Addison Wurtz
CS 530
notebooks/Week2

Lab Notebook – Week 1

Table of Contents:

- 02.1: TCP, HTTP
  - TCP #1: Sockets
    - Chat-GPT Prompt and Command to find IPv4 TCP Sockets in LISTEN State
    - Command Output
    - Service That Can Be Contacted From Any Interface on the Machine
    - Service That Can Only Be Contacted By Local Process
    - Run ss -tln4 on linux.cs.pdx.edu
    - Services linux.cs.pdx.edu Provides for External Access
    - Chat-GPT Prompt and Response for Using lsof
    - Output of lsof Command
  - TCP #2: Throughput
    - iperf
  - HTTP #3: Requests
    - Browser Requests for http://google.com
    - 1st Request
    - 2nd Request
    - 3rd Request
    - Location Response Header:
    - Set-Cookie:
    - Cookie:
    - Asynchronous HTTP Requests
- 02.2: DNS, Recap
  - DNS Reconnaissance #1 (dig)
    - Chat-GPT dig Command
    - Dig for www.pdx.edu records
    - Using IP Addresses from Records
  - Mashimaro.cs.pdx.edu
  - Iterative DNS Lookups
  - Reverse DNS Lookups
    - Chat-GPT dig/egrep/awk for IPv4 addresses the espn.go.com points to:
    - IPv4 addresses that espn.go.com points to:
    - DNS names the espn.go.com points to:
  - Host Enumeration
  - Geographic DNS #2
    - Geographic Locations of IP Addresses:
    - Using dig to resolve www.google.com from different DNS servers:

Addison Wurtz
CS 530
notebooks/Week2

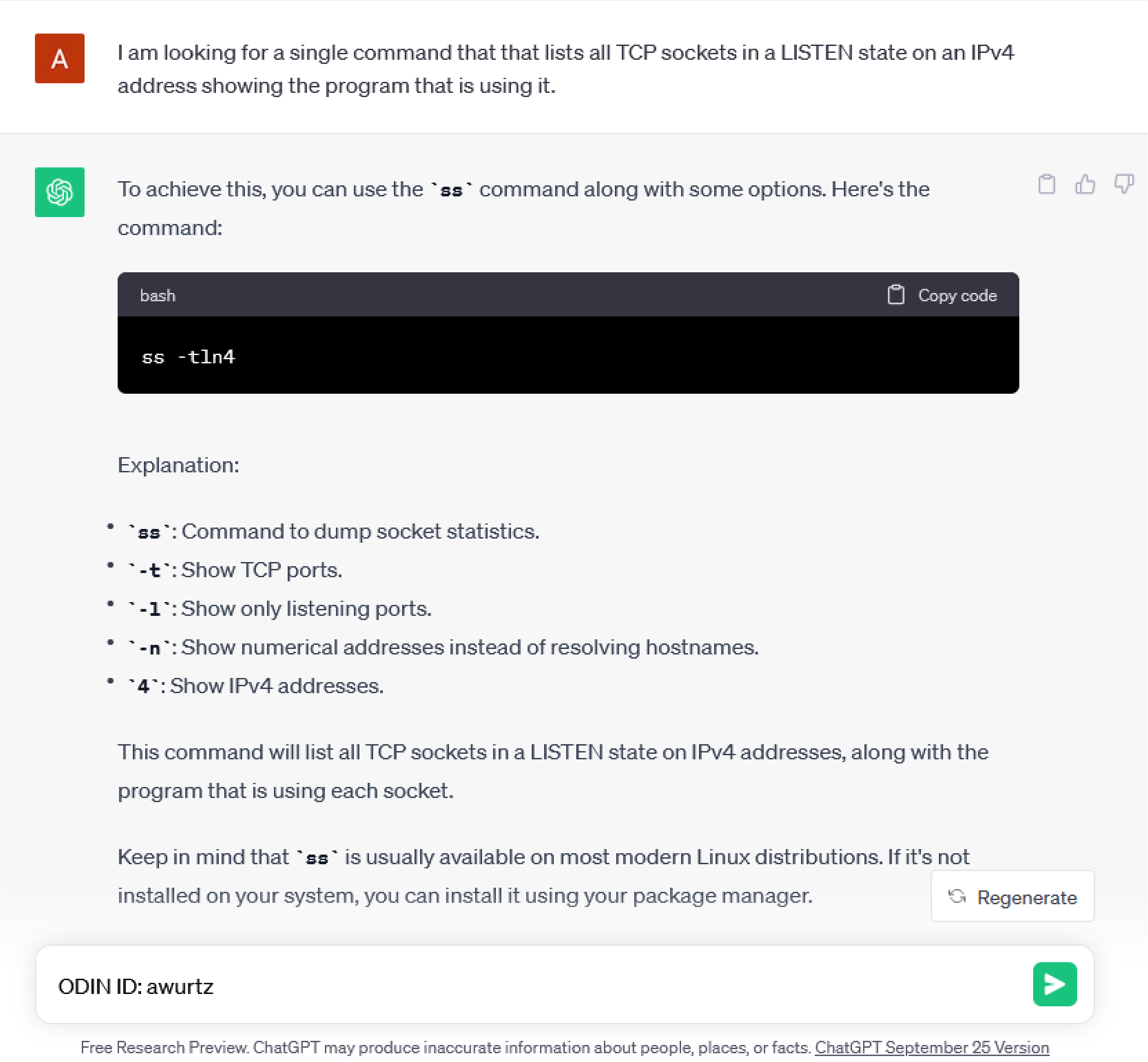
    - Traceroute:
- Wireshark Lab #3
    - Wireshark Packet Dump
    - First Packet Trace: Does the destination MAC address correspond to an interface on the VM, an interface on the default router or an interface on Google's web site?
    - Second Packet Trace: Does the destination MAC address correspond to an interface on the VM, an interface on the default router or an interface on Google's web site?
- Network Recap Lab #4
    - IP of awurtz.oregonctf.org
    - Collect Trace
    - Analyze Trace
    - Arp
    - DNS
    - TCP
    - HTTP

Addison Wurtz
CS 530
notebooks/Week2

02.1: TCP, HTTP

TCP #1: Sockets

Chat-GPT Prompt and Command to find IPv4 TCP Sockets in LISTEN State



A

I am looking for a single command that that lists all TCP sockets in a LISTEN state on an IPv4 address showing the program that is using it.

To achieve this, you can use the `ss` command along with some options. Here's the command:

```bash
ss -tln4
```

Explanation:

- `ss`: Command to dump socket statistics.
- `-t`: Show TCP ports.
- `-l`: Show only listening ports.
- `-n`: Show numerical addresses instead of resolving hostnames.
- `4`: Show IPv4 addresses.

This command will list all TCP sockets in a LISTEN state on IPv4 addresses, along with the program that is using each socket.

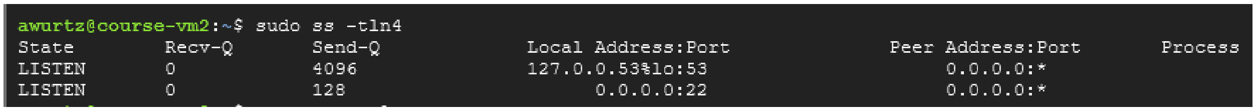
Keep in mind that `ss` is usually available on most modern Linux distributions. If it's not installed on your system, you can install it using your package manager.

ODIN ID: awurtz

Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT September 25 Version

Addison Wurtz
CS 530
notebooks/Week2

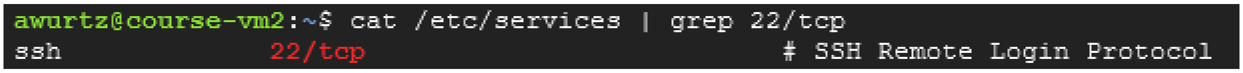
Command Output



```
awurtz@course-vm2:~$ sudo ss -tln4
State        Recv-Q       Send-Q            Local Address:Port              Peer Address:Port      Process
LISTEN       0            4096              127.0.0.53%lo:53                     0.0.0.0:*
LISTEN       0            128                     0.0.0.0:22                     0.0.0.0:*
```

Service That Can Be Contacted From Any Interface on the Machine

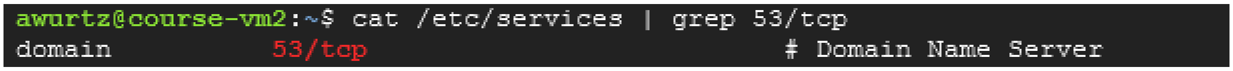
The SSH program can be contacted by any interface on the machine:



```
awurtz@course-vm2:~$ cat /etc/services | grep 22/tcp
ssh                 22/tcp                          # SSH Remote Login Protocol
```

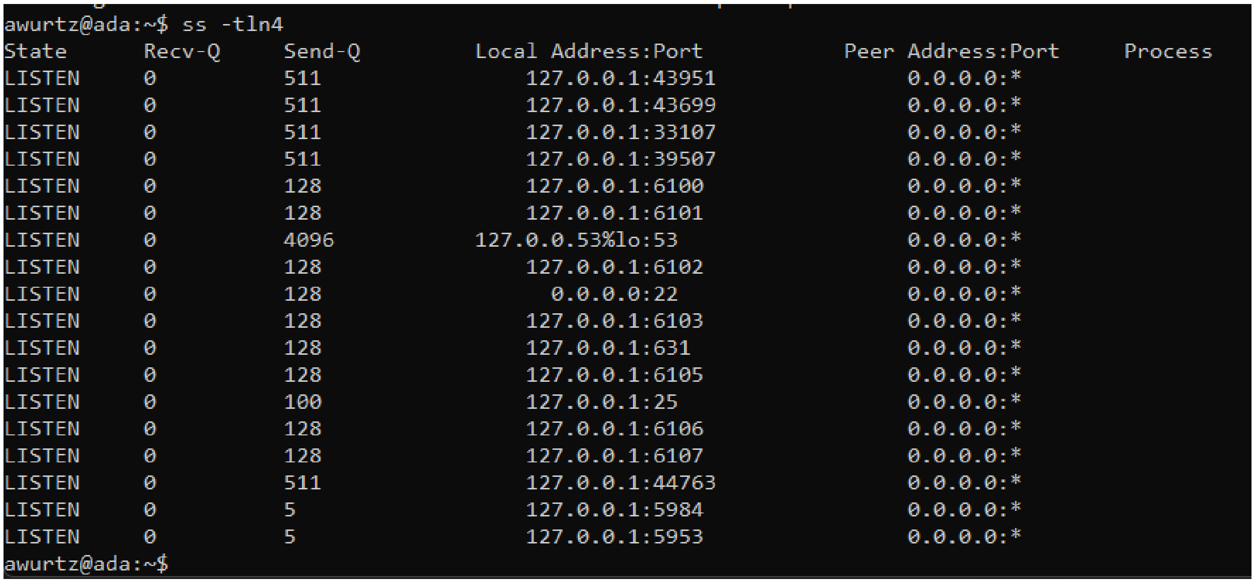
Service That Can Only Be Contacted By Local Process

The domain (DNS) program can only be contacted by local processes:



```
awurtz@course-vm2:~$ cat /etc/services | grep 53/tcp
domain              53/tcp                          # Domain Name Server
```

Run ss -tln4 on linux.cs.pdx.edu

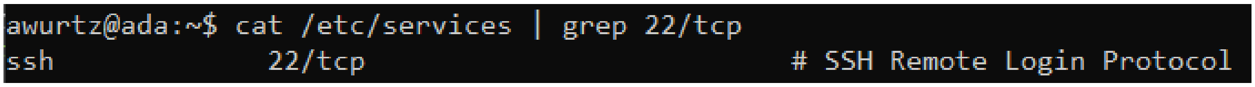


```
awurtz@ada:~$ ss -tln4
State      Recv-Q     Send-Q         Local Address:Port          Peer Address:Port     Process
LISTEN     0          511                127.0.0.1:43951              0.0.0.0:*
LISTEN     0          511                127.0.0.1:43699              0.0.0.0:*
LISTEN     0          511                127.0.0.1:33107              0.0.0.0:*
LISTEN     0          511                127.0.0.1:39507              0.0.0.0:*
LISTEN     0          128                127.0.0.1:6100               0.0.0.0:*
LISTEN     0          128                127.0.0.1:6101               0.0.0.0:*
LISTEN     0          4096           127.0.0.53%lo:53                 0.0.0.0:*
LISTEN     0          128                127.0.0.1:6102               0.0.0.0:*
LISTEN     0          128                  0.0.0.0:22                 0.0.0.0:*
LISTEN     0          128                127.0.0.1:6103               0.0.0.0:*
LISTEN     0          128                127.0.0.1:631                0.0.0.0:*
LISTEN     0          128                127.0.0.1:6105               0.0.0.0:*
LISTEN     0          100                127.0.0.1:25                 0.0.0.0:*
LISTEN     0          128                127.0.0.1:6106               0.0.0.0:*
LISTEN     0          128                127.0.0.1:6107               0.0.0.0:*
LISTEN     0          511                127.0.0.1:44763              0.0.0.0:*
LISTEN     0          5                  127.0.0.1:5984               0.0.0.0:*
LISTEN     0          5                  127.0.0.1:5953               0.0.0.0:*
awurtz@ada:~$
```

Addison Wurtz
CS 530
notebooks/Week2

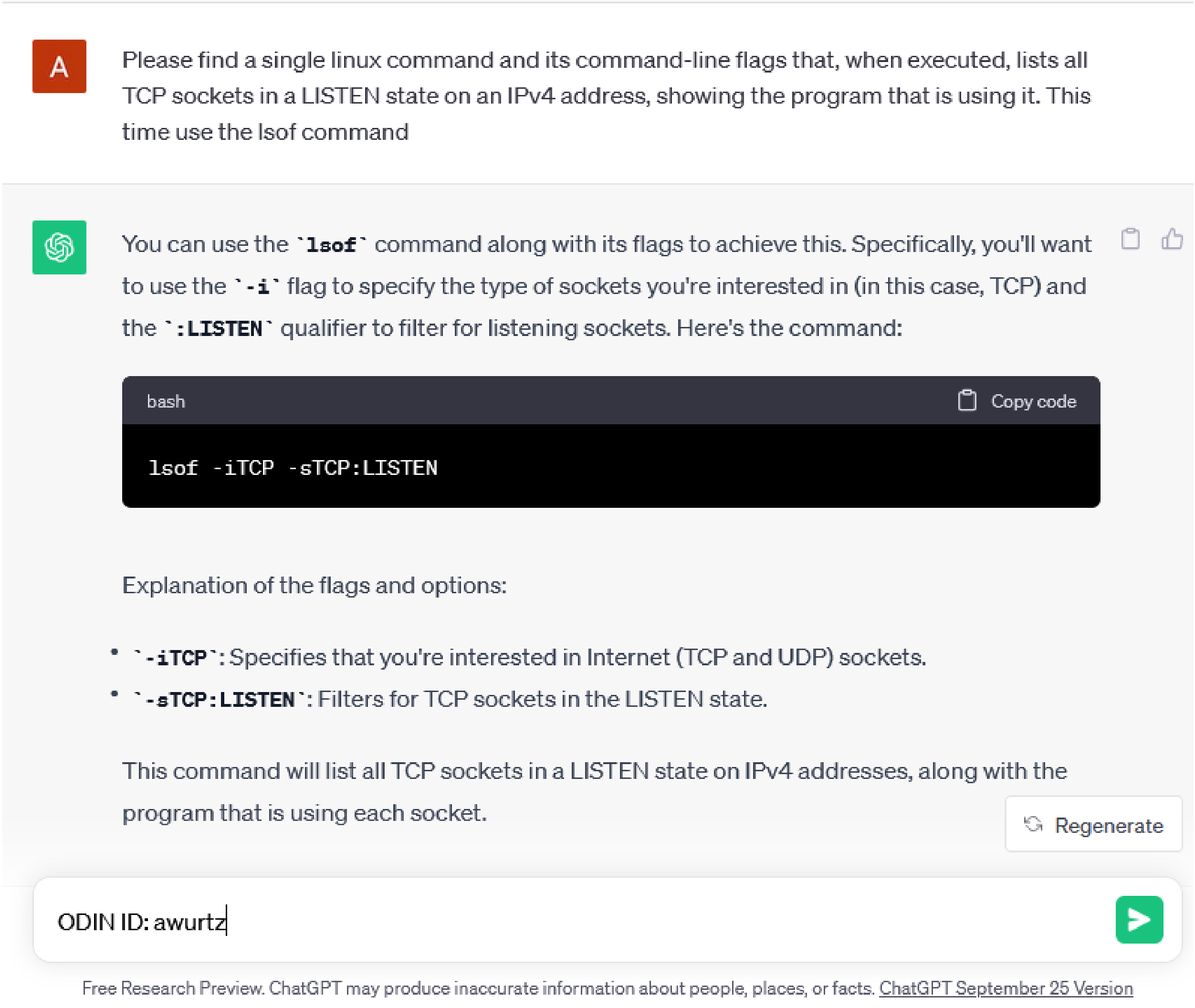
Services linux.cs.pdx.edu Provides for External Access

