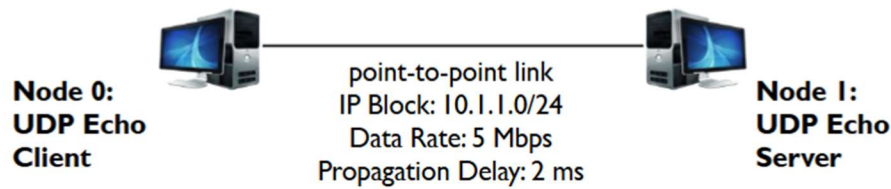


Located in ns-3.42/examples/tutorial/first.cc



Task 1:

a) Experimental Setup:

- i. 2 nodes
 - 1 network interface at each node
- ii. Point-to-point link:
 - Data Rate: 10 Mbps
 - Delay: 2 ms
- iii. IP address assignment: 192.168.2.0/24
- iv. Application:
 - UDP Echo Server on port 63
 - Packet size: 256 bytes

We take the “first.cc” example in the tutorials and only edit the parameters according to the above setup. The idea of this task is to understand how ns3 works and how it behaves:

```
Open ▾  snehas_myfirst.cc  Ln 70, Col 110  snehas_myfirst.cc
~/Desktop/ns3/ns-3.42/scratch

myfirst.cc  snehas_myfirst.cc

46  PointToPointHelper pointToPoint;
47  pointToPoint.SetDeviceAttribute("DataRate", StringValue("10Mbps")); //Modified Data Rate
48  pointToPoint.SetChannelAttribute("Delay", StringValue("2ms")); //Delay set to 2ms
49
50  NetDeviceContainer devices;
51  devices = pointToPoint.Install(nodes);
52
53  InternetStackHelper stack;
54  stack.Install(nodes);
55
56  Ipv4AddressHelper address;
57  address.SetBase("192.168.2.0", "255.255.255.0"); //Modified address from 1.1.1.0 to 192.168.2.0
58
59  Ipv4InterfaceContainer interfaces = address.Assign(devices);
60
61  UdpEchoServerHelper echoServer(63); //Modified port from 9 to 63
62
63  ApplicationContainer serverApps = echoServer.Install(nodes.Get(1));
64  serverApps.Start(Seconds(1.0));
65  serverApps.Stop(Seconds(10.0));
66
67  UdpEchoClientHelper echoClient(interfaces.GetAddress(1), 63); //Modified port from 9 to 63
68  echoClient.SetAttribute("MaxPackets", UintegerValue(1));
69  echoClient.SetAttribute("Interval", TimeValue(Seconds(1.0)));
70  echoClient.SetAttribute("PacketSize", UintegerValue(256)); //Modified Packet Size from 1024 to 256 bytes
71
```

Fig. 1. Modifications made to myfirst.cc

The above snap highlights the modifications made to “myfirst, as mentioned below:

- In line 47, we modify the data rate from 5 Mbps to 10 Mbps.
- In line 48, we verify if the propagation delay to 2ms.
- In line 57, we modify the IP address to 192.168.2.0
- In line 61, we modify the port value from 9 to 63.
- In line 70, we modify the packet size from 1024 to 256 bytes.”

b) Results - Terminal & Wireshark:

This was a straightforward task after the changes were made. The terminal output indicated successful initialization and execution of the simulation as shown below:

```
sneha@sneha-VirtualBox: ~/Desktop/ns3/ns-3.42
sneha@sneha-VirtualBox:~$ cd Desktop/ns3/
sneha@sneha-VirtualBox:~/Desktop/ns3$ cd ns-3.42/
sneha@sneha-VirtualBox:~/Desktop/ns3/ns-3.42$ ./ns3 run scratch/myfirst
[0/2] Re-checking globbed directories...
ninja: no work to do.
At time +2s client sent 1024 bytes to 1.1.1.2 port 9
At time +2.00369s server received 1024 bytes from 1.1.1.1 port 49153
At time +2.00369s server sent 1024 bytes to 1.1.1.1 port 49153
At time +2.00737s client received 1024 bytes from 1.1.1.2 port 9
sneha@sneha-VirtualBox:~/Desktop/ns3/ns-3.42$ ./ns3 run snehas_myfirst
[0/2] Re-checking globbed directories...
[2/2] Linking CXX executable /home/sne.../scratch/ns3.42-snehas_myfirst-default
At time +2s client sent 256 bytes to 192.168.2.2 port 63
At time +2.00223s server received 256 bytes from 192.168.2.1 port 49153
At time +2.00223s server sent 256 bytes to 192.168.2.1 port 49153
At time +2.00446s client received 256 bytes from 192.168.2.2 port 63
sneha@sneha-VirtualBox:~/Desktop/ns3/ns-3.42$
```

