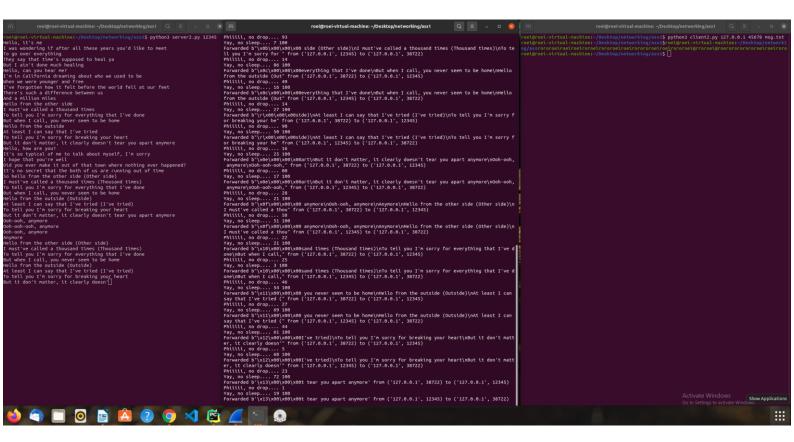
```
Frame 20: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on interface any, id (
 Linux cooked capture v1
▼ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
   0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
- Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
        0000 00.. = Differentiated Services Codepoint: Default (0)
     Identification: 0xf539 (62777)
   Flags: 0x40, Don't fragment
0.... = Reserved bit: Not set
.1.... = Don't fragment: Set
     ..0. .... = More fragments: Not set
Fragment Offset: 0
     Time to Live: 64
Protocol: UDP (17)
Header Checksum: 0x476a [validation disabled]
      [Header checksum status: Unverified]
     Source Address: 127.0.0.1
  User Datagram Protocol, Src Port: 12345, Dst Port: 36773
Data (43 bytes)
      00 00 03 04 00 06 00 00
                                  00 00 00 00 00 00 08 00
                                                              45 00 00 47 f5 39 40 00
                                  40 11 47 6a 7f 00 00 01
                                                                OHEN
0020
      7f 00 00 01 30 39 8f a5
                                 00 33 fe 46 52
                                                  Af.
0030
                   45
4d
0040
0050
```



