Experiment 2

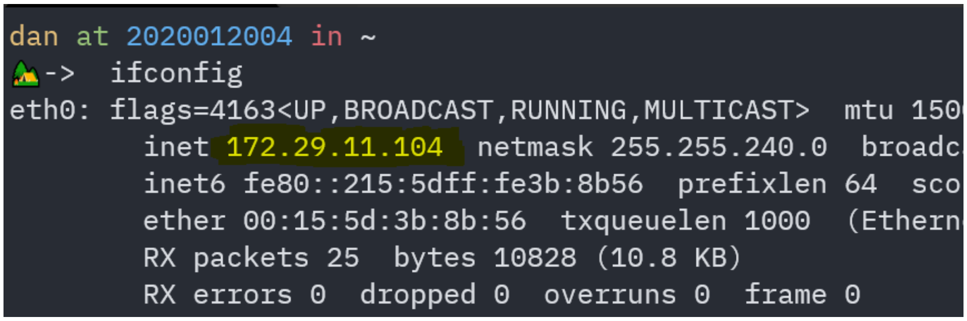
Name: Danyl Fernandes (72)
Class: TE COMPS
XIE ID: 2020012004
Date: 26-07-2021

Aim: Use basic networking commands in Linux (ifconfig, ping, traceroute, nslookup, netstat, ARP, host, ip, route)

Commands:

ifconfig:

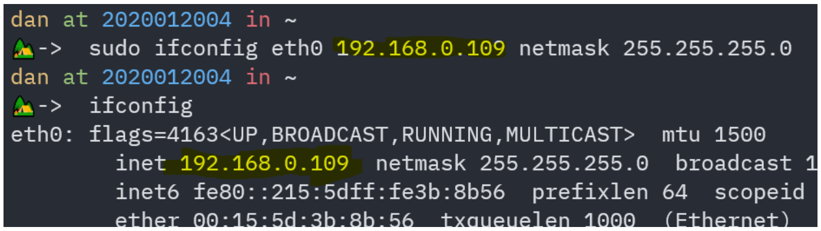
- ifconfig stands for Interface configuration command
- It helps you to see detailed information about your network interfaces and details like your IP address, Subnet mask, etc.
- You can also disable or temporarily turn off certain interfaces using this command
- This command also has a Windows version which is called "ipconfig"
- **Syntax:** ifconfig
- **Output:**



```
dan at 2020012004 in ~
⛰-> ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 150
        inet 172.29.11.104  netmask 255.255.240.0  broadc
        inet6 fe80::215:5dff:fe3b:8b56  prefixlen 64  sco
        ether 00:15:5d:3b:8b:56  txqueuelen 1000  (Ethern
        RX packets 25  bytes 10828 (10.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
```

ifconfig eth0 192.168.0.109 netmask 255.255.255.0:

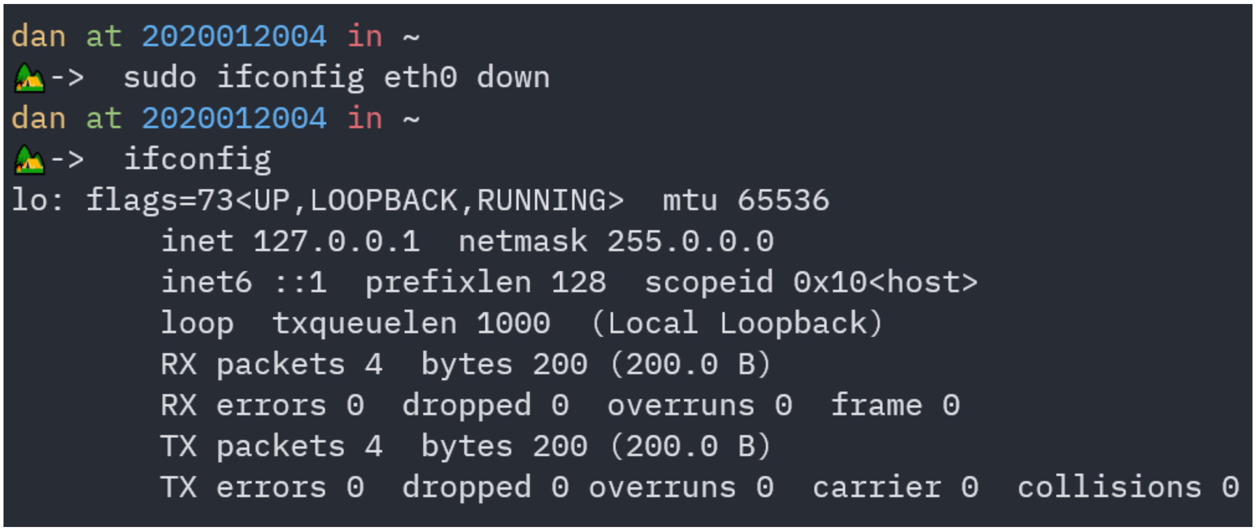
- ifconfig also helps if you want to change your IP address of an interface temporarily
- This command helps you to change your IP address for a given interface
- **Syntax:** ifconfig <interface_name> <new_ip> netmask <subnet>
- **Output:**



```
dan at 2020012004 in ~
⛰-> sudo ifconfig eth0 192.168.0.109 netmask 255.255.255.0
dan at 2020012004 in ~
⛰-> ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.109  netmask 255.255.255.0  broadcast 1
        inet6 fe80::215:5dff:fe3b:8b56  prefixlen 64  scopeid
        ether 00:15:5d:3b:8b:56  txqueuelen 1000  (Ethernet)
```

ifconfig eth0 down:

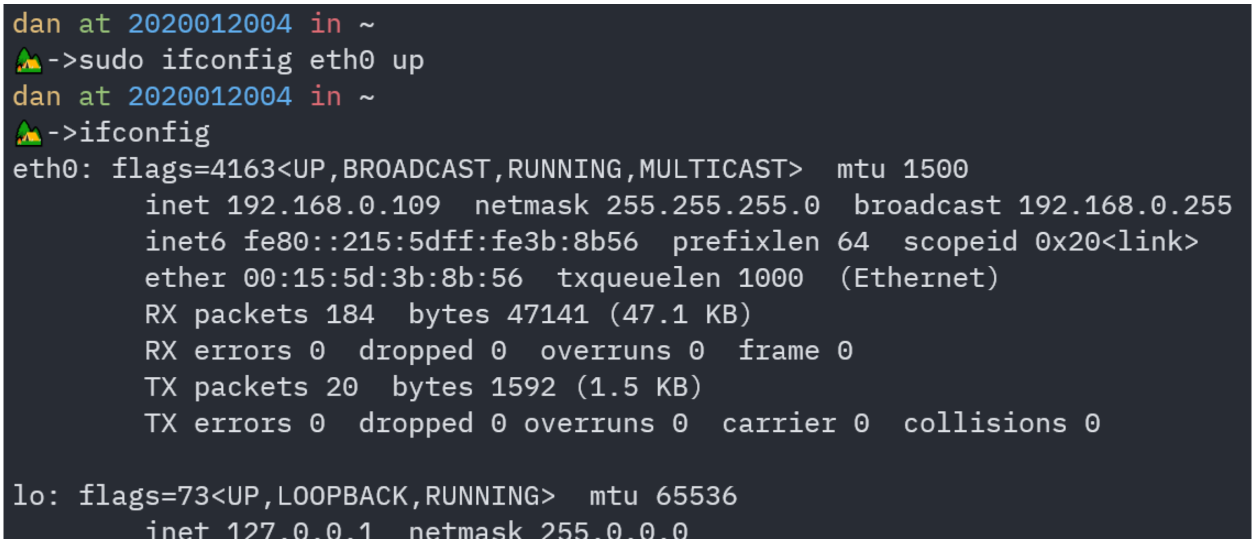
- This command will temporarily disable the interface you specify
- **Syntax:** ifconfig <interface_name> down
- **Output:**



```
dan at 2020012004 in ~
->  sudo ifconfig eth0 down
dan at 2020012004 in ~
->  ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4  bytes 200 (200.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 200 (200.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

ifconfig eth0 up:

- This command will enable the interface you specify
- **Syntax:** ifconfig <interface_name> up
- **Output:**

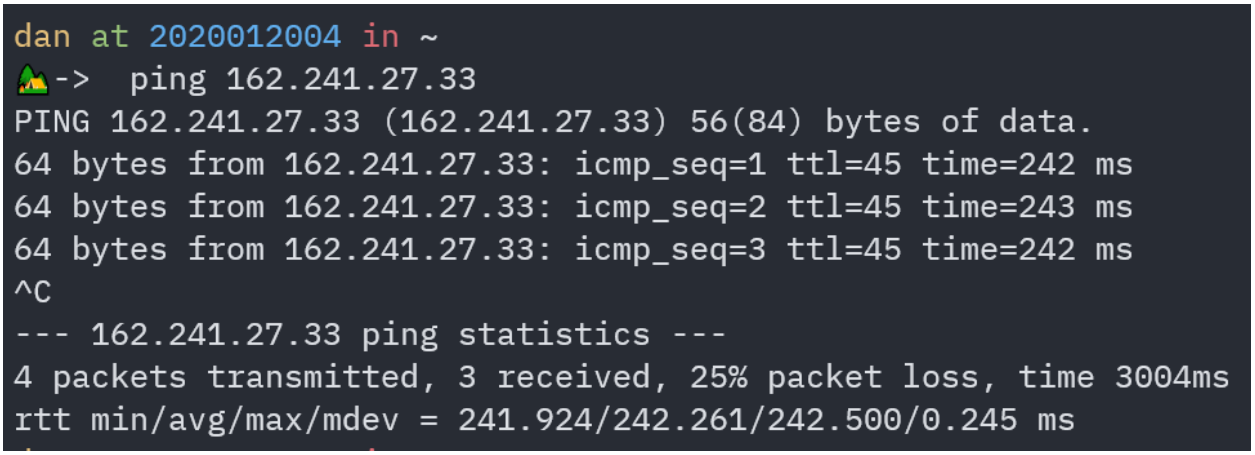


```
dan at 2020012004 in ~
->sudo ifconfig eth0 up
dan at 2020012004 in ~
->ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.109  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::215:5dff:fe3b:8b56  prefixlen 64  scopeid 0x20<link>
        ether 00:15:5d:3b:8b:56  txqueuelen 1000  (Ethernet)
        RX packets 184  bytes 47141 (47.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 1592 (1.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
```

ping 162.241.27.33:

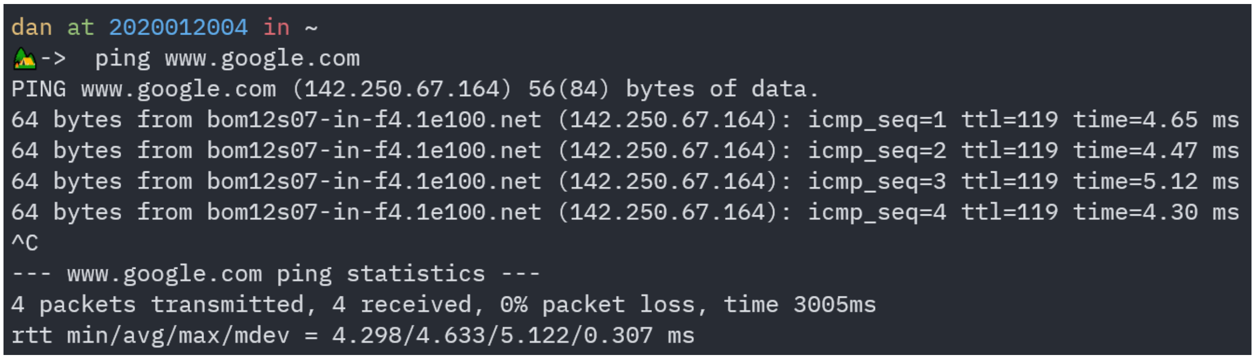
- The ping command helps us to test the reachability of any server connected to the internet
- You can specify the IP address or the domain name and the ping command will use ICMP packets to determine if the server is up or not
- **Syntax:** ping <ip or domain_name>
- **Output:**



```
dan at 2020012004 in ~
-> ping 162.241.27.33
PING 162.241.27.33 (162.241.27.33) 56(84) bytes of data.
64 bytes from 162.241.27.33: icmp_seq=1 ttl=45 time=242 ms
64 bytes from 162.241.27.33: icmp_seq=2 ttl=45 time=243 ms
64 bytes from 162.241.27.33: icmp_seq=3 ttl=45 time=242 ms
^C
--- 162.241.27.33 ping statistics ---
4 packets transmitted, 3 received, 25% packet loss, time 3004ms
rtt min/avg/max/mdev = 241.924/242.261/242.500/0.245 ms
```

ping www.google.com:

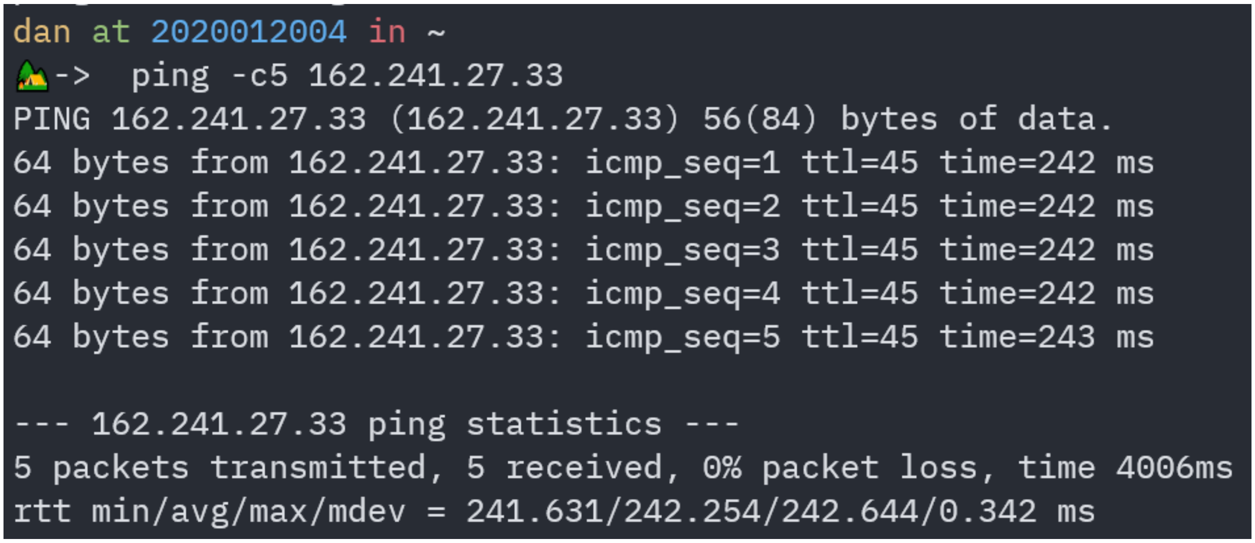
- This command is used to check the reachability to the domain google.com
- It will return responses from google.com, if it is up
- **Syntax:** ping <ip or domain_name>
- **Output:**