Fig. 2. Output Terminal

The highlighted command is used to obtain Wireshark packets:

```
myfirst.cc
snehas_myfirst.cc
~/Desktop/ns3/ns-3.42/scratch
Ln 75, Col 4
snehas_myfirst.cc
64 serverApps.Start(Seconds(1.0));
65 serverApps.Stop(Seconds(10.0));
66
67 UdpEchoClientHelper echoClient(interfaces.GetAddress(1), 63); //Modified port from 9 to 63
68 echoClient.SetAttribute("MaxPackets", IntegerValue(1));
69 echoClient.SetAttribute("Interval", TimeValue(Seconds(1.0)));
70 echoClient.SetAttribute("PacketSize", IntegerValue(256)); //Modified Packet Size from 1024 to 256 bytes
71
72 ApplicationContainer clientApps = echoClient.Install(nodes.Get(0));
73 clientApps.Start(Seconds(2.0));
74 clientApps.Stop(Seconds(10.0));
75 pointToPoint.EnablePcapAll("snehas_myfirst");
76
77 Simulator::Run();
78 Simulator::Destroy();
79 return 0;
80 }
```

Fig. 3. Pcap Generation

Wireshark packet traces show the expected flow of packets from a “point-to-point” connection with the new IP addresses. The first packet was sent from the client i.e. IP address: 192.168.2.1 to the server’s IP address: 192.168.2.2. The propagation delay was measured to be 2ms. The server promptly responded by echoing back the same packet.

