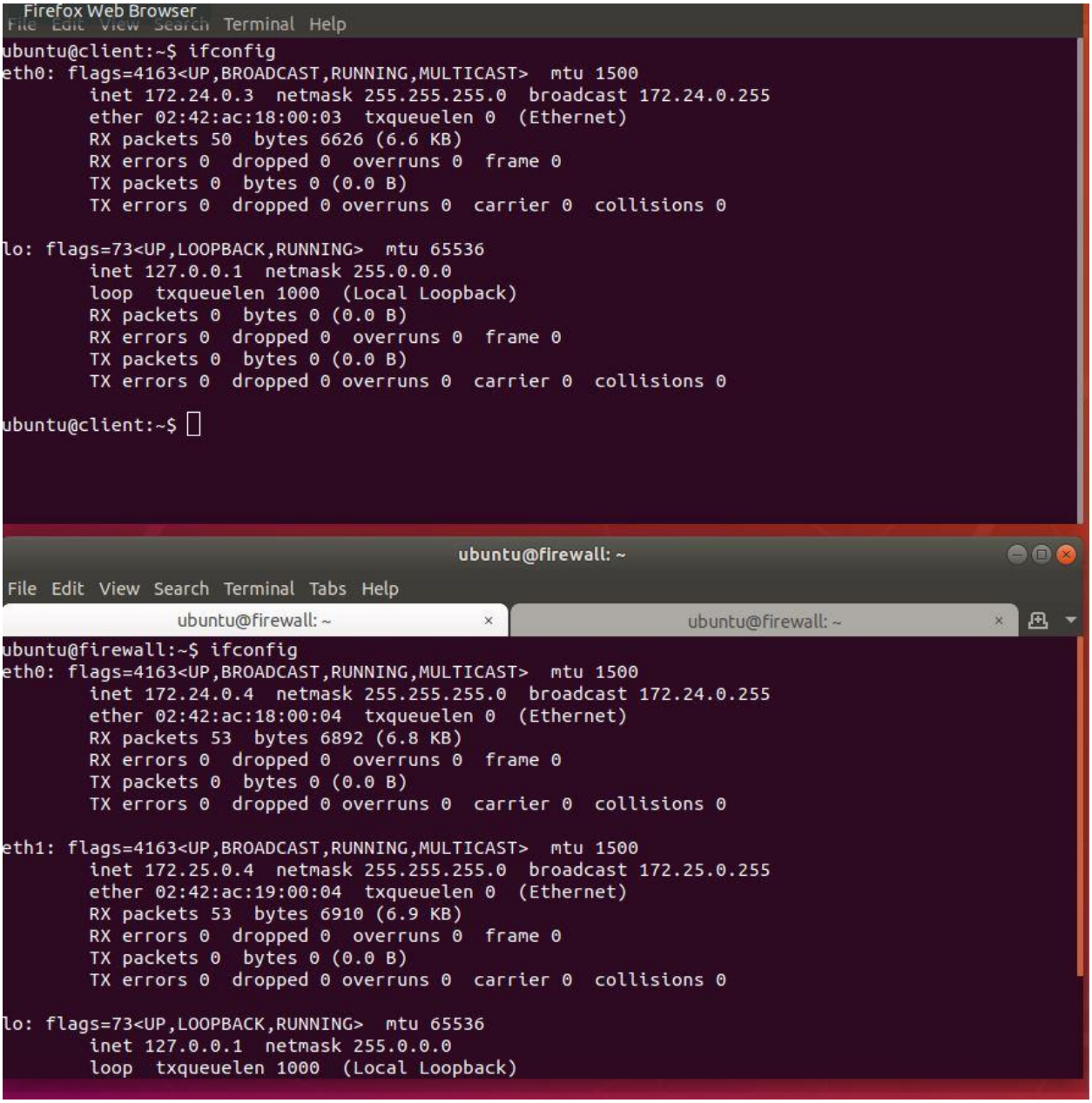
Assignment 4 – Linux Firewall

Task 1. Find IP addresses

a) Find the IP address of the client and the firewall.
b) Show the addresses in screenshots.