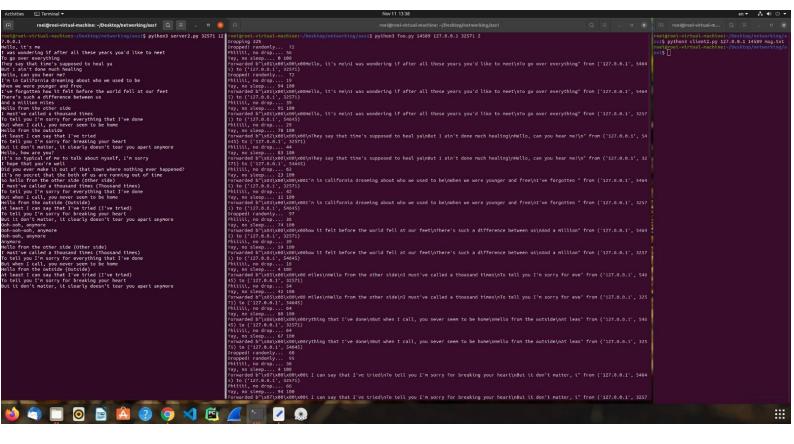
חלק 2

"playing nice" - 1 הרצת משימה

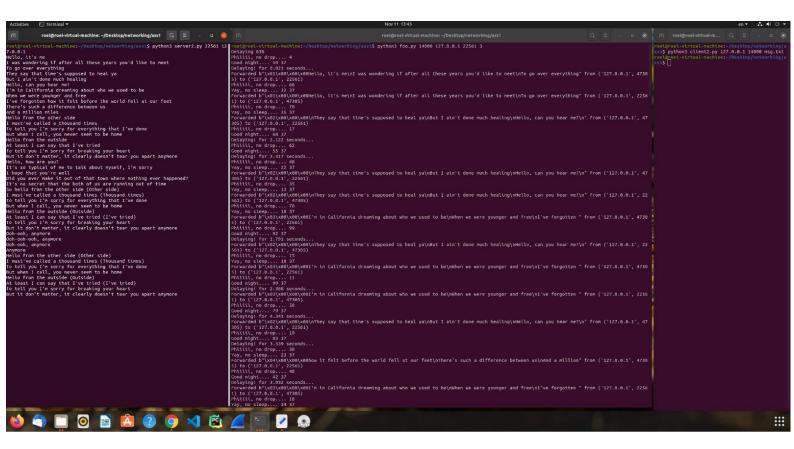
foo לא מעכב חבילות ולא זורק אותן, לכל חבילה שהוא מקבל הוא מוודא שגודלה אינו עולה על 100 בתים (אם כן הוא זורק אותה) ומעביר אותה לנמען. תצלום הרצה:



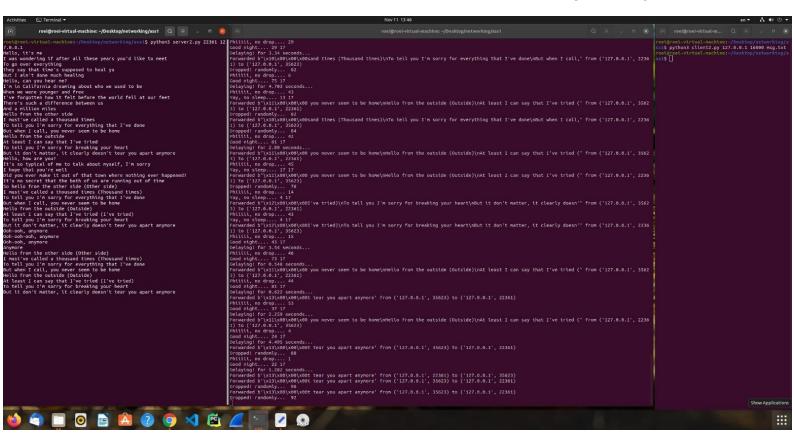
"playing basketball" הרצת משימה 2 – זריקת חבילות באקראיות



"playing grandpa" הרצת משימה 3 – עיכוב חבילות



הרצת משימה 4 – הכל ביחד



איך מימשנו

מימשנו עם stop&wait. כלומר, הלקוח שולח לשרת הודעה אחת בכל פעם ומחכה לתשובה. אם הוא לא מקבל את התשובה תוך הזמן שהגדרנו ל-timeout, הוא שולח שוב, ככה עד שהוא מקבל את התשובה.

בשביל שעיכובים של הודעות לא יגרמו לכך שהשרת לא ידפיס את אותה הודעה פעמיים, הגדרנו "פרוטוקול": הוספנו לתחילת כל הודעה תחילית עם המספר הסידורי של ההודעה (מתחיל ב- 1). ככה השרת יכול להדפיס הודעה רק אם המספר הסידורי שלה גדול ממספר ההודעות שהודפסו עד עכשיו. בדומה, כשהלקוח מקבל תשובה רה הודעה שלתשובה אליה הוא מצפה.

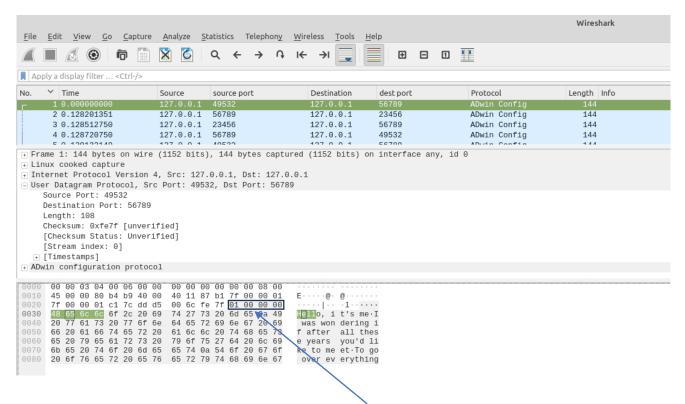
נדגים בעזרת צילומי מסך מההסנפות של ווירשארק: (השרת האזין בפורט 23456 ו- foo האזין בפורט 23456 5078. במקרה הזה יצא שהלקוח האזין בפורט 49532. לכן, כשנראה בווירשארק למשל סורס פורט 23456 ודסטינייטיון פורט 56789 נדע שמדובר בהודעה שנשלחה מהשרת ל- foo.)

