Assignment 1 SER 321 – Michael Krasnik {mkrasnik}

https://github.com/mkrasnik2001/ser321-spring25B-mkrasnik

1. Command line tasks (15 points)

Amazon Linux 2023.6.20250303:

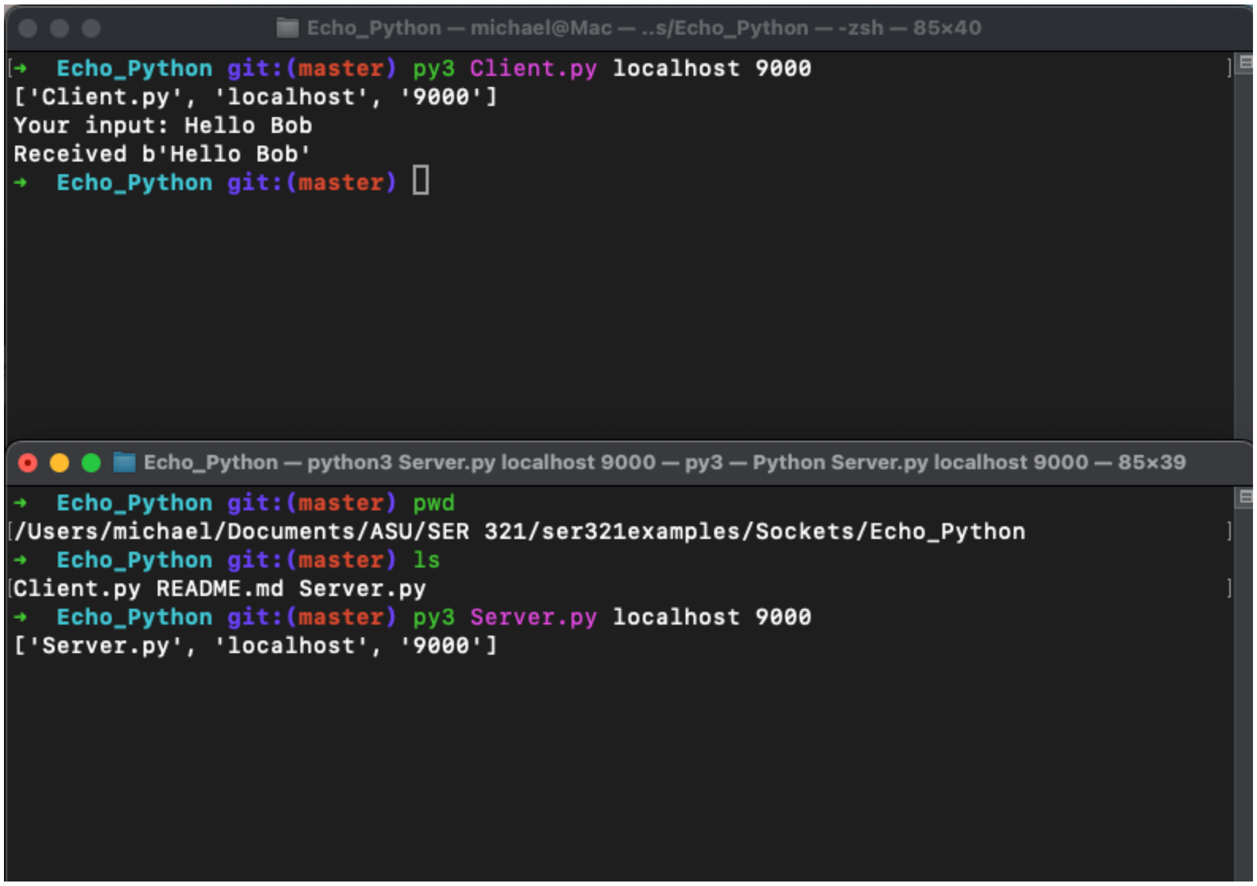
(jfyi “;” was not in the commands, adding it to represent 2 commands in that point)

1. `mkdir cli_assignment`
2. `cd cli_assignment`
3. `touch stuff.txt`
4. `cat >> stuff.txt`
5. `wc stuff.txt`
6. `cat >> stuff.txt`
7. `mkdir draft`
8. `mv stuff.txt draft`
9. `cd draft; touch .secret.txt`
10. `cd .. ;cp -r draft final`
11. `mv draft draft.remove`
12. `mv draft.remove final`
13. `cd ..; ls -la`
14. `zcat NASA_access_log_Aug95.gz | less`
15. `gunzip NASA_access_log_Aug95.gz`
16. `mv NASA_access_log_Aug95 logs.txt`
17. `mv logs.txt cli_assignment`
18. `head -n 100 logs.txt`
19. `head -n 100 logs.txt >> logs_top_100.txt`
20. `tail -n 100 logs.txt`
21. `tail -n 100 logs.txt >> logs_bottomg_100.txt`
22. `cat logs_top_100.txt logs_bottom_100.txt > logs_snapshot.txt`
23. `echo "mkrasnik: This is a great assignment $(date)" >> logs_snapshot.txt`
24. `less logs.txt`
25. `tail -n +2 marks.csv | cut -d’%’ -f1`
26. `tail -n +2 marks.csv | cut -d’%’ -f4 | sort -n`
27. `awk -F'%' 'NR > 1 { sum+=$3; count++ } END { if(count>0) print "avg:", sum/count }' marks.csv`

```
28.     awk -F'%' 'NR > 1 { sum+=$3; count++ } END {
  if(count>0) print "avg:", sum/count }' marks.csv > done.txt
29.     mv done.txt final
30.     cd final ; mv done.txt average.txt
```

