CS530 F23 Week 2 Lab
Prof. Feng

Amelia Miner
10/8/23

TABLE OF CONTENTS:

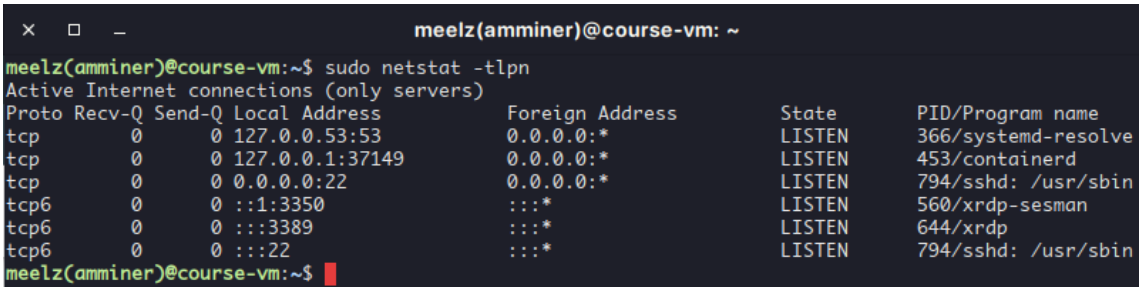
Amelia Miner
10/8/23

I. Lab 2.1 - TCP, HTTP

1. TCP #1: Sockets

- Using ChatGPT, find a single command and its command-line flags that, when executed, lists all TCP sockets in a LISTEN state on an IPv4 address, showing the program that is using it.
    - Take a screenshot of the prompt and the command that ChatGPT generates

    - Run the command using sudo and take a screenshot of the output to include in your lab notebook.

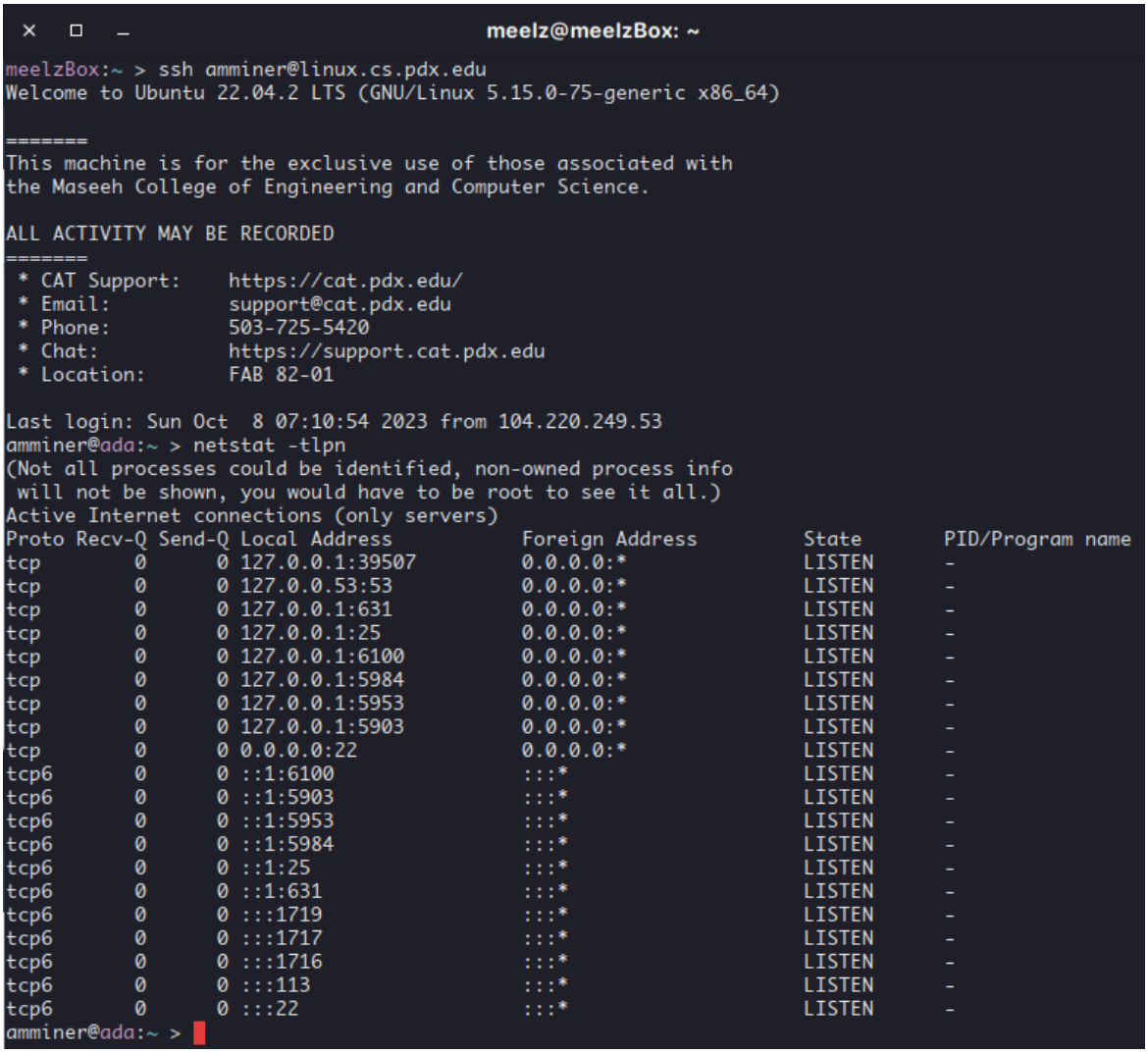


    - List a service that can be contacted from any interface on the machine. List a service that can only be contacted by local processes.

Amelia Miner

sshd can be can be contacted from any interface on the machine. containerd can only be contacted by local processes.

- **Login to linux.cs.pdx.edu**
  - Run the command again, but do not use sudo as this is a machine managed by CAT. Include a screenshot of the output.



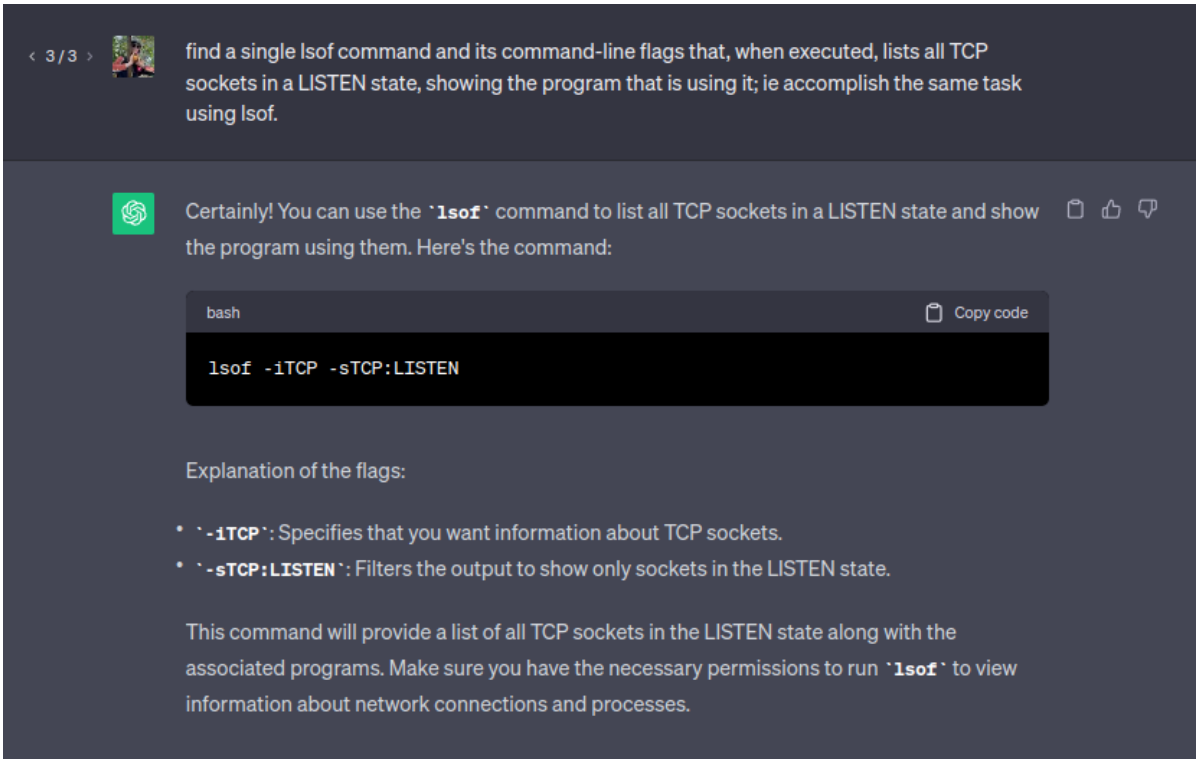
  - List the services this machine provides for external access.

    It looks like only ssh is available to external connections on linux.cs.pdx.edu.

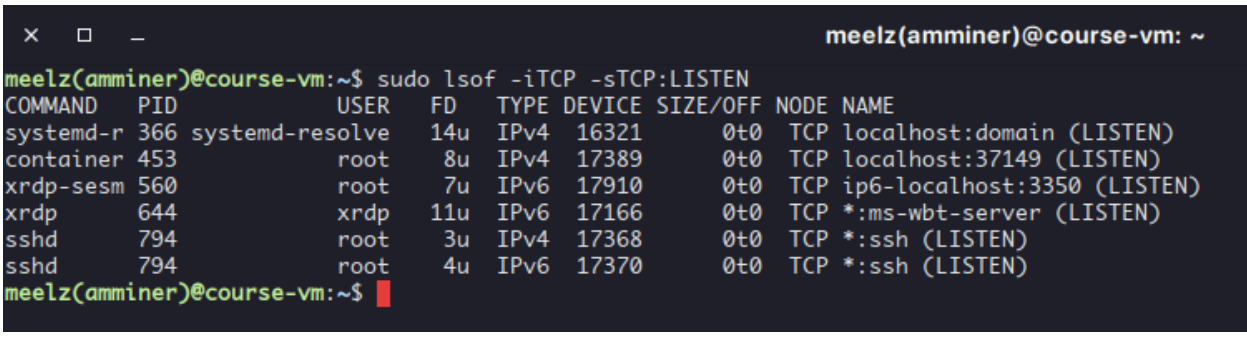
Amelia Miner
10/8/23

2. LSOF

- Using ChatGPT, find a single lsof command and its command-line flags that, when executed, lists all TCP sockets in a LISTEN state on an IPv4 address, showing the program that is using it. Note that you can leverage the conversation in the previous step and simply ask ChatGPT to repeat the task using lsof.
    - Take a screenshot of the prompt and the command that ChatGPT generates.



