https://github.com/martintc/ser321-spring2021-A-tcmart14

Todd Martin
tcmart14

Task 1.1

```
1. mkdir cli_assignment
2. cd cli_assignment
3. touch. stufftxt
4. cat << EOF > stuff.txt
5. cat -n stuff.txt
6. cat << EOF >> stuff.txt
7. mkdir draft
8. mv stuff.txt draft
9. cd draft && touch .secret.txt
10. mkdir final && cp stuff.txt .secret.txt final/ && mv final ../
11. mv ../draft ../draft.remove
12. mv ../draft.remove ../final
13. ls -Ral
14. gzip -l NASA_access_log_Aug95.gz
15. gzip -d NASA_access_log_Aug95.gz
16. mv NASA_access_log_Aug95 logs.txt
17. mv logs.txt cli_assignment/
18. sed -n 1,100p cli_assignment/logs.txt
19. sed -n 1,100p cli_assignment/logs.txt > logs_top_100.txt
20. tail cli_assignment/logs.txt -n 100
21. tail vli_assignment/logs.txt -n 100 > logs_bottom_100.txt
22. touch cli_assignment/logs_snapshot.txt && cat cli_assignment/logs_top100.txt
cli_assignment/logs_bottom_100.txt > cli_assignment/logs_snapshot.txt
23. echo 'tcmart14: This is a great assignment 1-13-21' >> cli_aassignments/logs_snapshot.txt
24. less cli_assignments/logs.txt
25. cat marks.csv | cut -f1 -d '%'
26. cat marks.csv | cut -f4 -d '%' | sort
27. cat marks.csv | cut -f3 -d '%' | awk  'BEGIN { total = 0.0; count = 0 } { total += $1; count += 1;
} END {|avg = total / count; print avg} '
28.  cat marks.csv | cut -f3 -d '%' | awk  'BEGIN { total = 0.0; count = 0 } { total += $1; count += 1;
} END {
                                                                                  |avg = total / count;
print avg} ' > done.txt
29. mv done.txt cli_assignment/final/
30. mv cli_assignment/final/done.txt /cli_assignment/final/average.txt
```