2. Some Setup and Examples (30 points)

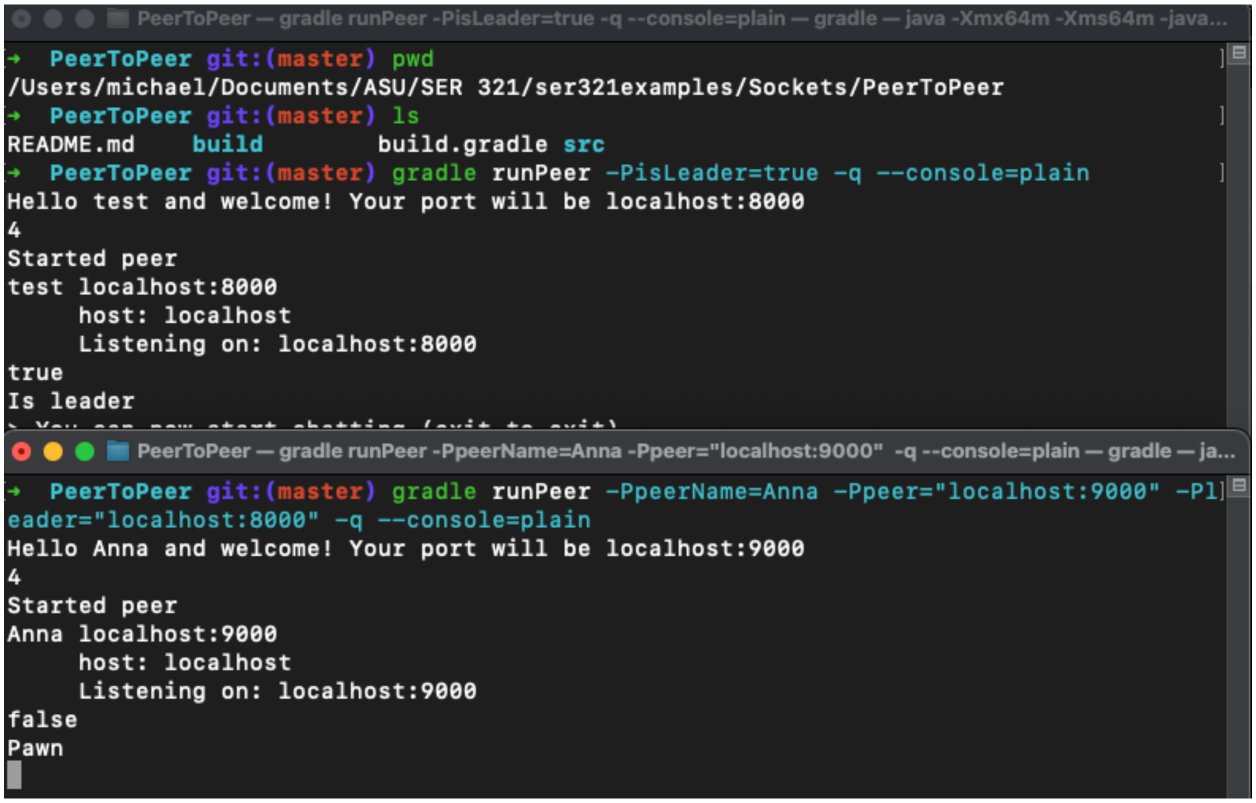
2.2 Example 1 (Echo_Python)



```
Echo_Python — michael@Mac — ..s/Echo_Python — -zsh — 85×40
→  Echo_Python git:(master) py3 Client.py localhost 9000
['Client.py', 'localhost', '9000']
Your input: Hello Bob
Received b'Hello Bob'
→  Echo_Python git:(master)
```

```
Echo_Python — python3 Server.py localhost 9000 — py3 — Python Server.py localhost 9000 — 85×39
→  Echo_Python git:(master) pwd
/Users/michael/Documents/ASU/SER 321/ser321examples/Sockets/Echo_Python
→  Echo_Python git:(master) ls
Client.py README.md Server.py
→  Echo_Python git:(master) py3 Server.py localhost 9000
['Server.py', 'localhost', '9000']
```

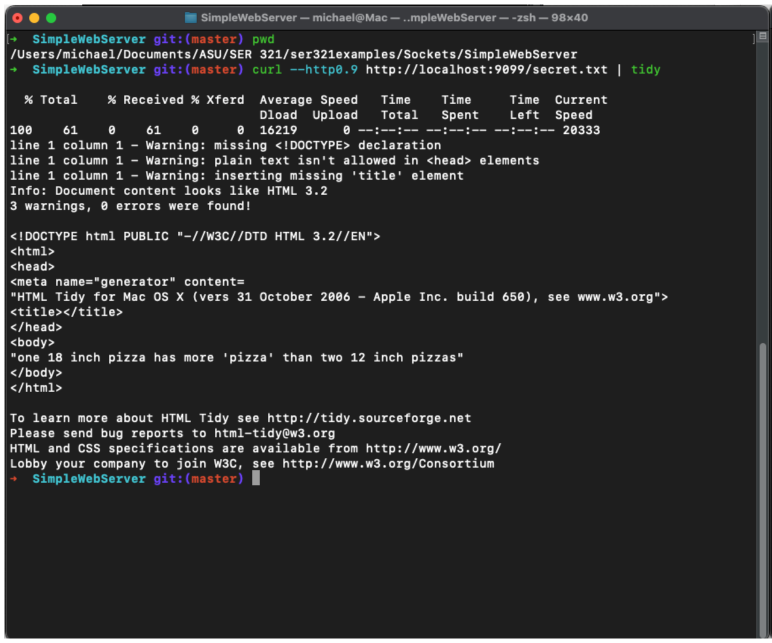
Example 2 (PeertoPeer)



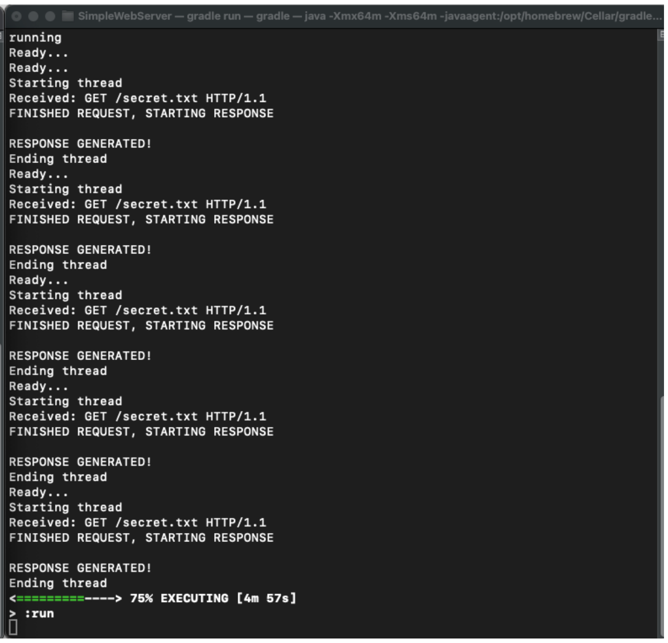
```
PeerToPeer — gradle runPeer -PisLeader=true -q --console=plain — gradle — java -Xmx64m -Xms64m -java...
→  PeerToPeer git:(master) pwd
/Users/michael/Documents/ASU/SER 321/ser321examples/Sockets/PeerToPeer
→  PeerToPeer git:(master) ls
README.md     build         build.gradle src
→  PeerToPeer git:(master) gradle runPeer -PisLeader=true -q --console=plain
Hello test and welcome! Your port will be localhost:8000
4
Started peer
test localhost:8000
     host: localhost
     Listening on: localhost:8000
true
Is leader
> You can now start chatting (exit to exit)
```

```
PeerToPeer — gradle runPeer -PpeerName=Anna -Ppeer="localhost:9000" -q --console=plain — gradle — ja...
→  PeerToPeer git:(master) gradle runPeer -PpeerName=Anna -Ppeer="localhost:9000" -Pl
eader="localhost:8000" -q --console=plain
Hello Anna and welcome! Your port will be localhost:9000
4
Started peer
Anna localhost:9000
     host: localhost
     Listening on: localhost:9000
false
Pawn
```