התעבורה בווירשארק: השרת האזין בפורט 23456 ו- foo האזין בפורט 56789. לכן השתמשנו בפילטר

udp && (udp.port == 12345 || udp.port == 45678)

כדי לסנן את כל התעבורה, ושמרנו בקובץ.

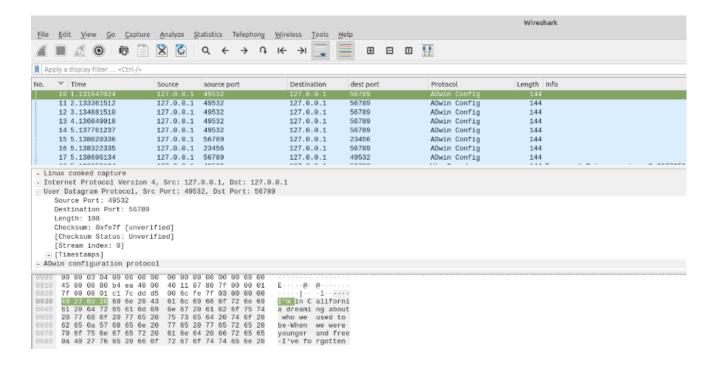
מקרה foo – 1 לא עושה בעיות.



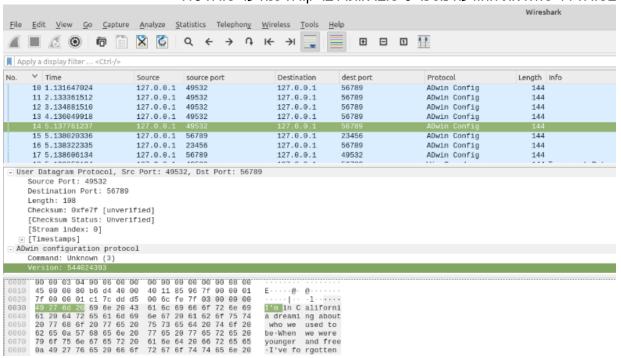
בשורה 1 (המודגשת) הלקוח שולח ל- foo את הודעה מספר 1. בשורה 2 foo מעביר אותה לשרת, בשורה 3 השרת שולח את התשובה ל- foo, ובשורה 4 foo מעביר את התשובה ללקוח.

הלקוח מקבל תשובה, ולכן בשורה 5 הוא כבר שולח ל- foo את הודעה מספר 2.

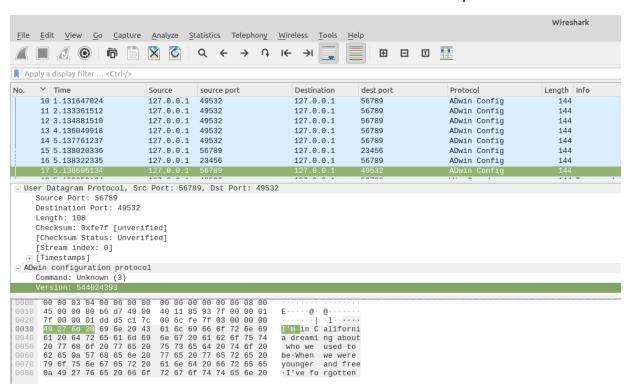
מקרה 2 – השרת זורק/מעכב הודעה



בשורה 10 (המודגשת) הלקוח שולח ל- foo את הודעה מספר 3. foo זורק/מעכב את ההודעה (זה לא ממש משנה לצורך העניין), ולכן עוברת שנייה, זמן ה- timeout שהגדרנו, והלקוח עדיין לא קיבל תשובה. לכן, בשורה 11 שולח את ההודעה מספר 3 שוב. אותו דבר קורה ככה עד שורה 15.

