https://github.com/gjmooney/ser321-spring2022-A-gjmooney

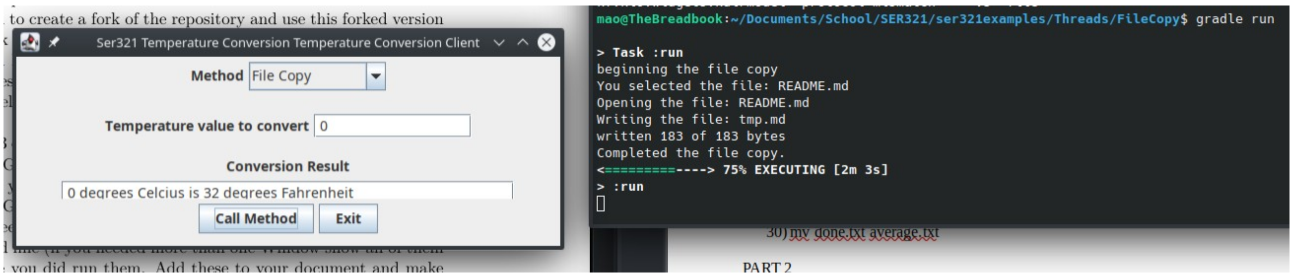
PART 1 - Using Linux

Command line tasks

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) cp -r . ../final
11) mv ../draft ../draft.remove
12) mv draft.remove final
13) ls -laR
14) zcat 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) head -100 logs.txt
19) head -100 logs.txt > logs_top_100.txt
20) tail -100 logs.txt
21) tail -100 logs.txt > logs_bottom_100.txt
22) cat logs_top_100.txt logs_bottom_100.txt > logs_snapshot.txt
23) cat >> logs_snapshot.txt
24) less logs.txt
25) awk '(NR>1)' marks.csv | cut -d % -f 1
26) cut -d % -f 4 | sort
27) awk '(NR>1)' marks.csv | cut -d % -f 3 | awk '{ total += $1; count++ } END { print total/count }'
28) awk '(NR>1)' marks.csv | cut -d % -f 3 | awk '{ total += $1; count++ } END { print total/count }' > done.txt
29) mv done.txt cli_assignment/final/
30) mv done.txt average.txt

2.2: Running examples

The first example I ran was FileCopy from the Threads folder. This example reads a user input from a text field and coverts the number to the equivalent Celsius/Fahrenheit temperature based on the users selection in a pulldown menu. It also has the ability to copy a file selected by the user.



to create a fork of the repository and use this forked version
Ser321 Temperature Conversion Temperature Conversion Client
Call Method
Exit
30) mv done.txt average.txt
PART 2

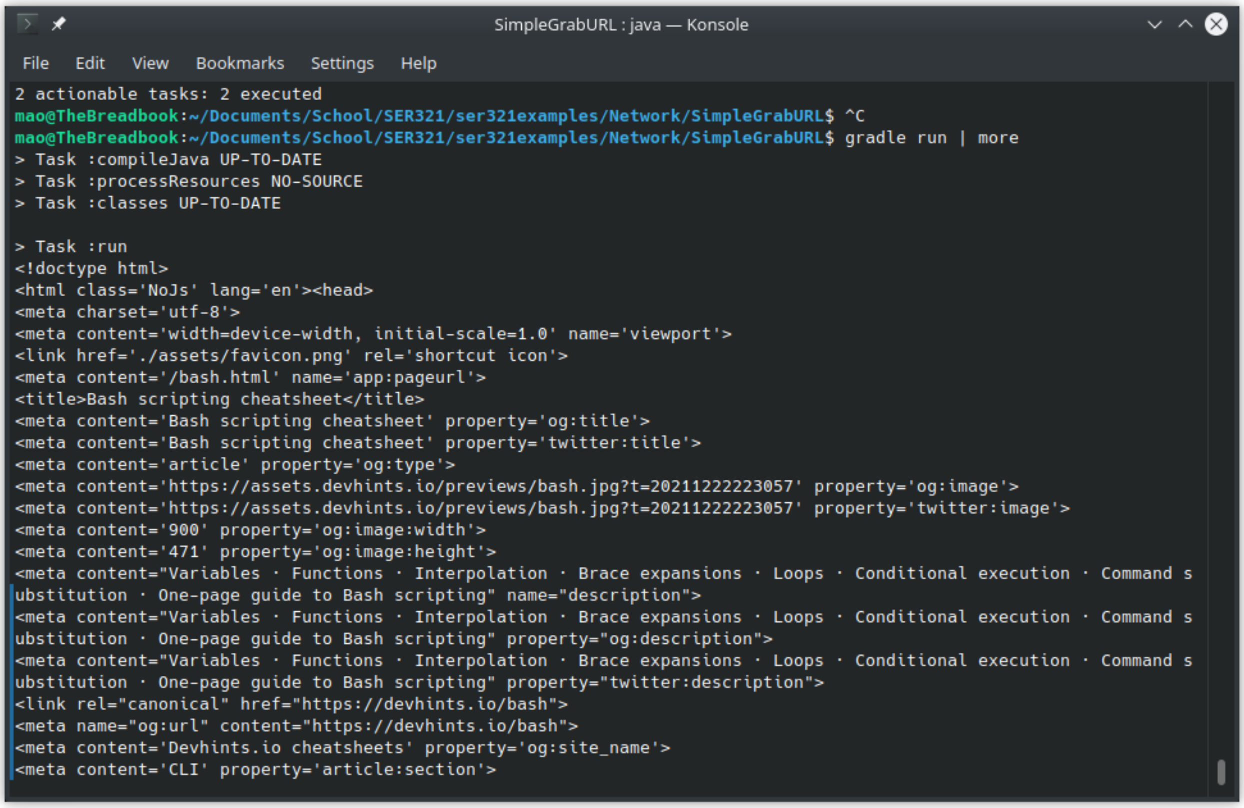
Method: File Copy

Temperature value to convert: 0

Conversion Result: 0 degrees Celcius is 32 degrees Fahrenheit

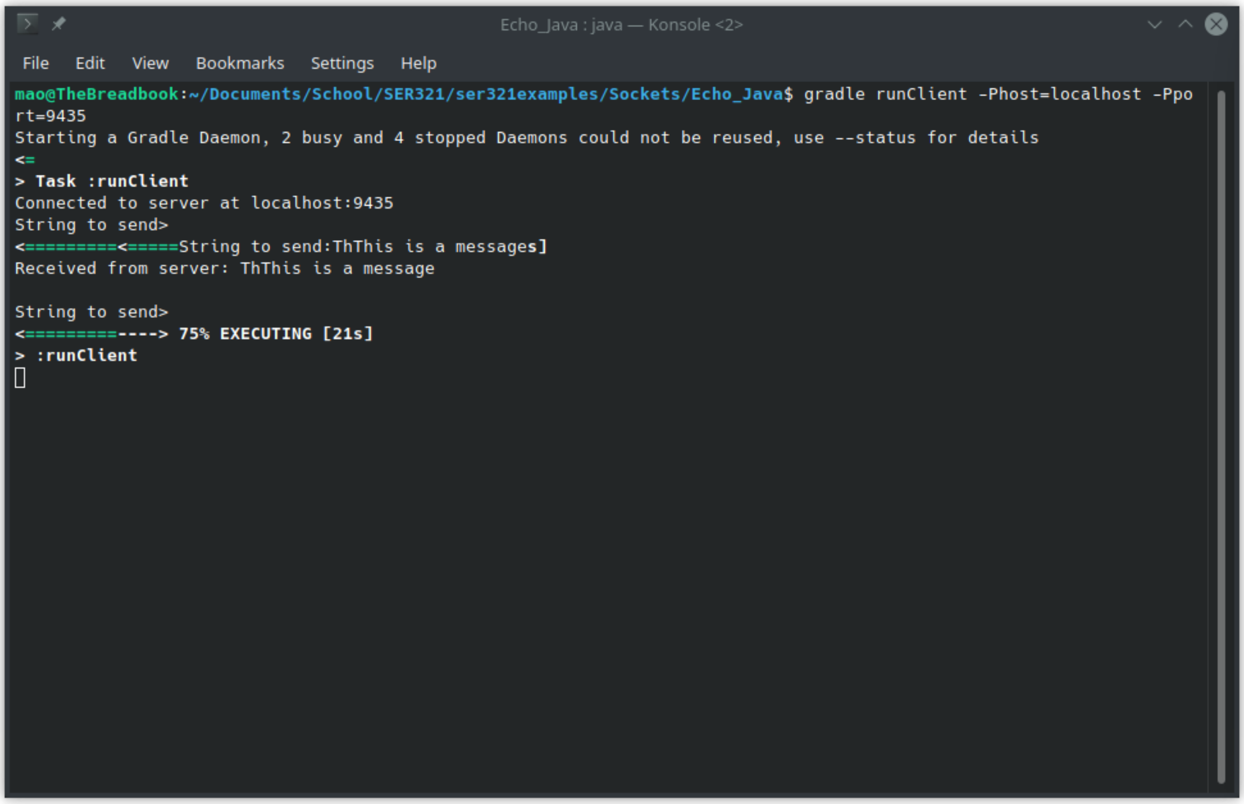
```
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Threads/FileCopy$ gradle run

> Task :run
beginning the file copy
You selected the file: README.md
Opening the file: README.md
Writing the file: tmp.md
written 183 of 183 bytes
Completed the file copy.
<=========----> 75% EXECUTING [2m 3s]
> :run
```

The second example I ran was SimpleGrabURL from the Networking folder. This example connects to a URL (defaulting to https://devhints.io/bash) and prints the HTML of that page to the console.

