

# SEED Labs – Packet Sniffing and Spoofing Lab

## Task 1.1A.

## Solution

נתקשנו להריץ את ה sniffer ללא הרשות מנהל, וזה נדחה!

**Run the sniffer with "sudo" :**

```
[12/15/18]seed@VM:~/Documents$ sudo python sniffer.py
[sudo] password for seed:
###[ Ethernet ]##
dst      = 52:54:00:12:35:02
src      = 08:00:27:28:a2:f4
type     = 0x800
###[ IP ]##
version  = 4
ihl      = 5
tos      = 0x0
len      = 84
id       = 55872
flags    = DF
frag     = 0
ttl      = 64
proto    = icmp
chksum   = 0x919c
src      = 10.0.2.15
dst      = 172.217.21.228
\options \
###[ ICMP ]##
type     = echo-request
code     = 0
chksum   = 0x39a8

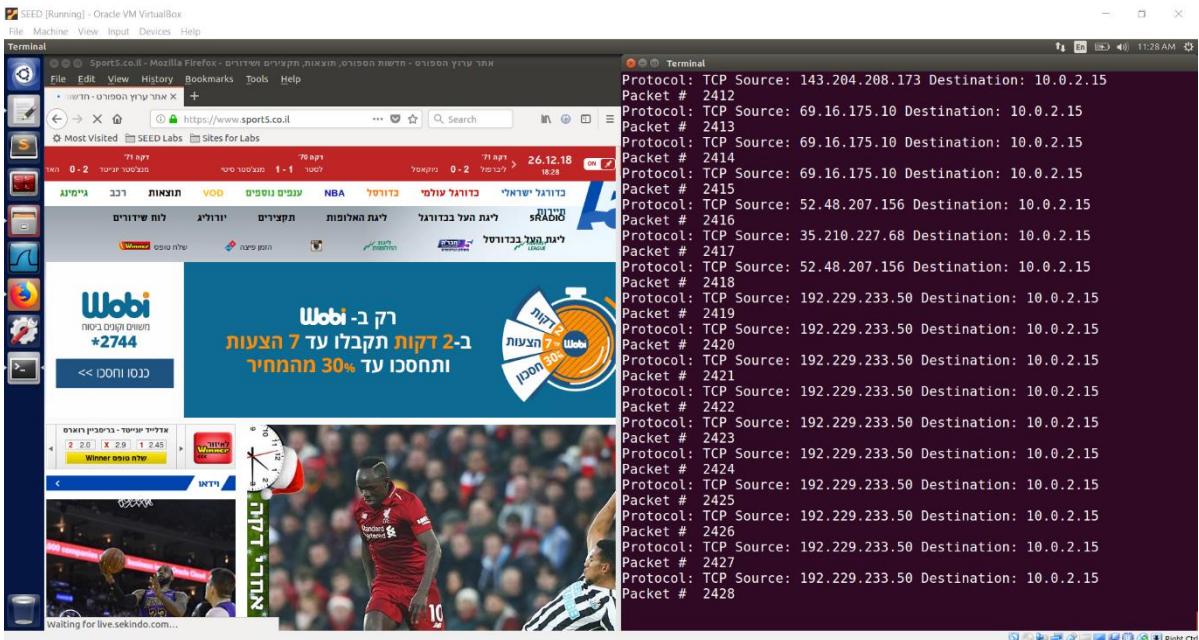
$ ping www.sport5.co.il
82.166.109.234) 56(84) bytes of data.
```

## **ICMP packets:**

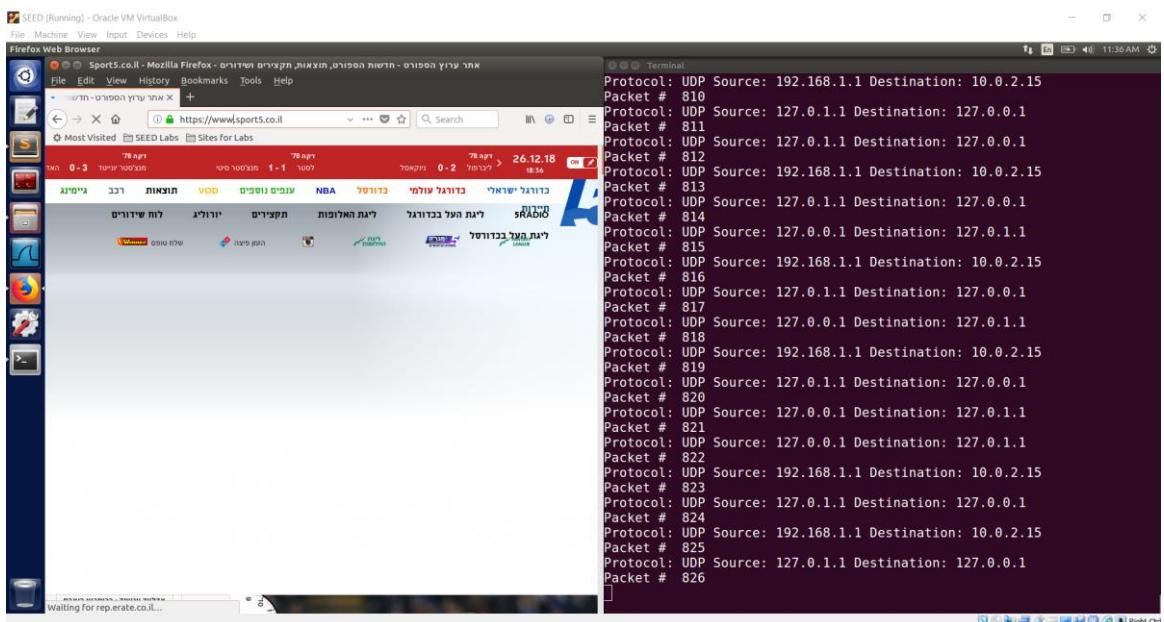
## Task 1.1B.

### TCP packets with specific destination:

הוספנו גם מונה של הפקות כדי לראות את גודל התעבורה של שכבת התעבורה. השתמשנו ב **sniffer** קצת יותר מתחכם שמאפשר לבקש בשורת פקודה איזה סוג של תעבורה לנטר.