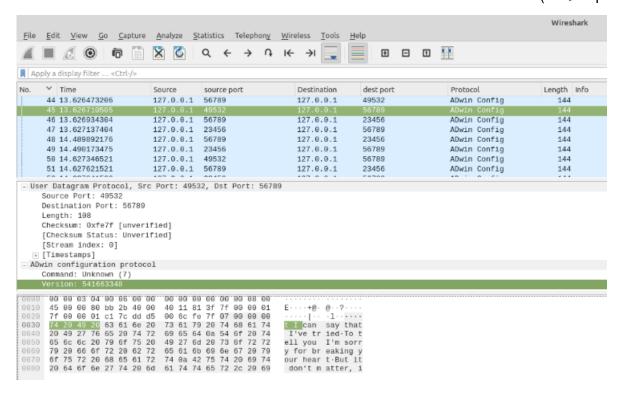


בשורה foo 15 סוף-סוף מעביר לשרת, ובשורות שלאחר מכן ניתן לראות שהשרת עונה ל- foo ובשורה 17 foo מחזיר את התשובה ללקוח.

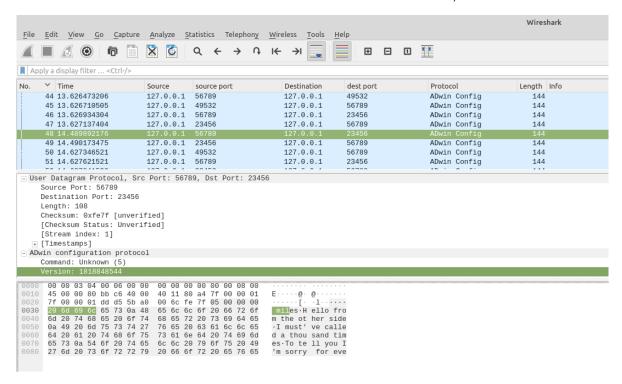


בשורה 18 הלקוח כבר ישלח את הודעה מספר 4.

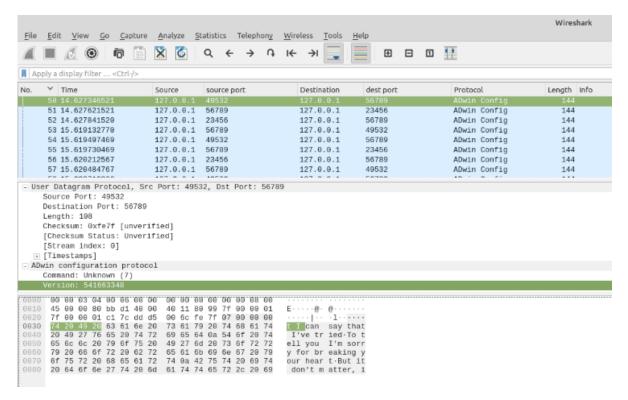
מקרה foo – 3 מעכב הודעה של הלקוח, וכתוצאה מכך היא מגיעה לשרת אחרי שהוא כבר הדפיס אותה. כלומר, השרת מקבל הודעה עם מספר שקטן ממספר ההודעה שאליו הוא מצפה. במקרה זה השרת לא ידפיס אותה שוב, אלה רק ישלח אותה בחזרה ללקוח (כדי שאם הלקוח לא קיבל את התשובה בפעם הקודמת הוא יקבל עכשיו).



בשורה 45 (המודגשת) הלקוח שולח בפעם הראשונה את הודעה 7 ל- foo. בשורה 46 foo מעביר אותה לשרת. השרת מדפיס אותה, ובשורה 47 הוא שולח את התשובה אליה ל- foo.



כפי שניתן לראות, בשורה 48 השרת פתאום מקבל מ- foo את הודעה 5. השרת לא ידפיס אותה, אלה רק ישלח בחזרה תשובה עליה (שורה 49).



ניתן לראות ש- foo זרק/עיכב את התשובה שהשרת שלח על הודעה 7, ולכן (בשורה 50) הלקוח שולח אותה שוב. בשורה 53 הלקוח כבר מקבל את התשובה עליה.

באופן דומה, אם foo מעכב תשובה של השרת, וכתוצאה מכך היא מגיעה ללקוח כשהוא כבר מצפה לתשובה על הודעה אחרת - כלומר הלקוח מקבל תשובה עם מספר שקטן מהמספר שאליו הוא מצפה, אז הלקוח יתעלם מהתשובה הזאת. הוא ימשיך לחכות לתשובה המתאימה.