    - Run the command using sudo and take a screenshot of the output to include in your lab notebook.



Amelia Miner

3. TCP #2: Throughput

- VMs instantiated:

  NAME: vm-europe-west1-d
  INTERNAL_IP: 10.132.0.2
  EXTERNAL_IP: 35.195.100.133

  NAME: vm-us-west1-b
  INTERNAL_IP: 10.138.0.6
  EXTERNAL_IP: 34.168.52.191

  NAME: vm-us-east1-b
  INTERNAL_IP: 10.142.0.3
  EXTERNAL_IP: 34.74.3.81

  NAME: vm-australia-southeast1-b
  INTERNAL_IP: 10.152.0.2
  EXTERNAL_IP: 35.189.16.39

Amelia Miner

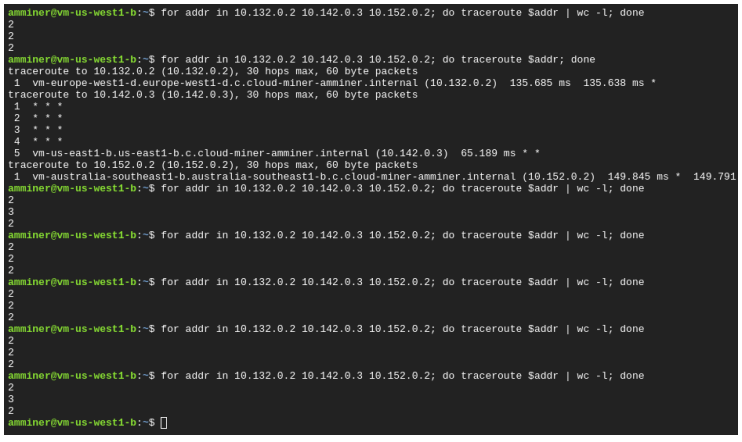
4. - iperf

- On each foreign machine, run `sudo iperf -s -p 80`.
- On the local machine, run `iperf -c <IP address> -p 80` for each foreign machine's internal IP.
- Show a screenshot of the measured bandwidth available between your us-west1-b VM and each of the other Compute Engine VMs. Explain the relative differences (or lack thereof) in your results.

```
amminer@vm-us-west1-b:~$ for addr in 10.132.0.2 10.142.0.3 10.152.0.2
> do
> iperf -c $addr -p 80 | tee $addr.txt
> done
------------------------------------------------------------
Client connecting to 10.132.0.2, TCP port 80
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  1] local 10.138.0.6 port 46862 connected with 10.132.0.2 port 80
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.2068 sec   185 MBytes   152 Mbits/sec
------------------------------------------------------------
Client connecting to 10.142.0.3, TCP port 80
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  1] local 10.138.0.6 port 47144 connected with 10.142.0.3 port 80
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0767 sec   435 MBytes   362 Mbits/sec
------------------------------------------------------------
Client connecting to 10.152.0.2, TCP port 80
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  1] local 10.138.0.6 port 38712 connected with 10.152.0.2 port 80
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.2228 sec   173 MBytes   142 Mbits/sec
amminer@vm-us-west1-b:~$ ls
10.132.0.2.txt  10.142.0.3.txt  10.152.0.2.txt
amminer@vm-us-west1-b:~$
```

(in the same order as above: eu-west, us-east, aus-southeast).

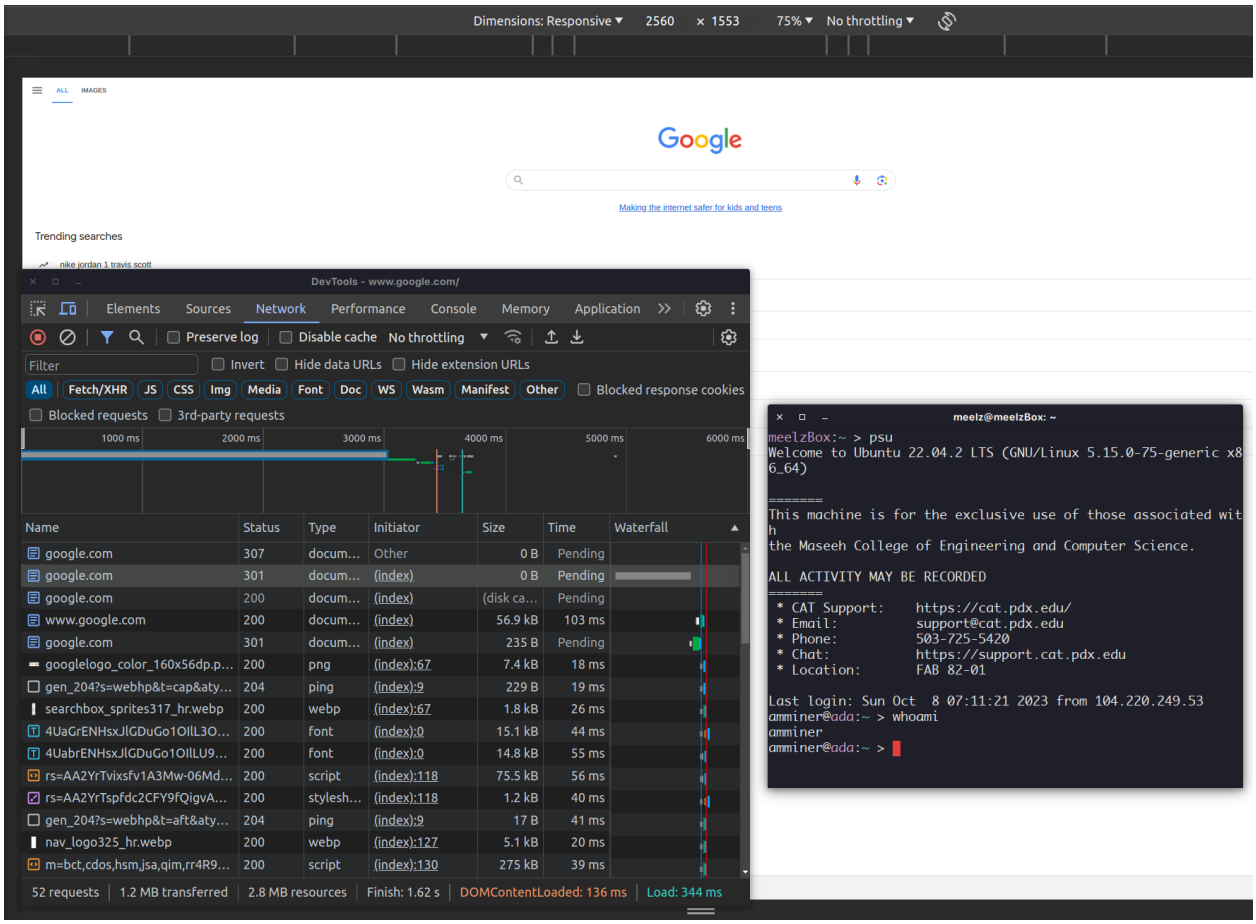
Bandwidth values correspond roughly with discrepancies I would expect based on physical distances between the machines. Interestingly, when I run traceroute to these machines, discrepancies in the number of hops are minimal, presumably because I'm routing over gcloud's internal network which creates a sort of virtual data link over the actual routers that underlie it, like what we were talking about last week in lecture?



Amelia Miner
10/8/23

5. HTTP #3: Requests

- Using chrome,visit http://google.com in incognito mode with quic/http3 enabled. Take a screenshot of the initial 3 requests that the browser makes:



- For each of the initial 3 requests:
  - What is the URL being requested?

    http://google.com,

    https://google.com,

    and http://google.com

Amelia Miner
10/8/23

- Explain the HTTP status code that is returned and what the code indicates

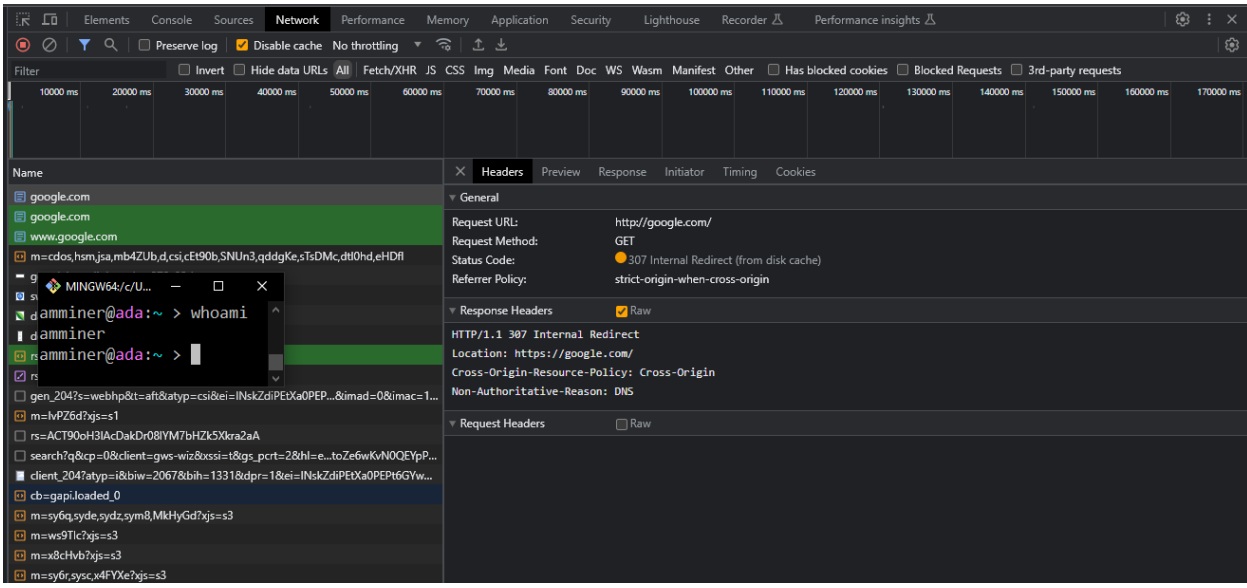
  307-temporary redirect. The response includes a new address to which the resource has been "temporarily" moved.