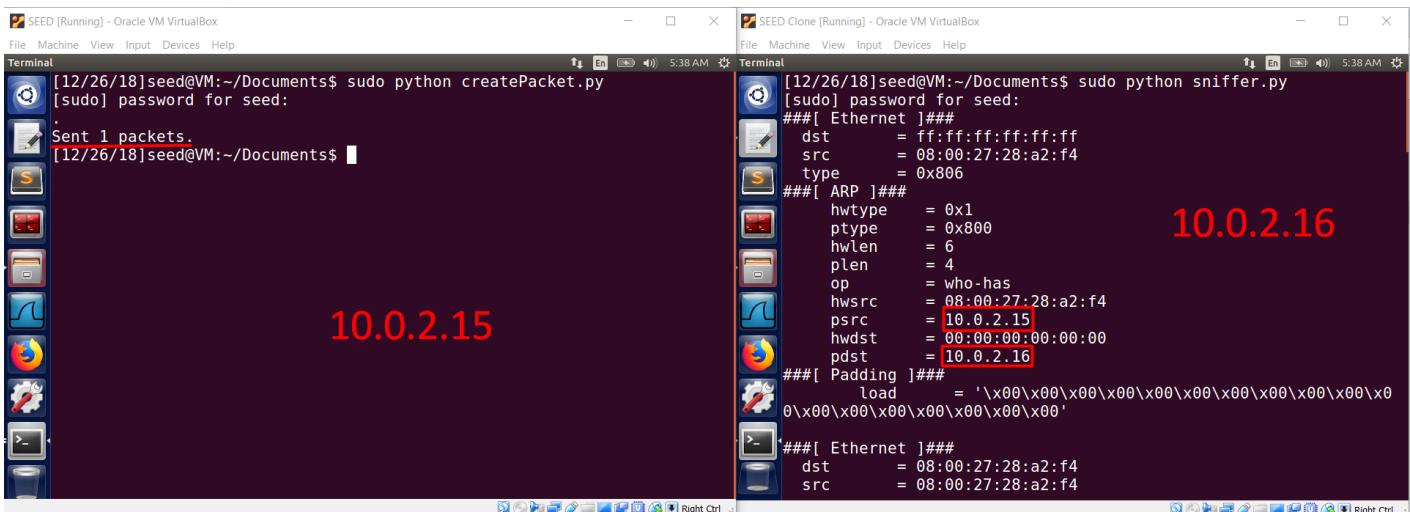
### UDP packets with specific destination:



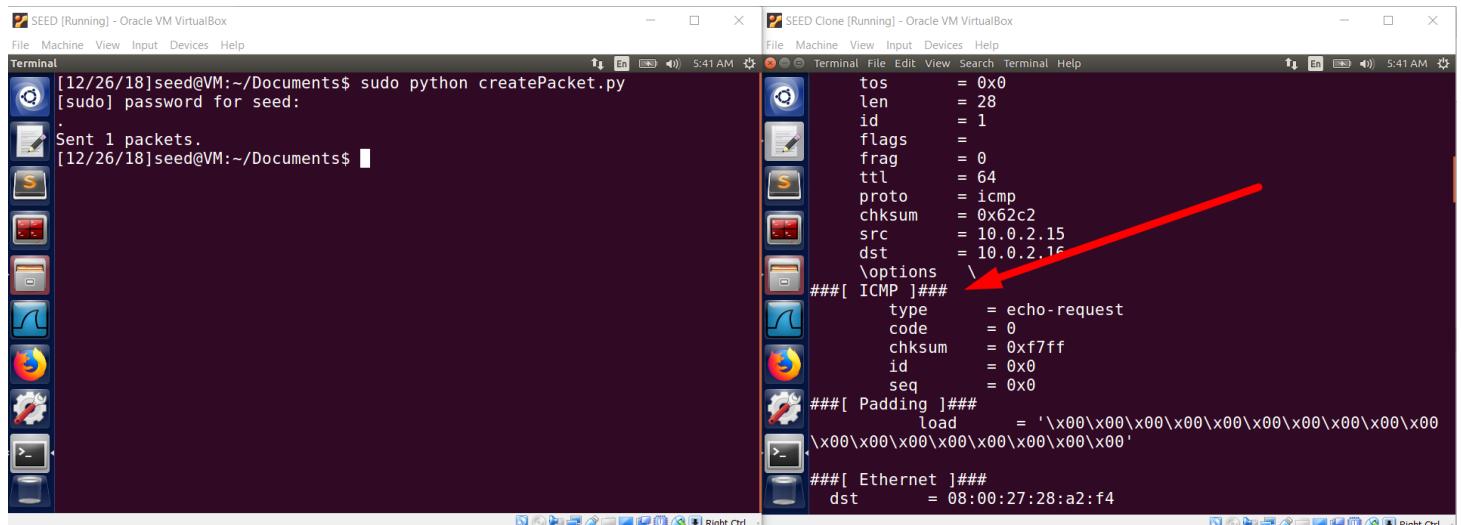
## Task 1.2

כדי לעשות את זה, במצבה א' (10.0.2.16) פתחנו את ה sniffer מהשאלה הקודמת, רק ביקשנו שיאזין הפעם למצבה ב' (10.0.2.15). ובמצבה ב' הרצינו את הקוד שיוצר את הפקטה, ושלחנו אותה אל מצבה א'.

**ניתן לראות את הממצאים:**



אפשר לראות שהפקטה עם השכבה של ICMP הגיעה ליעדה.



### Task 1.3

היעד שבחרנו להגעה אליו הוא 8.8.8.8 , אז הפעילו במחשב האישי את הפקודה tracert CD' לדעת איזה נתבים אנחנו עורבים בדרך ולאמת את זה:

```
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\eladn>tracert 8.8.8.8

Tracing route to google-public-dns-a.google.com [8.8.8.8]
over a maximum of 30 hops:

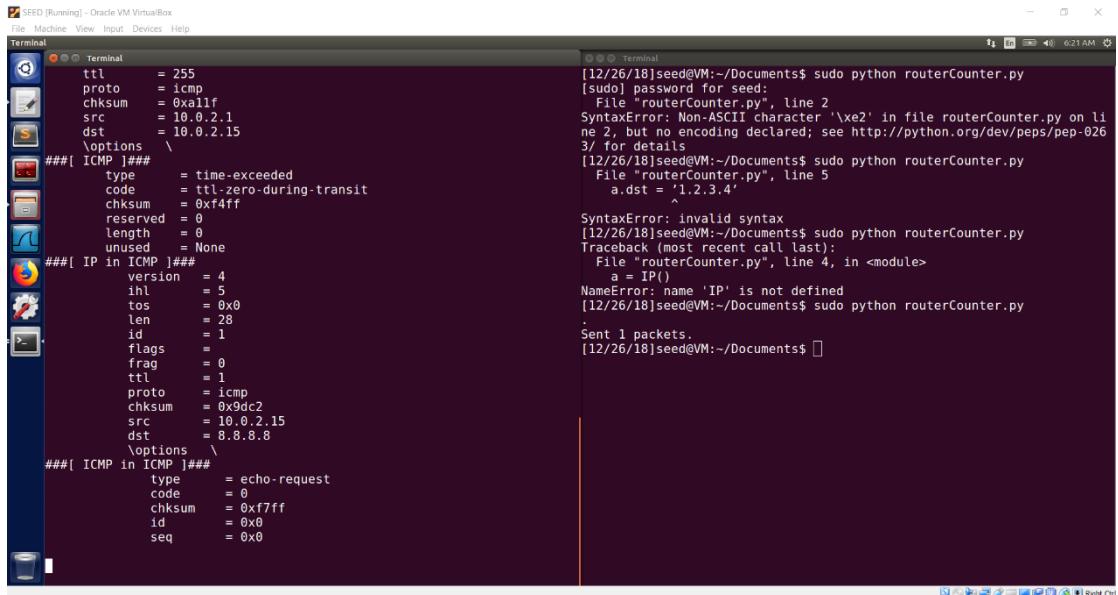
 1   2 ms    2 ms    3 ms  Broadcom.Home [192.168.1.1]
 2   16 ms   13 ms   10 ms  82.102.128.200
 3   *        *        *      Request timed out.
 4   12 ms   10 ms   13 ms  82.102.132.78
 5   64 ms   65 ms   84 ms  80.179.166.134.static.012.net.il [80.179.166.134]
 6   65 ms   65 ms   72 ms  72.14.216.121
 7   83 ms   83 ms   69 ms  108.170.252.1
 8   67 ms   81 ms   70 ms  108.170.228.239
 9   63 ms   64 ms   63 ms  google-public-dns-a.google.com [8.8.8.8]

Trace complete.

C:\Users\eladn>
```