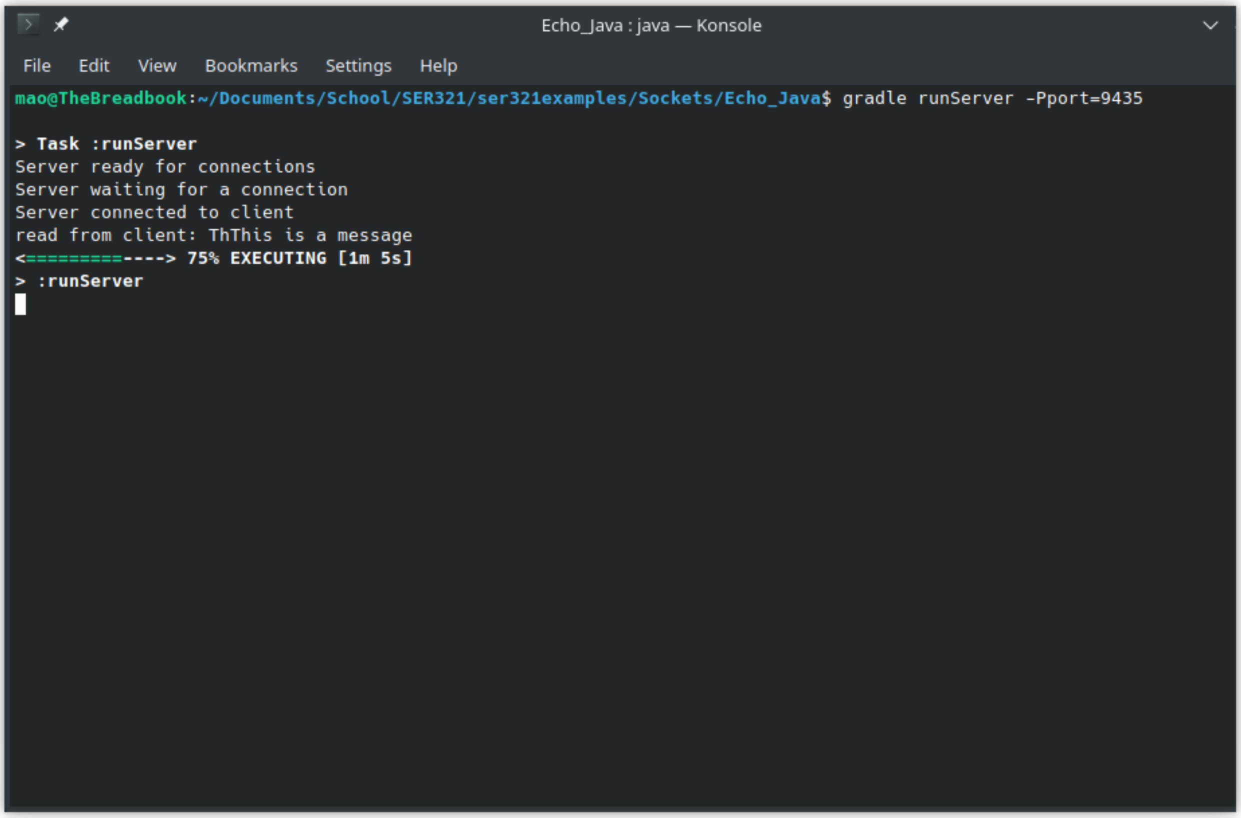


```
2 actionable tasks: 2 executed
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Network/SimpleGrabURL$ ^C
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Network/SimpleGrabURL$ gradle run | more
> Task :compileJava UP-TO-DATE
> Task :processResources NO-SOURCE
> Task :classes UP-TO-DATE

> Task :run
<!doctype html>
<html class='NoJs' lang='en'><head>
<meta charset='utf-8'>
<meta content='width=device-width, initial-scale=1.0' name='viewport'>
<link href='./assets/favicon.png' rel='shortcut icon'>
<meta content='/bash.html' name='app:pageurl'>
<title>Bash scripting cheatsheet</title>
<meta content='Bash scripting cheatsheet' property='og:title'>
<meta content='Bash scripting cheatsheet' property='twitter:title'>
<meta content='article' property='og:type'>
<meta content='https://assets.devhints.io/previews/bash.jpg?t=20211222223057' property='og:image'>
<meta content='https://assets.devhints.io/previews/bash.jpg?t=20211222223057' property='twitter:image'>
<meta content='900' property='og:image:width'>
<meta content='471' property='og:image:height'>
<meta content="Variables · Functions · Interpolation · Brace expansions · Loops · Conditional execution · Command s
ubstitution · One-page guide to Bash scripting" name="description">
<meta content="Variables · Functions · Interpolation · Brace expansions · Loops · Conditional execution · Command s
ubstitution · One-page guide to Bash scripting" property="og:description">
<meta content="Variables · Functions · Interpolation · Brace expansions · Loops · Conditional execution · Command s
ubstitution · One-page guide to Bash scripting" property="twitter:description">
<link rel="canonical" href="https://devhints.io/bash">
<meta name="og:url" content="https://devhints.io/bash">
<meta content='Devhints.io cheatsheets' property='og:site_name'>
<meta content='CLI' property='article:section'>
```

The third example I ran was Echo_Java from the Sockets folder. This example consists of a server that echos text typed into the console of the client.


Echo_Java : java — Konsole <2>
File Edit View Bookmarks Settings Help

```
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Sockets/Echo_Java$ gradle runClient -Phost=localhost -Ppo
rt=9435
Starting a Gradle Daemon, 2 busy and 4 stopped Daemons could not be reused, use --status for details
<=
> Task :runClient
Connected to server at localhost:9435
String to send>
<=========<=====String to send:ThThis is a messages]
Received from server: ThThis is a message

String to send>
<=========----> 75% EXECUTING [21s]
> :runClient
```


Echo_Java : java — Konsole
File Edit View Bookmarks Settings Help

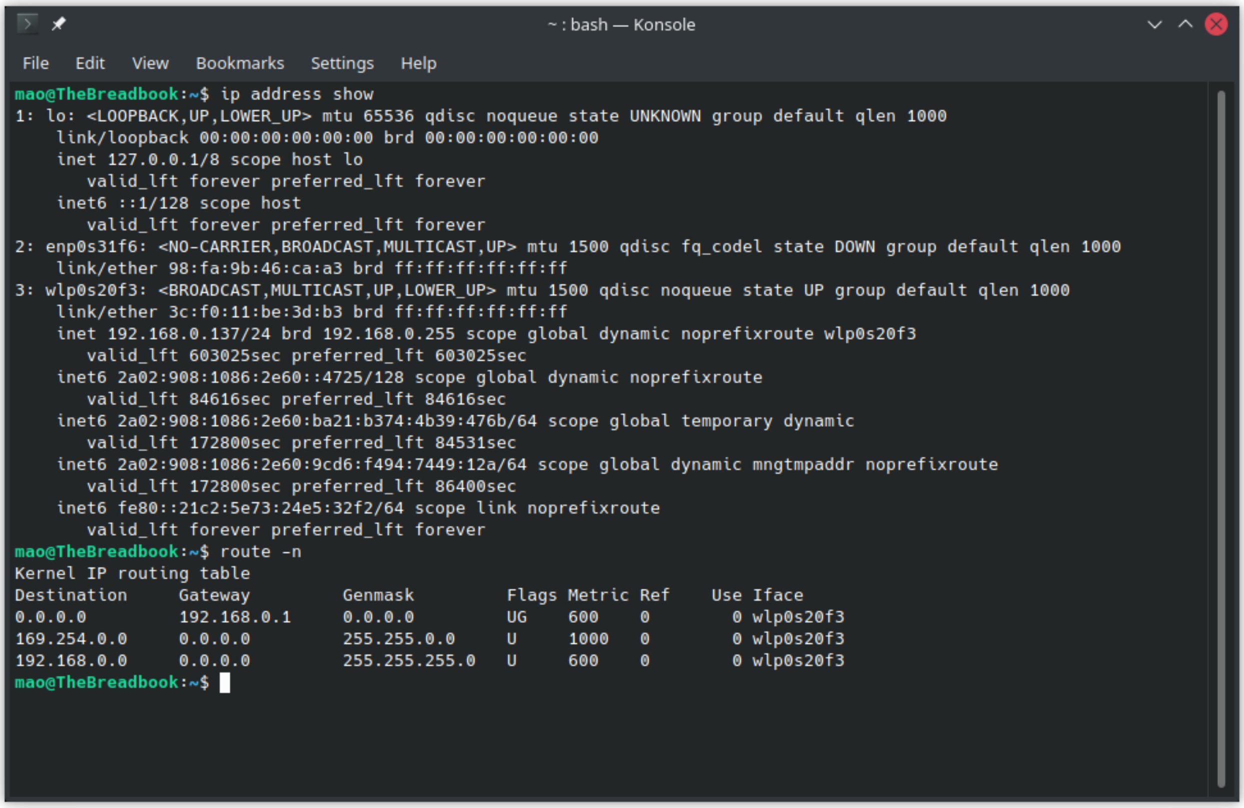
```
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Sockets/Echo_Java$ gradle runServer -Pport=9435

> Task :runServer
Server ready for connections
Server waiting for a connection
Server connected to client
read from client: ThThis is a message
<=========----> 75% EXECUTING [1m 5s]
> :runServer
```