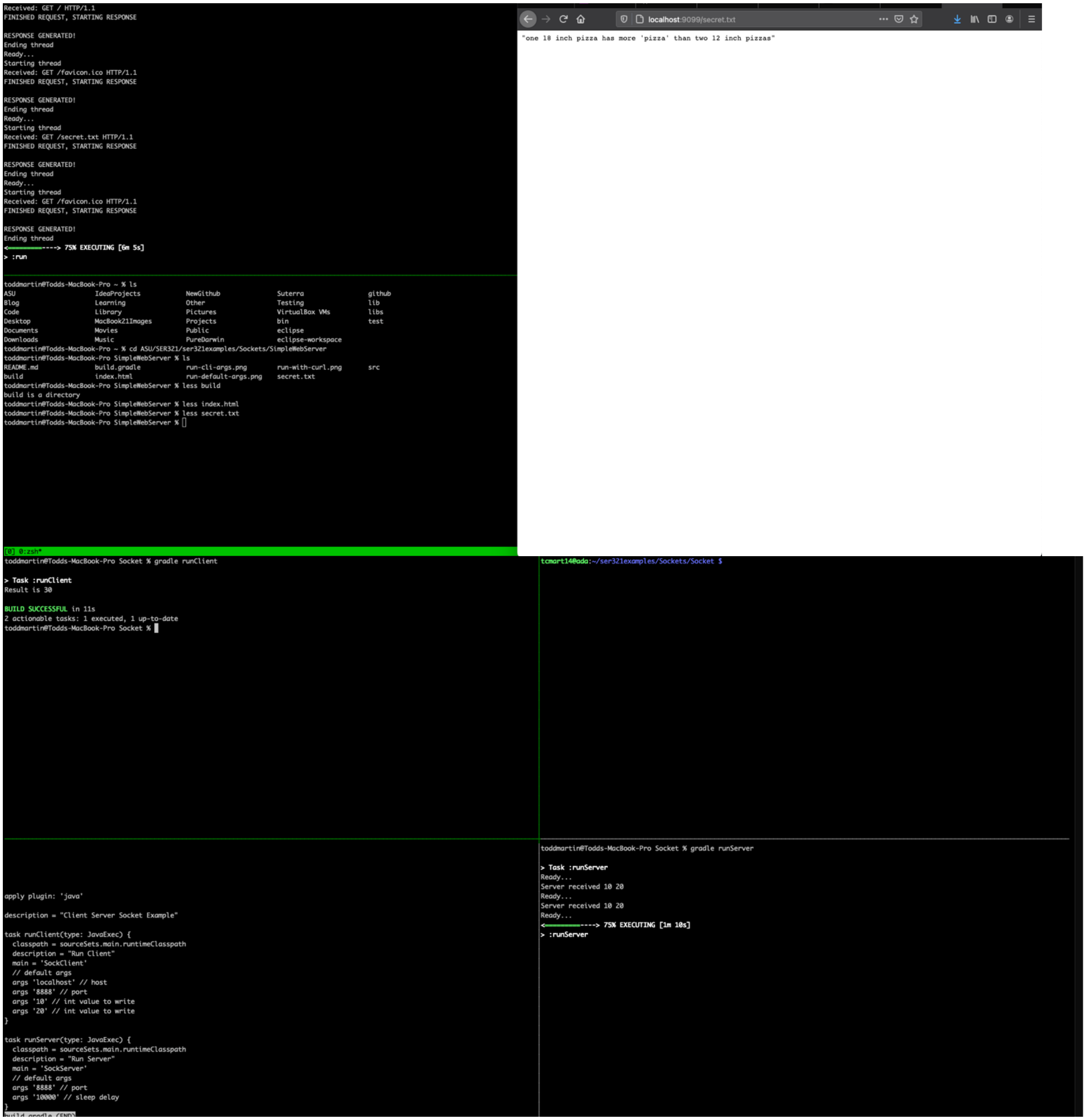
Task 1.2.1

Github link: https://github.com/martintc/ser321-spring2021-A-tcmart14

https://github.com/martintc/ser321-spring2021-A-tcmart14

Todd Martin
tcmart14

Task 1.2.2



```
Received: GET / HTTP/1.1
FINISHED REQUEST, STARTING RESPONSE

RESPONSE GENERATED!
Ending thread
Ready...
Starting thread
Received: GET /favicon.ico HTTP/1.1
FINISHED REQUEST, STARTING RESPONSE

RESPONSE GENERATED!
Ending thread
Ready...
Starting thread
Received: GET /secret.txt HTTP/1.1
FINISHED REQUEST, STARTING RESPONSE

RESPONSE GENERATED!
Ending thread
Ready...
Starting thread
Received: GET /favicon.ico HTTP/1.1
FINISHED REQUEST, STARTING RESPONSE

RESPONSE GENERATED!
Ending thread
<=========----> 75% EXECUTING [6m 5s]
> :run
```

```
toddmartin@Todds-MacBook-Pro ~ % ls
ASU                 IdeaProjects        NewGithub           Suterra             github
Blog                Learning            Other               Testing             lib
Code                Library             Pictures            VirtualBox VMs      libs
Desktop             MacBook21Images     Projects            bin                 test
Documents           Movies              Public              eclipse
Downloads           Music               PureDarwin          eclipse-workspace
toddmartin@Todds-MacBook-Pro ~ % cd ASU/SER321/ser321examples/Sockets/SimpleWebServer
toddmartin@Todds-MacBook-Pro SimpleWebServer % ls
README.md           build.gradle        run-cli-args.png    run-with-curl.png   src
build               index.html          run-default-args.png secret.txt
toddmartin@Todds-MacBook-Pro SimpleWebServer % less build
build is a directory
toddmartin@Todds-MacBook-Pro SimpleWebServer % less index.html
toddmartin@Todds-MacBook-Pro SimpleWebServer % less secret.txt
toddmartin@Todds-MacBook-Pro SimpleWebServer % 
```