Example 3 (SimpleWebServer)



```
SimpleWebServer — michael@Mac — ..mpleWebServer — -zsh — 98×40
→  SimpleWebServer git:(master) pwd
/Users/michael/Documents/ASU/SER 321/ser321examples/Sockets/SimpleWebServer
→  SimpleWebServer git:(master) curl --http0.9 http://localhost:9099/secret.txt | tidy

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    61    0    61    0     0  16219      0 --:--:-- --:--:-- --:--:-- 20333
line 1 column 1 - Warning: missing <!DOCTYPE> declaration
line 1 column 1 - Warning: plain text isn't allowed in <head> elements
line 1 column 1 - Warning: inserting missing 'title' element
Info: Document content looks like HTML 3.2
3 warnings, 0 errors were found!

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2//EN">
<html>
<head>
<meta name="generator" content=
"HTML Tidy for Mac OS X (vers 31 October 2006 - Apple Inc. build 650), see www.w3.org">
<title></title>
</head>
<body>
"one 18 inch pizza has more 'pizza' than two 12 inch pizzas"
</body>
</html>

To learn more about HTML Tidy see http://tidy.sourceforge.net
Please send bug reports to html-tidy@w3.org
HTML and CSS specifications are available from http://www.w3.org/
Lobby your company to join W3C, see http://www.w3.org/Consortium
→  SimpleWebServer git:(master)
```

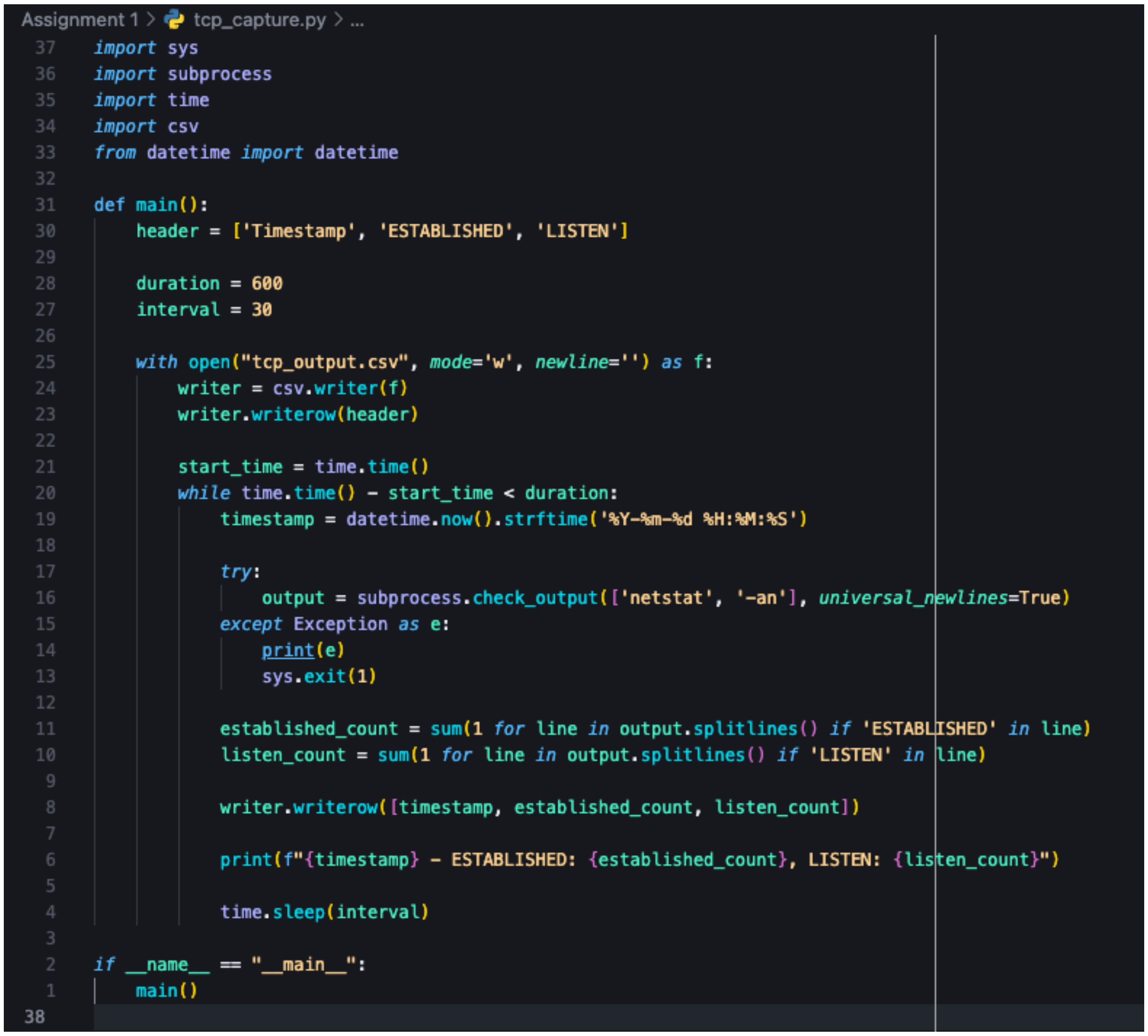


```
SimpleWebServer — gradle run — gradle — java -Xmx64m -Xms64m -javaagent:/opt/homebrew/Cellar/gradle...
running
Ready...
Ready...
Starting thread
Received: GET /secret.txt HTTP/1.1
FINISHED REQUEST, STARTING RESPONSE

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 /secret.txt 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 /secret.txt HTTP/1.1
FINISHED REQUEST, STARTING RESPONSE

RESPONSE GENERATED!
Ending thread
<=========----> 75% EXECUTING [4m 57s]
> :run
```

