Department of Computer Science

CS2005 Networks & Operating Systems Task 1

Academic Year 2023-24

Nafiz Ahamed Efty

2209061

Table of Contents

1. Introduction

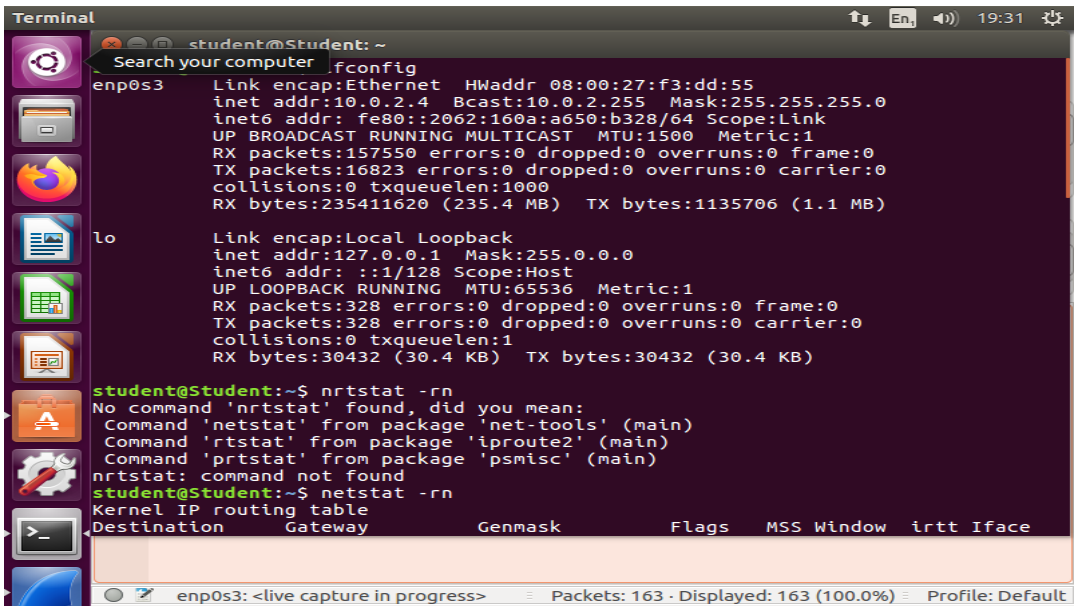
This report will begin by documenting a variety of configuration tests conducted on a network that consists of two interconnected subnets through a router. The primary goal of this section is to showcase the proper functionality of the network by utilizing diverse commands on the terminals of the hosts, including ifconfig, netstat, ping, and Wireshark. After the protocols employed by CalcClient and CalcServer will be documented and presented in a protocol table. The subsequent section will focus on providing documentation for the protocols associated with CalcClientUpdate and CalcServerUpdate, which represent the updated software version. This section will also include the identification of any related issues. Finally, a brief report will be directed to the NOSSoft managers, highlighting the presence of problems with their recent software update and urging them to address and resolve these issues promptly.

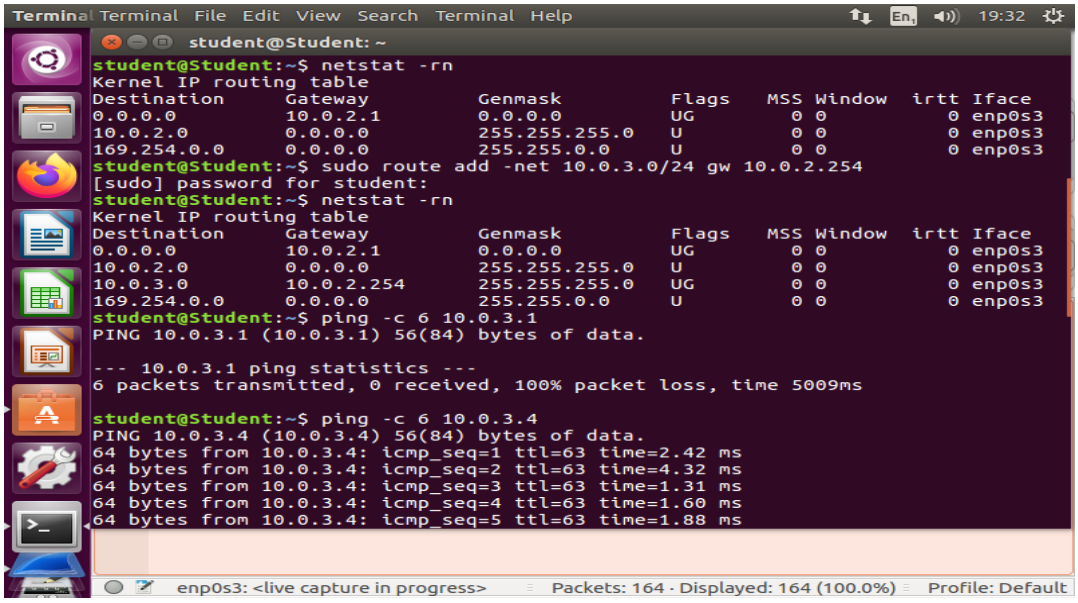
2. Test Network Documentation

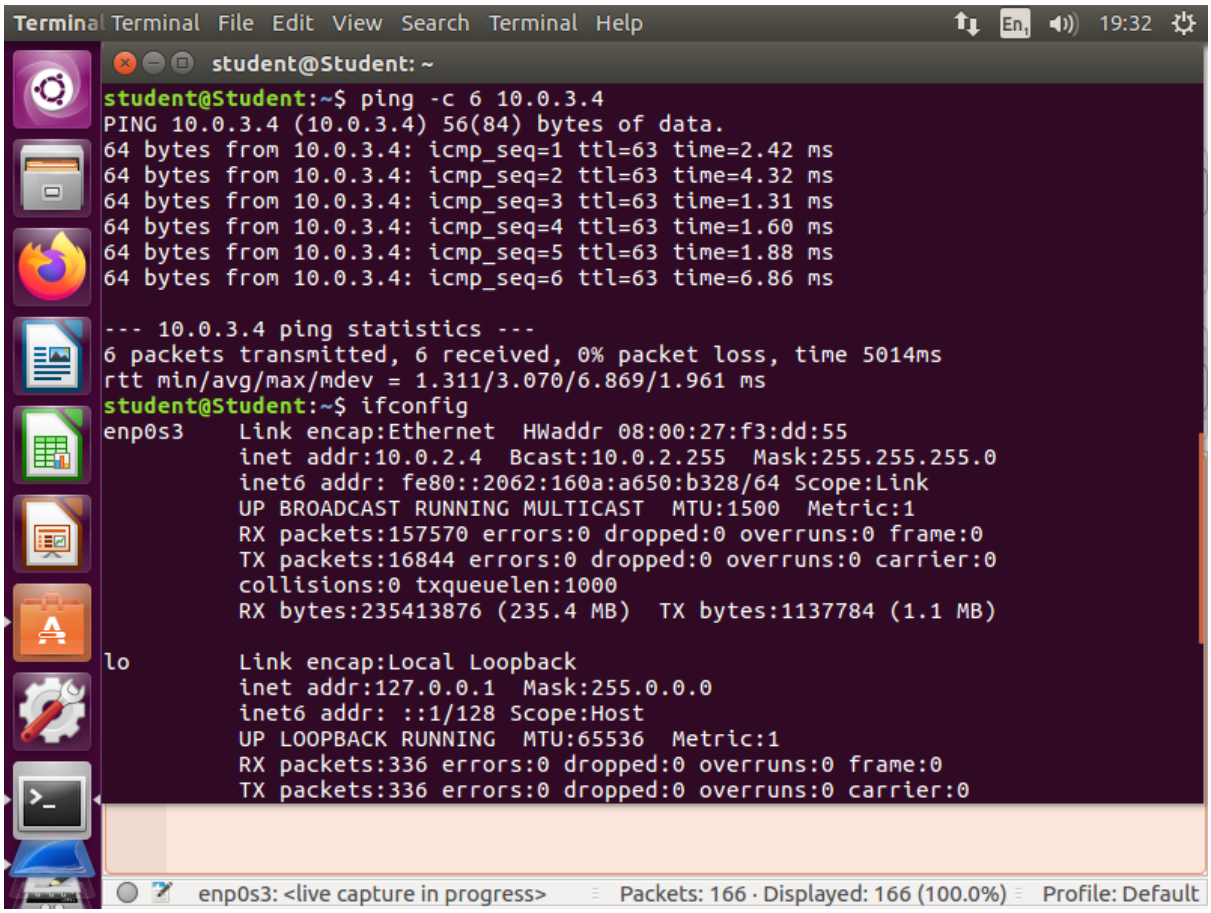
I am going to provide screenshots for proving the connection among Ubuntu1, Ubuntu2 and FreeBSD