הפעילנו באותו המקרה את ה host sniffer על ה host ככה שיקלוט כל פקטה שייצאת ונכנסת. ובטרמינל נוסף הפעילו את הקוד שיזכר פקודות לסריקת הנתבים שבדרך. וכל פעם הוספנו באופן ידני TTL 1

first round



## second round

```

SEED [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
###[ Ethernet ]###
dst      = 08:00:27:28:a2:f4
src      = 52:54:00:12:35:00
type     = 0x800
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0x30
len      = 56
id       = 663
flags    =
frag    = 0
ttl     = 63
proto   = icmp
chksum  = 0xab46
src     = 192.168.1.1
dst     = 10.0.2.15
\options \
###[ ICMP ]###
type    = time-exceeded
code   = ttl-zero-during-transit
checksum = 0xf4ff
reserved = 0
length  = 0
unused   = None
###[ IP in ICMP ]###
version  = 4
ihl      = 5
tos      = 0x0
len      = 28
id       = 1
flags    =
frag    = 0
ttl     = 2
proto   = icmp
###[ Ethernet ]###
dst      = 08:00:27:28:a2:f4
src      = 52:54:00:12:35:00
type     = 0x800
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0x20
len      = 56
id       = 664
flags    =
frag    = 0
ttl     = 253
proto   = icmp
checksum = 0xdcfc
src     = 82.102.128.200
dst     = 10.0.2.15
\options \
###[ ICMP ]###
type    = time-exceeded
code   = ttl-zero-during-transit
checksum = 0xf4ff
reserved = 0
length  = 0
unused   = None
###[ IP in ICMP ]###
version  = 4
ihl      = 5
tos      = 0x0
len      = 28
id       = 1

```

```

[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
[sudo] password for seed:
  File "routerCounter.py", line 2
SyntaxError: Non-ASCII character '\xe2' in file routerCounter.py on line 2, but no encoding declared; see http://python.org/dev/peps/pep-0263/ for details
[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
  File "routerCounter.py", line 5
    a.dst = '1.2.3.4'
               ^
SyntaxError: invalid syntax
[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
Traceback (most recent call last):
  File "routerCounter.py", line 4, in <module>
    a = IP()
NameError: name 'IP' is not defined
[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
Sent 1 packets.
[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
Sent 1 packets.
[12/26/18]seed@VM:~/Documents$ 

```

## Third round

```

SEED [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
###[ Ethernet ]###
dst      = 08:00:27:28:a2:f4
src      = 52:54:00:12:35:00
type     = 0x800
###[ IP ]###
version  = 4
ihl      = 5
tos      = 0x20
len      = 56
id       = 664
flags    =
frag    = 0
ttl     = 253
proto   = icmp
checksum = 0xdcfc
src     = 82.102.128.200
dst     = 10.0.2.15
\options \
###[ ICMP ]###
type    = time-exceeded
code   = ttl-zero-during-transit
checksum = 0xf4ff
reserved = 0
length  = 0
unused   = None
###[ IP in ICMP ]###
version  = 4
ihl      = 5
tos      = 0x0
len      = 28
id       = 1

```

```

[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
[sudo] password for seed:
  File "routerCounter.py", line 2
SyntaxError: Non-ASCII character '\xe2' in file routerCounter.py on line 2, but no encoding declared; see http://python.org/dev/peps/pep-0263/ for details
[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
  File "routerCounter.py", line 5
    a.dst = '1.2.3.4'
               ^
SyntaxError: invalid syntax
[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
Traceback (most recent call last):
  File "routerCounter.py", line 4, in <module>
    a = IP()
NameError: name 'IP' is not defined
[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
.
Sent 1 packets.
[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
.
Sent 1 packets.
[12/26/18]seed@VM:~/Documents$ sudo python routerCounter.py
.
Sent 1 packets.
[12/26/18]seed@VM:~/Documents$ 

```

## Fourth round:

עשינו את זה 5 ttl בגל שהיה timeout request ב 4.

### Fifth round:

**במה שבעמודים הבאים, ממוספרים**

7

8

כאן ניתן לראות כי הפקטה הגיעה אל יעדה, ונשלחת הודעה אל המקור שמננו נשלחה הפקטה..

