ECE 456
Lab 08

Pablo Corona Alvarez

Ifconfig:

```
pablocor@node-0:~$ ifconfig -v
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.3.19  netmask 255.240.0.0  broadcast 172.31.255.255
        inet6 fe80::34:52ff:fe8c:dfc2  prefixlen 64  scopeid 0x20<link>
        ether 02:34:52:8c:df:c2  txqueuelen 1000  (Ethernet)
        RX packets 13893  bytes 1086653 (1.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 19666  bytes 2585945 (2.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.1.2  netmask 255.255.255.0  broadcast 10.10.1.255
        inet6 fe80::c0:d6ff:fe70:932f  prefixlen 64  scopeid 0x20<link>
        ether 02:c0:d6:70:93:2f  txqueuelen 1000  (Ethernet)
        RX packets 92  bytes 6961 (6.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 28  bytes 4113 (4.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.2.2  netmask 255.255.255.0  broadcast 10.10.2.255
        inet6 fe80::31:abff:feae:3af4  prefixlen 64  scopeid 0x20<link>
        ether 02:31:ab:ae:3a:f4  txqueuelen 1000  (Ethernet)
        RX packets 70  bytes 4431 (4.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 29  bytes 4217 (4.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.3.2  netmask 255.255.255.0  broadcast 10.10.3.255
        inet6 fe80::bc:47ff:fe77:6a3  prefixlen 64  scopeid 0x20<link>
        ether 02:bc:47:77:06:a3  txqueuelen 1000  (Ethernet)
        RX packets 77  bytes 5393 (5.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 29  bytes 4217 (4.2 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
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pablocor@node-0:~$ ifconfig -s
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1500    13914      0      0 0         19691      0      0      0 BMRU
eth1      1500       92      0      0 0            28      0      0      0 BMRU
eth2      1500       70      0      0 0            29      0      0      0 BMRU
eth3      1500       77      0      0 0            29      0      0      0 BMRU
lo       65536        0      0      0 0             0      0      0      0 LRU
```

Figure 1: Ifconfig running

IFconfig is used to set up the kernel resident network interfaces, however it is typically not used in setting up interfaces but rather debugging or fine tuning the interfaces. When the command itself is ran it displays the status of active interfaces. However with the option -s we are able to make this list more simplified and compact

In order to run ifconfig we simply use ifconfig followed by any flag we are going to use. I attempted to use the other functionalities of ifconfig such as add, del, and -broadcast however i was unable to get this sections working properly.

Iwconfig:

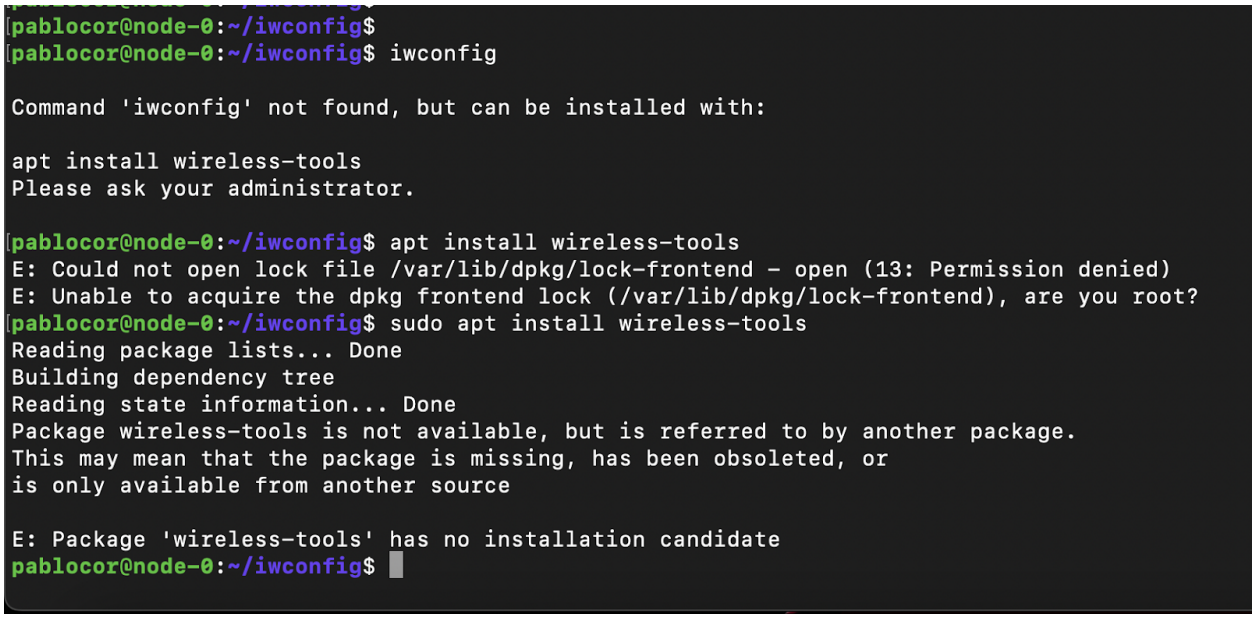

Figure 2: iwconfig attempts

Iwconfig works the same way that ifconfig does. The difference is that one shows wireless interfaces (iwconfig) while the other shows physical interfaces. If it were to work we can show many similarities between the two. We can show the information of all using the syntax

Iwconfig

But if we were looking to have the information of only one of the wireless interfaces all we would have to do is specify which interface we want information from. For example

Iwconfig mad-cow

Arp:

```
[pablocor@node-0:~$
[pablocor@node-0:~$
[pablocor@node-0:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
172.17.253.254           ether   fe:ff:ff:ff:ff:ff   C                     eth0
172.16.0.3               ether   fe:ff:ff:ff:ff:ff   C                     eth0
172.31.245.143           ether   fe:ff:ff:ff:ff:ff   C                     eth0
pc3.instageni.colorado.  ether   fe:ff:ff:ff:ff:ff   C                     eth0
172.16.0.1               ether   fe:ff:ff:ff:ff:ff   C                     eth0
pablocor@node-0:~$
```

Figure 3: Arp running displaying kernals

Arp can be used to manipulate or display the kernels IPv4 network cache. This is also able to add or delete entries within the table. Here we can see each of the different connections that this machine is on the same network as. I attempted using the arp command to add a connection to respond to requests; however, I feel this failed because I did not have access to root.

In order to run this command to see networks cache status one can simply use ARP
While adding and deleting entries is far more complicated.

Hostname:

```
[pablocor@node-0:~$
[pablocor@node-0:~$
[pablocor@node-0:~$ hostname
node-0.pablolab08.ch-geni-net.instageni.colorado.edu
[pablocor@node-0:~$ hostname -a
node-0.PabloLab08.ch-geni-net.instageni.colorado.edu
[pablocor@node-0:~$ hostname -d
instageni.colorado.edu
[pablocor@node-0:~$ hostname -f
pcvm3-19.instageni.colorado.edu
[pablocor@node-0:~$ hostname -i
172.17.3.19
[pablocor@node-0:~$ hostname -I
172.17.3.19 10.10.1.2 10.10.2.2 10.10.3.2
[pablocor@node-0:~$ hostname -s
node-0
pablocor@node-0:~$
```

Figure 4: Running hostname and its options

Hostname is capable of determining the machines IP address as well as the machine name. Depending on the option that is chosen it will return the name as the IP address of a string.