2.4: Set up your second system

I set up AWS as my second system.
https://youtu.be/rn1G7PLF1QE

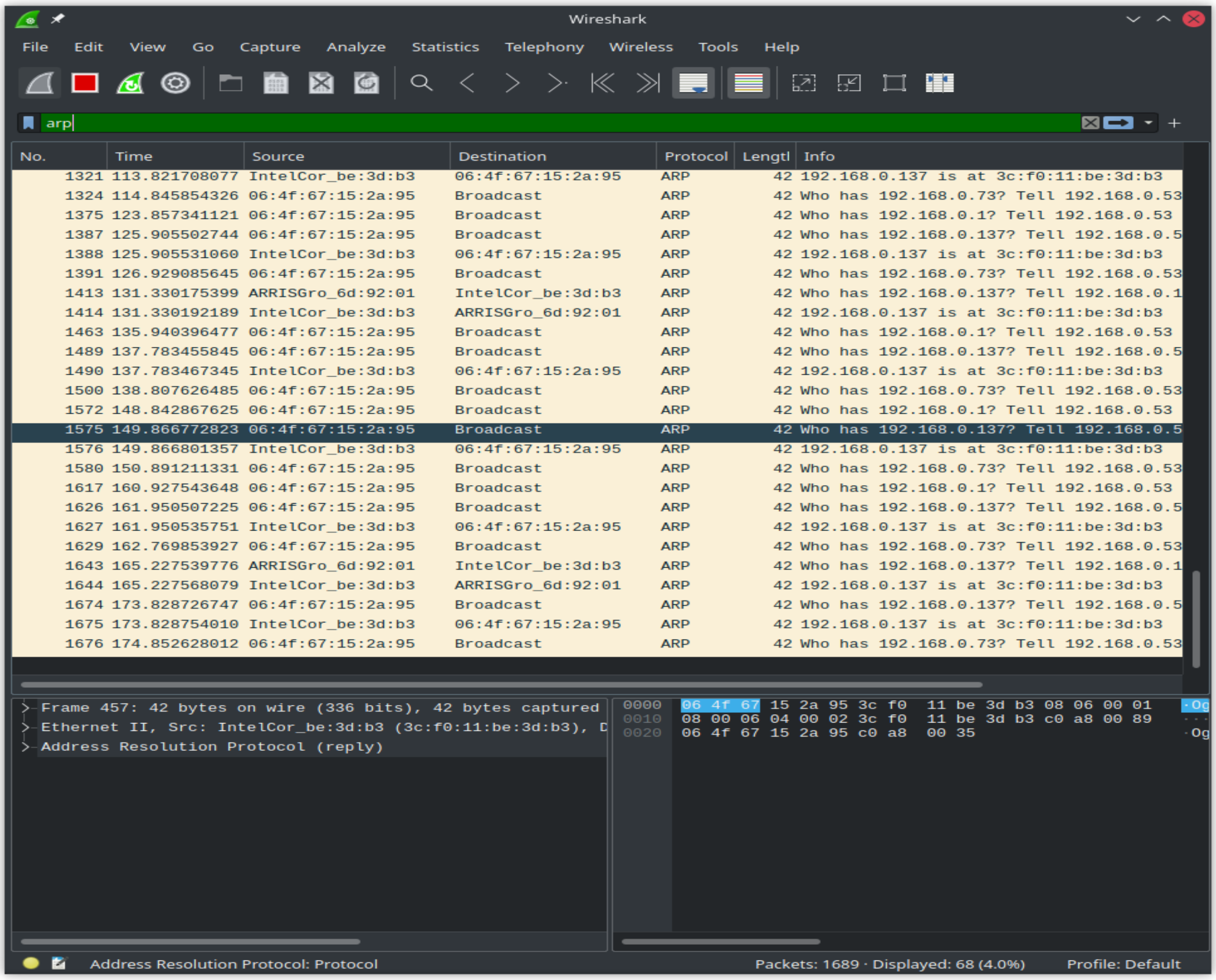
PART 2 - Networking

3.1.1 – Capture a trace



```
mao@TheBreadbook:~$ ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s31f6: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 98:fa:9b:46:ca:a3 brd ff:ff:ff:ff:ff:ff
3: wlp0s20f3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 3c:f0:11:be:3d:b3 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.137/24 brd 192.168.0.255 scope global dynamic noprefixroute wlp0s20f3
       valid_lft 603025sec preferred_lft 603025sec
    inet6 2a02:908:1086:2e60::4725/128 scope global dynamic noprefixroute
       valid_lft 84616sec preferred_lft 84616sec
    inet6 2a02:908:1086:2e60:ba21:b374:4b39:476b/64 scope global temporary dynamic
       valid_lft 172800sec preferred_lft 84531sec
    inet6 2a02:908:1086:2e60:9cd6:f494:7449:12a/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 172800sec preferred_lft 86400sec
    inet6 fe80::21c2:5e73:24e5:32f2/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
mao@TheBreadbook:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.0.1     0.0.0.0         UG    600    0        0 wlp0s20f3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 wlp0s20f3
192.168.0.0     0.0.0.0         255.255.255.0   U     600    0        0 wlp0s20f3
mao@TheBreadbook:~$
```

3.1.1.3 – Wireshark with ARP filter



```
arp
```

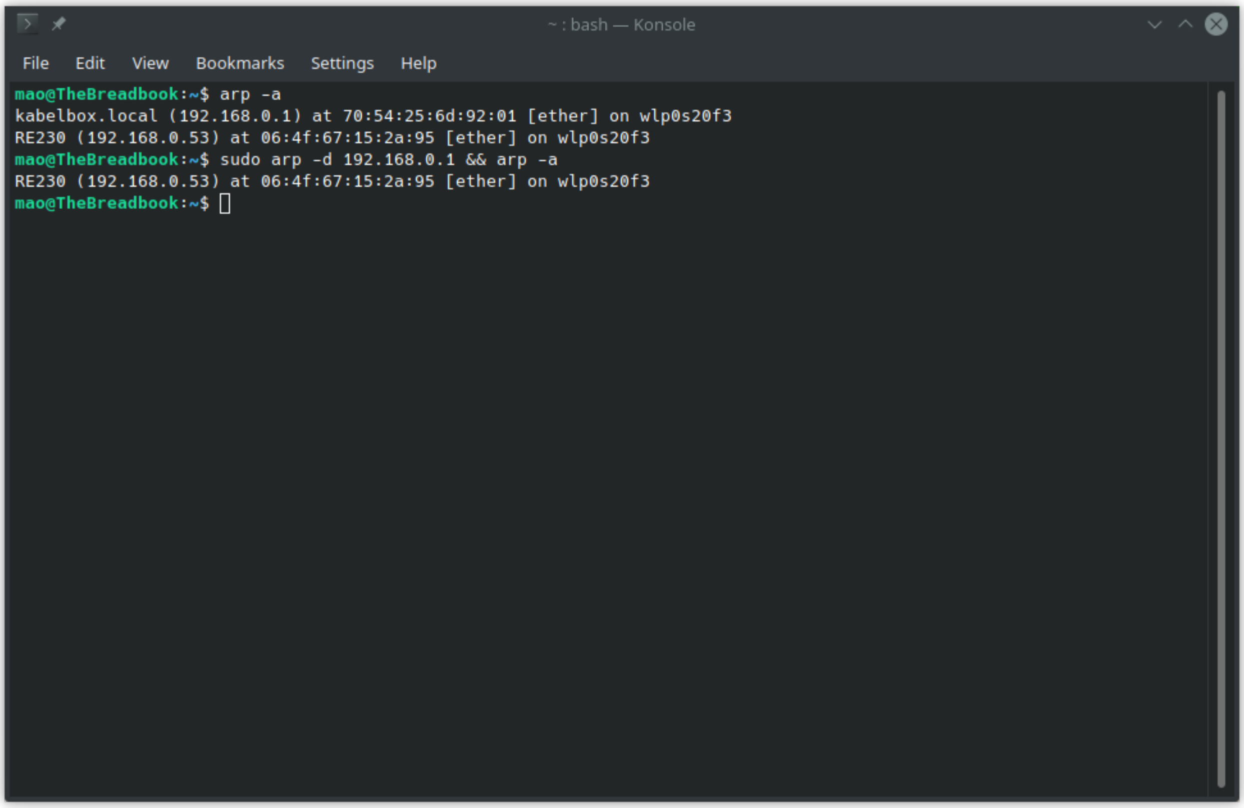
No.	Time	Source	Destination	Protocol	Lengtl	Info
1321	113.821708077	IntelCor_be:3d:b3	06:4f:67:15:2a:95	ARP	42	192.168.0.137 is at 3c:f0:11:be:3d:b3
1324	114.845854326	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.73? Tell 192.168.0.53
1375	123.857341121	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.53
1387	125.905502744	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.137? Tell 192.168.0.5
1388	125.905531060	IntelCor_be:3d:b3	06:4f:67:15:2a:95	ARP	42	192.168.0.137 is at 3c:f0:11:be:3d:b3
1391	126.929085645	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.73? Tell 192.168.0.53
1413	131.330175399	ARRISGro_6d:92:01	IntelCor_be:3d:b3	ARP	42	Who has 192.168.0.137? Tell 192.168.0.1
1414	131.330192189	IntelCor_be:3d:b3	ARRISGro_6d:92:01	ARP	42	192.168.0.137 is at 3c:f0:11:be:3d:b3
1463	135.940396477	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.53
1489	137.783455845	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.137? Tell 192.168.0.5
1490	137.783467345	IntelCor_be:3d:b3	06:4f:67:15:2a:95	ARP	42	192.168.0.137 is at 3c:f0:11:be:3d:b3
1500	138.807626485	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.73? Tell 192.168.0.53
1572	148.842867625	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.53
1575	149.866772823	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.137? Tell 192.168.0.5
1576	149.866801357	IntelCor_be:3d:b3	06:4f:67:15:2a:95	ARP	42	192.168.0.137 is at 3c:f0:11:be:3d:b3
1580	150.891211331	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.73? Tell 192.168.0.53
1617	160.927543648	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.53
1626	161.950507225	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.137? Tell 192.168.0.5
1627	161.950535751	IntelCor_be:3d:b3	06:4f:67:15:2a:95	ARP	42	192.168.0.137 is at 3c:f0:11:be:3d:b3
1629	162.769853927	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.73? Tell 192.168.0.53
1643	165.227539776	ARRISGro_6d:92:01	IntelCor_be:3d:b3	ARP	42	Who has 192.168.0.137? Tell 192.168.0.1
1644	165.227568079	IntelCor_be:3d:b3	ARRISGro_6d:92:01	ARP	42	192.168.0.137 is at 3c:f0:11:be:3d:b3
1674	173.828726747	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.137? Tell 192.168.0.5
1675	173.828754010	IntelCor_be:3d:b3	06:4f:67:15:2a:95	ARP	42	192.168.0.137 is at 3c:f0:11:be:3d:b3
1676	174.852628012	06:4f:67:15:2a:95	Broadcast	ARP	42	Who has 192.168.0.73? Tell 192.168.0.53

```
> Frame 457: 42 bytes on wire (336 bits), 42 bytes captured
> Ethernet II, Src: IntelCor_be:3d:b3 (3c:f0:11:be:3d:b3), D
> Address Resolution Protocol (reply)
```

```
0000  06 4f 67 15 2a 95 3c f0  11 be 3d b3 08 06 00 01   ·Og
0010  08 00 06 04 00 02 3c f0  11 be 3d b3 c0 a8 00 89   ···
0020  06 4f 67 15 2a 95 c0 a8  00 35                     ·Og
```