Ubuntu1

Here I use ifconfig command for checking IP address and netstat -rn command show the routing Table where have a scenario of adding destination and gateway for specific IP.

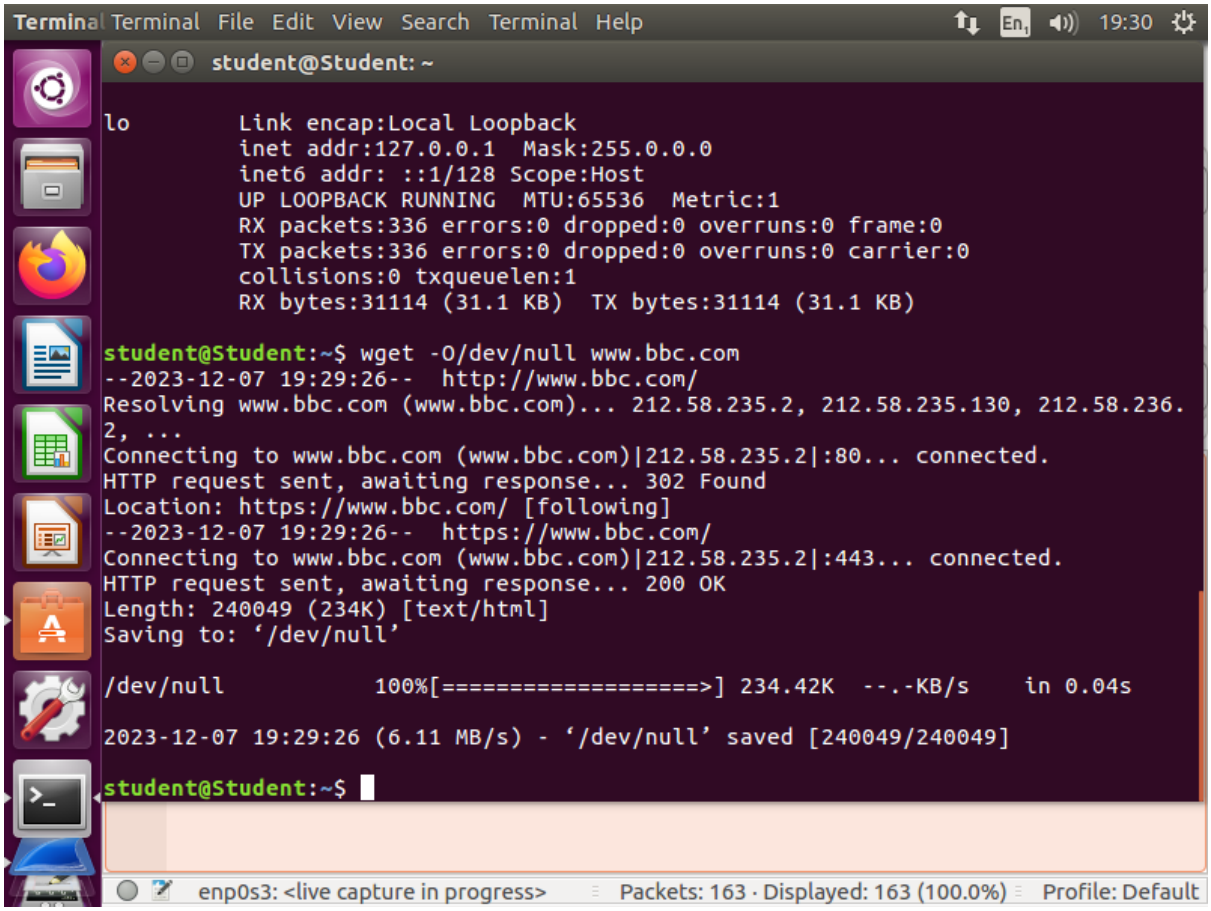




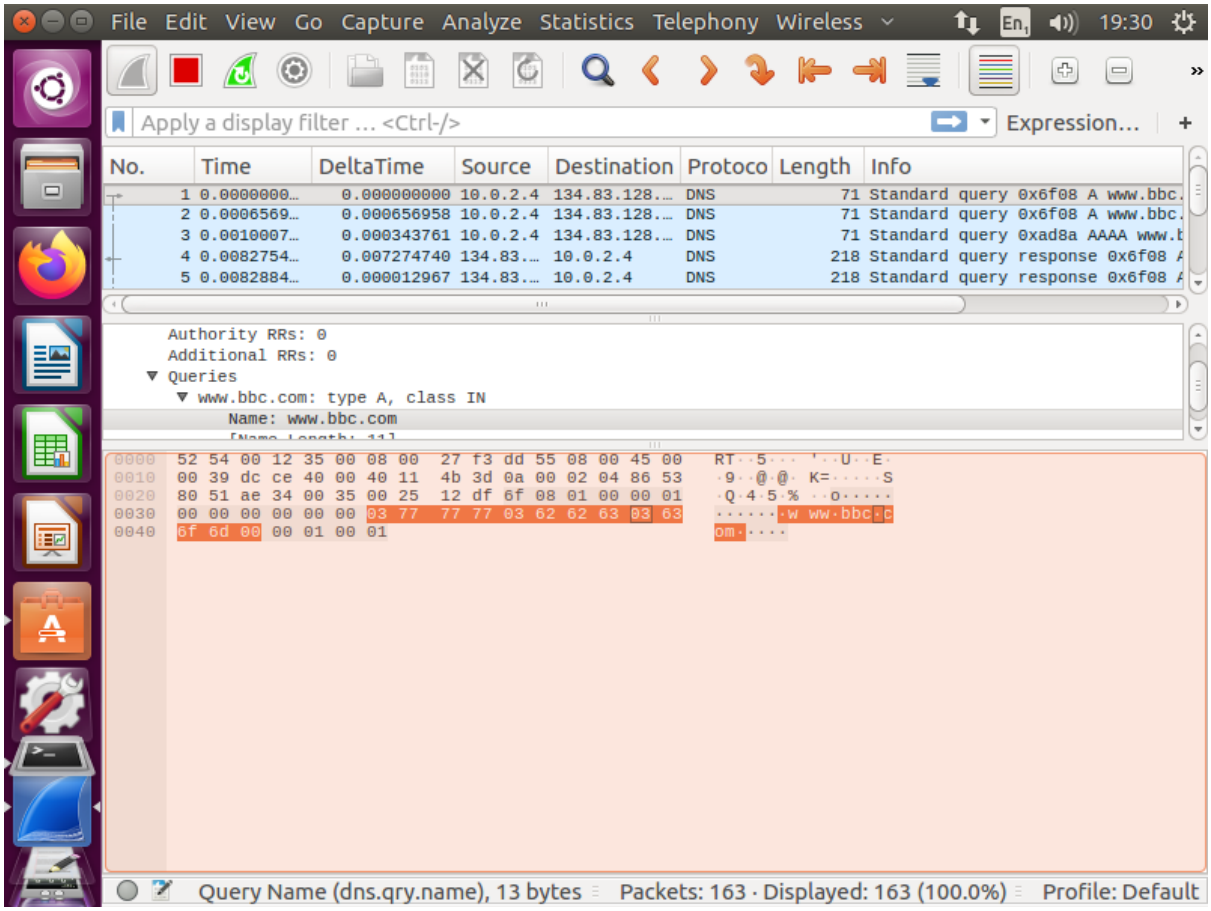


Wireshark -Ubuntu1

Wireshark is a widely used network protocol analyzer. It lets capture and inspect the