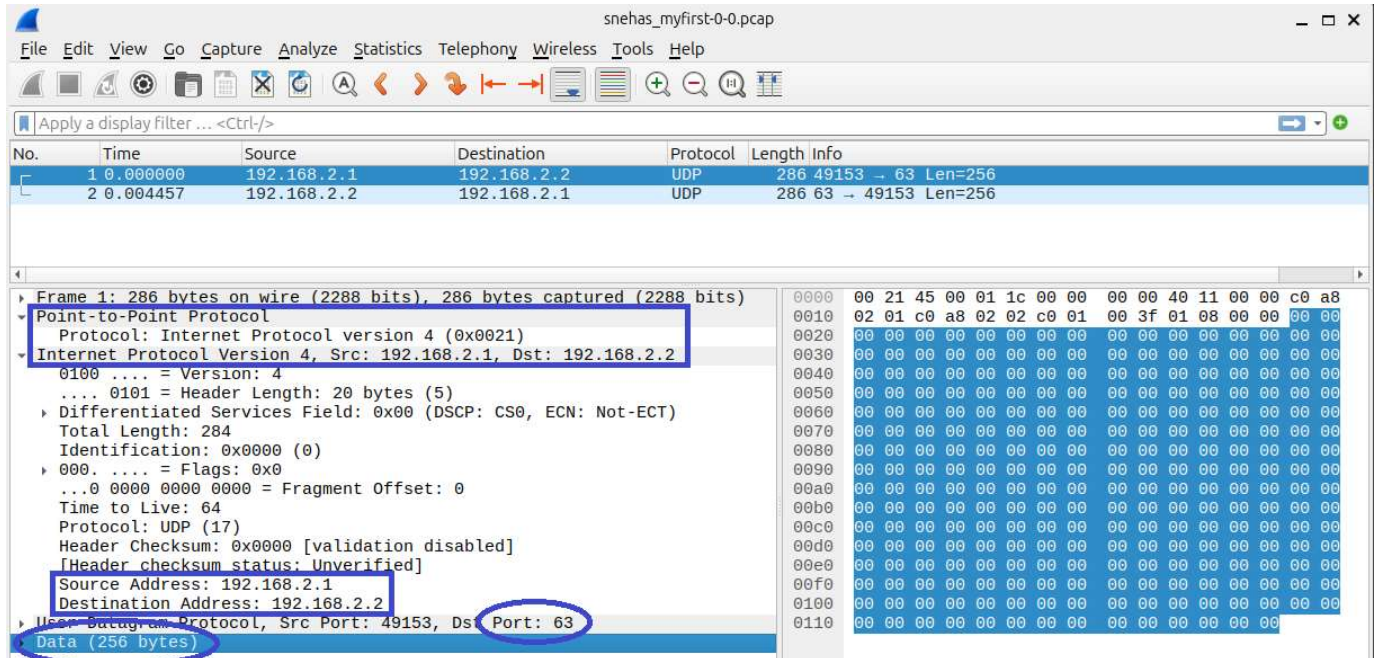


Fig. 3. 0-0.Pcap

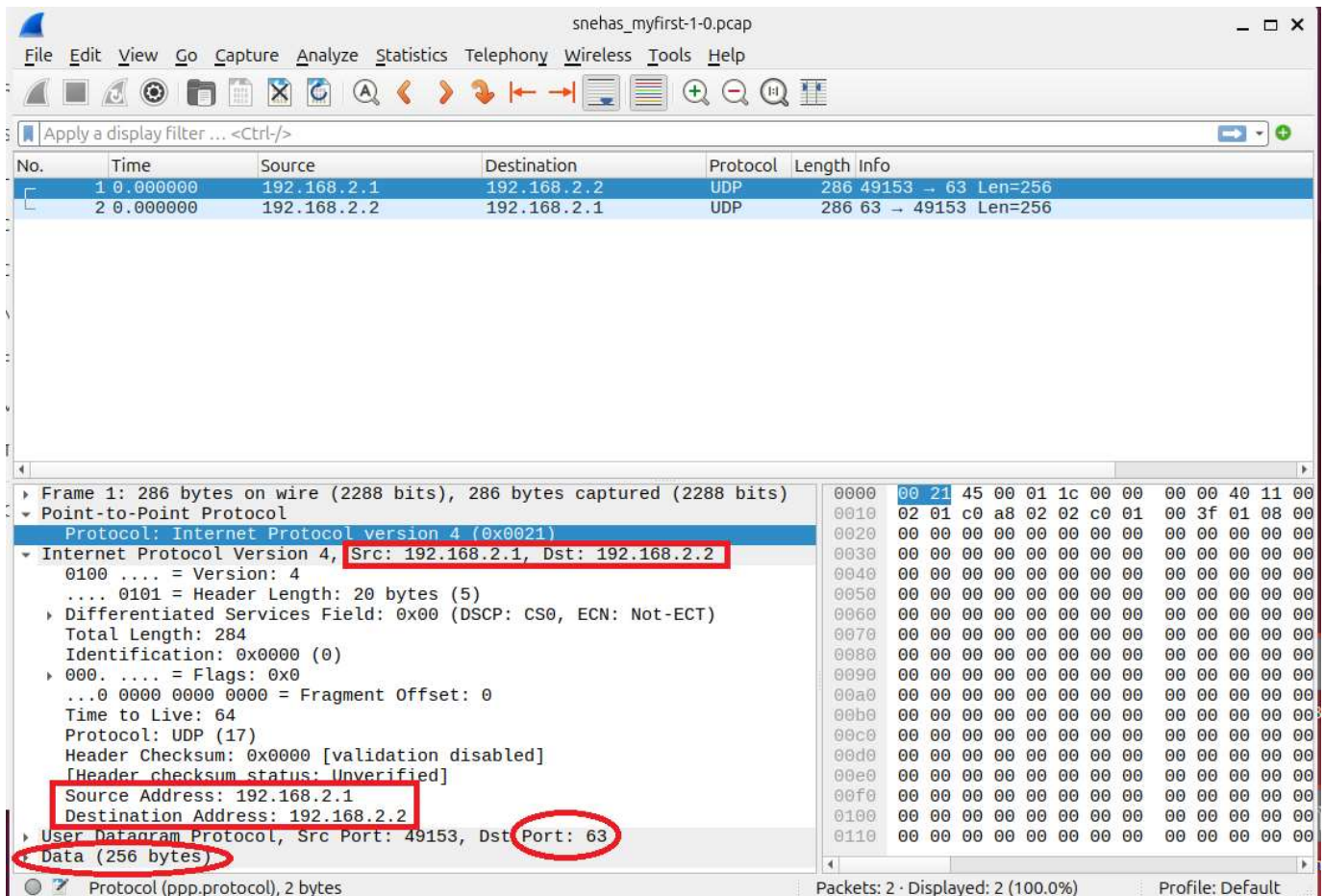


Fig. 4. 1-0.Pcap

Since it's a point-to-point connection, there are exactly 2 captures: from client to server and back. The highlighted features are essential in verifying the obtained results.

In order to visualize the connection on Wireshark, we make 2 main changes by including the “network-animation” module and by later on calling it in Line 77. These changes are highlighted below. The snaps below also indicate the output terminal & the animation obtained:

```

15
16 #include "ns3/applications-module.h"
17 #include "ns3/core-module.h"
18 #include "ns3/internet-module.h"
19 #include "ns3/network-module.h"
20 #include "ns3/point-to-point-module.h"
21 #include "ns3/netanim-module.h"
22
73 ApplicationContainer clientApps = echoClient.Install(nodes.Get(0));
74 clientApps.Start(Seconds(2.0));
75 clientApps.Stop(Seconds(10.0));
76
77 AnimationInterface anim("snehas_myfirst.xml");
78 pointToPoint.EnablePcapAll("snehas_myfirst");
79
80 Simulator::Run();
81 Simulator::Destroy();
82 return 0;
83 }

```

Fig. 5. Modifications to get Wireshark Animations

```

sneha@sneha-VirtualBox:~/Desktop/ns3/ns-3.42$ ./ns3 run snehas_myfirst
[0/2] Re-checking globbed directories...
[2/2] Linking CXX executable /home/sne.../scratch/ns3.42-snehas_myfirst-default
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary
At time +2s client sent 256 bytes to 192.168.2.2 port 63
At time +2.00223s server received 256 bytes from 192.168.2.1 port 49153
At time +2.00223s server sent 256 bytes to 192.168.2.1 port 49153
At time +2.00446s client received 256 bytes from 192.168.2.2 port 63
sneha@sneha-VirtualBox:~/Desktop/ns3/ns-3.42$