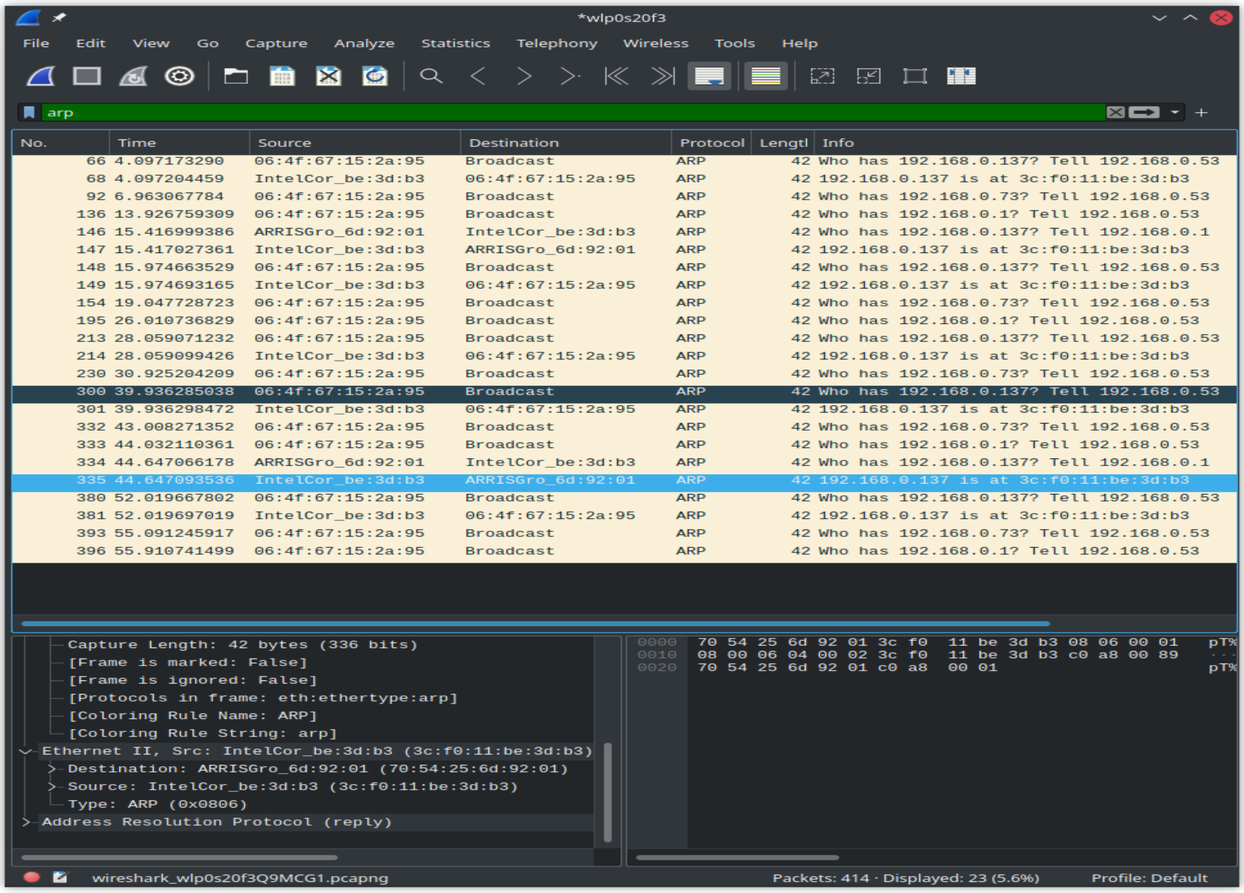
Address Resolution Protocol: Protocol   Packets: 1689 · Displayed: 68 (4.0%)   Profile: Default

3.1.1.4 – Remove default gateway



```
mao@TheBreadbook:~$ arp -a
kabelbox.local (192.168.0.1) at 70:54:25:6d:92:01 [ether] on wlp0s20f3
RE230 (192.168.0.53) at 06:4f:67:15:2a:95 [ether] on wlp0s20f3
mao@TheBreadbook:~$ sudo arp -d 192.168.0.1 && arp -a
RE230 (192.168.0.53) at 06:4f:67:15:2a:95 [ether] on wlp0s20f3
mao@TheBreadbook:~$
```

3.1.1.6 – Fetch a page


*wlp0s20f3
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
arp
```

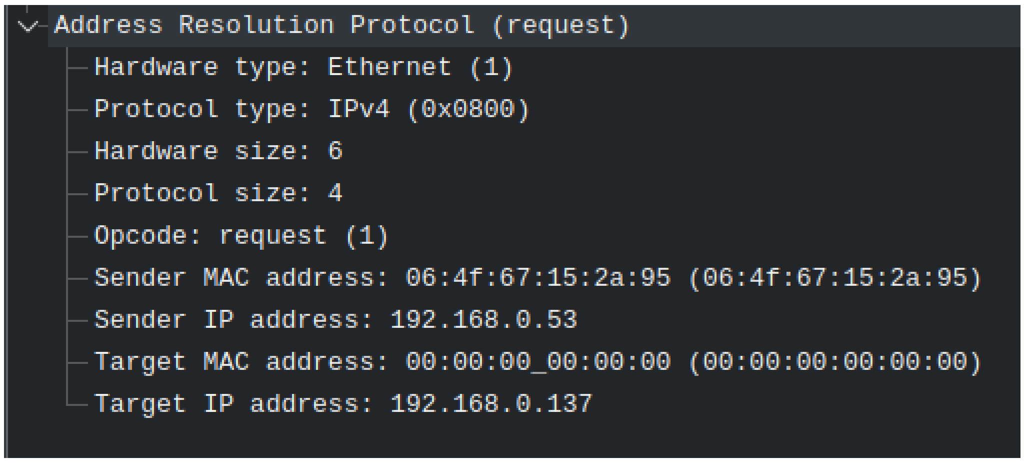
```
No.    Time          Source              Destination         Protocol Lengtl Info
    66 4.097173290   06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.137? Tell 192.168.0.53
    68 4.097204459   IntelCor_be:3d:b3   06:4f:67:15:2a:95   ARP      42 192.168.0.137 is at 3c:f0:11:be:3d:b3
    92 6.963067784   06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.73? Tell 192.168.0.53
   136 13.926759309  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.1? Tell 192.168.0.53
   146 15.416999386  ARRISGro_6d:92:01   IntelCor_be:3d:b3   ARP      42 Who has 192.168.0.137? Tell 192.168.0.1
   147 15.417027361  IntelCor_be:3d:b3   ARRISGro_6d:92:01   ARP      42 192.168.0.137 is at 3c:f0:11:be:3d:b3
   148 15.974663529  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.137? Tell 192.168.0.53
   149 15.974693165  IntelCor_be:3d:b3   06:4f:67:15:2a:95   ARP      42 192.168.0.137 is at 3c:f0:11:be:3d:b3
   154 19.047728723  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.73? Tell 192.168.0.53
   195 26.010736829  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.1? Tell 192.168.0.53
   213 28.059071232  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.137? Tell 192.168.0.53
   214 28.059099426  IntelCor_be:3d:b3   06:4f:67:15:2a:95   ARP      42 192.168.0.137 is at 3c:f0:11:be:3d:b3
   230 30.925204209  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.73? Tell 192.168.0.53
   300 39.936285038  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.137? Tell 192.168.0.53
   301 39.936298472  IntelCor_be:3d:b3   06:4f:67:15:2a:95   ARP      42 192.168.0.137 is at 3c:f0:11:be:3d:b3
   332 43.008271352  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.73? Tell 192.168.0.53
   333 44.032110361  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.1? Tell 192.168.0.53
   334 44.647066178  ARRISGro_6d:92:01   IntelCor_be:3d:b3   ARP      42 Who has 192.168.0.137? Tell 192.168.0.1
   335 44.647093536  IntelCor_be:3d:b3   ARRISGro_6d:92:01   ARP      42 192.168.0.137 is at 3c:f0:11:be:3d:b3
   380 52.019667802  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.137? Tell 192.168.0.53
   381 52.019697019  IntelCor_be:3d:b3   06:4f:67:15:2a:95   ARP      42 192.168.0.137 is at 3c:f0:11:be:3d:b3
   393 55.091245917  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.73? Tell 192.168.0.53
   396 55.910741499  06:4f:67:15:2a:95   Broadcast           ARP      42 Who has 192.168.0.1? Tell 192.168.0.53
```

```
    Capture Length: 42 bytes (336 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:arp]
    [Coloring Rule Name: ARP]
    [Coloring Rule String: arp]
v Ethernet II, Src: IntelCor_be:3d:b3 (3c:f0:11:be:3d:b3)
  > Destination: ARRISGro_6d:92:01 (70:54:25:6d:92:01)
  > Source: IntelCor_be:3d:b3 (3c:f0:11:be:3d:b3)
    Type: ARP (0x0806)
> Address Resolution Protocol (reply)
```

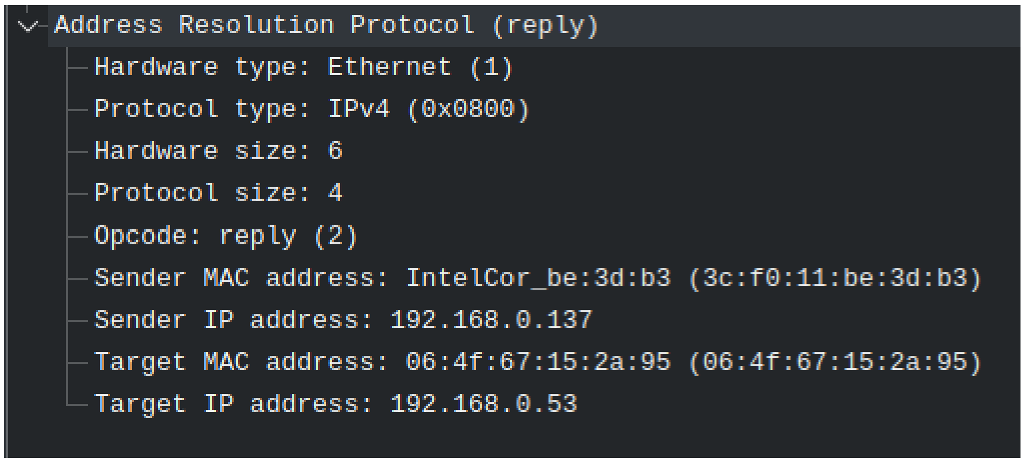
```
0000  70 54 25 6d 92 01 3c f0  11 be 3d b3 08 06 00 01   pT%
0010  08 00 06 04 00 02 3c f0  11 be 3d b3 c0 a8 00 89   ···
0020  70 54 25 6d 92 01 c0 a8  00 01                     pT%
```

wireshark_wlp0s20f3Q9MCG1.pcapng   Packets: 414 · Displayed: 23 (5.6%)   Profile: Default

3.1.2.2 – Inspect trace