Only ssh is listening externally. All other services are listening locally.



```
awurtz@ada:~$ cat /etc/services | grep 22/tcp
ssh             22/tcp                          # SSH Remote Login Protocol
```

Chat-GPT Prompt and Response for Using lsof



Please find a single linux 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. This time use the lsof command

You can use the `lsof` command along with its flags to achieve this. Specifically, you'll want to use the `-i` flag to specify the type of sockets you're interested in (in this case, TCP) and the `:LISTEN` qualifier to filter for listening sockets. Here's the command:

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

Explanation of the flags and options:

- `-iTCP`: Specifies that you're interested in Internet (TCP and UDP) sockets.
- `-sTCP:LISTEN`: Filters for TCP sockets in the LISTEN state.

This command will list all TCP sockets in a LISTEN state on IPv4 addresses, along with the program that is using each socket.

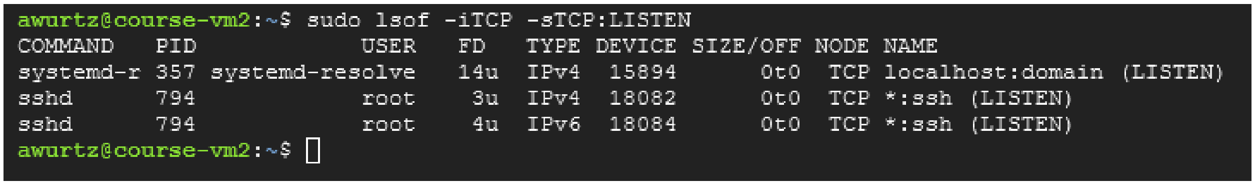
ODIN ID: awurtz

Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT September 25 Version

---

Addison Wurtz
CS 530
notebooks/Week2

Output of lsof Command

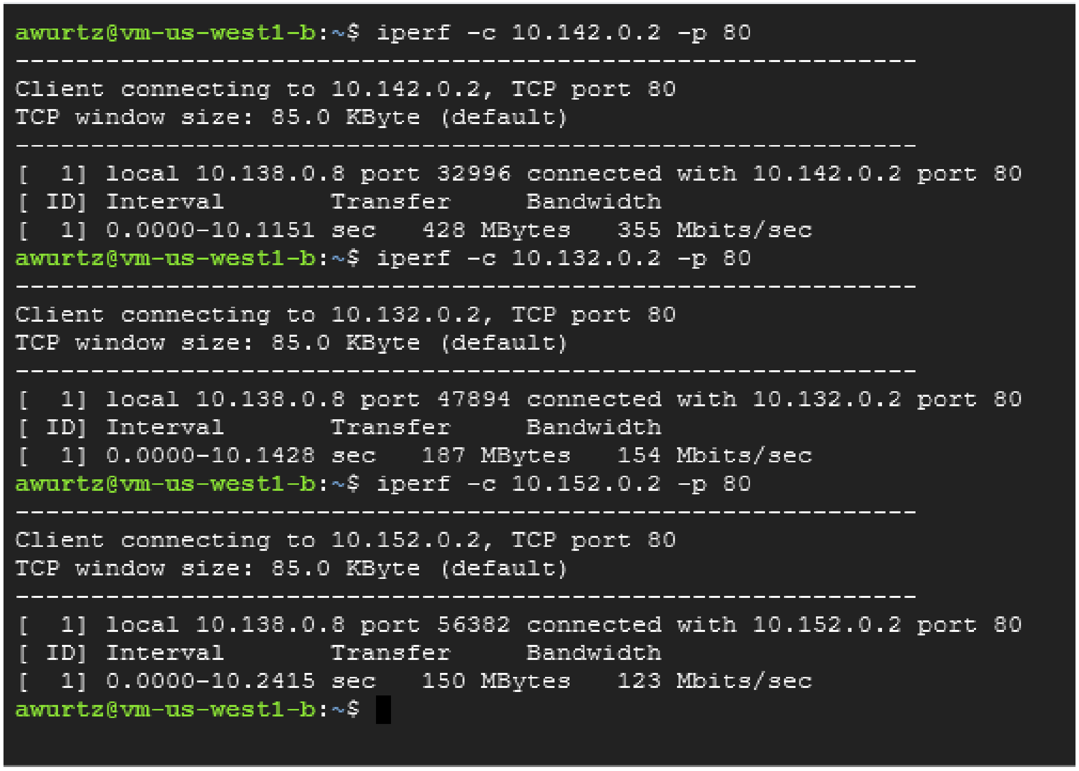


```
awurtz@course-vm2:~$ sudo lsof -iTCP -sTCP:LISTEN
COMMAND   PID            USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
systemd-r 357 systemd-resolve   14u  IPv4  15894      0t0  TCP localhost:domain (LISTEN)
sshd      794            root    3u  IPv4  18082      0t0  TCP *:ssh (LISTEN)
sshd      794            root    4u  IPv6  18084      0t0  TCP *:ssh (LISTEN)
awurtz@course-vm2:~$ 
```

TCP #2: Throughput

iperf

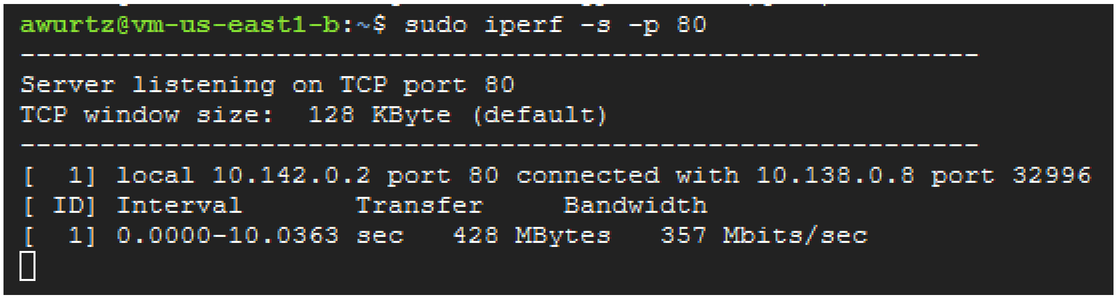
Running iperf from vm-us-west1-b against vms in US East, Europe, and Australia:



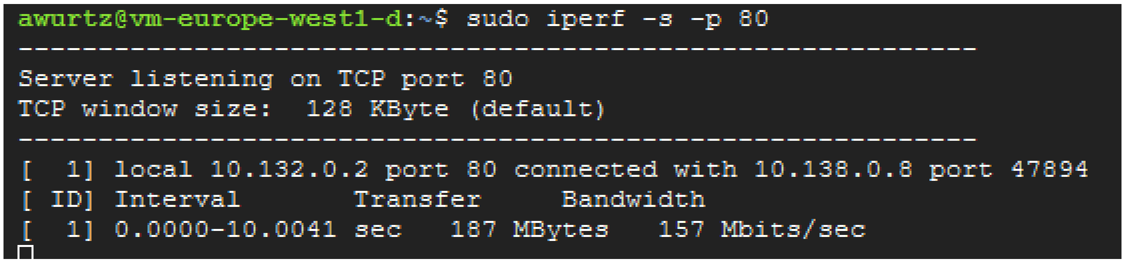
```
awurtz@vm-us-west1-b:~$ iperf -c 10.142.0.2 -p 80
------------------------------------------------------------
Client connecting to 10.142.0.2, TCP port 80
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  1] local 10.138.0.8 port 32996 connected with 10.142.0.2 port 80
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.1151 sec   428 MBytes   355 Mbits/sec
awurtz@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 47894 connected with 10.132.0.2 port 80
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.1428 sec   187 MBytes   154 Mbits/sec
awurtz@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 56382 connected with 10.152.0.2 port 80
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.2415 sec   150 MBytes   123 Mbits/sec
awurtz@vm-us-west1-b:~$ 
```

The bandwidth available between the us-west1-b VM and the US East VM are much higher than with the Europe and Australia VMs because it is physically much closer, meaning that more data can be transferred in the same time interval. There is slightly more bandwidth between US West and Europe than between US West and Australia, similarly indicating that the Europe datacenter must be closer.

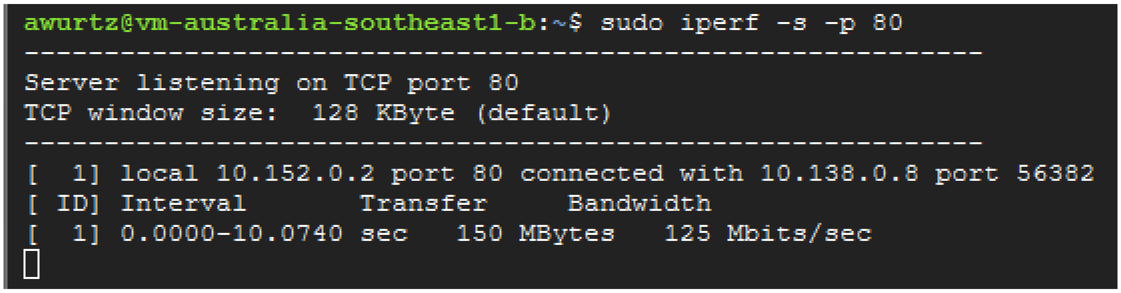
Addison Wurtz
CS 530
notebooks/Week2



```
awurtz@vm-us-east1-b:~$ sudo iperf -s -p 80
------------------------------------------------------------
Server listening on TCP port 80
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  1] local 10.142.0.2 port 80 connected with 10.138.0.8 port 32996
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0363 sec   428 MBytes   357 Mbits/sec
```



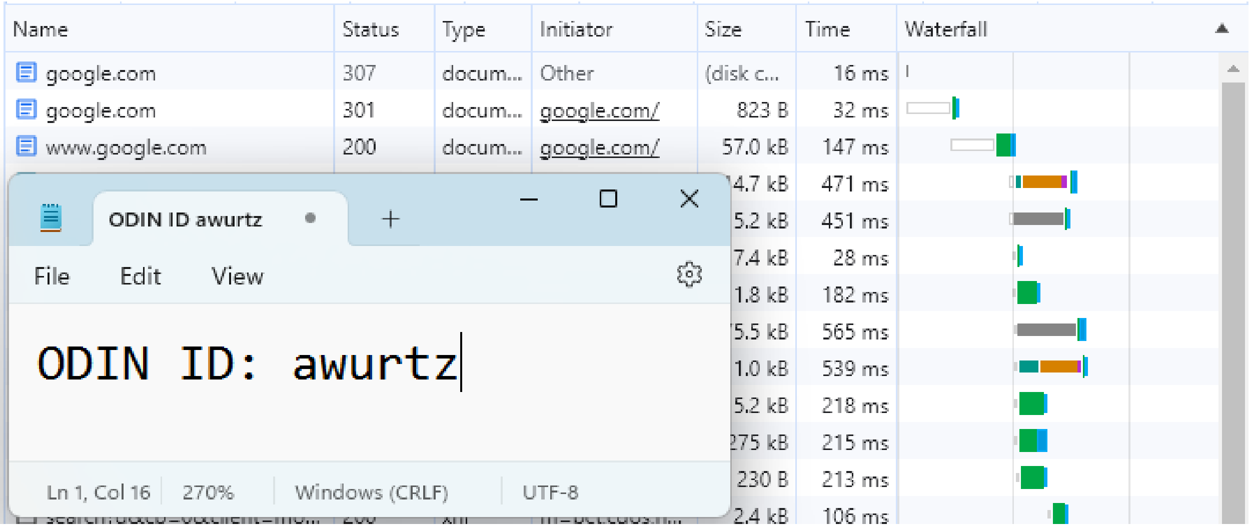
```
awurtz@vm-europe-west1-d:~$ sudo iperf -s -p 80
------------------------------------------------------------
Server listening on TCP port 80
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  1] local 10.132.0.2 port 80 connected with 10.138.0.8 port 47894
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0041 sec   187 MBytes   157 Mbits/sec
```



```
awurtz@vm-australia-southeast1-b:~$ sudo iperf -s -p 80
------------------------------------------------------------
Server listening on TCP port 80
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  1] local 10.152.0.2 port 80 connected with 10.138.0.8 port 56382
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.0740 sec   150 MBytes   125 Mbits/sec
```

HTTP #3: Requests

Browser Requests for http://google.com



Name	Status	Type	Initiator	Size	Time	Waterfall
google.com	307	docum...	Other	(disk c...	16 ms	
google.com	301	docum...	google.com/	823 B	32 ms	
www.google.com	200	docum...	google.com/	57.0 kB	147 ms	
				4.7 kB	471 ms	
				5.2 kB	451 ms	
				7.4 kB	28 ms	
				1.8 kB	182 ms	
				5.5 kB	565 ms	
				1.0 kB	539 ms	
				5.2 kB	218 ms	
				275 kB	215 ms	
				230 B	213 ms	
				2.4 kB	106 ms	

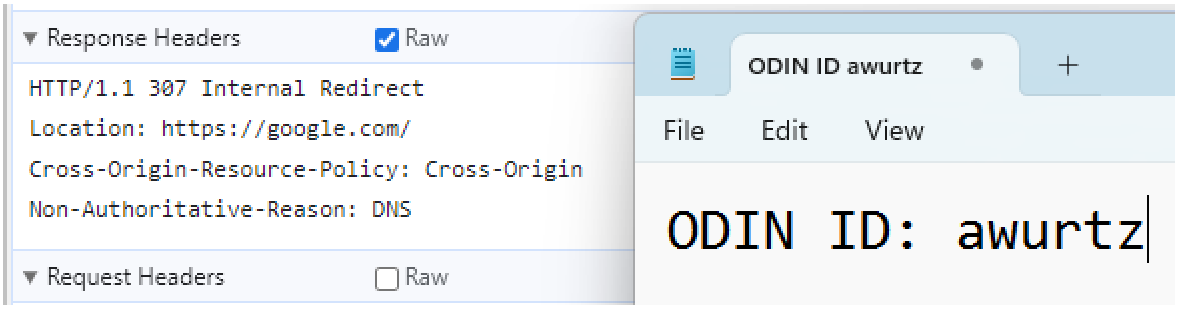
ODIN ID: awurtz

Addison Wurtz
CS 530
notebooks/Week2

1st Request

Request URL: http://google.com/

HTTP Status Code: “307 Internal Redirect (from disk cache)”
Indicates requested resource can be found at the new URI specified in the location response header (but only temporarily).



```
HTTP/1.1 307 Internal Redirect
Location: https://google.com/
Cross-Origin-Resource-Policy: Cross-Origin
Non-Authoritative-Reason: DNS
```