The syntax of this command is very simple and just follows the following
Hostname [-option]

Ifup/ifdown:

```
pablocor@node-0:~$ ifup

Command 'ifup' not found, but can be installed with:

apt install ifupdown
apt install ifupdown2
apt install netscript-2.4

Ask your administrator to install one of them.

pablocor@node-0:~$ ifdown

Command 'ifdown' not found, but can be installed with:

apt install ifupdown
apt install ifupdown2
apt install netscript-2.4

Ask your administrator to install one of them.

pablocor@node-0:~$ sudo apt install ifupdown
Reading package lists... Done
Building dependency tree
Reading state information... Done
Package ifupdown is not available, but is referred to by another package.
This may mean that the package is missing, has been obsoleted, or
is only available from another source

E: Package 'ifupdown' has no installation candidate
pablocor@node-0:~$
```

Figure 5: attempting to run and install ifup/ifdown

```
[pcorona@linuxa1 ~]$
[pcorona@linuxa1 ~]$ ifup eno1
Users cannot control this device.
[pcorona@linuxa1 ~]$ ifdown eno1
Users cannot control this device.
[pcorona@linuxa1 ~]$
```

Figure 6: running ifup/ifdown on ets Linux

ifup and ifdown can be used to put up or take down interfaces based on the definitions in the file /etc/network/interfaces. With the options that are available you can also chose a specific file to bring the interfaces from. But in general follows the same format as if config. While i was able to run this command on the Linux machines from the engineering department I found I could not start my own

interface, I believe that this is because I don't have a file specifying what each interface and its options are.

Ifup eth0 [-option]
Ifdown eth0 [-option]

Is the general syntax that the command follows. And doing so will set up the specific interface named eth0. It is a similar concept for ifdown except this would take down the interface instead of put it up

Route:

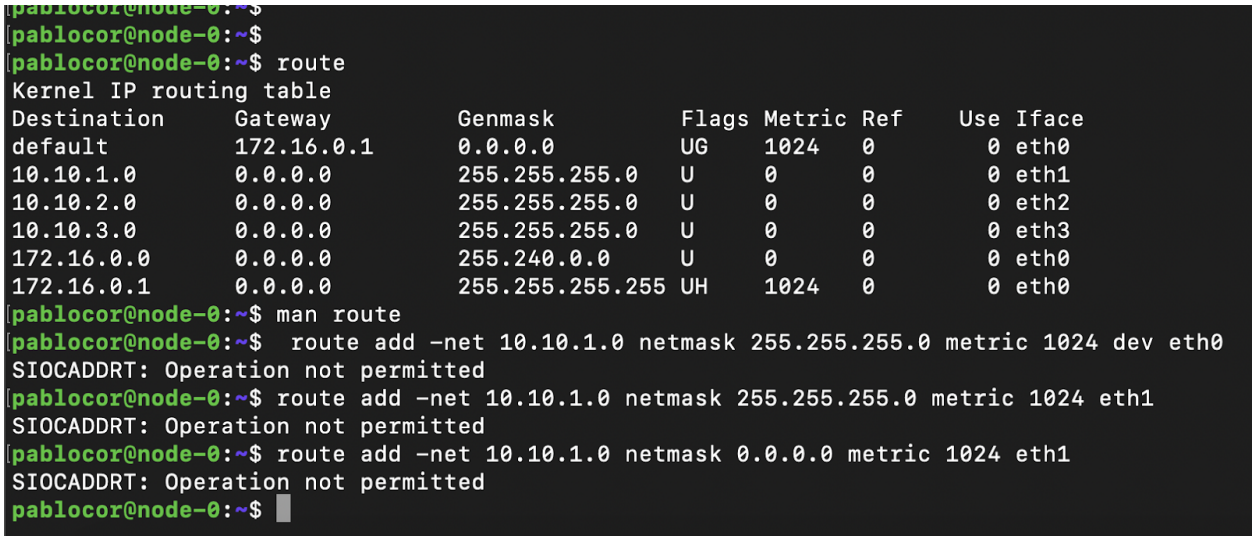

Figure 7: Route showing connections and different machines

Route is the command used in order to view and manipulate the ip routing table. As we can see there are all the visible routers.
route add -net 192.56.76.0 netmask 255.255.255.0 metric 1024 dev eth0
The command above is supposed to add a connection to networks 192.56.76.0 via eth0, however when this was changed into what I believe to be the appropriate connection I could still not have the operation be permitted. Once again switching over to the ets linux machines I ran these commands again but still ran into being held back by permissions.

Route
Route add -net [host] [targe]

The first instruction is going to show the Kernel IP routing table.
While the syntax in the second instruction is meant to either add a path from the host to the target or to remove the path if replacing add with del.

IP:

```
[pcorona@linuxa1 ~]$ ip route
default via 129.82.20.1 dev eno1 proto static metric 100
129.82.20.0/22 dev eno1 proto kernel scope link src 129.82.20.167 metric 100
[pcorona@linuxa1 ~]$ ip neigh
129.82.20.95 dev eno1 lladdr d0:67:26:e2:0d:b4 STALE
129.82.20.56 dev eno1 lladdr e6:62:a6:60:82:8c REACHABLE
129.82.20.160 dev eno1 lladdr d4:5d:64:bd:34:55 STALE
129.82.20.175 dev eno1 lladdr 00:1e:67:a6:86:27 STALE
129.82.21.111 dev eno1 lladdr 3c:ec:ef:46:35:b8 STALE
129.82.20.1 dev eno1 lladdr 00:00:0c:9f:f0:14 REACHABLE
129.82.20.182 dev eno1 lladdr ac:1f:6b:47:38:5a STALE
129.82.20.74 dev eno1 lladdr a0:36:9f:41:a8:5e REACHABLE
129.82.20.255 dev eno1 lladdr 3c:ec:ef:4d:fc:e8 STALE
129.82.21.10 dev eno1 lladdr ac:1f:6b:02:d1:50 STALE
129.82.20.88 dev eno1 lladdr a0:36:9f:55:83:36 STALE
129.82.22.19 dev eno1 lladdr 8c:ae:4c:fa:bc:33 STALE
129.82.20.110 dev eno1 lladdr ca:fe:1a:3d:f2:b8 STALE
129.82.20.168 dev eno1 lladdr ac:1f:6b:01:68:68 REACHABLE
129.82.20.2 dev eno1 lladdr 00:2a:6a:b7:fe:c1 STALE
129.82.20.76 dev eno1  FAILED
129.82.20.75 dev eno1  FAILED
129.82.20.248 dev eno1 lladdr ac:1f:6b:78:3b:83 STALE
129.82.20.89 dev eno1  FAILED
129.82.20.58 dev eno1 lladdr a0:36:9f:55:4f:76 STALE
129.82.21.98 dev eno1 lladdr ca:b0:ab:49:ba:8b REACHABLE
129.82.20.169 dev eno1 lladdr ac:1f:6b:01:6a:a4 STALE
129.82.21.105 dev eno1 lladdr 3c:ec:ef:46:2e:de STALE
129.82.20.3 dev eno1 lladdr 00:2a:6a:b8:23:c1 STALE
129.82.20.17 dev eno1 lladdr 54:80:28:fe:5f:00 STALE
129.82.20.90 dev eno1 lladdr a0:36:9f:55:83:36 REACHABLE
129.82.20.60 dev eno1 lladdr a0:36:9f:55:4f:76 REACHABLE
129.82.20.201 dev eno1 lladdr d0:67:26:e2:6f:48 STALE
129.82.23.248 dev eno1 lladdr 96:4e:35:64:31:15 DELAY
129.82.21.179 dev eno1 lladdr d0:50:99:07:fc:cf STALE
129.82.20.85 dev eno1 lladdr a0:36:9f:db:14:4a STALE
129.82.21.34 dev eno1 lladdr ce:41:f5:3e:c6:0c STALE
129.82.20.196 dev eno1 lladdr d0:67:26:e2:5f:fe STALE
129.82.20.195 dev eno1 lladdr d0:67:26:e2:1d:92 STALE
129.82.20.172 dev eno1 lladdr ac:1f:6b:01:66:ae STALE
129.82.21.108 dev eno1 lladdr 3c:ec:ef:46:34:82 STALE
129.82.20.171 dev eno1 lladdr ac:1f:6b:01:6a:98 STALE
129.82.20.178 dev eno1 lladdr 0c:c4:7a:ab:bc:aa STALE
129.82.20.86 dev eno1 lladdr a0:36:9f:55:81:3a STALE
129.82.21.102 dev eno1 lladdr 3c:ec:ef:46:35:64 STALE
129.82.20.179 dev eno1 lladdr 0c:c4:7a:ab:bc:ae STALE
129.82.20.87 dev eno1  FAILED
129.82.21.68 dev eno1 lladdr 3c:ec:ef:0d:2e:1e STALE
129.82.20.94 dev eno1 lladdr 54:80:28:fe:ff:00 STALE
129.82.20.198 dev eno1 lladdr d0:67:26:e2:cf:5a STALE
129.82.21.103 dev eno1 lladdr 3c:ec:ef:46:2e:a2 STALE
129.82.20.174 dev eno1 lladdr 00:1e:67:a6:b3:0d STALE
129.82.21.147 dev eno1 lladdr 00:0c:6c:0a:b7:97 STALE
129.82.20.66 dev eno1 lladdr 54:e0:32:d6:6b:80 STALE
[pcorona@linuxa1 ~]$
```