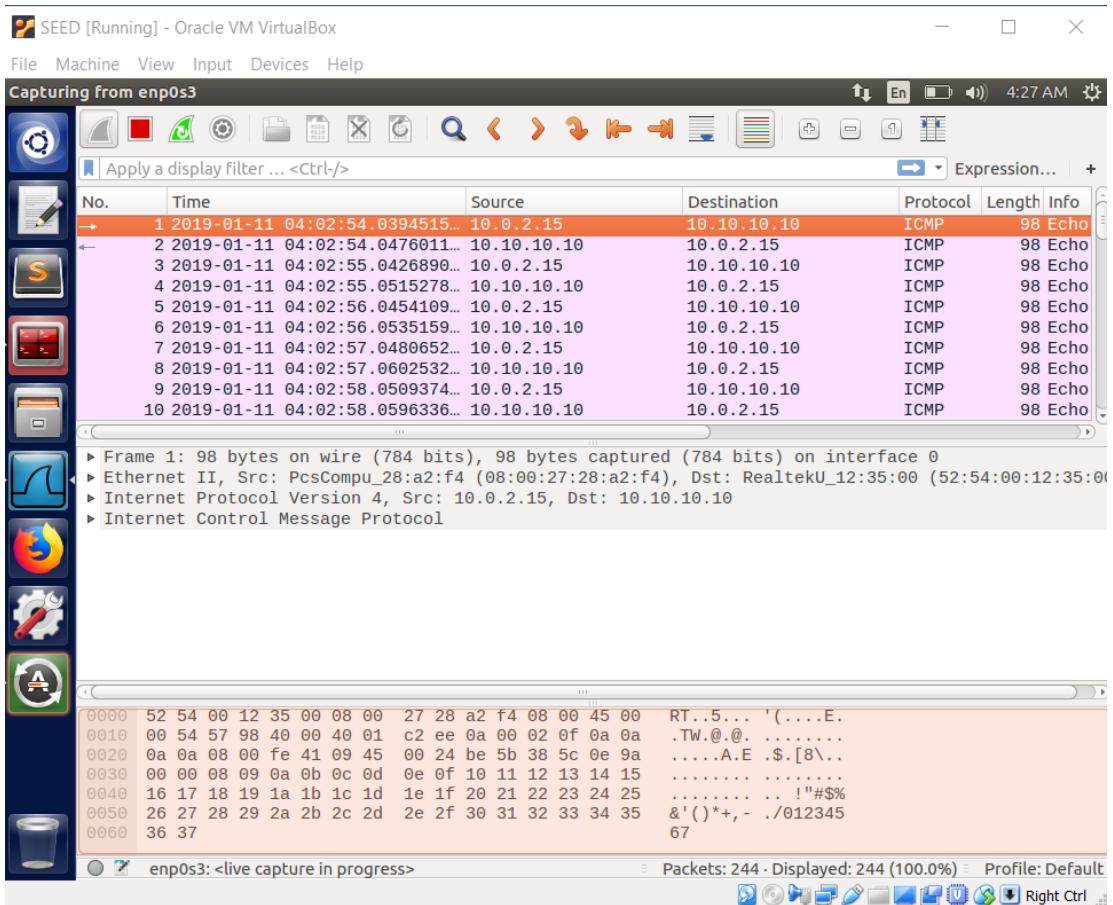
## Task 1.4

כדי לבער משימה כזאת, علينا להפעיל שתי מכונות. לוודא שיש להן שתי כתובות קיימות.

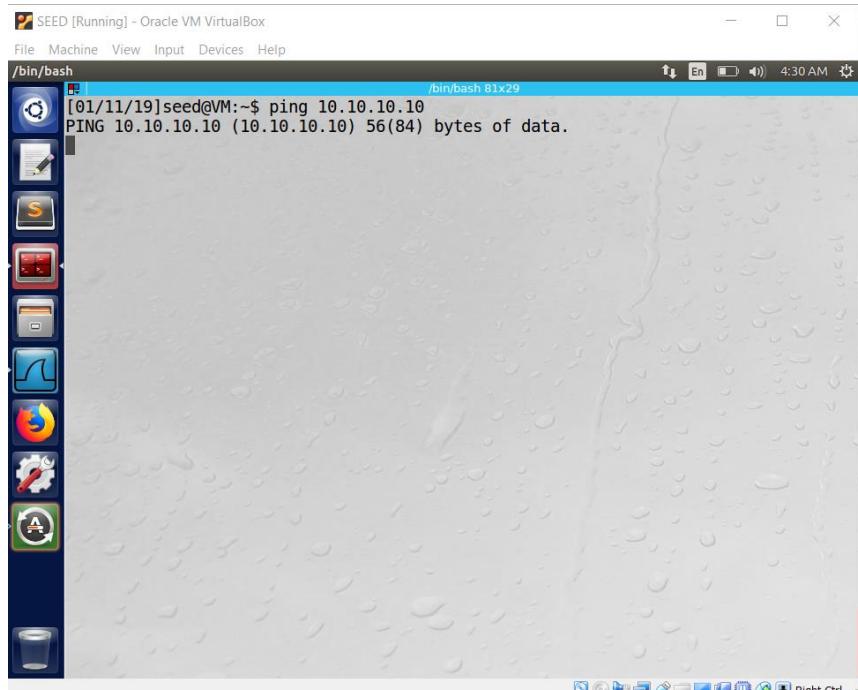
המבחן: הפעלו במכונה א' (10.0.2.4) את ה-sniffer and snooper שלנו כדי שינטר את התעבורה ברשת שלנו. במכונה ב' (10.0.2.15) עשינו ping ל-10.10.10.10 כתובת שאינה קיימת, וניתן לראותות שה-sniffer and snooper מזהה שנשלחות ברשת שלו בקשות ping ומוחזר מיד תשובה pong אל

The image shows two side-by-side terminal windows from Oracle VM VirtualBox. The left window is titled 'Terminal' and has the IP address '10.0.2.4' displayed prominently at the top. It contains a series of 'ping' command outputs to the IP 10.0.10.10, showing ICMP sequence numbers 1 through 26 with various times and byte counts. The right window is titled 'bin/bash' and has the IP address '10.0.2.15' displayed prominently at the top. It also contains a series of 'ping' command outputs to the same IP, showing ICMP sequence numbers 1 through 26 with similar times and byte counts. Both windows show a standard Linux desktop environment with icons for file operations, terminal, and system monitoring.

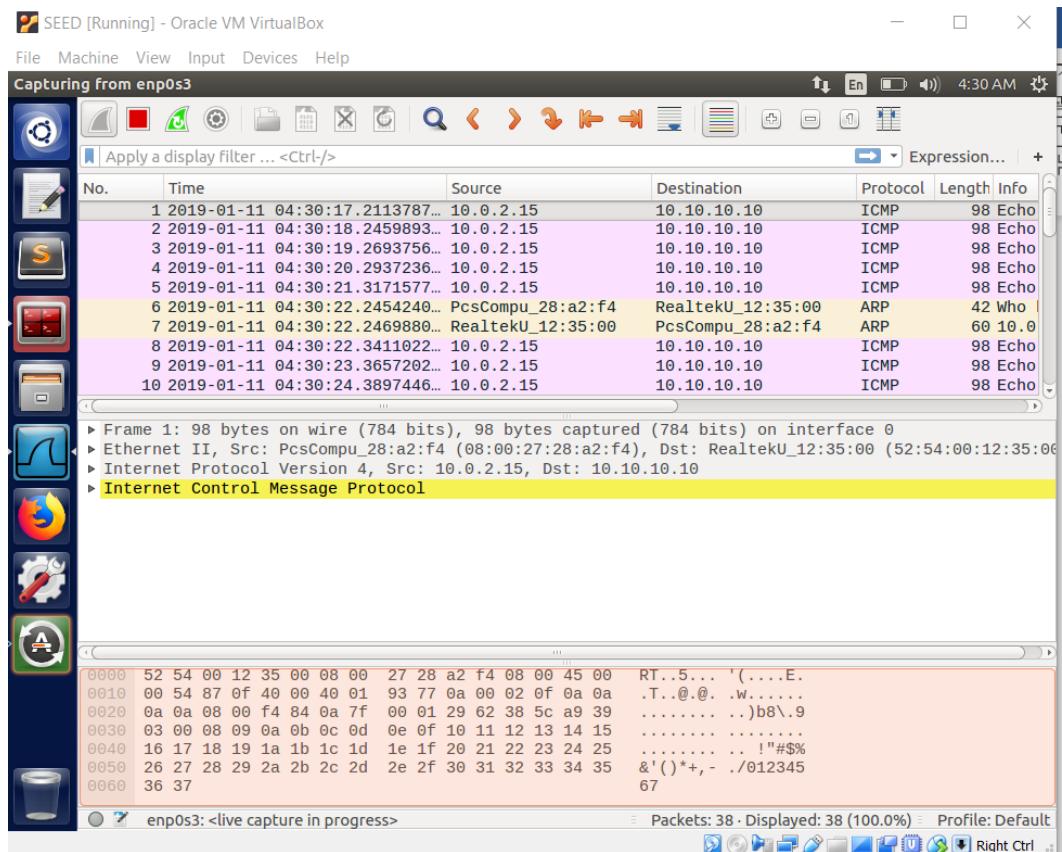
ניתן גם להבחן בזה בתעבורת רשת של wireshark.