  301-moved permanently. The response includes a new address to which the resource has been "permanently" moved.

  200-ok. The request was successful - here's those bits you asked for.

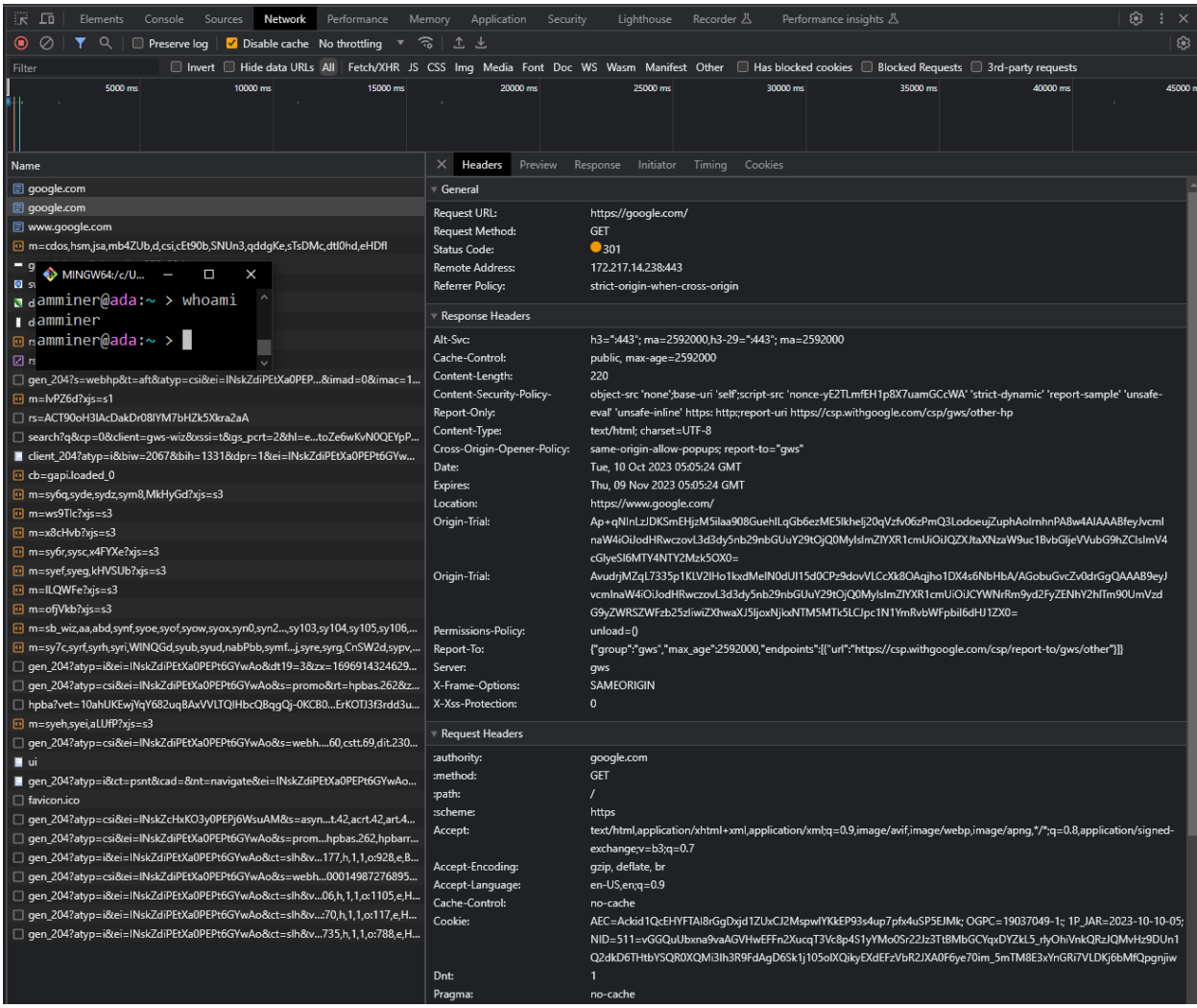
- Take a screenshot indicating the version of the HTTP protocol that is used for each request. (Hint: look at the response status line and alt-svc: HTTP response headers indicating HTTP/2 or HTTP/3).

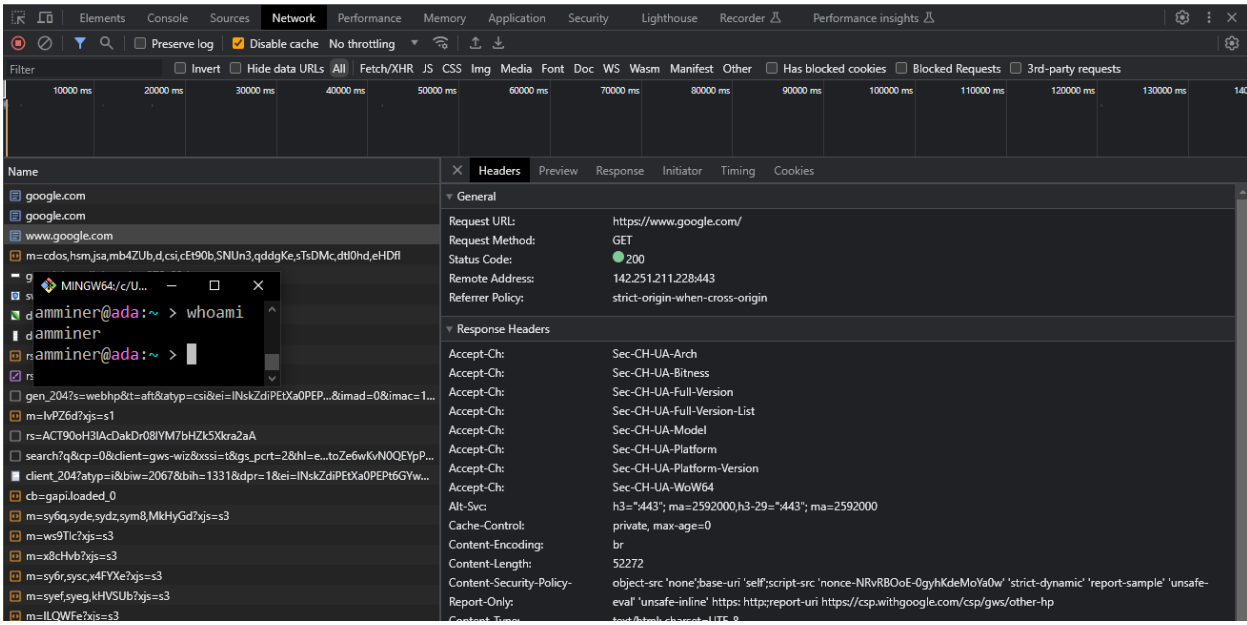
  My first request shows http/1.1. The other requests aren't showing any version info in their headers. alt-svc in the responses doesn't mean anything to me - what's wrong with my setup? Chrome also occasionally mixes up the wrong status code/name combinations, like 200-internal redirect, etc.



CS530 F23 Week 2 Lab
Prof. Feng

Amelia Miner
10/8/23





CS530 F23 Week 2 Lab
Prof. Feng

Amelia Miner
10/8/23

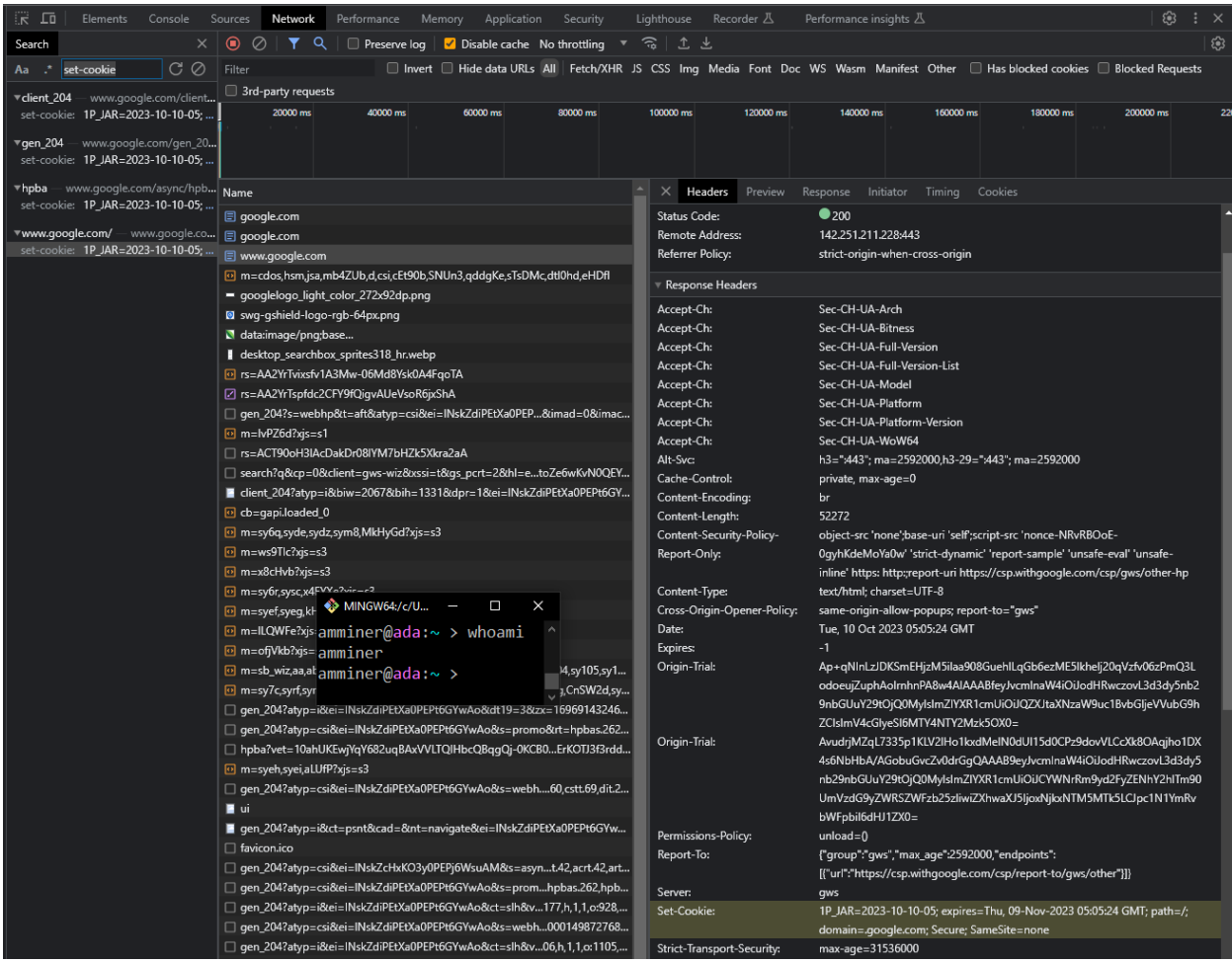
- Find the Location: response header for the two redirections.
    - What URL does the first redirection send the browser to?