localhost:9099/secret.txt

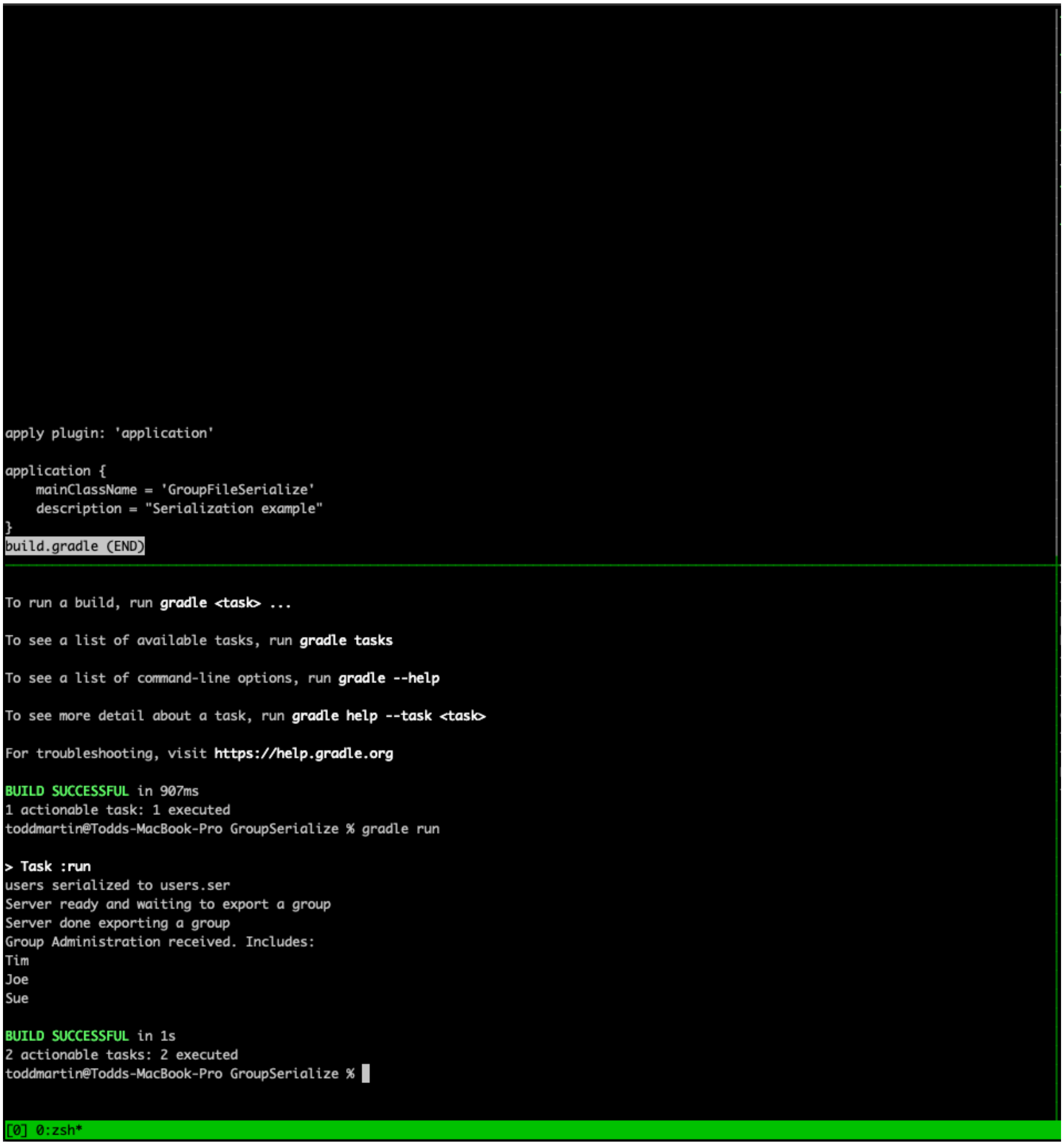
```
"one 18 inch pizza has more 'pizza' than two 12 inch pizzas"
```

```
[0] 0:zsh*
toddmartin@Todds-MacBook-Pro Socket % gradle runClient

> Task :runClient
Result is 30

BUILD SUCCESSFUL in 11s
2 actionable tasks: 1 executed, 1 up-to-date
toddmartin@Todds-MacBook-Pro Socket % 
```

```
tcmart14@ada:~/ser321examples/Sockets/Socket $
```

```
apply plugin: 'java'

description = "Client Server Socket Example"

task runClient(type: JavaExec) {
  classpath = sourceSets.main.runtimeClasspath
  description = "Run Client"
  main = 'SockClient'
  // default args
  args 'localhost' // host
  args '8888' // port
  args '10' // int value to write
  args '20' // int value to write
}

task runServer(type: JavaExec) {
  classpath = sourceSets.main.runtimeClasspath
  description = "Run Server"
  main = 'SockServer'
  // default args
  args '8888' // port
  args '10000' // sleep delay
}
build.gradle (END)
```

```
toddmartin@Todds-MacBook-Pro Socket % gradle runServer

> Task :runServer
Ready...
Server received 10 20
Ready...
Server received 10 20
Ready...
<=========----> 75% EXECUTING [1m 10s]
> :runServer
```



```
apply plugin: 'application'

application {
    mainClassName = 'GroupFileSerialize'
    description = "Serialization example"
}
build.gradle (END)
```

```
To run a build, run gradle <task> ...

To see a list of available tasks, run gradle tasks

To see a list of command-line options, run gradle --help

To see more detail about a task, run gradle help --task <task>

For troubleshooting, visit https://help.gradle.org

BUILD SUCCESSFUL in 907ms
1 actionable task: 1 executed
toddmartin@Todds-MacBook-Pro GroupSerialize % gradle run

> Task :run
users serialized to users.ser
Server ready and waiting to export a group
Server done exporting a group
Group Administration received. Includes:
Tim
Joe
Sue

BUILD SUCCESSFUL in 1s
2 actionable tasks: 2 executed
toddmartin@Todds-MacBook-Pro GroupSerialize %
```