data traveling back and forth on the network in real-time.

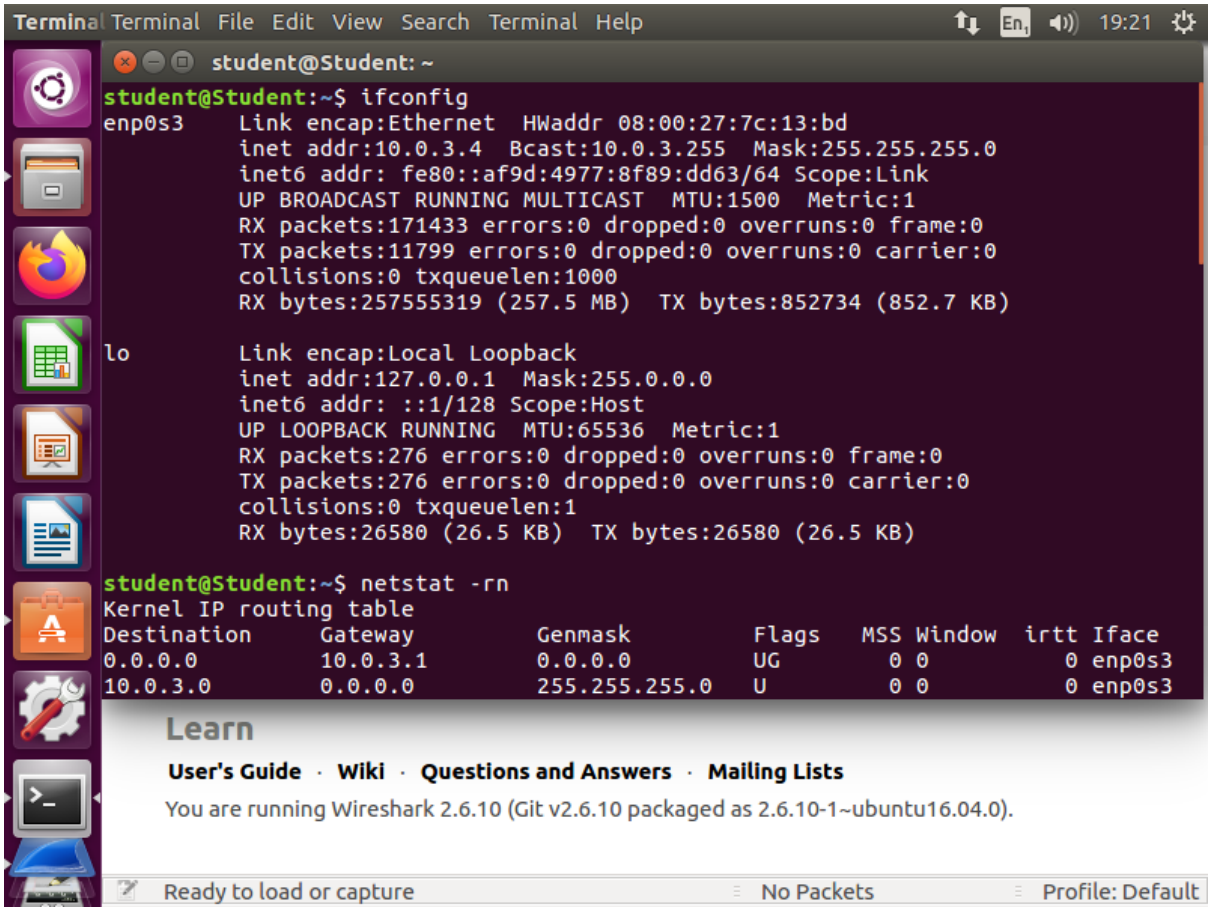


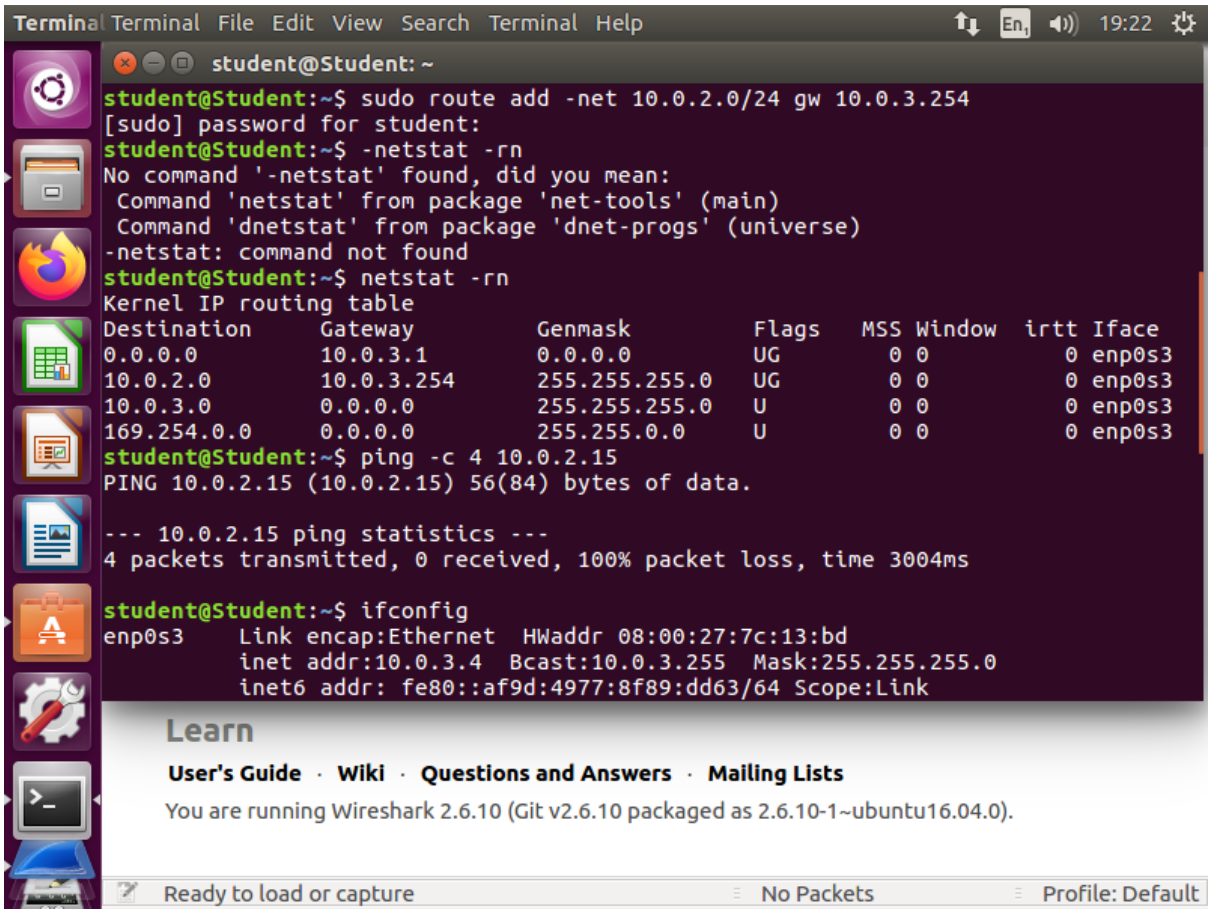
The Above Wireshark screenshots shows that source address is 10.0.2.4 and destination address is 212.58.235.2 because I search www.bbc.com from Ubuntu1 which IP address id 10.0.2.4 . and the command was ➔ wget -O/dev/null www.bbc.com .