לשם הבדיקה, בלי הפעלה של sniffers and Spoffer ניתן לראות ש ping לא פועל

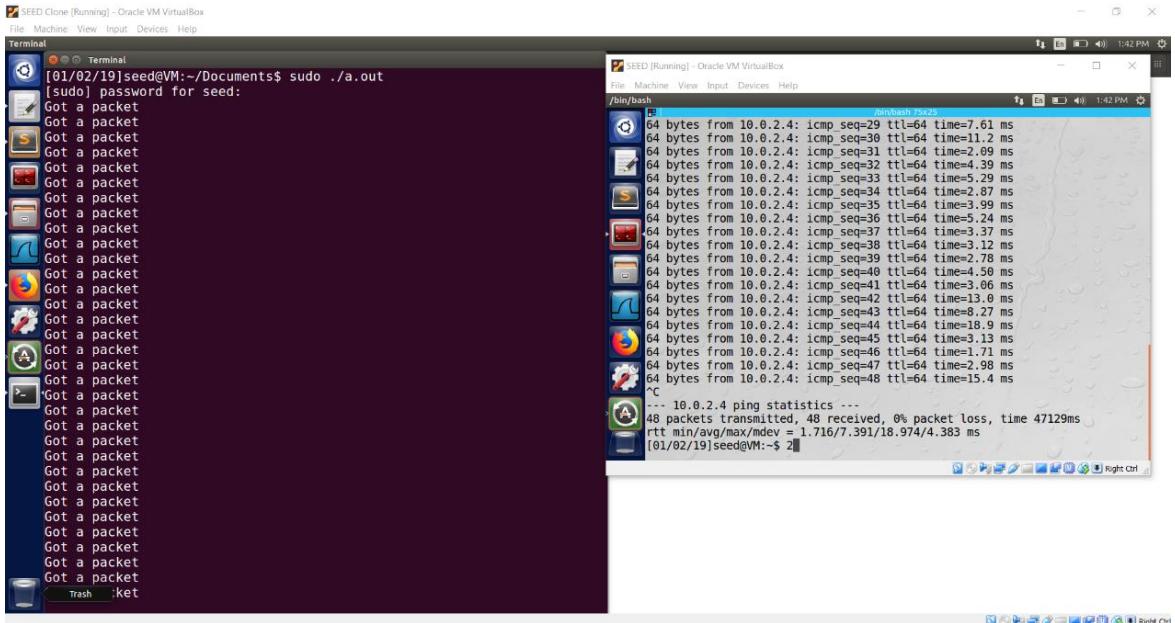


וגם ה **TCP** שום פונק מהכתובת המבוקשת:



## Task 2.1A

הרצינו את הקוד שנמצא במללה, ניתן לראות כי הפקות נתפסות. הרצינו עוד קוד נוסף בהמשך על מנת לראות את ה *sor ip*



## שאלה 1

מה שהקוד הראשון שראינו למעלה עשו הוא, פותח עורך של סשין עם icmp עם הפרוטוקול אינטרנט לאחר מכן מגדיר את הפילטר שלו פיו הוא יחוליט לסן. ולבסוף הוא עשו לו לאה לתפיסת הפאקאות עד אשר עוזרים אותו, ובמידה נתפסת פאקטה הוא ידפיס "נתפס".

## שאלה 2

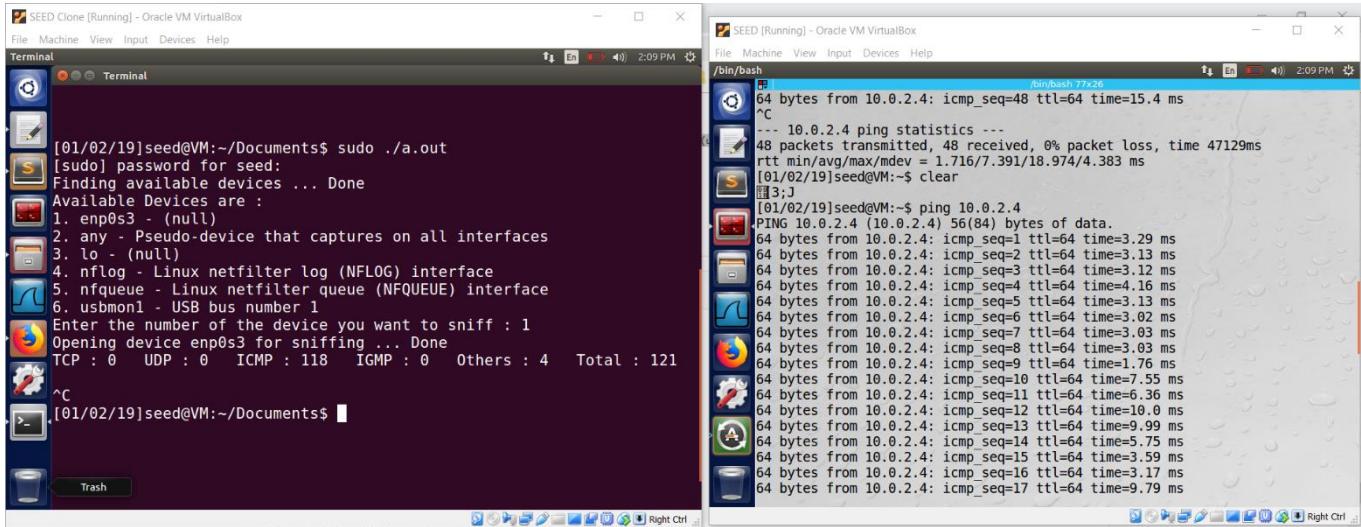
התוכנה לא מצלילה להריץ את הקובץ אחרי הקומpileציה של gcc . ובאמת רק בהרשאת מנהל התוכנה sniffer ירוץ.

### שאלה 3

בינטרנט נשלחים הרבה פקודות, ברגע שהוא דלוק זה אפשר לקבל כל מיני פקודות שאולן לא היו מתקנות, כי אין לא ממש אמורויות להישלח אלינו. בנויגוד לזה שהמנגן זהה הוא כבוי!