https://github.com/martintc/ser321-spring2021-A-tcmart14

Todd Martin
tcmart14

Task 1.2.3

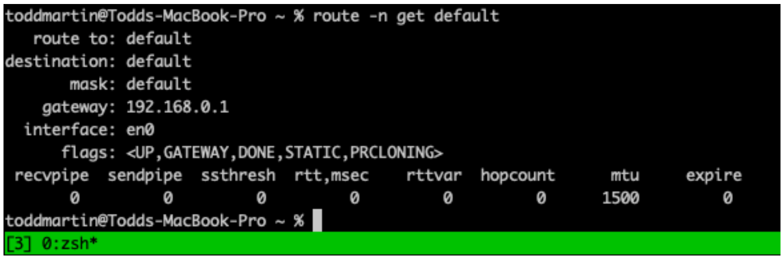
In repo

Task 1.2.4

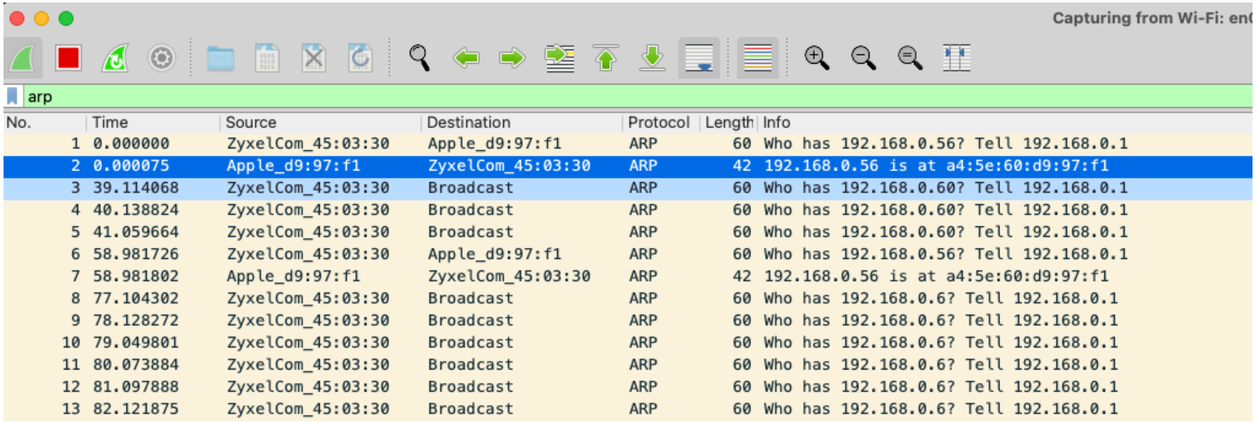
My second system is a raspberry pi 4b on my local area network running the current Raspberry Pi OS Stable based on Debian Buster.

Video is in repo and here is the link to my video demonstrating it on youtube:
https://www.youtube.com/watch?v=mjCsQl1tyg8&feature=youtu.be

Task 2.1

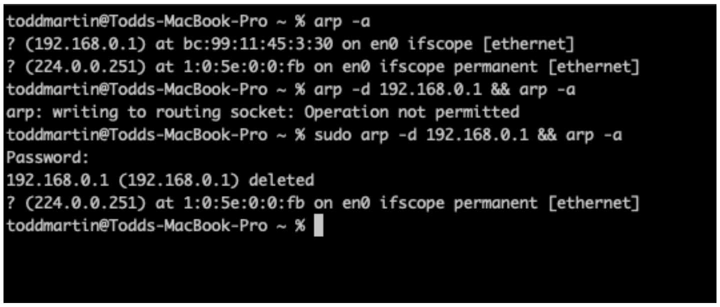


```
toddmartin@Todds-MacBook-Pro ~ % route -n get default
   route to: default
destination: default
       mask: default
    gateway: 192.168.0.1
  interface: en0
      flags: <UP,GATEWAY,DONE,STATIC,PRCLONING>
 recvpipe  sendpipe  ssthresh  rtt,msec    rttvar  hopcount      mtu     expire
       0         0         0         0         0         0      1500         0
toddmartin@Todds-MacBook-Pro ~ % 
[3] 0:zsh*
```