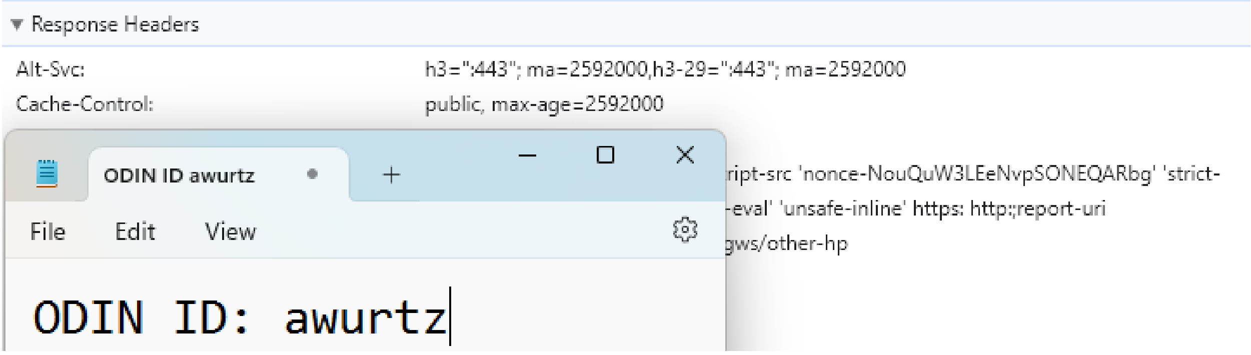
ODIN ID: awurtz

2nd Request

Request URL: https://google.com/

HTTP Status Code: 301 Moved Permanently
Indicates that the requested resource has been definitively moved to the URL given by the Location headers.

Alt-Svc indicates HTTP/3



```
Alt-Svc:          h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
Cache-Control:    public, max-age=2592000
                  ript-src 'nonce-NouQuW3LEeNvpSONEQARbg' 'strict-
                  -eval' 'unsafe-inline' https: http:;report-uri
                  gws/other-hp
```

ODIN ID: awurtz

Addison Wurtz
CS 530
notebooks/Week2

3rd Request

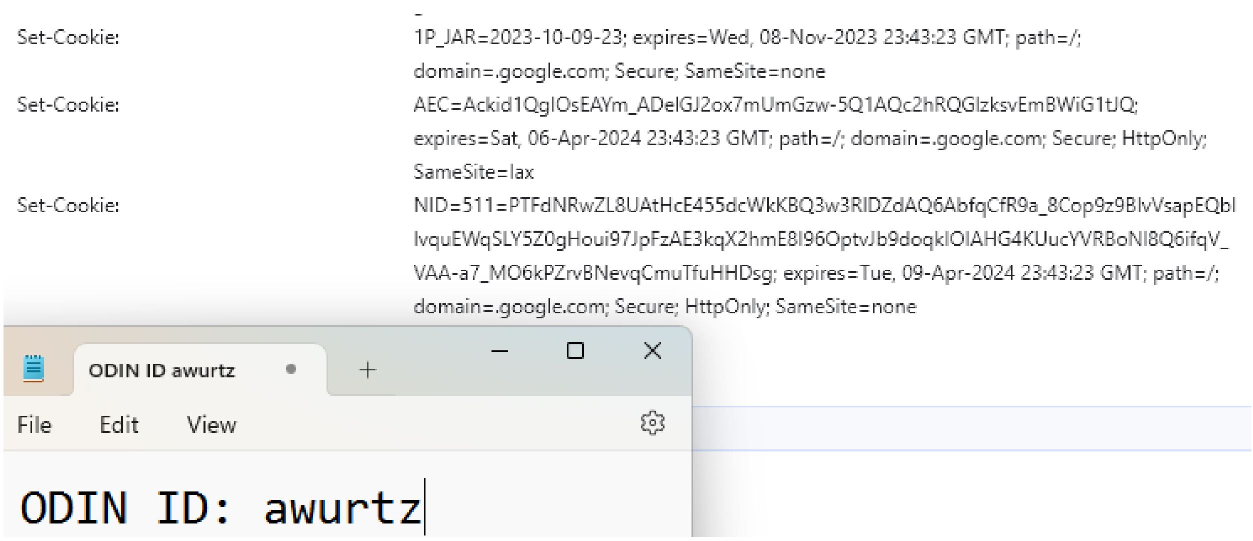
Request URL: https://www.google.com/

HTTP Status Code: 200 OK
Indicates that the request has succeeded.

Location Response Header:

The first redirection goes to https://google.com (https instead of http).
The second redirection goes to https://www.google.com (adds www.).

Set-Cookie:

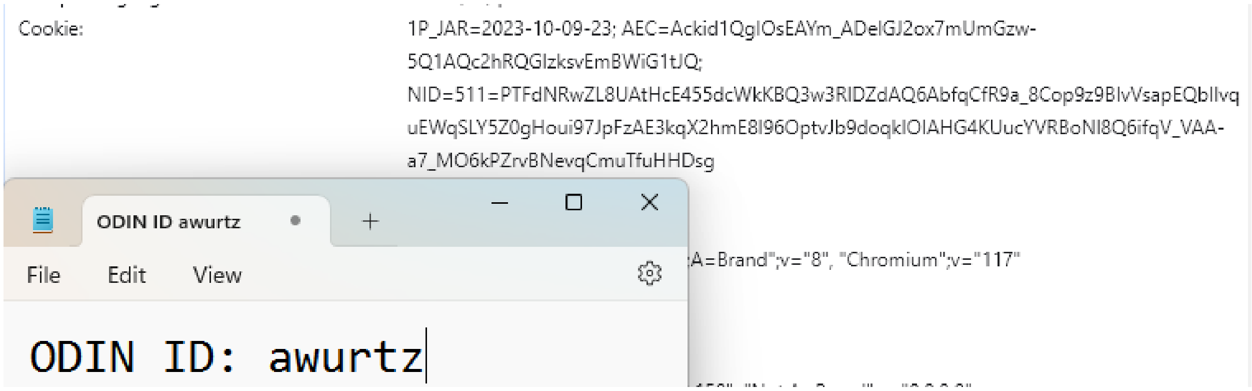


Set-Cookie:	1P_JAR=2023-10-09-23; expires=Wed, 08-Nov-2023 23:43:23 GMT; path=/; domain=.google.com; Secure; SameSite=none
Set-Cookie:	AEC=Ackid1QglOsEAYm_ADelGJ2ox7mUmGzw-5Q1AQc2hRQGlzksvEmBWiG1tJQ; expires=Sat, 06-Apr-2024 23:43:23 GMT; path=/; domain=.google.com; Secure; HttpOnly; SameSite=lax
Set-Cookie:	NID=511=PTFdNRwZL8UAtHcE455dcWkKBQ3w3RIDZdAQ6AbfqCfR9a_8Cop9z9BlvVsapEQbllvquEWqSLY5Z0gHoui97JpFzAE3kqX2hmE8l96OptvJb9doqklOIAHG4KUucYVRBoNI8Q6ifqV_VAA-a7_MO6kPZrvBNevqCmuTfuHHDsg; expires=Tue, 09-Apr-2024 23:43:23 GMT; path=/; domain=.google.com; Secure; HttpOnly; SameSite=none

ODIN ID: awurtz

Addison Wurtz
CS 530
notebooks/Week2

Cookie:



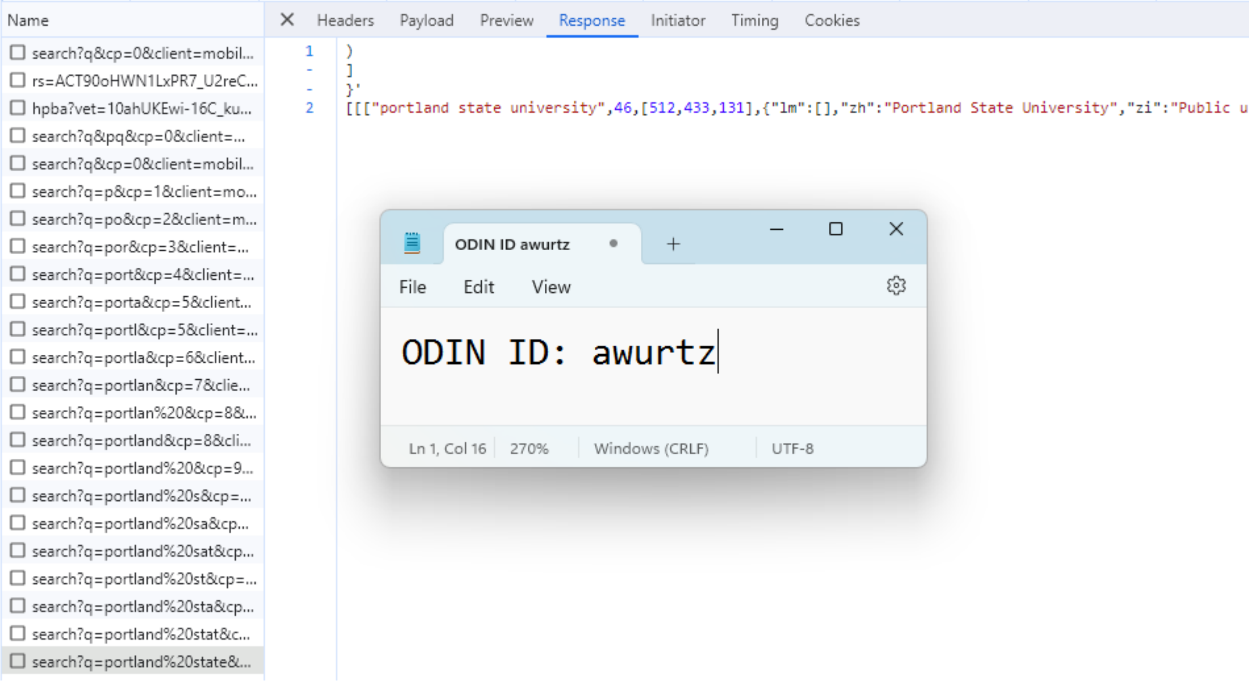
Cookie:

```
1P_JAR=2023-10-09-23; AEC=Ackid1QgIOsEAYm_ADelGJ2ox7mUmGzw-5Q1AQc2hRQGIzksvEmBWiG1tJQ; NID=511=PTFdNRwZL8UAtHcE455dcWkKBQ3w3RIDZdAQ6AbfqCfR9a_8Cop9z9BIvVsapEQbIlvquEWqSLY5Z0gHoui97JpFzAE3kqX2hmE8I96OptvJb9doqkIOIAHG4KUucYVRBoNI8Q6ifqV_VAA-a7_MO6kPZrvBNevqCmuTfuHHDsg
```

```
;A=Brand";v="8", "Chromium";v="117"
```

ODIN ID: awurtz

Asynchronous HTTP Requests



☐ search?q&cp=0&client=mobil...
☐ rs=ACT90oHWN1LxPR7_U2reC...
☐ hpba?vet=10ahUKEwi-16C_ku...
☐ search?q&pq&cp=0&client=...
☐ search?q&cp=0&client=mobil...
☐ search?q=p&cp=1&client=mo...
☐ search?q=po&cp=2&client=m...
☐ search?q=por&cp=3&client=...
☐ search?q=port&cp=4&client=...
☐ search?q=porta&cp=5&client...
☐ search?q=portl&cp=5&client=...
☐ search?q=portla&cp=6&client...
☐ search?q=portlan&cp=7&clie...
☐ search?q=portlan%20&cp=8&...
☐ search?q=portland&cp=8&cli...
☐ search?q=portland%20&cp=9...
☐ search?q=portland%20s&cp=...
☐ search?q=portland%20sa&cp...
☐ search?q=portland%20sat&cp...
☐ search?q=portland%20st&cp=...
☐ search?q=portland%20sta&cp...
☐ search?q=portland%20stat&c...
☐ search?q=portland%20state&...

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

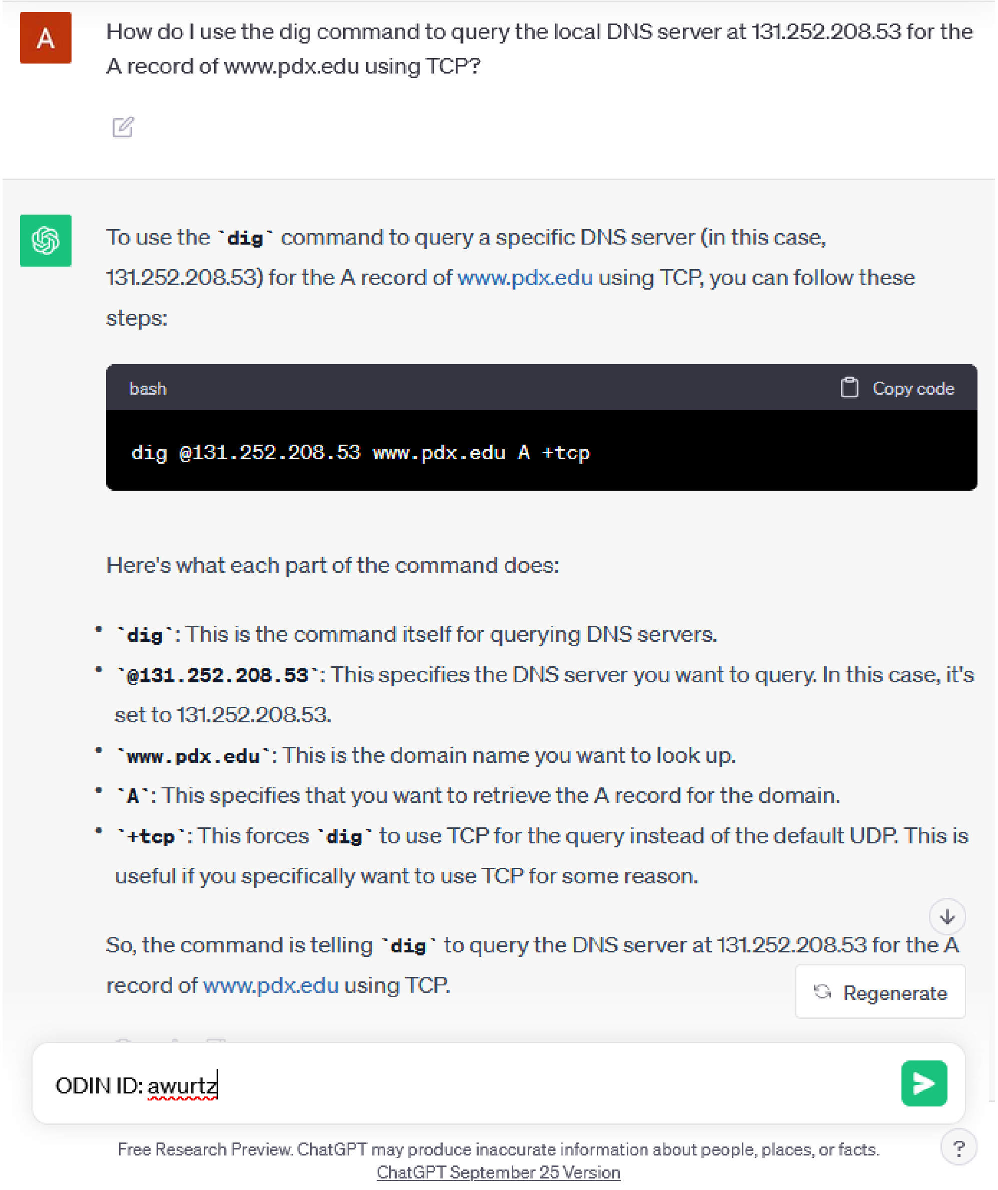
ODIN ID: awurtz

Addison Wurtz
CS 530
notebooks/Week2

02.2: DNS, Recap

DNS Reconnaissance #1 (dig)

Chat-GPT dig Command



How do I use the dig command to query the local DNS server at 131.252.208.53 for the A record of www.pdx.edu using TCP?