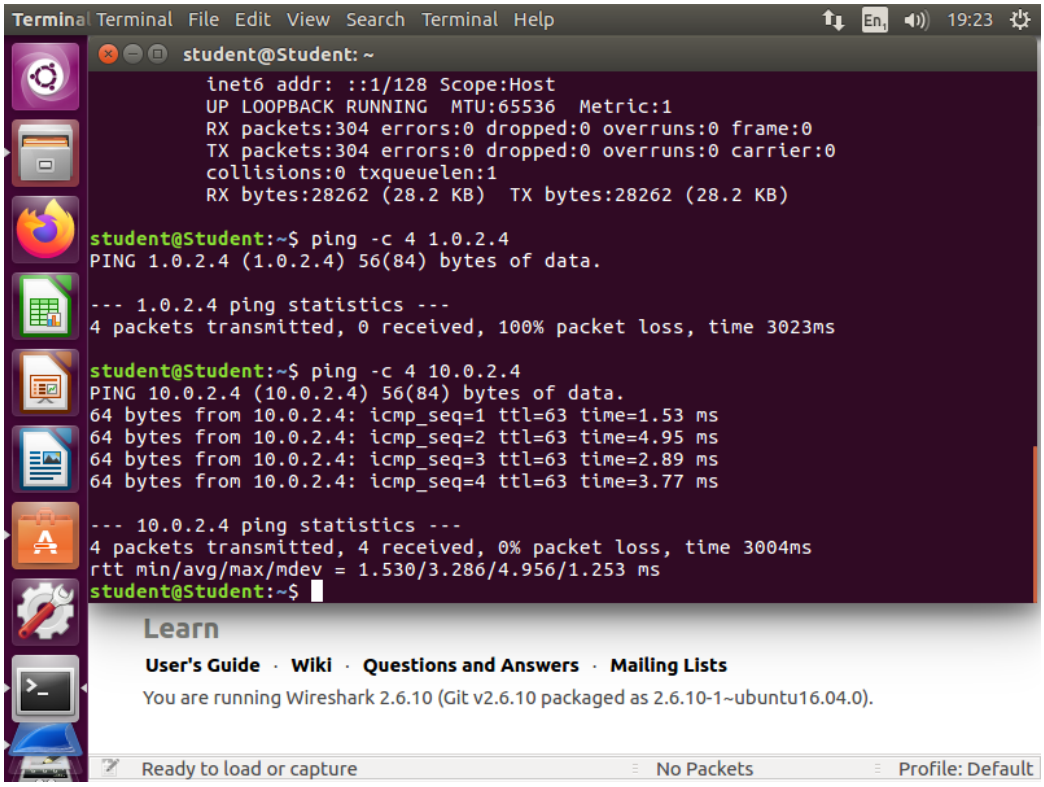


UBUNTU2

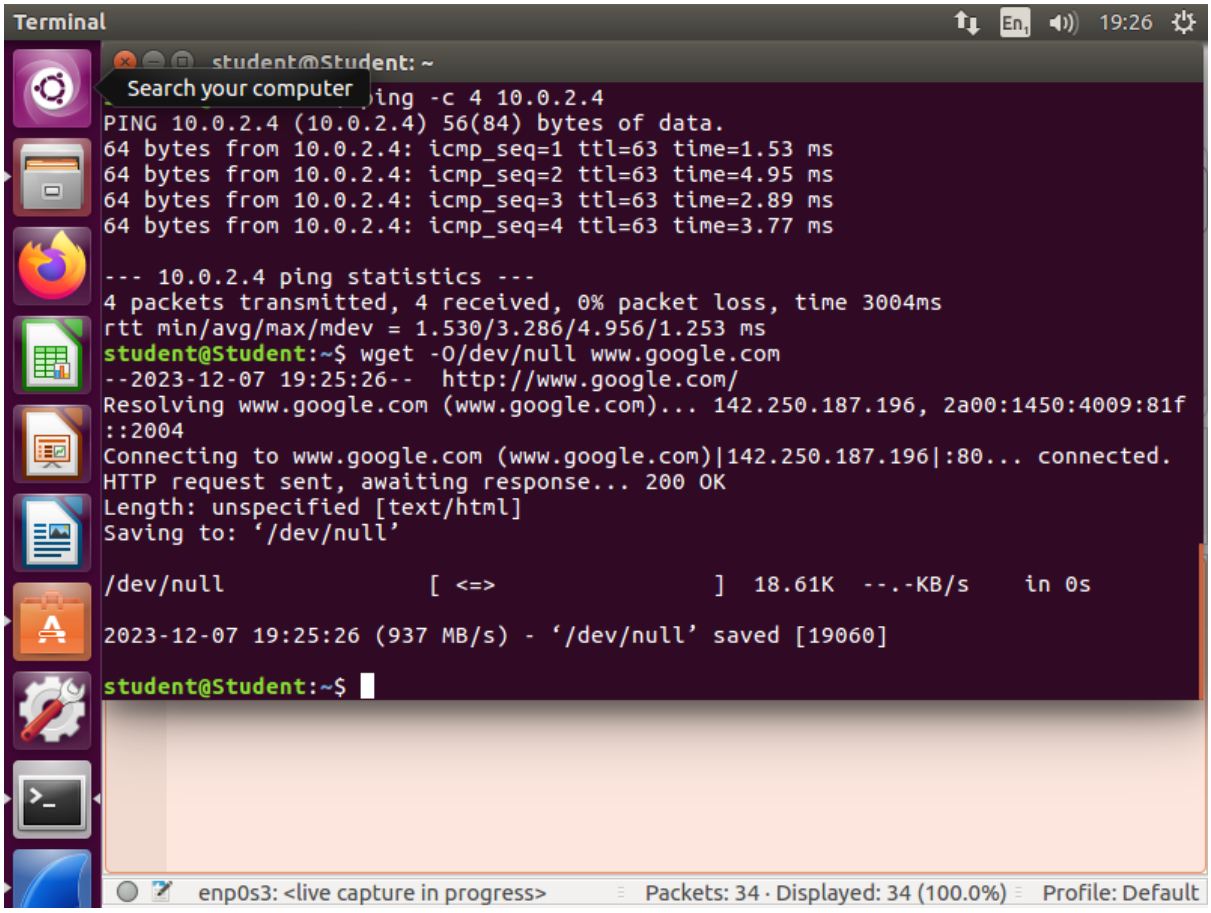
I use ifconfig and netstat -rn command ,add a destination and gateway.


