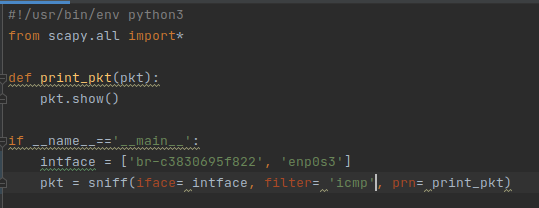
ב"ה

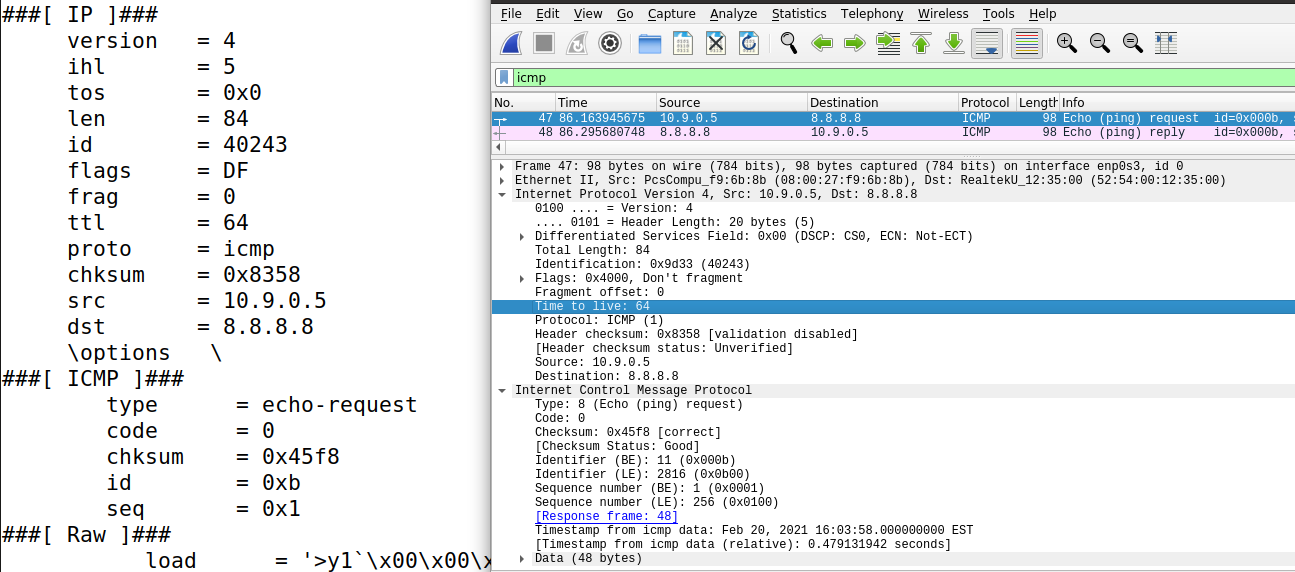
פרויקט גמר בתקשורת

משימה 1.1:

חלק A:

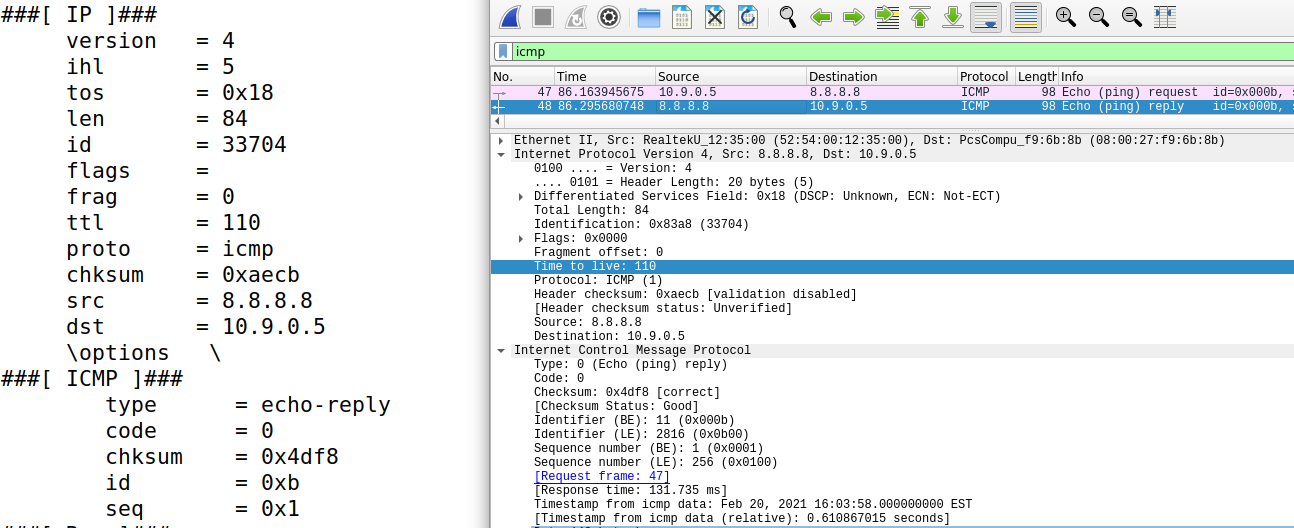
כאן התבקשנו לבנות סניפר בפייתון, ולפלטר לפי icmp. להלן הקוד:

ניתן לראות את תוצאת ההרצה של הקובץ עם הרשאת גישה (sudo), וניתן לראות שהיא זהה לפאקטות שאנו רואים בwireshark:

request:

wireshark

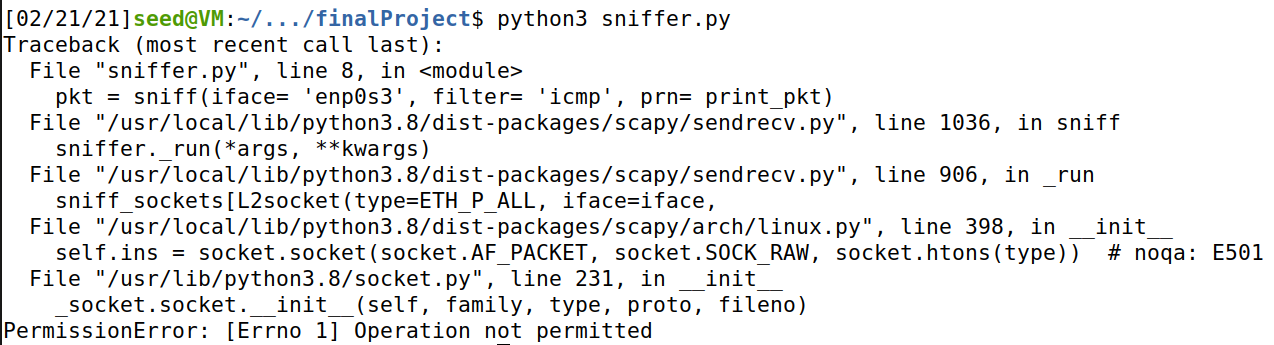
terminal

Reply:

wireshark

terminal

כאן ניתן לראות את תוצאת ההרצה של הקובץ בלי הרשאת גישה (sudo). הסיבה שמתקבלת שגיאה, כי האזנה לפקטות זה דבר שדורש הרשאת מנהל. בלי sudo אין הרשאת מנהל, ולכן לא ניתן להפעיל ,raw socket ולכן מתקבלת שגיאה.

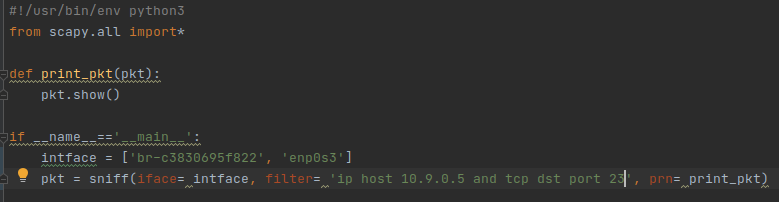
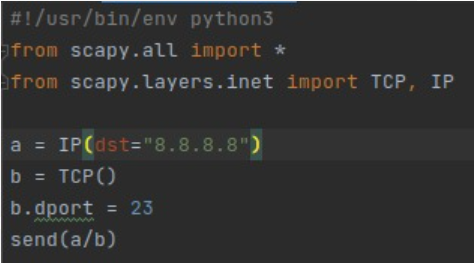