To use the `dig` command to query a specific DNS server (in this case, 131.252.208.53) for the A record of www.pdx.edu using TCP, you can follow these steps:

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

Here's what each part of the command does:

- `dig`: This is the command itself for querying DNS servers.
- `@131.252.208.53`: This specifies the DNS server you want to query. In this case, it's set to 131.252.208.53.
- `www.pdx.edu`: This is the domain name you want to look up.
- `A`: This specifies that you want to retrieve the A record for the domain.
- `+tcp`: This forces `dig` to use TCP for the query instead of the default UDP. This is useful if you specifically want to use TCP for some reason.

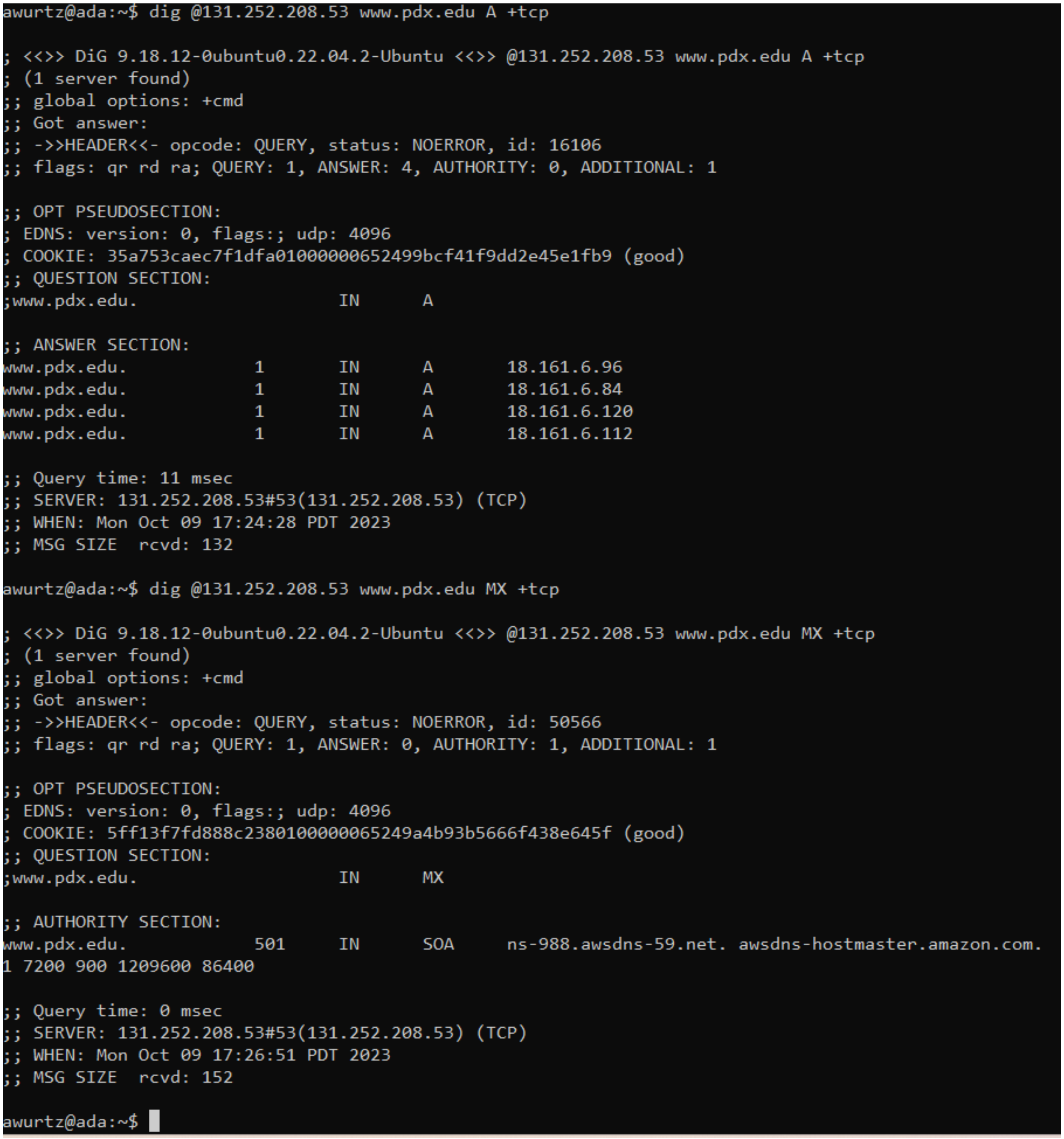
So, the command is telling `dig` to query the DNS server at 131.252.208.53 for the A record of www.pdx.edu using TCP.

ODIN ID: awurtz

Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT September 25 Version

Addison Wurtz
CS 530
notebooks/Week2

Dig for www.pdx.edu records

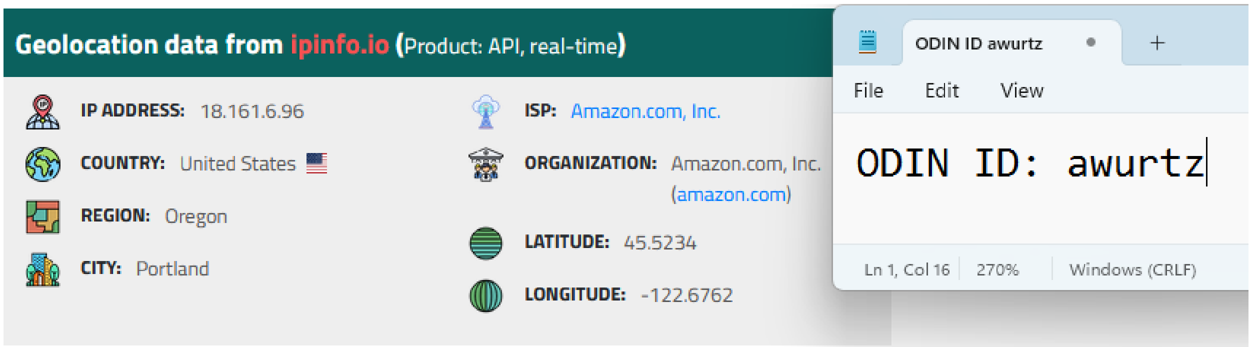


```
awurtz@ada:~$ dig @131.252.208.53 www.pdx.edu A +tcp

; <<>> DiG 9.18.12-0ubuntu0.22.04.2-Ubuntu <<>> @131.252.208.53 www.pdx.edu A +tcp
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16106
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 35a753caec7f1dfa01000000652499bcf41f9dd2e45e1fb9 (good)
;; QUESTION SECTION:
;www.pdx.edu.                   IN      A

;; ANSWER SECTION:
www.pdx.edu.            1       IN      A       18.161.6.96
www.pdx.edu.            1       IN      A       18.161.6.84
www.pdx.edu.            1       IN      A       18.161.6.120
www.pdx.edu.            1       IN      A       18.161.6.112

;; Query time: 11 msec
;; SERVER: 131.252.208.53#53(131.252.208.53) (TCP)
;; WHEN: Mon Oct 09 17:24:28 PDT 2023
;; MSG SIZE  rcvd: 132

awurtz@ada:~$ dig @131.252.208.53 www.pdx.edu MX +tcp

; <<>> DiG 9.18.12-0ubuntu0.22.04.2-Ubuntu <<>> @131.252.208.53 www.pdx.edu MX +tcp
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50566
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 5ff13f7fd888c2380100000065249a4b93b5666f438e645f (good)
;; QUESTION SECTION:
;www.pdx.edu.                   IN      MX

;; AUTHORITY SECTION:
www.pdx.edu.            501     IN      SOA     ns-988.awsdns-59.net. awsdns-hostmaster.amazon.com.
1 7200 900 1209600 86400

;; Query time: 0 msec
;; SERVER: 131.252.208.53#53(131.252.208.53) (TCP)
;; WHEN: Mon Oct 09 17:26:51 PDT 2023
;; MSG SIZE  rcvd: 152

awurtz@ada:~$ █
```

Using IP Addresses from Records

Amazon hosts www.pdx.edu:

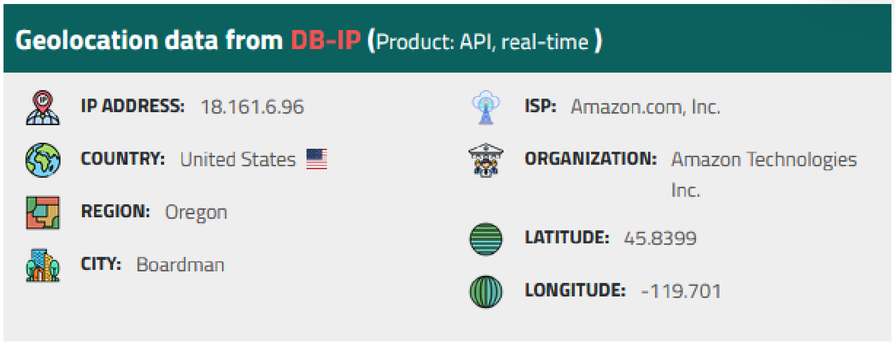
Addison Wurtz
CS 530
notebooks/Week2



Geolocation data from ipinfo.io (Product: API, real-time)

IP ADDRESS: 18.161.6.96
COUNTRY: United States
REGION: Oregon
CITY: Portland
ISP: Amazon.com, Inc.
ORGANIZATION: Amazon.com, Inc. (amazon.com)
LATITUDE: 45.5234
LONGITUDE: -122.6762

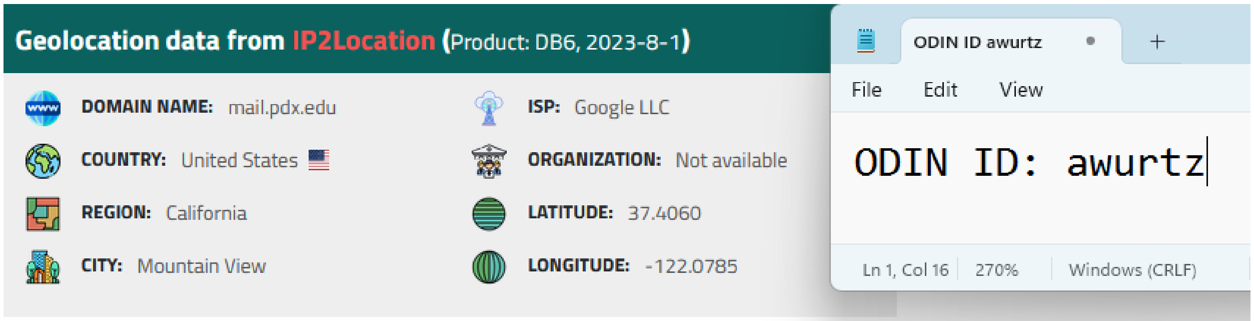
ODIN ID: awurtz



Geolocation data from DB-IP (Product: API, real-time)

IP ADDRESS: 18.161.6.96
COUNTRY: United States
REGION: Oregon
CITY: Boardman
ISP: Amazon.com, Inc.
ORGANIZATION: Amazon Technologies Inc.
LATITUDE: 45.8399
LONGITUDE: -119.701

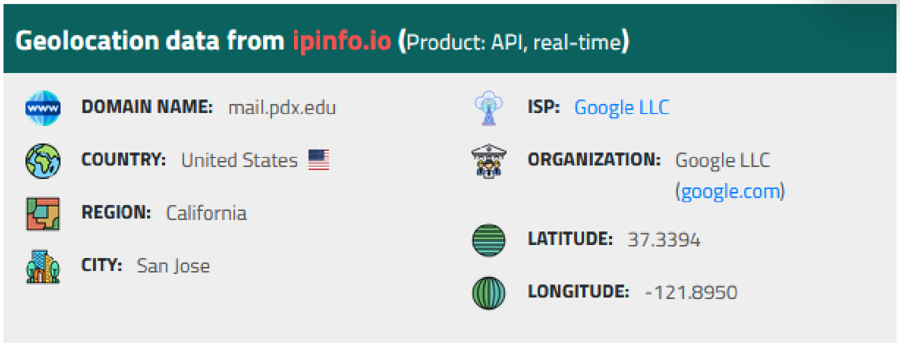
Google handles mail for pdx.edu



Geolocation data from IP2Location (Product: DB6, 2023-8-1)

DOMAIN NAME: mail.pdx.edu
COUNTRY: United States
REGION: California
CITY: Mountain View
ISP: Google LLC
ORGANIZATION: Not available
LATITUDE: 37.4060
LONGITUDE: -122.0785

ODIN ID: awurtz

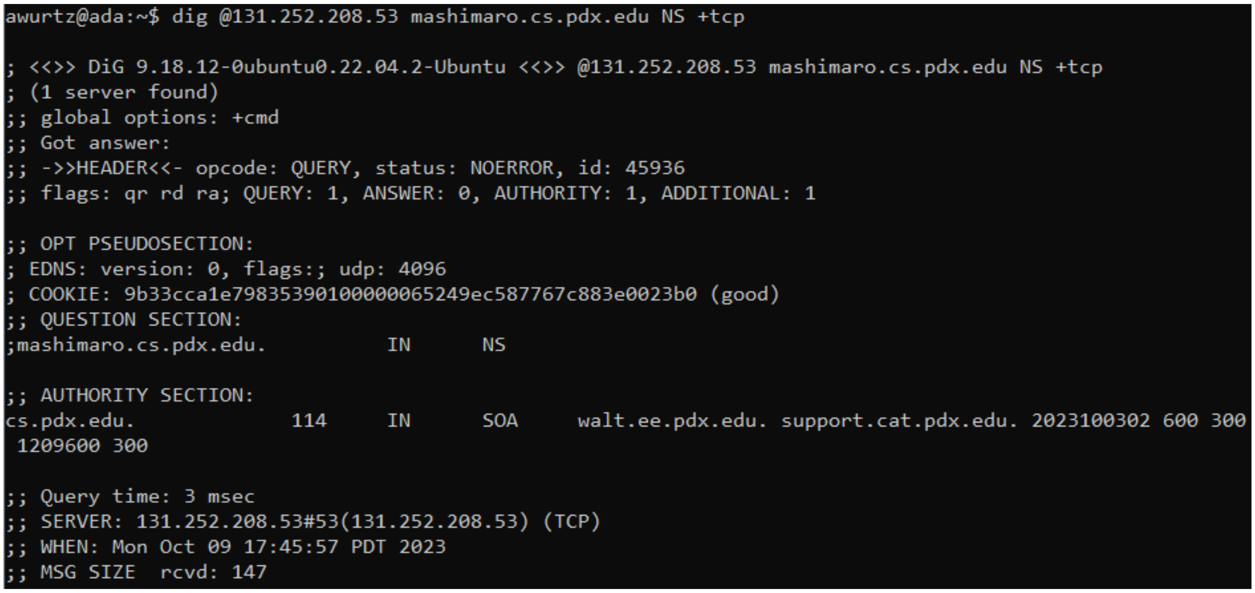


Geolocation data from ipinfo.io (Product: API, real-time)