```

Fig. 6. Output Terminal after NetAnim

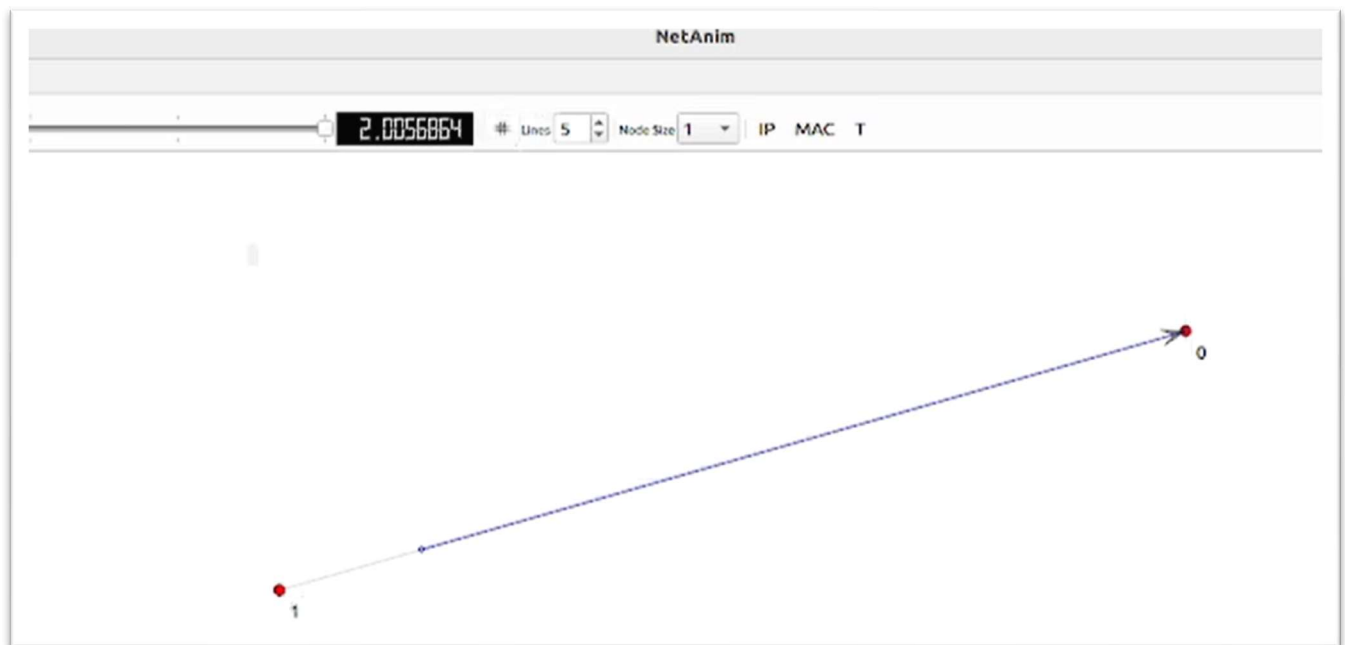


Fig. 7. Point-to-point connection on NetAnim

c) Conclusions:

In this task, a point-to-point network with two nodes, each having one network interface was successfully configured. We established this point-to-point link with a changed data rate of **10 Mbps** and a delay of **2 ms**. IP

addresses were assigned in the range 192.168.2.0/24. Additionally, we set up a UDP Echo Server on the **port 63** with a packet size of **256 bytes**. It is safe to say that this task served as an introduction to network configuration and simulation, providing hands-on experience in setting up nodes, links, and applications in ns-3. Additionally, we gained insight into interpreting packet traces using Wireshark and its animation features.

Task 2:

a) Experimental Setup:

i. The network contains:

- 3 nodes, namely Node 0 through Node 2, connected via a CSMA link in the 1st LAN (192.168.1.0/24)
- 3 nodes, namely Node 3 through Node 5, connected via a CSMA link in the 2nd LAN (192.168.2.0/24)
- 2 nodes, Nodes 2 and 3 are in the point-to-point link, connecting both LANs

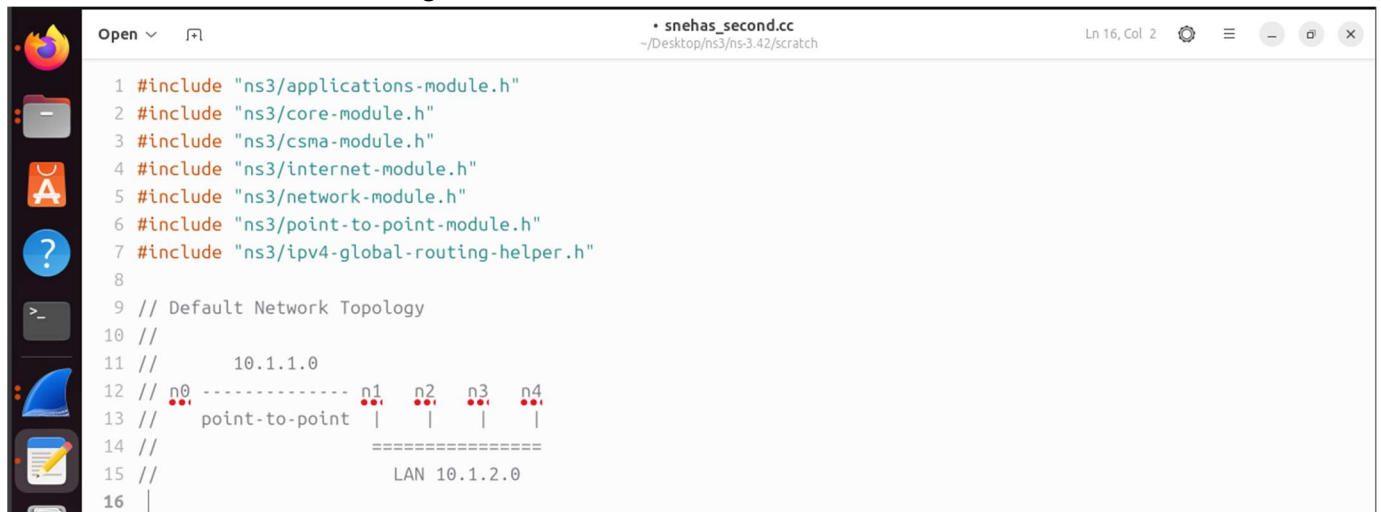
ii. The applications running in the network are:

- UDP Echo Server at Node 1 (192.168.1.1): Listening on port 21
- UDP Echo Client at Node 5 (192.168.2.3): Sends 2 UDP Echo packets to the server at times 4s and 7s

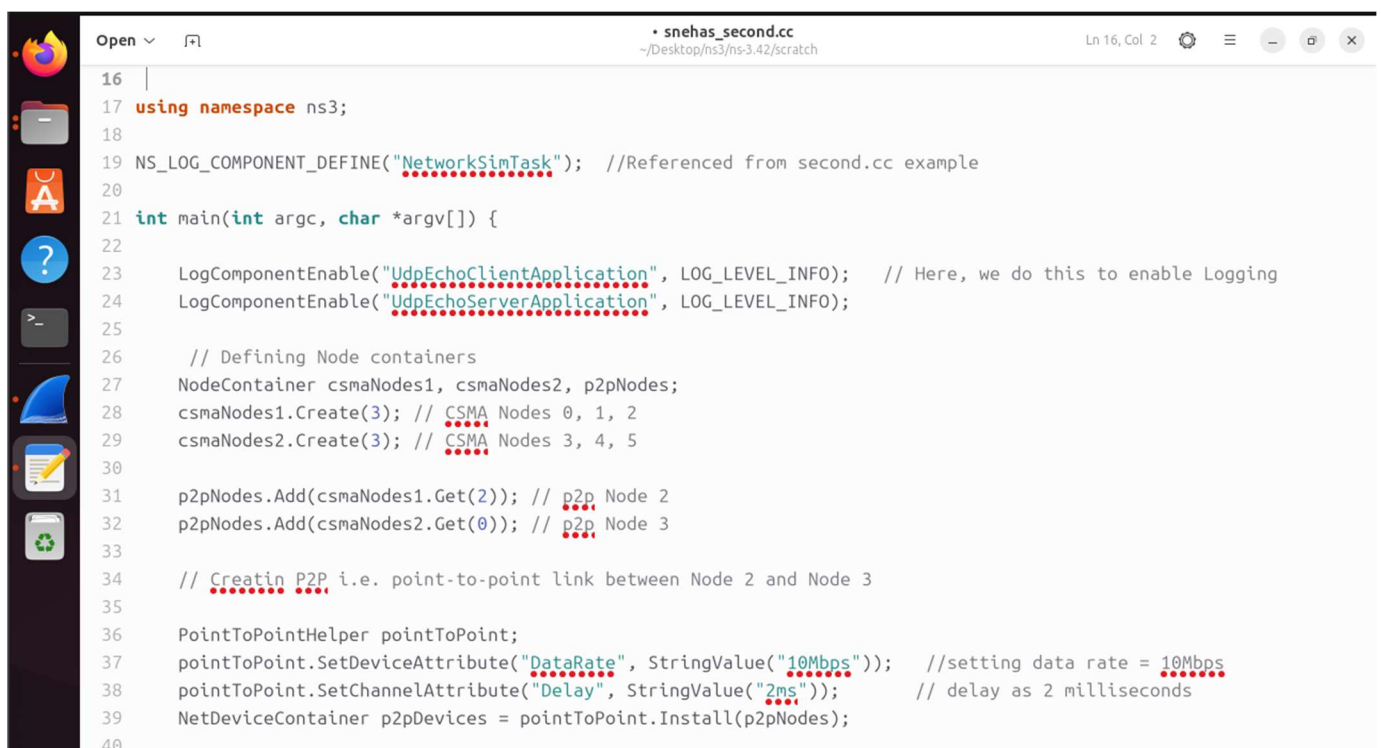
iii. Enable packet tracing with promiscuous mode only in Node 2 in the point-to-point interface and Node 4.

b) Modifications & Results:

The tutorials' basic “second.cc” example inspires this code. However, to reflect on the changes, along with the comments stating the reason behind it are shown in the screenshots below:



```
1 #include "ns3/applications-module.h"
2 #include "ns3/core-module.h"
3 #include "ns3/csma-module.h"
4 #include "ns3/internet-module.h"
5 #include "ns3/network-module.h"
6 #include "ns3/point-to-point-module.h"
7 #include "ns3/ipv4-global-routing-helper.h"
8
9 // Default Network Topology
10 //
11 //      10.1.1.0
12 // n0 ----- n1 n2 n3 n4
13 // point-to-point | | | |
14 //
15 //      LAN 10.1.2.0
16 |
```



```
16 |
17 using namespace ns3;
18
19 NS_LOG_COMPONENT_DEFINE("NetworkSimTask"); //Referenced from second.cc example
20
21 int main(int argc, char *argv[]) {
22
23     LogComponentEnable("UdpEchoClientApplication", LOG_LEVEL_INFO); // Here, we do this to enable Logging
24     LogComponentEnable("UdpEchoServerApplication", LOG_LEVEL_INFO);
25
26     // Defining Node containers
27     NodeContainer csmaNodes1, csmaNodes2, p2pNodes;
28     csmaNodes1.Create(3); // CSMA Nodes 0, 1, 2
29     csmaNodes2.Create(3); // CSMA Nodes 3, 4, 5
30
31     p2pNodes.Add(csmaNodes1.Get(2)); // p2p Node 2
32     p2pNodes.Add(csmaNodes2.Get(0)); // p2p Node 3
33
34     // Creatin P2P i.e. point-to-point link between Node 2 and Node 3
35
36     PointToPointHelper pointToPoint;
37     pointToPoint.SetDeviceAttribute("DataRate", StringValue("10Mbps")); //setting data rate = 10Mbps
38     pointToPoint.SetChannelAttribute("Delay", StringValue("2ms")); // delay as 2 milliseconds
39     NetDeviceContainer p2pDevices = pointToPoint.Install(p2pNodes);
40 }
```

```
Open ▾ | snehas_second.cc | Ln 67, Col 1
~/Desktop/ns3/ns-3.42/scratch

41 // CSMA setup for first LAN (192.168.1.0/24)
42 CsmHelper csma1;
43 csma1.SetChannelAttribute("DataRate", StringValue("100Mbps")); //setting data rate = 100Mbps
44 csma1.SetChannelAttribute("Delay", TimeValue(MicroSeconds(10))); // delay as 10 microseconds
45 NetDeviceContainer csmaDevices1 = csma1.Install(csmaNodes1);
46
47
48 // CSMA setup for second LAN (192.168.2.0/24)
49 CsmHelper csma2;
50 csma2.SetChannelAttribute("DataRate", StringValue("100Mbps")); //same settings as above
51 csma2.SetChannelAttribute("Delay", TimeValue(MicroSeconds(10)));
52 NetDeviceContainer csmaDevices2 = csma2.Install(csmaNodes2);
53
54 // Installing network stacks for the 2 csma nodes
55
56 InternetStackHelper stack;
57 stack.Install(csmaNodes1);
58 stack.Install(csmaNodes2);
59
60 Ipv4AddressHelper address; // Assigning IP addresses
61 address.SetBase("192.168.1.0", "255.255.255.0"); // Assigning addresses to CSMA1 (192.168.1.0/24)
62 Ipv4InterfaceContainer csmaInterfaces1 = address.Assign(csmaDevices1);
63
64 address.SetBase("192.168.2.0", "255.255.255.0"); // Assigning addresses to CSMA2 (192.168.2.0/24)
65 Ipv4InterfaceContainer csmaInterfaces2 = address.Assign(csmaDevices2);
66
67
```

```
Open ▾ | snehas_second.cc | Ln 94, Col 26
~/Desktop/ns3/ns-3.42/scratch

67 address.SetBase("192.168.3.0", "255.255.255.0"); // Assigning addresses to the Point-to-Point link (192.168.3.0/24)
68 Ipv4InterfaceContainer p2pInterfaces = address.Assign(p2pDevices);
69
70 // Setting up the UDP Echo Server on Node 1
71 UdpEchoServerHelper echoServer(21); // also set the port to 21
72 ApplicationContainer serverApps = echoServer.Install(csmaNodes1.Get(1)); // for Node 1
73 serverApps.Start(Seconds(1.0));
74 serverApps.Stop(Seconds(10.0));
75
76 // Now setting up the UDP Echo Client on Node 5
77 UdpEchoClientHelper echoClient(csmaInterfaces1.GetAddress(1), 21); //port 21 on the client side as well
78 echoClient.SetAttribute("MaxPackets", UintegerValue(2));
79 echoClient.SetAttribute("Interval", TimeValue(Seconds(3.0)));
80 echoClient.SetAttribute("PacketSize", UintegerValue(1024));
81
82 ApplicationContainer clientApps = echoClient.Install(csmaNodes2.Get(2)); // for Node 5
83 clientApps.Start(Seconds(4.0));
84 clientApps.Stop(Seconds(10.0));
85
86 // To generate the pcap files, enable packet capture via Wireshark
87 pointToPoint.EnablePcap("task2-p2p", p2pDevices.Get(0), true);
88 csma2.EnablePcap("task2-csma", csmaDevices2.Get(1), true);
89
90 // Enable routing
91 Ipv4GlobalRoutingHelper::PopulateRoutingTables();
92
93 Simulator::Run();
94 Simulator::Destroy();
95 return 0;
```