Terminal Terminal File Edit View Search Terminal Help
En
19:21
student@Student: ~
student@Student:~$ ifconfig
enp0s3 Link encap:Ethernet HWaddr 08:00:27:7c:13:bd
inet addr:10.0.3.4 Bcast:10.0.3.255 Mask:255.255.255.0
inet6 addr: fe80::af9d:4977:8f89:dd63/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:171433 errors:0 dropped:0 overruns:0 frame:0
TX packets:11799 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:257555319 (257.5 MB) TX bytes:852734 (852.7 KB)
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:276 errors:0 dropped:0 overruns:0 frame:0
TX packets:276 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1
RX bytes:26580 (26.5 KB) TX bytes:26580 (26.5 KB)
student@Student:~$ netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 10.0.3.1 0.0.0.0 UG 0 0 0 enp0s3
10.0.3.0 0.0.0.0 255.255.255.0 U 0 0 0 enp0s3
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
You are running Wireshark 2.6.10 (Git v2.6.10 packaged as 2.6.10-1~ubuntu16.04.0).
Ready to load or capture
No Packets
Profile: Default


Terminal Terminal File Edit View Search Terminal Help
En
19:22
student@Student: ~
student@Student:~$ sudo route add -net 10.0.2.0/24 gw 10.0.3.254
[sudo] password for student:
student@Student:~$ -netstat -rn
No command '-netstat' found, did you mean:
Command 'netstat' from package 'net-tools' (main)
Command 'dnetstat' from package 'dnet-progs' (universe)
-netstat: command not found
student@Student:~$ netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 10.0.3.1 0.0.0.0 UG 0 0 0 enp0s3
10.0.2.0 10.0.3.254 255.255.255.0 UG 0 0 0 enp0s3
10.0.3.0 0.0.0.0 255.255.255.0 U 0 0 0 enp0s3
169.254.0.0 0.0.0.0 255.255.0.0 U 0 0 0 enp0s3
student@Student:~$ ping -c 4 10.0.2.15
PING 10.0.2.15 (10.0.2.15) 56(84) bytes of data.
--- 10.0.2.15 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3004ms
student@Student:~$ ifconfig
enp0s3 Link encap:Ethernet HWaddr 08:00:27:7c:13:bd
inet addr:10.0.3.4 Bcast:10.0.3.255 Mask:255.255.255.0
inet6 addr: fe80::af9d:4977:8f89:dd63/64 Scope:Link
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
You are running Wireshark 2.6.10 (Git v2.6.10 packaged as 2.6.10-1~ubuntu16.04.0).
Ready to load or capture
No Packets
Profile: Default



Wireshark-Ubuntu2



For ensuring that Ubuntu2 can communicate with the www (world wide web). FreeBSD is working here. The command is ➔ Wget -O/dev/null www.google.com. The Ubuntu2 server can transfer and receive data from Internet .But the destination cannot get real IP .Because of using FreeBSD NAT Connection.

steps:

.

Ubuntu2

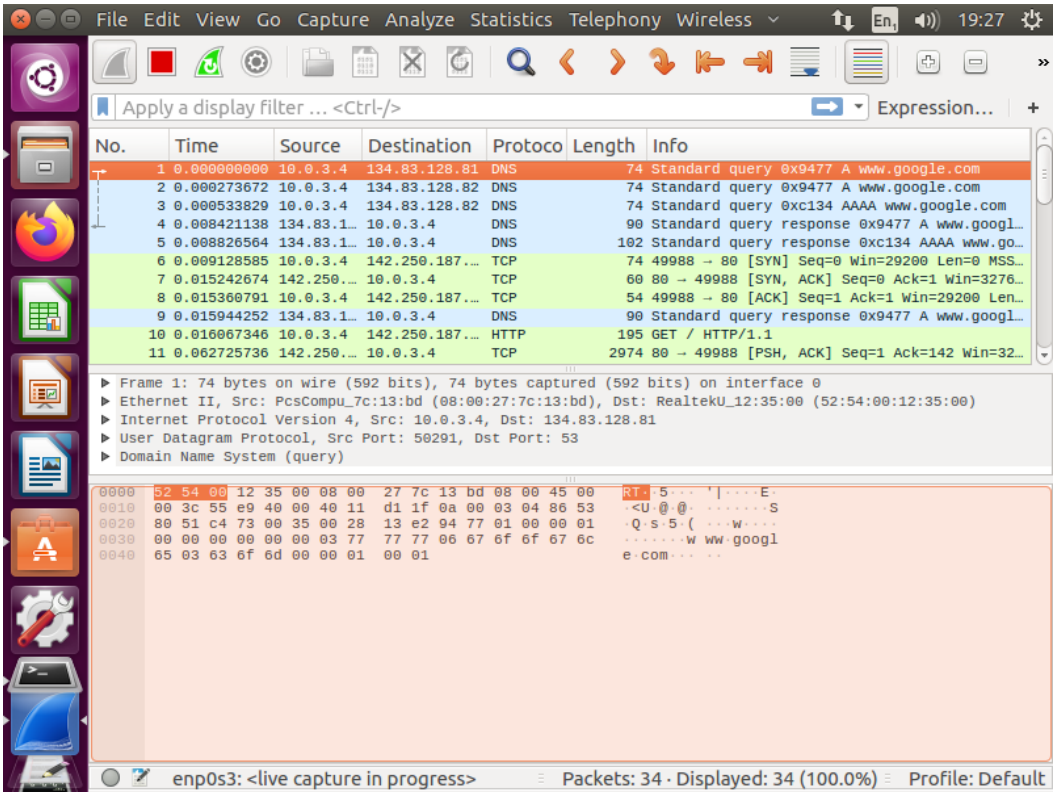
IP= 10.0.2.4

FreeBSD

Make the routing table change the IP and set a new IP.

Internet	Receive the packets from the new IP, respond, and send the packets by new IP and FreeBSD matching the IP with the routing table and then send the packets to root IP.

I captured the data transmission with Wireshark. I try to communicate with the internet. I searched www.google.com from Ubuntu2 with command. In Wireshark the Destination was 34.134.83.81 the source was 10.0.3.4. The source port is 50291 and the destination port is 53. Both are TCP Port.



FreeBSD

Ifconfig ➔ ifconfig has been widely used, modern Linux distributions are shifting towards using the ip command for network configuration. The ip command is more feature-rich and provides a more unified and consistent interface for configuring network settings

Such as:

1. Displaying Interface Information
2. Activating and Deactivating Interfaces
3. Setting IP Addresses
4. Changing MAC Addresses

```
        ether 08:00:27:60:b3:82
        inet 10.0.3.254 netmask 0xffffff00 broadcast 10.0.3.255
        inet6 fe80::a00:27ff:fe60:b382%em1 prefixlen 64 scopeid 0x2
        nd6 options=23<PERFORMNUD,ACCEPT_RTADV,AUTO_LINKLOCAL>
        media: Ethernet autoselect (1000baseT <full-duplex>)
        status: active
em2: flags=8802<BROADCAST,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=9b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM>
        ether 08:00:27:00:22:1e
        nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
        media: Ethernet autoselect (1000baseT <full-duplex>)
        status: active
em3: flags=8802<BROADCAST,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=9b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM>
        ether 08:00:27:d7:0f:6f
        nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
        media: Ethernet autoselect (1000baseT <full-duplex>)
        status: active
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> metric 0 mtu 16384
        options=600003<RXCSUM,TXCSUM,RXCSUM_IPV6,TXCSUM_IPV6>
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x5
        inet 127.0.0.1 netmask 0xff000000
        nd6 options=21<PERFORMNUD,AUTO_LINKLOCAL>
root@freebsd:~ #
```

Ping -c 4 10.0.2.4➔

Trying to Send 4 data packets from router (FreeBSD) to server Ununtu1 for checking internet connection.