```
dan at 2020012004 in ~
-> ping www.google.com
PING www.google.com (142.250.67.164) 56(84) bytes of data.
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=1 ttl=119 time=4.65 ms
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=2 ttl=119 time=4.47 ms
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=3 ttl=119 time=5.12 ms
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=4 ttl=119 time=4.30 ms
^C
--- www.google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 4.298/4.633/5.122/0.307 ms
```

ping -c5 162.241.27.33:

- This command uses the -c (count) option that the ping command provides.
- This option will run the ping command only for the number of times specified.
- In this particular instance we have specified 5 counts.
- It will return responses from google.com, if it is up
- **Syntax:** ping -c <no_of_counts> <ip or domain_name>
- **Output:**

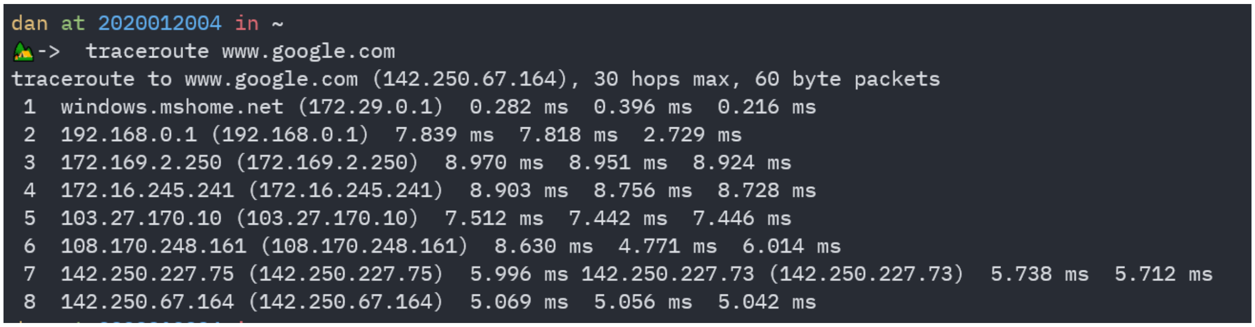


```
dan at 2020012004 in ~
 ->  ping -c5 162.241.27.33
PING 162.241.27.33 (162.241.27.33) 56(84) bytes of data.
64 bytes from 162.241.27.33: icmp_seq=1 ttl=45 time=242 ms
64 bytes from 162.241.27.33: icmp_seq=2 ttl=45 time=242 ms
64 bytes from 162.241.27.33: icmp_seq=3 ttl=45 time=242 ms
64 bytes from 162.241.27.33: icmp_seq=4 ttl=45 time=242 ms
64 bytes from 162.241.27.33: icmp_seq=5 ttl=45 time=243 ms

--- 162.241.27.33 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 241.631/242.254/242.644/0.342 ms
```

traceroute www.google.com:

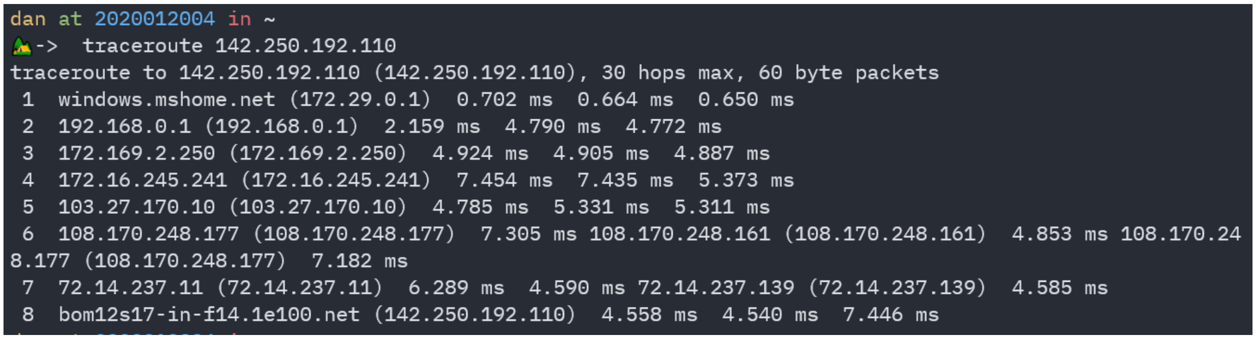
- The traceroute command shows us the no. of hops that it takes to reach from your computer to a destination domain.
- It uses ICMP, TCP or UDP probing to send these packets and identify the routers that it encounters on the way to the destination.
- This command checks the route to google.com
- **Syntax:** traceroute <ip or domain_name>
- **Output:**



```
dan at 2020012004 in ~
 ->  traceroute www.google.com
traceroute to www.google.com (142.250.67.164), 30 hops max, 60 byte packets
 1  windows.mshome.net (172.29.0.1)  0.282 ms  0.396 ms  0.216 ms
 2  192.168.0.1 (192.168.0.1)  7.839 ms  7.818 ms  2.729 ms
 3  172.169.2.250 (172.169.2.250)  8.970 ms  8.951 ms  8.924 ms
 4  172.16.245.241 (172.16.245.241)  8.903 ms  8.756 ms  8.728 ms
 5  103.27.170.10 (103.27.170.10)  7.512 ms  7.442 ms  7.446 ms
 6  108.170.248.161 (108.170.248.161)  8.630 ms  4.771 ms  6.014 ms
 7  142.250.227.75 (142.250.227.75)  5.996 ms 142.250.227.73 (142.250.227.73)  5.738 ms  5.712 ms
 8  142.250.67.164 (142.250.67.164)  5.069 ms  5.056 ms  5.042 ms
```

traceroute 142.250.192.110:

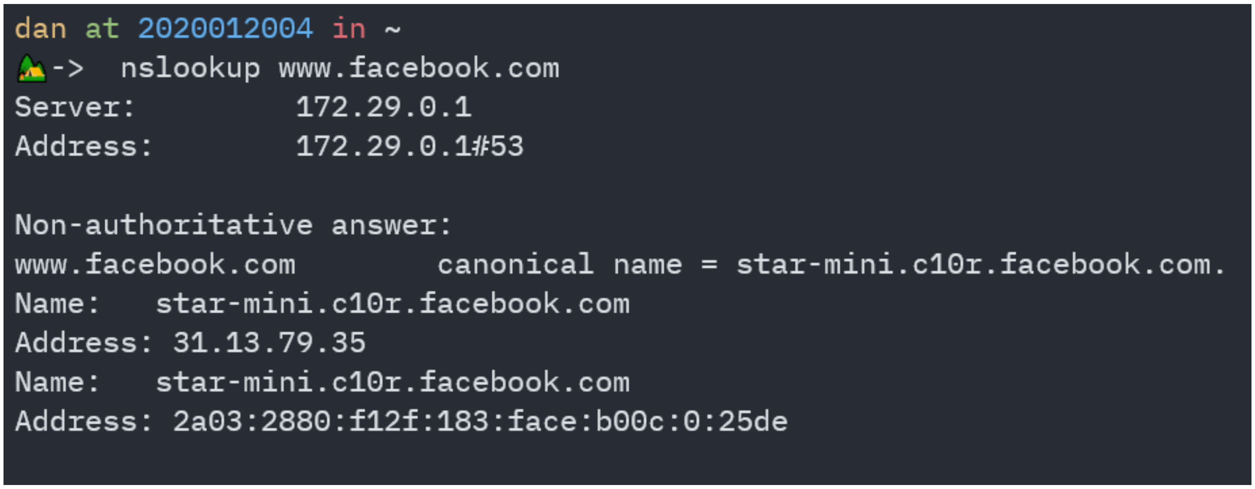
- The traceroute command also allows us to use an IP address directly.
- It does the same thing that it does for domain names to identify the route to the destination
- This command checks the route to the IP address 142.250.192.110
- **Syntax:** traceroute <ip or domain_name>
- **Output:**