We build this scratch and get the code running, fix any errors, and then get it finalized to get the output as shown below:

```

sneha@sneha-VirtualBox: ~/Desktop/ns3/ns-3.42
tap-device-creator-default\" -D__LINUX__ -I/home/sneha/Desktop/ns3/ns-3.42/build/include -Os -g -DNDEBUG -std=c++20 -fPI
E -fno-semantic-interposition -fdiagnostics-color=always -Wall -Wpedantic -MD -MT scratch/CMakeFiles/scratch_snehas_se
cond.dir/snehas_second.cc.o -MF scratch/CMakeFiles/scratch_snehas_second.dir/snehas_second.cc.o.d -o scratch/CMakeFiles/
scratch_snehas_second.dir/snehas_second.cc.o -c /home/sneha/Desktop/ns3/ns-3.42/scratch/snehas_second.cc
/home/sneha/Desktop/ns3/ns-3.42/scratch/snehas_second.cc: In function 'int main(int, char**)':
/home/sneha/Desktop/ns3/ns-3.42/scratch/snehas_second.cc:59:50: error: 'Microseconds' was not declared in this scope
   59 |         csma1.SetChannelAttribute("Delay", TimeValue(Microseconds(10)));
      |
ninja: build stopped: subcommand failed.
sneha@sneha-VirtualBox:~/Desktop/ns3/ns-3.42$ ./ns3 run scratch/snehas_second
[0/2] Re-checking globbed directories...
[2/2] Linking CXX executable /home/sne...d/scratch/ns3.42-snehas_second-default
At time +4s client sent 1024 bytes to 192.168.1.2 port 21
At time +4.0161s server received 1024 bytes from 192.168.2.3 port 49153
At time +4.0161s server sent 1024 bytes to 192.168.2.3 port 49153
At time +4.0292s client received 1024 bytes from 192.168.1.2 port 21
At time +7s client sent 1024 bytes to 192.168.1.2 port 21
At time +7.00303s server received 1024 bytes from 192.168.2.3 port 49153
At time +7.00303s server sent 1024 bytes to 192.168.2.3 port 49153
At time +7.00607s client received 1024 bytes from 192.168.1.2 port 21

