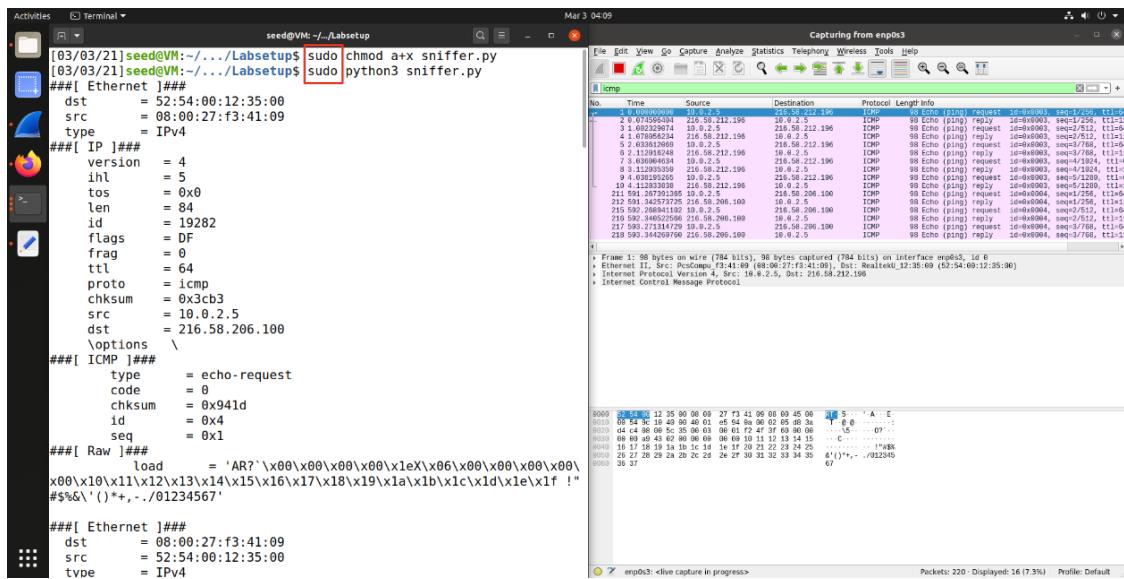


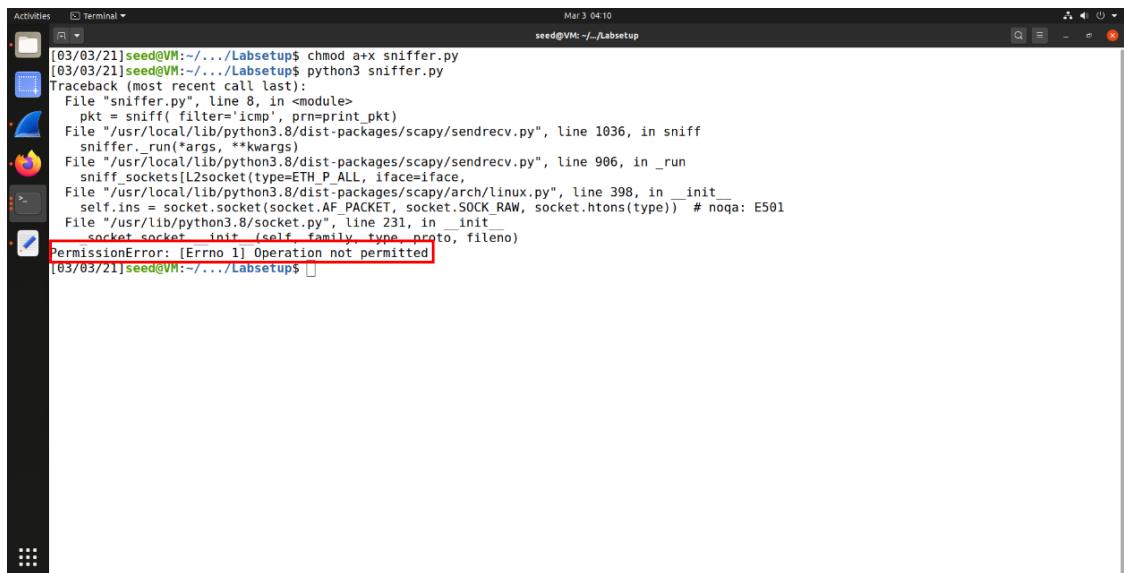
## תקשורת ומחשבוב – פרוייקט גמר:

### Sniffing Packets :A1.1

עם הרשות root:



ללא הרשות root:



לאחר שהרכינו את תוכנת ההסנפה עם הרשות מנהל (sudo), ראיינו כי התוכנה אכן מסניפה פקודות כמצופה ממנה כמתואר בתמונה הראשונה. לאחר מכן הרכינו את תוכנת ההסנפה ללא הרשות מנהל והתוכנה שלנו קרסה, קיבלנו את הערכה: PermissionError: [Errno 1] Operation not permitted

:B1.1

## Filter – ICMP:

## Filter – TCP in port 23:

```
###[ Ethernet ]##
dst      = 52:54:00:12:35:00
src      = 08:00:27:c8:92:b2
type     = IPv4
###[ IP ]##
version  = 4
ihl      = 5
len      = 60
tos      = 0x10
len      = 60
id       = 64955
flags    = DF
frag     = 0
ttl      = 64
proto    = tcp
checksum = 0x20c
src      = 10.0.2.6
dst      = 8.8.8.23
options  \
###[ TCP ]##
sport    = 35846
dport    = telnet
seq      = 4175210575
ack      = 0
dataofs = 10
reserved = 0
flags    = S
window   = 64240
checksum = 0xc39f
urgptr  = 0
options  = [('MSS', 1460), ('SAckOK', b''), ('Timestamp', (24
77567060, 0)), ('NOP', None), ('WScale', 7)]
^C[03/07/21]seed@VM:~/.../python$
```

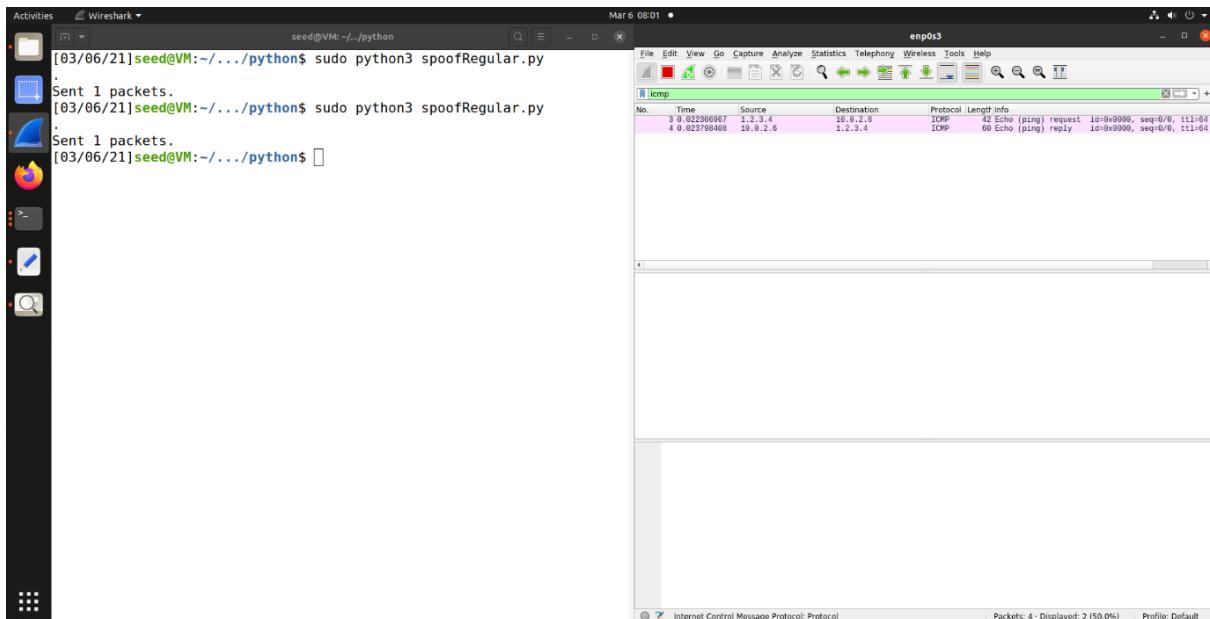
### **Filter – Subnet:**

סיננו את כתובות subnet ה-IP: "8.8.8.0" עד IP: "8.8.8.24". על פי התמונה ניתן לראות כי תוכנית הסנספה הסניפה ping reply&request ל-