arp

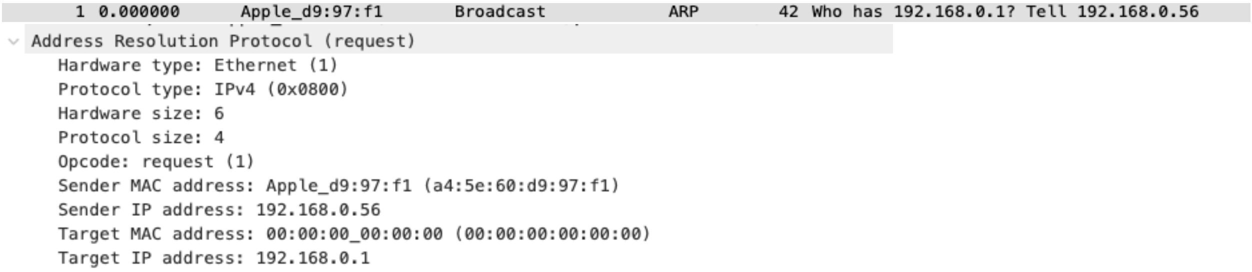
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	ZyxelCom_45:03:30	Apple_d9:97:f1	ARP	60	Who has 192.168.0.56? Tell 192.168.0.1
2	0.000075	Apple_d9:97:f1	ZyxelCom_45:03:30	ARP	42	192.168.0.56 is at a4:5e:60:d9:97:f1
3	39.114068	ZyxelCom_45:03:30	Broadcast	ARP	60	Who has 192.168.0.60? Tell 192.168.0.1
4	40.138824	ZyxelCom_45:03:30	Broadcast	ARP	60	Who has 192.168.0.60? Tell 192.168.0.1
5	41.059664	ZyxelCom_45:03:30	Broadcast	ARP	60	Who has 192.168.0.60? Tell 192.168.0.1
6	58.981726	ZyxelCom_45:03:30	Apple_d9:97:f1	ARP	60	Who has 192.168.0.56? Tell 192.168.0.1
7	58.981802	Apple_d9:97:f1	ZyxelCom_45:03:30	ARP	42	192.168.0.56 is at a4:5e:60:d9:97:f1
8	77.104302	ZyxelCom_45:03:30	Broadcast	ARP	60	Who has 192.168.0.6? Tell 192.168.0.1
9	78.128272	ZyxelCom_45:03:30	Broadcast	ARP	60	Who has 192.168.0.6? Tell 192.168.0.1
10	79.049801	ZyxelCom_45:03:30	Broadcast	ARP	60	Who has 192.168.0.6? Tell 192.168.0.1
11	80.073884	ZyxelCom_45:03:30	Broadcast	ARP	60	Who has 192.168.0.6? Tell 192.168.0.1
12	81.097888	ZyxelCom_45:03:30	Broadcast	ARP	60	Who has 192.168.0.6? Tell 192.168.0.1
13	82.121875	ZyxelCom_45:03:30	Broadcast	ARP	60	Who has 192.168.0.6? Tell 192.168.0.1



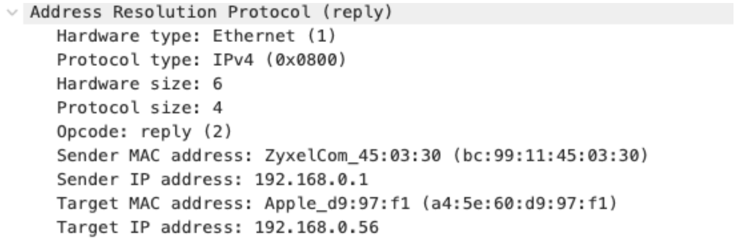
```
toddmartin@Todds-MacBook-Pro ~ % arp -a
? (192.168.0.1) at bc:99:11:45:3:30 on en0 ifscope [ethernet]
? (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifscope permanent [ethernet]
toddmartin@Todds-MacBook-Pro ~ % arp -d 192.168.0.1 && arp -a
arp: writing to routing socket: Operation not permitted
toddmartin@Todds-MacBook-Pro ~ % sudo arp -d 192.168.0.1 && arp -a
Password:
192.168.0.1 (192.168.0.1) deleted
? (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifscope permanent [ethernet]
toddmartin@Todds-MacBook-Pro ~ % 
```