      https://google.com

    - What URL does the second redirection send the browser to?

      https://www.google.com/

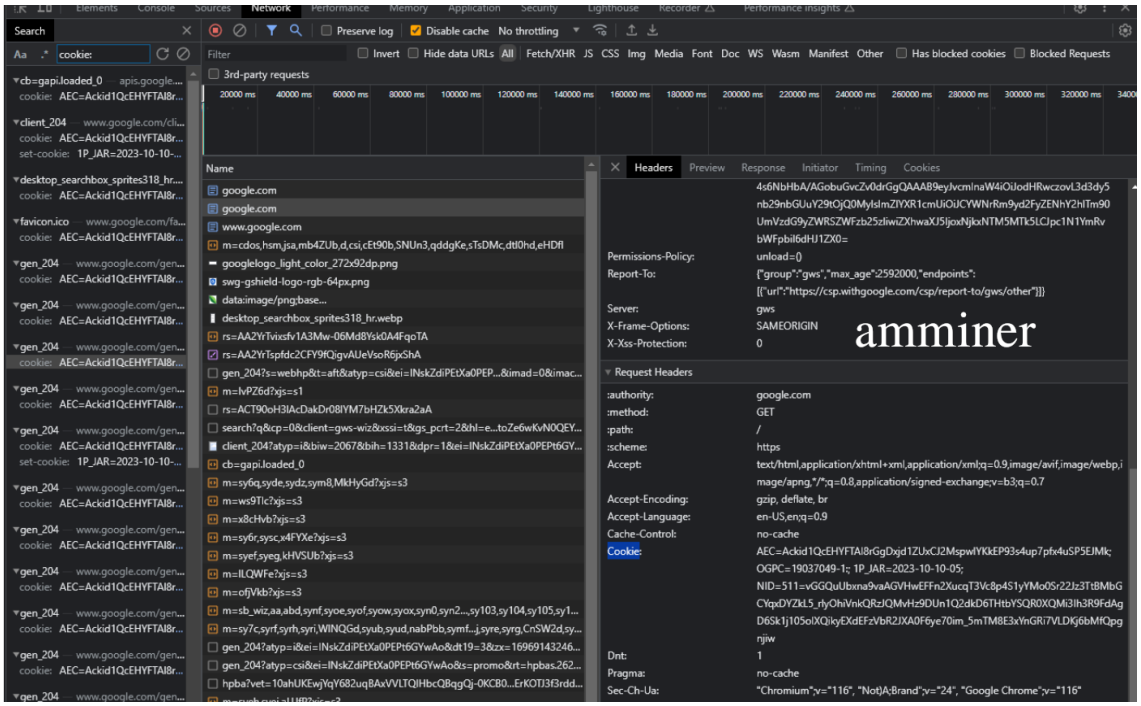
- Examine the HTTP request and response headers for cookies throughout the requests.
    - Take a screenshot of when cookies are set via Set-Cookie:
      In the third request:

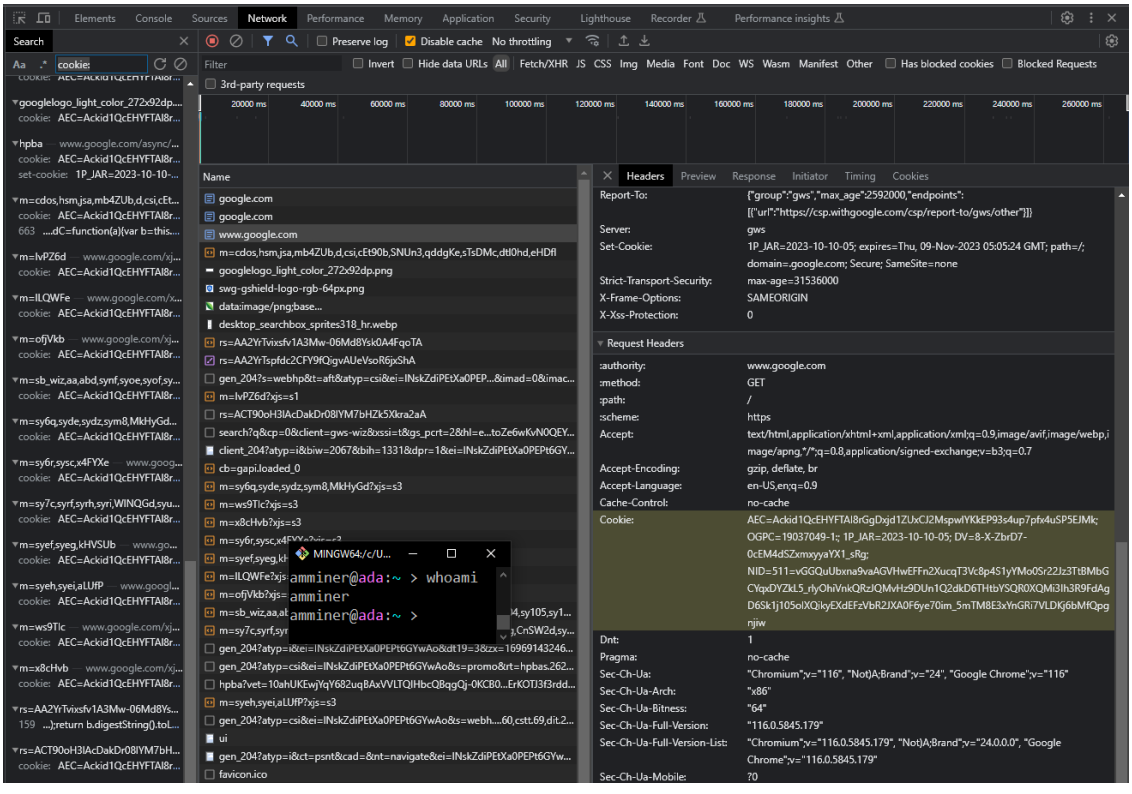


CS530 F23 Week 2 Lab
Prof. Feng

Amelia Miner
10/8/23

- Take a screenshot of when cookies are attached via Cookie:
  In the second and third requests:



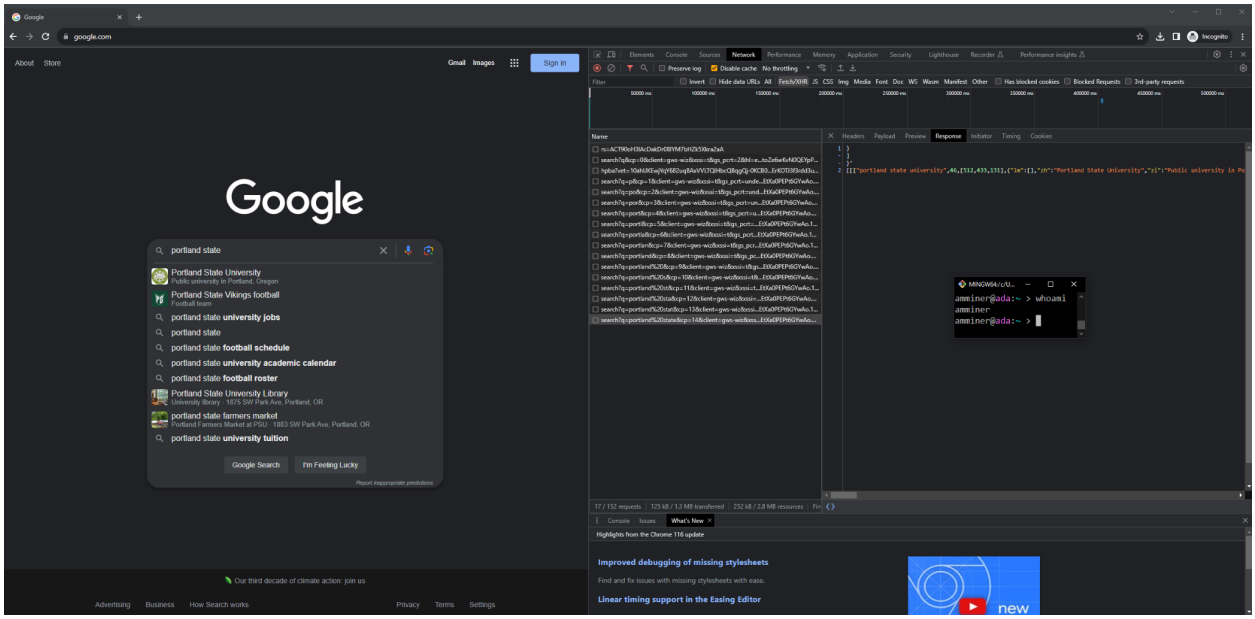


CS530 F23 Week 2 Lab
Prof. Feng

Amelia Miner
10/8/23

6. - Asynchronous HTTP requests

- Show the asynchronous HTTP requests made by typing in the search bar:

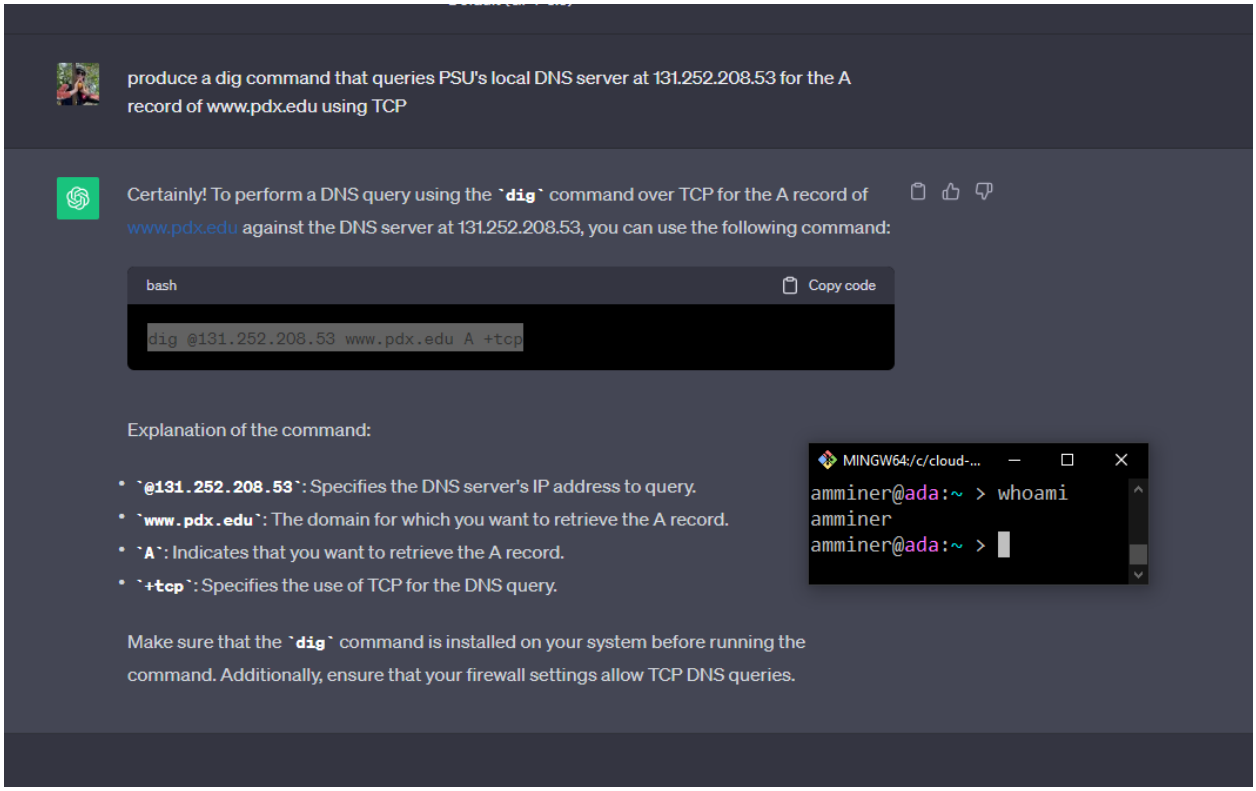


Amelia Miner
10/8/23

II. Lab 2.2 - TODO

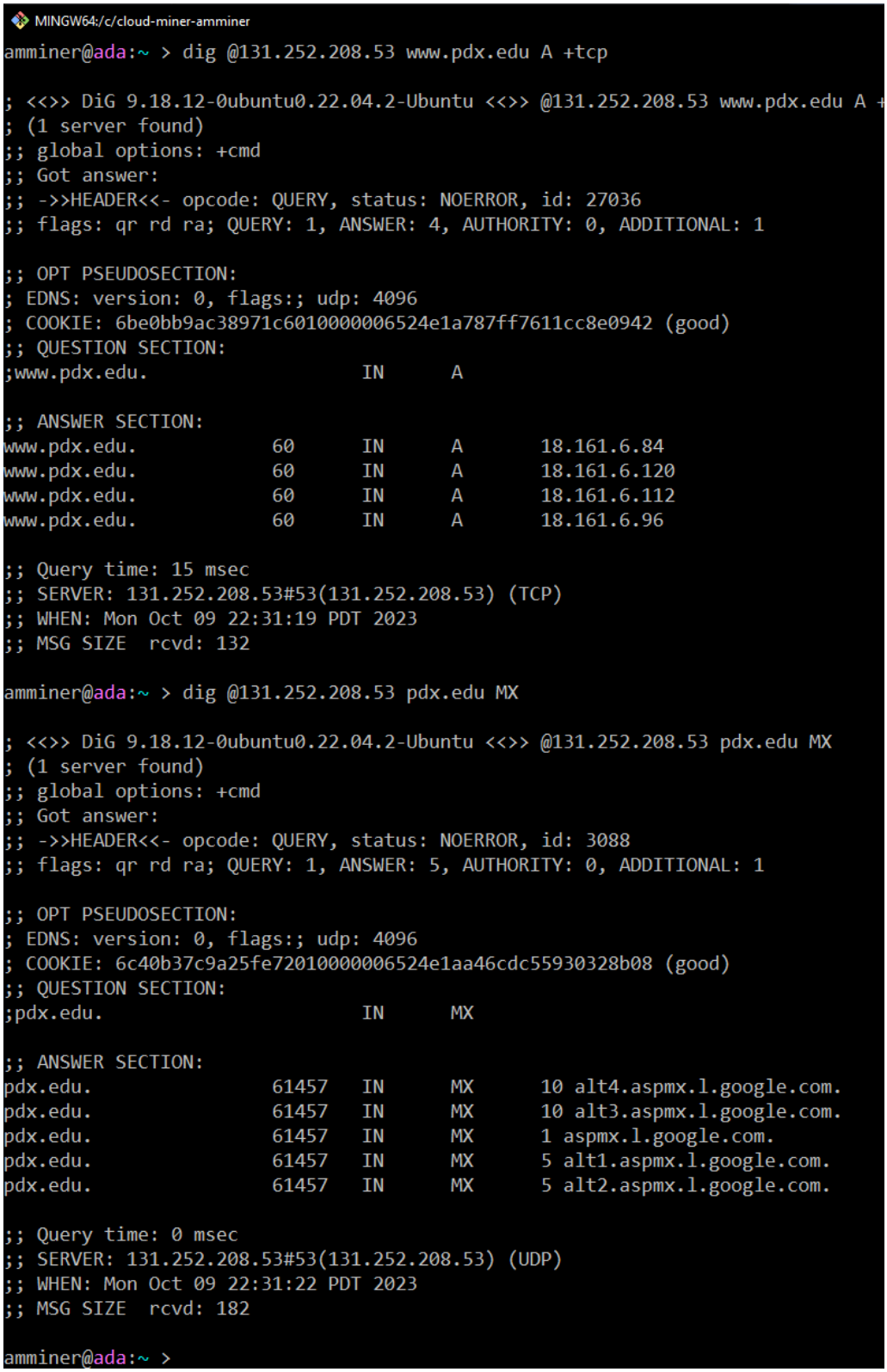
1. DNS reconnaissance #1 (dig)

- Using ChatGPT, produce a dig command that queries PSU's local DNS server at 131.252.208.53 for the A record of www.pdx.edu using TCP.



Amelia Miner

- Run the command to find the record. Then, use dig to do the same for the MX record of pdx.edu.



```
MINGW64:/c/cloud-miner-amminer
amminer@ada:~ > dig @131.252.208.53 www.pdx.edu A +tcp

; <<>> DiG 9.18.12-0ubuntu0.22.04.2-Ubuntu <<>> @131.252.208.53 www.pdx.edu A +
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27036
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 6be0bb9ac38971c6010000006524e1a787ff7611cc8e0942 (good)
;; QUESTION SECTION:
;www.pdx.edu.                   IN      A

;; ANSWER SECTION:
www.pdx.edu.            60      IN      A       18.161.6.84
www.pdx.edu.            60      IN      A       18.161.6.120
www.pdx.edu.            60      IN      A       18.161.6.112
www.pdx.edu.            60      IN      A       18.161.6.96

;; Query time: 15 msec
;; SERVER: 131.252.208.53#53(131.252.208.53) (TCP)
;; WHEN: Mon Oct 09 22:31:19 PDT 2023
;; MSG SIZE  rcvd: 132

amminer@ada:~ > dig @131.252.208.53 pdx.edu MX

; <<>> DiG 9.18.12-0ubuntu0.22.04.2-Ubuntu <<>> @131.252.208.53 pdx.edu MX
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 3088
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 6c40b37c9a25fe72010000006524e1aa46cdc55930328b08 (good)
;; QUESTION SECTION:
;pdx.edu.                       IN      MX

;; ANSWER SECTION:
pdx.edu.                61457   IN      MX      10 alt4.aspmx.l.google.com.
pdx.edu.                61457   IN      MX      10 alt3.aspmx.l.google.com.
pdx.edu.                61457   IN      MX      1 aspmx.l.google.com.
pdx.edu.                61457   IN      MX      5 alt1.aspmx.l.google.com.
pdx.edu.                61457   IN      MX      5 alt2.aspmx.l.google.com.

;; Query time: 0 msec
;; SERVER: 131.252.208.53#53(131.252.208.53) (UDP)
;; WHEN: Mon Oct 09 22:31:22 PDT 2023
;; MSG SIZE  rcvd: 182

amminer@ada:~ >
```

Amelia Miner
10/8/23

Using the IP addresses contained in these records, utilize IP address information services at https://www.iplocation.net/ to answer the following questions