## Spoofing ICMP Packets :1.2

התוקף המשמש בתוכנתה ל-IP "1.2.3.4" אשר אין קיימן כל ברשט ולמעשה שלוח ping request ל-VM השני (הנתזק). מחשבו של הנתזק מוחזיר והודעת reply חזרה, אולי קיבל ping מכתובות אינטרנטית אמיתי ואינו יודע למעשה כי הכתובת זו אינה קיימת.

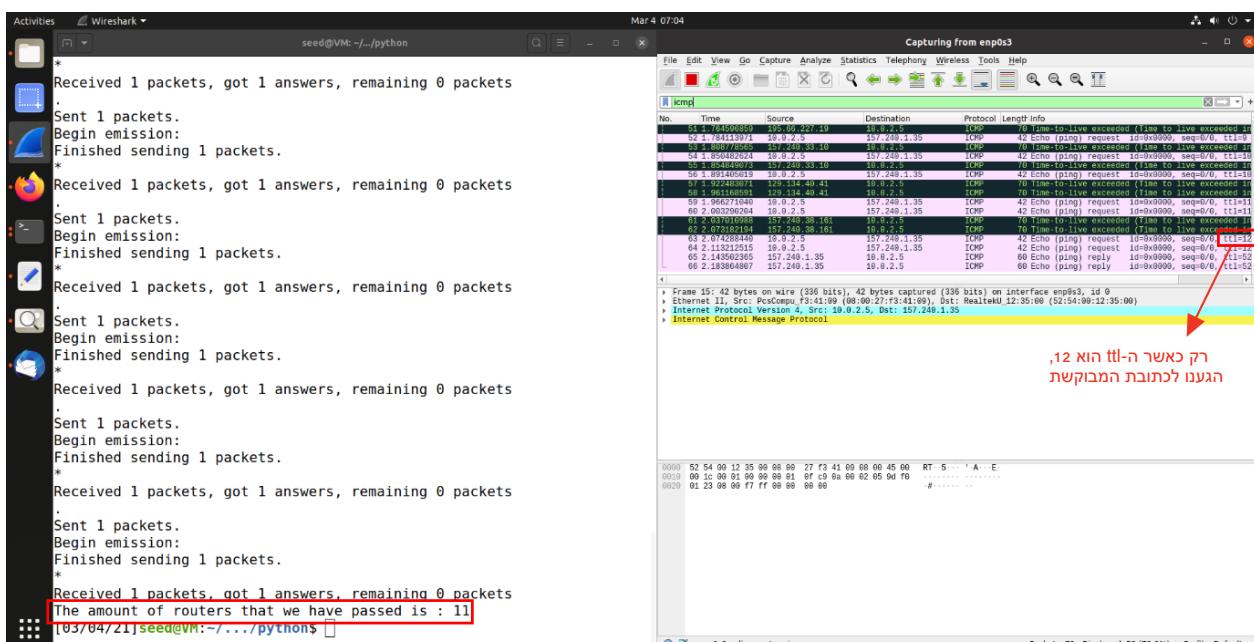
ניתן לראות בתמונה כי שלחנו למחשב הנמצא ב-VM השני ping מכתובות "1.2.3.4" אשר אינה קיימת והמחשב הנמצא ב-VM השני ענה בחזרה.



## Traceroute :1.3

בשאלה זו התבוננו לדעת כמה ראותרים עוברים מהרשות האישית שלנו עד כתובת IP מסוימת בעת שליחת Ping לכתובת זו. אנו בדקנו זאת על ידי שליחת Ping לכתובת של Facebook וגילינו שככל שה-TTL (Time To Live) יותר גדול כך אפשר להגיע לראותר יותר רחוק עד הגיעו לכתובת המבויקשת.

לפי התמונה ניתן לראות שהיו 11 מעברים בין ראותרים עד שהגענו לכתובת של Facebook.



## Sniffing and-then Spoofing :1.4

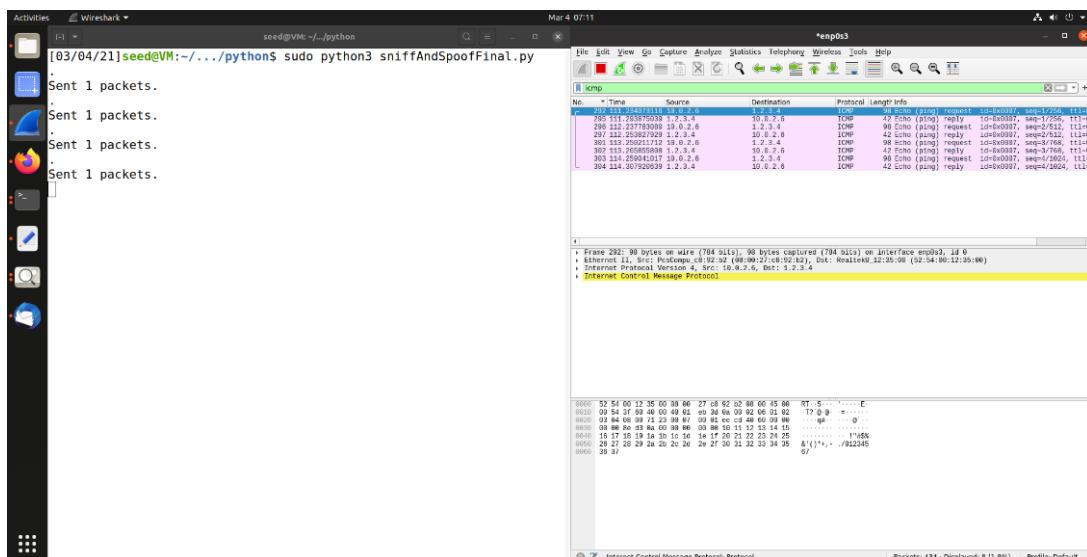
בכדי לענות על שאלת זו פתחנו שני מחשבים המודמים תוקף ונתקף.

הנתקף מפעיל את התוכנית sniffAndSpoof.py לצורן גיבית מיידע מן הנטקף.