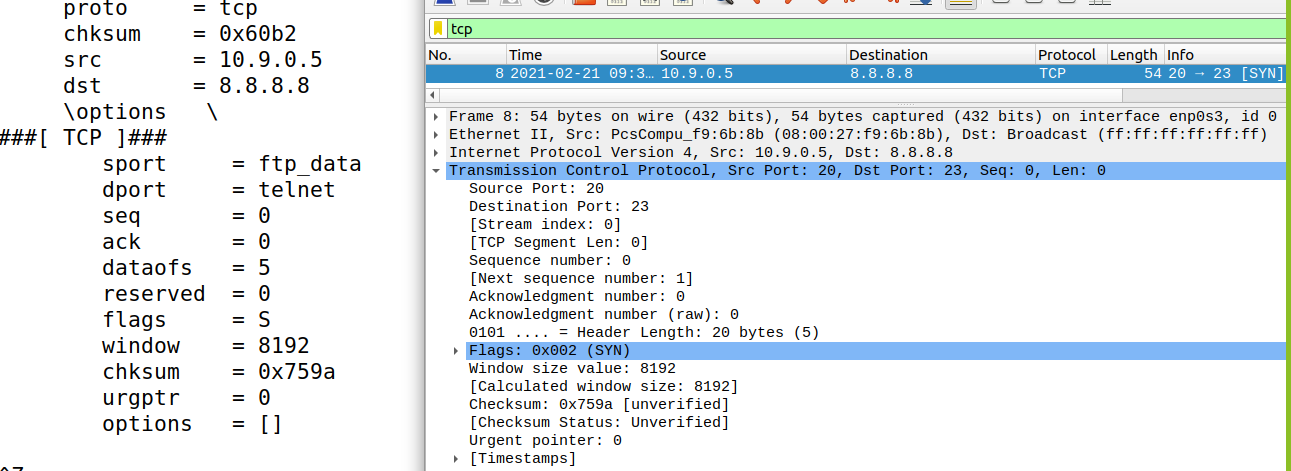
חלק B:

בחלק הזה נתבקשנו לפלטר את הסניפר שבנינו לפי סוגי פקטות:

-לפי icmp: ניתן לראות זאת בתמונה בחלק A, שם הפילטר כבר מסונן לפי icmp

-לפי TCP: כאן אנו מפלטרים פקטות TCP עם פורט יעד 23 עבור IP מסוים (8.8.8.8):

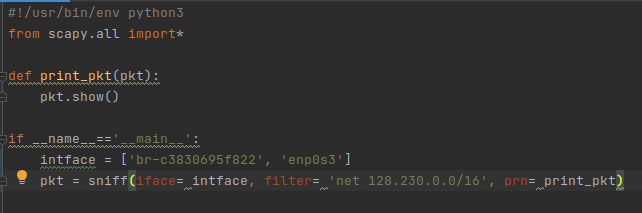
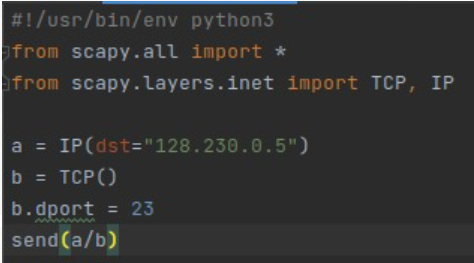
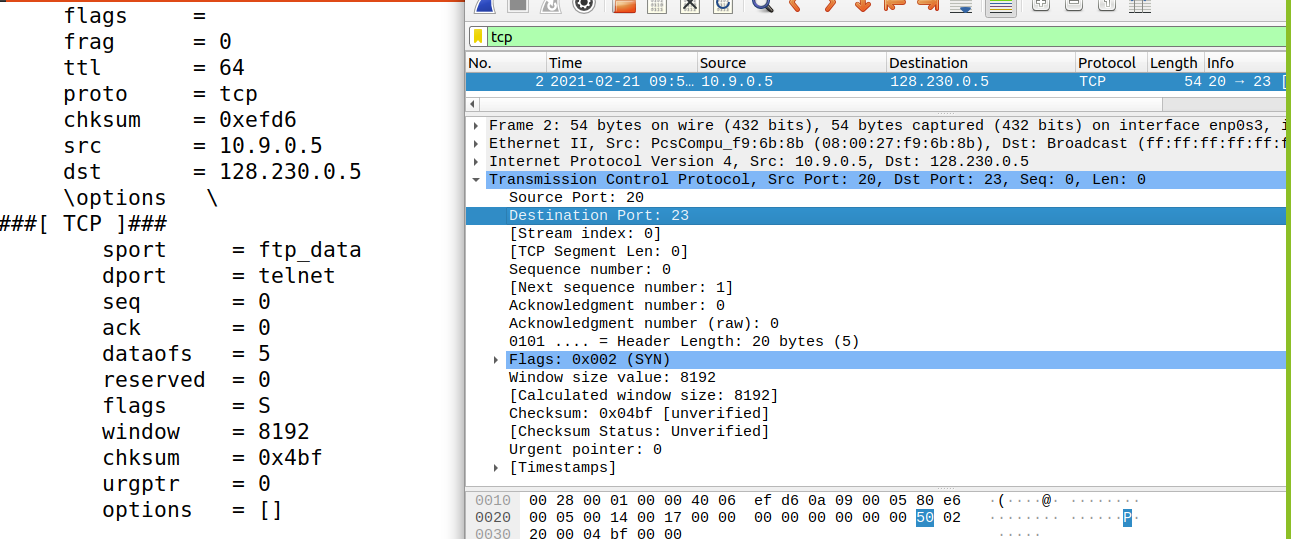