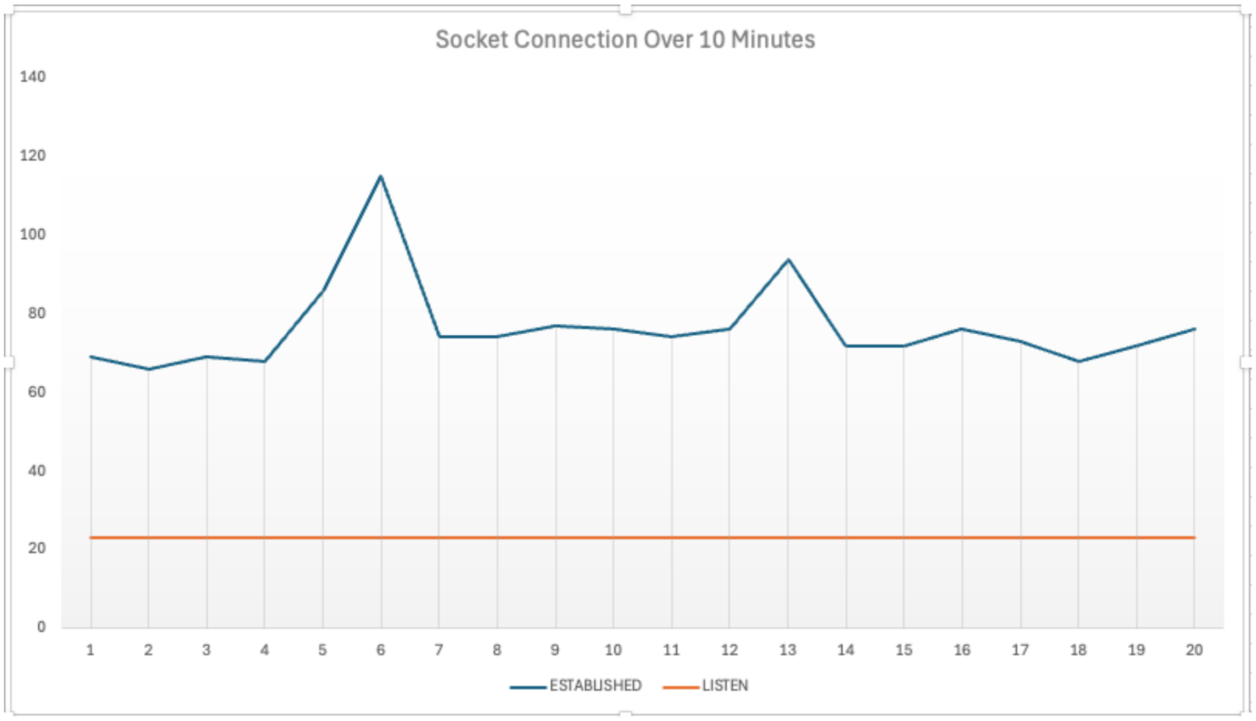
2.4

https://www.youtube.com/watch?v=0VQmK8nWNuQ

3.1

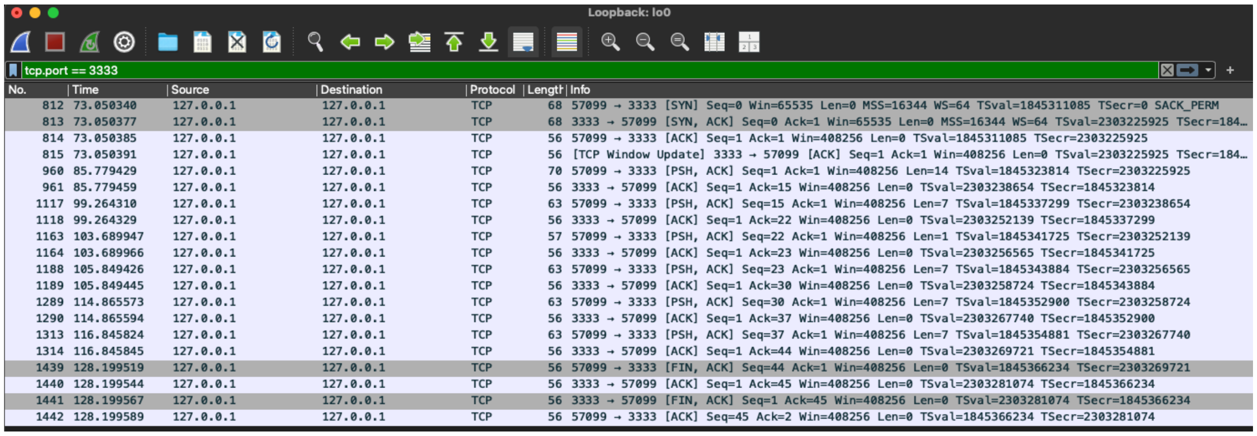


```python
import sys
import subprocess
import time
import csv
from datetime import datetime

def main():
    header = ['Timestamp', 'ESTABLISHED', 'LISTEN']

    duration = 600
    interval = 30

    with open("tcp_output.csv", mode='w', newline='') as f:
        writer = csv.writer(f)
        writer.writerow(header)

        start_time = time.time()
        while time.time() - start_time < duration:
            timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')

            try:
                output = subprocess.check_output(['netstat', '-an'], universal_newlines=True)
            except Exception as e:
                print(e)
                sys.exit(1)

            established_count = sum(1 for line in output.splitlines() if 'ESTABLISHED' in line)
            listen_count = sum(1 for line in output.splitlines() if 'LISTEN' in line)

            writer.writerow([timestamp, established_count, listen_count])

            print(f"{timestamp} - ESTABLISHED: {established_count}, LISTEN: {listen_count}")

            time.sleep(interval)

if __name__ == "__main__":
    main()
```



(Socket connections established every 30 seconds for 10 minutes)