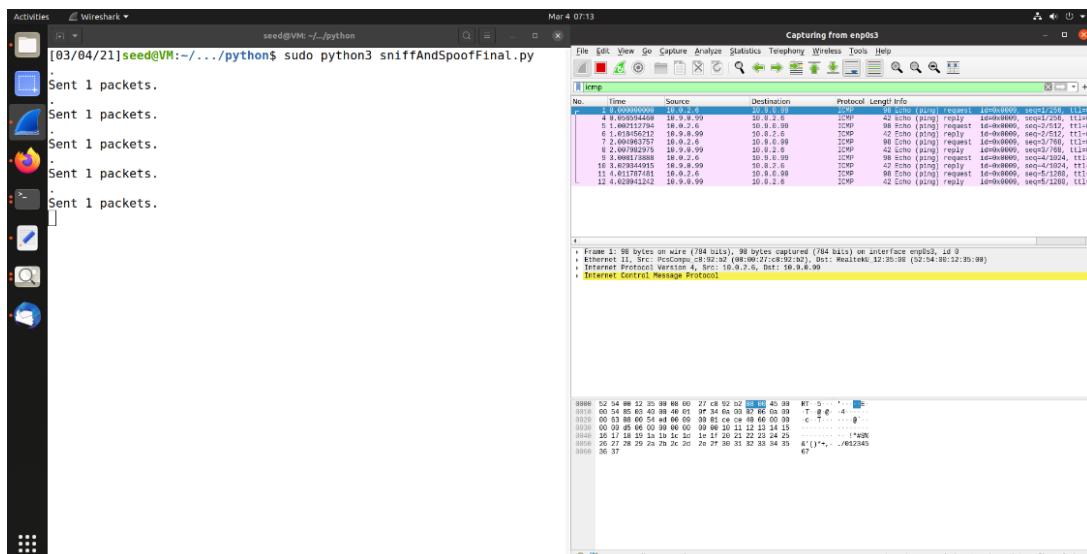
בהתחלת הנטקף שולח ping לשתי כתובות אשר לא קיימות, הנטקף מסניף את פקודותicmp העוברות בראשת, ושולח לנתקף הודעה reply מאותו ping Cainו הכתובת קיימת.

בנוסף לכך, הנטקף שולח Ping גם לכטובות קיימת ("8.8.8.8") אשר שולחת לו הודעה reply חזרה. הנטקף חושב שהוא קיבל reply מ-"8.8.8.8" אך האמת שהנתקף הוא זה ששלח את הודעה reply. משום שזו כתובות הקיימת כבר באינטרנט, תישלח תשובה פעמים – גם מהנתקף וגם מהכתובת עצמה (המקורית), ניתן לראות זאת בתמונה למטה – תודפס לנו הודעה (!DUP) בסוף שורת ההודעה הכפולה שמתאריה למשתמש שפקטה זו נשלחת משני מקומות שונים.

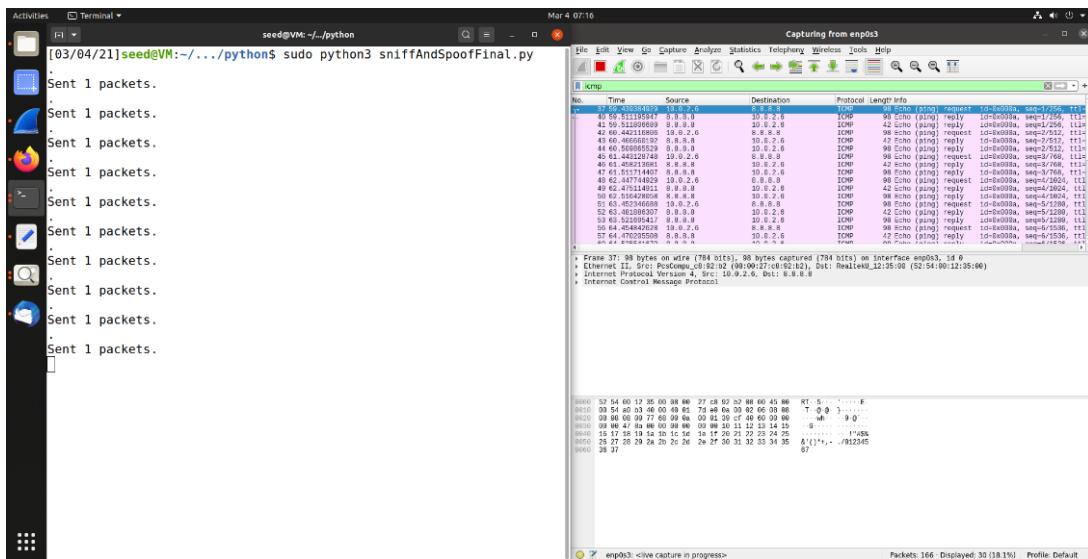
Ping: "1.2.3.4"



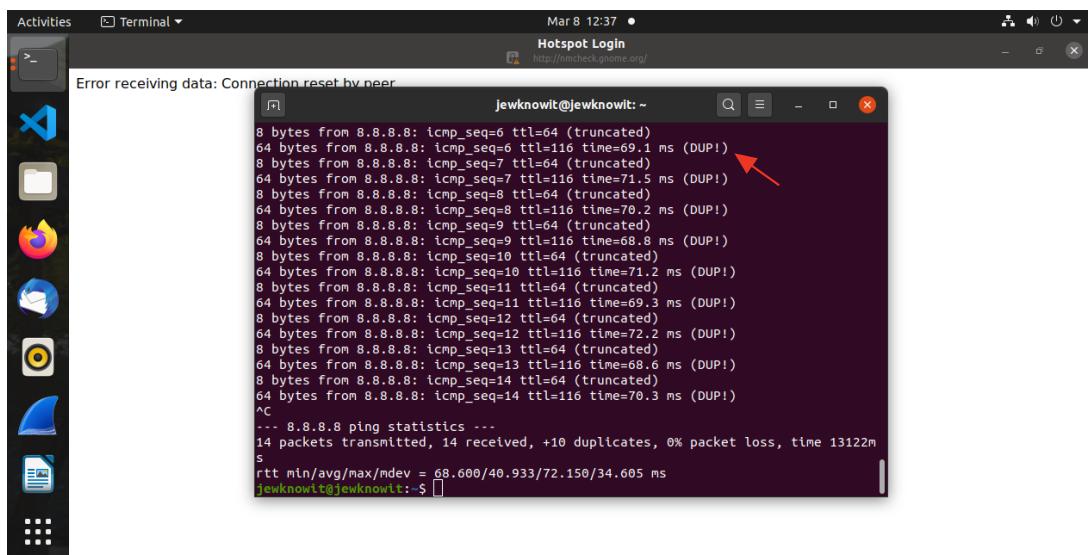
Ping "10.9.0.99"



## Ping "8.8.8.8"



שליחת הודה מכותבת "8.8.8.8" לנתקן.



## Understanding How a Sniffer Works :A2.1

### • Question 1:

Please use your own words to describe the sequence of the library calls that are essential for sniffer programs. This is meant to be a summary, not detailed explanation like the one in the tutorial or book.

תשובה:

תחילה, אנו צריכים לאתחל את המחשב בו אנו רוצים להשתמש בתוכנת ההסנפה ולבחר אות האינטראפיו אותו אנו רוצים להסניף, אנו משתמשים בשיטה של ספריית pcap: `pcap.open_live` אשר מאפשרת לנו להסניף את האינטראפיו. תוכנת ההסנפה יכולה להסניף את כל התעבורה האינטרנטית הנמצאת על אותה הרשת, כמו כן, כל משתמש אשר עבד על הרשות חשוף לתקוף.

על מנת לגנוב את המידע הרלוונטי, התוקף מפעיל סינון ראשי של פרוטוקולים ספציפיים כמו למשל `.pcap_setfilter`, `tcp`, `icmp` אשר אותו הוא חושך להשיג, על ידי שימוש בפעולות `pcap_compile`.

