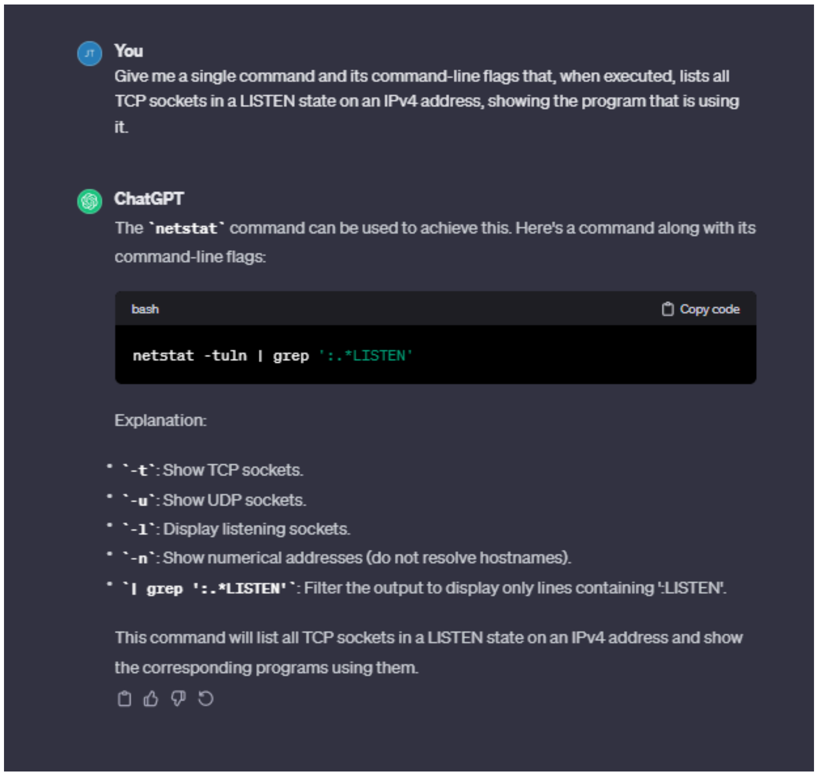
Odin ID: jtn7

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



You

Give me 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.

ChatGPT

The `netstat` command can be used to achieve this. Here's a command along with its command-line flags:

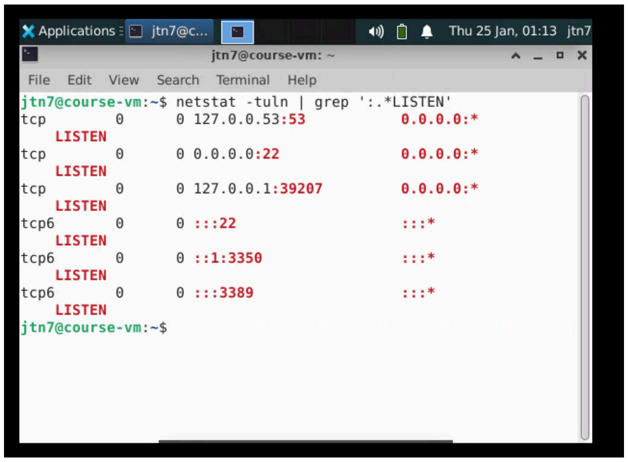
```bash
netstat -tuln | grep ':.*LISTEN'
```

Explanation:

- `-t`: Show TCP sockets.
- `-u`: Show UDP sockets.
- `-l`: Display listening sockets.
- `-n`: Show numerical addresses (do not resolve hostnames).
- `| grep ':.*LISTEN'`: Filter the output to display only lines containing ':LISTEN'.

This command will list all TCP sockets in a LISTEN state on an IPv4 address and show the corresponding programs using them.

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



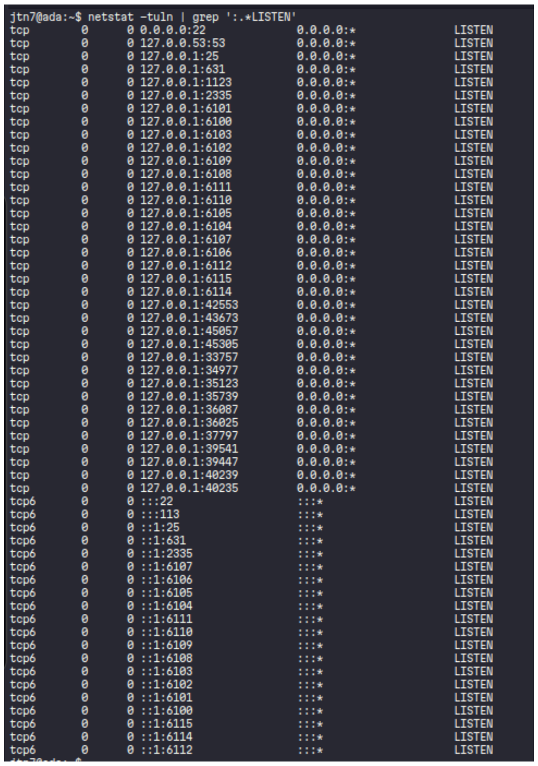
```
jtn7@course-vm:~$ netstat -tuln | grep ':.*LISTEN'
tcp        0      0 127.0.0.53:53           0.0.0.0:*
    LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*
    LISTEN
tcp        0      0 127.0.0.1:39207         0.0.0.0:*
    LISTEN
tcp6       0      0 :::22                   :::*
    LISTEN
tcp6       0      0 ::1:3350                :::*
    LISTEN
tcp6       0      0 :::3389                 :::*
    LISTEN
jtn7@course-vm:~$
```

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

Port 22 can be contacted from any interface on the machine because connected to 0.0.0.0.0
Port 39207 can only be contacted by local processes because it is connected to localhost or 127.0.0.1

Odin ID: jtn7

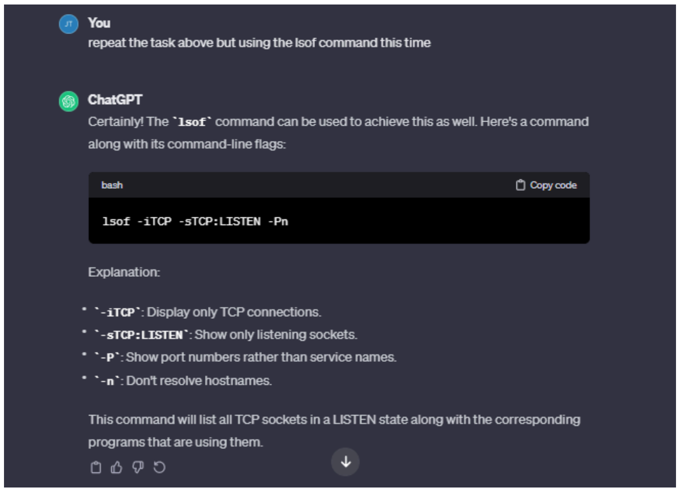
Run the command again, but do not use sudo as this is a machine managed by CAT. Include a screenshot of the output.



```
jtn7@ada:~$ netstat -tuln | grep ':.*LISTEN'
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:1123          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:2335          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6101          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6100          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6103          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6102          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6109          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6108          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6111          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6110          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6105          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6104          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6107          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6106          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6112          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6115          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6114          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:42553         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:43673         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:45057         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:45305         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:33757         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:34977         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:35123         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:35739         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:36087         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:36025         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:37797         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:39541         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:39447         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:40239         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:40235         0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 :::113                  :::*                    LISTEN
tcp6       0      0 ::1:25                  :::*                    LISTEN
tcp6       0      0 ::1:631                 :::*                    LISTEN
tcp6       0      0 ::1:2335                :::*                    LISTEN
tcp6       0      0 ::1:6107                :::*                    LISTEN
tcp6       0      0 ::1:6106                :::*                    LISTEN
tcp6       0      0 ::1:6105                :::*                    LISTEN
tcp6       0      0 ::1:6104                :::*                    LISTEN
tcp6       0      0 ::1:6111                :::*                    LISTEN
tcp6       0      0 ::1:6110                :::*                    LISTEN
tcp6       0      0 ::1:6109                :::*                    LISTEN
tcp6       0      0 ::1:6108                :::*                    LISTEN
tcp6       0      0 ::1:6103                :::*                    LISTEN
tcp6       0      0 ::1:6102                :::*                    LISTEN
tcp6       0      0 ::1:6101                :::*                    LISTEN
tcp6       0      0 ::1:6100                :::*                    LISTEN
tcp6       0      0 ::1:6115                :::*                    LISTEN
tcp6       0      0 ::1:6114                :::*                    LISTEN
tcp6       0      0 ::1:6112                :::*                    LISTEN
```

List the services that this machine provides for external access
Services like port 22, 113 and some others are for external access.