## Task 2.1B

### פקודות icmp בין שתי מכונות וירטואליות ספציפיות:



```
*****ICMP Packet*****
Ethernet Header
|-Destination Address : 08-00-27-5B-23-84
|-Source Address      : 08-00-27-28-A2-F4
|-Protocol           : 8

IP Header
|-IP Version         : 4
|-IP Header Length   : 5 DWORDS or 20 Bytes
|-Type Of Service    : 0
|-IP Total Length    : 84 Bytes(Size of Packet)
|-Identification     : 49015
|-TTL                : 64
|-Protocol          : 1
|-Checksum           : 25375
|-Source IP          : 10.0.2.15
|-Destination IP     : 10.0.2.4

ICMP Header
|-Type : 8 |-Code : 0
|-Checksum : 11893

IP Header
08 00 27 5B 23 84 08 00 27 28 A2 F4 08 00 45 00 ... '#...'(....E.
00 54 BF 77 .T.w
UDP Header
40 00 40 01 @.@

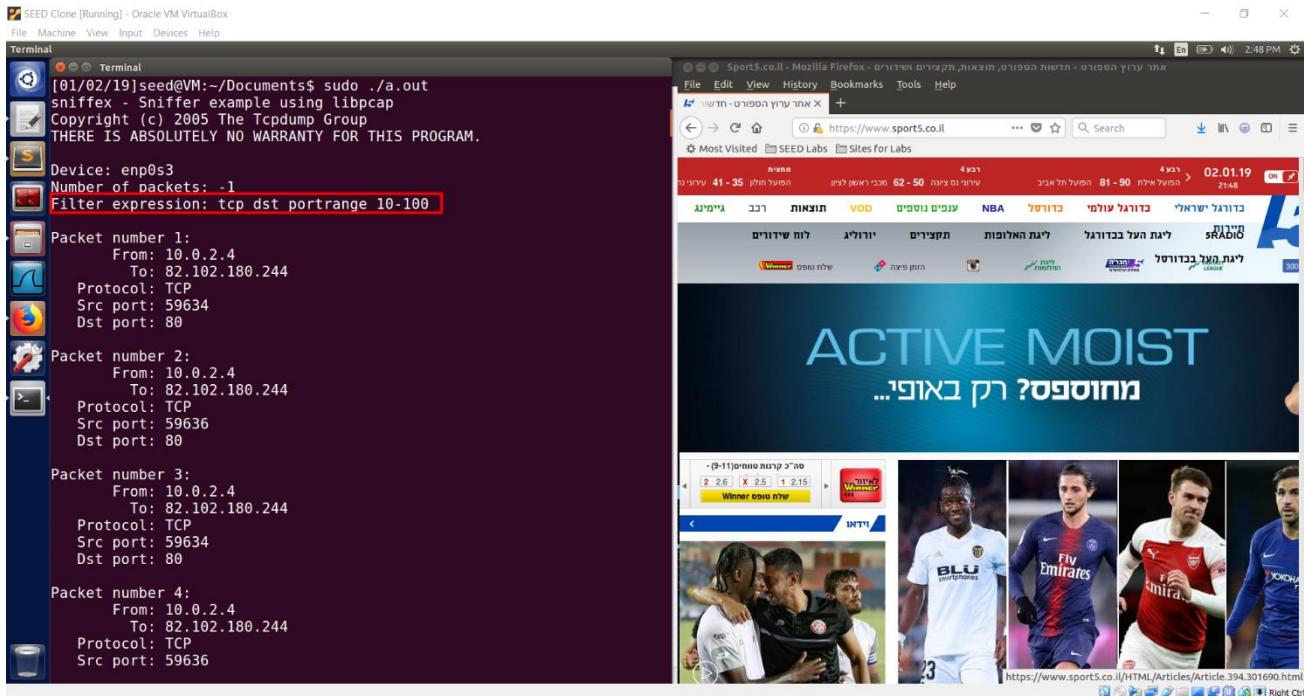
Data Payload
OE E6 00 01 71 0B 2D 5C 23 39 0E 00 08 09 0A 0B ....q.-\#9.....
0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B ..... !#$%&'()*+
1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B ,-.01234567
2C 2D 2E 2F 30 31 32 33 34 35 36 37

#####

```

בקובץ טקסט שנוצר אחרי הפעלת ה  
sniffer ניתן לראות במודיק את כל  
המידע על כל פקטה:

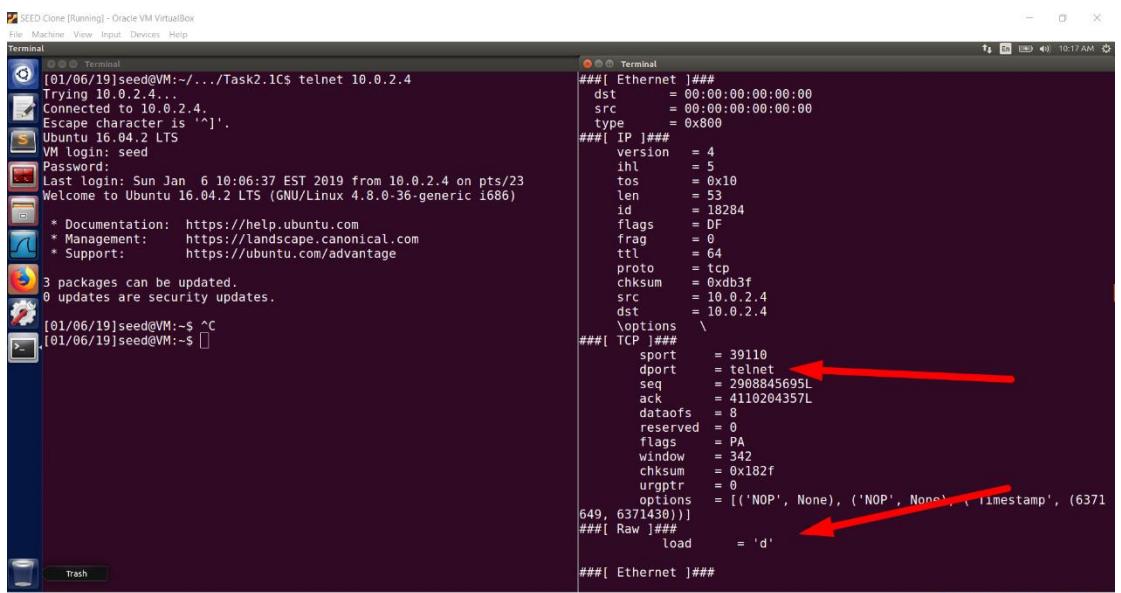
### פקטוות עם טווח מסויים של פורטים:



### Task 2.1C

ניתן כמובן לראות שהפקטוות שנתפסו הן בפrotocol tcp כמו שנתבקשנו!