Figure 8: IP

Ip is capable of showing/manipulating routing, devices, policies and tunneling.
Here are two instructions that we have run on IP. The first command is to show the routing table by using the route option in conjunction with IP. In the second command we are looking for the current neighbor table from the kernel. Unfortunately I was unable to do any manipulation of the routing or devices. As this required the use of the administrator privileges which i did not have access to on the Geni machines nor the ETS machines on the school campus

Ping:

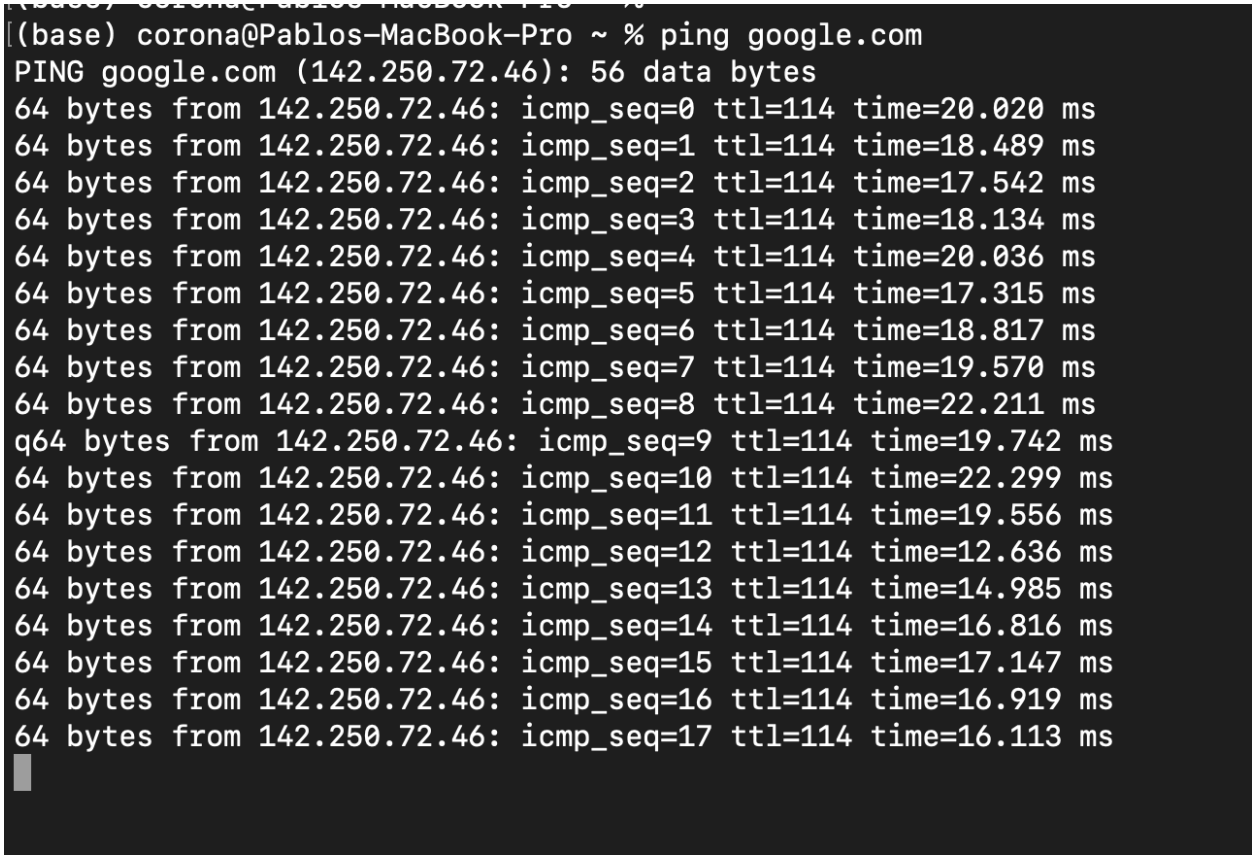

Figure 9: pinging google

Ping [-options] [host]

Here we can see that the ping command will send 64 bytes to google and then it will receive them again and record the time that it takes for the ping to come back to the machine. As we can also see from the figure that the ping will endlessly be sent over and over until terminated or if its specified to only send on packet and wait for the return of that single packet.

Pathping:
Unfortunately I was unable to run this machine as I do not have a windows machine at home and this lab was completed from home. The path ping instruction follows the following syntax

pathping [/n] [/h <maximumhops>] [/g <hostlist>] [/p <Period>] [/q <numqueries> [/w <timeout>] [/i <IPaddress>] [/4 <IPv4>] [/6 <IPv6>][<targetname>]

The importance of this instruction is that it functions in the same form as ping does but also provides information about lost packets along the path to its final destination. This can be important as it can be used to troubleshoot which particular router along the path is the cause of the issues that we can see arise.

Host:

```
(base) corona@Pablos-MacBook-Pro ~ %
(base) corona@Pablos-MacBook-Pro ~ % host colostate.edu
colostate.edu has address 129.82.103.79
colostate.edu has address 129.82.103.91
colostate.edu has address 129.82.103.64
colostate.edu has address 129.82.103.93
colostate.edu has address 129.82.103.78
colostate.edu has address 129.82.103.16
colostate.edu mail is handled by 0 colostate-edu.mail.protection.outlook.com.
(base) corona@Pablos-MacBook-Pro ~ %
(base) corona@Pablos-MacBook-Pro ~ %
(base) corona@Pablos-MacBook-Pro ~ %
(base) corona@Pablos-MacBook-Pro ~ %
(base) corona@Pablos-MacBook-Pro ~ %
(base) corona@Pablos-MacBook-Pro ~ % host google.com
google.com has address 142.250.72.46
google.com has IPv6 address 2607:f8b0:400f:804::200e
google.com mail is handled by 10 smtp.google.com.
(base) corona@Pablos-MacBook-Pro ~ %
```

Figure 10: Running host on two different domain names.

The Hose functions work very similar to nslookup, though can be only used for a singular domain name as where in nslookup you are capable of looking up multiple DNS by running in interactive mode rather than non interactive.