תוכנת ההסנפה משתמשת בשיטת `pcap_loop`, מההה לטעבורה ברשות, ברגע שתזיהה תעבורה, התוכנית תסניף את כל הפקות הרצויות על פי הסינון המבוקש. לפי שיטה זו התוכנה לא תפסיק לעבוד עד שהתוקף קיבל את המידע הרצוי.

### • Question 2:

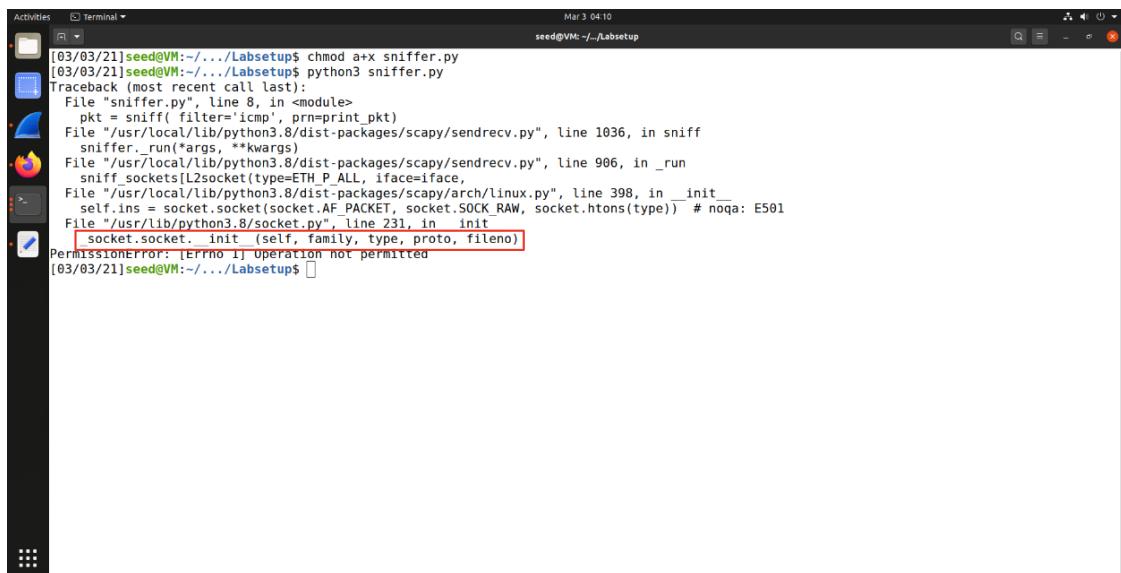
Why do you need the root privilege to run a sniffer program? Where does the program fail if it is executed without the root privilege?

תשובה:

במערכות הפעלה Linux, יש את הצורך להפעיל את פיקודת "הרשאות מנהל" (sudo) על מנת להשתמש בתוכנת ההסנפה משום שבתוכנה זו קיימ שימוש ב-Raw Sockets המצריכים הרשות למשוך האינטרנטית המבוקש "הרשאות מנהל" בעצמו.

כלומר, על מנת ליצור Raw Socket יש צורך בהרשאות מנהל, כי רק כך יוכל לעבור דרך הממשק האינטרנטית.

במידה ולא ניתנה "הרשאות מנהל", תוכנת ההסנפה תקרס בעת יצירת ה-Raw Socket. בתמונה שלפנינו ניתן לראות כי התוכנית ללא הרשות מנהל, קורסת בשורה בה יש ניסיון ליצור raw socket.



The screenshot shows a terminal window on a Linux desktop environment. The terminal output is as follows:

```
[03/03/21]seed@VM:~/.../Labsetup$ chmod a+x sniffer.py
[03/03/21]seed@VM:~/.../Labsetup$ python3 sniffer.py
Traceback (most recent call last):
  File "sniffer.py", line 8, in <module>
    pkt = sniff(filter='icmp', prn=print_pkt)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 1036, in sniff
    sniffer._run(*args, **kwargs)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 906, in _run
    sniff_socket=Lsocket(type=ETH_P_ALL, iface=iface,
  File "/usr/local/lib/python3.8/dist-packages/scapy/arch/linux.py", line 398, in __init__
    self.ins = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.htons(type)) # noqa: E501
  File "/usr/lib/python3.8/socket.py", line 231, in __init__
    socket.socket.__init__(self, family, type, proto, fileno)
PermissionError: [Errno 1] operation not permitted
[03/03/21]seed@VM:~/.../Labsetup$
```

The last line of the output, which contains the permission error, is highlighted with a red box.

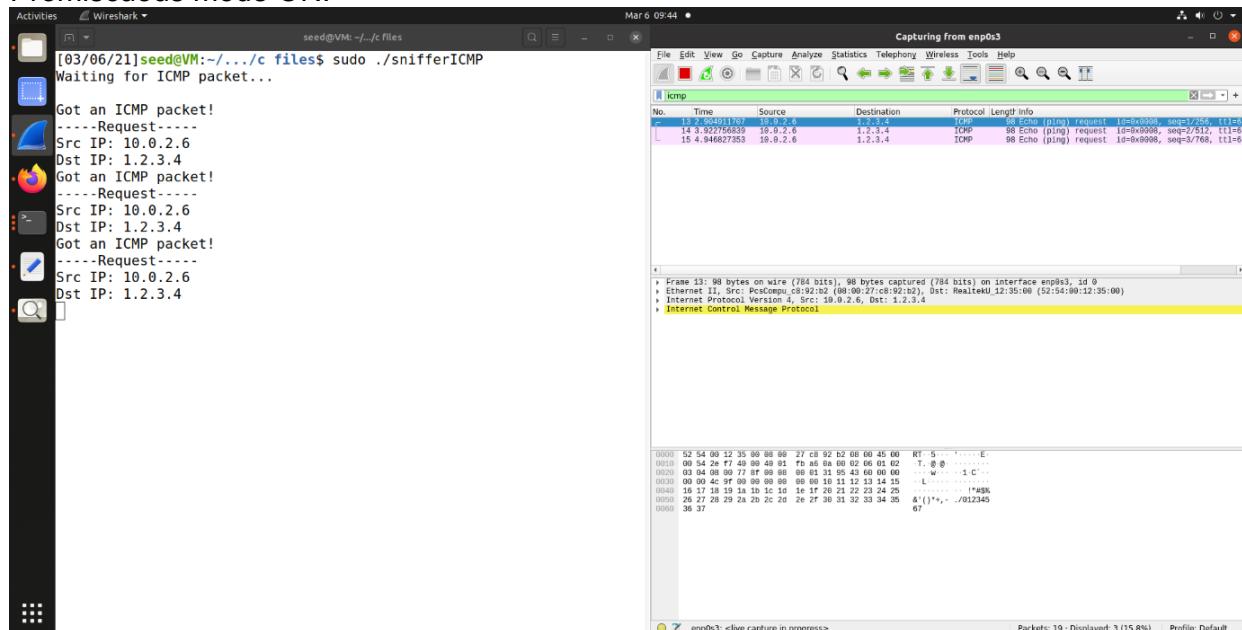
- **Question 3:**

Please turn on and turn off the promiscuous mode in your sniffer program. Can you demonstrate the difference when this mode is on and off? Please describe how you can demonstrate this.