```
dan at 2020012004 in ~
-> traceroute 142.250.192.110
traceroute to 142.250.192.110 (142.250.192.110), 30 hops max, 60 byte packets
 1  windows.mshome.net (172.29.0.1)  0.702 ms  0.664 ms  0.650 ms
 2  192.168.0.1 (192.168.0.1)  2.159 ms  4.790 ms  4.772 ms
 3  172.169.2.250 (172.169.2.250)  4.924 ms  4.905 ms  4.887 ms
 4  172.16.245.241 (172.16.245.241)  7.454 ms  7.435 ms  5.373 ms
 5  103.27.170.10 (103.27.170.10)  4.785 ms  5.331 ms  5.311 ms
 6  108.170.248.177 (108.170.248.177)  7.305 ms 108.170.248.161 (108.170.248.161)  4.853 ms 108.170.24
8.177 (108.170.248.177)  7.182 ms
 7  72.14.237.11 (72.14.237.11)  6.289 ms  4.590 ms 72.14.237.139 (72.14.237.139)  4.585 ms
 8  bom12s17-in-f14.1e100.net (142.250.192.110)  4.558 ms  4.540 ms  7.446 ms
```

nslookup www.facebook.com:

- The nslookup command helps us get DNS information for a given domain or IP
- It can give a Non-Authorized or Authorized answer
- It gives all the information about the IP, domain and name server
- This command does a lookup on facebook.com
- **Syntax:** nslookup <ip or domain_name>
- **Output:**

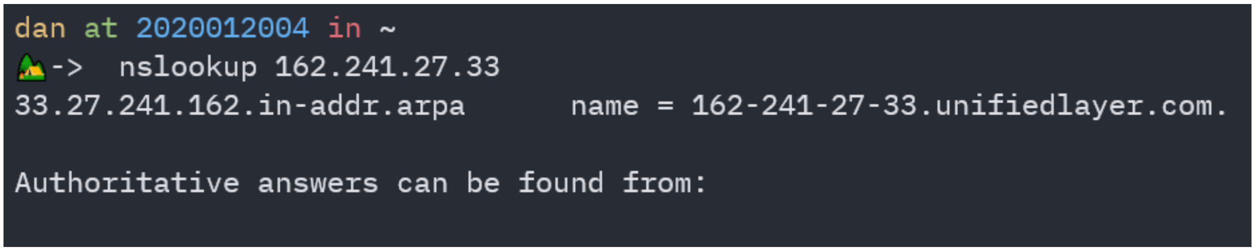


```
dan at 2020012004 in ~
-> nslookup www.facebook.com
Server:         172.29.0.1
Address:        172.29.0.1#53

Non-authoritative answer:
www.facebook.com        canonical name = star-mini.c10r.facebook.com.
Name:   star-mini.c10r.facebook.com
Address: 31.13.79.35
Name:   star-mini.c10r.facebook.com
Address: 2a03:2880:f12f:183:face:b00c:0:25de
```

nslookup 162.241.27.33:

- The nslookup command helps us get DNS information for a given domain or IP
- It can give a Non-Authorized or Authorized answer
- It gives all the information about the IP, domain and name server
- This command does a lookup on facebook.com
- **Syntax:** nslookup <ip or domain_name>
- **Output:**

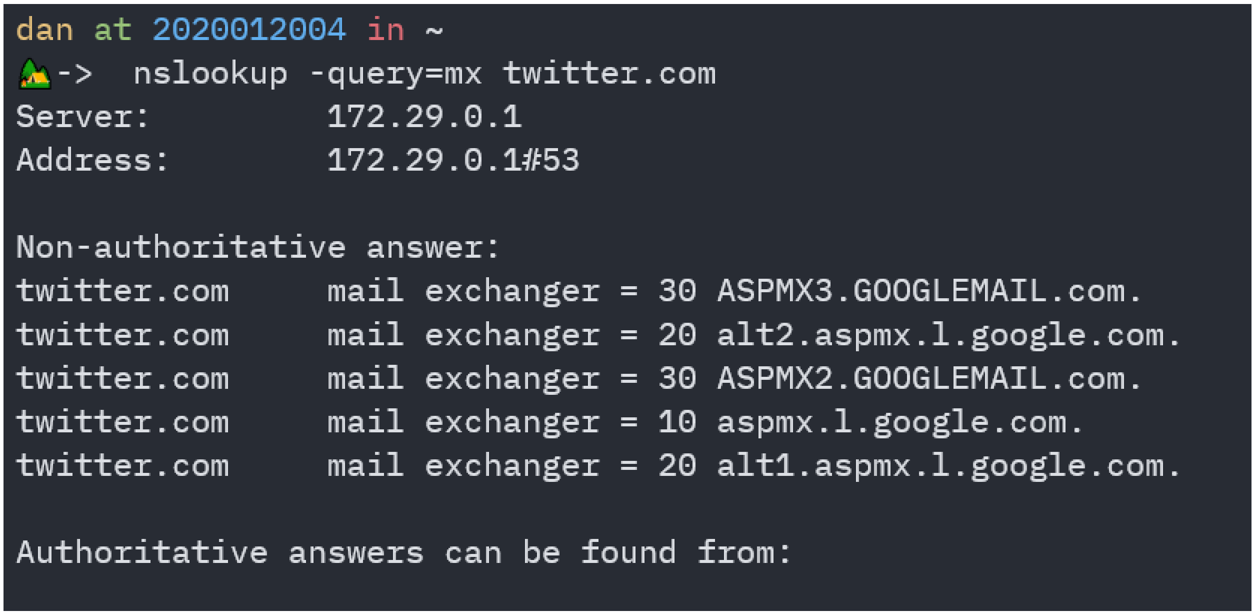


```
dan at 2020012004 in ~
-> nslookup 162.241.27.33
33.27.241.162.in-addr.arpa      name = 162-241-27-33.unifiedlayer.com.