Host [-options] [name]

The most interesting thing about host is the difference that it will tell you what addresses the domain you are looking up has as well as the different types of protocols that those addresses are running on.

Netstat:

```
(base) corona@Pablos-MacBook-Pro ~ % netstat
Active Internet connections
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
tcp4       0      0  10.0.0.52.61541        ec2-52-4-187-135.https ESTABLISHED
tcp6       0      0  2601:282:8001:f9.61523 2601:282:8001:f9.56861 ESTABLISHED
tcp4       0      0  10.0.0.52.61332        10.0.0.157.56856       ESTABLISHED
tcp4       0      0  129.82.39.50.61449     linuxa1.engr.col.ms-wb ESTABLISHED
tcp6       0      0  2601:282:8001:f9.61446 jd-in-xbc.1e100..5228  ESTABLISHED
tcp4       0      0  10.0.0.52.61444        secure.colostate.https ESTABLISHED
tcp4       0      0  10.0.0.52.61428        10.0.0.85.8009         ESTABLISHED
tcp4       0      0  10.0.0.52.59922        104.46.162.226.https   ESTABLISHED
tcp4       0      0  10.0.0.52.59842        20.44.10.122.https     ESTABLISHED
tcp4       0      0  10.84.125.238.63095    20.189.173.6.https     ESTABLISHED
tcp4       0      0  10.0.0.52.63042        20.189.173.2.https     ESTABLISHED
tcp4       0      0  10.0.0.52.63039        20.42.65.89.https      ESTABLISHED
tcp4       0      0  10.0.0.52.52182        20.42.65.85.https      ESTABLISHED
tcp4       0      0  10.0.0.52.52150        13.69.109.131.https    ESTABLISHED
tcp4       0      0  10.0.0.52.50819        20.50.73.9.https       ESTABLISHED
tcp4       0      0  10.0.0.52.50803        20.189.173.4.https     ESTABLISHED
tcp4       0      0  10.0.0.52.50800        104.46.162.226.https   ESTABLISHED
tcp4       0      0  10.0.0.52.50786        13.69.239.72.https     ESTABLISHED
tcp4       0      0  10.0.0.52.50783        20.189.173.13.https    ESTABLISHED
tcp4       0      0  10.0.0.52.63953        20.189.173.14.https    ESTABLISHED
tcp4       0      0  10.0.0.52.53803        20.189.173.11.https    ESTABLISHED
tcp6       0      0  pablos-macbook-p.1028  fe80::c353:7229:.1025  ESTABLISHED
tcp6       0      0  pablos-macbook-p.1024  fe80::c353:7229:.1024  ESTABLISHED
tcp6       0      0  pablos-macbook-p.black fe80::5f81:44f9:.65172 ESTABLISHED
tcp6       0      0  pablos-macbook-p.1024  fe80::5f81:44f9:.1024  ESTABLISHED
tcp4       0      0  10.0.0.52.61540        20.42.72.131.https     TIME_WAIT
tcp4       0      0  10.0.0.52.61542        51.104.15.253.https    TIME_WAIT
tcp4       0      0  10.0.0.52.61543        51.104.15.253.https    TIME_WAIT
tcp4       0      0  10.0.0.52.54765        ec2-54-189-10-23.443   ESTABLISHED
tcp4       0      0  10.0.0.52.54706        17.57.144.102.443      ESTABLISHED
udp6       0      0  2601:282:8001:f9.61330 ia-in-f102.1e100.https
udp6       0      0  2601:282:8001:f9.54522 den16s08-in-x04..https
udp6       0      0  2601:282:8001:f9.59768 den16s08-in-x04..https
udp6       0      0  2601:282:8001:f9.50022 den08s06-in-x0e..https
udp4       0      0  *.xserveraid           *.*
udp4       0      0  10.0.0.52.56300        *.*
udp4       0      0  *.51644                *.*
udp4       0      0  *.57050                *.*
udp6       0      0  2601:282:8001:f9.54072 den16s05-in-x0e..https
udp4       0      0  *.*                    *.*
udp4       0      0  *.*                    *.*
udp4       0      0  *.*                    *.*
udp4       0      0  *.*                    *.*
udp4       0      0  *.*                    *.*
udp6     761      0  fe80::9861:d4bc:.cap   fe80::51ce:4c9:4.1025
udp4       0      0  *.*                    *.*
udp6     950      0  fe80::6ae2:2520:.cap   fe80::b570:22dd:.1025
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp4       0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp46      0      0  *.mdns                 *.*
udp4       0      0  *.*                    *.*
udp4       0      0  *.*                    *.*
udp4       0      0  *.*                    *.*
udp4       0      0  *.*                    *.*
```

Figure 11: First snippet of netsat

The netstat command shows all the information on network connection, routing tables. The thing that we can really benefit from this is that it allows us to pull up all the information regarding a specified protocol or family. The first thing that netsat will display is the active sockets for each protocol followed by the family which was chosen to display information about. The third list of information it will display is information regarding packet traffic on the configured network interfaces. And the fourth will give more information on the family that was chosen.

Netstat [-options] [family]

Traceroute:

```
Version 1.4a12+Darwin
Usage: traceroute [-adDeFInrSvx] [-A as_server] [-f first_ttl] [-g gateway] [-i iface]
        [-M first_ttl] [-m max_ttl] [-p port] [-P proto] [-q nqueries] [-s src_addr]
        [-t tos] [-w waittime] [-z pausemsecs] host [packetlen]
(base) corona@Pablos-MacBook-Pro ~ % traceroute google.com
traceroute to google.com (142.250.72.46), 64 hops max, 52 byte packets
 1  10.0.0.1 (10.0.0.1)  6.621 ms  5.324 ms  4.555 ms
 2  cm-1-acr01.fortcollins.co.denver.comcast.net (96.120.13.137)  15.432 ms  17.396 ms  14.898 ms
 3  ae-151-1204-rur02.fortcollins.co.denver.comcast.net (96.108.138.157)  16.641 ms  15.130 ms  14.795 ms
 4  ae-2-rur01.fortcollins.co.denver.comcast.net (68.85.89.229)  20.219 ms  15.399 ms  15.247 ms
 5  ae-33-ar01.denver.co.denver.comcast.net (68.86.103.37)  14.973 ms  14.654 ms  15.051 ms
 6  96.216.22.245 (96.216.22.245)  18.428 ms  17.395 ms  16.785 ms
 7  ae-501-ar01.denver.co.denver.comcast.net (96.216.22.130)  24.179 ms  20.584 ms  18.424 ms
 8  be-36011-cs01.1601milehigh.co.ibone.comcast.net (96.110.43.241)  15.665 ms  17.214 ms
    be-36031-cs03.1601milehigh.co.ibone.comcast.net (96.110.43.249)  17.030 ms
 9  be-3102-pe02.910fifteenth.co.ibone.comcast.net (96.110.38.114)  17.122 ms
    be-3202-pe02.910fifteenth.co.ibone.comcast.net (96.110.38.118)  16.657 ms
    be-3402-pe02.910fifteenth.co.ibone.comcast.net (96.110.38.126)  16.500 ms
10  50.248.118.30 (50.248.118.30)  16.873 ms  23.399 ms
    23.30.206.218 (23.30.206.218)  18.412 ms
11  * * *
12  den16s08-in-f14.1e100.net (142.250.72.46)  15.837 ms
    142.251.51.220 (142.251.51.220)  16.997 ms  15.754 ms
```

Figure 12:

This command is used in order to track the route that a packet traveled through in order to reach its host. When going from your machine to the machine that you specified as the host it will take a maximum of 64 steps in order to reach its destination and initially starts as 40 bytes however this can be modified using options.