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.
Take a screenshot of the prompt and the command that ChatGPT generates



You
repeat the task above but using the lsof command this time

ChatGPT
Certainly! The `lsof` command can be used to achieve this as well. Here's a command along with its command-line flags:

```bash
lsof -iTCP -sTCP:LISTEN -Pn
```

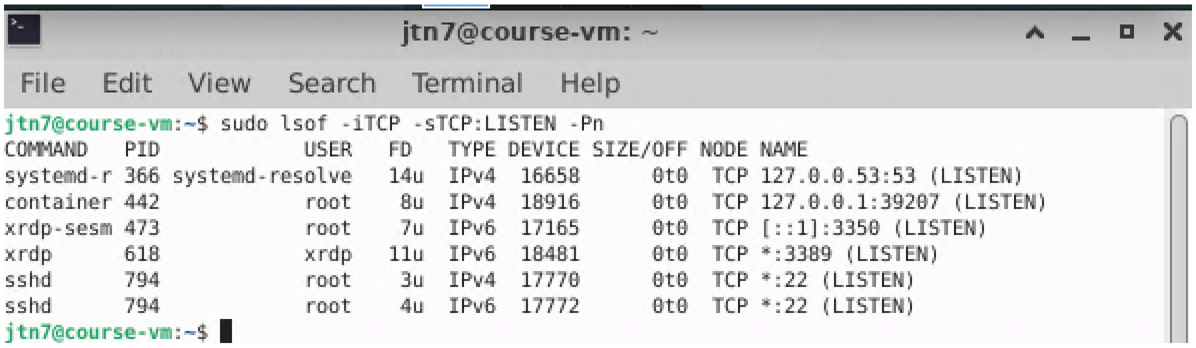
Explanation:

- `-iTCP`: Display only TCP connections.
- `-sTCP:LISTEN`: Show only listening sockets.
- `-P`: Show port numbers rather than service names.
- `-n`: Don't resolve hostnames.

This command will list all TCP sockets in a LISTEN state along with the corresponding programs that are using them.

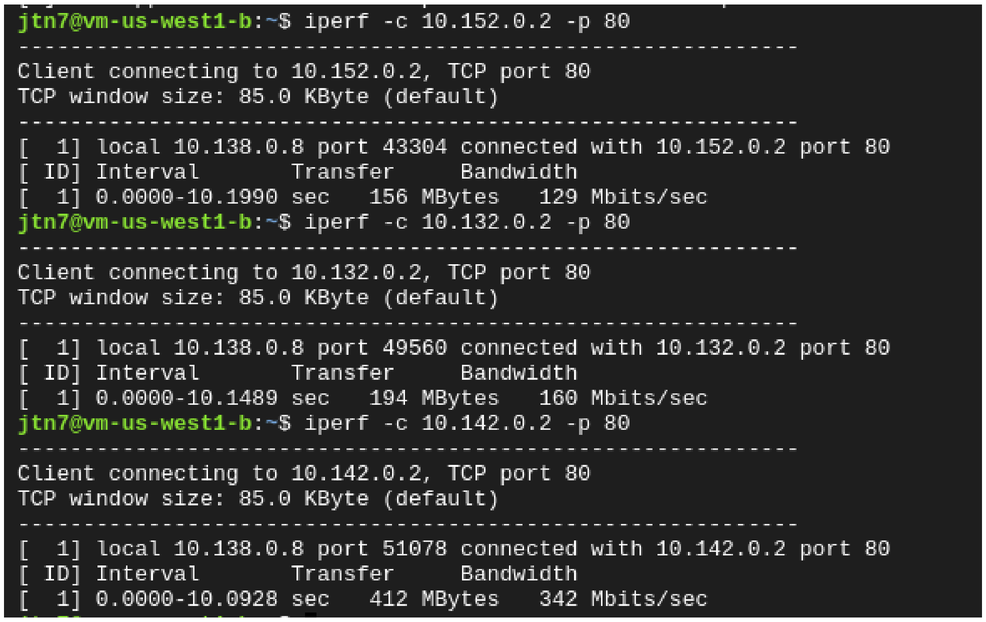
Odin ID: jtn7

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



```
jtn7@course-vm:~$ sudo lsof -iTCP -sTCP:LISTEN -Pn
COMMAND    PID            USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
systemd-r 366 systemd-resolve   14u  IPv4  16658      0t0  TCP 127.0.0.53:53 (LISTEN)
container 442            root    8u  IPv4  18916      0t0  TCP 127.0.0.1:39207 (LISTEN)
xrdp-sesm 473            root    7u  IPv6  17165      0t0  TCP [::1]:3350 (LISTEN)
xrdp      618            xrdp   11u  IPv6  18481      0t0  TCP *:3389 (LISTEN)
sshd      794            root    3u  IPv4  17770      0t0  TCP *:22 (LISTEN)
sshd      794            root    4u  IPv6  17772      0t0  TCP *:22 (LISTEN)
jtn7@course-vm:~$
```

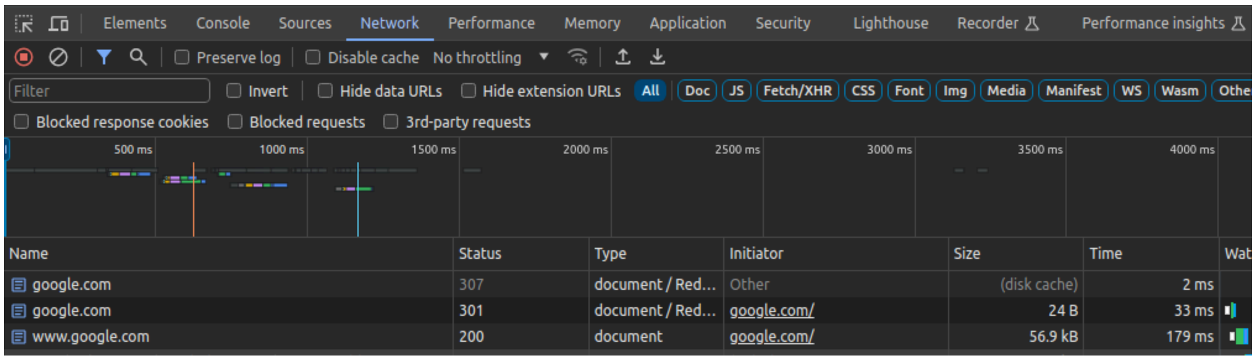
Show a screenshot of the measured bandwidth available between your us-west1-b VM and each of the other Compute Engine VMs. Explain the relative differences (or lack thereof) in your results.



```
jtn7@vm-us-west1-b:~$ iperf -c 10.152.0.2 -p 80
------------------------------------------------------------
Client connecting to 10.152.0.2, TCP port 80
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  1] local 10.138.0.8 port 43304 connected with 10.152.0.2 port 80
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.1990 sec   156 MBytes   129 Mbits/sec
jtn7@vm-us-west1-b:~$ iperf -c 10.132.0.2 -p 80
------------------------------------------------------------
Client connecting to 10.132.0.2, TCP port 80
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  1] local 10.138.0.8 port 49560 connected with 10.132.0.2 port 80
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.1489 sec   194 MBytes   160 Mbits/sec
jtn7@vm-us-west1-b:~$ iperf -c 10.142.0.2 -p 80
------------------------------------------------------------
Client connecting to 10.142.0.2, TCP port 80
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  1] local 10.138.0.8 port 51078 connected with 10.142.0.2 port 80
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0928 sec   412 MBytes   342 Mbits/sec
```

The physical location of the VMs had on impact on bandwidth. The closer a region was to us-west, the more bandwidth there was between us-west and that particular region.