Authoritative answers can be found from:
```

nslookup -query=mx twitter.com:

- This command does a lookup but the -query=mx option tells it that it should look for mail server on the domain that we have specified
- It returns all the mail servers that may exist on the twitter.com domain name
- This command does a lookup on twitter.com
- **Syntax:** nslookup -query=mx <ip or domain_name>
- **Output:**



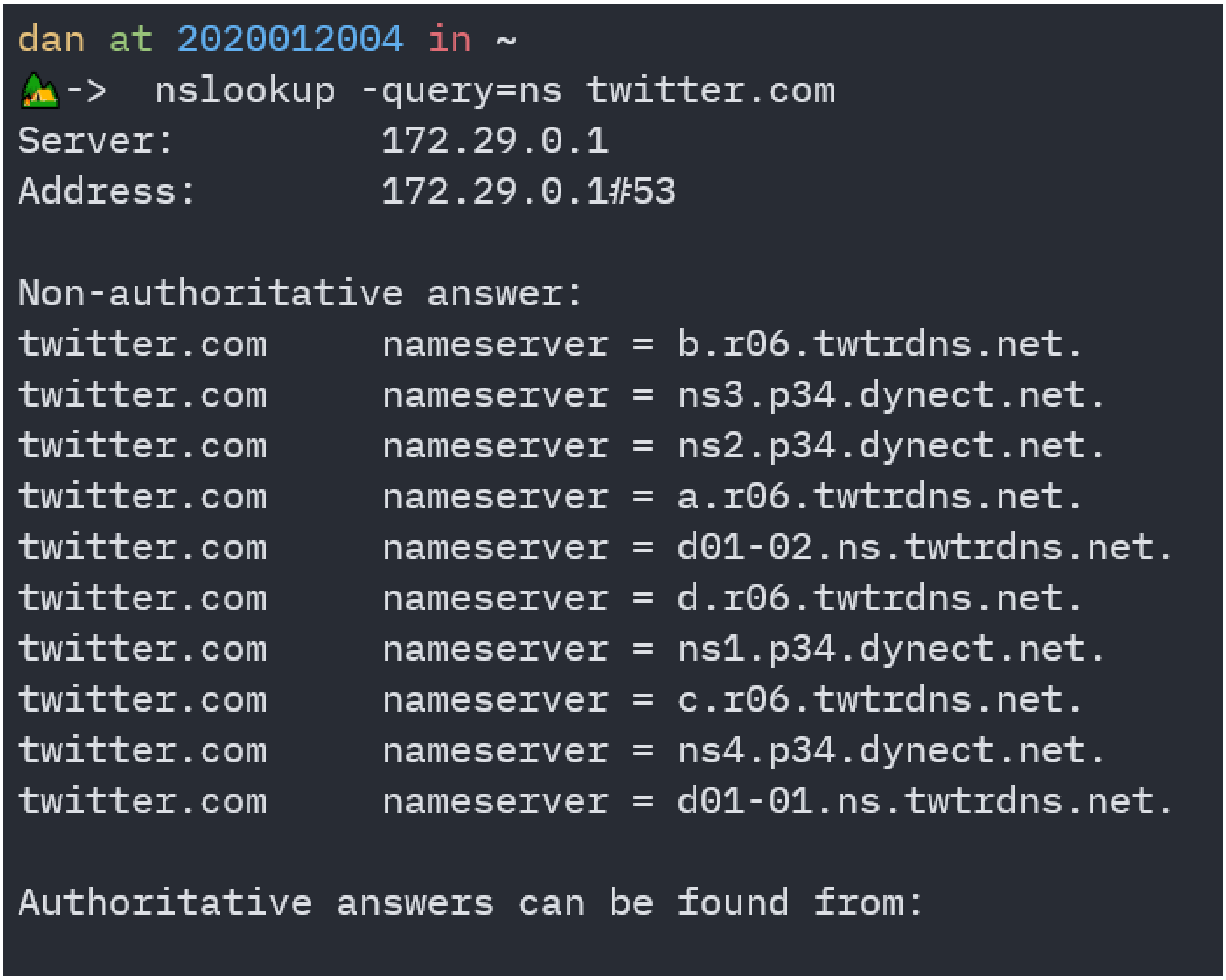
```
dan at 2020012004 in ~
-> nslookup -query=mx twitter.com
Server:         172.29.0.1
Address:        172.29.0.1#53

Non-authoritative answer:
twitter.com     mail exchanger = 30 ASPMX3.GOOGLEMAIL.com.
twitter.com     mail exchanger = 20 alt2.aspmx.l.google.com.
twitter.com     mail exchanger = 30 ASPMX2.GOOGLEMAIL.com.
twitter.com     mail exchanger = 10 aspmx.l.google.com.
twitter.com     mail exchanger = 20 alt1.aspmx.l.google.com.

Authoritative answers can be found from:
```

nslookup -query=ns twitter.com:

- This command uses the -query=ns option that tells it that it should look for name servers only on the domain that we have specified.
- This command does a lookup on twitter.com
- **Syntax:** nslookup -query=ns <ip or domain_name>
- **Output:**



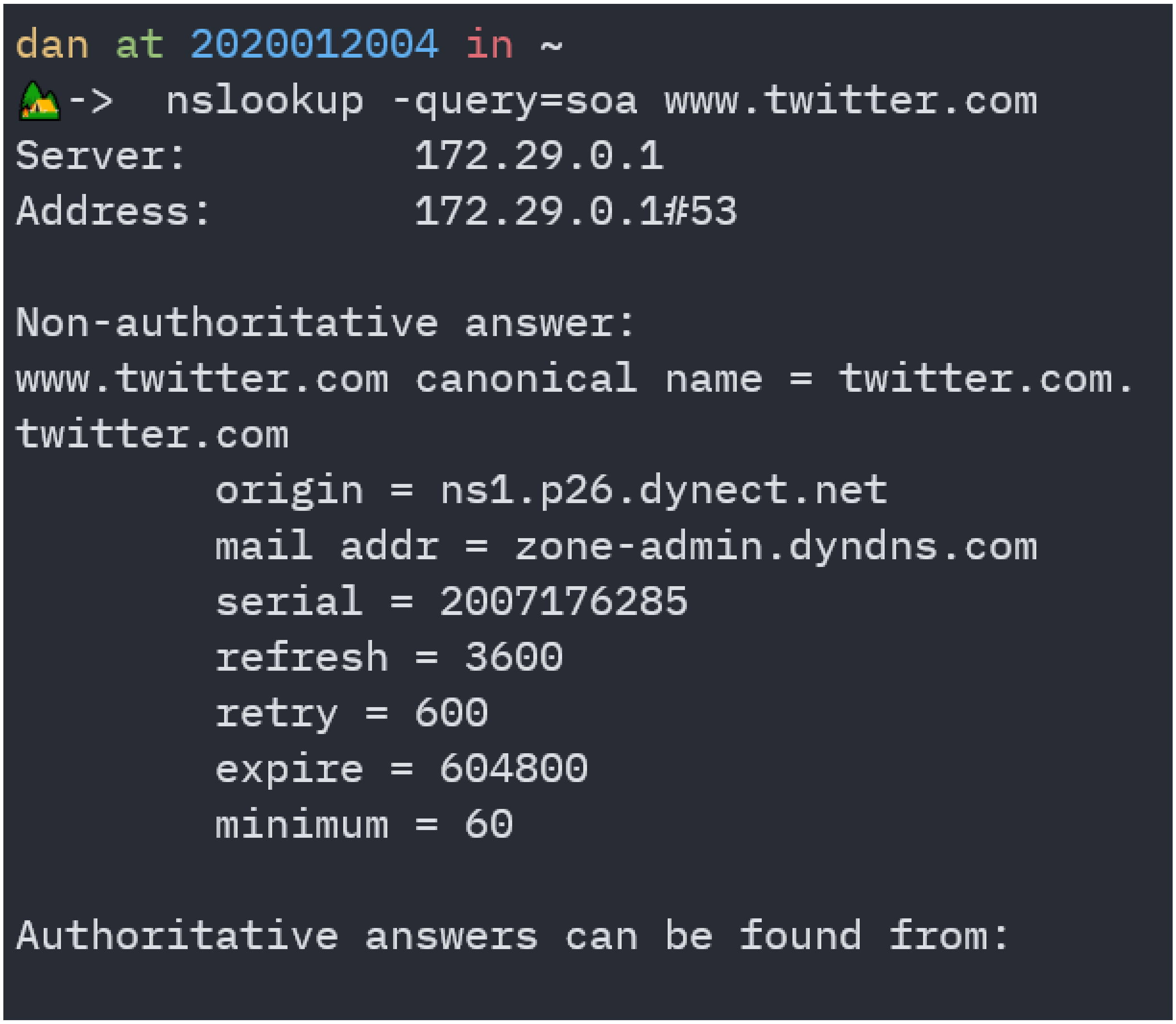
```
dan at 2020012004 in ~
⛺->  nslookup -query=ns twitter.com
Server:         172.29.0.1
Address:        172.29.0.1#53