The NAT router 1 and 2. - 10.0.2.254 means freebsd connected to Nat route 1. - 10.0.3.254 means freebsd connected to Nat route 2.

```
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x5
        inet 127.0.0.1 netmask 0xff000000
        nd6 options=21<PERFORMNUD,AUTO_LINKLOCAL>
root@freebsd:~ # ping -c 4 10.0.2.4
PING 10.0.2.4 (10.0.2.4): 56 data bytes
64 bytes from 10.0.2.4: icmp_seq=0 ttl=64 time=1.875 ms
64 bytes from 10.0.2.4: icmp_seq=1 ttl=64 time=0.754 ms
64 bytes from 10.0.2.4: icmp_seq=2 ttl=64 time=0.994 ms
64 bytes from 10.0.2.4: icmp_seq=3 ttl=64 time=0.677 ms

--- 10.0.2.4 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.677/1.075/1.875/0.476 ms
root@freebsd:~ # ping -c 4 10.0.3.4
PING 10.0.3.4 (10.0.3.4): 56 data bytes
64 bytes from 10.0.3.4: icmp_seq=0 ttl=64 time=1.974 ms
64 bytes from 10.0.3.4: icmp_seq=1 ttl=64 time=0.826 ms
64 bytes from 10.0.3.4: icmp_seq=2 ttl=64 time=0.854 ms
64 bytes from 10.0.3.4: icmp_seq=3 ttl=64 time=0.507 ms

--- 10.0.3.4 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.507/1.040/1.974/0.556 ms
root@freebsd:~ #
```

4 packets transmitted, 4 packets received, no packet loss.

Checking complete. Connection is ok.

Ping -c 4 10.0.3.4➔

Now I will check the connectivity FreeBSD and Ubuntu2

The packets are not loss.100% transmit and receive.

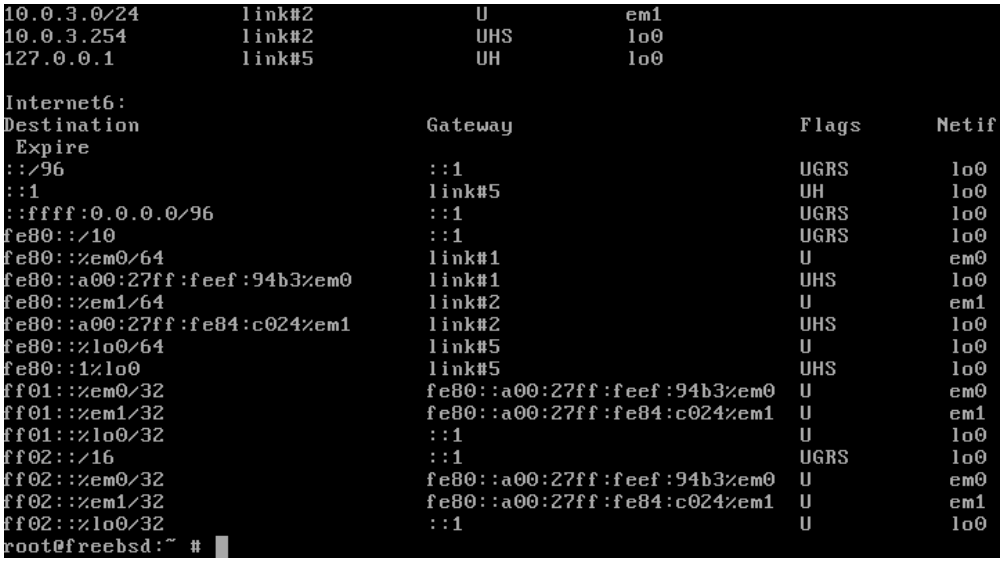
```
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.677/1.075/1.875/0.476 ms
root@freebsd:~ # ping -c 4 10.0.3.4
PING 10.0.3.4 (10.0.3.4): 56 data bytes
64 bytes from 10.0.3.4: icmp_seq=0 ttl=64 time=1.974 ms
64 bytes from 10.0.3.4: icmp_seq=1 ttl=64 time=0.826 ms
64 bytes from 10.0.3.4: icmp_seq=2 ttl=64 time=0.854 ms
64 bytes from 10.0.3.4: icmp_seq=3 ttl=64 time=0.507 ms

--- 10.0.3.4 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.507/1.040/1.974/0.556 ms
root@freebsd:~ # ping -c 6 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=113 time=3.964 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=3.639 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=113 time=3.766 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=113 time=4.456 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=113 time=3.833 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=113 time=3.903 ms

--- 8.8.8.8 ping statistics ---
6 packets transmitted, 6 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 3.639/3.927/4.456/0.258 ms
root@freebsd:~ #
```

netstat -rn ➔

This command helps me to find out the Destination and gateway



3.calcClient and calcServer Documentation

In this segment, I executed calcClient on Ubuntu2 and calcServer on Ubuntu1 terminals. Subsequently, I systematically captured the protocol for all transactions using Wireshark, yielding the following outcomes. It is noteworthy that I have detected certain issues within the new version, and these will be itemized in the subsequent section.