```

Fig. 8. Output Terminal

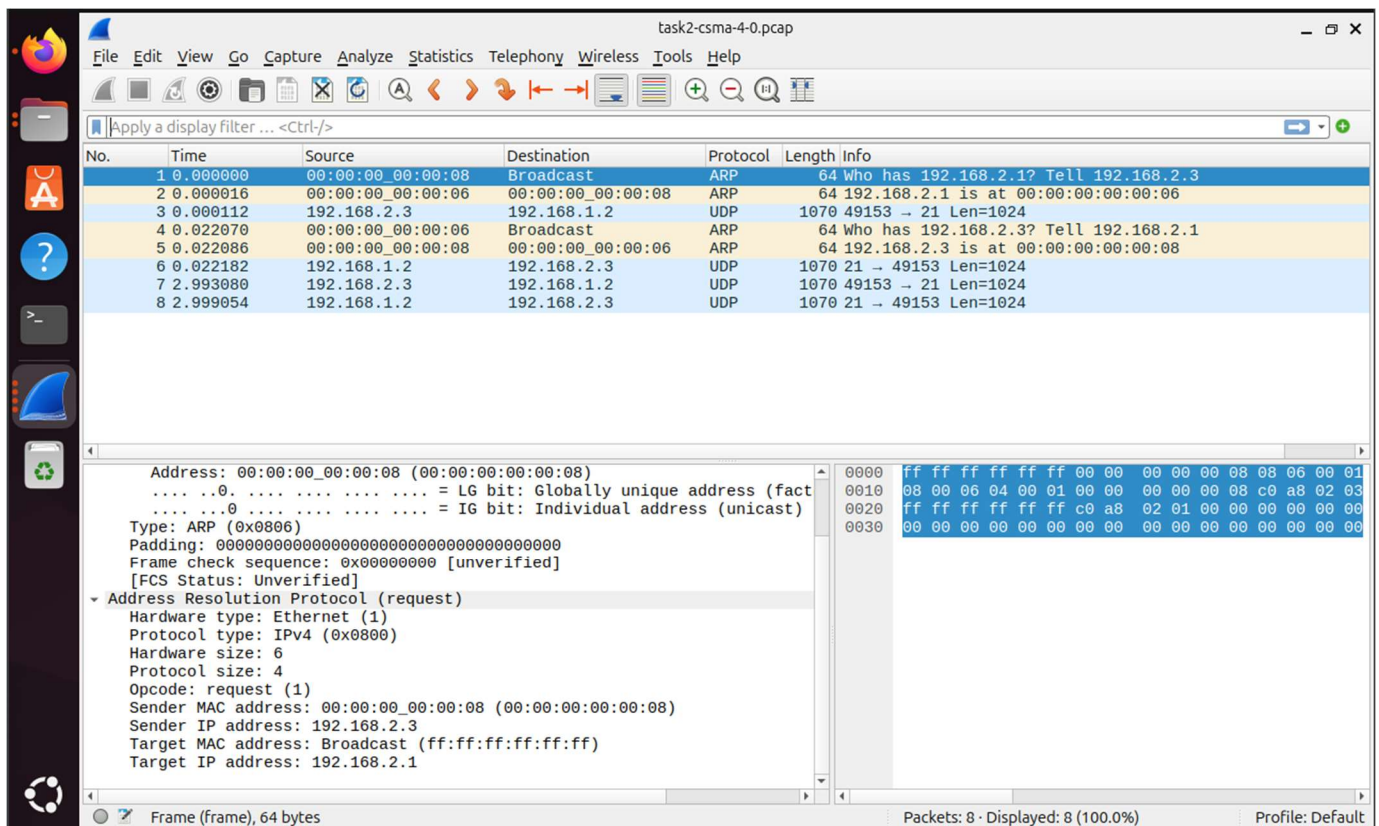


Fig. 9. 4-0.Pcap capture for the Task 2

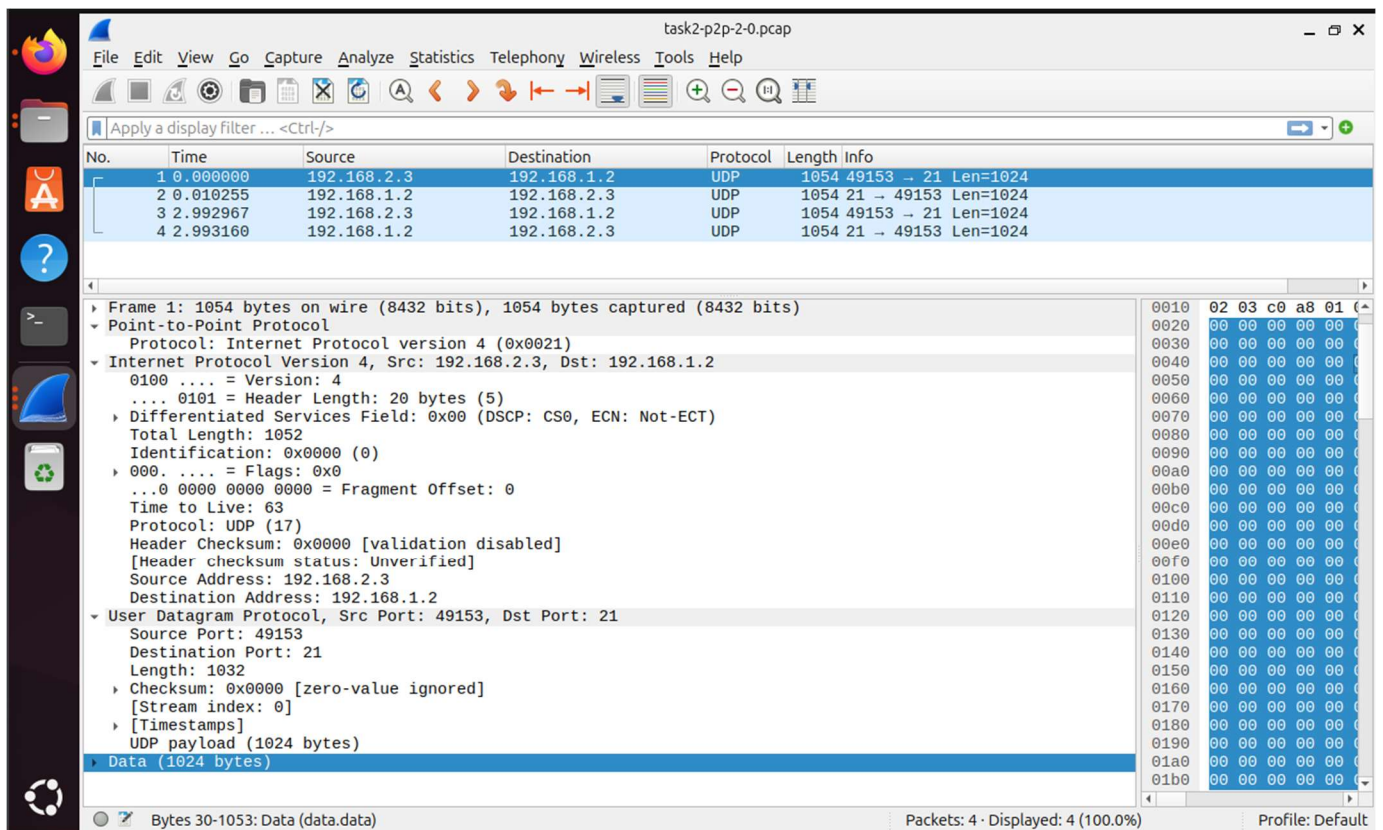


Fig. 10. 2-0.Pcap capture for the Task 2

c) Conclusions:

In this 2nd task, we modified the diagram by considering the first shared bus and the 2nd shared CSMA bus by connecting with a point-to-point node. We modified the value on **port 21** of the UDP Echo Client at Node 5.

It also Sends 2 UDP Echo packets to the server at times 4s and 7s Enable packet tracing only in Nodes 2 and 4. This task provided me with hands-on experience with the ns3 simulator and gained some knowledge and skills regarding how to connect multiple clients and servers using point-to-point node devices.

The output and simulation show expected behavior for both the **UDP Echo Client-Server** interaction and the underlying **network topology** involving two LANs connected via a point-to-point link. The packet transmission and reception times match the configuration, demonstrating that your simulation setup is functioning as expected with proper routing, timing, and packet handling.

Observed Network Behavior:

1. UDP Echo Communication:

- ✓ The client located at **192.168.2.3 (Node 5)** successfully sent two packets of **1024 bytes** to the server at **192.168.1.2 (Node 1)** at two different times: **4s** and **7s**.
- ✓ The packets were routed correctly through the network, passing across the **point-to-point link** and through both CSMA LANs.

2. Packet Reception:

- ✓ At **4.0161s**, the server (**Node 1**) received the first packet, and at **7.00303s**, the second packet was received.
- ✓ The packet sizes, being consistently **1024 bytes**, indicate that the payload was transmitted without fragmentation or issues.

3. Port Information:

- ✓ The server was listening on **port 21**, and the client sent packets from a dynamically allocated port (49153).
- ✓ The communication was correctly set up for **UDP**, as no connection establishment phase (like in TCP) is required.