```
Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: 06:4f:67:15:2a:95 (06:4f:67:15:2a:95)
    Sender IP address: 192.168.0.53
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.0.137
```



```
Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: IntelCor_be:3d:b3 (3c:f0:11:be:3d:b3)
    Sender IP address: 192.168.0.137
    Target MAC address: 06:4f:67:15:2a:95 (06:4f:67:15:2a:95)
    Target IP address: 192.168.0.53
```

3.1.3 – Details of ARP over Ethernet

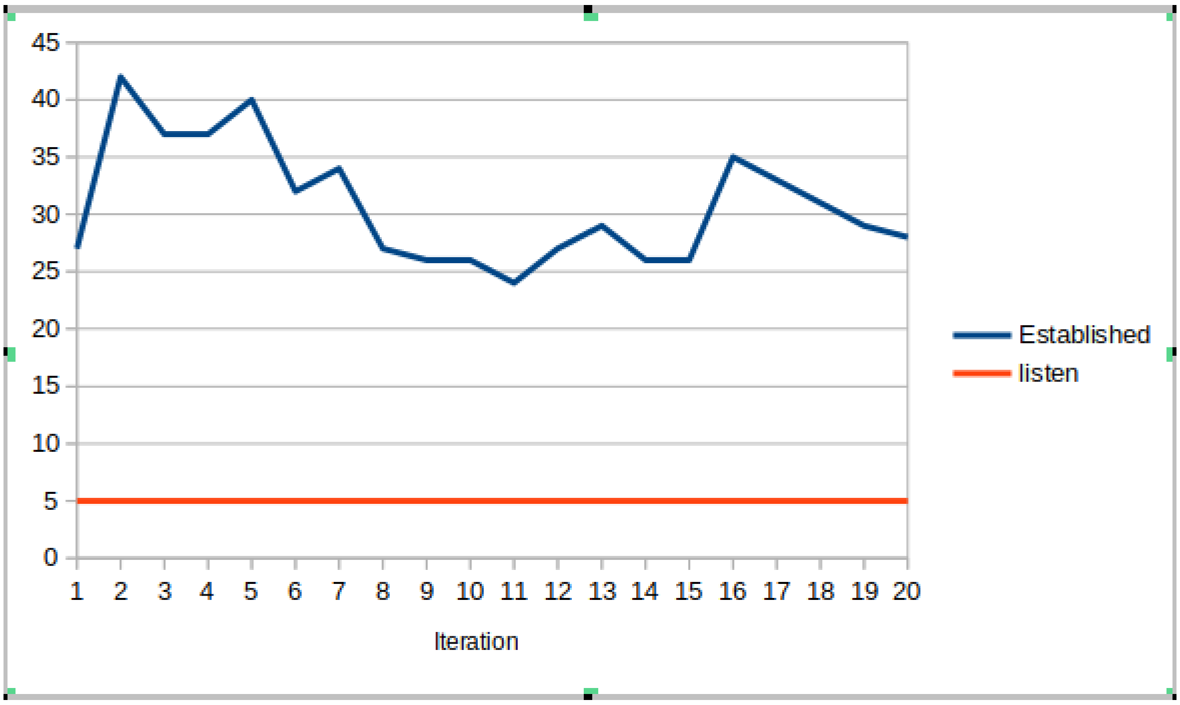
1. Opcode for request = 1 Opcode for reply = 2
2. 28 Bytes for request and reply (source)
3. 00:00:00:00:00:00
4. 0x0806 is the type value used for ARP

3.1.3.2 – Understanding TCP network sockets

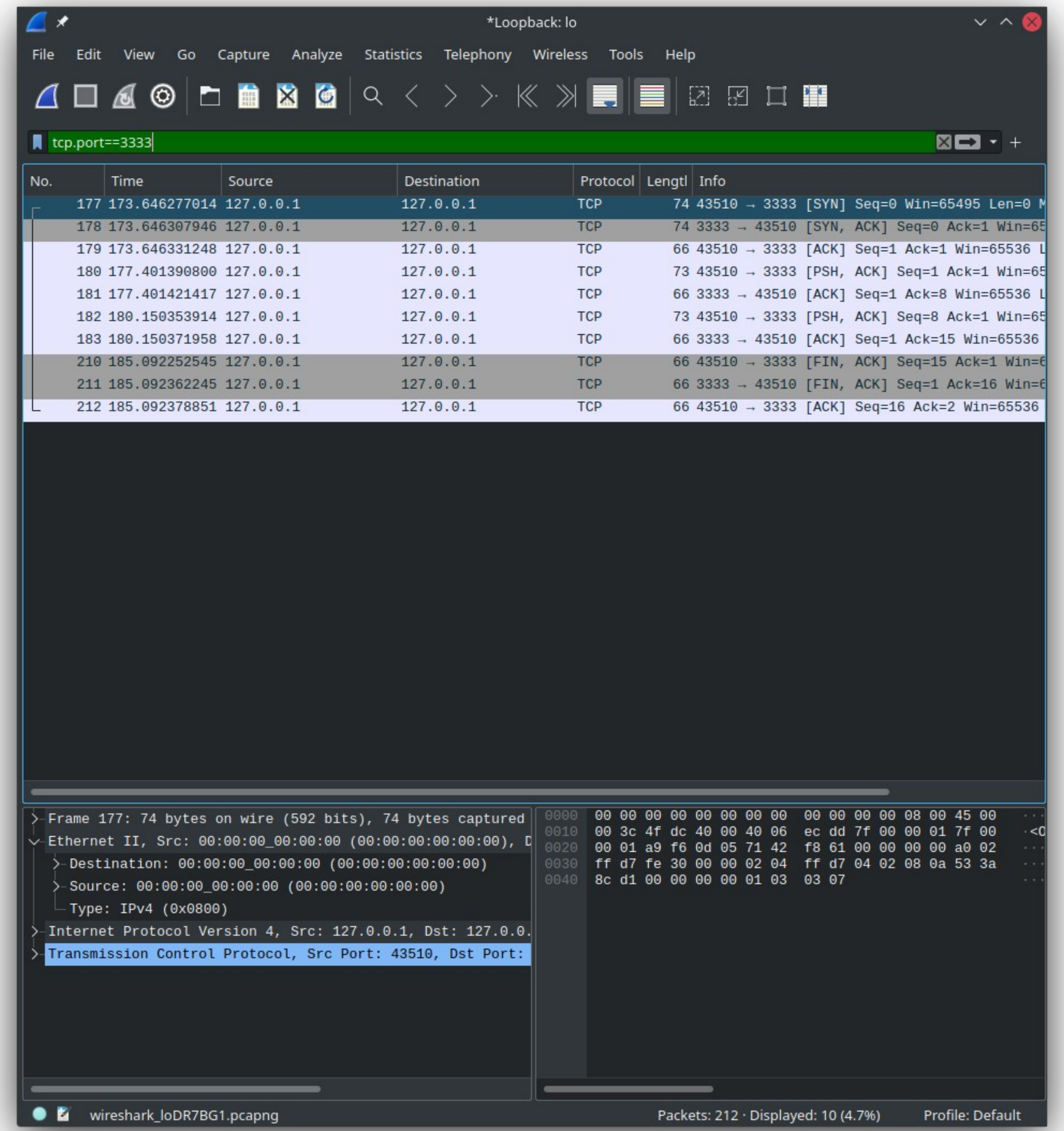
Command:

```
while true; do date >> watch.txt; netstat -a |  awk '{print $6}' | grep -i 'established\|listen' >> watch.txt; sleep 30; done
```

Graph



3.1.3.3.1 – Sniffing TCP traffic


*Loopback: lo
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
tcp.port==3333
```

```
No.  Time          Source     Destination  Protocol Lengtl Info
 177 173.646277014 127.0.0.1  127.0.0.1    TCP        74 43510 → 3333 [SYN] Seq=0 Win=65495 Len=0 M
 178 173.646307946 127.0.0.1  127.0.0.1    TCP        74 3333 → 43510 [SYN, ACK] Seq=0 Ack=1 Win=65
 179 173.646331248 127.0.0.1  127.0.0.1    TCP        66 43510 → 3333 [ACK] Seq=1 Ack=1 Win=65536 L
 180 177.401390800 127.0.0.1  127.0.0.1    TCP        73 43510 → 3333 [PSH, ACK] Seq=1 Ack=1 Win=65
 181 177.401421417 127.0.0.1  127.0.0.1    TCP        66 3333 → 43510 [ACK] Seq=1 Ack=8 Win=65536 L
 182 180.150353914 127.0.0.1  127.0.0.1    TCP        73 43510 → 3333 [PSH, ACK] Seq=8 Ack=1 Win=65
 183 180.150371958 127.0.0.1  127.0.0.1    TCP        66 3333 → 43510 [ACK] Seq=1 Ack=15 Win=65536
 210 185.092252545 127.0.0.1  127.0.0.1    TCP        66 43510 → 3333 [FIN, ACK] Seq=15 Ack=1 Win=6
 211 185.092362245 127.0.0.1  127.0.0.1    TCP        66 3333 → 43510 [FIN, ACK] Seq=1 Ack=16 Win=6
 212 185.092378851 127.0.0.1  127.0.0.1    TCP        66 43510 → 3333 [ACK] Seq=16 Ack=2 Win=65536
```