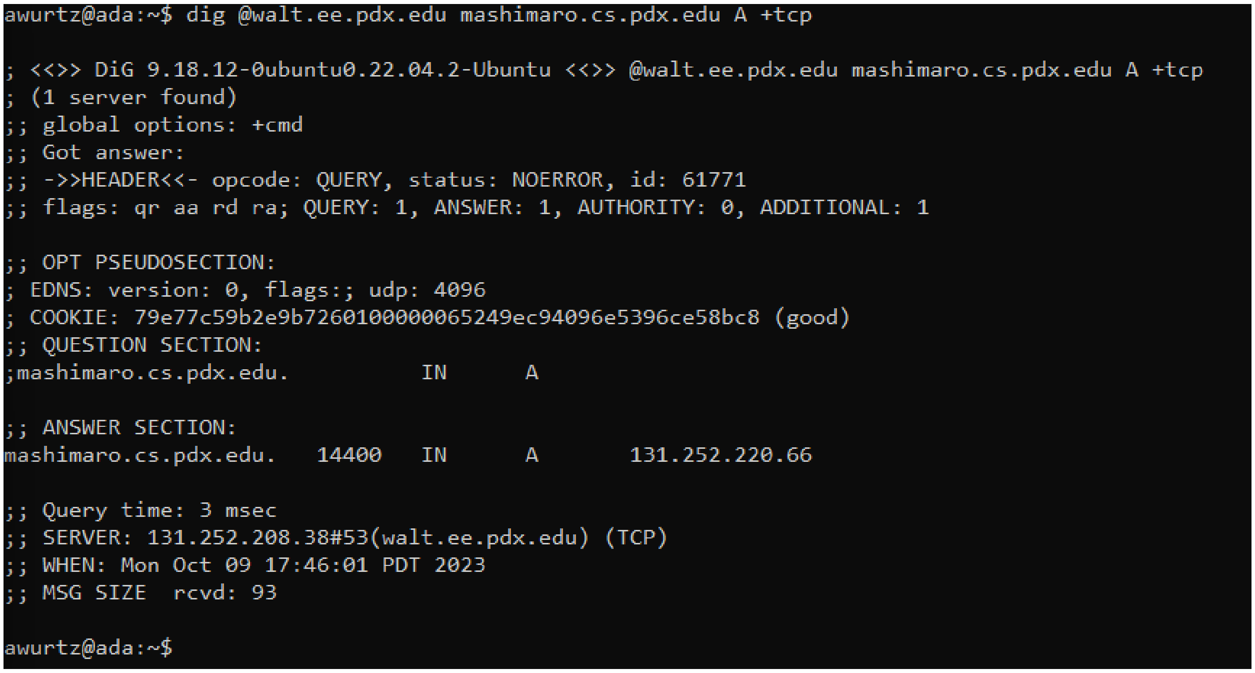
DOMAIN NAME: mail.pdx.edu
COUNTRY: United States
REGION: California
CITY: San Jose
ISP: Google LLC
ORGANIZATION: Google LLC (google.com)
LATITUDE: 37.3394
LONGITUDE: -121.8950

Addison Wurtz
CS 530
notebooks/Week2

Mashimaro.cs.pdx.edu



```
awurtz@ada:~$ dig @131.252.208.53 mashimaro.cs.pdx.edu NS +tcp

; <<>> DiG 9.18.12-0ubuntu0.22.04.2-Ubuntu <<>> @131.252.208.53 mashimaro.cs.pdx.edu NS +tcp
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45936
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 9b33cca1e798353901000000065249ec587767c883e0023b0 (good)
;; QUESTION SECTION:
;mashimaro.cs.pdx.edu.          IN      NS

;; AUTHORITY SECTION:
cs.pdx.edu.             114     IN      SOA     walt.ee.pdx.edu. support.cat.pdx.edu. 2023100302 600 300
 1209600 300

;; Query time: 3 msec
;; SERVER: 131.252.208.53#53(131.252.208.53) (TCP)
;; WHEN: Mon Oct 09 17:45:57 PDT 2023
;; MSG SIZE  rcvd: 147
```



```
awurtz@ada:~$ dig @walt.ee.pdx.edu mashimaro.cs.pdx.edu A +tcp

; <<>> DiG 9.18.12-0ubuntu0.22.04.2-Ubuntu <<>> @walt.ee.pdx.edu mashimaro.cs.pdx.edu A +tcp
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61771
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 79e77c59b2e9b72601000000065249ec94096e5396ce58bc8 (good)
;; QUESTION SECTION:
;mashimaro.cs.pdx.edu.          IN      A

;; ANSWER SECTION:
mashimaro.cs.pdx.edu.   14400   IN      A       131.252.220.66

;; Query time: 3 msec
;; SERVER: 131.252.208.38#53(walt.ee.pdx.edu) (TCP)
;; WHEN: Mon Oct 09 17:46:01 PDT 2023
;; MSG SIZE  rcvd: 93

awurtz@ada:~$
```

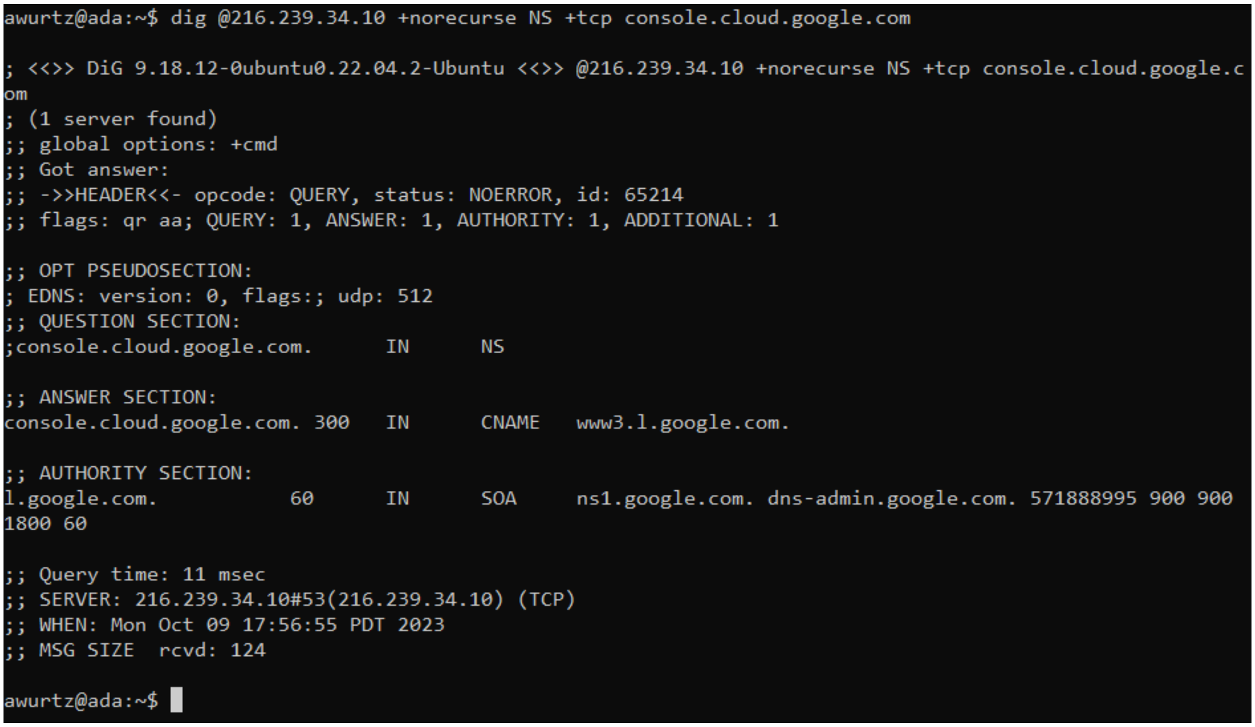
Iterative DNS Lookups

Commands:

- dig
- dig @192.5.5.241 +norecurse NS +tcp console.cloud.google.com
- dig @192.12.94.30 +norecurse NS +tcp console.cloud.google.com

Addison Wurtz
CS 530
notebooks/Week2

- dig @216.239.34.10 +norecurse NS +tcp console.cloud.google.com

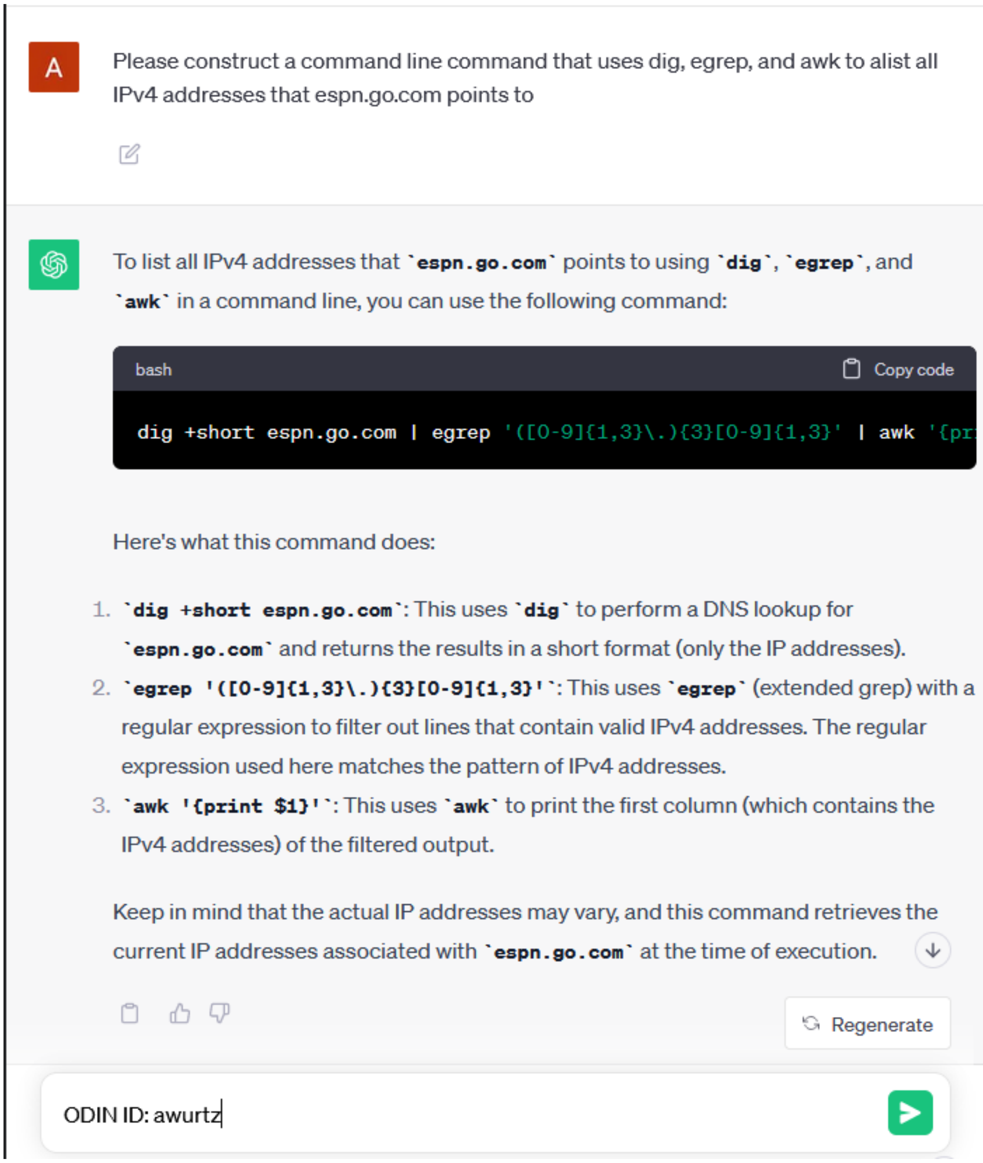


```
awurtz@ada:~$ dig @216.239.34.10 +norecurse NS +tcp console.cloud.google.com

; <<>> DiG 9.18.12-0ubuntu0.22.04.2-Ubuntu <<>> @216.239.34.10 +norecurse NS +tcp console.cloud.google.c
om
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 65214
;; flags: qr aa; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;console.cloud.google.com.      IN      NS

;; ANSWER SECTION:
console.cloud.google.com. 300   IN      CNAME   www3.l.google.com.

;; AUTHORITY SECTION:
l.google.com.           60      IN      SOA     ns1.google.com. dns-admin.google.com. 571888995 900 900
1800 60

;; Query time: 11 msec
;; SERVER: 216.239.34.10#53(216.239.34.10) (TCP)
;; WHEN: Mon Oct 09 17:56:55 PDT 2023
;; MSG SIZE  rcvd: 124

awurtz@ada:~$ 
```

Addison Wurtz
CS 530
notebooks/Week2

Reverse DNS Lookups

Chat-GPT dig/egrep/awk for IPv4 addresses the espn.go.com points to:



A

Please construct a command line command that uses dig, egrep, and awk to alist all IPv4 addresses that espn.go.com points to

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

```bash
dig +short espn.go.com | egrep '([0-9]{1,3}\.){3}[0-9]{1,3}' | awk '{pr
```

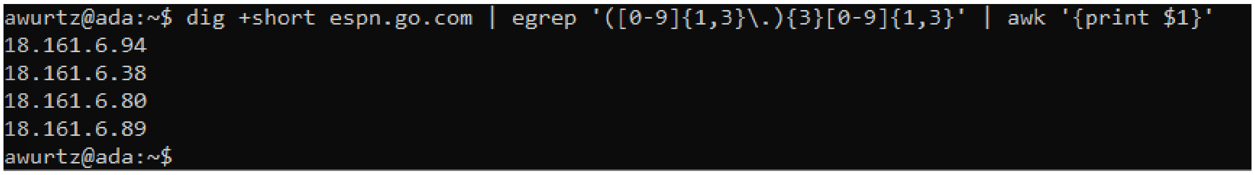
Here's what this command does:

1. `dig +short espn.go.com`: This uses `dig` to perform a DNS lookup for `espn.go.com` and returns the results in a short format (only the IP addresses).
2. `egrep '([0-9]{1,3}\.){3}[0-9]{1,3}'`: This uses `egrep` (extended grep) with a regular expression to filter out lines that contain valid IPv4 addresses. The regular expression used here matches the pattern of IPv4 addresses.
3. `awk '{print $1}'`: This uses `awk` to print the first column (which contains the IPv4 addresses) of the filtered output.

Keep in mind that the actual IP addresses may vary, and this command retrieves the current IP addresses associated with `espn.go.com` at the time of execution.