The following syntax is used in order to run the command

Traceroute [-options] [host]

nslookup:

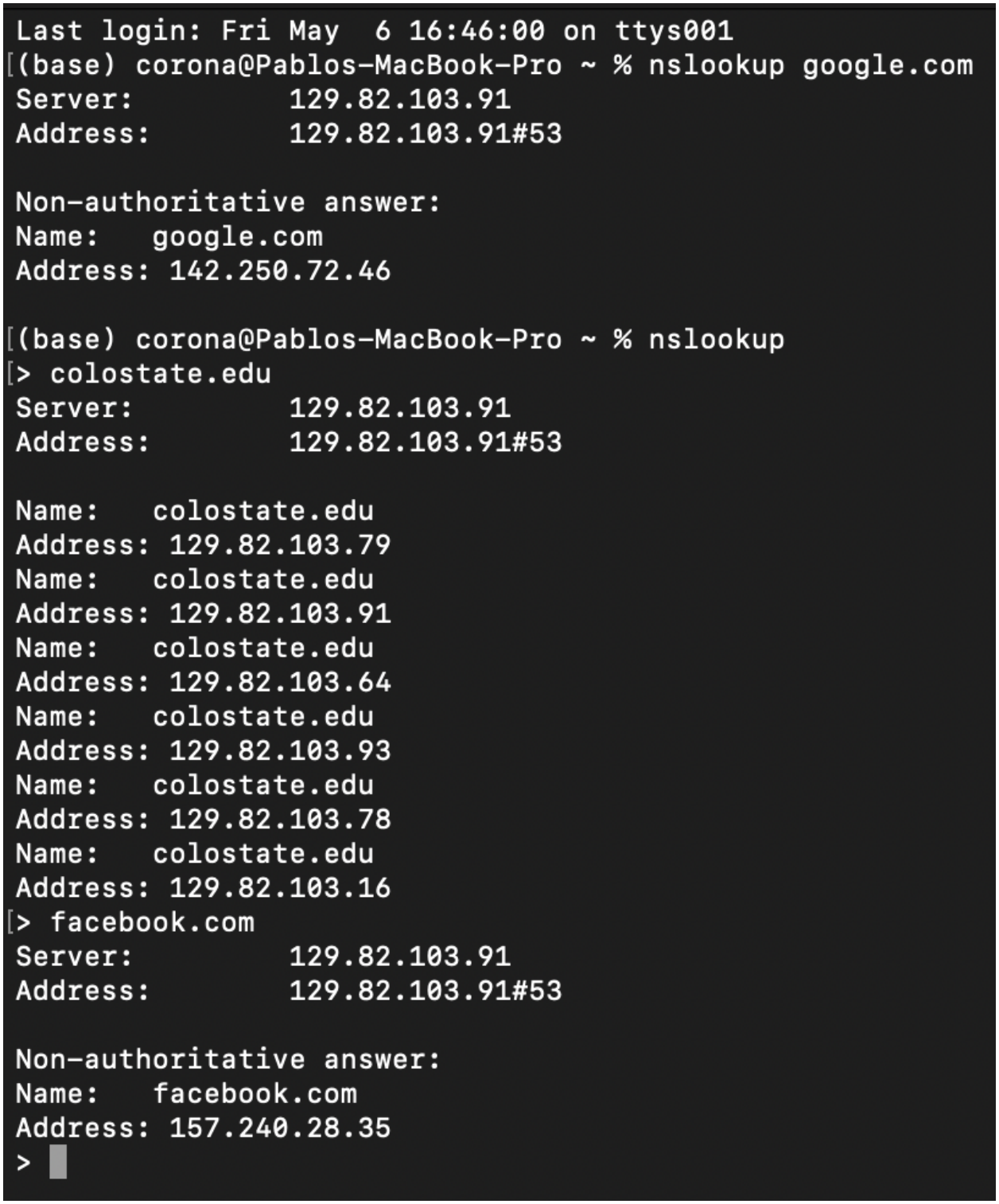

Figure 13: Nslookup running in both interactive mode and non interactive mode.

Nslookup is a program to query Internet domain name servers. There are two modes that can be used interactive, where you are able to query multiple domains, this is activated when no domain is given or

there is an option that was chosen while running. The second mode is non interactive and it is when you specify the specific domain name that you want to query. The following is the syntax for running nslookup

Nslookup [-options] [domain]

Dig:

```
(base) corona@Pablos-MacBook-Pro ~ %
(base) corona@Pablos-MacBook-Pro ~ %
(base) corona@Pablos-MacBook-Pro ~ % dig colostate.edu

; <<>> DiG 9.10.6 <<>> colostate.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53570
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1224
;; QUESTION SECTION:
;colostate.edu.                 IN      A

;; ANSWER SECTION:
colostate.edu.          600     IN      A       129.82.103.93
colostate.edu.          600     IN      A       129.82.103.78
colostate.edu.          600     IN      A       129.82.103.16
colostate.edu.          600     IN      A       129.82.103.79
colostate.edu.          600     IN      A       129.82.103.91
colostate.edu.          600     IN      A       129.82.103.64

;; Query time: 22 msec
;; SERVER: 129.82.103.91#53(129.82.103.91)
;; WHEN: Sat May 07 08:52:54 MDT 2022
;; MSG SIZE  rcvd: 138
```

Figure 14: Dig performing a query on colostate.edu

This resulted in DIG finding the information on the same 6 DNS that we have found previously using different commands such as NSlookup aswell as host. The interesting part about this is the additional amount of information that it will give. The Man page specified that this is often used by DNS architects as the information it provides is in a clear cut format.

Dig [-option] [Name]

Above is the syntax for running this command.

Tcptraceroute:

Unfortunately I was unable to run this command, after trying to run this command on my own machine I saw that it was not installed, because I don't have a linux distro on my computer. I attempted to use the ETD and GENI machines first. After seeing that it was not installed on either of these machines, I attempted to install on both machines however I ran into the issue that i needed administrator privileges to install the command.

Looking at the man page we can see that this would act in a fashion that is similar to traceroute, the one difference is that we would be using a TCP connection rather than having an UDP connection. The following line displays the syntax of the instruction.

Tcptraceroute [-options] [host]

Finger:

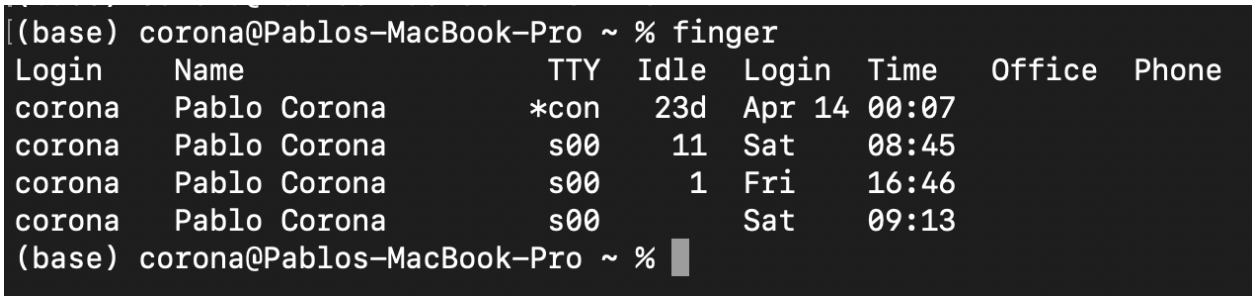

Figure 15: Running finger on personal computer

When running a finger it will display information about the system users and the last time that they logged in. Interestingly enough when i ran this command on my personal computer it returned saying that there were 4 users currently logged into my machine with some of them being idle for a while and some being idle for even 23days. TO me this did not make sense as im the only user on this machine but maybe the way the operating system operates is that it opens multiple sessions at a time?