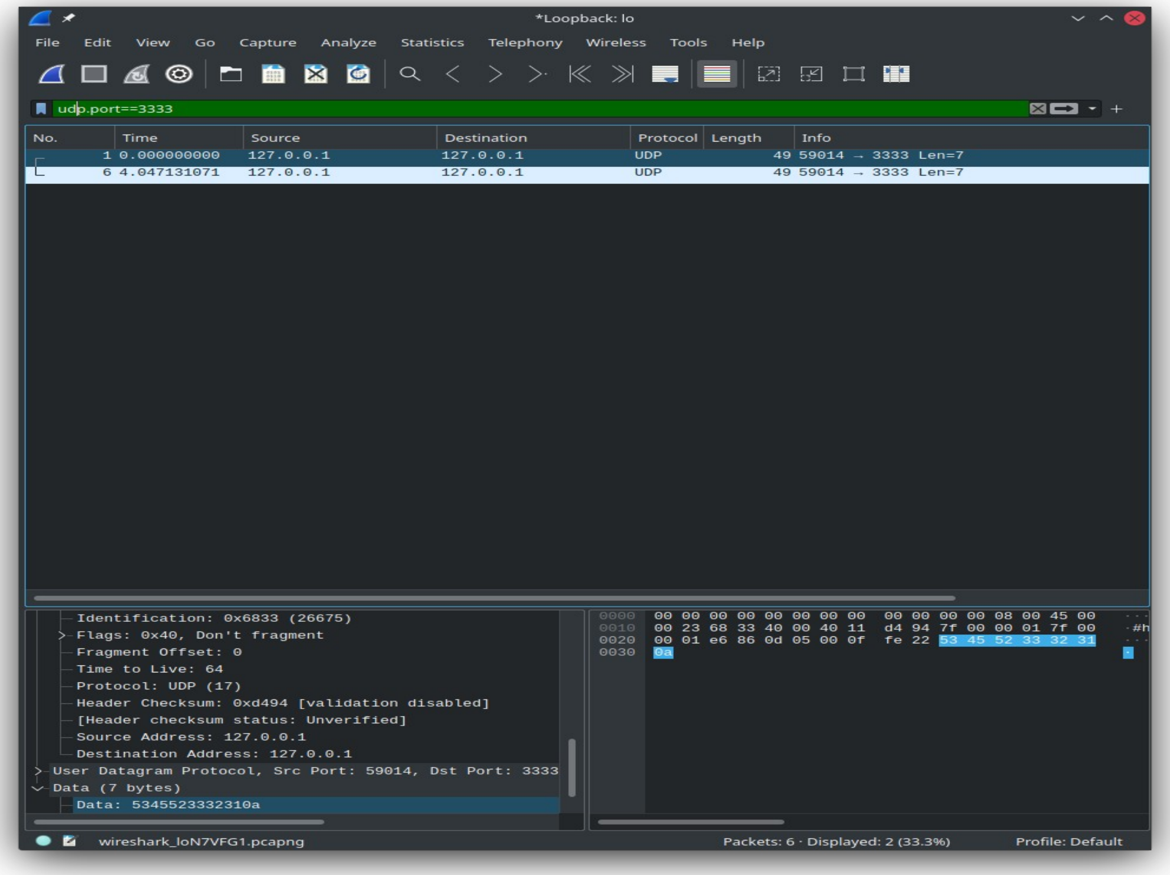
```
> Frame 177: 74 bytes on wire (592 bits), 74 bytes captured
v Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), D
  > Destination: 00:00:00_00:00:00 (00:00:00:00:00:00)
  > Source: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Type: IPv4 (0x0800)
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.
> Transmission Control Protocol, Src Port: 43510, Dst Port:
```

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ···
0010  00 3c 4f dc 40 00 40 06  ec dd 7f 00 00 01 7f 00   ·<C
0020  00 01 a9 f6 0d 05 71 42  f8 61 00 00 00 00 a0 02   ···
0030  ff d7 fe 30 00 00 02 04  ff d7 04 02 08 0a 53 3a   ···
0040  8c d1 00 00 00 00 01 03  03 07                     ···
```

```
wireshark_loDR7BG1.pcapng        Packets: 212 · Displayed: 10 (4.7%)        Profile: Default
```

a) nc -k -l 3333 runs the netcat command, the -l flag tells it to listen on port 3333, and the -k flag tells it to continue listening for connections after one is completed. nc 127.0.0.1 3333 tells netcat to connect to 127.0.0.1 (the loopback ip) on port 3333.
b) It took 2 frames to capture the 2 lines.
c) It took 2 packets to capture the 2 lines.
d) The whole process took 10 packets.
e) 690 Bytes
f) 98% was overhead

3.1.3.3.1 – Sniffing UDP Traffic


*Loopback: lo
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
udp.port==3333
```

```
No.    Time          Source      Destination  Protocol  Length  Info
     1 0.000000000   127.0.0.1   127.0.0.1    UDP           49  59014 → 3333 Len=7
     6 4.047131071   127.0.0.1   127.0.0.1    UDP           49  59014 → 3333 Len=7
```

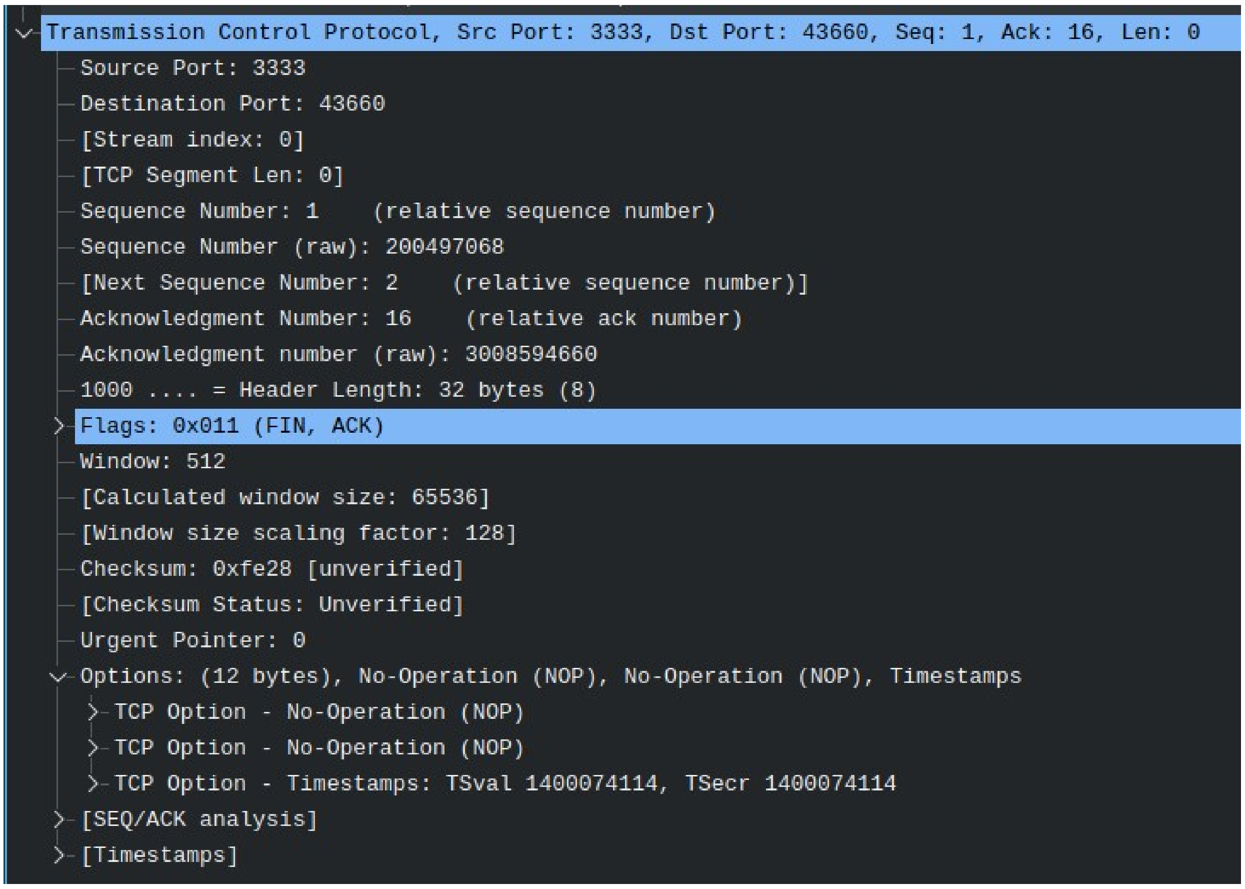
```
    Identification: 0x6833 (26675)
  > Flags: 0x40, Don't fragment
    Fragment Offset: 0
    Time to Live: 64
    Protocol: UDP (17)
    Header Checksum: 0xd494 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 127.0.0.1
    Destination Address: 127.0.0.1
> User Datagram Protocol, Src Port: 59014, Dst Port: 3333
v Data (7 bytes)
    Data: 5345523332310a
```