Take a screenshot of the initial 3 requests that the browser makes for your lab notebook.



Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder Performance insights
Preserve log Disable cache No throttling
Filter Invert Hide data URLs Hide extension URLs All Doc JS Fetch/XHR CSS Font Img Media Manifest WS Wasm Othe
Blocked response cookies Blocked requests 3rd-party requests
500 ms 1000 ms 1500 ms 2000 ms 2500 ms 3000 ms 3500 ms 4000 ms

Name	Status	Type	Initiator	Size	Time
google.com	307	document / Red...	Other	(disk cache)	2 ms
google.com	301	document / Red...	google.com/	24 B	33 ms
www.google.com	200	document	google.com/	56.9 kB	179 ms

Odin ID: jtn7

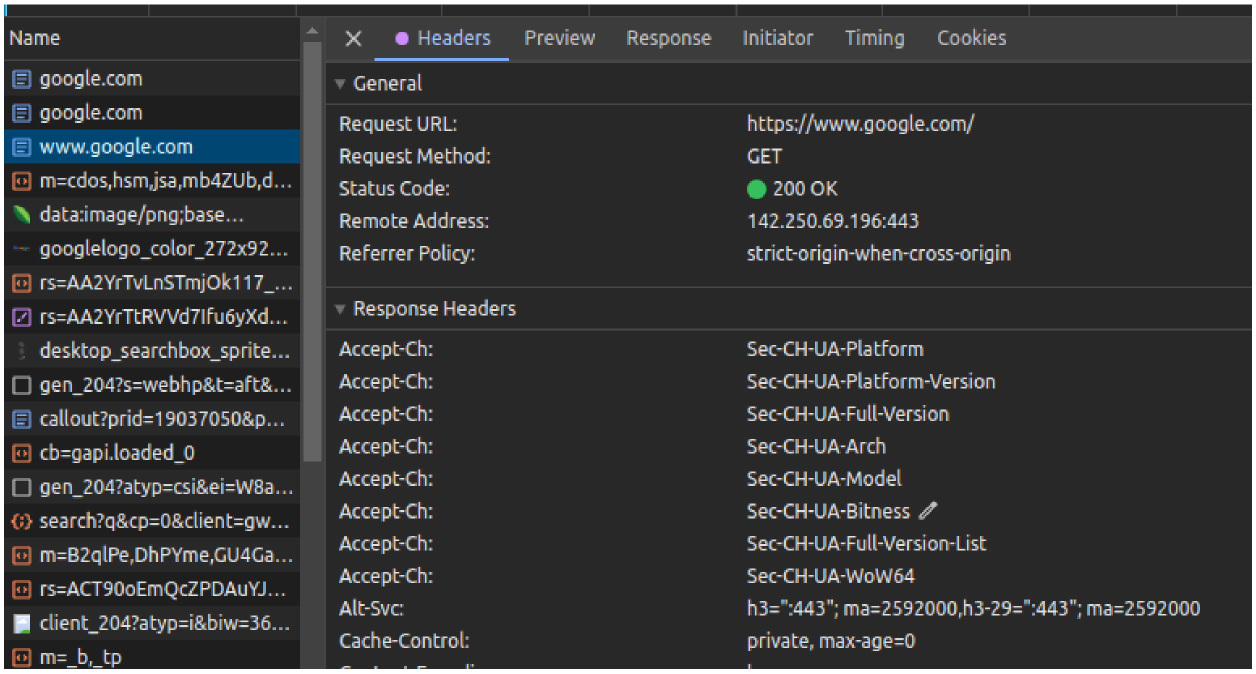
For each of the three requests, click on the request and examine the HTTP status code returned as well as the HTTP request and response headers for the request. Answer the following questions:

- What is the URL being requested?
  http://google.com

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

1. First we request http://google.com which comes back with a 307 code meaning internal redirect.
2. Then, google redirects us to https://google.com which comes back with a 301 code meaning moved permanently, meaning that this url will permanently move the user to another location
3. Finally we arrive at https://www.google.com with a 200 code meaning success.

- 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).


Name
google.com
google.com
www.google.com
m=cdos,hsm,jsa,mb4ZUb,d...
data:image/png;base...
googlelogo_color_272x92...
rs=AA2YrTvLnSTmjOk117_...
rs=AA2YrTtRVVd7Ifu6yXd...
desktop_searchbox_sprite...
gen_204?s=webhp&t=aft&...
callout?prid=19037050&p...
cb=gapi.loaded_0
gen_204?atyp=csi&ei=W8a...
search?q&cp=0&client=gw...
m=B2qlPe,DhPYme,GU4Ga...
rs=ACT90oEmQcZPDAuYJ...
client_204?atyp=i&biw=36...
m=_b,_tp
Headers
Preview
Response
Initiator
Timing
Cookies

General

Request URL:	https://www.google.com/
Request Method:	GET
Status Code:	● 200 OK
Remote Address:	142.250.69.196:443
Referrer Policy:	strict-origin-when-cross-origin

Response Headers

Accept-Ch:	Sec-CH-UA-Platform
Accept-Ch:	Sec-CH-UA-Platform-Version
Accept-Ch:	Sec-CH-UA-Full-Version
Accept-Ch:	Sec-CH-UA-Arch
Accept-Ch:	Sec-CH-UA-Model
Accept-Ch:	Sec-CH-UA-Bitness
Accept-Ch:	Sec-CH-UA-Full-Version-List
Accept-Ch:	Sec-CH-UA-WoW64
Alt-Svc:	h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
Cache-Control:	private, max-age=0

Find the Location: response header for the two redirections.

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

We get redirected from http://google.com to https://google.com

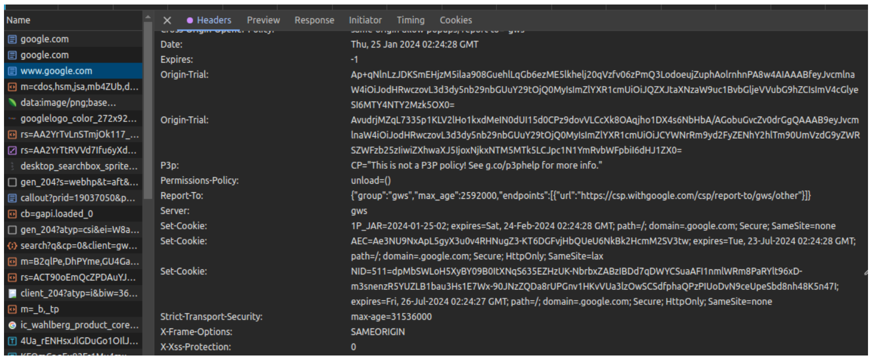
- What URL does the second redirection send the browser to?

We then get redirected from https://google.com to https://www.google.com

Odin ID: jtn7

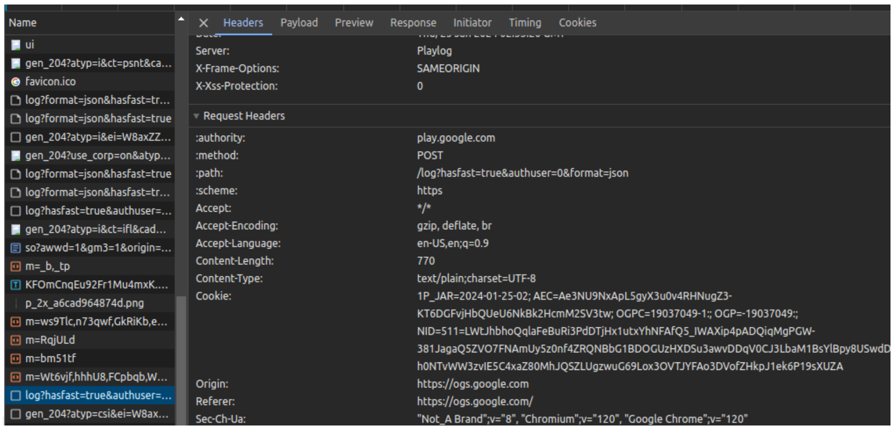
Examine the HTTP request and response headers for cookies throughout the requests.