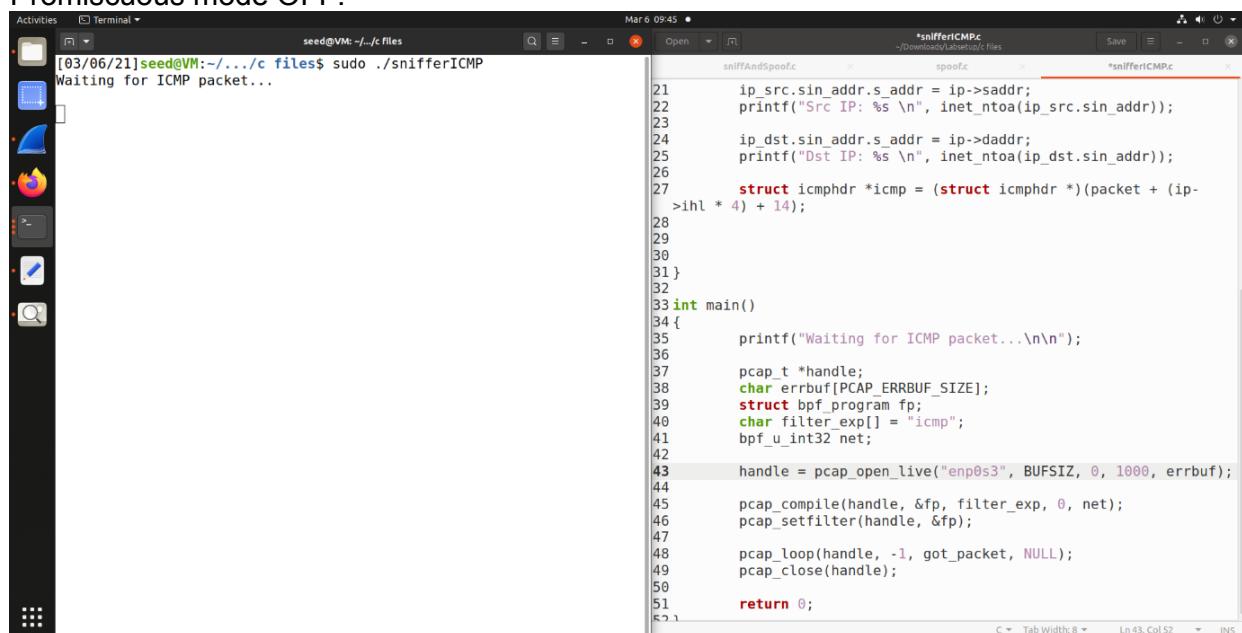
## תשובה:

ניתן להסיק מן התמונה שכאשר Promiscuous mode כבוי, כלומר במצב "0", תוכנת ההסנהה שלנו לא מצליחה לגלוות שום פקודות העוברות בתעבורה האינטראנטית מה-VM השני (הנתקף). לעומת זאת, כאשר Promiscuous mode>DЛОק כלומר במצב "1", תוכנת ההסנהה שלנו מצליחה להסניף מידע מן המשתמשים המצויים בראשת.

Promiscuous mode ON:



Promiscuous mode OFF:

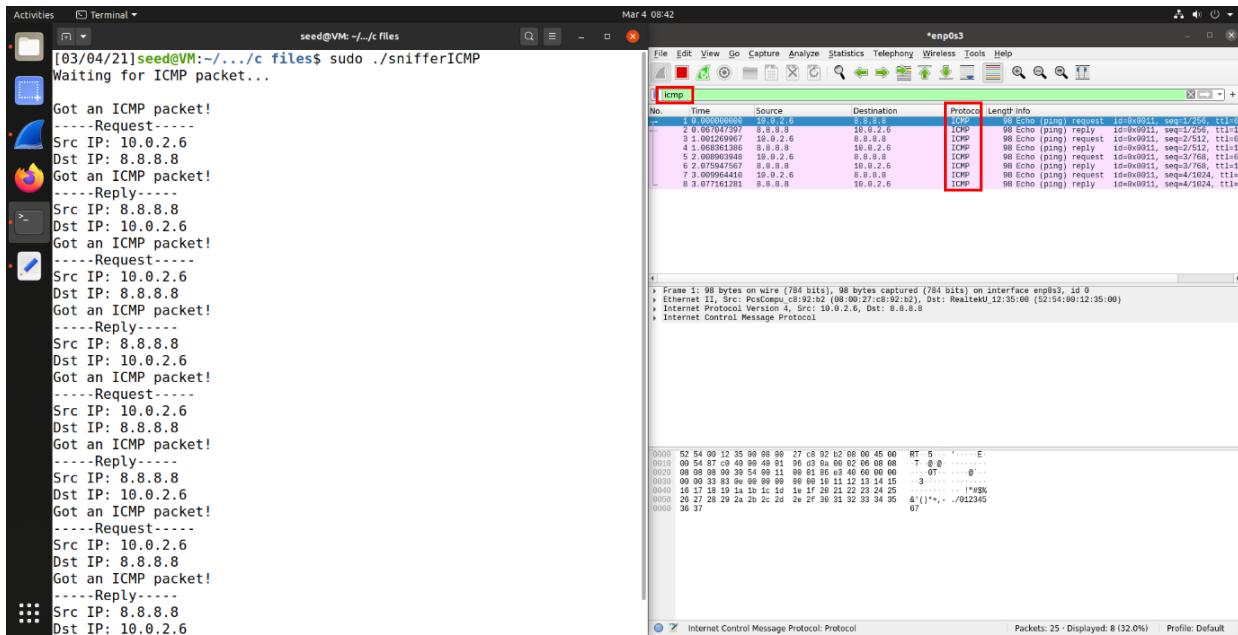


## Writing Filters :B2.1

בשאלה זו, התקשנו ליצור תוכנות הסנהה ב-C אשר מפעילות מגנוני סינון רלוונטיים לשאלת, אולם ניתן לראות לפי התמונות המוצגות:

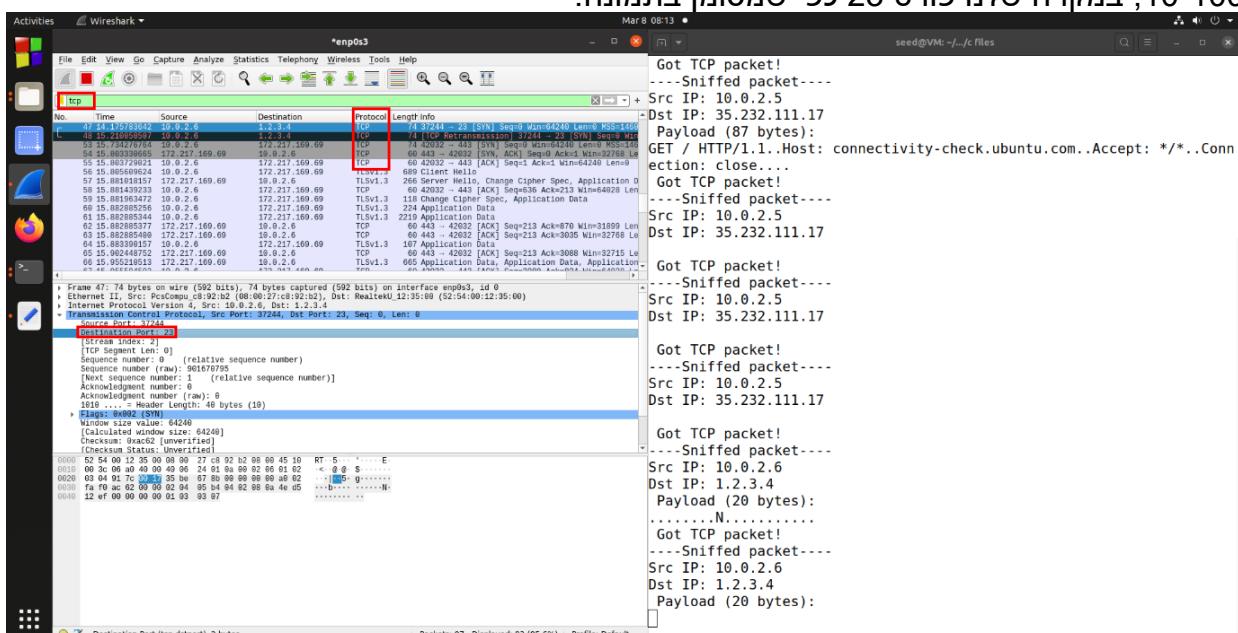
### Filter – ICMP:

.(reply&request) של הנתקף icmp של הנתקף (icmp)



### Filter – TCP:

בתרמונה זו, ניתן לראות שתוכנת ההסנהה מסניפה את פקודות tcp של הנתקף בטוחו של פורטים 10-100, במקרה שלנו פורט 23 כי שמוסמן בתמונה.