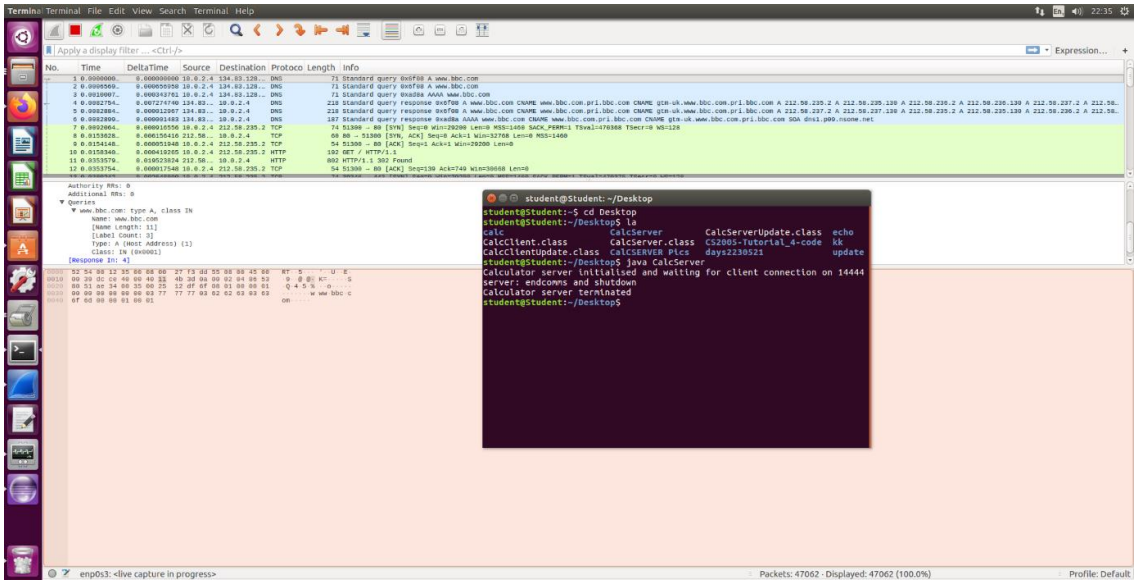
Protocol

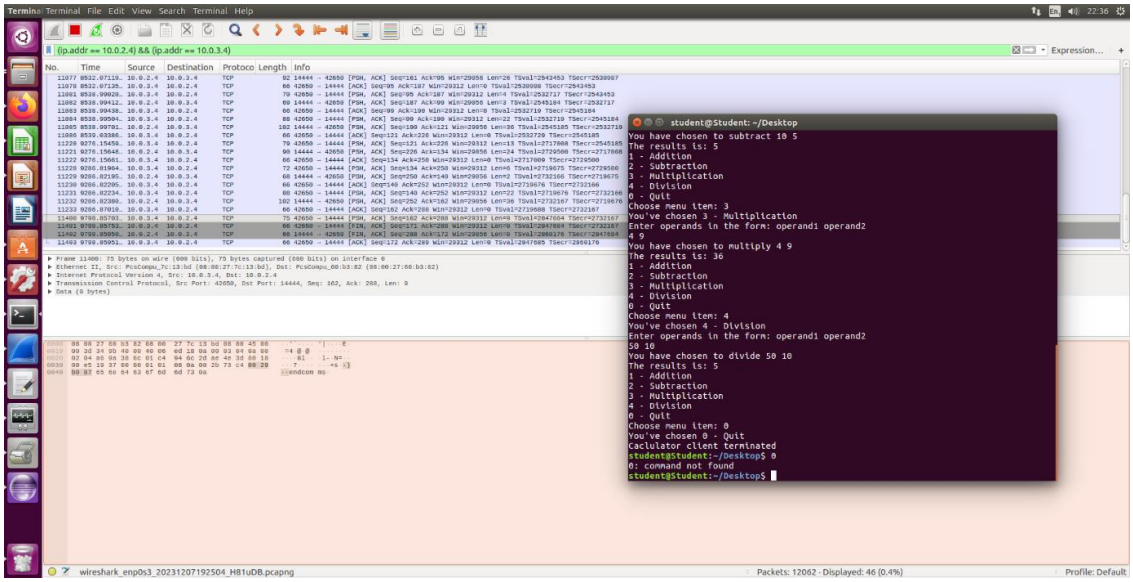
	Ubuntu2 **calcClient**	**Ubuntu1** calcServer
1		[Run CalCServer]
2	[Run CalcClient]	
	[Type Server IP and Enter]	
		[Server Response]
3		SEND “ calculator server ready and waiting “

4	RECEIVE “calculator server ready and waiting”	
5	SEND “1”	
6		RECEIVE “1”
7	SEND “add operands”	
8		RECEIVE “add operands”
9		SEND “send operands to add”
10	RECEIVE “send operands to add”	
11	SEND “4 and 8”	
12		RECEIVE “4 and 8”
13		SEND “12”
14	PRINT “12”	
15	SEND “next operation please”	
16		RECEIVE “next operation please”
17		SEND “calculator Server ready and waiting”
18	RECEIVE “calculator Server ready and waiting”	
19	SEND “2”	
20		RECEIVE “2”
21	SEND “sub operands”	
22		RECEIVE “sub operands”
23		SEND “send operands to subtract”
24	RECEIVE “send operands to subtract”	
25	SEND “10 and 5”	
26		Receive “10 and 5”

27		SEND “5”
28	PRINT “5”	
29	SEND “next to operation please”	
30		RECEIVE “next operation please”
31		SEND “calculator Server ready and waiting”
32	RECEIVE “calculator Server ready and waiting”	
33	SEND “3”	
34		RECEIVE “3”
35	SEND “multi operands”	
36		RECEIVE “multi operands”
37		SEND “send operands to multiply”
38	RECEIVE “send operands to multiply”	
39	SEND “4 and 9”	RECEIVE “4 and 9”
40		SEND “36”
41	PRINT “36”	
42	SEND “next operation please”	
43		RECEIVE “next operation please”
44		SEND “calculator Server ready and waiting”
45	RECEIVE “calculator Server ready and waiting”	
46	SEND “4”	
47		RECEIVE “4”
48	SEND “div operands”	
49		RECEIVE “div operands”

50		SEND “send operands to divide”
51	RECEIVE “send operands to divide”	
52	SEND “50 10”	
53		RECEIVE “50 10”
54		SEND “5”
55	PRINT “5”	
56	SEND “next operation please”	
57		RECEIVE “next operation please”
58		SEND “calculator Server ready and waiting”
59	RECEIVE “calculator Server ready and waiting”	
60	SEND “0”	
61		RECEIVE “0”
62		SEND “endcom ms”
63		[TERMINATE]





4. calcClientUpdate and calcServerUpdate Documentation

In this segment, I executed calcClientUpdate on Ubuntu2 and calcServerUpdate on Ubuntu1 terminals. Subsequently, I methodically recorded the protocol for all transactions using Wireshark, resulting in the obtained outcomes. It is essential to note that I have pinpointed certain issues within the latest version, and these will be outlined in the upcoming section.