וכאן ניתן לראות שקיבלנו אותן פקטת TCP בטרמינל ובווירשארק:

wireshark

terminal

-כאן נתבקשנו ללכוד פקטות שרצות ברשת פנימית מסוימת (128.230.0.0/16):

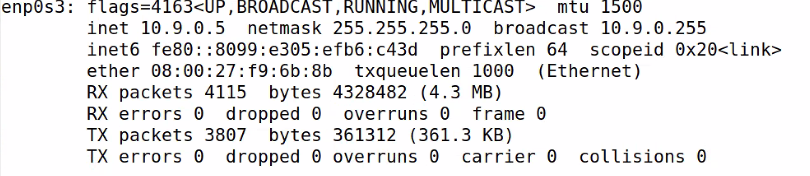
כאן ניתן לראות שקיבלנו תוצאה זהה בטרמינל ובווירשארק:

wireshark

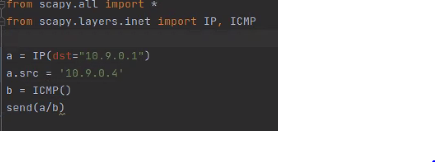
terminal

משימה 1.2:

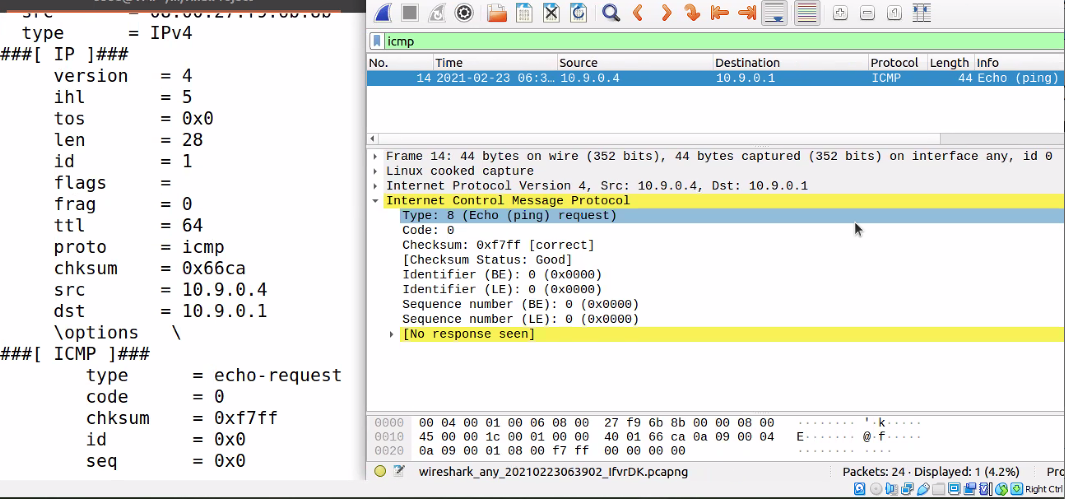
בשאלה זו אנו מתבקשים לבצע spoofing ברשת פנימית באמצעות icmp request:

ראשית, נשים לב מה הip שלנו: 10.9.0.5:



כעת, נבצע spoofing: נתחזה לip 10.9.0.4 ונשלח פאקטת icmp לip 10.9.0.1:



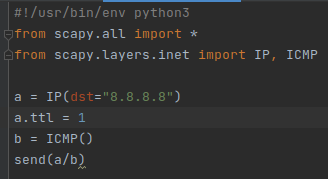
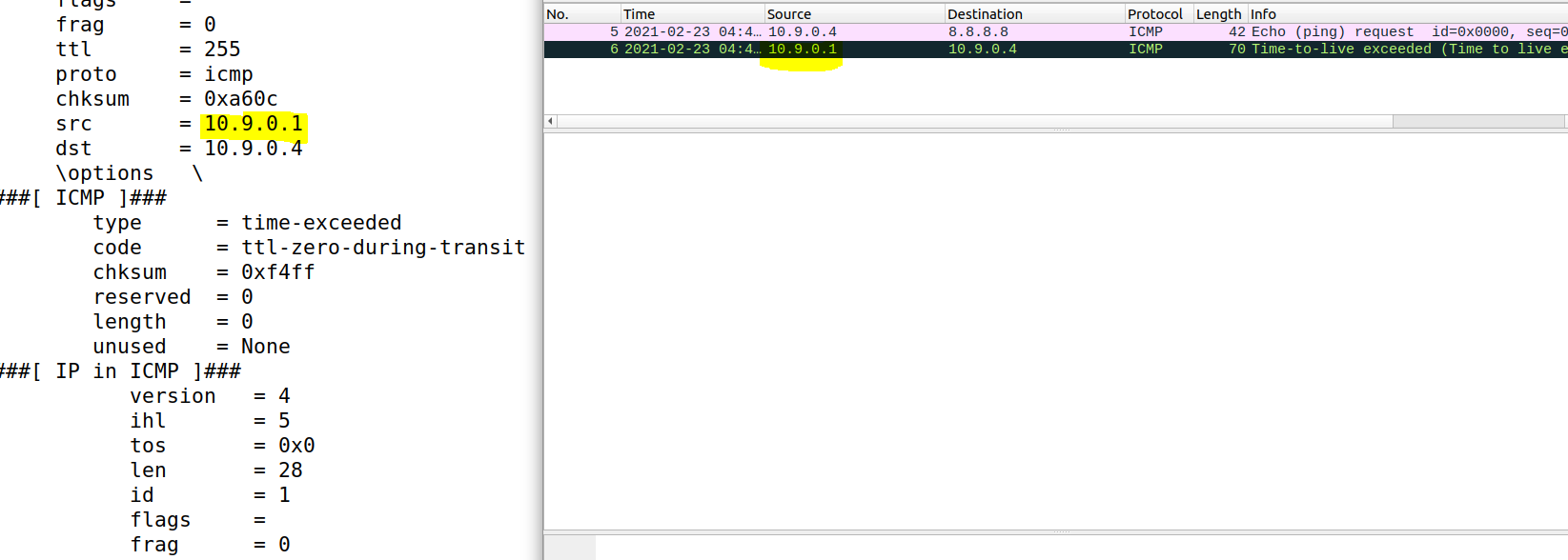
נריץ את הסניפר בטרמינל שמפלטר לפי icmp ונריץ את הקובץ בציור הנ"ל בטרמינל. ניתן לראות שבפרטי הrequest הsrc הוא 10.9.0.4 (כשבפועל, הip שלנו הוא 10.9.0.5):

wireshark

terminal

משימה 1.3:

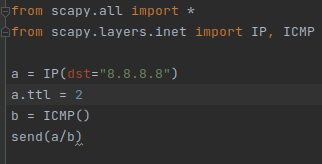
נשלח בקשת icmp לip 8.8.8.8 ונראה כמה נתבים הפקטה צריכה לעבור בדרך על מנת להגיע ליעדה.