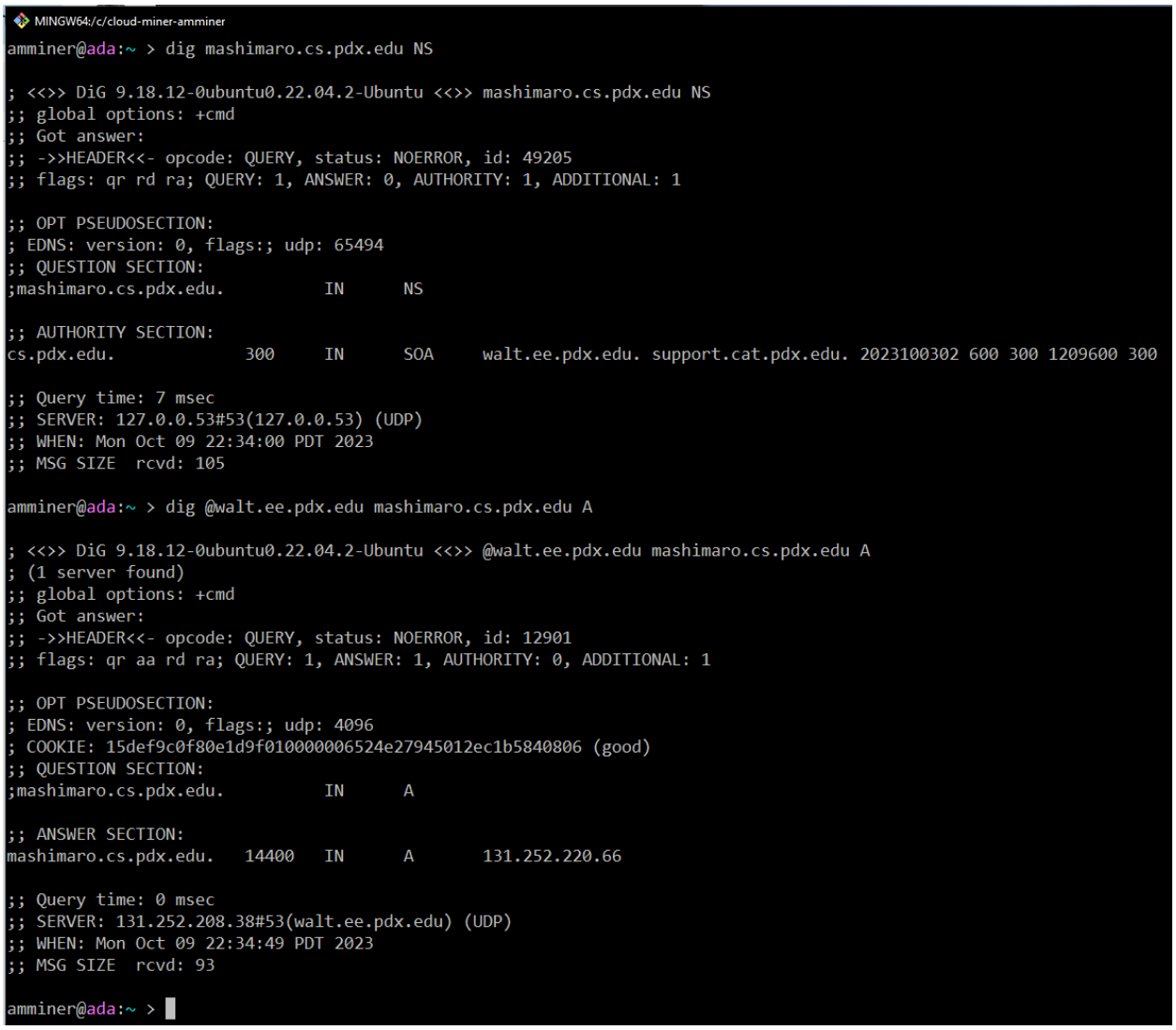
- What cloud provider hosts the web site for www.pdx.edu?

  Amazon.

- What cloud provider handles mail for pdx.edu?

  Google.

- Use `dig` to find the authoritative server (`NS` record type, AUTHORITY section response) for `mashimaro.cs.pdx.edu` and then query that server for the A record of `mashimaro.cs.pdx.edu`.



Amelia Miner

2. Iterative DNS lookups

- dig told me the f root server is named f.root-servers.net.
- dig f.root-servers.net told me the f root server's IP address is 192.5.5.241. Now we can really start. I'll resolve example.com:
- dig @192.5.5.241 example.com NS +tcp points us toward several .com TLD servers and conveniently includes their A records in addition to the requested NS records. I'll go with f.gtld-servers.net. 172800 IN A 192.35.51.30.
- dig @192.35.51.30 example.com NS +tcp points us down another layer to a server owned by IANA, as we would expect for example.com. Annoyingly we don't get free A records back this time. An additional query to the same TLD server for the A record associated with a.iana-servers.net yields 199.43.135.53 - this looks to be the authoritative server.
- Finally:

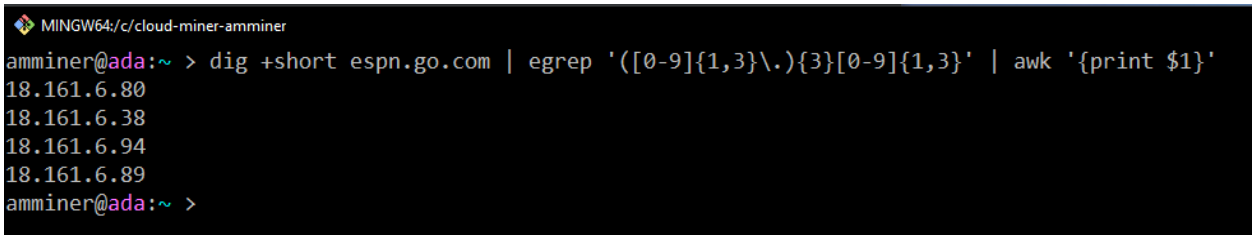
```
amminer@ada:~ > dig @199.43.135.53 example.com A +tcp

; <<>> DiG 9.18.12-0ubuntu0.22.04.2-Ubuntu <<>> @199.43.135.53 example.com A +tcp
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27290
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;example.com.                   IN      A

;; ANSWER SECTION:
example.com.            86400   IN      A       93.184.216.34

;; Query time: 27 msec
;; SERVER: 199.43.135.53#53(199.43.135.53) (TCP)
;; WHEN: Mon Oct 09 22:47:32 PDT 2023
;; MSG SIZE  rcvd: 56

amminer@ada:~ >
```

3. Reverse DNS lookups

- Using ChatGPT, produce a single command line with commands `dig`, `egrep`, and `awk`, to list all IPv4 addresses that `espn.go.com` points to.

Amelia Miner
10/8/23

- Run the command

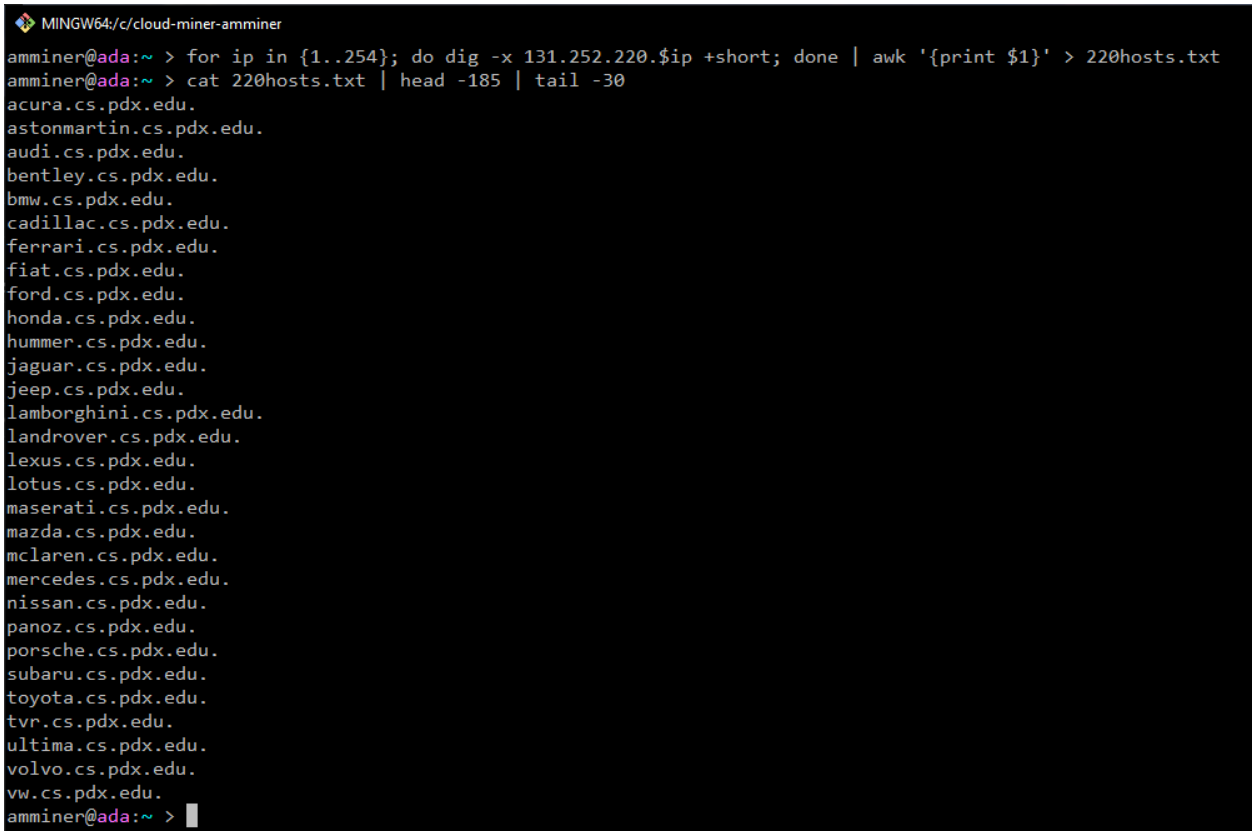


- Find the flag for dig that allows one to perform reverse lookups on IPv4 addresses. Take that list of addresses and create a single for loop in the shell that iterates over the list and performs a reverse lookup of each IP address to find each address's associated DNS name. Pipe output of the loop to egrep and awk so that the output consists only of the DNS names.

Amelia Miner
10/8/23

4. Host enumeration

- Using a for loop, perform a reverse DNS lookup for each IP address on the 131.252.220.0/24 subnet. Note that some addresses on the subnet do not have names bound to them and will not return a record. Take the output of the loop and pipe it to egrep and awk to list just the names of the hosts, then redirect the final output to a file called 220hosts.txt, using the > character to perform output redirection to a file.
- Within the range of hosts is a set of car manufacturer names. Using the head and tail commands, craft a command in the format below that returns their names.
  `cat 220hosts.txt | head -<number_of_lines_1> | tail -<number_of_lines_2>`



One could automate this easily if the first and last manufacturers are known.