Finger [-options]

Whois:

```
(base) corona@Pablos-MacBook-Pro ~ % whois google.com
% IANA WHOIS server
% for more information on IANA, visit http://www.iana.org
% This query returned 1 object

refer:        whois.verisign-grs.com

domain:       COM

organisation: VeriSign Global Registry Services
address:      12061 Bluemont Way
address:      Reston Virginia 20190
address:      United States

contact:      administrative
name:         Registry Customer Service
organisation: VeriSign Global Registry Services
address:      12061 Bluemont Way
address:      Reston Virginia 20190
address:      United States
phone:        +1 703 925-6999
fax-no:       +1 703 948 3978
e-mail:       info@verisign-grs.com

contact:      technical
name:         Registry Customer Service
organisation: VeriSign Global Registry Services
address:      12061 Bluemont Way
address:      Reston Virginia 20190
address:      United States
phone:        +1 703 925-6999
fax-no:       +1 703 948 3978
e-mail:       info@verisign-grs.com

nserver:      A.GTLD-SERVERS.NET 192.5.6.30 2001:503:a83e:0:0:0:2:30
nserver:      B.GTLD-SERVERS.NET 192.33.14.30 2001:503:231d:0:0:0:2:30
nserver:      C.GTLD-SERVERS.NET 192.26.92.30 2001:503:83eb:0:0:0:0:30
nserver:      D.GTLD-SERVERS.NET 192.31.80.30 2001:500:856e:0:0:0:0:30
nserver:      E.GTLD-SERVERS.NET 192.12.94.30 2001:502:1ca1:0:0:0:0:30
nserver:      F.GTLD-SERVERS.NET 192.35.51.30 2001:503:d414:0:0:0:0:30
nserver:      G.GTLD-SERVERS.NET 192.42.93.30 2001:503:eea3:0:0:0:0:30
nserver:      H.GTLD-SERVERS.NET 192.54.112.30 2001:502:8cc:0:0:0:0:30
nserver:      I.GTLD-SERVERS.NET 192.43.172.30 2001:503:39c1:0:0:0:0:30
nserver:      J.GTLD-SERVERS.NET 192.48.79.30 2001:502:7094:0:0:0:0:30
nserver:      K.GTLD-SERVERS.NET 192.52.178.30 2001:503:d2d:0:0:0:0:30
nserver:      L.GTLD-SERVERS.NET 192.41.162.30 2001:500:d937:0:0:0:0:30
nserver:      M.GTLD-SERVERS.NET 192.55.83.30 2001:501:b1f9:0:0:0:0:30
ds-rdata:     30909 8 2 E2D3C916F6DEEAC73294E8268FB5885044A833FC5459588F4A9184CFC41A5766

whois:        whois.verisign-grs.com
```

Figure 16: running whois instructioion on the google.com domain name (first part)

When running the whois command it will start by searching and querying the Internet Assigned number authority and follow referrals from whois servers that have more information and specific details about

the query that was searched. Eventually it does find a LOT of information on who that specific domain is, like when the domain was registered by who it was registered and what companies even own this information. This definitely surprised me as i expected this type of information to not be that easy to access and such

Whois [-options] [domain]

Wget:

Figure 17:

The wget command is used in order to download non-interactive files from the internet, supporting http, https, and ftp. The importance of this is the fact that the wget command will still be downloading files without the need for users presence as many modern browsers require the client to be there while downloading which can be a hindrance when downloading large files. There are a plethora of options to choose from on the WGET manual allowing for immediate background downloads or to even write the output into a specified filename.

Wget [-options] [URL]

Curl:

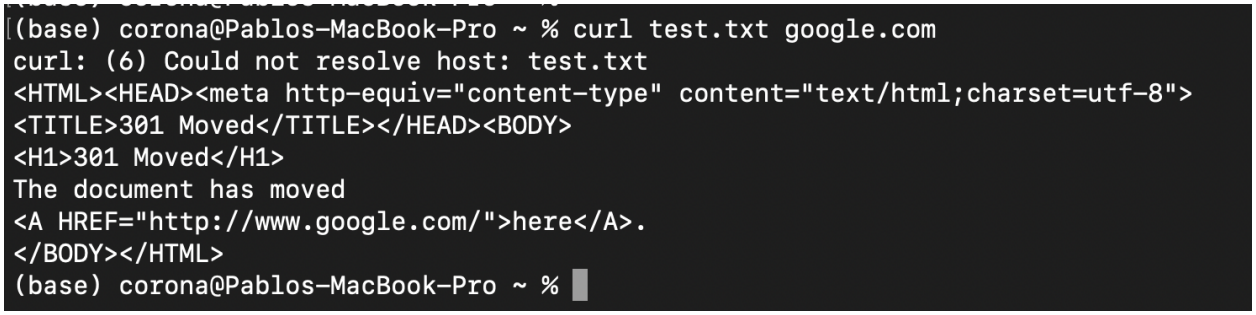

Figure 18: transferring test.txt to google.com using curl

The curl tool is useful for transferring information between different machines as it supports a very large amount of protocols. We are able to transfer into a machine if we know its domain or its IP address. The other thing that I found interesting was that I was still able to transfer a file from my machine to the google.com domain. This makes me wonder what other options there are.

Syntax:
Curl [-options] [-url]

RSH:

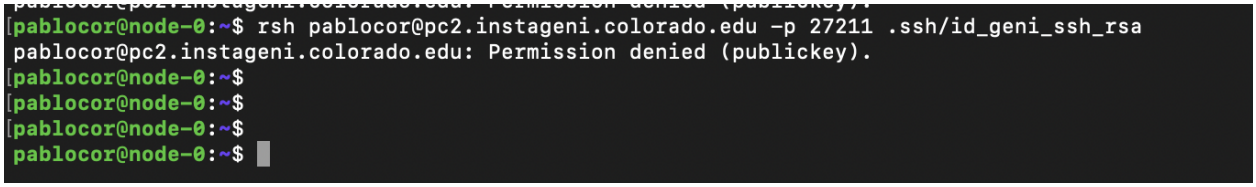

Figure19: attempting to RSH into geni machine.

Here we are attempting to connect to the specific host which in this case would be the second virtual machine on my Geni Slice. Unfortunately I was unable to run this command on either the ETS linux machines and the Geni machines, everytime i would attempt to access either I would have my permission denied for public key. This command has the same functionality as one of our projects has had, it is able to connect to a machine and then provide remote commands. Reading through the man page we can see that this is intended to provide a secure connection when the network is not safe/secure.

Syntax:
Rsh [-options] [host]

FTP:

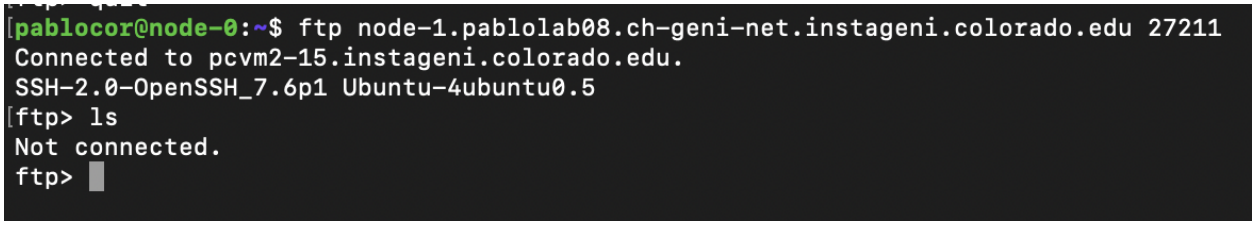

Figure 20: Attempting to use FTP between two geni machines

Ftp is the user interface to the Internet standard File Transfer Protocol. The program allows a user to transfer files to and from a remote network site. When we tried to do this something else that i foun interesting was that when i type help after running the FTP command and connecting, it would prompt you with some standard linux commands. This made me wonder that it may be possible to use the FTP protocol for doing remote commands as well. Though this functionality is not specified anywhere within the FTP man pages.