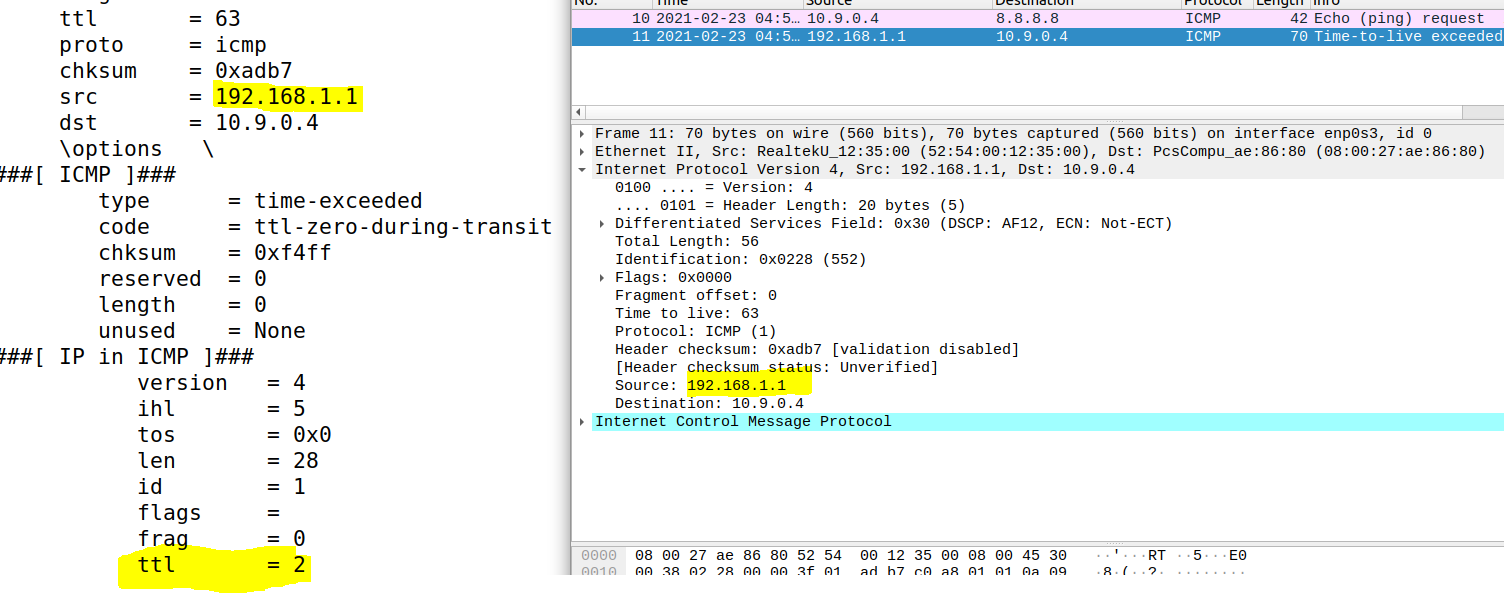
כאשר ttl = 1, הפאקטה תשרוד עד הנתב שכתובתו היא 10.9.0.1:

wireshark

terminal

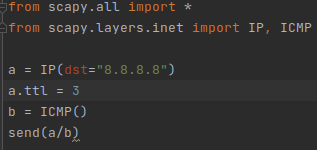
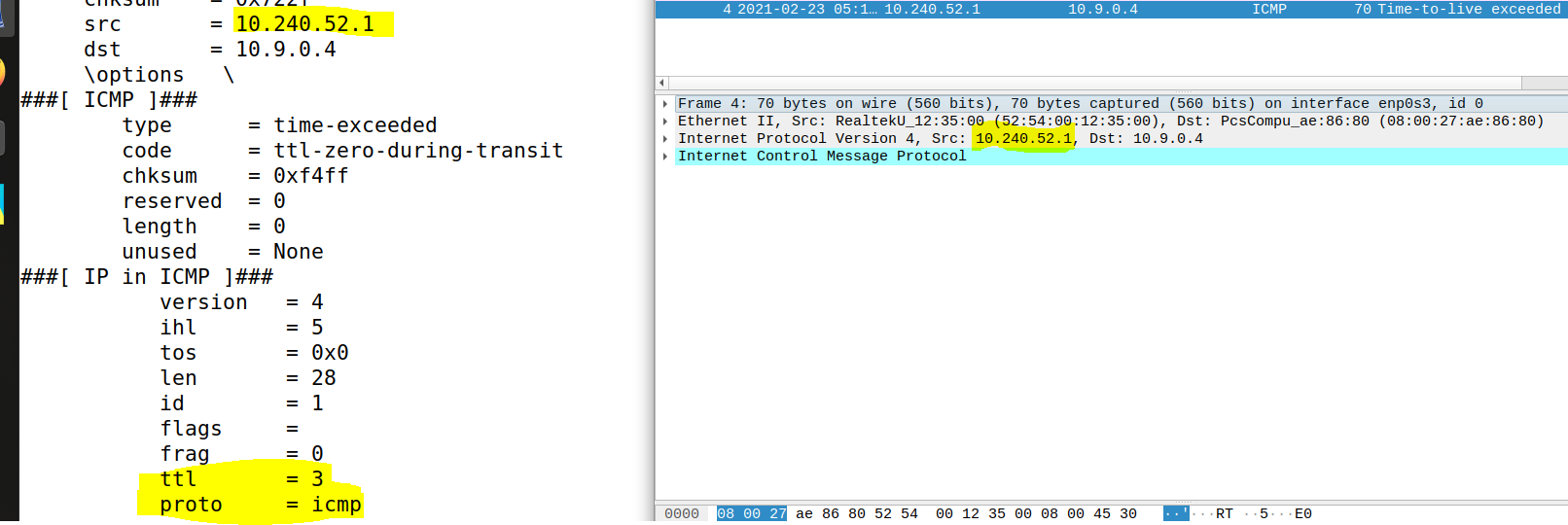
התחנה השנייה היא נתב שכתובתו 192.168.1.1:





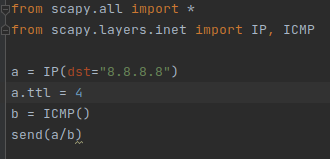
wireshark

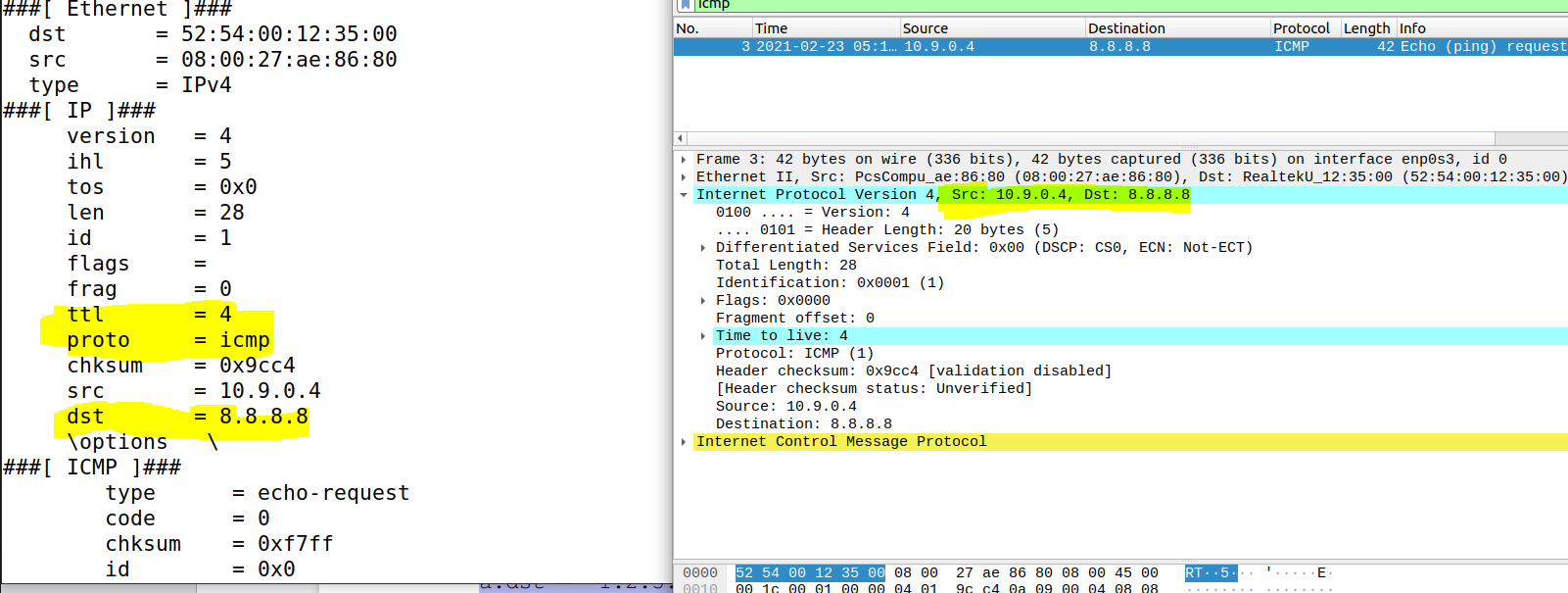
terminal

התחנה השלישית היא 10.240.52.1:

wireshark

terminal

התחנה הרביעית היא האחרונה, כלומר 8.8.8.8 וttl = 4:



wireshark

terminal

ניתן לראות שהפעם בניגוד לפעמים הקודמות, אין הודעה על נפילת הפקטה בדרך.