כדי לבצע את המשימה הזאת, הרצינו את ה telnet בטרמינל אחד, ובשני הרצינו את ה sniffer ככה שהוא מס肯 את port 23 שזה הרשות הפנימית. ונitinן לראות עד כמה הprotokol פרוץ ובפקטוות שנתפסו, נמצאה הסיסמה 'dees' בחלקם של כמה פקטוות!



הנה האות הבאה:

```
[01/06/19]seed@VM:~/.../Task2.1C$ telnet 10.0.2.4
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Sun Jan  6 10:06:37 EST 2019 from 10.0.2.4 on pts/23
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

3 packages can be updated.
0 updates are security updates.

[01/06/19]seed@VM:~$ ^C
[01/06/19]seed@VM:~$ 
```

```
Terminal
ihl      = 5
tos     = 0x10
len      = 53
id      = 18285
flags   = DF
frag    = 0
ttl     = 64
proto   = tcp
cksum   = 0xdb3e
src     = 10.0.2.4
dst     = 10.0.2.4
\options \
###[ TCP ]###
sport    = 39110
dport    = telnet
seq     = 2908845696L
ack     = 4110204357L
dataofs = 8
reserved = 0
flags   = PA
window  = 342
cksum   = 0x182f
urgptr  = 0
options = [ ('NOP', None), ('NOP', None), ('Timestamp', (63714, 6371660)) ]
###[ Raw ]###
load    = 'e'

###[ Ethernet ]###
dst     = 00:00:00:00:00:00
src     = 00:00:00:00:00:00
type   = 0x800
###[ IP ]###
version = 4
ihl    = 5

File Machine View Input Devices Help
Terminal
10:21 AM
```

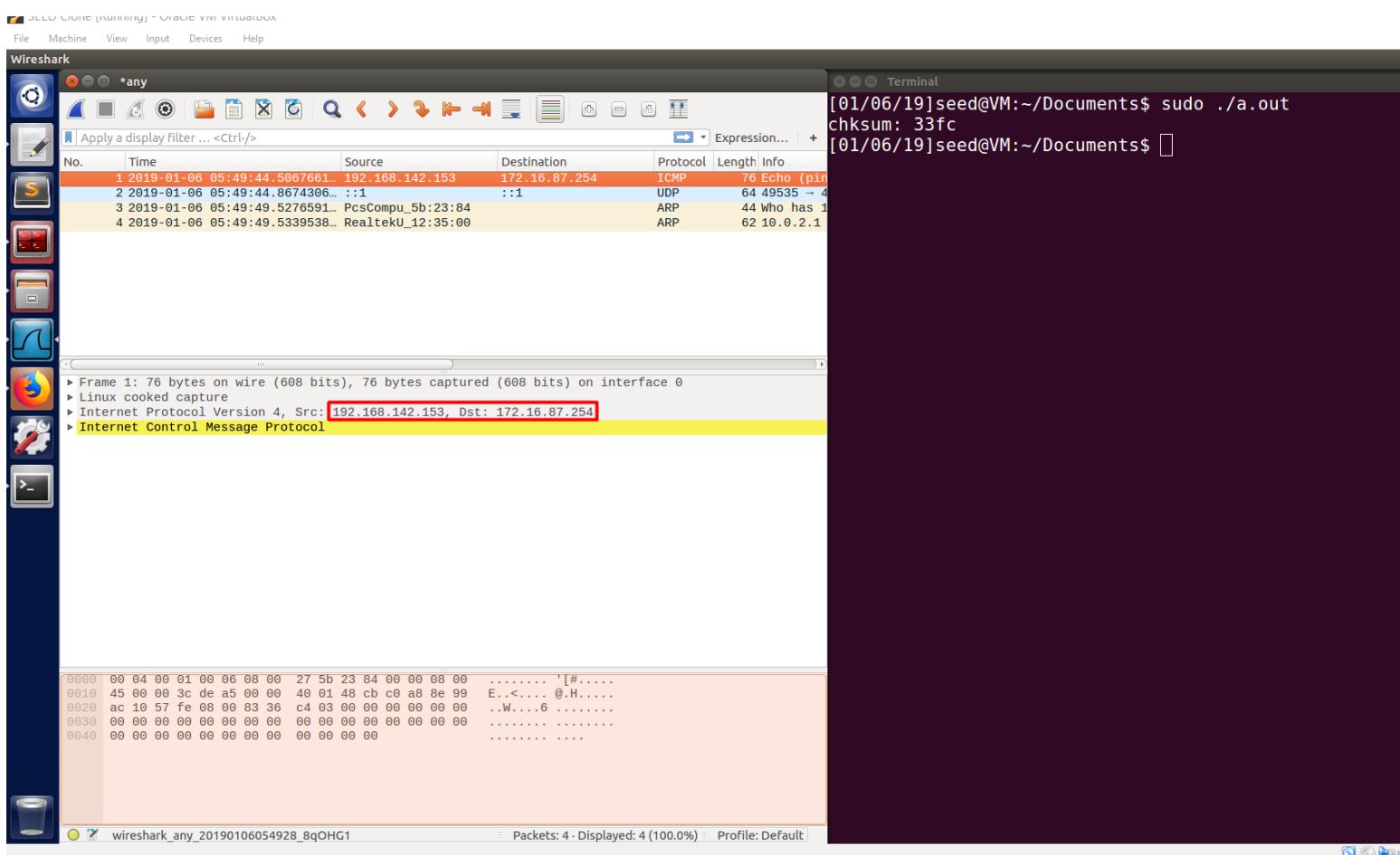
## Task 2.2A

כדי לבצע את המשימה הצעת הפעלנו את בדקנו שה Wireshark באמת רואה את התעבורה של הSocket שעשינו.

הגדכנו את הפקטה שלו בקוד עם מקור של 192.168.142.153 ויעד 172.16.87.254 באופן שריםותי

```
/*Time to live.
 Maximum number of hops that the packet can
 pass while travelling through its destination.*/
ip.ip_ttl = 64;
//Upper layer (Layer III) protocol number:
ip.ip_p = IPPROTO_ICMP;
/*We set the checksum value to zero before passing the packet
 into the checksum function. Note that this checksum is
 calculate over the IP header only. Upper layer protocols
 have their own checksum fields, and must be calculated seperately.*/
ip.ip_sum = 0x0;
/*Source IP address, this might well be any IP address that
 may or may NOT be one of the assigned address to one of our interfaces:*/
ip.ip_src.s_addr = inet_addr("192.168.142.153");
// Destination IP address:
ip.ip_dst.s_addr = inet_addr("172.16.87.254");
/*We pass the IP header and its length into the internet checksum
 function. The function returns us as 16-bit checksum value for
 the header:*/
ip.ip_sum = in_cksum((unsigned short *)&ip, sizeof(ip));
/*We're finished preparing our IP header. Let's copy it into
 the very begining of our packet:*/
memcpy(packet, &ip, sizeof(ip));
```

וניתן לראות שהפקטה נתפסה מהסוקט ששלחנו !