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00
0010  00 23 68 33 40 00 40 11  d4 94 7f 00 00 01 7f 00   ·#h
0020  00 01 e6 86 0d 05 00 0f  fe 22 53 45 52 33 32 31
0030  0a
```

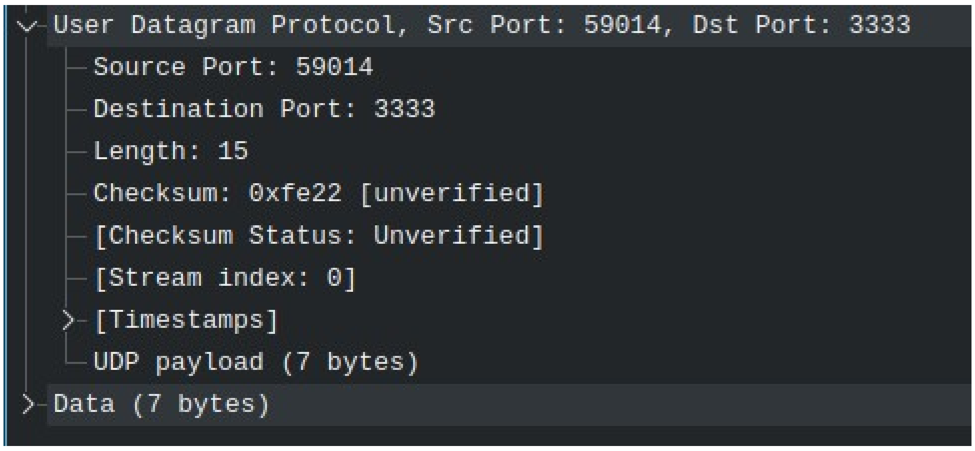
```
wireshark_loN7VFG1.pcapng    Packets: 6 · Displayed: 2 (33.3%)    Profile: Default
```

a) nc -k -l -u 3333 runs the netcat command, the -l flag tells it to listen on port 3333, and the -k flag tells it to continue listening for connections after one is completed, the -u flag tell it to use UDP instead of TCP. nc 127.0.0.1 3333 tells netcat to connect to 127.0.0.1 (the loopback ip) on port 3333, the -u flag tells it to use UDP instead of TCP.
b) 2 frames
c) 2 packets
d) 2 packets
e) 98 bytes total, 86% was overhead
f) UDP uses less overhead because it does not deal with establishing a connection, acknowledgment, sequence, replies, or other QoS things.

Relevant parts of packet trace for part f



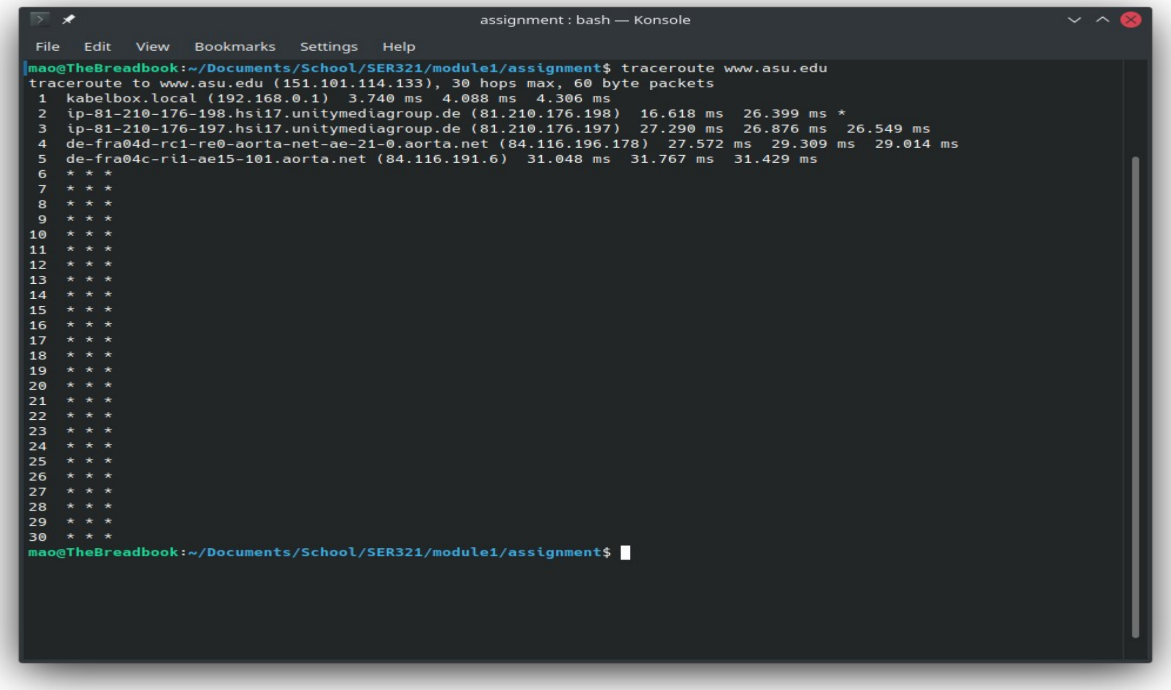
```
Transmission Control Protocol, Src Port: 3333, Dst Port: 43660, Seq: 1, Ack: 16, Len: 0
    Source Port: 3333
    Destination Port: 43660
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 200497068
    [Next Sequence Number: 2    (relative sequence number)]
    Acknowledgment Number: 16    (relative ack number)
    Acknowledgment number (raw): 3008594660
    1000 .... = Header Length: 32 bytes (8)
    Flags: 0x011 (FIN, ACK)
    Window: 512
    [Calculated window size: 65536]
    [Window size scaling factor: 128]
    Checksum: 0xfe28 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
    Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
        TCP Option - No-Operation (NOP)
        TCP Option - No-Operation (NOP)
        TCP Option - Timestamps: TSval 1400074114, TSecr 1400074114
    [SEQ/ACK analysis]
    [Timestamps]
```



```
User Datagram Protocol, Src Port: 59014, Dst Port: 3333
    Source Port: 59014
    Destination Port: 3333
    Length: 15
    Checksum: 0xfe22 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 0]
    [Timestamps]
    UDP payload (7 bytes)
Data (7 bytes)
```

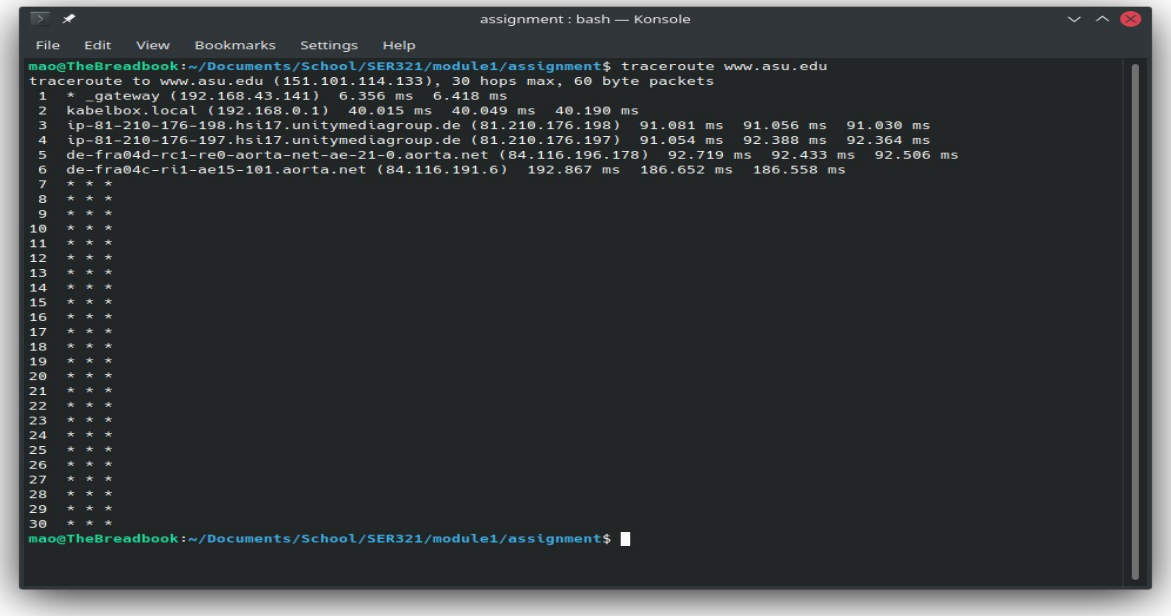
3.1.3.4 – IP routing

Home Network


assignment : bash — Konsole
File Edit View Bookmarks Settings Help

```
mao@TheBreadbook:~/Documents/School/SER321/module1/assignment$ traceroute www.asu.edu
traceroute to www.asu.edu (151.101.114.133), 30 hops max, 60 byte packets
 1  kabelbox.local (192.168.0.1)  3.740 ms  4.088 ms  4.306 ms
 2  ip-81-210-176-198.hsi17.unitymediagroup.de (81.210.176.198)  16.618 ms  26.399 ms *
 3  ip-81-210-176-197.hsi17.unitymediagroup.de (81.210.176.197)  27.290 ms  26.876 ms  26.549 ms
 4  de-fra04d-rc1-re0-aorta-net-ae-21-0.aorta.net (84.116.196.178)  27.572 ms  29.309 ms  29.014 ms
 5  de-fra04c-ri1-ae15-101.aorta.net (84.116.191.6)  31.048 ms  31.767 ms  31.429 ms
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
mao@TheBreadbook:~/Documents/School/SER321/module1/assignment$
```

Mobile Hotspot

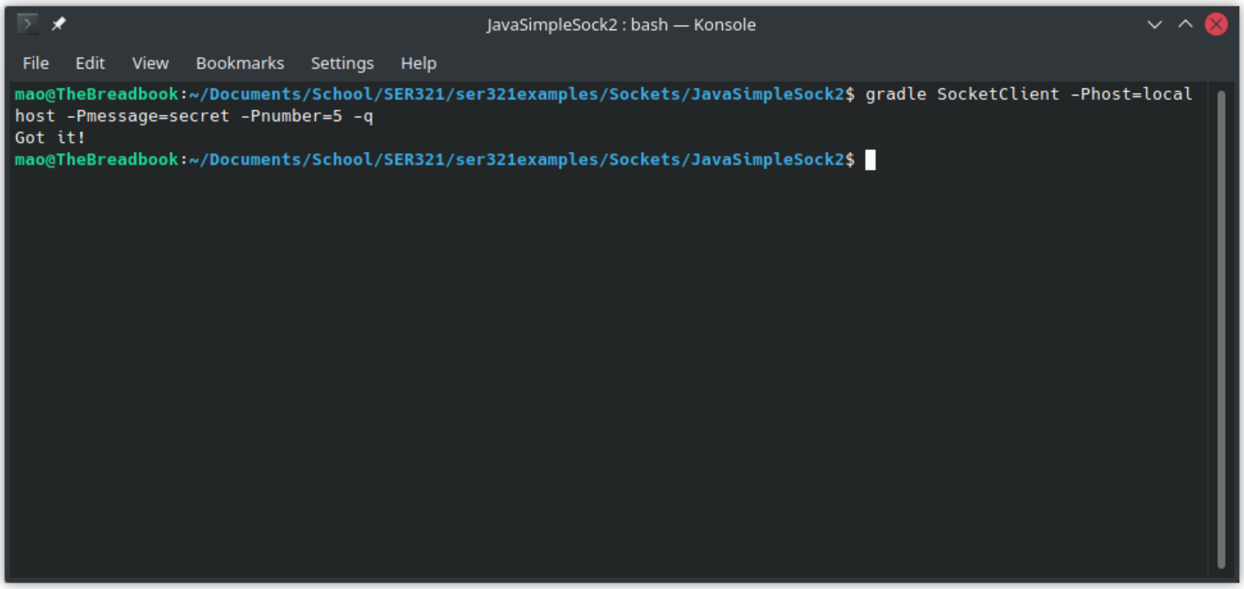

assignment : bash — Konsole
File Edit View Bookmarks Settings Help