Syntax:
FTP [host] [port]

SSH:

```
Last login: Sat May  7 10:44:54 on ttys001
(base) corona@Pablos-MacBook-Pro ~ % ssh -i .ssh/id_geni_ssh_rsa pablocor@pc2.instageni.colorado.edu -p 27211
Enter passphrase for key '.ssh/id_geni_ssh_rsa':
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-169-generic x86_64)

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

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
New release '20.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat May  7 10:38:52 2022 from 73.14.183.33
pablocor@node-1:~$
```

Figure 21: Using SSH to connect to geni machines.

The purpose of SSH is to allow users to connect to a remote machine and execute whatever commands they may need to execute. It is intended to provide a secure encrypted connection when there is two untrusted hosts over an insecure network. We have use SSH extensively in this class and in some of my other classes, what i learned from having a quick glance over the contents of the SSH man pages we can see that it is alos capable of providing different types of connection wheter that is IPV4 IPV6 or X11

Syntax:
Ssh [-options] [host] [port #]

Part II:

```
[(base) corona@Pablos-MacBook-Pro part2_portion1 % nc whois.cymru.com 43 < list2
[(base) corona@Pablos-MacBook-Pro part2_portion1 % nc whois.cymru.com 43 -i 1 < list2
AS      | IP               | AS Name
NA      | 10.0.0.1         | NA
AS      | IP               | AS Name
7922    | 96.120.13.137    | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 96.108.138.157   | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 68.85.89.229     | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 68.86.103.37     | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 96.216.22.245    | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 96.216.22.130    | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 96.110.43.253    | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 96.110.43.245    | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 96.110.43.241    | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 96.110.38.122    | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 96.110.38.114    | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 23.30.206.218    | COMCAST-7922, US
AS      | IP               | AS Name
7922    | 50.248.118.30    | COMCAST-7922, US
AS      | IP               | AS Name
15169   | 108.170.254.65   | GOOGLE, US
AS      | IP               | AS Name
15169   | 108.170.254.81   | GOOGLE, US
AS      | IP               | AS Name
15169   | 142.251.48.5     | GOOGLE, US
AS      | IP               | AS Name
15169   | 172.253.75.176   | GOOGLE, US
AS      | IP               | AS Name
15169   | 108.170.252.203  | GOOGLE, US
AS      | IP               | AS Name
15169   | 142.251.48.5     | GOOGLE, US
AS      | IP               | AS Name
15169   | 142.250.72.68    | GOOGLE, US
AS      | IP               | AS Name
15169   | 108.170.254.81   | GOOGLE, US
AS      | IP               | AS Name
15169   | 108.170.254.65   | GOOGLE, US
(base) corona@Pablos-MacBook-Pro part2_portion1 %
```

Figure 22: running NC on multiple targets.

96.120.13.137
OrgName: Comcast Cable Communications, LLC
OrgId: CCCS
Address: 1800 Bishops Gate Blvd

City: Mt Laurel
StateProv: NJ
PostalCode: 08054
Country: US
RegDate: 2001-09-18
Updated: 2020-11-18
Ref: https://rdap.arin.net/registry/entity/CCCS

50.248.118.30
OrgName: Comcast Cable Communications, LLC
OrgId: CCCS
Address: 1800 Bishops Gate Blvd
City: Mt Laurel
StateProv: NJ
PostalCode: 08054
Country: US
RegDate: 2001-09-18
Updated: 2020-11-18
Ref: https://rdap.arin.net/registry/entity/CCCS

68.86.103.37
OrgName: Comcast Cable Communications, LLC
OrgId: CCCS
Address: 1800 Bishops Gate Blvd
City: Mt Laurel
StateProv: NJ
PostalCode: 08054
Country: US
RegDate: 2001-09-18
Updated: 2020-11-18
Ref: https://rdap.arin.net/registry/entity/CCCS

108.170.254.65
OrgName: Google LLC
OrgId: GOGL
Address: 1600 Amphitheatre Parkway
City: Mountain View
StateProv: CA
PostalCode: 94043
Country: US
RegDate: 2000-03-30
Updated: 2019-10-31

172.253.75.176
OrgName: Google LLC
OrgId: GOGL
Address: 1600 Amphitheatre Parkway
City: Mountain View
StateProv: CA
PostalCode: 94043
Country: US
RegDate: 2000-03-30
Updated: 2019-10-31

108.170.254.65
OrgName: Google LLC

OrgId: GOGL
Address: 1600 Amphitheatre Parkway
City: Mountain View
StateProv: CA
PostalCode: 94043
Country: US
RegDate: 2000-03-30
Updated: 2019-10-31

Interestingly enough when i ran this test my route was only through two addresses and what seemed to be many different machines at the same address, most likely being routed to the correct address and location.