3.2

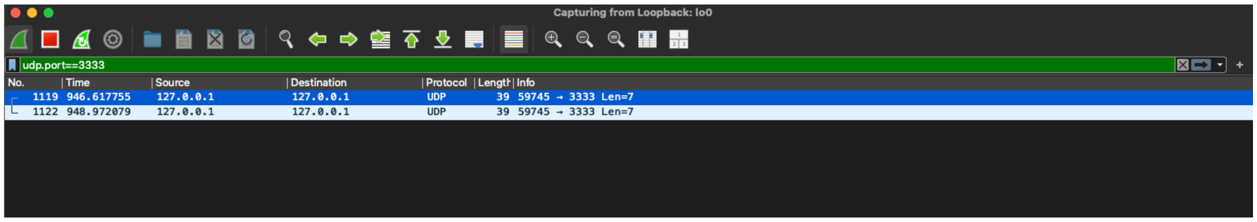


```
tcp.port == 3333
No.   Time        Source     Destination  Protocol  Length Info
  812 73.050340   127.0.0.1  127.0.0.1    TCP       68 57099 → 3333 [SYN] Seq=0 Win=65535 Len=0 MSS=16344 WS=64 TSval=1845311085 TSecr=0 SACK_PERM
  813 73.050377   127.0.0.1  127.0.0.1    TCP       68 3333 → 57099 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=16344 WS=64 TSval=2303225925 TSecr=184…
  814 73.050385   127.0.0.1  127.0.0.1    TCP       56 57099 → 3333 [ACK] Seq=1 Ack=1 Win=408256 Len=0 TSval=1845311085 TSecr=2303225925
  815 73.050391   127.0.0.1  127.0.0.1    TCP       56 [TCP Window Update] 3333 → 57099 [ACK] Seq=1 Ack=1 Win=408256 Len=0 TSval=2303225925 TSecr=184…
  960 85.779429   127.0.0.1  127.0.0.1    TCP       70 57099 → 3333 [PSH, ACK] Seq=1 Ack=1 Win=408256 Len=14 TSval=1845323814 TSecr=2303225925
  961 85.779459   127.0.0.1  127.0.0.1    TCP       56 3333 → 57099 [ACK] Seq=1 Ack=15 Win=408256 Len=0 TSval=2303238654 TSecr=1845323814
 1117 99.264310   127.0.0.1  127.0.0.1    TCP       63 57099 → 3333 [PSH, ACK] Seq=15 Ack=1 Win=408256 Len=7 TSval=1845337299 TSecr=2303238654
 1118 99.264329   127.0.0.1  127.0.0.1    TCP       56 3333 → 57099 [ACK] Seq=1 Ack=22 Win=408256 Len=0 TSval=2303252139 TSecr=1845337299
 1163 103.689947  127.0.0.1  127.0.0.1    TCP       57 57099 → 3333 [PSH, ACK] Seq=22 Ack=1 Win=408256 Len=1 TSval=1845341725 TSecr=2303252139
 1164 103.689966  127.0.0.1  127.0.0.1    TCP       56 3333 → 57099 [ACK] Seq=1 Ack=23 Win=408256 Len=0 TSval=2303256565 TSecr=1845341725
 1188 105.849426  127.0.0.1  127.0.0.1    TCP       63 57099 → 3333 [PSH, ACK] Seq=23 Ack=1 Win=408256 Len=7 TSval=1845343884 TSecr=2303256565
 1189 105.849445  127.0.0.1  127.0.0.1    TCP       56 3333 → 57099 [ACK] Seq=1 Ack=30 Win=408256 Len=0 TSval=2303258724 TSecr=1845343884
 1289 114.865573  127.0.0.1  127.0.0.1    TCP       63 57099 → 3333 [PSH, ACK] Seq=30 Ack=1 Win=408256 Len=7 TSval=1845352900 TSecr=2303258724
 1290 114.865594  127.0.0.1  127.0.0.1    TCP       56 3333 → 57099 [ACK] Seq=1 Ack=37 Win=408256 Len=0 TSval=2303267740 TSecr=1845352900
 1313 116.845824  127.0.0.1  127.0.0.1    TCP       63 57099 → 3333 [PSH, ACK] Seq=37 Ack=1 Win=408256 Len=7 TSval=1845354881 TSecr=2303267740
 1314 116.845845  127.0.0.1  127.0.0.1    TCP       56 3333 → 57099 [ACK] Seq=1 Ack=44 Win=408256 Len=0 TSval=2303269721 TSecr=1845354881
 1439 128.199519  127.0.0.1  127.0.0.1    TCP       56 57099 → 3333 [FIN, ACK] Seq=44 Ack=1 Win=408256 Len=0 TSval=1845366234 TSecr=2303269721
 1440 128.199544  127.0.0.1  127.0.0.1    TCP       56 3333 → 57099 [ACK] Seq=1 Ack=45 Win=408256 Len=0 TSval=2303281074 TSecr=1845366234
 1441 128.199567  127.0.0.1  127.0.0.1    TCP       56 3333 → 57099 [FIN, ACK] Seq=1 Ack=45 Win=408256 Len=0 TSval=2303281074 TSecr=1845366234
 1442 128.199589  127.0.0.1  127.0.0.1    TCP       56 57099 → 3333 [ACK] Seq=45 Ack=2 Win=408256 Len=0 TSval=1845366234 TSecr=2303281074
```

(TCP Stream)

a) The first command we inputted started a server that listened to the TCP port 3333. This server was started using netcat which is a network utility program. The -l flag defines what port to listen to and the -k flag allows for multiple connections. The second command opened a client connection to that same port on the local machine (127.0.0.1). By specifying the port number after the IP we are telling the client to go to connect to that port for the server service.

b) 4 frames were sent back and forth to capture these 2 lines.
c) 2 packets client packets were sent in this TCP stream for the data.
d) 7 packets were sent for the entire conversation.
e) 7 bytes for SER321 and 7 bytes for Rocks! Totaling to 14 bytes for the data sent.
f) 63 + 56 + 63 + 56 = 238 bytes sent over the wire.
g) 34 times the size of the data was the actual overhead.



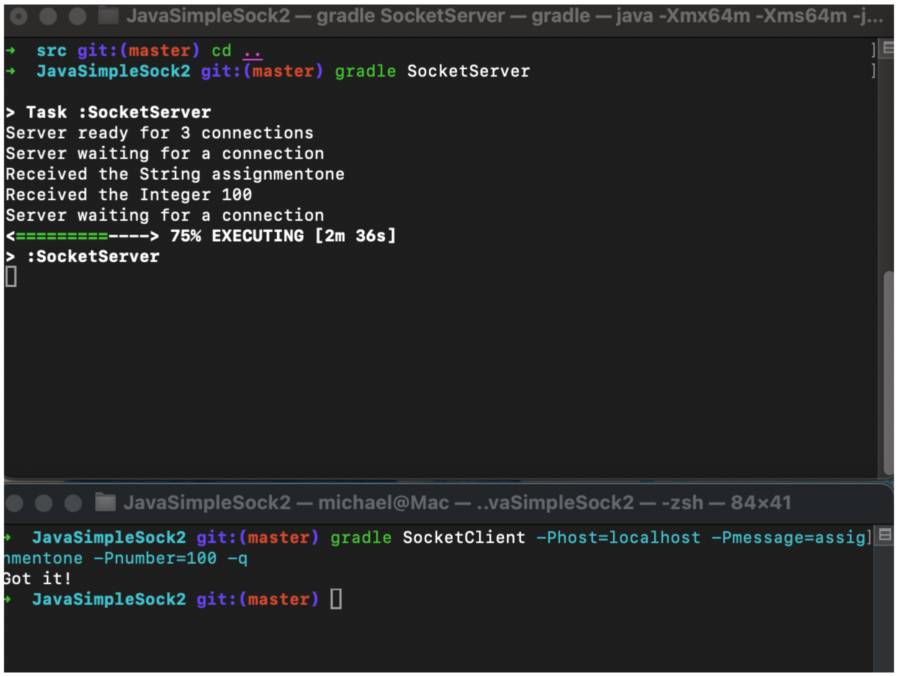
```
udp.port==3333
No.   Time        Source     Destination  Protocol Length Info
1119  946.617755  127.0.0.1  127.0.0.1    UDP      39     59745 → 3333 Len=7
1122  948.972079  127.0.0.1  127.0.0.1    UDP      39     59745 → 3333 Len=7
```