5. Geographic DNS #2

- Visit https://www.iplocation.net/ and look up the geographical location of the following DNS servers: 131.252.208.53 and 198.82.247.66. What geographic locations do ipinfo.io and DB-IP return?

  ipinfo.io says Portland, OR. DB-IP is not on the results page.

Amelia Miner

- Then, using dig, resolve www.google.com from each of the DNS servers. As Google forward deploys its web content close to its users, its name will resolve differently in different geographic locations.

  ```
  dig @<DNS_server_IP> www.google.com
  ```

- Record one address for www.google.com from each result for your lab notebook.

  @131.252.208.53: www.google.com resolves to 142.251.215.228.

  @198.82.247.66: www.google.com resolves to 142.251.163.106.

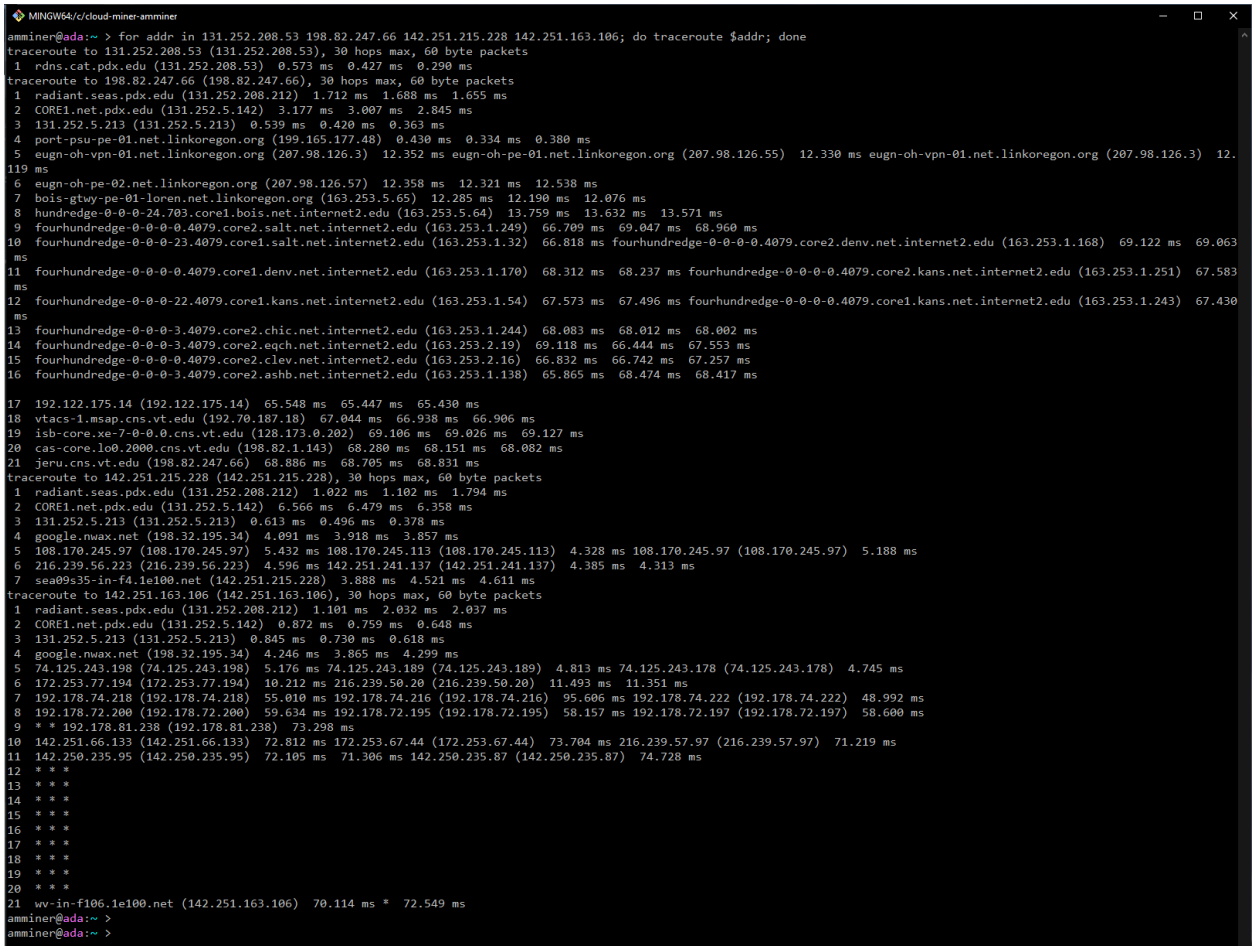
- What are the geographic coordinates of each DNS server and the IP address it resolves for www.google.com?

  142.251.215.228 looks to be in Seattle.

  142.251.163.106 looks to be in Mountain View, CA, but ipinfo.io seems to think it's in Ashburn, VA. My inclination is to expect google to obfuscate things through their mountain view facilities, so I would guess Ashburn, final answer…

- traceroute the DNS servers and the IPs you found for google.com:

Amelia Miner
10/8/23



6. Wireshark Lab #3

- RDP into your gcloud linux VM. In a terminal, using commands from prior labs, find the addresses and interfaces on the VM. Make a note of:
    - The IP address of the VM

      10.138.0.2

    - The name of the local virtual ethernet interface

      eth0

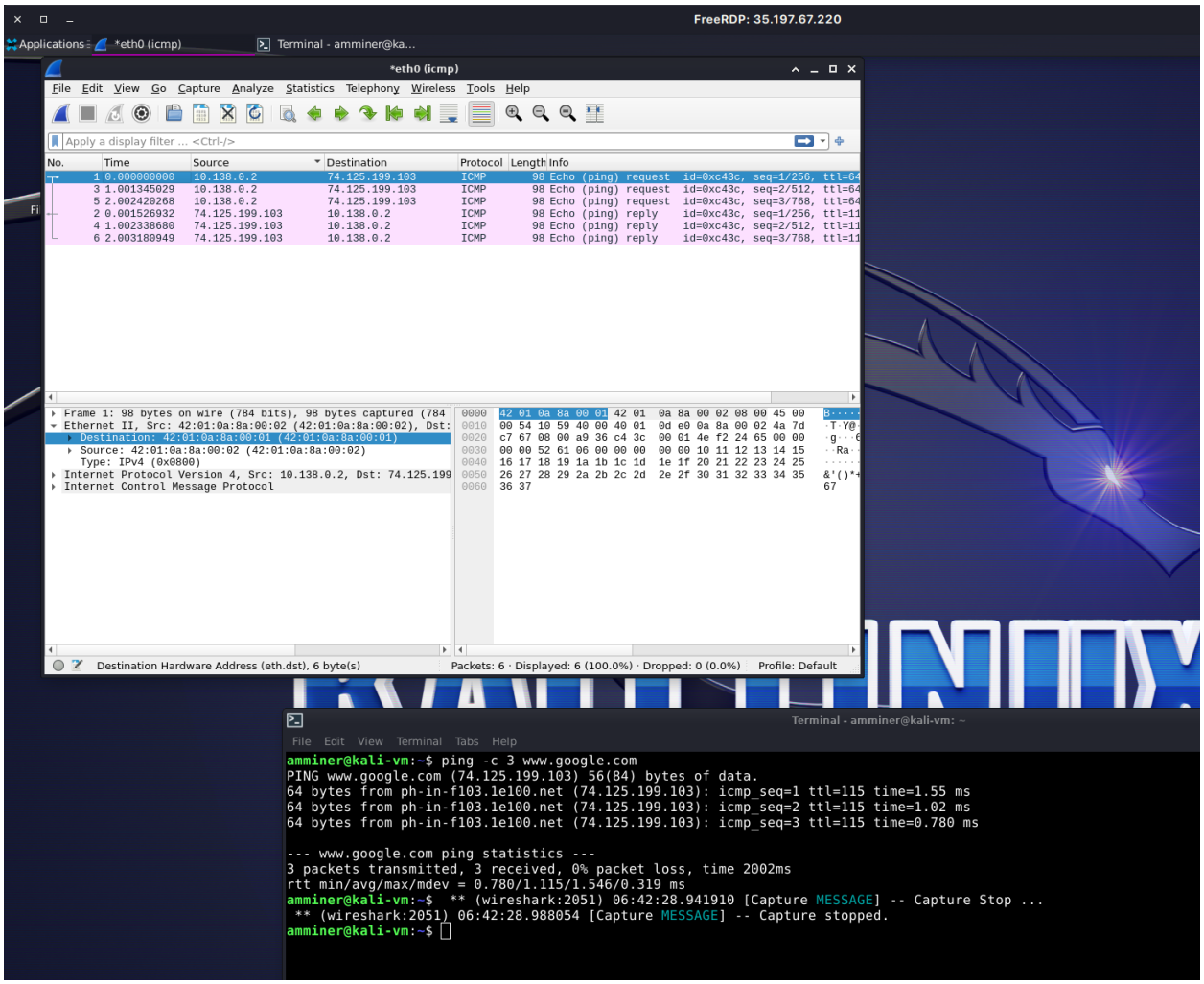
    - The IP address of the default router

      10.138.0.1

Amelia Miner
10/8/23

7. -

- Capture ICMP on eth0 and send 3 pings to www.google.com.
- Click on the first packet in the top window of the wireshark UI. Then, in the middle window, expand the data-link layer packet and click on the destination hardware addresses. See which bytes in the payload window this corresponds to.



- Does the destination MAC address correspond to an interface on the VM, an interface on the default router or an interface on Google's web site?
  T**he destination MAC is the MAC of the default router:**

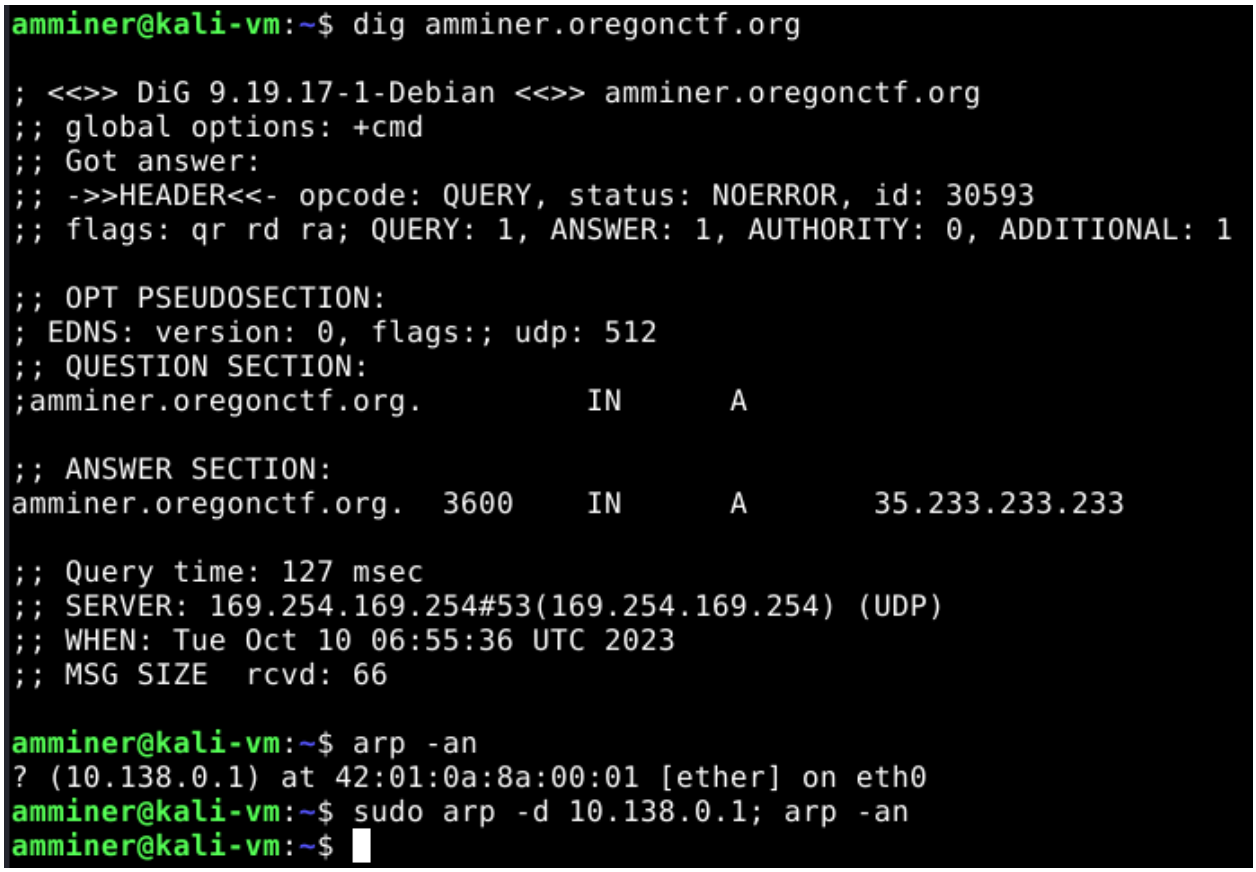
```
amminer@kali-vm:~$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         10.138.0.1      0.0.0.0         UG        0 0          0 eth0
10.138.0.1      0.0.0.0         255.255.255.255 UH        0 0          0 eth0
172.17.0.0      0.0.0.0         255.255.0.0     U         0 0          0 docker0
amminer@kali-vm:~$ arp -a 10.138.0.1
? (10.138.0.1) at 42:01:0a:8a:00:01 [ether] on eth0
amminer@kali-vm:~$
```

- Click on the next packet in the trace. Does the destination MAC address correspond to an interface on the VM, an interface on the default router or an interface on Google's

web site?
The next packet is another ping, same destination address. There are 3 pings and 3 replies. The replies have a destination MAC address corresponding with my VM's eth0 interface.

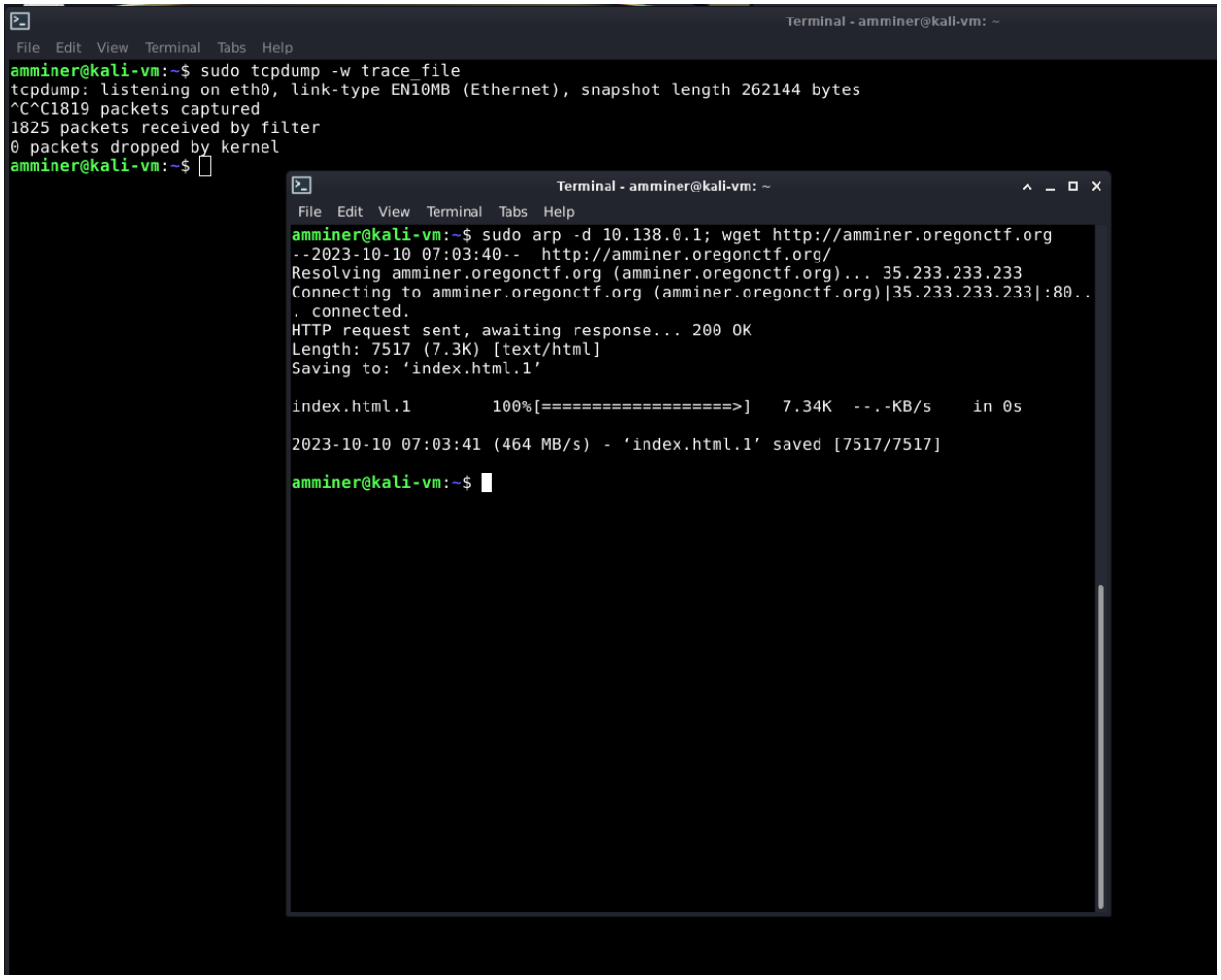
8. Network Recap Lab #4

- Find the IP address of <OdinId>.oregonctf.org, replacing <OdinId> with your OdinId



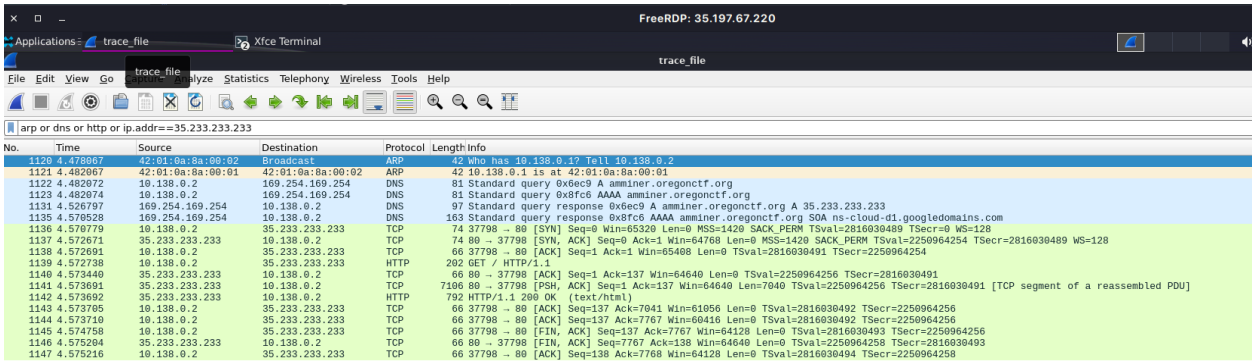
Amelia Miner
10/8/23

9. Collect trace



10. Analyze trace

- Open the cap in wireshark and filter as shown:



- ARP
    - What packet numbers in the trace are the result of the VM attempting to get the hardware address of the default router?
      1120, 1121
    - What is this hardware address?
      42:01:0a:8a:00:01
- DNS
    - What packet numbers in the trace correspond to the DNS request for the web site?
      1122, 1123
    - What is the IP address of the local DNS server being queried?
      169.254.169.254
- TCP
    - What packet numbers in the trace correspond to the initial TCP handshake for the web request?
      1136, 1137, 1138 (syn, syn/ack, ack)
    - How long does it take to perform the initial TCP handshake?
      A little under 0.002 seconds.
- HTTP
    - What packet numbers in the trace correspond to the actual HTTP request and response?
      1139 is the GET request and 1142 is the 200-OK response.
    - How long does it take to process the HTTP request after the handshake?
      Just under 0.001 seconds.

11. Clean up

Done!