- Take a screenshot of when cookies are set via Set-Cookie:


Name
Headers
Preview
Response
Initiator
Timing
Cookies
google.com
google.com
www.google.com
m=cdos,hsm,jsa,mb4ZUb,d...
data:image/png;base...
googlelogo_color_272x92...
rs=AA2YrTvLnSTmjOk117_...
rs=AA2YrTtRVVd7Ifu6yXd...
desktop_searchbox_sprite...
gen_204?s=webhp&t=aft&...
callout?prid=19037050&p...
cb=gapi.loaded_0
gen_204?atyp=csi&ei=W8a...
search?q&cp=0&client=gw...
m=B2qlPe,DhPYme,GU4Ga...
rs=ACT90oEmQcZPDAuYJ...
client_204?atyp=i&biw=36...
m=_b,_tp
ic_wahlberg_product_core...
4Ua_rENHsxJlGDuGo1OlJ...

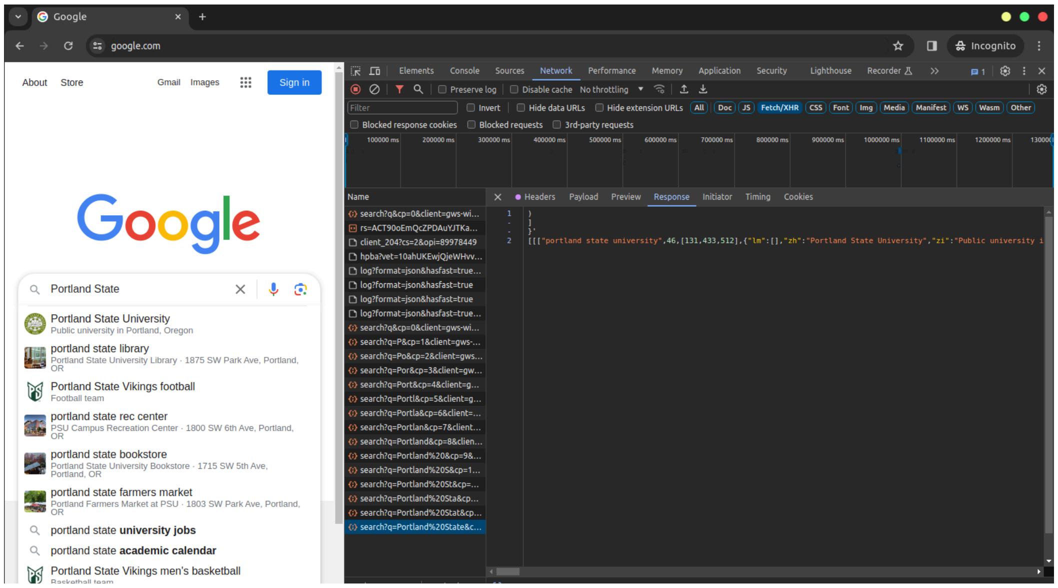
```
Date:                        Thu, 25 Jan 2024 02:24:28 GMT
Expires:                     -1
Origin-Trial:                Ap+qNlnLzJDKSmEHjzM5ilaa908GuehlLqGb6ezME5lkhelj20qVzfv06zPmQ3LodoeujZuphAolrnhnPA8w4AIAAABfeyJvcmlna
                             W4iOiJodHRwczovL3d3dy5nb29nbGUuY29tOjQ0MyIsImZlYXR1cmUiOiJQZXJtaXNzaW9uc1BvbGljeVVubG9hZCIsImV4cGlye
                             SI6MTY4NTY2Mzk5OX0=
Origin-Trial:                AvudrjMZqL7335p1KLV2lHo1kxdMeIN0dUI15d0CPz9dovVLCcXk8OAqjho1DX4s6NbHbA/AGobuGvcZv0drGgQAAAB9eyJvcm
                             lnaW4iOiJodHRwczovL3d3dy5nb29nbGUuY29tOjQ0MyIsImZlYXR1cmUiOiJCYWNrRm9yd2FyZENhY2hlTm90UmVzdG9yZWR
                             SZWFzb25zIiwiZXhwaXJ5IjoxNjkxNTM5MTk5LCJpc1N1YmRvbWFpbiI6dHJ1ZX0=
P3p:                         CP="This is not a P3P policy! See g.co/p3phelp for more info."
Permissions-Policy:          unload=()
Report-To:                   {"group":"gws","max_age":2592000,"endpoints":[{"url":"https://csp.withgoogle.com/csp/report-to/gws/other"}]}
Server:                      gws
Set-Cookie:                  1P_JAR=2024-01-25-02; expires=Sat, 24-Feb-2024 02:24:28 GMT; path=/; domain=.google.com; Secure; SameSite=none
Set-Cookie:                  AEC=Ae3NU9NxApL5gyX3u0v4RHNugZ3-KT6DGFvjHbQUeU6NkBk2HcmM2SV3tw; expires=Tue, 23-Jul-2024 02:24:28 GMT;
                             path=/; domain=.google.com; Secure; HttpOnly; SameSite=lax
Set-Cookie:                  NID=511=dpMbSWLoH5XyBY09B0ItXNqS635EZHzUK-NbrbxZABzIBDd7qDWYCSuaAFI1nmlWRm8PaRYlt96xD-
                             m3snenzR5YUZLB1bau3Hs1E7Wx-90JNzZQDa8rUPGnv1HKvVUa3lzOwSCSdfphaQPzPIUoDvN9ceUpeSbd8nh48K5n47I;
                             expires=Fri, 26-Jul-2024 02:24:27 GMT; path=/; domain=.google.com; Secure; HttpOnly; SameSite=none
Strict-Transport-Security:   max-age=31536000
X-Frame-Options:             SAMEORIGIN
X-Xss-Protection:            0
```

- Take a screenshot of when cookies are attached via Cookie:


Name
Headers
Payload
Preview
Response
Initiator
Timing
Cookies
ui
gen_204?atyp=i&ct=psnt&ca...
favicon.ico
log?format=json&hasfast=tr...
log?format=json&hasfast=true
gen_204?atyp=i&ei=W8axZZ...
gen_204?use_corp=on&atyp...
log?format=json&hasfast=true
log?format=json&hasfast=tr...
log?hasfast=true&authuser=...
gen_204?atyp=i&ct=ifl&cad...
so?awwd=1&gm3=1&origin=...
m=_b,_tp
KFOmCnqEu92Fr1Mu4mxK...
p_2x_a6cad964874d.png
m=ws9Tlc,n73qwf,GkRiKb,e...
m=RqjULd
m=bm51tf
m=Wt6vjf,hhhU8,FCpbqb,W...
log?hasfast=true&authuser=...
gen_204?atyp=csi&ei=W8ax...

```
Server:                      Playlog
X-Frame-Options:             SAMEORIGIN
X-Xss-Protection:            0

▾ Request Headers
:authority:                  play.google.com
:method:                     POST
:path:                       /log?hasfast=true&authuser=0&format=json
:scheme:                     https
Accept:                      */*
Accept-Encoding:             gzip, deflate, br
Accept-Language:             en-US,en;q=0.9
Content-Length:              770
Content-Type:                text/plain;charset=UTF-8
Cookie:                      1P_JAR=2024-01-25-02; AEC=Ae3NU9NxApL5gyX3u0v4RHNugZ3-
                             KT6DGFvjHbQUeU6NkBk2HcmM2SV3tw; OGPC=19037049-1:; OGP=-19037049:;
                             NID=511=LWtJhbhoQqlaFeBuRi3PdDTjHx1utxYhNFAfQ5_IWAXip4pADQiqMgPGW-
                             381JagaQ5ZVO7FNAmUy5z0nf4ZRQNBbG1BDOGUzHXDSu3awvDDqV0CJ3LbaM1BsYlBpy8USwdD
                             h0NTvWW3zvIE5C4xaZ80MhJQSZLUgzwuG69Lox3OVTJYFAo3DVofZHkpJ1ek6P19sXUZA
Origin:                      https://ogs.google.com
Referer:                     https://ogs.google.com/
Sec-Ch-Ua:                   "Not_A Brand";v="8", "Chromium";v="120", "Google Chrome";v="120"
```

- Show the requests and responses in the listing. Click on the last request sent, then click on the response to see that its payload has returned the data that is then rendered on the search page similar to what is shown below for "rabbid"