Non-authoritative answer:
twitter.com     nameserver = b.r06.twtrdns.net.
twitter.com     nameserver = ns3.p34.dynect.net.
twitter.com     nameserver = ns2.p34.dynect.net.
twitter.com     nameserver = a.r06.twtrdns.net.
twitter.com     nameserver = d01-02.ns.twtrdns.net.
twitter.com     nameserver = d.r06.twtrdns.net.
twitter.com     nameserver = ns1.p34.dynect.net.
twitter.com     nameserver = c.r06.twtrdns.net.
twitter.com     nameserver = ns4.p34.dynect.net.
twitter.com     nameserver = d01-01.ns.twtrdns.net.

Authoritative answers can be found from:
```

nslookup -query=soa twitter.com:

- By default, nslookup returns Non-Authorized answers.
- If we want to get back an Authorized answer, we must specify the -query=soa option.
- This option soa means Start of Authority and it is another type of DNS record like mx and ns
- This command does a lookup on twitter.com
- **Syntax:** nslookup -query=soa <ip or domain_name>
- **Output:**



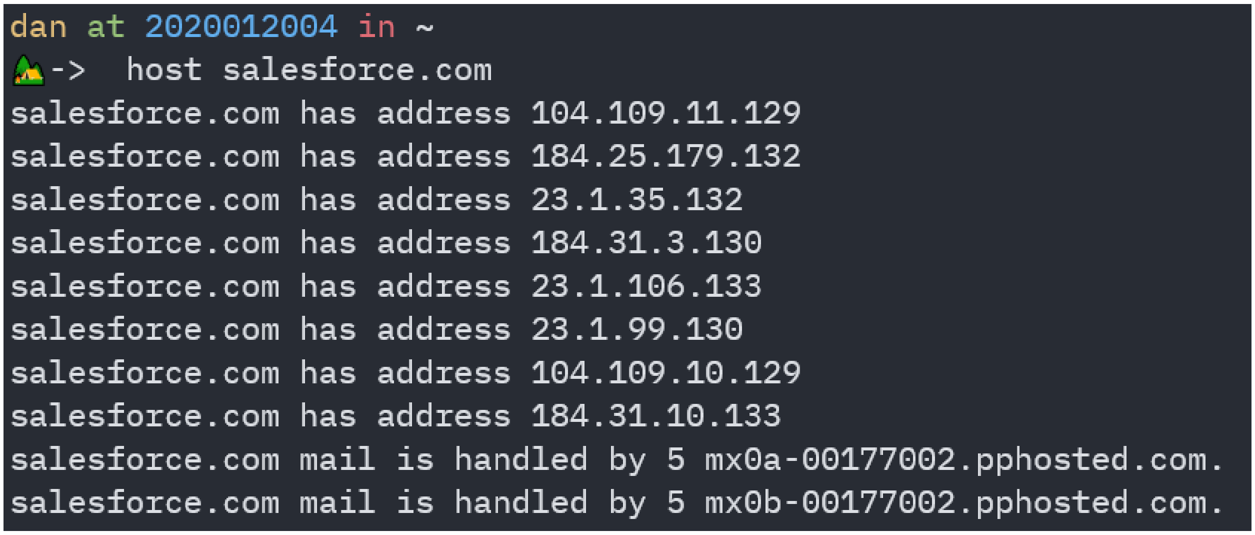
```
dan at 2020012004 in ~
 ->  nslookup -query=soa www.twitter.com
Server:         172.29.0.1
Address:        172.29.0.1#53

Non-authoritative answer:
www.twitter.com canonical name = twitter.com.
twitter.com
        origin = ns1.p26.dynect.net
        mail addr = zone-admin.dyndns.com
        serial = 2007176285
        refresh = 3600
        retry = 600
        expire = 604800
        minimum = 60

Authoritative answers can be found from:
```

host salesforce.com:

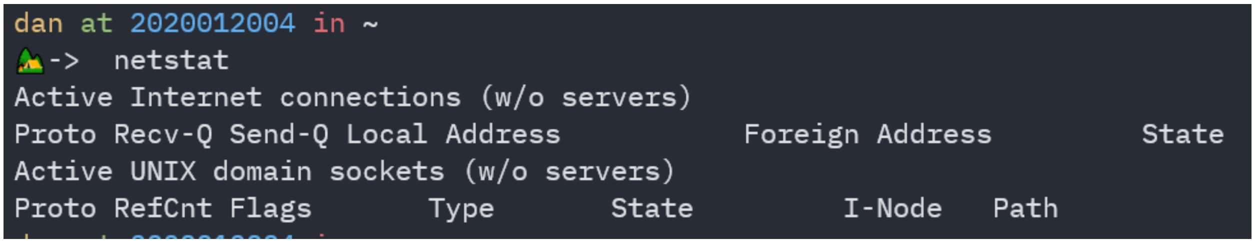
- The host command in Linux is used to Domain Name Server lookup operations.
- It is useful if you want to find the IP address of any given domain name
- This command does a lookup on salesforce.com
- **Syntax:** host <domain_name>
- **Output:**