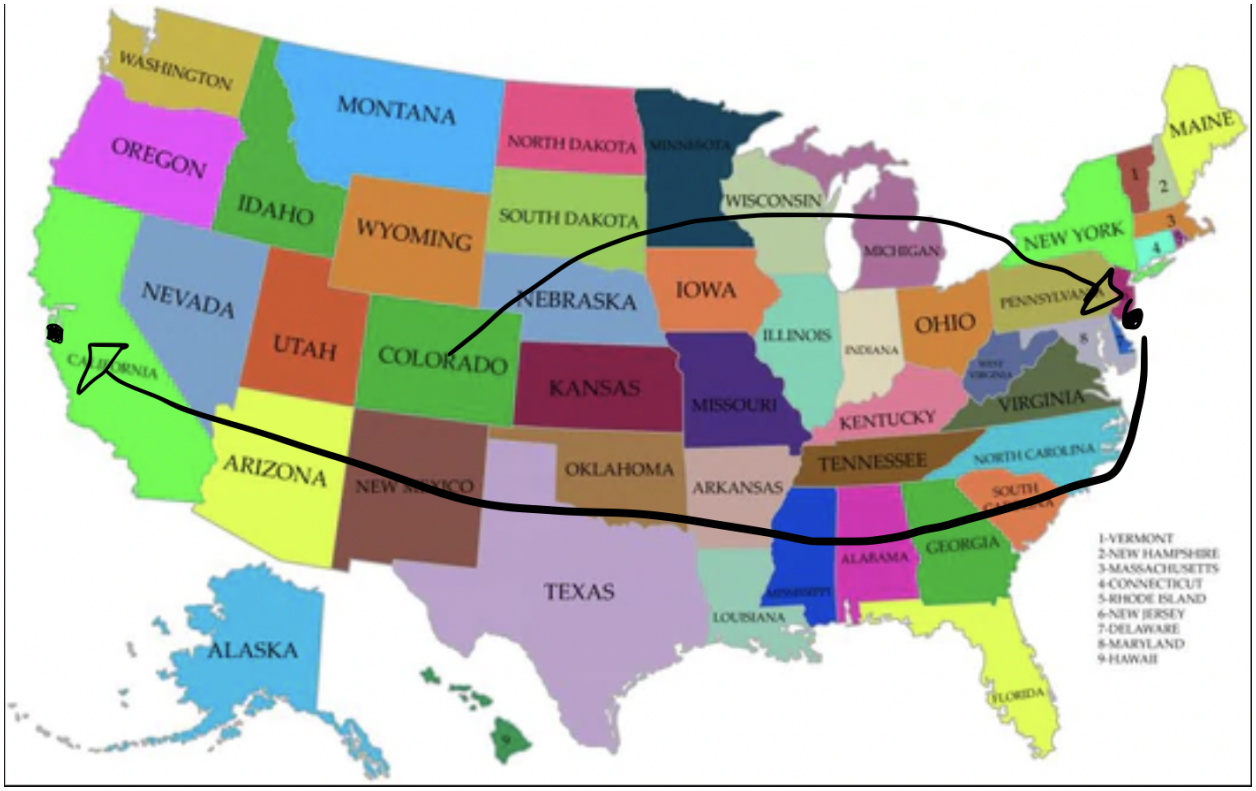

Figure 23: Path the packet traveled through to reach google.com

Part III

```
[pcorona@linuxa1 nmap-7.60]$ ./nmap -A -P0 -T4 192.82.20.50

Starting Nmap 7.60 ( https://nmap.org ) at 2022-05-07 15:19 MDT
Warning: 192.82.20.50 giving up on port because retransmission cap hit (6).
Nmap scan report for 192.82.20.50
Host is up (0.044s latency).
All 1000 scanned ports on 192.82.20.50 are closed (971) or filtered (29)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 83.55 seconds
[pcorona@linuxa1 nmap-7.60]$ ./nmap -A -P0 -T4 192.82.20.167

Starting Nmap 7.60 ( https://nmap.org ) at 2022-05-07 15:21 MDT
Nmap scan report for 192.82.20.167
Host is up (0.041s latency).
Not shown: 999 filtered ports
PORT     STATE  SERVICE VERSION
5060/tcp closed sip

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 129.42 seconds
[pcorona@linuxa1 nmap-7.60]$
```

Figure 24: Running nmap on linux 1 machine

A lof the connection that we were performing a host scan on contained very many closed ports, or they were filtered. There were no active and open ports for either of the two.

Part IV

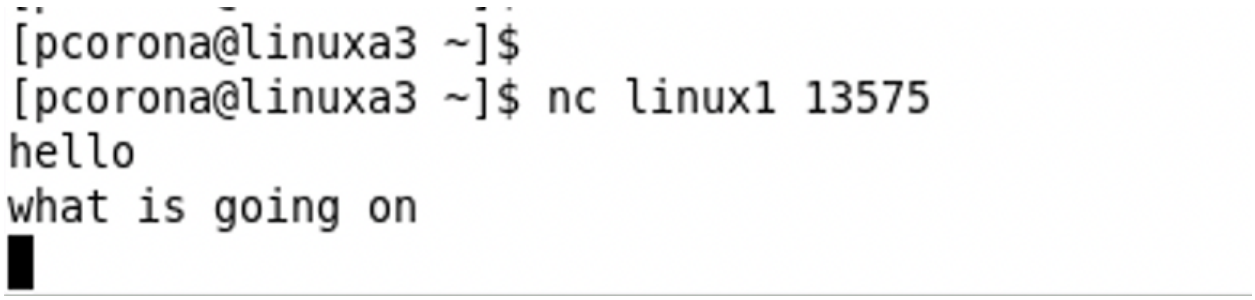

Figure 25: Connecting to Linux1 from linux3

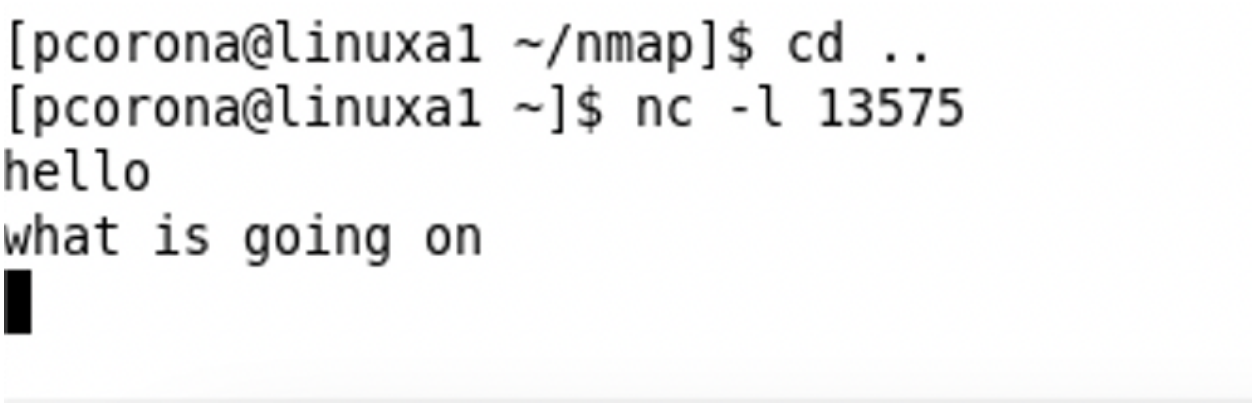

Figure 26: setting up linux1 to listening mode.

Here we are able to see the NC is able to work as a simple method to communicate with. In this particular method when we are setting NC to listen mode we open TCP server to listen and wait for a connection from another device.

```
[pcorona@linuxa1 ~]$ cat send_file | nc linux3 13575
[pcorona@linuxa1 ~]$ gedit send_file &
[1] 36278
[pcorona@linuxa1 ~]$ cat send_file | nc linux3 13575
[1]  + Done                          gedit send_file
[pcorona@linuxa1 ~]$
```

Figure 27: Creating a file on Linux1 and sending to Linux3 after modifying the file.

```
[pcorona@linuxa3 ~/test]$ nc -l 13575 > recv_file
[pcorona@linuxa3 ~/test]$ ls
recv_file
[pcorona@linuxa3 ~/test]$ cat recv_file
Ebter a masters
[pcorona@linuxa3 ~/test]$
```

Figure 28: Receiving a File and checking to see if file transfer was successful

```
[pcorona@linuxa3 ~/test]$ nc linux1 13575
ls
```

Figure 29: COnnecting to linux 1 and Sending the ls command

```
[pcorona@linuxa1 ~]$
[pcorona@linuxa1 ~]$ nc -l 13575 | sh
Ansoft          CDS.log.1               default.dwf3work  ECE 202  ECE HW 6.pdf  MDS00005.jpg
B111Lab (new)   CDS.log.1.cdslck        Default.rdp       ECE 251  ettcp         Music
B111Lab (new2)  CDS.log.2               Desktop           ECE331   IOzone        netperf_install
B111Lab (old)   CDS.log.2.cdslck        desktop.ini       ECE451   LabVIEW Data  nmap
benchmarks      CDS.log.3               Documents         ECE452   MATLAB        Official B11 website.zip
cadence         CDS.log.cdslck          Downloads         ECE456   MDS00001.jpg  panic.log.linuxa5.engr.colostate.edu
CDS.log         Custom Office Templates ECE 102           ECE571   MDS00002.jpg  panic.log.linuxa5.engr.colostate.edu
```

Figure 30: Creating a shell for linux1 and running LS from linux3

```
[pcorona@linuxa3 ~/test]$ nc -zv node-1 1-100
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Could not resolve hostname "node-1": Name or service not known. QUITTING.
[pcorona@linuxa3 ~/test]$ nc -zv linux1 1-100
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: No route to host.
[pcorona@linuxa3 ~/test]$
```

Figure 31: Scanning node-1 and linux1

Her we can see that when were using NC for port scanning that it wasn't able to find any rout from linux3 to linux1 from the CSU linux machines.