https://github.com/martintc/ser321-spring2021-A-tcmart14

Todd Martin
tcmart14



```
      1 0.000000      Apple_d9:97:f1        Broadcast           ARP        42  Who has 192.168.0.1? Tell 192.168.0.56
∨ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: Apple_d9:97:f1 (a4:5e:60:d9:97:f1)
    Sender IP address: 192.168.0.56
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.0.1
```

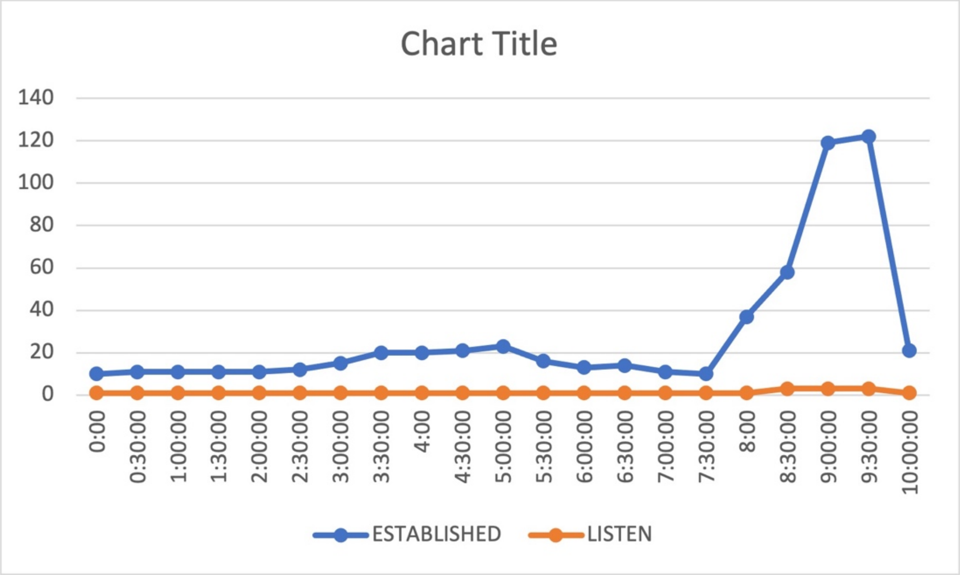


```
∨ Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: ZyxelCom_45:03:30 (bc:99:11:45:03:30)
    Sender IP address: 192.168.0.1
    Target MAC address: Apple_d9:97:f1 (a4:5e:60:d9:97:f1)
    Target IP address: 192.168.0.56
```

1. For request: request (1). For reply: reply (2)
2. 28 bytes for a request and 28 bytes for a reply
3. 00:00:00:00:00:00
4. 1

Todd Martin
tcmart14

Task 2.2



Data is included in file as named by the task

Task 2.3

Step 1:

a) 8 frames for the total exchange between the client and server, this includes the syn-ack-syn exchanges.
b) 2 frames from the "client" to send the data and 2 frames from the "Server" to acknowledge receiving the data
c) For both sides, including the syn-ack-syn exchange at the beginning, a total of 486 bytes were exchanged. The data for "SER321" and "Rocks!" consisted of 14 bytes with 238 bytes between client and server for the exchange. If just looking at the total bytes for the exchange versus the total bytes including the initial syn-ask-syn exchange, that is 51.03% not needed.

Step 2:

a) 2 frames
b) 2 packets
c) 78 bytes
d) The main difference between the UDP and TCP connections was the amount of overhead of the packets. TCP requires a syn-ack-syn for initial connection and has a chain of custody built in to ensure the integrity of packet transmission. UDP lacks these features and doesn't care to make sure whether or not a packet was received.

Relevant captures are in the git repo.

Task 2.4

Traceroute from my local machine on home network is the file 2.4.2-traceroute.csv

Traceroute from another network such as a coffee shop was not possible due to covid restrictions.

Traceroute from SSH into general.asu.edu is saved in repo as Task2.4_ASU_Route.txt

a) SSH into ASU was faster
b) Traceroute from home network had the fewest hops
c) Lines 2 and 3 which are ran through the same network, but different nodes on the network.