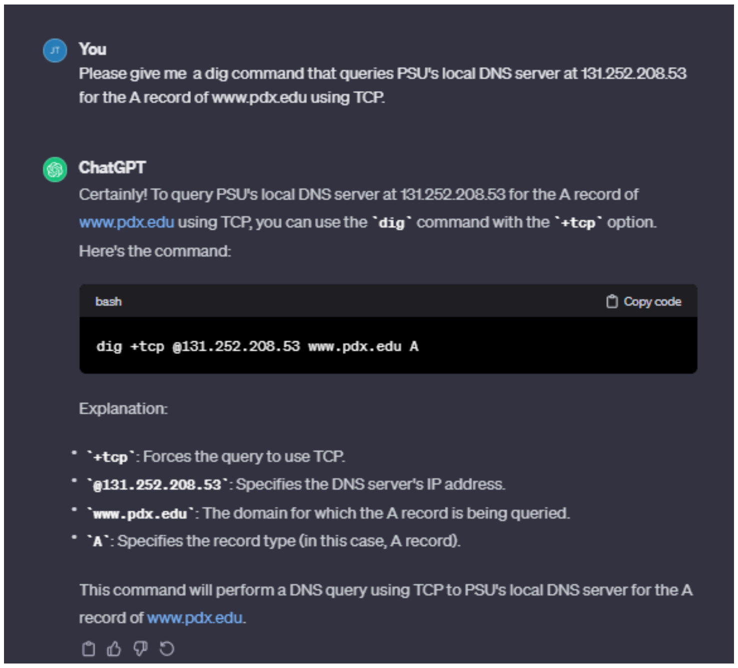

Google
google.com
Incognito
About
Store
Gmail
Images
Sign in
Google
Portland State
Portland State University
Public university in Portland, Oregon
portland state library
Portland State University Library · 1875 SW Park Ave, Portland, OR
Portland State Vikings football
Football team
portland state rec center
PSU Campus Recreation Center · 1800 SW 6th Ave, Portland, OR
portland state bookstore
Portland State University Bookstore · 1715 SW 5th Ave, Portland, OR
portland state farmers market
Portland Farmers Market at PSU · 1803 SW Park Ave, Portland, OR
portland state university jobs
portland state academic calendar
Portland State Vikings men's basketball
Basketball team
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Recorder
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
Hide extension URLs
All
Doc
JS
Fetch/XHR
CSS
Font
Img
Media
Manifest
WS
Wasm
Other
Blocked response cookies
Blocked requests
3rd-party requests
Name
Headers
Payload
Preview
Response
Initiator
Timing
Cookies
search?q&cp=0&client=gws-wi...
rs=ACT90oEmQcZPDAuYJTKa...
client_204?cs=2&opi=89978449
hpba?vet=10ahUKEwjQjeWHvv...
log?format=json&hasfast=true...
log?format=json&hasfast=true
log?format=json&hasfast=true
log?format=json&hasfast=true...
search?q&cp=0&client=gws-wi...
search?q=P&cp=1&client=gws-...
search?q=Po&cp=2&client=gws...
search?q=Por&cp=3&client=gw...
search?q=Port&cp=4&client=g...
search?q=Portl&cp=5&client=g...
search?q=Portla&cp=6&client=...
search?q=Portlan&cp=7&client...
search?q=Portland&cp=8&clien...
search?q=Portland%20&cp=9&...
search?q=Portland%20S&cp=1...
search?q=Portland%20St&cp=...
search?q=Portland%20Sta&cp...
search?q=Portland%20Stat&cp...
search?q=Portland%20State&c...

```
)
]
}'
[[["portland state university",46,[131,433,512],{"lm":[],"zh":"Portland State University","zi":"Public university i
```

Odin ID: jtn7

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.

- Take a screenshot of the prompt and the dig command produced.



You
Please give me 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.

ChatGPT
Certainly! To query PSU's local DNS server at 131.252.208.53 for the A record of www.pdx.edu using TCP, you can use the `dig` command with the `+tcp` option. Here's the command:

```bash
dig +tcp @131.252.208.53 www.pdx.edu A
```

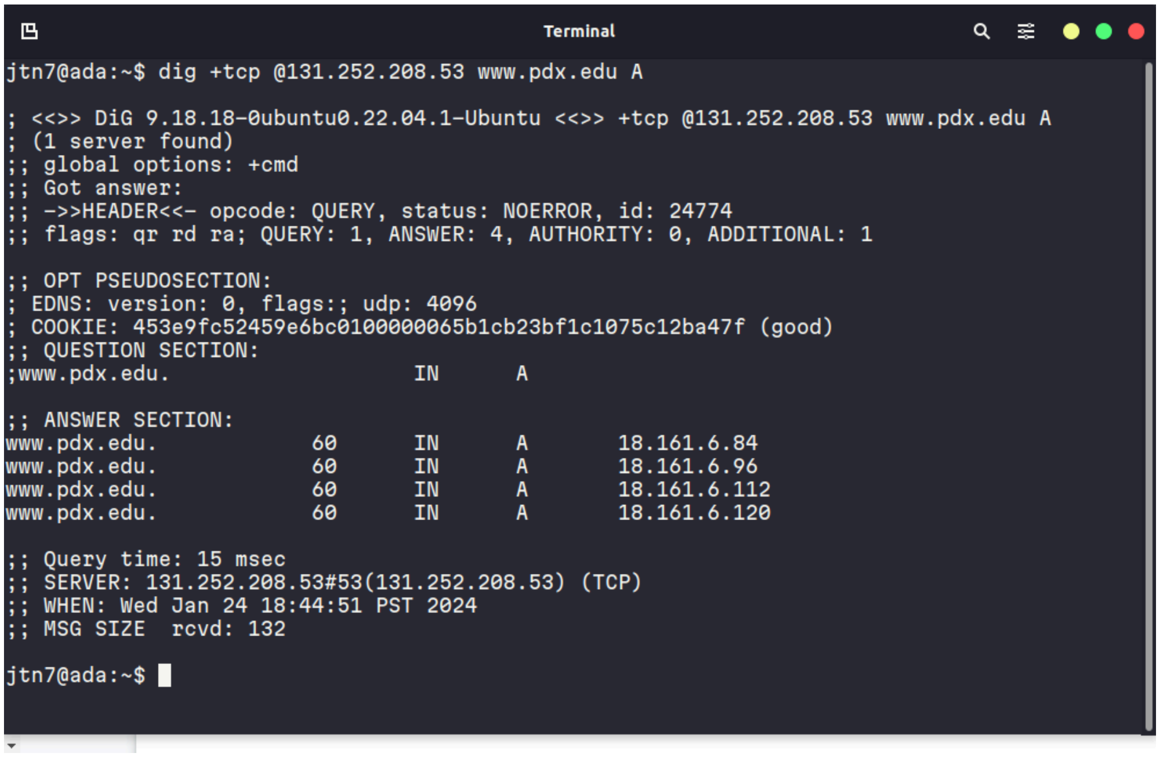
Explanation:

- `+tcp`: Forces the query to use TCP.
- `@131.252.208.53`: Specifies the DNS server's IP address.
- `www.pdx.edu`: The domain for which the A record is being queried.
- `A`: Specifies the record type (in this case, A record).

This command will perform a DNS query using TCP to PSU's local DNS server for the A record of www.pdx.edu.