ODIN ID: awurtz

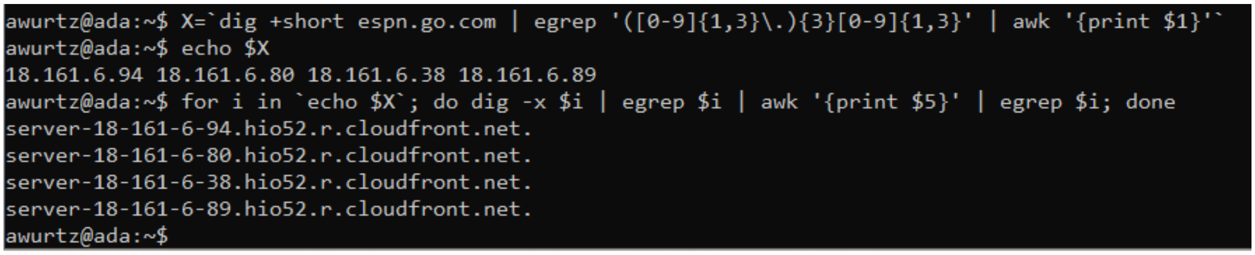
IPv4 addresses that espn.go.com points to:



```
awurtz@ada:~$ dig +short espn.go.com | egrep '([0-9]{1,3}\.){3}[0-9]{1,3}' | awk '{print $1}'
18.161.6.94
18.161.6.38
18.161.6.80
18.161.6.89
awurtz@ada:~$
```

Addison Wurtz
CS 530
notebooks/Week2

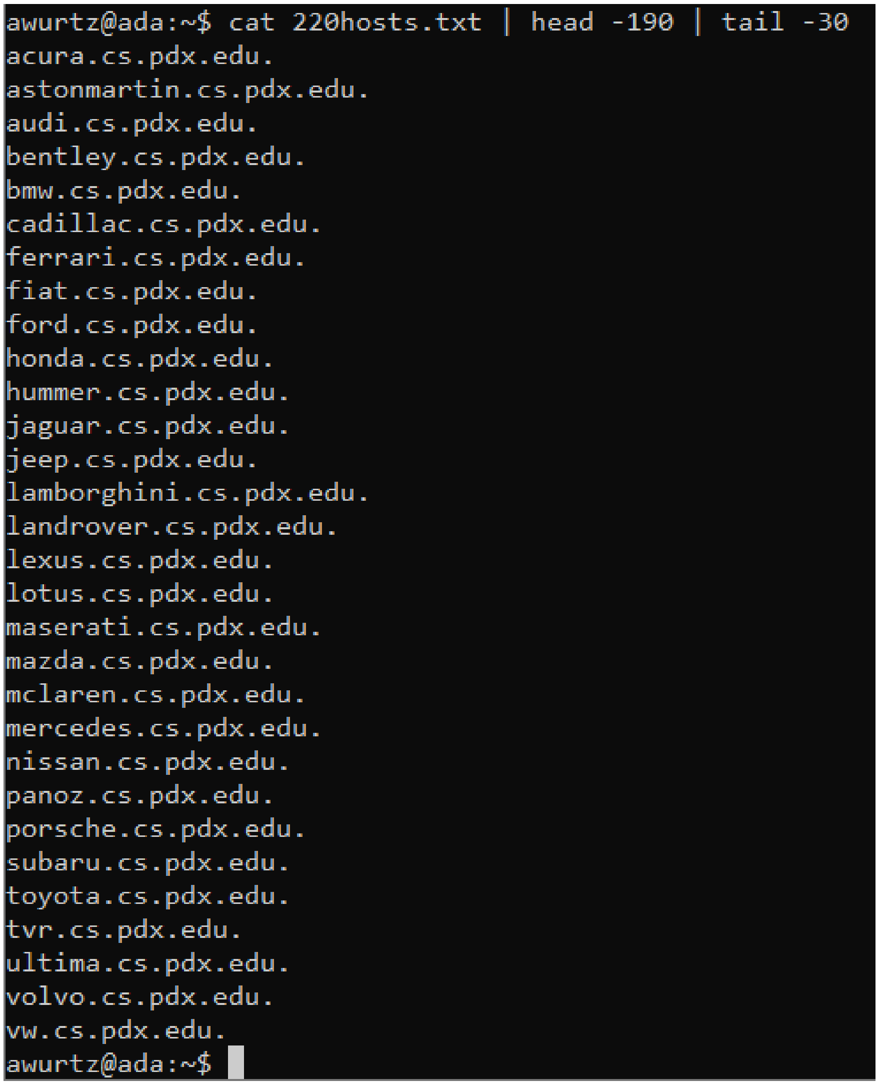
DNS names the espn.go.com points to:



```
awurtz@ada:~$ X=`dig +short espn.go.com | egrep '([0-9]{1,3}\.){3}[0-9]{1,3}' | awk '{print $1}'`
awurtz@ada:~$ echo $X
18.161.6.94 18.161.6.80 18.161.6.38 18.161.6.89
awurtz@ada:~$ for i in `echo $X`; do dig -x $i | egrep $i | awk '{print $5}' | egrep $i; done
server-18-161-6-94.hio52.r.cloudfront.net.
server-18-161-6-80.hio52.r.cloudfront.net.
server-18-161-6-38.hio52.r.cloudfront.net.
server-18-161-6-89.hio52.r.cloudfront.net.
awurtz@ada:~$
```

Addison Wurtz
CS 530
notebooks/Week2

Host Enumeration



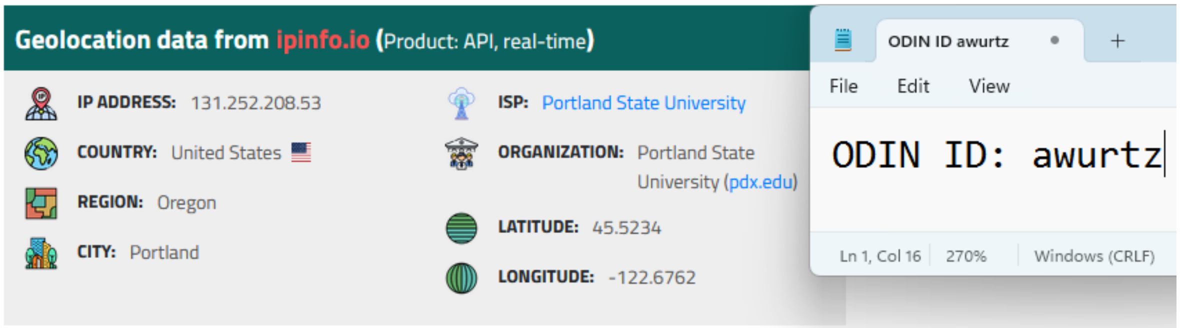
```
awurtz@ada:~$ cat 220hosts.txt | head -190 | tail -30
acura.cs.pdx.edu.
astonmartin.cs.pdx.edu.
audi.cs.pdx.edu.
bentley.cs.pdx.edu.
bmw.cs.pdx.edu.
cadillac.cs.pdx.edu.
ferrari.cs.pdx.edu.
fiat.cs.pdx.edu.
ford.cs.pdx.edu.
honda.cs.pdx.edu.
hummer.cs.pdx.edu.
jaguar.cs.pdx.edu.
jeep.cs.pdx.edu.
lamborghini.cs.pdx.edu.
landrover.cs.pdx.edu.
lexus.cs.pdx.edu.
lotus.cs.pdx.edu.
maserati.cs.pdx.edu.
mazda.cs.pdx.edu.
mclaren.cs.pdx.edu.
mercedes.cs.pdx.edu.
nissan.cs.pdx.edu.
panoz.cs.pdx.edu.
porsche.cs.pdx.edu.
subaru.cs.pdx.edu.
toyota.cs.pdx.edu.
tvr.cs.pdx.edu.
ultima.cs.pdx.edu.
volvo.cs.pdx.edu.
vw.cs.pdx.edu.
awurtz@ada:~$ 
```

Addison Wurtz
CS 530
notebooks/Week2

Geographic DNS #2

Geographic Locations of IP Addresses:

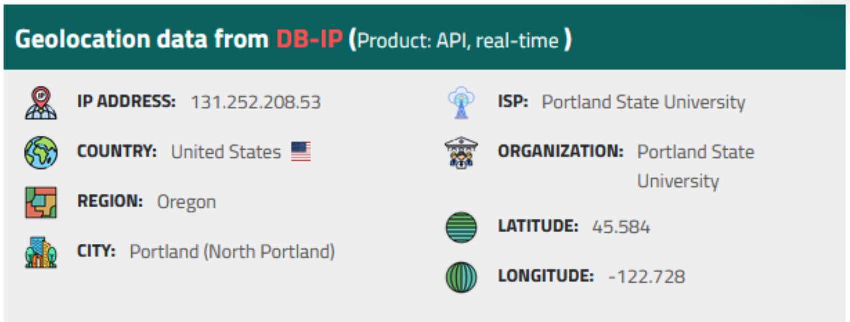
131.252.208.53: Portland, Oregon



Geolocation data from ipinfo.io (Product: API, real-time)

IP ADDRESS: 131.252.208.53
COUNTRY: United States
REGION: Oregon
CITY: Portland
ISP: Portland State University
ORGANIZATION: Portland State University (pdx.edu)
LATITUDE: 45.5234
LONGITUDE: -122.6762

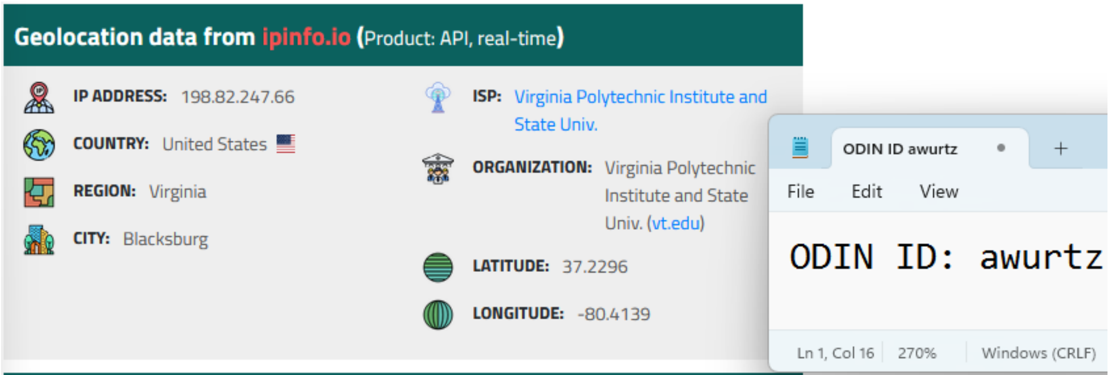
ODIN ID: awurtz



Geolocation data from DB-IP (Product: API, real-time)

IP ADDRESS: 131.252.208.53
COUNTRY: United States
REGION: Oregon
CITY: Portland (North Portland)
ISP: Portland State University
ORGANIZATION: Portland State University
LATITUDE: 45.584
LONGITUDE: -122.728

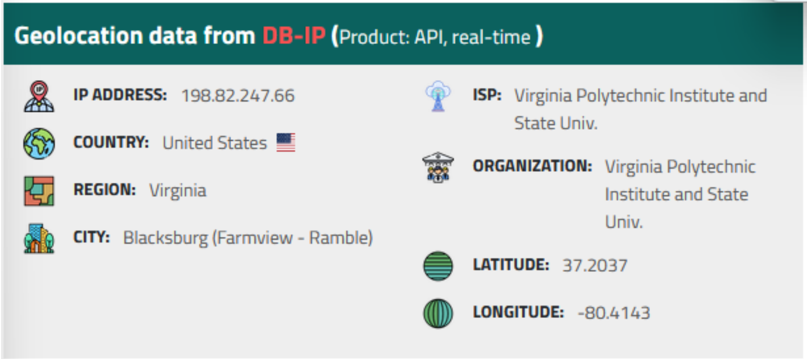
198.82.247.66: Blacksburg, Virginia



Geolocation data from ipinfo.io (Product: API, real-time)

IP ADDRESS: 198.82.247.66
COUNTRY: United States
REGION: Virginia
CITY: Blacksburg
ISP: Virginia Polytechnic Institute and State Univ.
ORGANIZATION: Virginia Polytechnic Institute and State Univ. (vt.edu)
LATITUDE: 37.2296
LONGITUDE: -80.4139

ODIN ID: awurtz



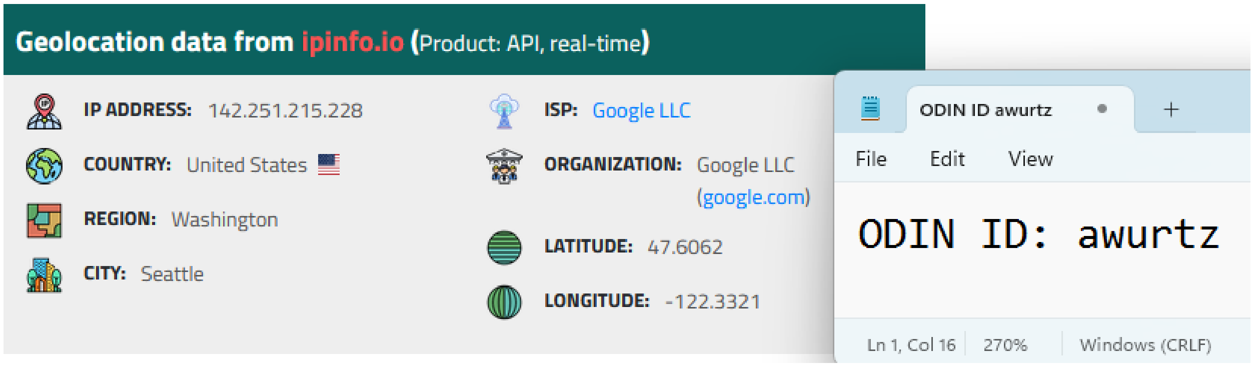
Geolocation data from DB-IP (Product: API, real-time)

IP ADDRESS: 198.82.247.66
COUNTRY: United States
REGION: Virginia
CITY: Blacksburg (Farmview - Ramble)
ISP: Virginia Polytechnic Institute and State Univ.
ORGANIZATION: Virginia Polytechnic Institute and State Univ.
LATITUDE: 37.2037
LONGITUDE: -80.4143

Addison Wurtz
CS 530
notebooks/Week2

Using dig to resolve www.google.com from different DNS servers:

Portland State (131.252.208.53):
IP for www.google.com: 142.251.215.228

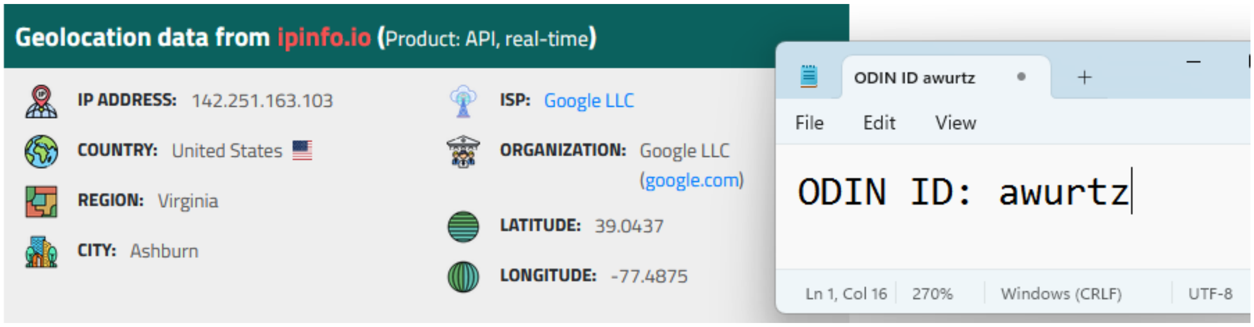


Geolocation data from ipinfo.io (Product: API, real-time)