משימה 1.4:

משימה 2.1A:

1. נסביר את הקריאות בסניפר שכתבנו:

pcap\_open\_live – פתיחת raw socket והשמת כרטיס הרשת במצב האזנה

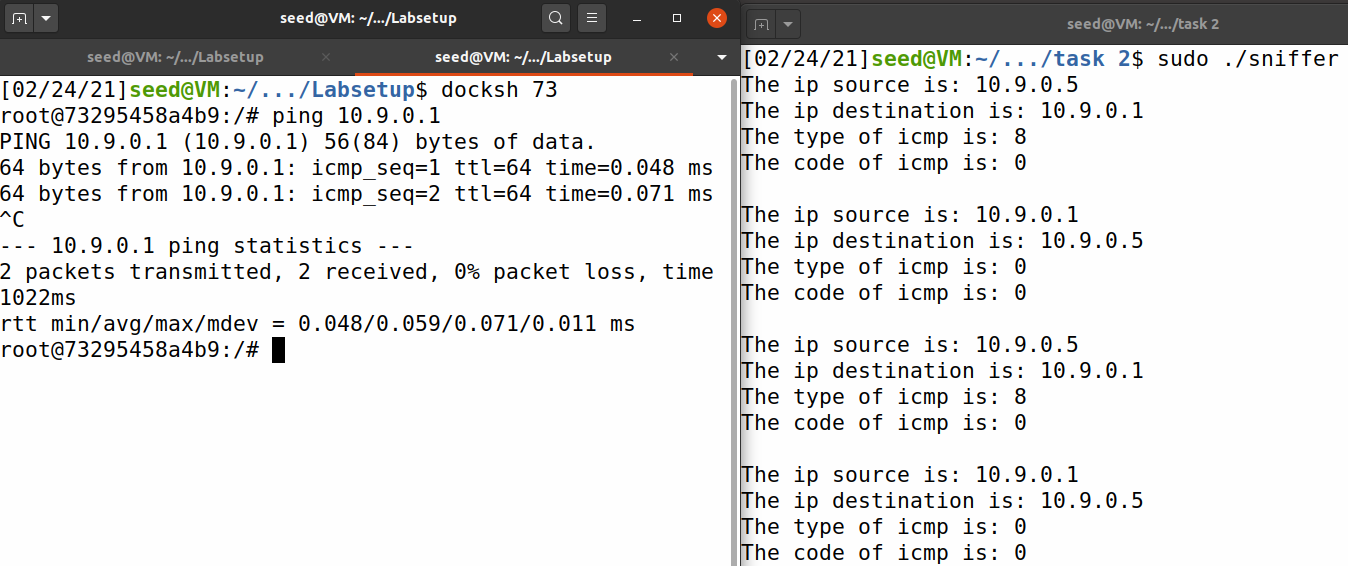
pcap\_compile&pcap\_setfilter – פלטור פאקטות: כלומר, אנו מגדירים לסוקט אילו פקטות להרים (פקטות מסוג icmp, או מסוג tcp, או מסוג udp וכיו"ב), ע"פ סינטקס BPF

pcap\_loop – פקודה זו אומרת לבצע פונקציה x, אחרי שהורמה פאקטה ע"פ הפילטר שהגדרנו (בקוד שלנו, x זו המתודה got\_packet שמדפיסה את פרטי הפאקטה)

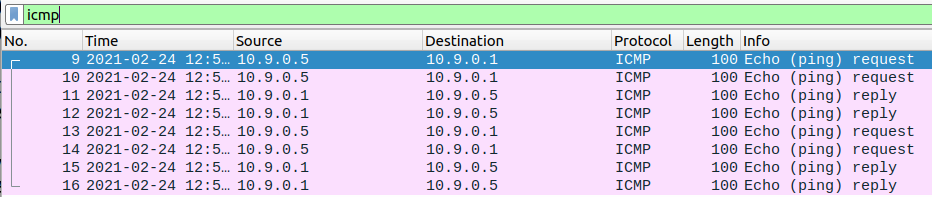
pcap\_close - סגירת הraw socket

1. הסיבה שדרושה הרשאת מנהל היא

משימה 2.1B:

-נסניף תעבורת icmp בין 2 הקונטיינרים שלנו בdocker: host, שכתובתו 10.9.0.5 ישלח ping ל10.9.0.1 (הכתובת של seed-attacker). הקוד של הסניפר מצורף בתיקייה שנקראת task2):

בחלון השמאלי שלחנו ping ל10.9.0.1, ובחלון הימני ניתן לראות שהסניפר מדפיס תעבורת icmp: על כל request (type=8), נקבל reply ((type=0. את הסוקט שכתבנו בקוד, הגדרנו עבור interface של אחד הקונטיינרים, ולכן רואים reply אחרי request יחיד, ולהפך.



כאן בwireshark, הגדרנו האזנה לכל ה interfaces ברשת הפנימית, ולכן כל request ו replyמאוגדים בזוגות: פעם אחר שהמקור אכן שלח, ועפ אחת עבור היעד שאכן קיבל.