(UDP Stream)

a) Similarly to TCP we used netcat to start a server that allows multiple connections into port 3333 where the -u flag defined the protocol to be UDP in this case. We connected a client to the local machine's IP and to this UDP port 3333.
b) 2 frames were needed to capture these 2 lines.
c) 2 packets were used to capture these 2 lines.
d) 2 packets were also used to capture the whole process.
e) 39 + 39 = 78 bytes went over the entire wire.
f) 14 bytes were used to capture only the data. (7 bytes per string)
g) There was only 5 times more overhead overall sent compared to just the bytes sent for the data.
h) The difference is massive as far as the relative overhead between UDP and TCP. UDP had 6 times less relative overhead than TCP. This is because TCP adds the overhead of managing the connection, its termination and it's reliability to make sure the data is actually sent over whereas UDP is connectionless and doesn't do any of that, it uses a much smaller header and does not require extra packets to manage the connection.

3.3.1

https://youtu.be/c9qEwEV0LjE

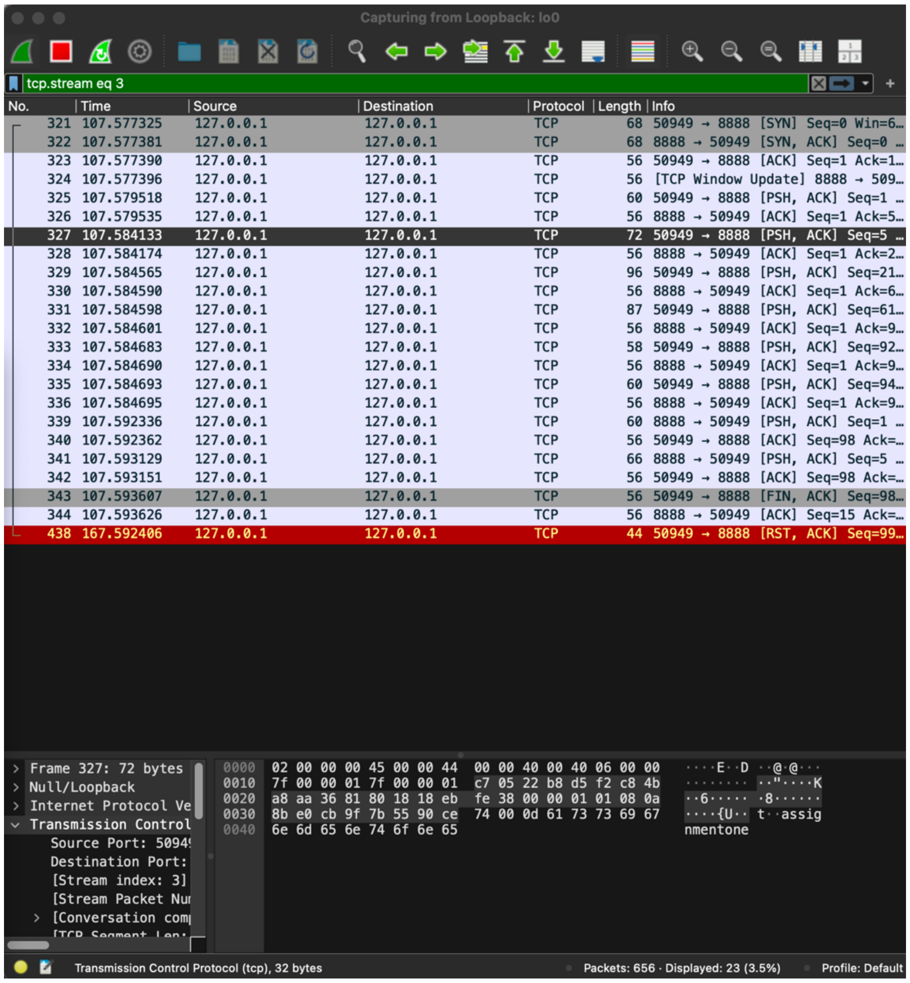


JavaSimpleSock2 — gradle SocketServer — gradle — java -Xmx64m -Xms64m -j...

```
→  src git:(master) cd ..
→  JavaSimpleSock2 git:(master) gradle SocketServer

> Task :SocketServer
Server ready for 3 connections
Server waiting for a connection
Received the String assignmentone
Received the Integer 100
Server waiting for a connection
<=========----> 75% EXECUTING [2m 36s]
> :SocketServer
```

JavaSimpleSock2 — michael@Mac — ..vaSimpleSock2 — -zsh — 84×41

```
→  JavaSimpleSock2 git:(master) gradle SocketClient -Phost=localhost -Pmessage=assig
nmentone -Pnumber=100 -q
Got it!
→  JavaSimpleSock2 git:(master)
```