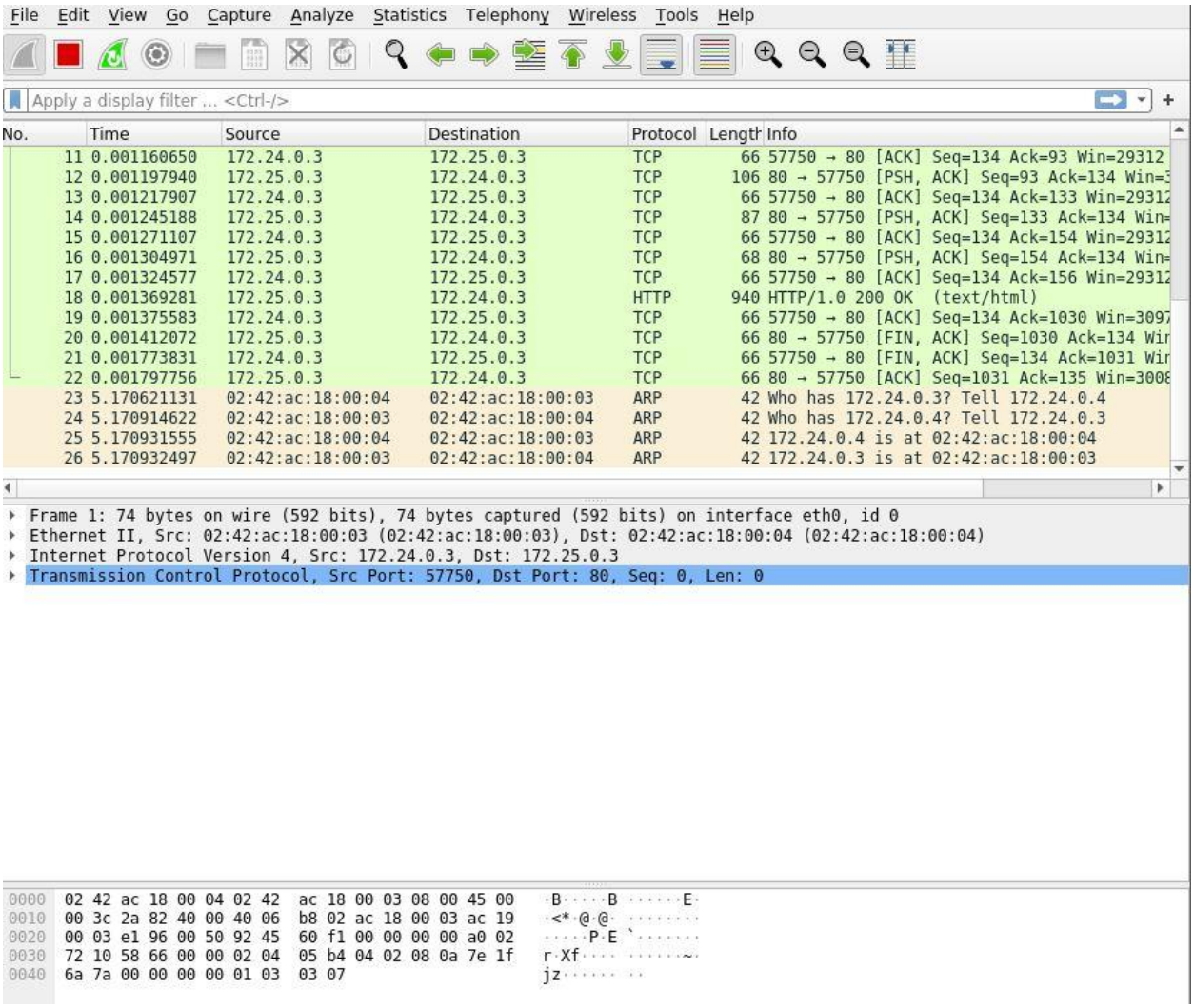


Task 2. Nmap scan

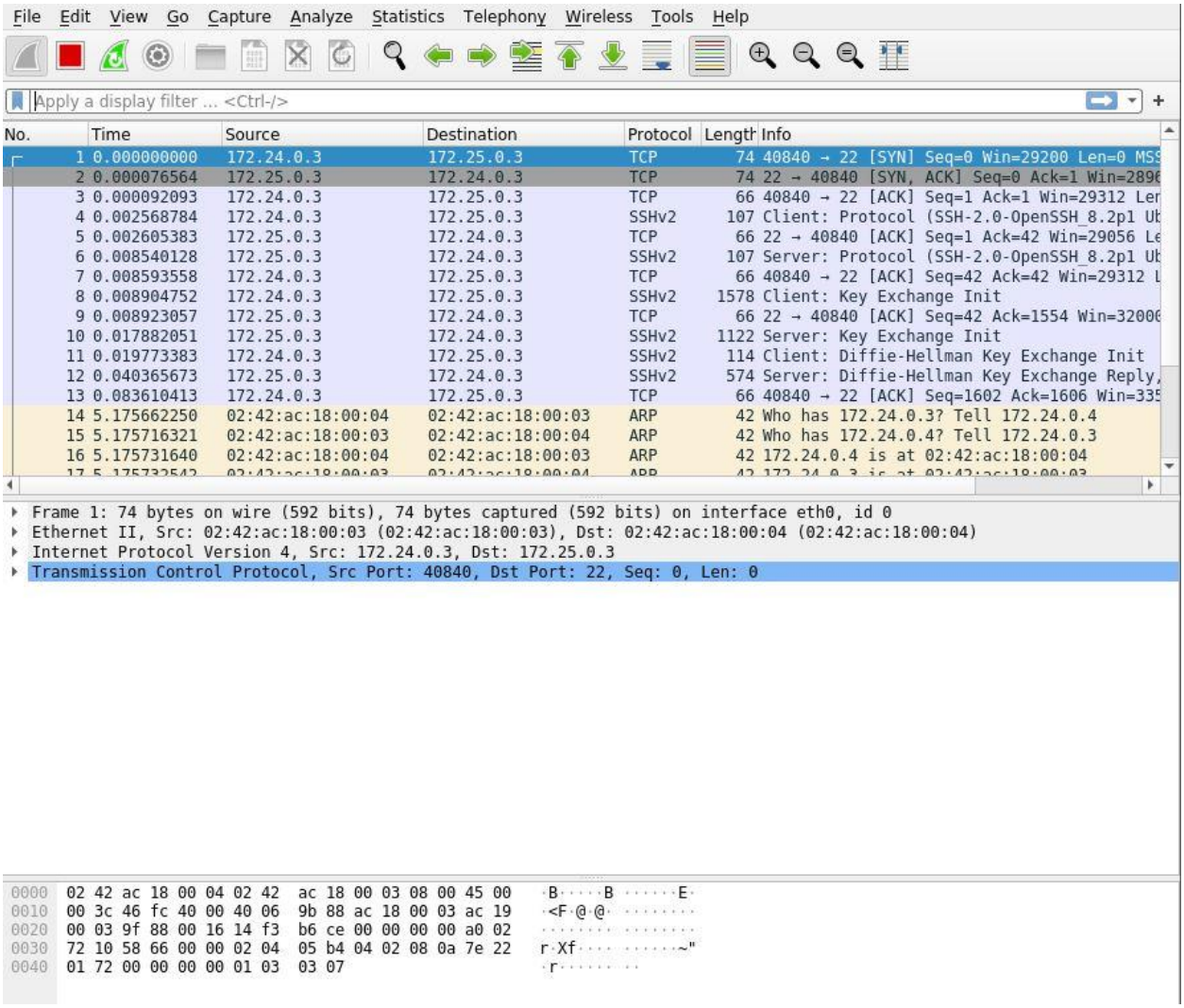
a) Perform a nmap scan on the client for open ports on the server. Show the output in a screenshot.

```
ubuntu@client:~$ nmap server
Starting Nmap 7.80 ( https://nmap.org ) at 2023-10-26 19:40 UTC
Nmap scan report for server (172.25.0.3)
Host is up (0.00021s latency).
Not shown: 997 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
23/tcp open  telnet
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.07 seconds
ubuntu@client:~$ wget server
--2023-10-26 19:40:40--  http://server/
Resolving server (server)... 172.25.0.3
Connecting to server (server)|172.25.0.3|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 874 [text/html]
Saving to: 'index.html.1'

index.html.1             100%[==============================>]     874  --.-KB/s    in 0s

2023-10-26 19:40:40 (9.96 MB/s) - 'index.html.1' saved [874/874]
```

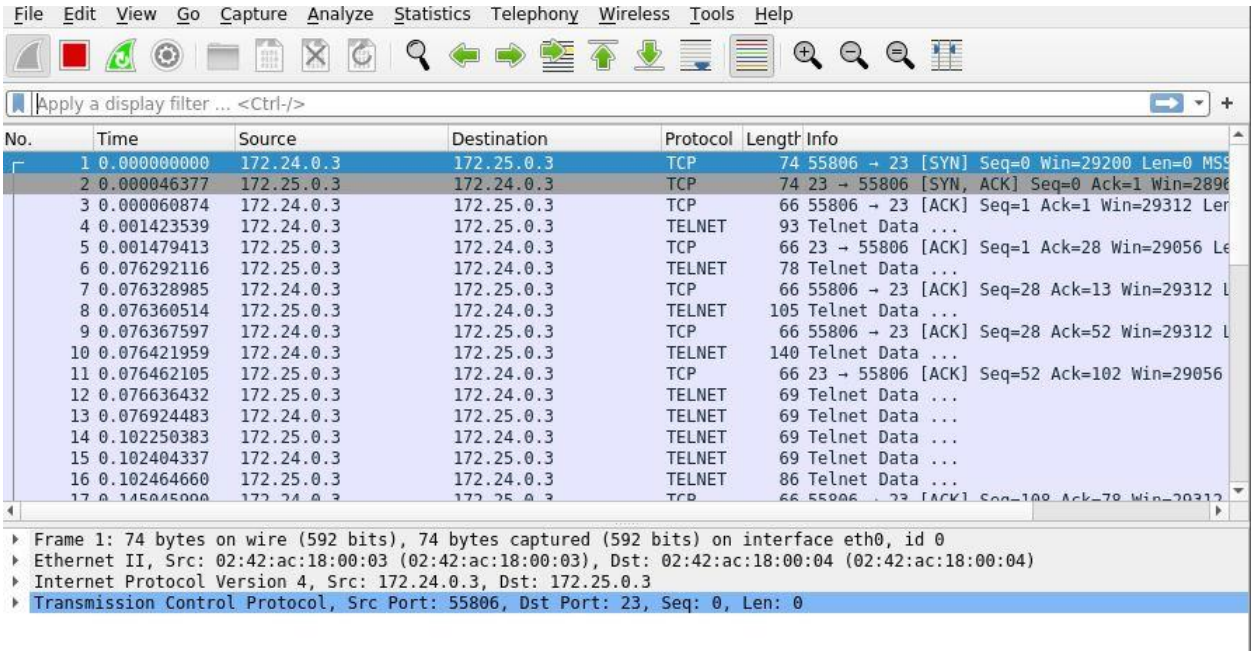
b) Run *wget* and report captured packets on Wireshark in a screenshot. To capture packets for a new command, you need to stop/start capturing without exiting Wireshark.



c) Run *ssh* and report captured packets on Wireshark in a screenshot.



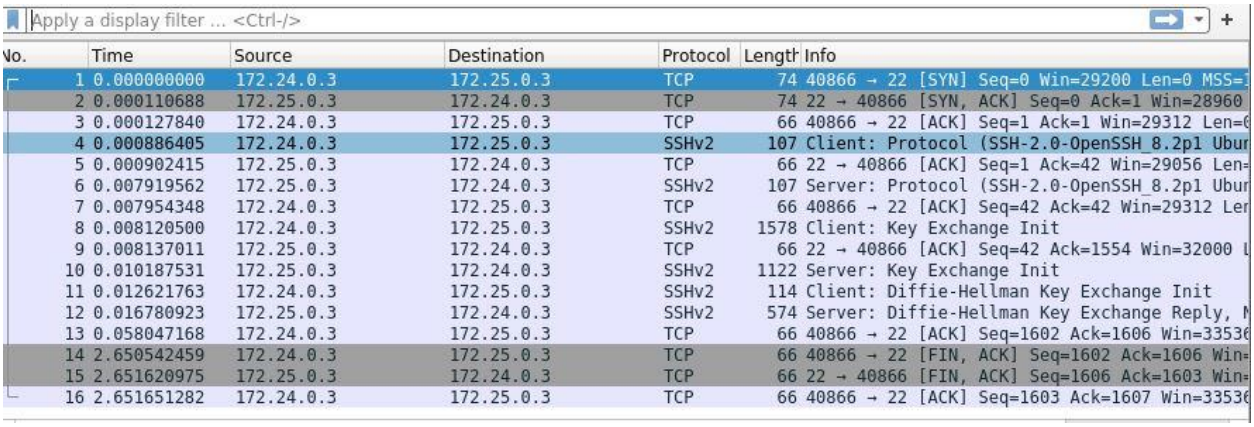
d) Run *telnet* and report captured packets on Wireshark in a screenshot.



Task 3. Use iptables to limit traffic to the server

a) Show that ssh traffic is allowed. On the client, run ssh while capturing traffic on the firewall. Report these two activities in two screenshots. Explain how you know ssh traffic is allowed.

Wireshark view:



Client View:

Iptables view on the firewall after adding our new rules:

```
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
NFLOG      all  --  anywhere             anywhere             limit: avg 2/min burst 5 nflog-pref
ix  "IPTABLES DROPPED"

Chain OUTPUT (policy DROP)
target     prot opt source               destination
```

Explanation: **We know that ssh traffic is allowed because the rule to allow it is enabled in our iptables configuration as well as the three-way handshake between the client and host being completed.**

b) Show that HTTP traffic is allowed. Report the same as you did for ssh traffic.

Wireshark View:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.24.0.3	172.25.0.3	TCP	74	57974 → 80 [SYN] Seq=0 Win=29200 Len=0 M
2	0.000047720	172.25.0.3	172.24.0.3	TCP	74	80 → 57974 [SYN, ACK] Seq=0 Ack=1 Win=28
3	0.000076895	172.24.0.3	172.25.0.3	TCP	66	57974 → 80 [ACK] Seq=1 Ack=1 Win=29312 L
4	0.000244149	172.24.0.3	172.25.0.3	HTTP	199	GET / HTTP/1.1
5	0.000286719	172.25.0.3	172.24.0.3	TCP	66	80 → 57974 [ACK] Seq=1 Ack=134 Win=30080
6	0.001135403	172.25.0.3	172.24.0.3	TCP	83	80 → 57974 [PSH, ACK] Seq=1 Ack=134 Win=
7	0.001162423	172.24.0.3	172.25.0.3	TCP	66	57974 → 80 [ACK] Seq=134 Ack=18 Win=2931
8	0.001195555	172.25.0.3	172.24.0.3	TCP	104	80 → 57974 [PSH, ACK] Seq=18 Ack=134 Win
9	0.001216264	172.24.0.3	172.25.0.3	TCP	66	57974 → 80 [ACK] Seq=134 Ack=56 Win=2931
10	0.001252011	172.25.0.3	172.24.0.3	TCP	103	80 → 57974 [PSH, ACK] Seq=56 Ack=134 Win
11	0.001271819	172.24.0.3	172.25.0.3	TCP	66	57974 → 80 [ACK] Seq=134 Ack=93 Win=2931
12	0.001299591	172.25.0.3	172.24.0.3	TCP	106	80 → 57974 [PSH, ACK] Seq=93 Ack=134 Win
13	0.001318707	172.24.0.3	172.25.0.3	TCP	66	57974 → 80 [ACK] Seq=134 Ack=133 Win=293
14	0.001343944	172.25.0.3	172.24.0.3	TCP	87	80 → 57974 [PSH, ACK] Seq=133 Ack=134 Wi
15	0.001363280	172.24.0.3	172.25.0.3	TCP	66	57974 → 80 [ACK] Seq=134 Ack=154 Win=293
16	0.001386775	172.25.0.3	172.24.0.3	TCP	68	80 → 57974 [PSH, ACK] Seq=154 Ack=134 Wi
17	0.001405610	172.24.0.3	172.25.0.3	TCP	66	57974 → 80 [ACK] Seq=134 Ack=156 Win=293

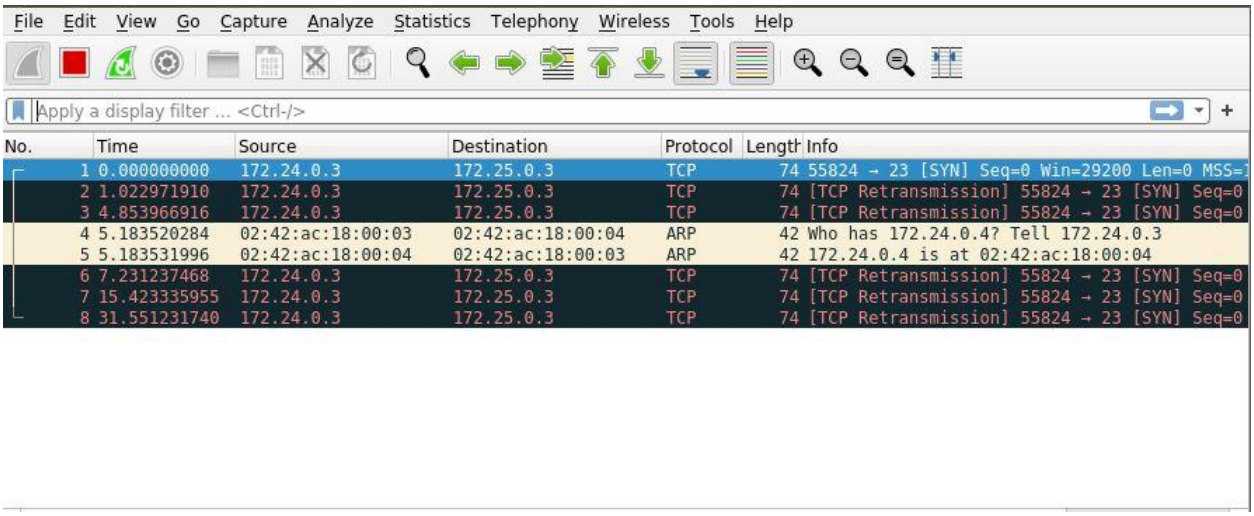
Client View:

```
ubuntu@client:~$ wget server
--2023-10-26 21:05:43--  http://server/
Resolving server (server)... 172.25.0.3
Connecting to server (server)|172.25.0.3|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 874 [text/html]
Saving to: 'index.html.8'

index.html.8             100%[=================================>]     874  --.-KB/s    in 0s

2023-10-26 21:05:43 (262 MB/s) - 'index.html.8' saved [874/874]

ubuntu@client:~$
```

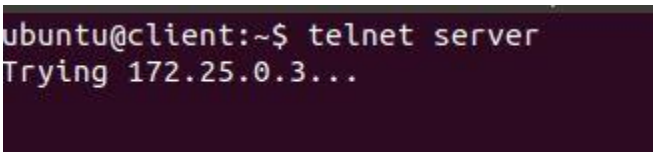
Explanation: **I allowed a rule in the file to accept traffic on port 80 thus allowing http connection to the server. The packet capture proves this because a three-way handshake is established and packets are transferred**

c) Show that telnet traffic is blocked. Report the same as you did for ssh traffic.

Wireshark View:



Client View:



Explanation: **Telnet traffic is blocked because of the rules we've established in our iptables configuration. Only HTTP and SSH traffic is allowed. The packet capture proves this because it is unable to establish a three-way handshake. Syn packets are sent with no ack response.**

d) At the end, perform a nmap scan on the client for open ports on the server. Show the output in a screenshot.

```
ubuntu@client:~$ nmap server
Starting Nmap 7.80 ( https://nmap.org ) at 2023-10-26 21:09 UTC
Nmap scan report for server (172.25.0.3)
Host is up (0.00027s latency).
Not shown: 998 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 6.44 seconds
ubuntu@client:~$
```

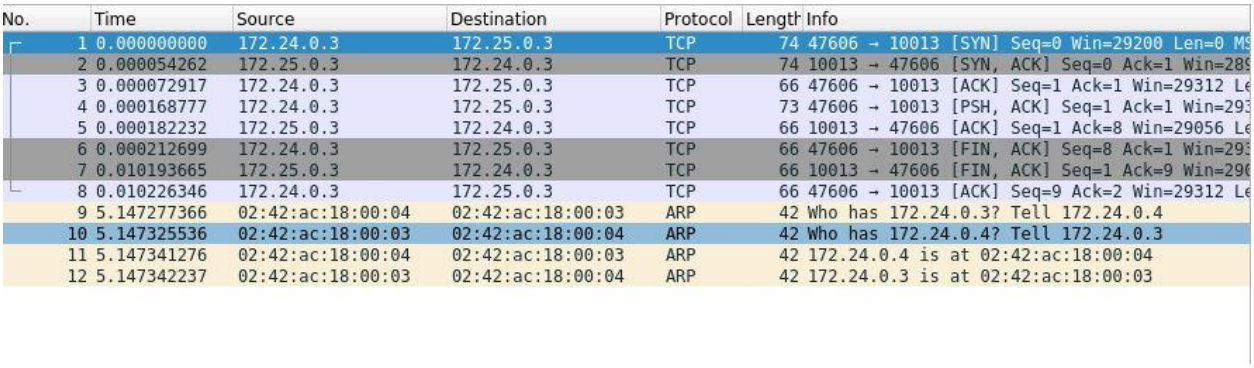
Task 4. Open a new service port

a) Show that wizbang traffic is allowed. On the client, run wizbang while capturing traffic on the firewall. Report these two activities in two screenshots. Explain how you know wizbang traffic is allowed.

New iptables config:

```
ubuntu@firewall:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:10013
NFLOG      all  --  anywhere             anywhere             limit: avg 2/min burst 5 nflog-pref
ix  "IPTABLES DROPPED"

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ubuntu@firewall:~$
```

Wireshark View:



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.24.0.3	172.25.0.3	TCP	74	47606 → 10013 [SYN] Seq=0 Win=29200 Len=0 MS
2	0.000054262	172.25.0.3	172.24.0.3	TCP	74	10013 → 47606 [SYN, ACK] Seq=0 Ack=1 Win=28
3	0.000072917	172.24.0.3	172.25.0.3	TCP	66	47606 → 10013 [ACK] Seq=1 Ack=1 Win=29312 Le
4	0.000168777	172.24.0.3	172.25.0.3	TCP	73	47606 → 10013 [PSH, ACK] Seq=1 Ack=1 Win=293
5	0.000182232	172.25.0.3	172.24.0.3	TCP	66	10013 → 47606 [ACK] Seq=1 Ack=8 Win=29056 Le
6	0.000212699	172.24.0.3	172.25.0.3	TCP	66	47606 → 10013 [FIN, ACK] Seq=8 Ack=1 Win=293
7	0.010193665	172.25.0.3	172.24.0.3	TCP	66	10013 → 47606 [FIN, ACK] Seq=1 Ack=9 Win=290
8	0.010226346	172.24.0.3	172.25.0.3	TCP	66	47606 → 10013 [ACK] Seq=9 Ack=2 Win=29312 Le
9	5.147277366	02:42:ac:18:00:04	02:42:ac:18:00:03	ARP	42	Who has 172.24.0.3? Tell 172.24.0.4
10	5.147325536	02:42:ac:18:00:03	02:42:ac:18:00:04	ARP	42	Who has 172.24.0.4? Tell 172.24.0.3
11	5.147341276	02:42:ac:18:00:04	02:42:ac:18:00:03	ARP	42	172.24.0.4 is at 02:42:ac:18:00:04
12	5.147342237	02:42:ac:18:00:03	02:42:ac:18:00:04	ARP	42	172.24.0.3 is at 02:42:ac:18:00:03

Client View:

```
ubuntu@client:~$ sudo ./wizbang Hello!
Sending instruction Hello!
bye
ubuntu@client:~$
```

Explanation: **I allowed traffic on port 10013 which is the port wizbang was using to connect to our server. We can see that it works because the three-way handshake is completed.**

b) At the end, perform a nmap scan on the client for open ports on the server. Show the output in a screenshot.

```
ubuntu@client:~$ nmap server
Starting Nmap 7.80 ( https://nmap.org ) at 2023-10-26 21:37 UTC
Nmap scan report for server (172.25.0.3)
Host is up (0.00026s latency).
Not shown: 998 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 4.84 seconds
ubuntu@client:~$
```