IP ADDRESS: 142.251.215.228
COUNTRY: United States
REGION: Washington
CITY: Seattle
ISP: Google LLC
ORGANIZATION: Google LLC (google.com)
LATITUDE: 47.6062
LONGITUDE: -122.3321

ODIN ID: awurtz

Virginia Tech (198.82.247.66):
IP for www.google.com: 142.251.163.103



Geolocation data from ipinfo.io (Product: API, real-time)

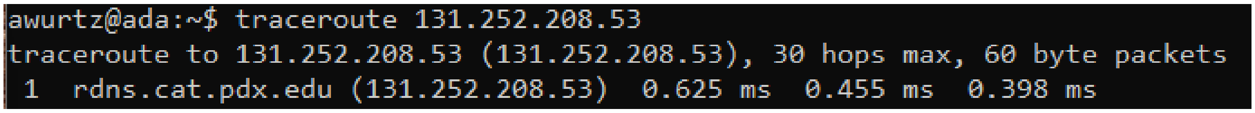
IP ADDRESS: 142.251.163.103
COUNTRY: United States
REGION: Virginia
CITY: Ashburn
ISP: Google LLC
ORGANIZATION: Google LLC (google.com)
LATITUDE: 39.0437
LONGITUDE: -77.4875

ODIN ID: awurtz

Addison Wurtz
CS 530
notebooks/Week2

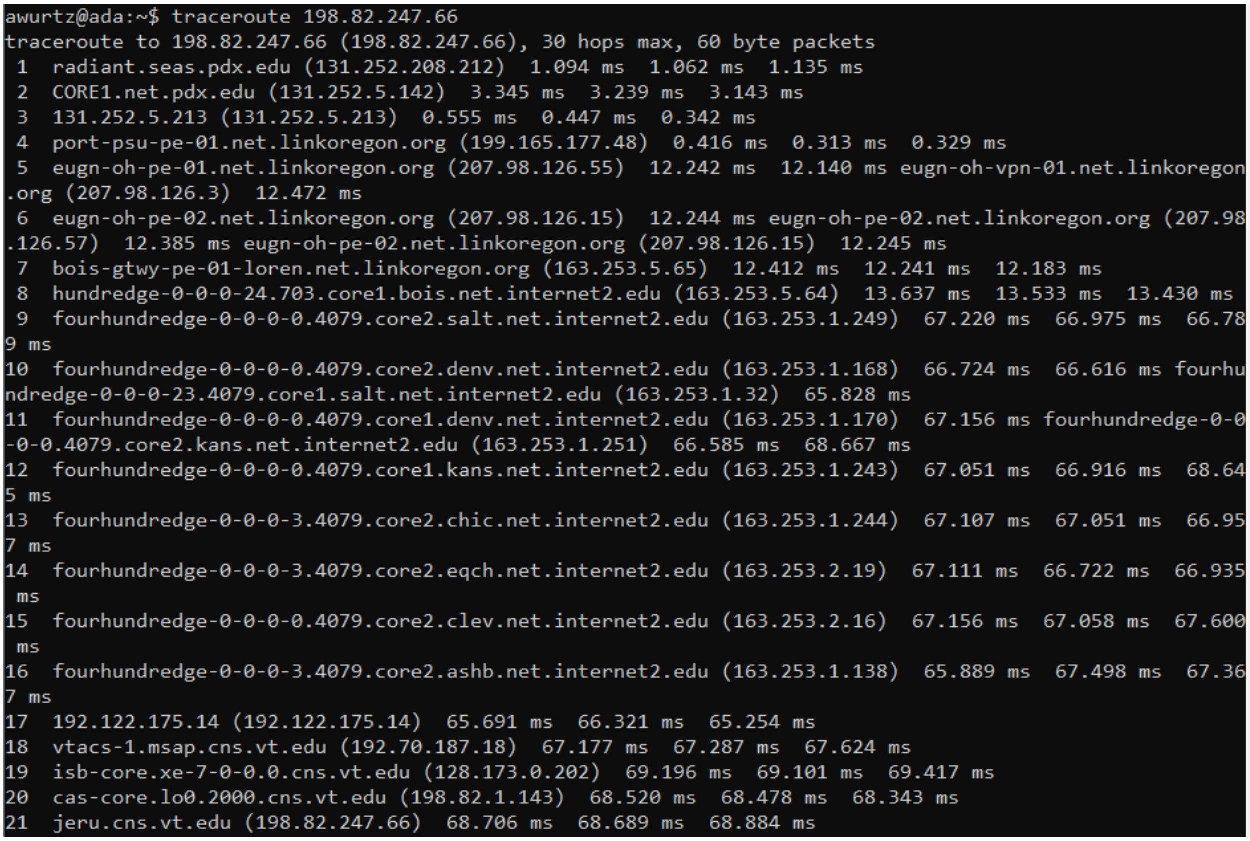
Traceroute:

131.252.208.53



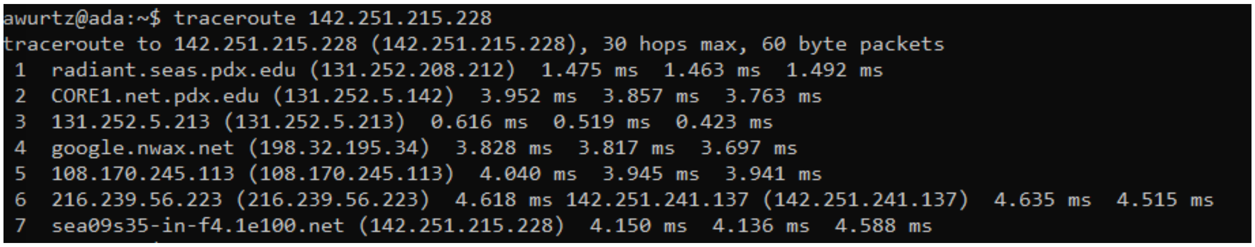
```
awurtz@ada:~$ traceroute 131.252.208.53
traceroute to 131.252.208.53 (131.252.208.53), 30 hops max, 60 byte packets
 1  rdns.cat.pdx.edu (131.252.208.53)  0.625 ms  0.455 ms  0.398 ms
```

198.82.247.66



```
awurtz@ada:~$ traceroute 198.82.247.66
traceroute to 198.82.247.66 (198.82.247.66), 30 hops max, 60 byte packets
 1  radiant.seas.pdx.edu (131.252.208.212)  1.094 ms  1.062 ms  1.135 ms
 2  CORE1.net.pdx.edu (131.252.5.142)  3.345 ms  3.239 ms  3.143 ms
 3  131.252.5.213 (131.252.5.213)  0.555 ms  0.447 ms  0.342 ms
 4  port-psu-pe-01.net.linkoregon.org (199.165.177.48)  0.416 ms  0.313 ms  0.329 ms
 5  eugn-oh-pe-01.net.linkoregon.org (207.98.126.55)  12.242 ms  12.140 ms eugn-oh-vpn-01.net.linkoregon
.org (207.98.126.3)  12.472 ms
 6  eugn-oh-pe-02.net.linkoregon.org (207.98.126.15)  12.244 ms eugn-oh-pe-02.net.linkoregon.org (207.98
.126.57)  12.385 ms eugn-oh-pe-02.net.linkoregon.org (207.98.126.15)  12.245 ms
 7  bois-gtwy-pe-01-loren.net.linkoregon.org (163.253.5.65)  12.412 ms  12.241 ms  12.183 ms
 8  hundredge-0-0-0-24.703.core1.bois.net.internet2.edu (163.253.5.64)  13.637 ms  13.533 ms  13.430 ms
 9  fourhundredge-0-0-0-0.4079.core2.salt.net.internet2.edu (163.253.1.249)  67.220 ms  66.975 ms  66.78
9 ms
10  fourhundredge-0-0-0-0.4079.core2.denv.net.internet2.edu (163.253.1.168)  66.724 ms  66.616 ms fourhu
ndredge-0-0-0-23.4079.core1.salt.net.internet2.edu (163.253.1.32)  65.828 ms
11  fourhundredge-0-0-0-0.4079.core1.denv.net.internet2.edu (163.253.1.170)  67.156 ms fourhundredge-0-0
-0-0.4079.core2.kans.net.internet2.edu (163.253.1.251)  66.585 ms  68.667 ms
12  fourhundredge-0-0-0-0.4079.core1.kans.net.internet2.edu (163.253.1.243)  67.051 ms  66.916 ms  68.64
5 ms
13  fourhundredge-0-0-0-3.4079.core2.chic.net.internet2.edu (163.253.1.244)  67.107 ms  67.051 ms  66.95
7 ms
14  fourhundredge-0-0-0-3.4079.core2.eqch.net.internet2.edu (163.253.2.19)  67.111 ms  66.722 ms  66.935
 ms
15  fourhundredge-0-0-0-0.4079.core2.clev.net.internet2.edu (163.253.2.16)  67.156 ms  67.058 ms  67.600
 ms
16  fourhundredge-0-0-0-3.4079.core2.ashb.net.internet2.edu (163.253.1.138)  65.889 ms  67.498 ms  67.36
7 ms
17  192.122.175.14 (192.122.175.14)  65.691 ms  66.321 ms  65.254 ms
18  vtacs-1.msap.cns.vt.edu (192.70.187.18)  67.177 ms  67.287 ms  67.624 ms
19  isb-core.xe-7-0-0.0.cns.vt.edu (128.173.0.202)  69.196 ms  69.101 ms  69.417 ms
20  cas-core.lo0.2000.cns.vt.edu (198.82.1.143)  68.520 ms  68.478 ms  68.343 ms
21  jeru.cns.vt.edu (198.82.247.66)  68.706 ms  68.689 ms  68.884 ms
```

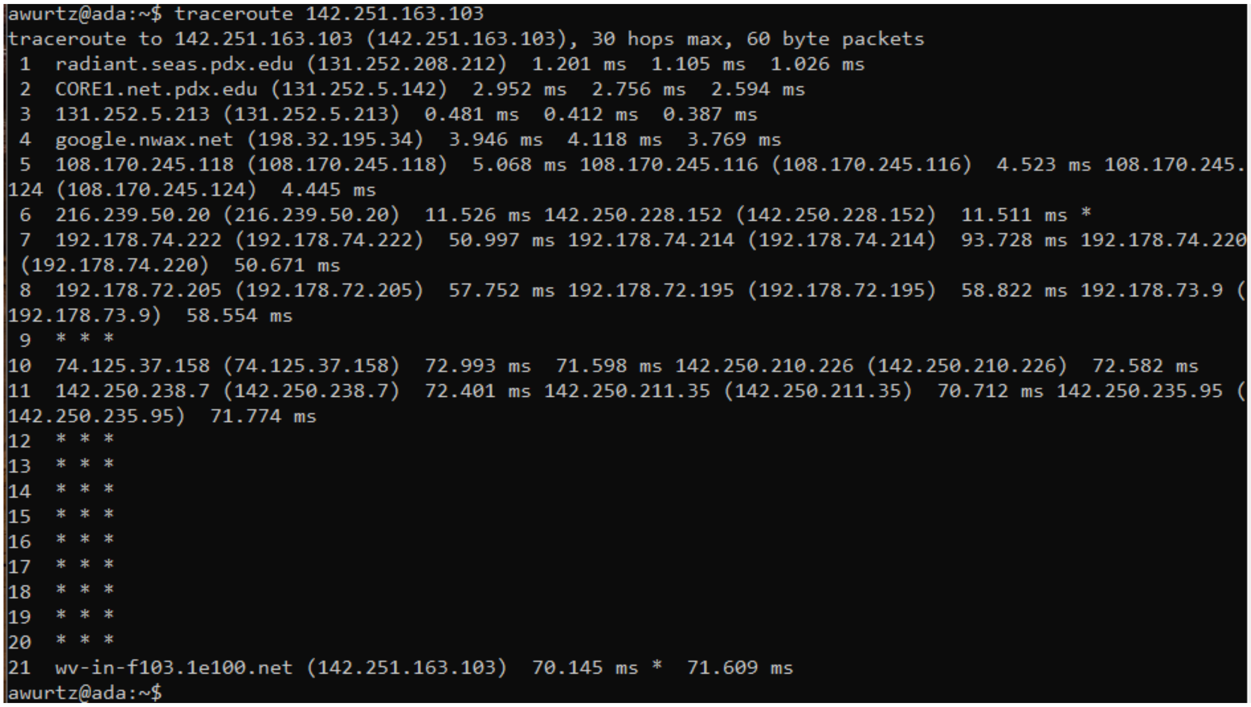
142.251.215.228



```
awurtz@ada:~$ traceroute 142.251.215.228
traceroute to 142.251.215.228 (142.251.215.228), 30 hops max, 60 byte packets
 1  radiant.seas.pdx.edu (131.252.208.212)  1.475 ms  1.463 ms  1.492 ms
 2  CORE1.net.pdx.edu (131.252.5.142)  3.952 ms  3.857 ms  3.763 ms
 3  131.252.5.213 (131.252.5.213)  0.616 ms  0.519 ms  0.423 ms
 4  google.nwax.net (198.32.195.34)  3.828 ms  3.817 ms  3.697 ms
 5  108.170.245.113 (108.170.245.113)  4.040 ms  3.945 ms  3.941 ms
 6  216.239.56.223 (216.239.56.223)  4.618 ms 142.251.241.137 (142.251.241.137)  4.635 ms  4.515 ms
 7  sea09s35-in-f4.1e100.net (142.251.215.228)  4.150 ms  4.136 ms  4.588 ms
```