```
dan at 2020012004 in ~
⛰->  host salesforce.com
salesforce.com has address 104.109.11.129
salesforce.com has address 184.25.179.132
salesforce.com has address 23.1.35.132
salesforce.com has address 184.31.3.130
salesforce.com has address 23.1.106.133
salesforce.com has address 23.1.99.130
salesforce.com has address 104.109.10.129
salesforce.com has address 184.31.10.133
salesforce.com mail is handled by 5 mx0a-00177002.pphosted.com.
salesforce.com mail is handled by 5 mx0b-00177002.pphosted.com.
```

netstat:

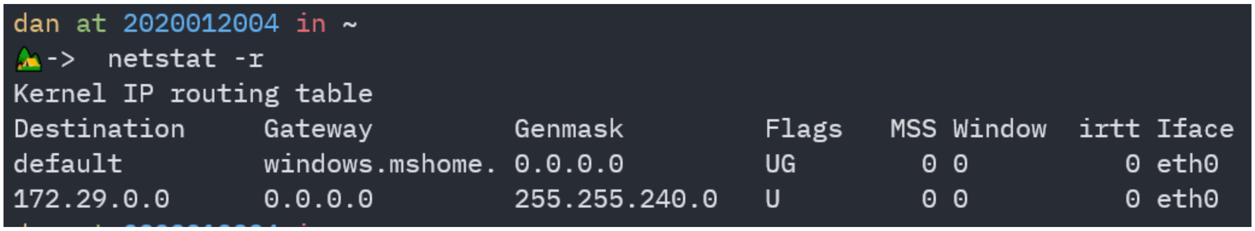
- This is an extensive command that displays network connections for Transmission Control Protocol, routing tables, and a number of network interfaces and network protocol statistics.
- It helps to see active connections, ports and services.
- It gives a very verbose output and we can use various options to filter out the unnecessary options.
- **Syntax:** netstat
- **Output:**



```
dan at 2020012004 in ~
⛰->  netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
```

netstat -r:

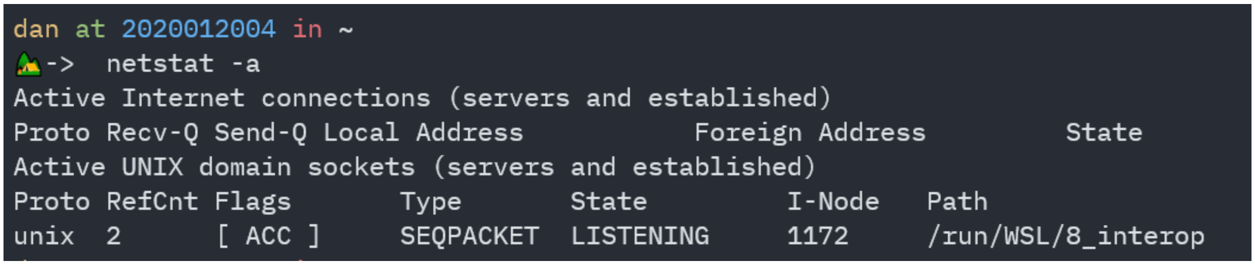
- This option -r stands for routing tables.
- It displays all the kernel routing tables for the current system.
- **Syntax:** netstat -r
- **Output:**



```
dan at 2020012004 in ~
-> netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
default         windows.mshome. 0.0.0.0         UG        0 0          0 eth0
172.29.0.0      0.0.0.0         255.255.240.0   U         0 0          0 eth0
```

netstat -a:

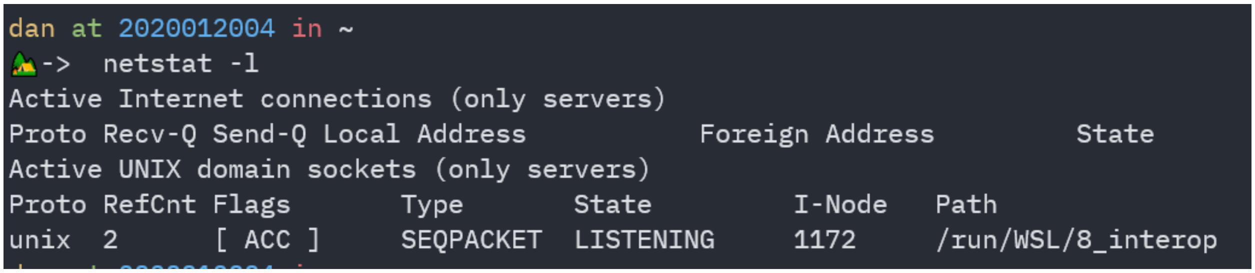
- This option -a stands for all and it shows both listening and non-listening, TCP established connection sockets.
- **Syntax:** netstat -a
- **Output:**



```
dan at 2020012004 in ~
-> netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ACC ]     SEQPACKET  LISTENING     1172     /run/WSL/8_interop
```

netstat -l:

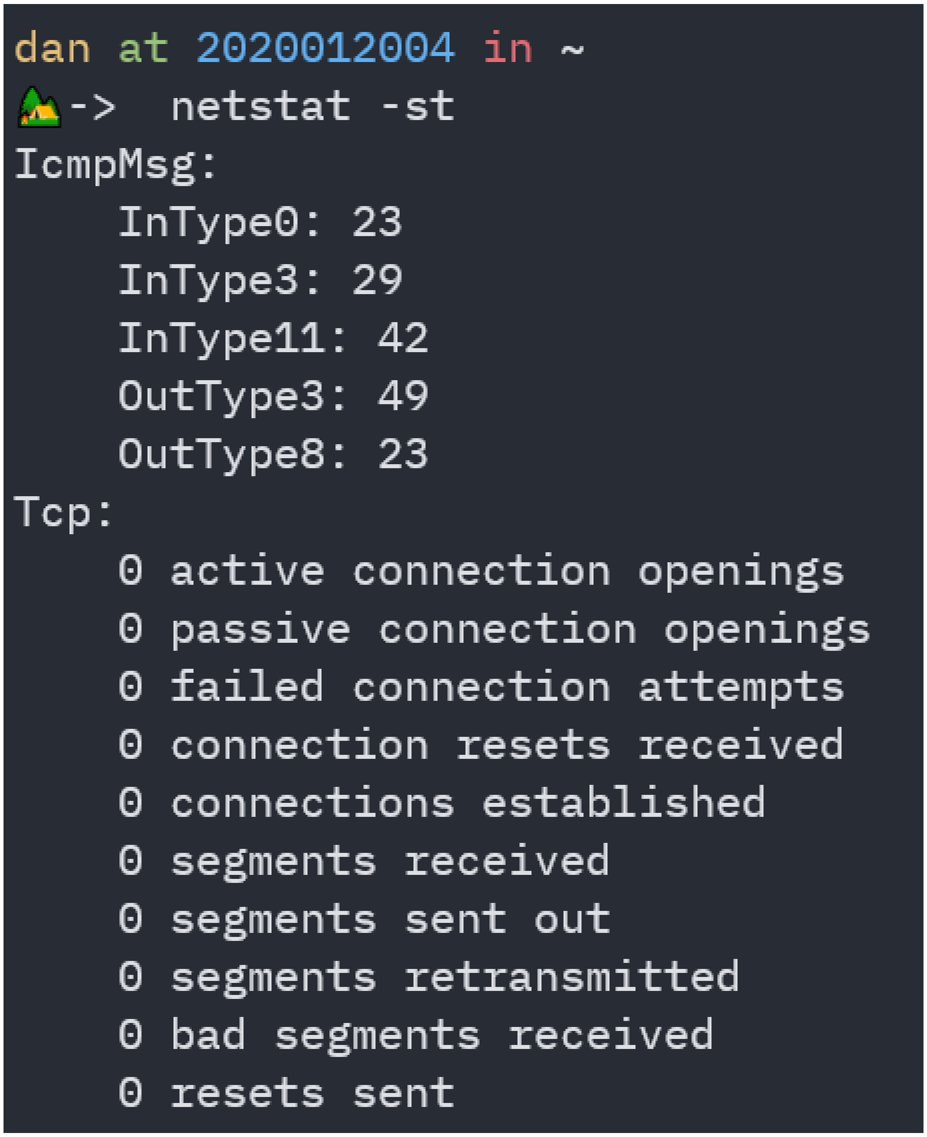
- The default netstat command omits the output that this command gives us.
- This command shows all the listening sockets on the current machine configuration.
- **Syntax:** netstat -l
- **Output:**