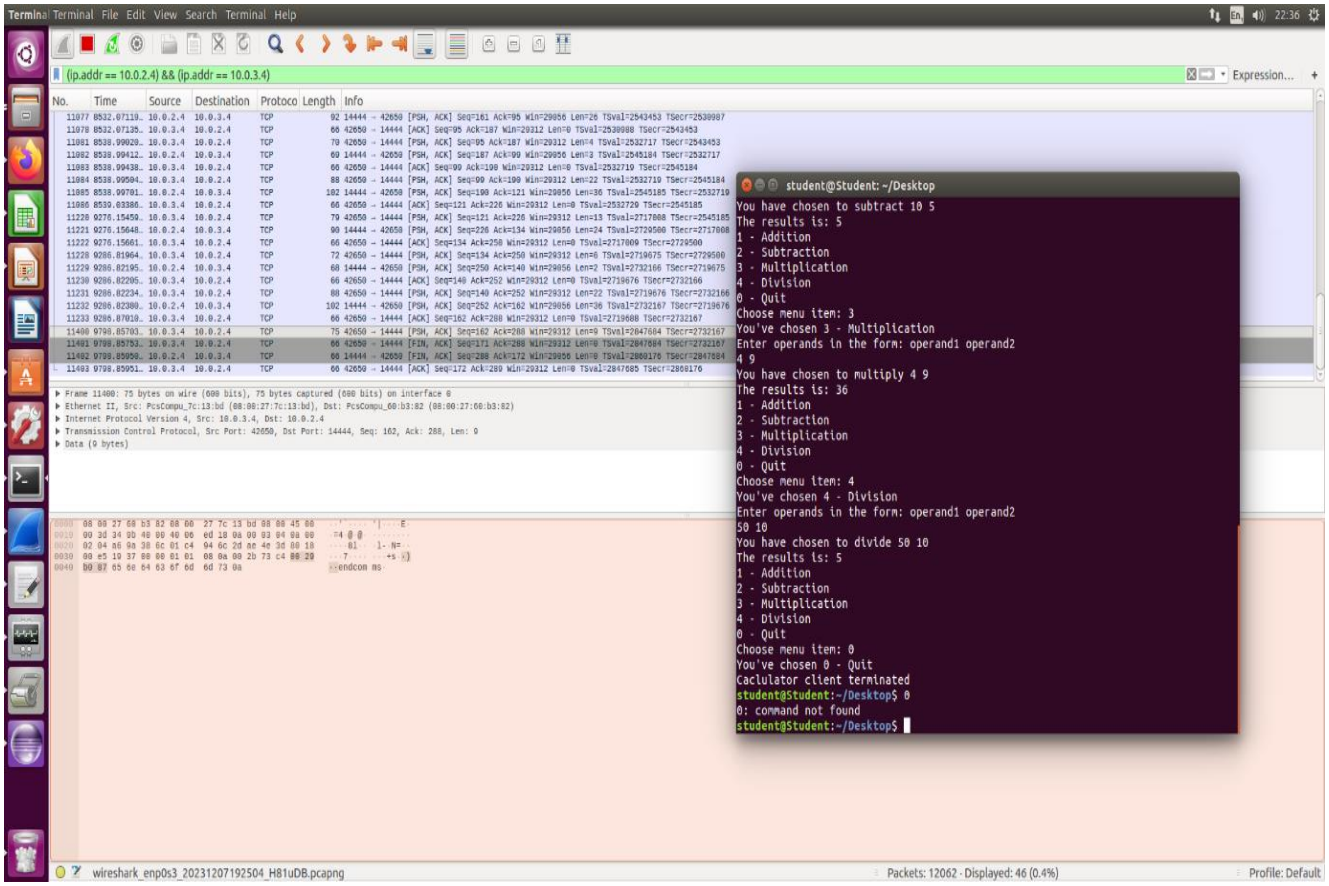
Protocol

	Ubuntu2 CalcClientUpdate	Ubuntu1 CalcServerUpdate
1		[RUN CalcServerUpdate]
2	[RUN CalcClientUpdate]	
	[Type Server IP and Enter]	
		[connect with the client]
3		SEND “calculator server ready and waiting” TO CalcClientUpdate
4	RECEIVE” calculator server ready and waiting” FROM CalcServerUpdate	
5	SEND “1” TO Update Server	
6		RECEIVE “1” FROM client
7	SEND “add operands”	
8		RECEIVE “add operands”
9		SEND “send operands to add”
10	RECEIVE “send operands to add”	
11	SEND “5 and 5”	

12		RECEIVE “5 and 5”
13		SEND “0”
14	PRINT “0”	
15	SEND “next operation please”	
16		RECEIVE “next operation please”
17		SEND “calculator Server ready and waiting”
18	RECEIVE “calculator Server ready and waiting”	
19	SEND “2”	
20		RECEIVE “2”
21	SEND “**sub operands**”	
22		RECEIVE “sub operands”
23		SEND “send operands to subtract”
24	RECEIVE “send operands to subtract”	
25	**SEND “4 AND 2”**	
26		**RECEIVE “4 AND 2”**
27		**SEND “6”**
28	**PRINT “6”**	
29	SEND “next operation please”	
30		RECEIVE “next operation please”
31		SEND “calculator Server ready and waiting”

32	RECEIVE “calculator Server ready and waiting”	
33	SEND “3”	
34		RECEIVE “3”
35		
36	SEND “multi operands”	
37		RECEIVE “multi operands”
38		SEND “send operands to multiply”
39	RECEIVE “send operands to multiply”	
40	SEND “6 AND 3”	
41		RECEIVE “6 AND 3”
42		**SEND “36”**
43	**PRINT “36”**	
44	SEND “next operation please”	
45		RECEIVE “next operation please”
46		SEND “calculator Server ready and waiting”
47	RECEIVE “calculator Server ready and waiting”	
48	**SEND “4”**	
49		**RECEIVE “4”**
50		**SEND “endcom ms”**

51	**RECEIVE “endcom ms”**	
52		**[TERMINATE]**
53	[RUN CalcClientUpdate]	[RUN CalcServerUpdate]
54		SEND “calculator server ready and waiting “
55	RECEIVE” calculator server ready and waiting “	
56	SEND “0”	
57		RECEIVE “0”
58		**SEND “send operands to divide”**
59	**RECEIVE “send operands to divide”**	
60	SEND “15 and 3”	
61		RECEIVE “15 and 3”
62		[Calculated BY Serder and send the value to Client]
63	PRINT “5”	



5. Report to the NOSSoft Management

I am writing a report about CalcClientUpdate and CalcServerUpdate to the Nessoft Management

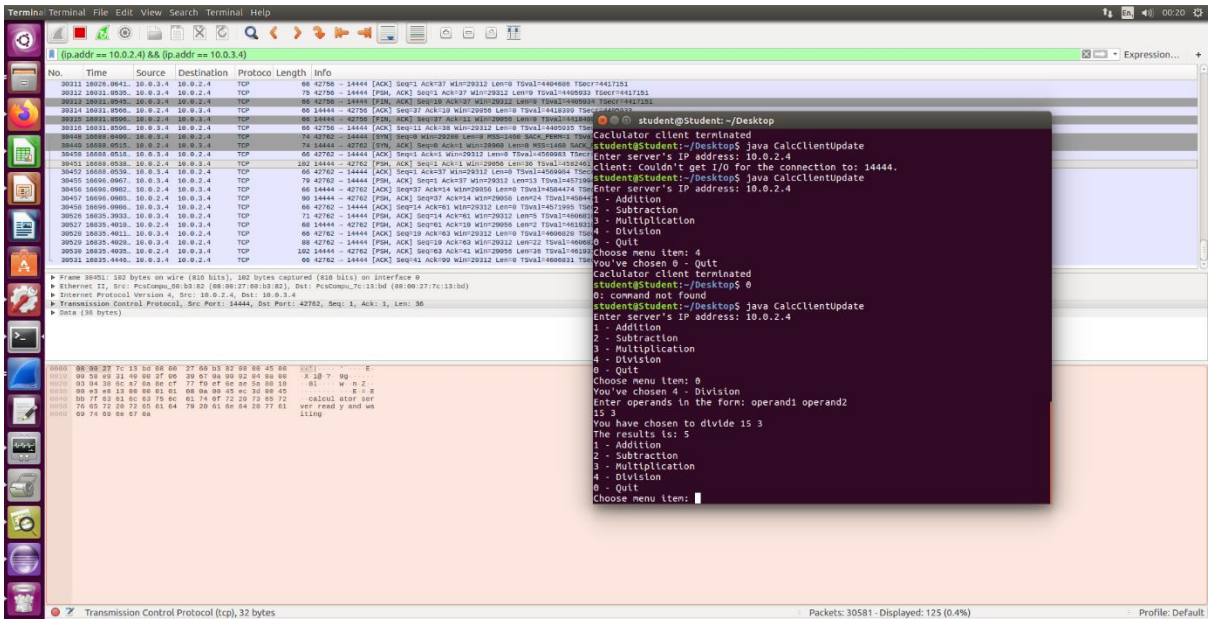
and serve some strong documents that the updated version is not working properly.

Problem 1: Additional problem ➔when user inputs 5 and 5. The server returned 0. The server makes it subtract. (add a photo)

Problem 2: Subtract problem ➔when user input 4 and 2. The server returns 6. The server makes it Addition. (add photo)

Problem 3: Multiplication problem ➔input 6 and 3 and output 36 from server. (add photo)

Problem 4: when the user inputs option 4 which is a response for divide, but the server takes it 0. If the user types 0 it will terminate. But here is doing the reverse. Input 4, the server takes 0, and input 0, the server takes 4. (add photo)



6. Conclusions

In this report, I initiated by establishing a network consisting of two subnets named Ubuntu1, Ubuntu2, and the FreeBSD router. Subsequently, I performed various configuration tests to verify the correct functioning of the network. These tests involved the utilization of diverse commands such as ifconfig, netstat -rn, and ping -c. Additionally, I utilized the Firefox browser to access the google, and bbc.com capturing transactions using Wireshark. The successful completion of these tests bolstered my confidence in the connectivity of the network. Following this, I transitioned to the next section where I began working with the initial version of NOSSoft. I executed calcClient in the Ubuntu1 terminal and calcServer on Ubuntu1, subsequently observing the transactions between them using Wireshark and documenting these interactions in a table. Similarly, I repeated this process for the updated version of NOSSoft . However, during the observation of transactions in the updated version, I identified several problems. Finally, I compiled a report detailing the identified issues and presented it to the NOSSoft managers, aiming to convince them of the imperative need to address and rectify these problems in the new update.