Addison Wurtz
CS 530
notebooks/Week2

142.251.163.103



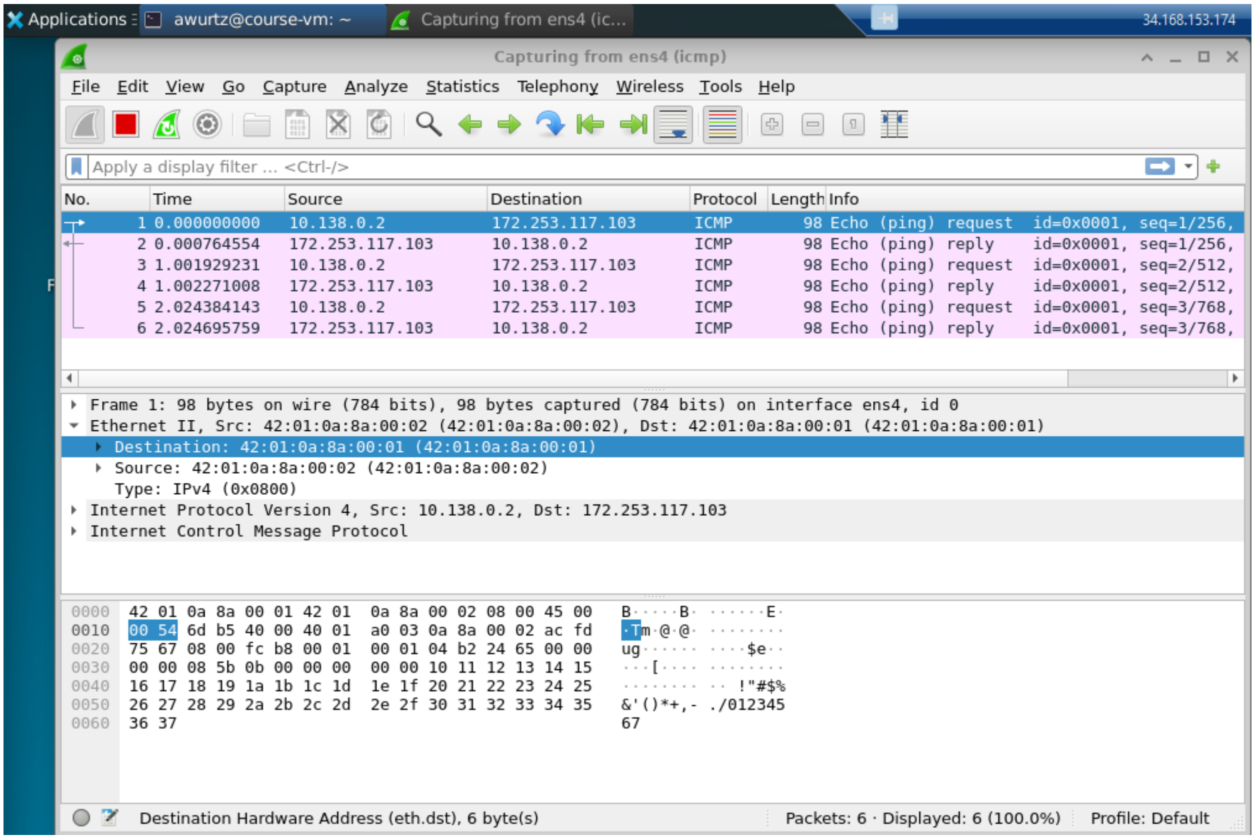
```
awurtz@ada:~$ traceroute 142.251.163.103
traceroute to 142.251.163.103 (142.251.163.103), 30 hops max, 60 byte packets
 1  radiant.seas.pdx.edu (131.252.208.212)  1.201 ms  1.105 ms  1.026 ms
 2  CORE1.net.pdx.edu (131.252.5.142)  2.952 ms  2.756 ms  2.594 ms
 3  131.252.5.213 (131.252.5.213)  0.481 ms  0.412 ms  0.387 ms
 4  google.nwax.net (198.32.195.34)  3.946 ms  4.118 ms  3.769 ms
 5  108.170.245.118 (108.170.245.118)  5.068 ms 108.170.245.116 (108.170.245.116)  4.523 ms 108.170.245.
124 (108.170.245.124)  4.445 ms
 6  216.239.50.20 (216.239.50.20)  11.526 ms 142.250.228.152 (142.250.228.152)  11.511 ms *
 7  192.178.74.222 (192.178.74.222)  50.997 ms 192.178.74.214 (192.178.74.214)  93.728 ms 192.178.74.220
 (192.178.74.220)  50.671 ms
 8  192.178.72.205 (192.178.72.205)  57.752 ms 192.178.72.195 (192.178.72.195)  58.822 ms 192.178.73.9 (
192.178.73.9)  58.554 ms
 9  * * *
10  74.125.37.158 (74.125.37.158)  72.993 ms  71.598 ms 142.250.210.226 (142.250.210.226)  72.582 ms
11  142.250.238.7 (142.250.238.7)  72.401 ms 142.250.211.35 (142.250.211.35)  70.712 ms 142.250.235.95 (
142.250.235.95)  71.774 ms
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  wv-in-f103.1e100.net (142.251.163.103)  70.145 ms *  71.609 ms
awurtz@ada:~$
```

Wireshark Lab #3

- The IP address of the VM: 34.168.153.174
- The name of the local virtual ethernet interface: _gateway
- The IP address of the default router: 10.138.0.1

Addison Wurtz
CS 530
notebooks/Week2

Wireshark Packet Dump


Applications
awurtz@course-vm: ~
Capturing from ens4 (ic...
34.168.153.174
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	172.253.117.103	ICMP	98	Echo (ping) request id=0x0001, seq=1/256,
2	0.000764554	172.253.117.103	10.138.0.2	ICMP	98	Echo (ping) reply id=0x0001, seq=1/256,
3	1.001929231	10.138.0.2	172.253.117.103	ICMP	98	Echo (ping) request id=0x0001, seq=2/512,
4	1.002271008	172.253.117.103	10.138.0.2	ICMP	98	Echo (ping) reply id=0x0001, seq=2/512,
5	2.024384143	10.138.0.2	172.253.117.103	ICMP	98	Echo (ping) request id=0x0001, seq=3/768,
6	2.024695759	172.253.117.103	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: 172.253.117.103
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 6d b5 40 00 40 01  a0 03 0a 8a 00 02 ac fd   ·Tm·@·@· ········
0020  75 67 08 00 fc b8 00 01  00 01 04 b2 24 65 00 00   ug······ ····$e··
0030  00 00 08 5b 0b 00 00 00  00 00 10 11 12 13 14 15   ···[···· ········
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ········ ·· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                              67
```

Destination Hardware Address (eth.dst), 6 byte(s) Packets: 6 · Displayed: 6 (100.0%) Profile: Default

---

First Packet 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 destination MAC address corresponds to an interface on the default router.

```
awurtz@course-vm:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
_gateway                 ether   42:01:0a:8a:00:01   C                     ens4
```

Second Packet 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?

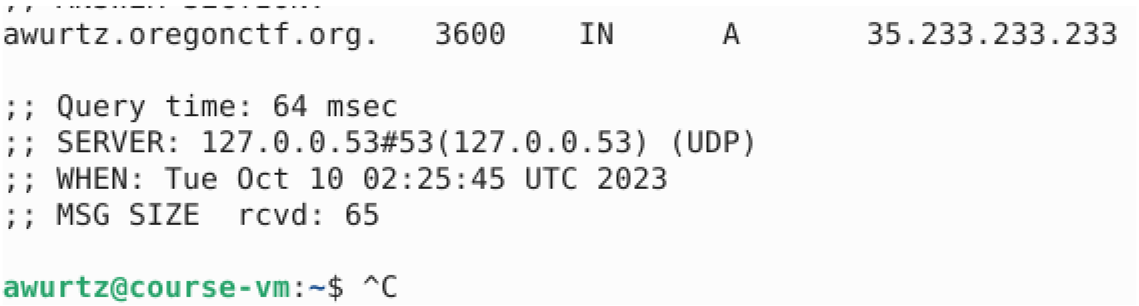
The destination corresponds to an interface on the VM (that’s where the ping reply is going).

Addison Wurtz
CS 530
notebooks/Week2

Network Recap Lab #4

IP of awurtz.oregonctf.org

35.233.233.233

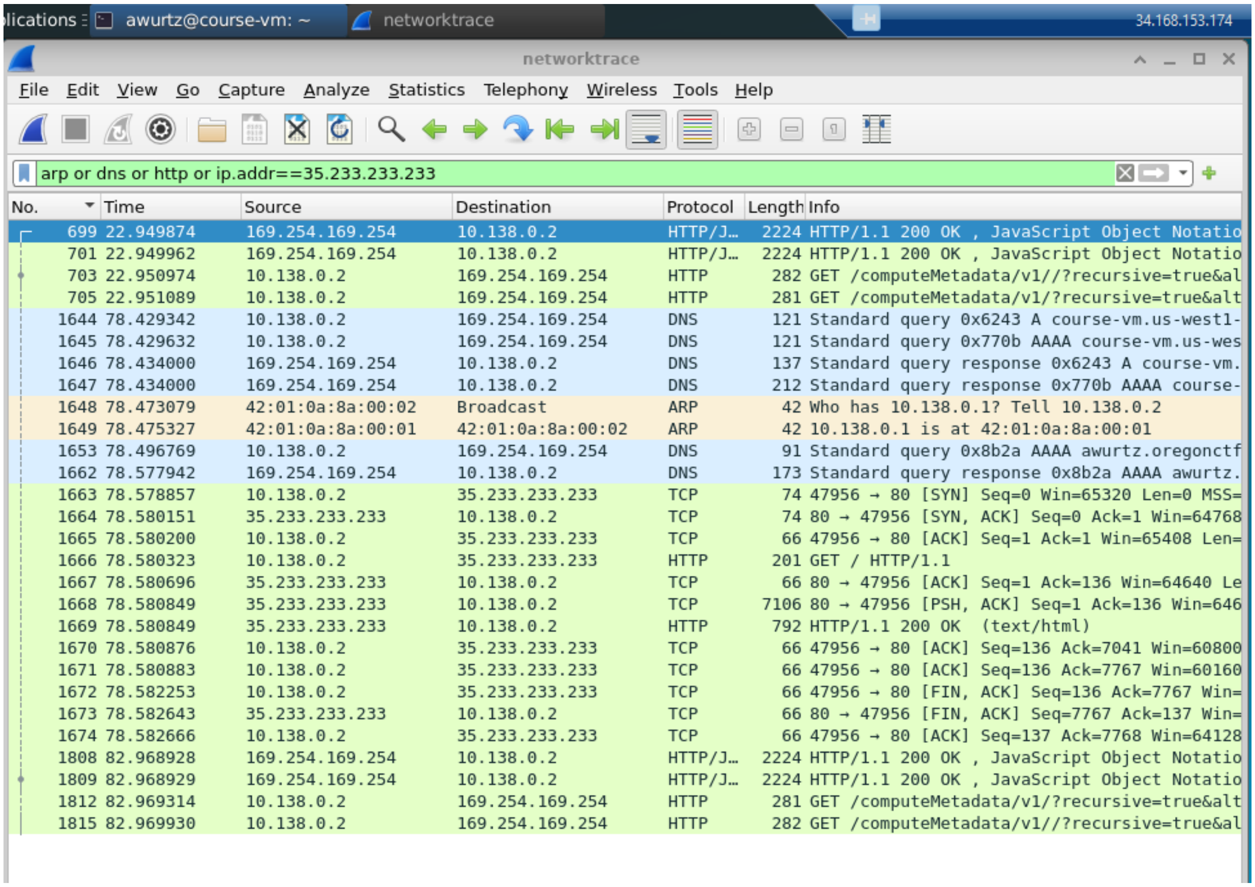


```
;; ANSWER SECTION:
awurtz.oregonctf.org.   3600    IN      A       35.233.233.233

;; Query time: 64 msec
;; SERVER: 127.0.0.53#53(127.0.0.53) (UDP)
;; WHEN: Tue Oct 10 02:25:45 UTC 2023
;; MSG SIZE  rcvd: 65

awurtz@course-vm:~$ ^C
```

Collect Trace

Addison Wurtz
CS 530
notebooks/Week2

Analyze Trace


lications
awurtz@course-vm: ~
networktrace
34.168.153.174
networktrace
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
699	22.949874	169.254.169.254	10.138.0.2	HTTP/J…	2224	HTTP/1.1 200 OK , JavaScript Object Notatio
701	22.949962	169.254.169.254	10.138.0.2	HTTP/J…	2224	HTTP/1.1 200 OK , JavaScript Object Notatio
703	22.950974	10.138.0.2	169.254.169.254	HTTP	282	GET /computeMetadata/v1//?recursive=true&al
705	22.951089	10.138.0.2	169.254.169.254	HTTP	281	GET /computeMetadata/v1/?recursive=true&alt
1644	78.429342	10.138.0.2	169.254.169.254	DNS	121	Standard query 0x6243 A course-vm.us-west1-
1645	78.429632	10.138.0.2	169.254.169.254	DNS	121	Standard query 0x770b AAAA course-vm.us-wes
1646	78.434000	169.254.169.254	10.138.0.2	DNS	137	Standard query response 0x6243 A course-vm.
1647	78.434000	169.254.169.254	10.138.0.2	DNS	212	Standard query response 0x770b AAAA course-
1648	78.473079	42:01:0a:8a:00:02	Broadcast	ARP	42	Who has 10.138.0.1? Tell 10.138.0.2
1649	78.475327	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
1653	78.496769	10.138.0.2	169.254.169.254	DNS	91	Standard query 0x8b2a AAAA awurtz.oregonctf
1662	78.577942	169.254.169.254	10.138.0.2	DNS	173	Standard query response 0x8b2a AAAA awurtz.
1663	78.578857	10.138.0.2	35.233.233.233	TCP	74	47956 → 80 [SYN] Seq=0 Win=65320 Len=0 MSS=
1664	78.580151	35.233.233.233	10.138.0.2	TCP	74	80 → 47956 [SYN, ACK] Seq=0 Ack=1 Win=64768
1665	78.580200	10.138.0.2	35.233.233.233	TCP	66	47956 → 80 [ACK] Seq=1 Ack=1 Win=65408 Len=
1666	78.580323	10.138.0.2	35.233.233.233	HTTP	201	GET / HTTP/1.1
1667	78.580696	35.233.233.233	10.138.0.2	TCP	66	80 → 47956 [ACK] Seq=1 Ack=136 Win=64640 Le
1668	78.580849	35.233.233.233	10.138.0.2	TCP	7106	80 → 47956 [PSH, ACK] Seq=1 Ack=136 Win=646
1669	78.580849	35.233.233.233	10.138.0.2	HTTP	792	HTTP/1.1 200 OK (text/html)
1670	78.580876	10.138.0.2	35.233.233.233	TCP	66	47956 → 80 [ACK] Seq=136 Ack=7041 Win=60800
1671	78.580883	10.138.0.2	35.233.233.233	TCP	66	47956 → 80 [ACK] Seq=136 Ack=7767 Win=60160
1672	78.582253	10.138.0.2	35.233.233.233	TCP	66	47956 → 80 [FIN, ACK] Seq=136 Ack=7767 Win=
1673	78.582643	35.233.233.233	10.138.0.2	TCP	66	80 → 47956 [FIN, ACK] Seq=7767 Ack=137 Win=
1674	78.582666	10.138.0.2	35.233.233.233	TCP	66	47956 → 80 [ACK] Seq=137 Ack=7768 Win=64128
1808	82.968928	169.254.169.254	10.138.0.2	HTTP/J…	2224	HTTP/1.1 200 OK , JavaScript Object Notatio
1809	82.968929	169.254.169.254	10.138.0.2	HTTP/J…	2224	HTTP/1.1 200 OK , JavaScript Object Notatio
1812	82.969314	10.138.0.2	169.254.169.254	HTTP	281	GET /computeMetadata/v1/?recursive=true&alt
1815	82.969930	10.138.0.2	169.254.169.254	HTTP	282	GET /computeMetadata/v1//?recursive=true&al

---

Arp

- **What packet numbers in the trace are the result of the VM attempting to get the hardware address of the default router?**

  Packets 1648 and 1649.

- **What is this hardware address?**

  The hardware address is 42:01:0a:8a:00:01

DNS

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

Packets 1644, 1645, and 1653.

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

  IP address of the local DNS is 10.138.0.2

TCP

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

  Packets 1663, 1664, and 1665.

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

  78.580200 - 78.578857 = 0.001343 seconds = 1.343 milliseconds

HTTP

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

  Packets 1666 and 1669 (also 1808 and 1809).

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

  78.580849 - 78.580323 = 0.00052599999 seconds = 0.526 milliseconds