```
mao@TheBreadbook:~/Documents/School/SER321/module1/assignment$ traceroute www.asu.edu
traceroute to www.asu.edu (151.101.114.133), 30 hops max, 60 byte packets
 1  * _gateway (192.168.43.141)  6.356 ms  6.418 ms
 2  kabelbox.local (192.168.0.1)  40.015 ms  40.049 ms  40.190 ms
 3  ip-81-210-176-198.hsi17.unitymediagroup.de (81.210.176.198)  91.081 ms  91.056 ms  91.030 ms
 4  ip-81-210-176-197.hsi17.unitymediagroup.de (81.210.176.197)  91.054 ms  92.388 ms  92.364 ms
 5  de-fra04d-rc1-re0-aorta-net-ae-21-0.aorta.net (84.116.196.178)  92.719 ms  92.433 ms  92.506 ms
 6  de-fra04c-ri1-ae15-101.aorta.net (84.116.191.6)  192.867 ms  186.652 ms  186.558 ms
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
mao@TheBreadbook:~/Documents/School/SER321/module1/assignment$
```

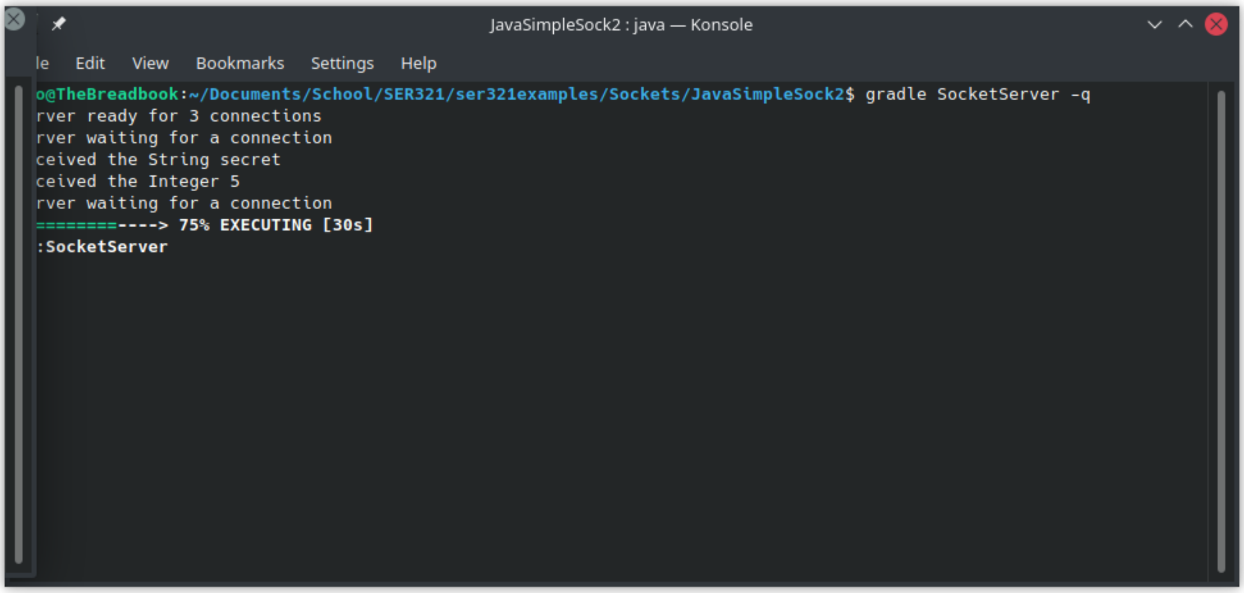
- My home network was faster
- My home network has one fewer hop

3.1.3.5.1 – Running things locally

https://youtu.be/zKYFLqk4ibU

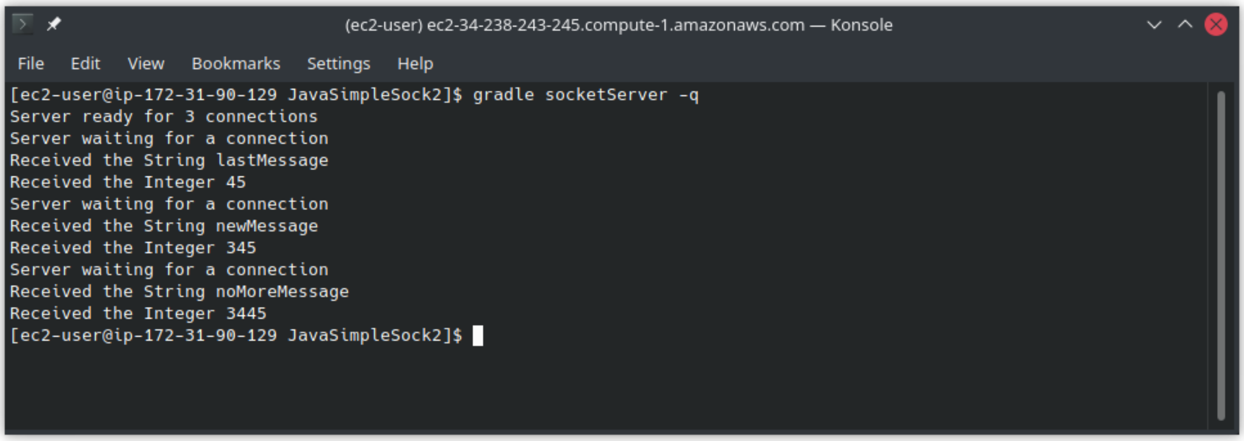

JavaSimpleSock2 : bash — Konsole
File Edit View Bookmarks Settings Help

```
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Sockets/JavaSimpleSock2$ gradle SocketClient -Phost=local
host -Pmessage=secret -Pnumber=5 -q
Got it!
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Sockets/JavaSimpleSock2$
```

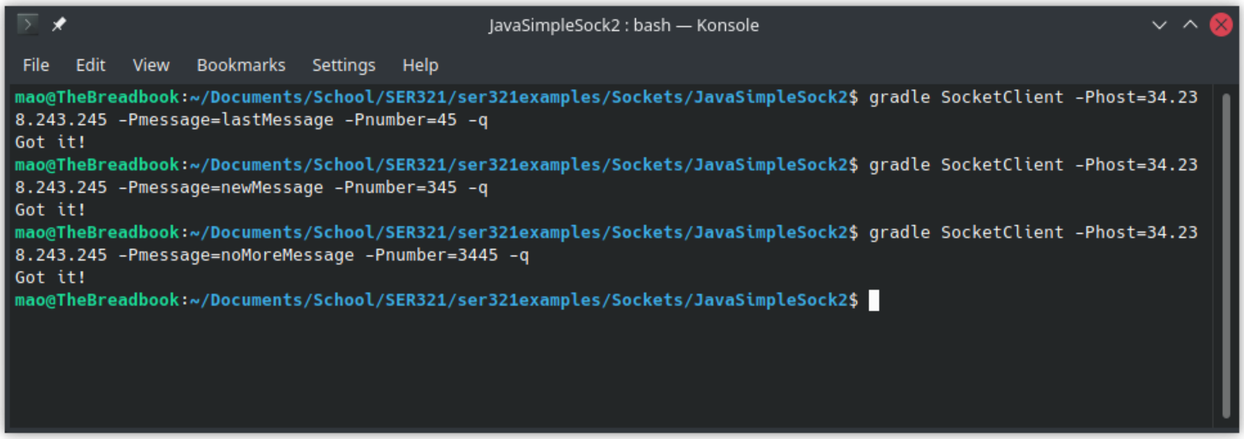

JavaSimpleSock2 : java — Konsole
le Edit View Bookmarks Settings Help

```
o@TheBreadbook:~/Documents/School/SER321/ser321examples/Sockets/JavaSimpleSock2$ gradle SocketServer -q
rver ready for 3 connections
rver waiting for a connection
ceived the String secret
ceived the Integer 5
rver waiting for a connection
========----> 75% EXECUTING [30s]
:SocketServer
```

3.1.3.5.2 – Server on AWS


(ec2-user) ec2-34-238-243-245.compute-1.amazonaws.com — Konsole
File Edit View Bookmarks Settings Help

```
[ec2-user@ip-172-31-90-129 JavaSimpleSock2]$ gradle socketServer -q
Server ready for 3 connections
Server waiting for a connection
Received the String lastMessage
Received the Integer 45
Server waiting for a connection
Received the String newMessage
Received the Integer 345
Server waiting for a connection
Received the String noMoreMessage
Received the Integer 3445
[ec2-user@ip-172-31-90-129 JavaSimpleSock2]$ 
```


JavaSimpleSock2 : bash — Konsole
File Edit View Bookmarks Settings Help

```
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Sockets/JavaSimpleSock2$ gradle SocketClient -Phost=34.23
8.243.245 -Pmessage=lastMessage -Pnumber=45 -q
Got it!
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Sockets/JavaSimpleSock2$ gradle SocketClient -Phost=34.23
8.243.245 -Pmessage=newMessage -Pnumber=345 -q
Got it!
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Sockets/JavaSimpleSock2$ gradle SocketClient -Phost=34.23
8.243.245 -Pmessage=noMoreMessage -Pnumber=3445 -q
Got it!
mao@TheBreadbook:~/Documents/School/SER321/ser321examples/Sockets/JavaSimpleSock2$ 
```

Compared to running locally, I had to change Wireshark to sniff on my WiFi connection instead of the loopback. In the Gradle calls I had to change the -Phost argument to be the IP address of the AWS server instead of localhost

3.1.3.5.3/4 – Client on AWS

No, running the server locally and the client on AWS does not work. This does not work because of the way local networks and routers interact. The AWS instance communicates with my router, the router then broadcasts the traffic intended for my laptop over the local network, my laptop recognizes it is the destination for the frame and picks it up. The IP of my laptop (192.168.x.x) is only for the local network and not available to the AWS instance. One way to reach my local network from the outside would be to set up port forwarding on my router.