Capturing from Loopback: lo0

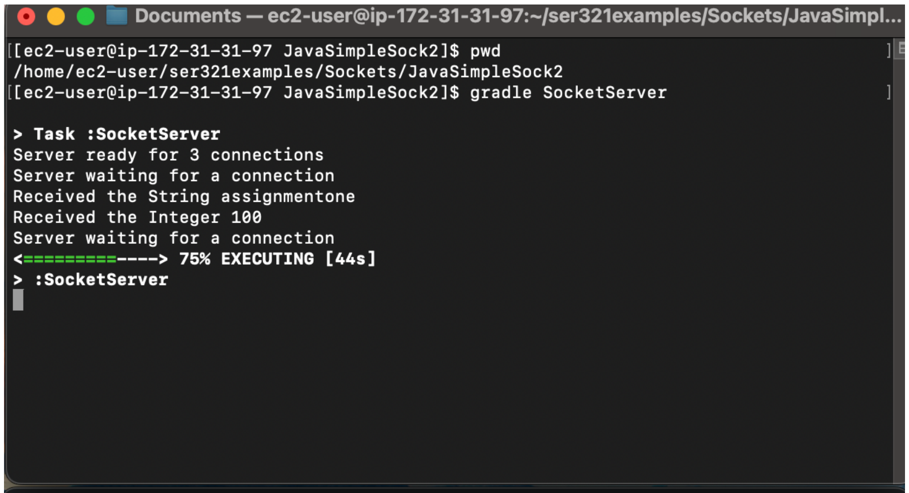
tcp.stream eq 3

```
No.   Time        Source      Destination  Protocol Length Info
  321 107.577325  127.0.0.1   127.0.0.1    TCP        68 50949 → 8888 [SYN] Seq=0 Win=6…
  322 107.577381  127.0.0.1   127.0.0.1    TCP        68 8888 → 50949 [SYN, ACK] Seq=0 …
  323 107.577390  127.0.0.1   127.0.0.1    TCP        56 50949 → 8888 [ACK] Seq=1 Ack=1…
  324 107.577396  127.0.0.1   127.0.0.1    TCP        56 [TCP Window Update] 8888 → 509…
  325 107.579518  127.0.0.1   127.0.0.1    TCP        60 50949 → 8888 [PSH, ACK] Seq=1 …
  326 107.579535  127.0.0.1   127.0.0.1    TCP        56 8888 → 50949 [ACK] Seq=1 Ack=5…
  327 107.584133  127.0.0.1   127.0.0.1    TCP        72 50949 → 8888 [PSH, ACK] Seq=5 …
  328 107.584174  127.0.0.1   127.0.0.1    TCP        56 8888 → 50949 [ACK] Seq=1 Ack=2…
  329 107.584565  127.0.0.1   127.0.0.1    TCP        96 50949 → 8888 [PSH, ACK] Seq=21…
  330 107.584590  127.0.0.1   127.0.0.1    TCP        56 8888 → 50949 [ACK] Seq=1 Ack=6…
  331 107.584598  127.0.0.1   127.0.0.1    TCP        87 50949 → 8888 [PSH, ACK] Seq=61…
  332 107.584601  127.0.0.1   127.0.0.1    TCP        56 8888 → 50949 [ACK] Seq=1 Ack=9…
  333 107.584683  127.0.0.1   127.0.0.1    TCP        58 50949 → 8888 [PSH, ACK] Seq=92…
  334 107.584690  127.0.0.1   127.0.0.1    TCP        56 8888 → 50949 [ACK] Seq=1 Ack=9…
  335 107.584693  127.0.0.1   127.0.0.1    TCP        60 50949 → 8888 [PSH, ACK] Seq=94…
  336 107.584695  127.0.0.1   127.0.0.1    TCP        56 8888 → 50949 [ACK] Seq=1 Ack=9…
  339 107.592336  127.0.0.1   127.0.0.1    TCP        60 8888 → 50949 [PSH, ACK] Seq=1 …
  340 107.592362  127.0.0.1   127.0.0.1    TCP        56 50949 → 8888 [ACK] Seq=98 Ack=…
  341 107.593129  127.0.0.1   127.0.0.1    TCP        66 8888 → 50949 [PSH, ACK] Seq=5 …
  342 107.593151  127.0.0.1   127.0.0.1    TCP        56 50949 → 8888 [ACK] Seq=98 Ack=…
  343 107.593607  127.0.0.1   127.0.0.1    TCP        56 50949 → 8888 [FIN, ACK] Seq=98…
  344 107.593626  127.0.0.1   127.0.0.1    TCP        56 8888 → 50949 [ACK] Seq=15 Ack=…
  438 167.592406  127.0.0.1   127.0.0.1    TCP        44 50949 → 8888 [RST, ACK] Seq=99…
```

```
> Frame 327: 72 bytes
> Null/Loopback
> Internet Protocol Ve
v Transmission Control
    Source Port: 5094
    Destination Port:
    [Stream index: 3]
    [Stream Packet Nu
  > [Conversation com
    [TCP Segment Len:

0000  02 00 00 00 45 00 00 44  00 00 40 00 40 06 00 00   ····E··D ··@·@···
0010  7f 00 00 01 7f 00 00 01  c7 05 22 b8 d5 f2 c8 4b   ········ ··"····K
0020  a8 aa 36 81 80 18 18 eb  fe 38 00 00 01 01 08 0a   ··6····· ·8······
0030  8b e0 cb 9f 7b 55 90 ce  74 00 0d 61 73 73 69 67   ····{U·· t··assig
0040  6e 6d 65 6e 74 6f 6e 65                           nmentone
```

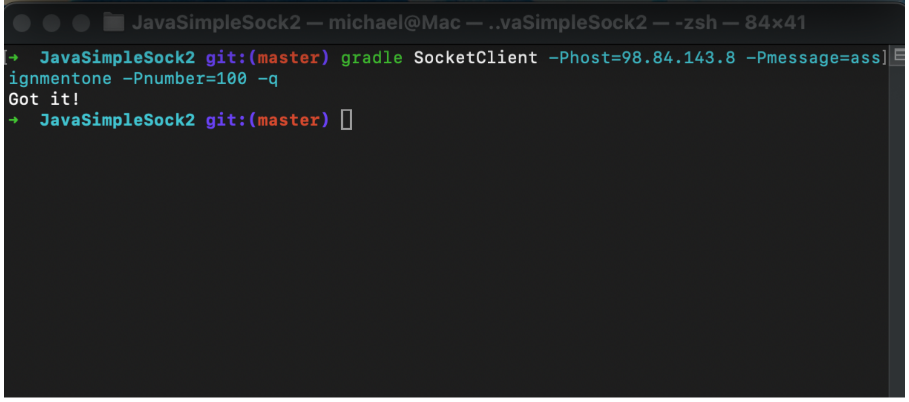
Transmission Control Protocol (tcp), 32 bytes · Packets: 656 · Displayed: 23 (3.5%) · Profile: Default