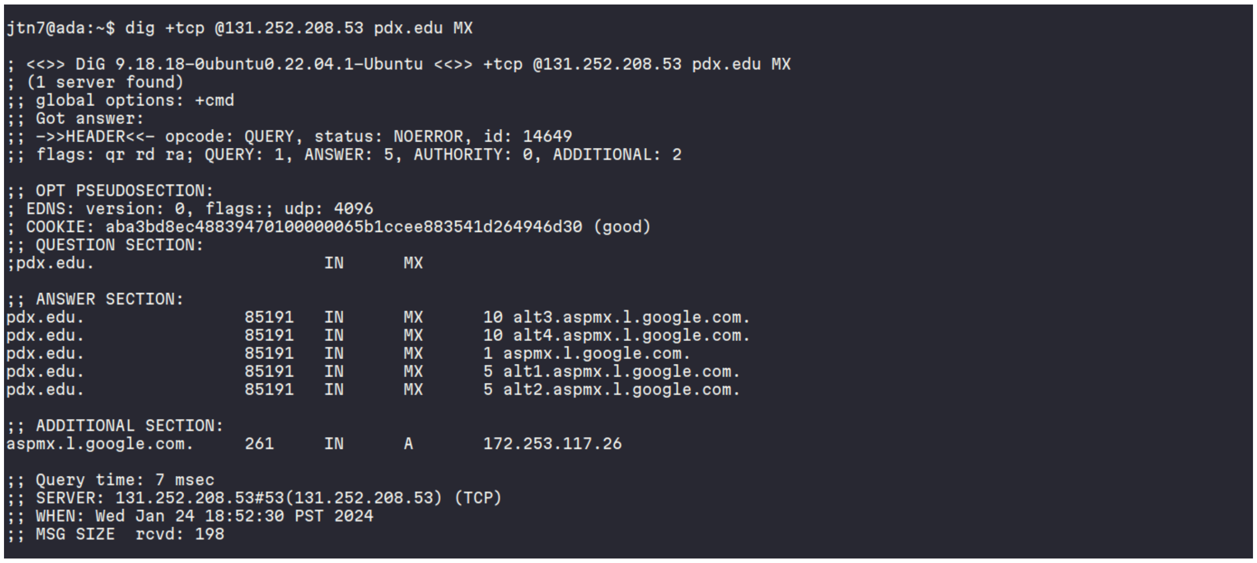
Run the command to find the record. Then, use dig to do the same for the MX record of pdx.edu.

- Take a screenshot of the records returned for your lab notebook.



```
jtn7@ada:~$ dig +tcp @131.252.208.53 www.pdx.edu A

; <<>> DiG 9.18.18-0ubuntu0.22.04.1-Ubuntu <<>> +tcp @131.252.208.53 www.pdx.edu A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24774
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 453e9fc52459e6bc0100000065b1cb23bf1c1075c12ba47f (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       18.161.6.96
www.pdx.edu.            60      IN      A       18.161.6.112
www.pdx.edu.            60      IN      A       18.161.6.120

;; Query time: 15 msec
;; SERVER: 131.252.208.53#53(131.252.208.53) (TCP)
;; WHEN: Wed Jan 24 18:44:51 PST 2024
;; MSG SIZE  rcvd: 132

jtn7@ada:~$ 
```

Odin ID: jtn7



```
jtn7@ada:~$ dig +tcp @131.252.208.53 pdx.edu MX

; <<>> DiG 9.18.18-0ubuntu0.22.04.1-Ubuntu <<>> +tcp @131.252.208.53 pdx.edu MX
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 14649
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: aba3bd8ec4883947010000065b1ccee883541d264946d30 (good)
;; QUESTION SECTION:
;pdx.edu.                       IN      MX

;; ANSWER SECTION:
pdx.edu.                85191   IN      MX      10 alt3.aspmx.l.google.com.
pdx.edu.                85191   IN      MX      10 alt4.aspmx.l.google.com.
pdx.edu.                85191   IN      MX      1 aspmx.l.google.com.
pdx.edu.                85191   IN      MX      5 alt1.aspmx.l.google.com.
pdx.edu.                85191   IN      MX      5 alt2.aspmx.l.google.com.

;; ADDITIONAL SECTION:
aspmx.l.google.com.     261     IN      A       172.253.117.26

;; Query time: 7 msec
;; SERVER: 131.252.208.53#53(131.252.208.53) (TCP)
;; WHEN: Wed Jan 24 18:52:30 PST 2024
;; MSG SIZE  rcvd: 198
```

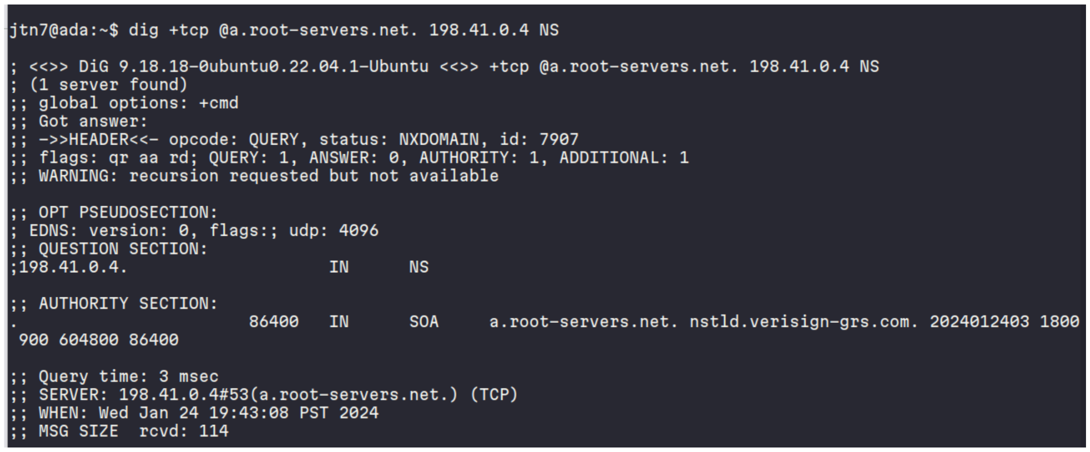
- What cloud provider hosts the web site for www.pdx.edu?
  Amazon

- What cloud provider handles mail for pdx.edu?
  Google

- List all of the iterative dig commands performed for the lookup

dig
dig +tcp @f.root-servers.net 192.5.5.241 NS
dig +tcp @a.root-servers.net. 198.41.0.4 NS

- Take a screenshot of the results of the final query for your lab notebook.

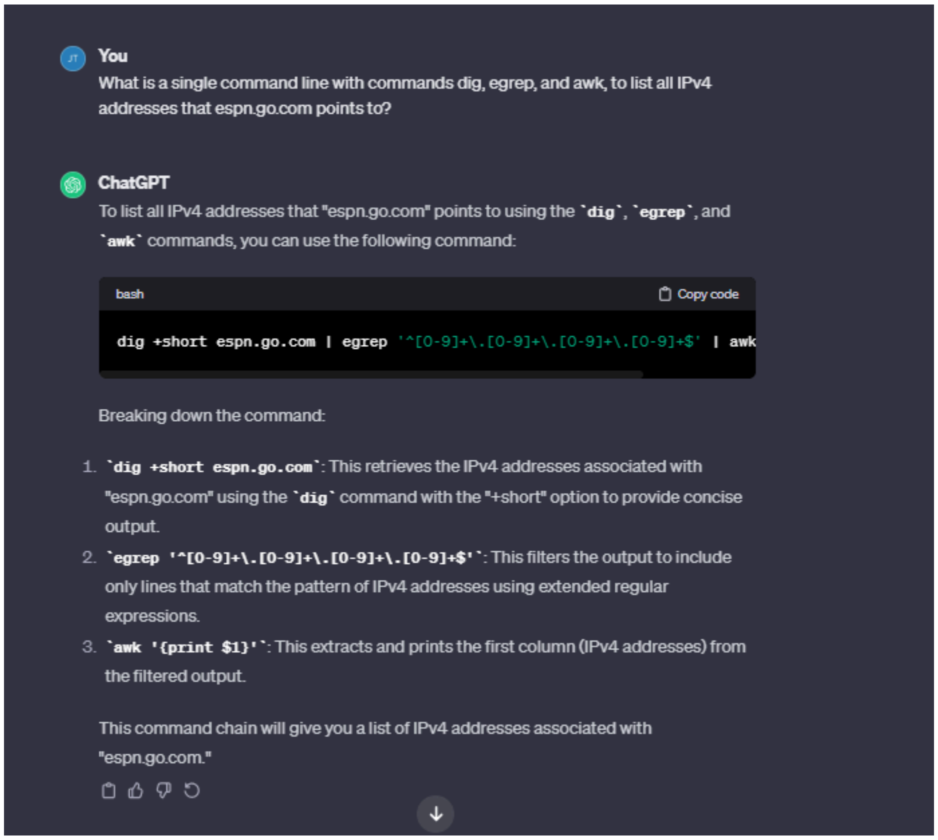


```
jtn7@ada:~$ dig +tcp @a.root-servers.net. 198.41.0.4 NS

; <<>> DiG 9.18.18-0ubuntu0.22.04.1-Ubuntu <<>> +tcp @a.root-servers.net. 198.41.0.4 NS
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 7907
;; flags: qr aa rd; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;198.41.0.4.                    IN      NS

;; AUTHORITY SECTION:
.                       86400   IN      SOA     a.root-servers.net. nstld.verisign-grs.com. 2024012403 1800
 900 604800 86400

;; Query time: 3 msec
;; SERVER: 198.41.0.4#53(a.root-servers.net.) (TCP)
;; WHEN: Wed Jan 24 19:43:08 PST 2024
;; MSG SIZE  rcvd: 114
```

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

Odin ID: jtn7

- Take a screenshot of the prompt and the command produced



You
What is a single command line with commands dig, egrep, and awk, to list all IPv4 addresses that espn.go.com points to?