## Task 2.2B

שינויו את הכתובת של היעד בקוד למכונה נוספת שהפעלנו, וניתן לראות כי wireshark מראה בטעורה כי נשלחה הודעה icmp echo request אל הכתובת המדוורת!

No.	Time	Source	Destination	Protocol	Length	Info
1	2019-01-06 06:49:45.3510198...	::1		UDP	64	49535 → 48905 Len=0
2	2019-01-06 06:49:45.6666244...	192.168.142.153	10.0.2.15	ICMP	76	Echo (ping) request id=0xc403, seq=0/0, ttl=64 (no response found!)
3	2019-01-06 06:49:50.5717373...	10.0.2.3	255.255.255.255	DHCP	592	DHCP ACK - Transaction ID 0x72f4a20a
4	2019-01-06 06:49:50.6795708...	PcsCompu_5b:23:84		ARP	44	Who has 10.0.2.15? Tell 10.0.2.4
5	2019-01-06 06:49:50.6875543...	PcsCompu_28:a2:f4		ARP	62	10.0.2.15 is at 08:00:27:28:a2:f4
6	2019-01-06 06:50:04.8907645...	10.0.2.15	10.0.2.4	ICMP	100	Echo (ping) request id=0xc001, seq=1/256, ttl=64 (reply in 7)
7	2019-01-06 06:50:04.8907885...	10.0.2.4	10.0.2.15	ICMP	100	Echo (ping) reply id=0xc001, seq=1/256, ttl=64 (request in 6)
8	2019-01-06 06:50:04.8912729...	::1		UDP	64	49535 → 48905 Len=0
9	2019-01-06 06:50:05.8979822...	10.0.2.15	10.0.2.4	ICMP	100	Echo (ping) request id=0xc001, seq=2/512, ttl=64 (reply in 10)
10	2019-01-06 06:50:05.8980279...	10.0.2.4	10.0.2.15	ICMP	100	Echo (ping) reply id=0xc001, seq=2/512, ttl=64 (request in 9)
11	2019-01-06 06:50:06.9004999...	10.0.2.15	10.0.2.4	ICMP	100	Echo (ping) request id=0xc001, seq=3/768, ttl=64 (reply in 12)
12	2019-01-06 06:50:06.9005464...	10.0.2.4	10.0.2.15	ICMP	100	Echo (ping) reply id=0xc001, seq=3/768, ttl=64 (request in 11)
13	2019-01-06 06:50:07.9052489...	10.0.2.15	10.0.2.4	ICMP	100	Echo (ping) request id=0xc001, seq=4/1024, ttl=64 (reply in 14)
14	2019-01-06 06:50:07.9056524...	10.0.2.4	10.0.2.15	ICMP	100	Echo (ping) reply id=0xc001, seq=4/1024, ttl=64 (request in 13)

תשובות לשאלות:

4. אי אפשר לשנות את האורך של ה packetipo משום שהוא צריך להיות האורך האמיתי, אחרת זה ישלח אחריו זה הודעה שגיאיה!

5. אצלנו לא בקוד אין יימוש לחישוב של checksum זה נעשה באופן אוטומטי על ידי המערכת!

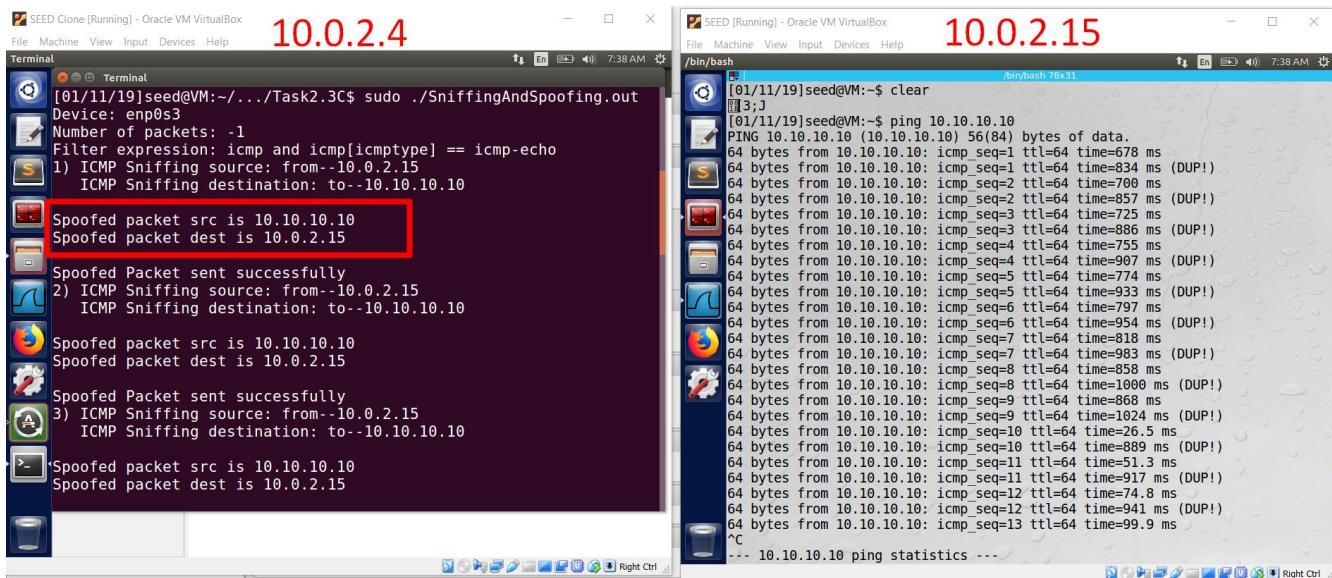
6. יש צורך בהרשות מנהל, משום שכל המידע שייה ב socket שנחננו נבנה, יהיה שונה ממה שהמערכת נונת בדרך כלל, ואם זה לא יהיה באישורה לא יוכל להוציא את זה לפועל.

## Task 2.3

השינוי העיקרי מפעם קודמת הוא

בדומה לשאלה 1.4 בחלק הקודם . כדי לבצע משימה כזאת , علينا להפעיל שתי מכונות . לוודא שיש להן שתי כתובות קיימות.

הפעלנו במכונה א'(10.0.2.4) את ה `sniffer and spoofer` שלנו כדי Shinnter את התעבורה ברשת שלנו. במכונה ב' (10.0.2.15) עשינו ping ל 10.10.10.10 כתובת שאינה קיימת, ונitinן לראות שהפוך sniffer and spoofer מזהה שנשלחות ברשת שלו בקשוט ping ומחזיר מיד תשובת pong אל המבוקש:



נראה כי המכונה ב' הצלילה לקבל פונג בכל מקרה , ובאמת ניתן של ה wireshark מראה כי נשלחות שתי הודעות pong על כל Ping אחד שנשלח:

