

CNET\_LAB\_Assignment#1

22i-1552\_Syed Arham Ahmed

First, I wrote code to send messages to the server until the client enters “end”. Until then the server prints whatever the client enters. The changes that I made in my code so that I could use another computer act as a client and send messages to the server is as below:

A computer screen shot of a program code

Description automatically generated

The above code is my server side code and In line 21 I have changed the code from “server\_address.sin\_addr.s\_addr = INADDR\_ANY;” to “server\_address.sin\_addr.s\_addr = inet\_addr("192.168.1.10");” in the brackets I have specified the IP address of my server which is a Kali VM. I also included the arpa/inet.h header file in the code.

That’s the only change I made in the server side, now let’s take a look at the client side.

  
The above code is my client side code that is being run on an Ubuntu vm on a separate computer.  
Again over here I have also specified the server IP in line 21, changed it from “server\_address.sin\_addr.s\_addr = INADDR\_ANY;” to the line you can see in the screenshot above.

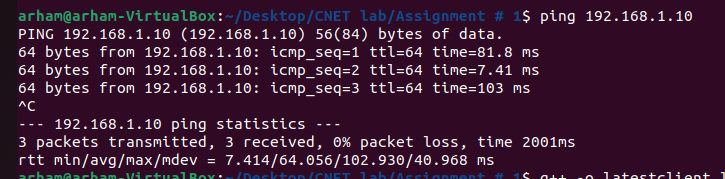
Now these are the changes that I made to the code itself, but this was not it, I had to do further things so that my computers could communicate as currently they were un-able to.

First, I changed the network settings of my Kali VM and changed it from NAT to Bridged Adapter with the intel R Wi-Fi name as shown in the screenshot below. I did the same for the Ubuntu VM on my other computer. These changes enabled both of my VM’s to be on the same network, **as a bridged adapter acts as a switch between two computers.**

A screenshot of a computer

Description automatically generated

Now that I had configured everything, I pinged my server from the client side just to make sure they were able to connect.



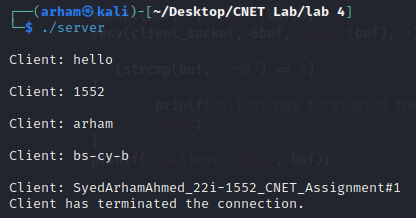
Now that everything was working fine, I first compiled my server code and ran it ie: KALI VM, then on my other computer I compiled my client code and ran it ie: Ubuntu VM, I then gave input to server and server side displayed my input which shows that I achieved my goal.

**Client-Side Terminal:**

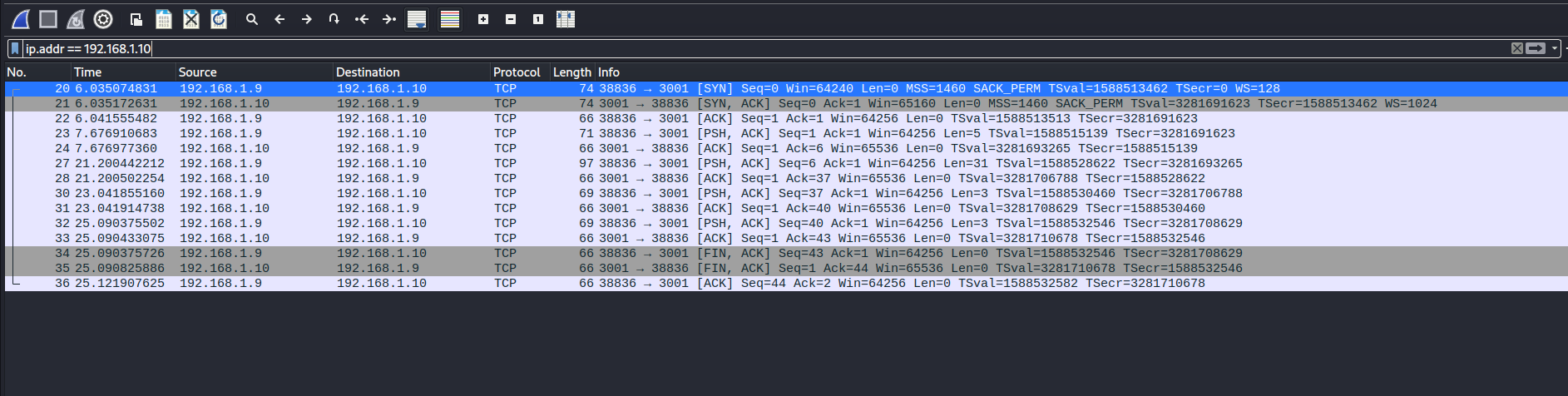
A computer screen shot of a program

Description automatically generated

**Server-Side Terminal:**

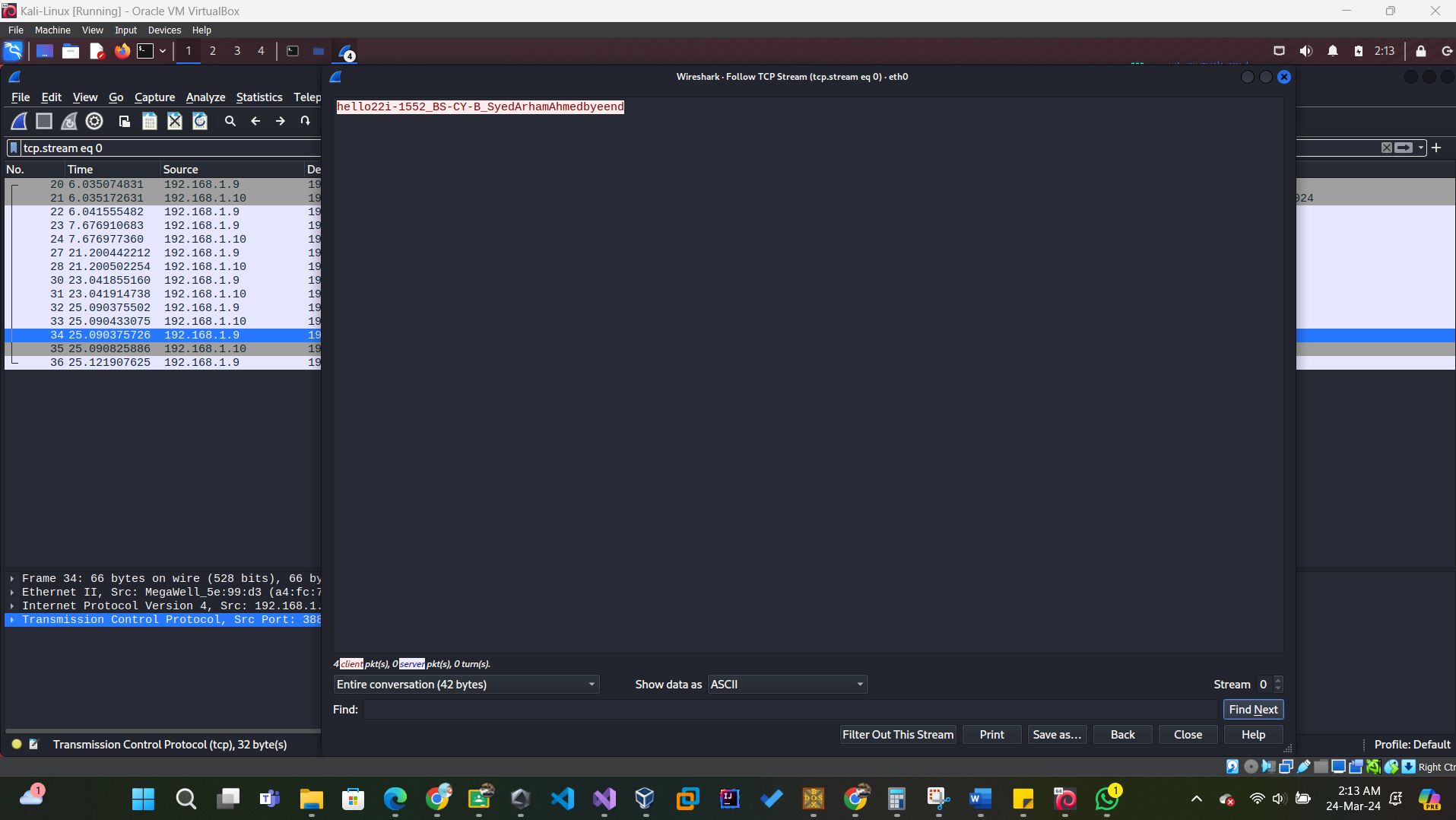


Now that we have seen that our client server connection is working properly, we will use Wireshark to capture the packets and see what messages were exchanged.



As we can see in the screenshot that I have filtered out the packets that were sent and received from my server to the client.

Now I will open the packets and see my messages.



As we can see in the above screenshot, these are the messages that my client computer sent to my server computer.

A screen shot of a computer

Description automatically generated

Now let’s answer the questions asked in the assignment manual.

**Q.1. What is the source IP address?**

The source IP address is the client as it initiates the connection, so it is 192.168.1.9

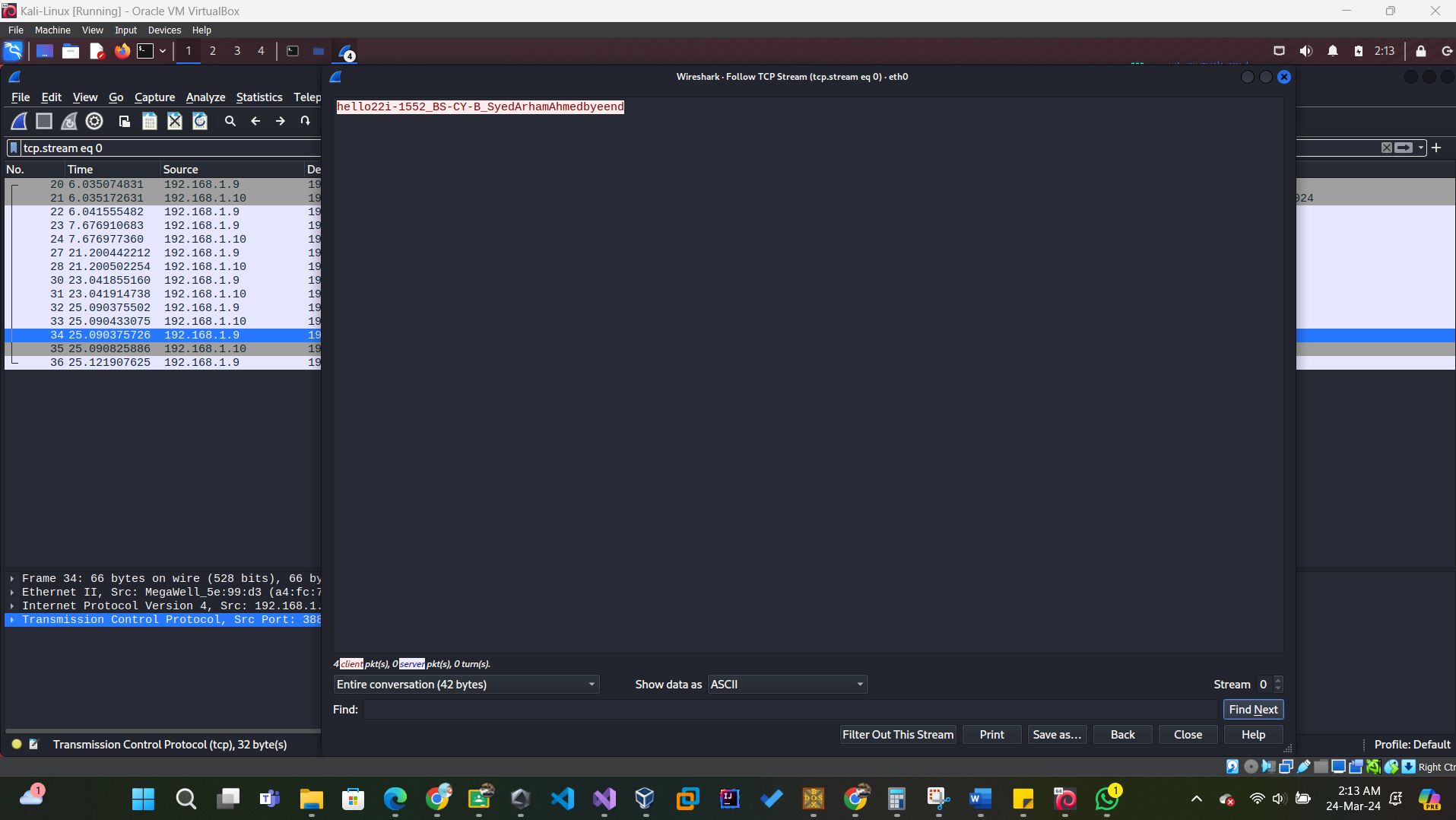
**Q.2. What is the destination IP address?**

The destination IP address is the server which received the client’s message, so it is 192.168.1.10

**Q.3. Using Wireshark, find out what message was sent by client to server.**

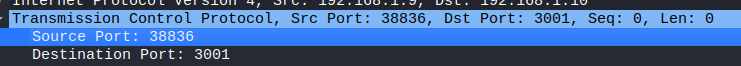
Already answered it above, just attaching the screenshot for reference. The messages were:

Hello,22i-1552\_BS-CY-B\_SyedArhamAhmed,bye,end



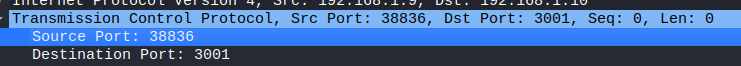
**Q.4. What is the source port number?**

As I mentioned above that client is the source, so its port number is 38836.

****

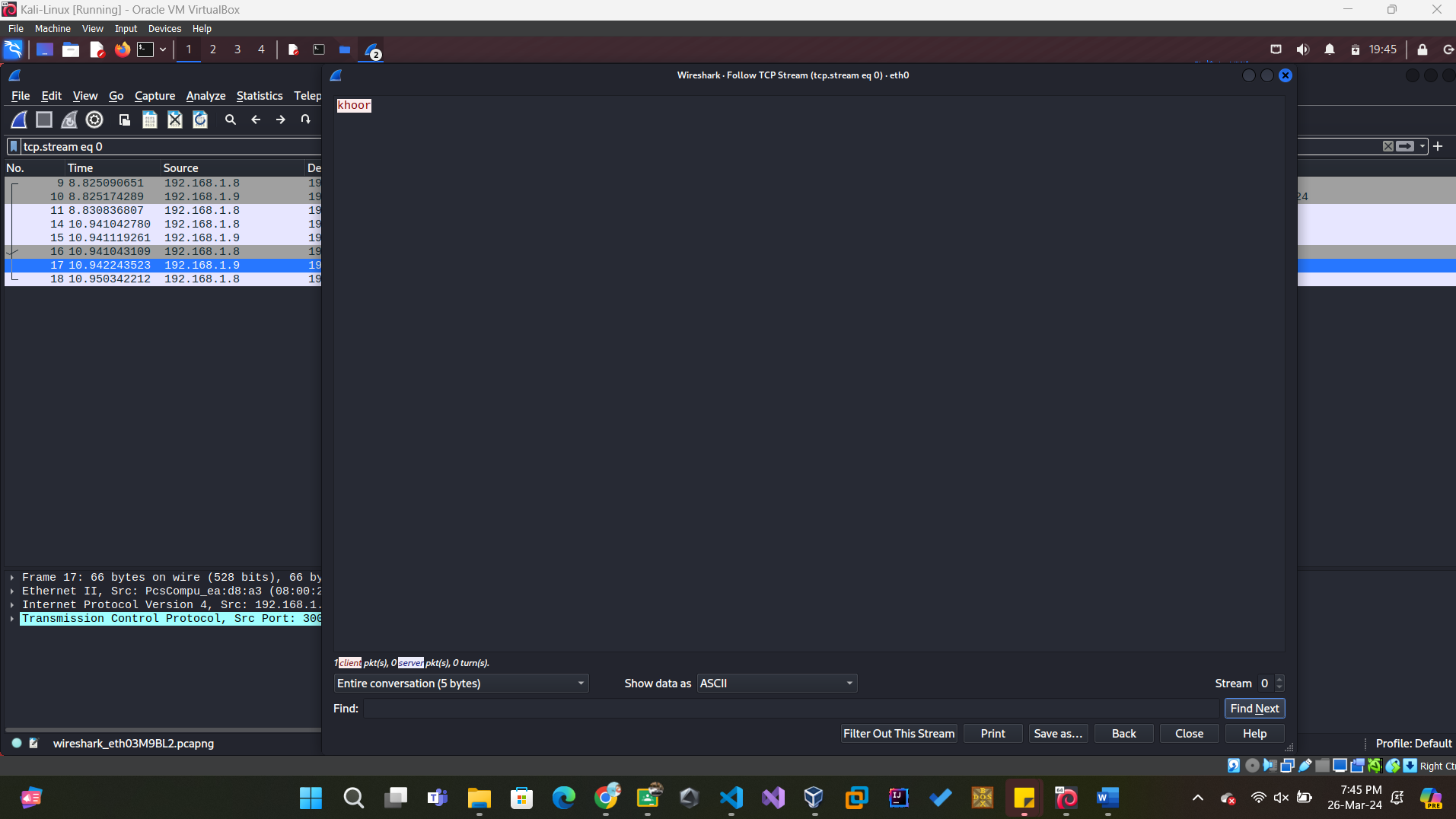
**Q.5. What is the destination port number?**

The destination is the server, and its port number is 3001.

****

Now I will modify my code so that Wireshark can’t just tell me the message directly. For that I can just simply encrypt the message before sending it and decrypt it on the server side.

**Q.6. After modifying code, show what difference you have observed in Wireshark after step no 11.**

****

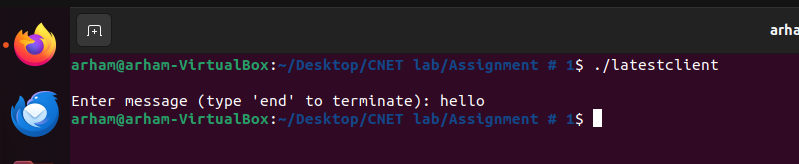
As now I have applied encryption to my message at the client i.e. senders end, Wireshark shows me the encrypted word and the packet sniffer cannot make any sense out of it.

A screenshot of a computer

Description automatically generated  
But on the receiver side the message hello is received correctly as I have applied decryption at the server end.

A computer screen shot of a computer

Description automatically generated  
The above code is my server code which is decrypting the received message. I have used Caesar cipher here.

A screen shot of a computer

Description automatically generated

The above is my client code i.e. sender code and I have encrypted my message here using Caesar cipher. I entered “hello” which was encrypted and become “khoor” as we can see in the Wireshark packet, but on the receiver side we can easily understand and view the message as it was decrypted after being received.

That’s all for the report, thank you for reading till the end!