ChatGPT
To list all IPv4 addresses that "espn.go.com" points to using the `dig`, `egrep`, and `awk` commands, you can use the following command:

```bash
dig +short espn.go.com | egrep '^[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+$' | awk
```

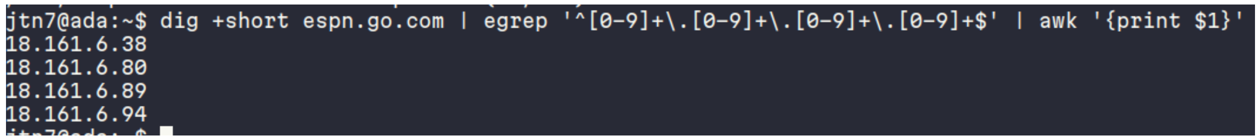
Breaking down the command:

1. `dig +short espn.go.com`: This retrieves the IPv4 addresses associated with "espn.go.com" using the `dig` command with the "+short" option to provide concise output.
2. `egrep '^[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+$'`: This filters the output to include only lines that match the pattern of IPv4 addresses using extended regular expressions.
3. `awk '{print $1}'`: This extracts and prints the first column (IPv4 addresses) from the filtered output.

This command chain will give you a list of IPv4 addresses associated with "espn.go.com."

Run the command

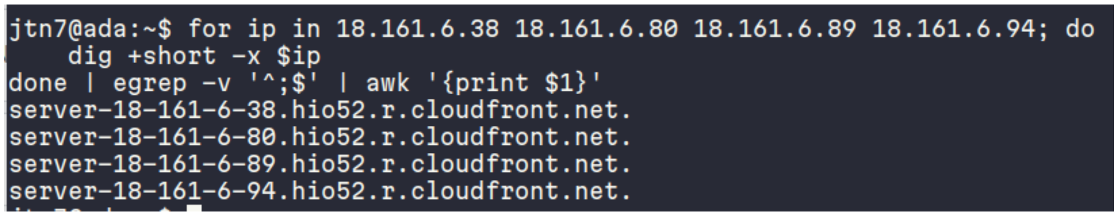
- Take a screenshot of its results for your lab notebook



```
jtn7@ada:~$ dig +short espn.go.com | egrep '^[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+$' | awk '{print $1}'
18.161.6.38
18.161.6.80
18.161.6.89
18.161.6.94
```

Find the flag for dig that allows one to perform reverse lookups on IPv4 addresses. Given the flag and the shell tutorial above, 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. As with the previous step, pipe the output of the for loop to egrep and awk so that the output consists only of the DNS names.

- Take a screenshot of the command and its results for your lab notebook



```
jtn7@ada:~$ for ip in 18.161.6.38 18.161.6.80 18.161.6.89 18.161.6.94; do
    dig +short -x $ip
done | egrep -v '^;$' | awk '{print $1}'
server-18-161-6-38.hio52.r.cloudfront.net.
server-18-161-6-80.hio52.r.cloudfront.net.
server-18-161-6-89.hio52.r.cloudfront.net.
server-18-161-6-94.hio52.r.cloudfront.net.
```

Odin ID: jtn7

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?

131.252.208.53: Portland, Oregon
198.82.247.66: Blacksburg, Virginia

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

131.252.208.53 -> 142.251.211.228
198.82.247.66 -> 142.250.31.105

Go back to https://www.iplocation.net/ and look up the geographical location of the two IP addresses you recorded.

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

142.251.211.228 -> Seattle, Washington
142.250.31.105 -> Ashburn, Virginia

- 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?

It corresponds to an interface on the VM, the MAC address of the VM.

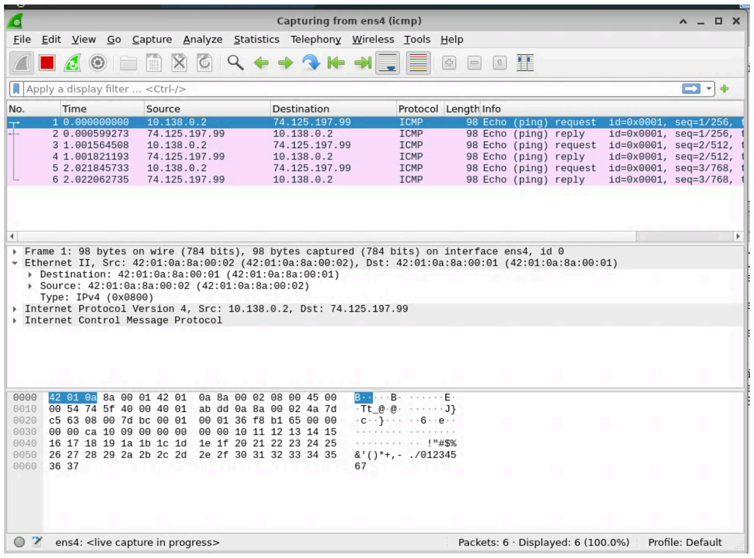
Click on the next packet in the trace.

- 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?

It also corresponds to an interface on the VM, the MAC address of the VM.

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.