3.3.2



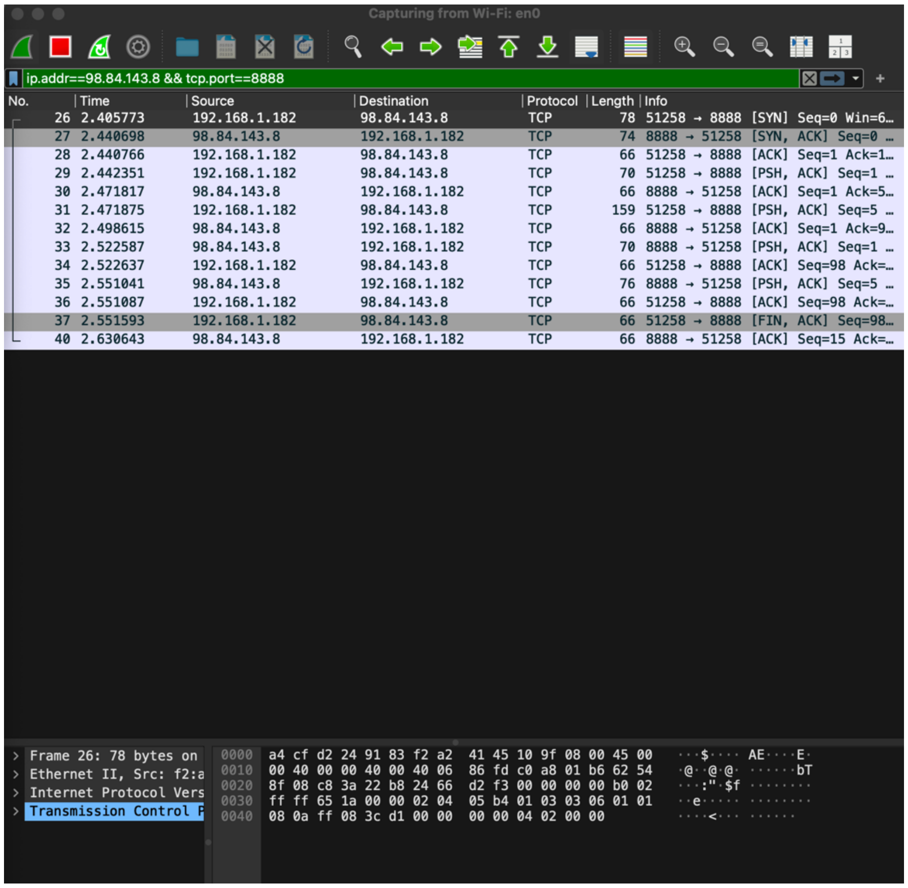
Documents — ec2-user@ip-172-31-31-97:~/ser321examples/Sockets/JavaSimpl…

```
[ec2-user@ip-172-31-31-97 JavaSimpleSock2]$ pwd
/home/ec2-user/ser321examples/Sockets/JavaSimpleSock2
[ec2-user@ip-172-31-31-97 JavaSimpleSock2]$ gradle SocketServer

> Task :SocketServer
Server ready for 3 connections
Server waiting for a connection
Received the String assignmentone
Received the Integer 100
Server waiting for a connection
<=========----> 75% EXECUTING [44s]
> :SocketServer
```



JavaSimpleSock2 — michael@Mac — ..vaSimpleSock2 — -zsh — 84×41

```
→  JavaSimpleSock2 git:(master) gradle SocketClient -Phost=98.84.143.8 -Pmessage=ass
ignmentone -Pnumber=100 -q
Got it!
→  JavaSimpleSock2 git:(master) 
```



Capturing from Wi-Fi: en0

```
ip.addr==98.84.143.8 && tcp.port==8888
```

```
No.   Time       Source          Destination     Protocol  Length Info
   26 2.405773   192.168.1.182   98.84.143.8     TCP       78  51258 → 8888 [SYN] Seq=0 Win=6…
   27 2.440698   98.84.143.8     192.168.1.182   TCP       74  8888 → 51258 [SYN, ACK] Seq=0 …
   28 2.440766   192.168.1.182   98.84.143.8     TCP       66  51258 → 8888 [ACK] Seq=1 Ack=1…
   29 2.442351   192.168.1.182   98.84.143.8     TCP       70  51258 → 8888 [PSH, ACK] Seq=1 …
   30 2.471817   98.84.143.8     192.168.1.182   TCP       66  8888 → 51258 [ACK] Seq=1 Ack=5…
   31 2.471875   192.168.1.182   98.84.143.8     TCP      159  51258 → 8888 [PSH, ACK] Seq=5 …
   32 2.498615   98.84.143.8     192.168.1.182   TCP       66  8888 → 51258 [ACK] Seq=1 Ack=9…
   33 2.522587   98.84.143.8     192.168.1.182   TCP       70  8888 → 51258 [PSH, ACK] Seq=1 …
   34 2.522637   192.168.1.182   98.84.143.8     TCP       66  51258 → 8888 [ACK] Seq=98 Ack=…
   35 2.551041   98.84.143.8     192.168.1.182   TCP       76  8888 → 51258 [PSH, ACK] Seq=5 …
   36 2.551087   192.168.1.182   98.84.143.8     TCP       66  51258 → 8888 [ACK] Seq=98 Ack=…
   37 2.551593   192.168.1.182   98.84.143.8     TCP       66  51258 → 8888 [FIN, ACK] Seq=98…
   40 2.630643   98.84.143.8     192.168.1.182   TCP       66  8888 → 51258 [ACK] Seq=15 Ack=…
```

```
> Frame 26: 78 bytes on      0000  a4 cf d2 24 91 83 f2 a2  41 45 10 9f 08 00 45 00   ···$···· AE····E·
> Ethernet II, Src: f2:a     0010  00 40 00 00 40 00 40 06  86 fd c0 a8 01 b6 62 54   ·@··@·@· ······bT
> Internet Protocol Vers     0020  8f 08 c8 3a 22 b8 24 66  d2 f3 00 00 00 00 b0 02   ···:"·$f ········
> Transmission Control P     0030  ff ff 65 1a 00 00 02 04  05 b4 01 03 03 06 01 01   ··e····· ········
                             0040  08 0a ff 08 3c d1 00 00  00 00 04 02 00 00         ····<··· ······
```

The data sizes of the frames sent over increased with the server now being on AWS instead of local, because it had to go through more network layers as a pose to doing it locally. In Wireshark I had to change the capture from the loopback to my wifi adapter and filter by the ip of the EC2 instance and the tcp port number 8888. On the client side I didn't had to change anything, I just had to make sure that tcp port 8888 was in the range of allowed ports under security in the EC2 settings on AWS. The only change locally done was for the gradle call replacing pHost from "localhost" with the public IPv4 address of the EC2.

3.3.3

This way will cause issues, and you cannot do it the same way as 3.3.2. The difference is that your local IP is not accessible to the public internet and therefore requires additional steps to expose it.

3.3.4

Your localhost ip address is private whereas the AWS EC2 instance ip is public. The first way we did was just access that ip address and routed the connection to an allowed port, if we were to do it the other way we would have to setup port forwarding on my internet provider's admin dashboard to allow for a tcp connection to be made to my local network from the outside world. It's tedious, but doable. Local Ips are not routable over the public internet unless you explicitly set up the firewall in a way that allows your ip to be exposed. My router handles outbound requests no problem, because that's how we are able to access the public internet, however inbound requests are blocked by default usually by every internet provider.