```
dan at 2020012004 in ~
-> netstat -l
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
Active UNIX domain sockets (only servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ACC ]     SEQPACKET  LISTENING     1172     /run/WSL/8_interop
```

netstat -st:

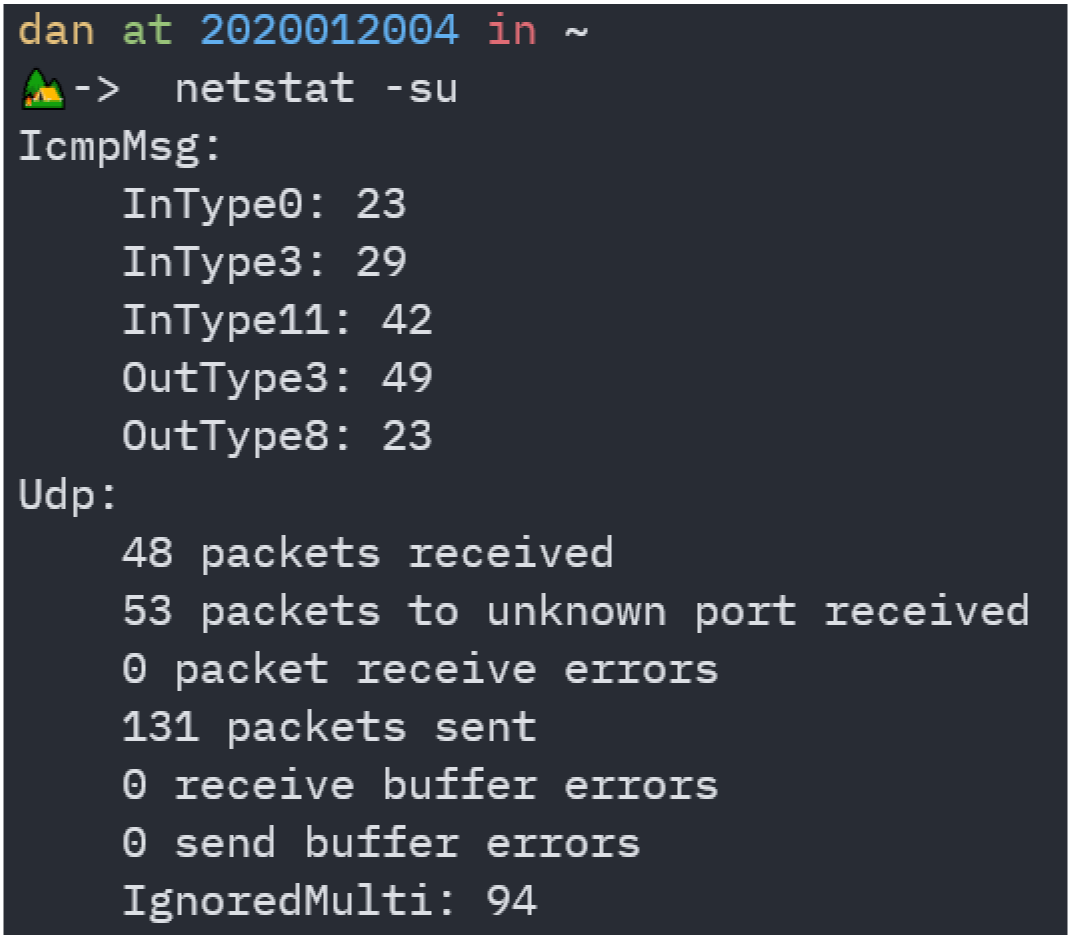
- This command specifies two options: the s and t options.
- The -s option stands for statistics and it displays the summary statistics for the protocol that you specify.
- The -t option stands for the TCP protocol for which the statistics shall be displayed.
- **Syntax:** netstat -st
- **Output:**



```
dan at 2020012004 in ~
🏕-> netstat -st
IcmpMsg:
    InType0: 23
    InType3: 29
    InType11: 42
    OutType3: 49
    OutType8: 23
Tcp:
    0 active connection openings
    0 passive connection openings
    0 failed connection attempts
    0 connection resets received
    0 connections established
    0 segments received
    0 segments sent out
    0 segments retransmitted
    0 bad segments received
    0 resets sent
```

netstat -su:

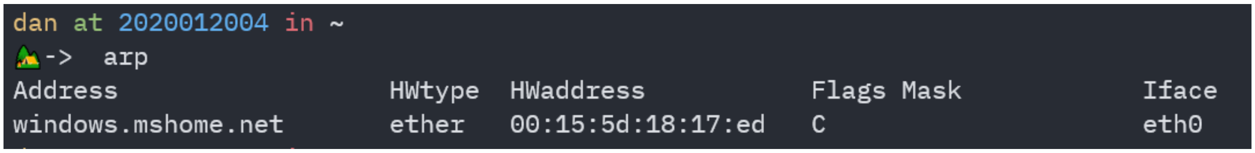
- This command specifies two options: the s and u options.
- The -s option stands for statistics and it displays the summary statistics for the protocol that you specify.
- The -u option stands for the UDP protocol for which the statistics shall be displayed.
- **Syntax:** netstat -su
- **Output:**



```
dan at 2020012004 in ~
🏕-> netstat -su
IcmpMsg:
    InType0: 23
    InType3: 29
    InType11: 42
    OutType3: 49
    OutType8: 23
Udp:
    48 packets received
    53 packets to unknown port received
    0 packet receive errors
    131 packets sent
    0 receive buffer errors
    0 send buffer errors
    IgnoredMulti: 94
```

arp:

- This command displays the Internet-to-Ethernet address translation tables used by the address resolution protocol (ARP).
- It provides various options to get specific data regarding the address tables.
- The primary function of this table is to resolve the IP address of a system to its MAC address.
- It works between the Data Link and the Network layer.
- **Syntax:** arp
- **Output:**



```
dan at 2020012004 in ~
 ->  arp
Address                  HWtype  HWaddress           Flags Mask            Iface
windows.mshome.net       ether   00:15:5d:18:17:ed   C                     eth0
```

Conclusion: Hence we learned to use basic networking commands in Linux.