- Take a screenshot of the bytes in the packet dump window as shown below

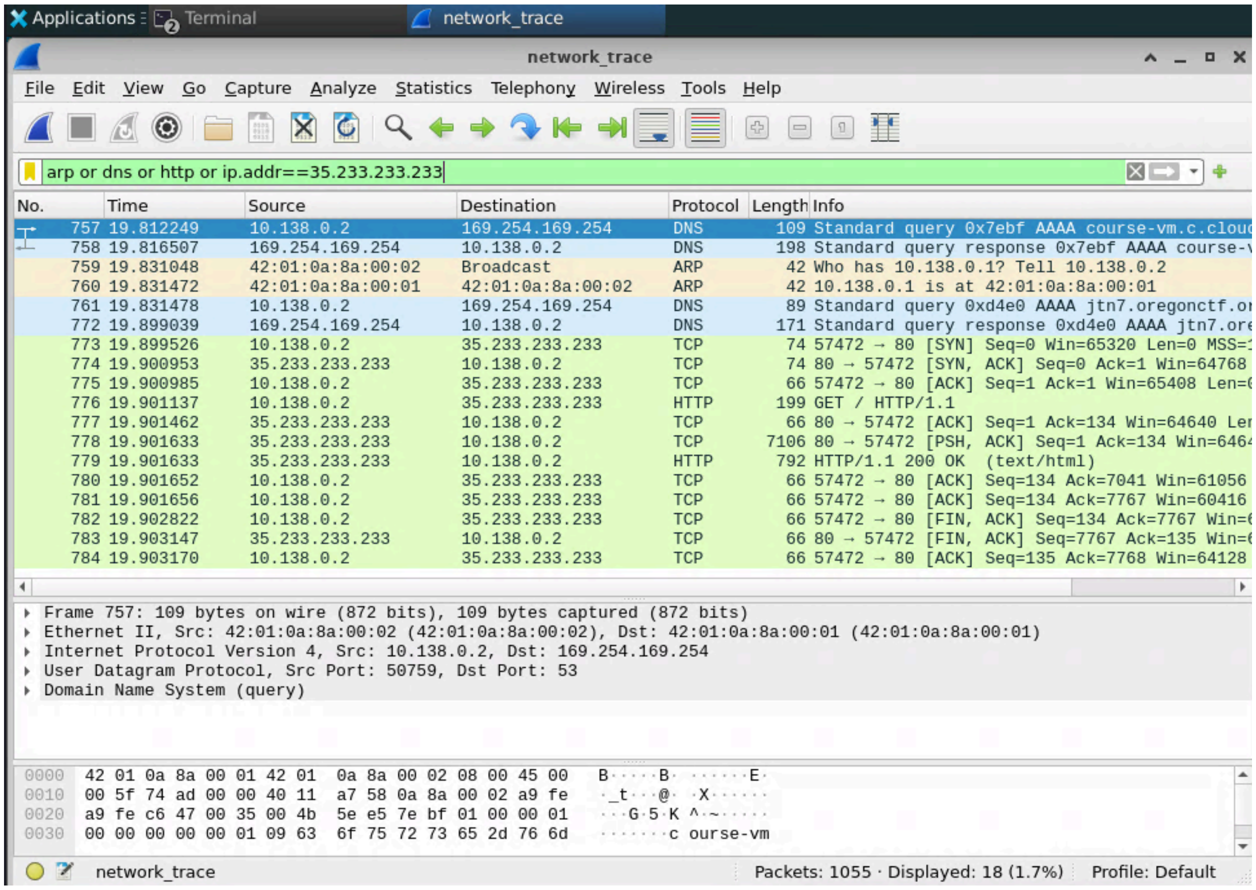

Capturing from ens4 (icmp)
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.  Time           Source          Destination     Protocol Length Info
  1 0.000000000    10.138.0.2      74.125.197.99   ICMP      98 Echo (ping) request  id=0x0001, seq=1/256,
  2 0.000599273    74.125.197.99   10.138.0.2      ICMP      98 Echo (ping) reply    id=0x0001, seq=1/256,
  3 1.001564508    10.138.0.2      74.125.197.99   ICMP      98 Echo (ping) request  id=0x0001, seq=2/512,
  4 1.001821193    74.125.197.99   10.138.0.2      ICMP      98 Echo (ping) reply    id=0x0001, seq=2/512,
  5 2.021845733    10.138.0.2      74.125.197.99   ICMP      98 Echo (ping) request  id=0x0001, seq=3/768,
  6 2.022062735    74.125.197.99   10.138.0.2      ICMP      98 Echo (ping) reply    id=0x0001, seq=3/768,

Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface ens4, id 0
Ethernet II, Src: 42:01:0a:8a:00:02 (42:01:0a:8a:00:02), Dst: 42:01:0a:8a:00:01 (42:01:0a:8a:00:01)
  Destination: 42:01:0a:8a:00:01 (42:01:0a:8a:00:01)
  Source: 42:01:0a:8a:00:02 (42:01:0a:8a:00:02)
  Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 10.138.0.2, Dst: 74.125.197.99
Internet Control Message Protocol

0000  42 01 0a 8a 00 01 42 01  0a 8a 00 02 08 00 45 00   B····B· ······E·
0010  00 54 74 5f 40 00 40 01  ab dd 0a 8a 00 02 4a 7d   ·Tt_@·@· ······J}
0020  c5 63 08 00 7d bc 00 01  00 01 36 f8 b1 65 00 00   ·c··}··· ··6··e··
0030  00 00 ca 10 09 00 00 00  00 00 10 11 12 13 14 15   ········ ········
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ········ ·· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                              67
```

ens4: <live capture in progress>   Packets: 6 · Displayed: 6 (100.0%)   Profile: Default

Odin ID: jtn7

- Take a screenshot of the all of the packets returned within Wireshark that includes their packet numbers


Applications
Terminal
network_trace
network_trace
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
arp or dns or http or ip.addr==35.233.233.233

No.  Time       Source             Destination        Protocol Length Info
757 19.812249 10.138.0.2          169.254.169.254    DNS       109 Standard query 0x7ebf AAAA course-vm.c.clouc
758 19.816507 169.254.169.254     10.138.0.2         DNS       198 Standard query response 0x7ebf AAAA course-\
759 19.831048 42:01:0a:8a:00:02   Broadcast          ARP        42 Who has 10.138.0.1? Tell 10.138.0.2
760 19.831472 42:01:0a:8a:00:01   42:01:0a:8a:00:02  ARP        42 10.138.0.1 is at 42:01:0a:8a:00:01
761 19.831478 10.138.0.2          169.254.169.254    DNS        89 Standard query 0xd4e0 AAAA jtn7.oregonctf.or
772 19.899039 169.254.169.254     10.138.0.2         DNS       171 Standard query response 0xd4e0 AAAA jtn7.ore
773 19.899526 10.138.0.2          35.233.233.233     TCP        74 57472 → 80 [SYN] Seq=0 Win=65320 Len=0 MSS=1
774 19.900953 35.233.233.233      10.138.0.2         TCP        74 80 → 57472 [SYN, ACK] Seq=0 Ack=1 Win=64768
775 19.900985 10.138.0.2          35.233.233.233     TCP        66 57472 → 80 [ACK] Seq=1 Ack=1 Win=65408 Len=0
776 19.901137 10.138.0.2          35.233.233.233     HTTP      199 GET / HTTP/1.1
777 19.901462 35.233.233.233      10.138.0.2         TCP        66 80 → 57472 [ACK] Seq=1 Ack=134 Win=64640 Len
778 19.901633 35.233.233.233      10.138.0.2         TCP      7106 80 → 57472 [PSH, ACK] Seq=1 Ack=134 Win=6464
779 19.901633 35.233.233.233      10.138.0.2         HTTP      792 HTTP/1.1 200 OK  (text/html)
780 19.901652 10.138.0.2          35.233.233.233     TCP        66 57472 → 80 [ACK] Seq=134 Ack=7041 Win=61056
781 19.901656 10.138.0.2          35.233.233.233     TCP        66 57472 → 80 [ACK] Seq=134 Ack=7767 Win=60416
782 19.902822 10.138.0.2          35.233.233.233     TCP        66 57472 → 80 [FIN, ACK] Seq=134 Ack=7767 Win=6
783 19.903147 35.233.233.233      10.138.0.2         TCP        66 80 → 57472 [FIN, ACK] Seq=7767 Ack=135 Win=6
784 19.903170 10.138.0.2          35.233.233.233     TCP        66 57472 → 80 [ACK] Seq=135 Ack=7768 Win=64128

▸ Frame 757: 109 bytes on wire (872 bits), 109 bytes captured (872 bits)
▸ Ethernet II, Src: 42:01:0a:8a:00:02 (42:01:0a:8a:00:02), Dst: 42:01:0a:8a:00:01 (42:01:0a:8a:00:01)
▸ Internet Protocol Version 4, Src: 10.138.0.2, Dst: 169.254.169.254
▸ User Datagram Protocol, Src Port: 50759, Dst Port: 53
▸ Domain Name System (query)

0000  42 01 0a 8a 00 01 42 01  0a 8a 00 02 08 00 45 00   B·····B· ······E·
0010  00 5f 74 ad 00 00 40 11  a7 58 0a 8a 00 02 a9 fe   ·_t···@· ·X······
0020  a9 fe c6 47 00 35 00 4b  5e e5 7e bf 01 00 00 01   ···G·5·K ^·~·····
0030  00 00 00 00 00 01 09 63  6f 75 72 73 65 2d 76 6d   ·······c ourse-vm

network_trace                          Packets: 1055 · Displayed: 18 (1.7%)    Profile: Default
```

ARP

- What packet numbers in the trace are the result of the VM attempting to get the hardware address of the default router?
  Packets 759 and 760
- What is this hardware address?
  10.138.0.1 (the default router) is at hardware address 42:01:0a:8a:00:01

DNS

- What packet numbers in the trace correspond to the DNS request for the web site?

Packet 757 and 761 are the queries and packets 758 and 772 are the query responses; these packets correspond to the DNS request for the website.

- What is the IP address of the local DNS server being queried?

10.138.0.2 is trying to query the local DNS server with address 169.254.169.254

Odin ID: jtn7

TCP

- What packet numbers in the trace correspond to the initial TCP handshake for the web request?

Packets 773, 774, and 775

- How long does it take to perform the initial TCP handshake?

It took only 0.001459 seconds to perform the initial TCP handshake.

HTTP

- What packet numbers in the trace correspond to the actual HTTP request and response?

Packets 776 and 779.

- How long does it take to process the HTTP request after the handshake?

It took 0.000496 seconds.