## Sniffing Passwords :C2.1

```
Activities Terminal Mar 4 11:20 seed@VM: ~/c/files
Payload (12 bytes):
....CDt...$.
Got TCP packet!
---Sniffed packet---
Src IP: 10.0.2.6
Dst IP: 10.0.2.5
Payload (13 bytes):
....CDun...d.
Got TCP packet!
---Sniffed packet---
Src IP: 10.0.2.6
Dst IP: 10.0.2.5
Payload (13 bytes):
....CDv!..e.
Got TCP packet!
---Sniffed packet---
Src IP: 10.0.2.6
Dst IP: 10.0.2.5
Payload (13 bytes):
....CDw...&L.
Got TCP packet!
---Sniffed packet---
Src IP: 10.0.2.6
Dst IP: 10.0.2.5
Payload (13 bytes):
....cDw...'.
Got TCP packet!
---Sniffed packet---
Src IP: 10.0.2.6
Dst IP: 10.0.2.5
Payload (14 bytes):
....cDx...5..
Got TCP packet!
---Sniffed packet---
Src IP: 10.0.2.6
```

בשאלה זו התוקף רוצה להסניף את סיסמתו של הנתקף, הנתקף מנסה לגשת אל ה – VM על ידי פועלות: telnet 10.0.2.5. לאחר מכן, המשתמש מתבקש להכניס את סיסמתו לצורך התחברות. התוקף מסניף את פקודות TCP העוברות בתעבורה האינטרנטית ומגלה את סיסמתו של הנתקף על ידי המידע המסופק מה – data של הפקודות.

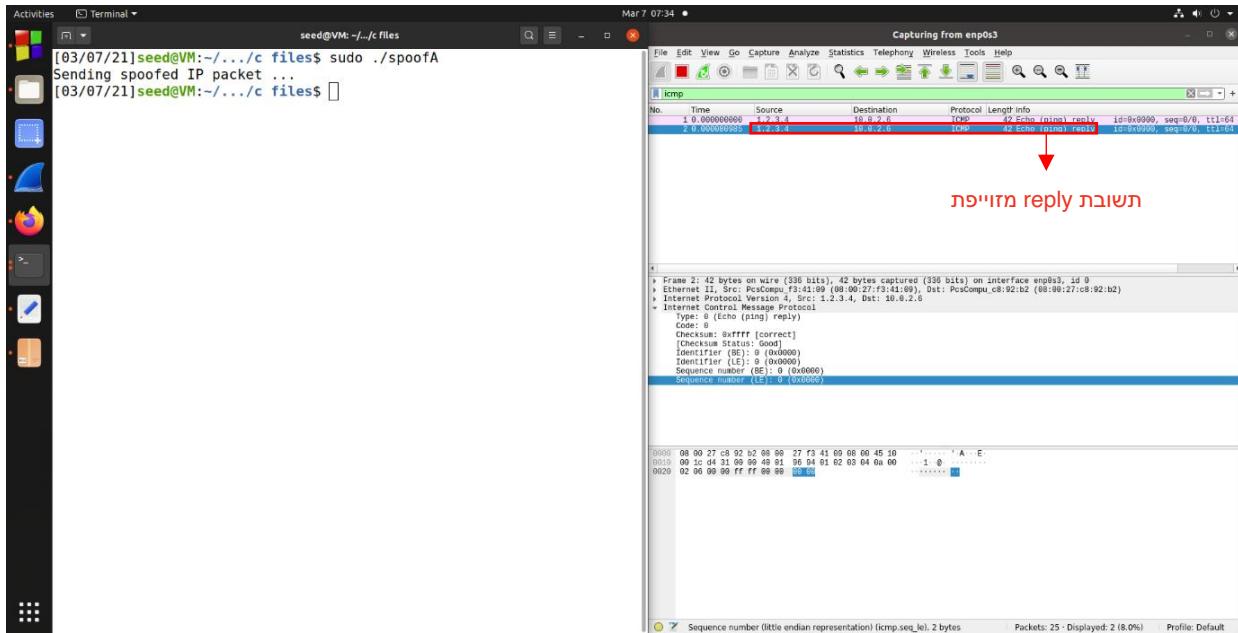
בהתחלתה כל אחד יכול להתבלבל מכל השורות של האותיות והתוויים הרנדומליים, אך לאחר הסתכלות נוספת ניתן למצוא דפוס מסוים אשר בסופו ניתן לגלוות את הסיסמה של הנתקף.

הדפוס הוא רצף של מחוזות שבسوفן נמצאת אות, כאשר נקרא אותה ברצף מלמעלה למטה, תגליה הסיסמה של המשתמש.

בתמונה המוצגת לעיל, הסיסמה של המשתמש היא "dees".

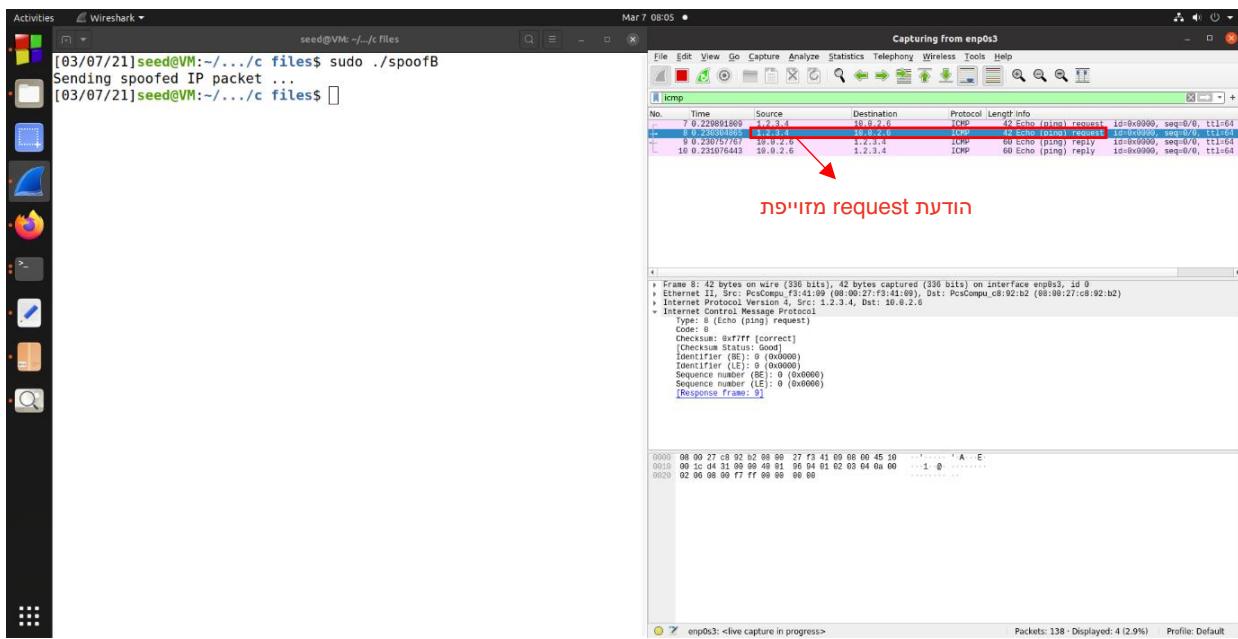
## Write a spoofing program :A2.2

בשאלה הבאה הຕבקשו לשלוח תשובת reply מזויפת המכובה "1.2.3.4" אל כתובות הנטקף כפי שניתן לראות בתמונה שלפנינו.



## Spoof an ICMP Echo Request :B2.2

בשאלה זו לעומת זאת, שלחנו מה-VM של התוקף הודעת ping request המכובה שאינה אמיתית ("1.2.3.4"). כפי שניתן לראות בתמונה שלפנינו, לאחר שליחת ה-ping request המחשב של ה-VM הנטקף ענה בחיווב להודעה המזויפת.



• Question 4:

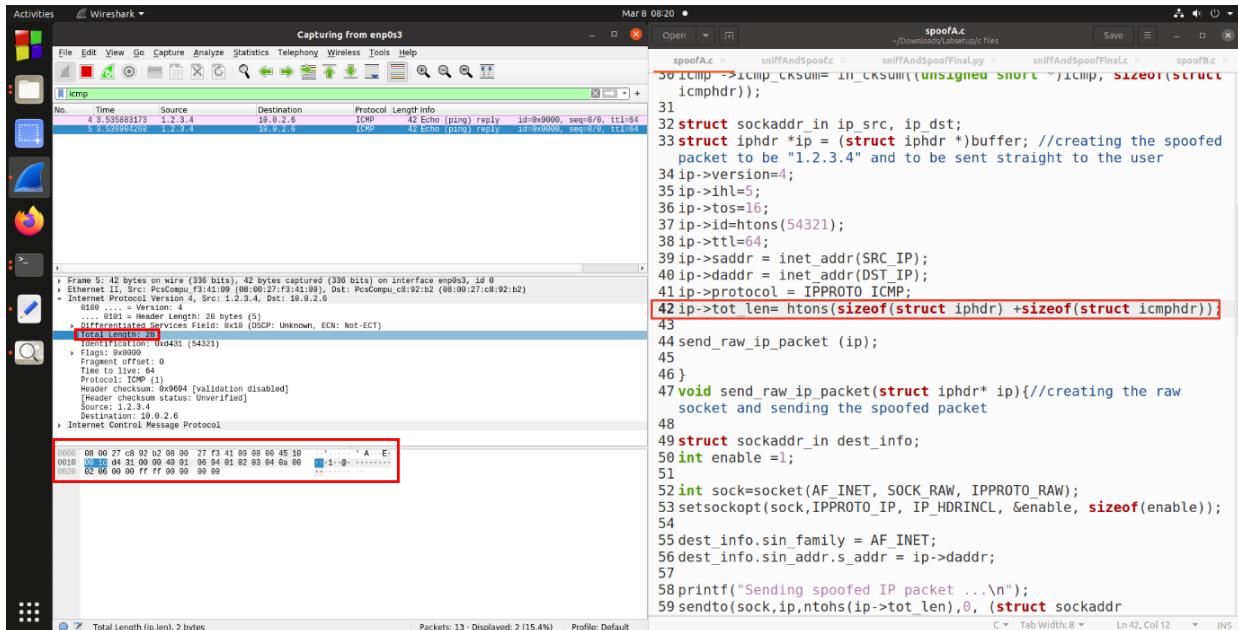
Can you set the IP packet length field to an arbitrary value, regardless of how big the actual packet is?

תשובה:

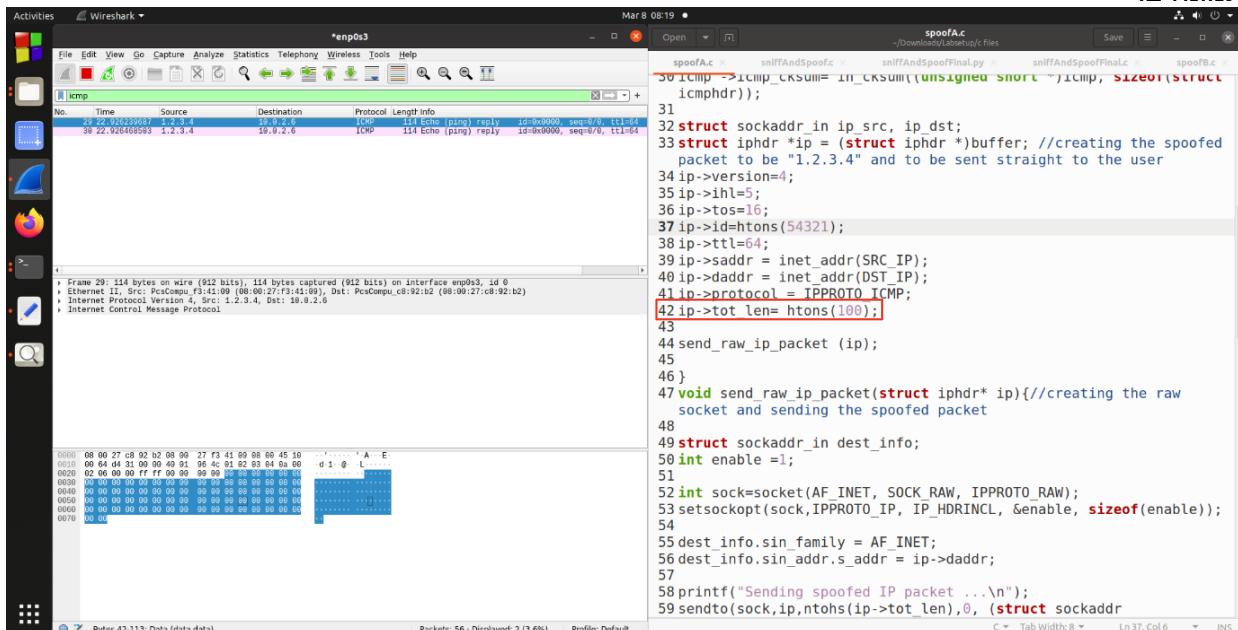
כפי שניתן לראות בתמונה 1, זהה פקטה שגודל אורך ה-IP שלה לא שונה. אורך ה-IP הינו 28 וה-data של סומן גם כן. לעומת זאת, בתמונה 2, שינו את אורך ה-IP להיות בגודל 100, וכפי שניתן לראות לראות מודפס כרגע והדבר היחיד שנוסף ל-data הוא שורות של "0" עד הגודל 100.

לסיכום, שינוי אורך ה-IP אינו משנה דבר.

תמונה 1:



תמונה 2:



• **Question 5:**

Using the raw socket programming, do you have to calculate the checksum for the IP header?

תשובה:

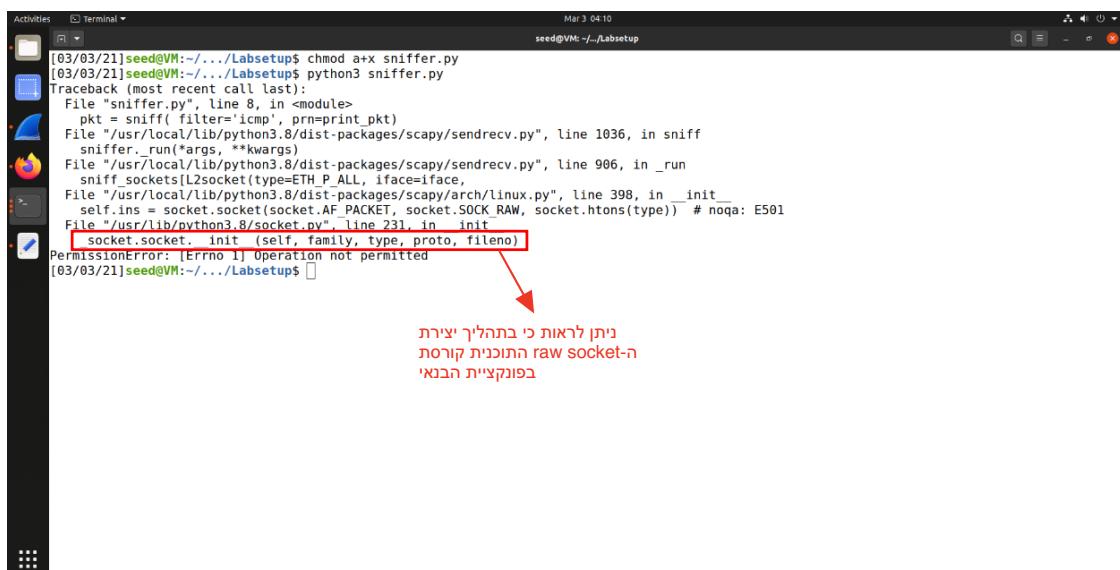
לא חייב לחשב את ה-checksum של ה-**header** IP מאחר והוא מחושב באופן אוטומטי על ידי מערכת הפעלה לעומת ה-**header** ICMP אשר מצריך מייתנו חישוב MADIK של ה-**header**, אחרת, הפקטה לא תצליח להישלח.

• **Question 6:**

Why do you need the root privilege to run the programs that use raw sockets? Where does the program fail if executed without the root privilege?

תשובה:

אנו צריכים גישת ניהול בכדי להריץ תוכנות המשתמש ב-raw sockets משום שפעולה זו מוגנת מראש על ידי הרשות ניהול כפי שהסבירנו ב-[Question 2](#). פעולה זו מוגנת באמצעות הרשות ניהול בכדי לספק הגנה מגורמים עוינים המ竊取 מהמחשב ולרשעת הפרטיה של המשתמש, במטרה למנוע מהם לבצע שינויים לא רצויים.  
כאשר ננסה להריץ את התוכנה ללא הרשות ניהול, היא תיפול בעת פעולה ייצרת ה-raw socket, כפי שניתן לראות בתמונה שלפנינו.



The screenshot shows a terminal window with the following command and output:

```
[03/03/21]seed@VM:~/.../Labsetup$ chmod a+x sniffer.py
[03/03/21]seed@VM:~/.../Labsetup$ python3 sniffer.py
Traceback (most recent call last):
  File "sniffer.py", line 8, in <module>
    pkt = sniff(filter='icmp', prn=print_pkt)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 1036, in sniff
    sniff._run(*args, **kwargs)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 906, in _run
    sniff_sockets[L2socket(type=ETH_P_ALL, iface=iface,
  File "/usr/local/lib/python3.8/dist-packages/scapy/arch/linux.py", line 398, in __init__
    self.ins = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.htons(type)) # noqa: E501
  File "/usr/lib/python3.8/socket.py", line 231, in __init__
    socket.socket.__init__(self, family, type, proto, fileno)
PermissionError: [Errno 1] Operation not permitted
[03/03/21]seed@VM:~/.../Labsetup$
```

A red arrow points from the text "בונקציית הבנאי" to the line of code where the socket is initialized, specifically to the argument `proto`.

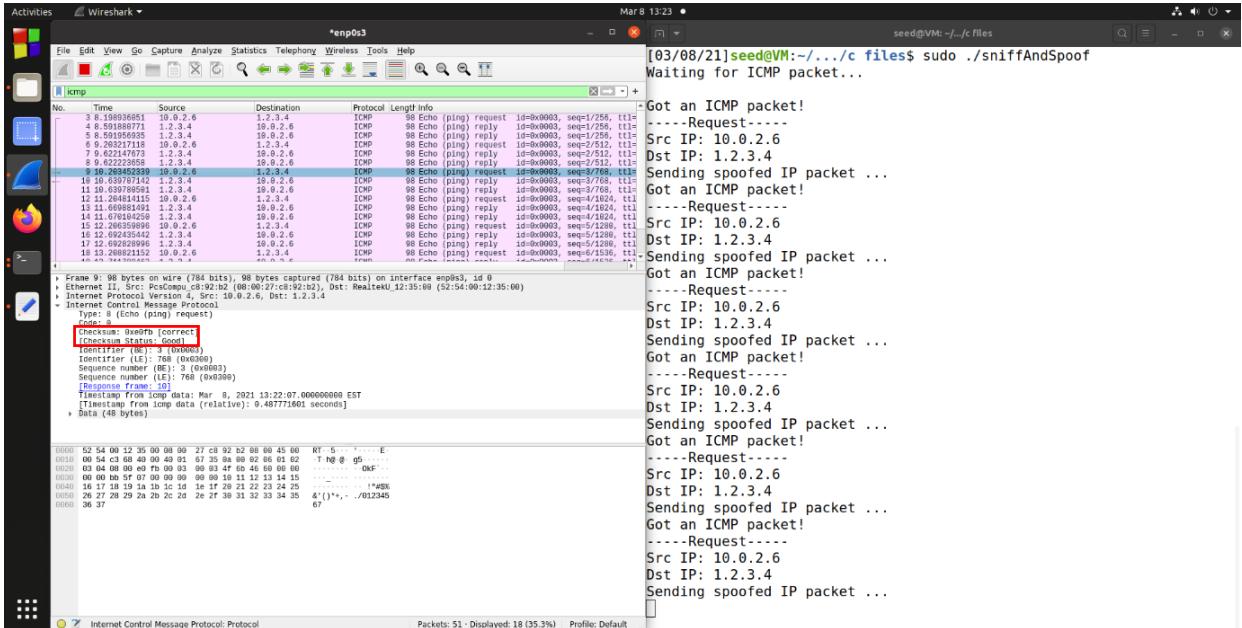
בונקציית הבנאי  
בונקציית הבנאי  
בונקציית הבנאי

## Sniff and then Spoof :2.3

בשאלה זו, התוקף שלנו מצליך להסניף את הכתובת המבוクשת (של הנתקף) בזמן אמת. הפעם, לא רק שהትוקף הצליח להסניף את הכתובת, הוא גם למשהו הצליח להתחזות לכתובת המבוクשת ולשלוח ממנה הודעה חוזרת (reply). ב כדי לראות שakan נשלחה פקעת reply נcona, כלומר מזויף אך תקין, אפשר לשים לב שה-sh- checksum גDEL ב-ping- reply .0x0800 ב-ה-request.

במידה וה-checksum לא היה גדול בצורה זו, לפקעת request לא היה חזר מענה והתוכנית לא הייתה משיגה את התוצאה המבוクשת.

תמונה 1:



תמונה 2:

