PRACTICAL RECORD

of

ADVANCED COMPUTER NETWORKS LABORTATORY

(CSPE-352)

ACADEMIC YEAR: 2020-21

SUBMITTED BY:

RAJAN KATARIA
18103076
CSE / 6TH SEMESTER
GROUP: G4

SUBMITTED TO:

Dr. KUNWAR PAL
ASSISTANT PROFESSOR
CSE DEPARTMENT

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
DR. B.R. AMBEDKAR NATIONAL INSTITUTE OF TECHNOLOGY,
JALANDHAR - 144011

TABLE OF CONTENTS

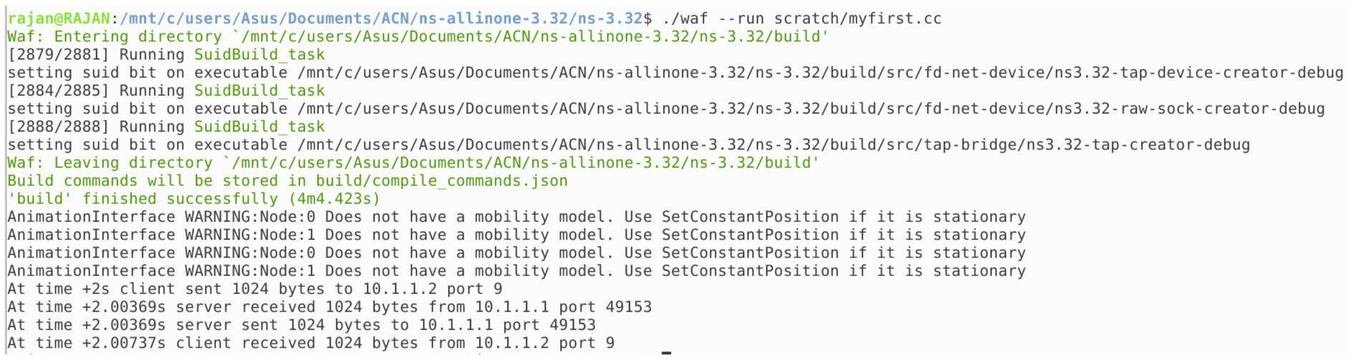
Lab Assignment - 6

6.1 Create a simple topology of two nodes (Node1, Node2) separated by a point-to-point link.

Code:

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/netanim-module.h"

// Default Network Topology
//
//       10.1.1.0
// n0 -------------- n1
//    point-to-point
//

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");

int
main (int argc, char *argv[])
{
  CommandLine cmd (__FILE__);
  cmd.Parse (argc, argv);

  Time::SetResolution (Time::NS);
  LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
  LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

  NodeContainer nodes;
  nodes.Create (2);

  PointToPointHelper pointToPoint;
  pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
  pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));

  NetDeviceContainer devices;
  devices = pointToPoint.Install (nodes);
  InternetStackHelper stack;
  stack.Install (nodes);

  Ipv4AddressHelper address;
  address.SetBase ("10.1.1.0", "255.255.255.0");

  Ipv4InterfaceContainer interfaces = address.Assign (devices);

  UdpEchoServerHelper echoServer (9);

  ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));
  serverApps.Start (Seconds (1.0));
  serverApps.Stop (Seconds (10.0));

  UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
  echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
  echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
  echoClient.SetAttribute ("PacketSize", UintegerValue (1024));

  ApplicationContainer clientApps = echoClient.Install(nodes.Get (0));
  clientApps.Start (Seconds (2.0));
  clientApps.Stop (Seconds (10.0));

  pointToPoint.EnablePcapAll("channel0");

  AnimationInterface anim("myfirst.xml");
  anim.SetConstantPosition(nodes.Get(0),10.0,10.0);
  anim.SetConstantPosition(nodes.Get(1),60.0,30.0);

  AsciiTraceHelper ascii;
  pointToPoint.EnableAsciiAll(ascii.CreateFileStream("p2p.tr"));

  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

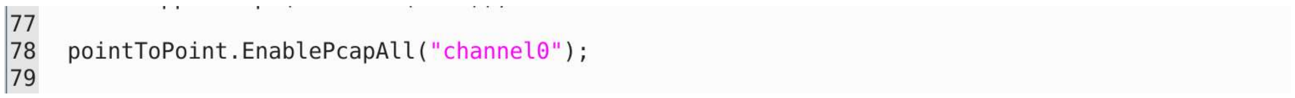
Terminal Output:



```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ ./waf --run scratch/myfirst.cc
Waf: Entering directory `/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build'
[2879/2881] Running SuidBuild_task
setting suid bit on executable /mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build/src/fd-net-device/ns3.32-tap-device-creator-debug
[2884/2885] Running SuidBuild_task
setting suid bit on executable /mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build/src/fd-net-device/ns3.32-raw-sock-creator-debug
[2888/2888] Running SuidBuild_task
setting suid bit on executable /mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build/src/tap-bridge/ns3.32-tap-creator-debug
Waf: Leaving directory `/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (4m4.423s)
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 1024 bytes to 10.1.1.2 port 9
At time +2.00369s server received 1024 bytes from 10.1.1.1 port 49153
At time +2.00369s server sent 1024 bytes to 10.1.1.1 port 49153
At time +2.00737s client received 1024 bytes from 10.1.1.2 port 9
```

a. Create pcap file for each node.

The below written command has been used in the above program o create pcap files for all the nodes.
pointToPoint.EnablePcapAll("channel0");

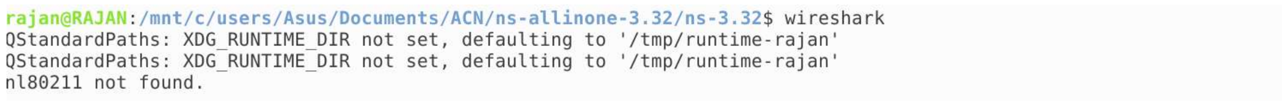


```

  pointToPoint.EnablePcapAll("channel0");

```

This will create two .pcap files, channel0-0-0.pcap and channel0-0-1.pcap.

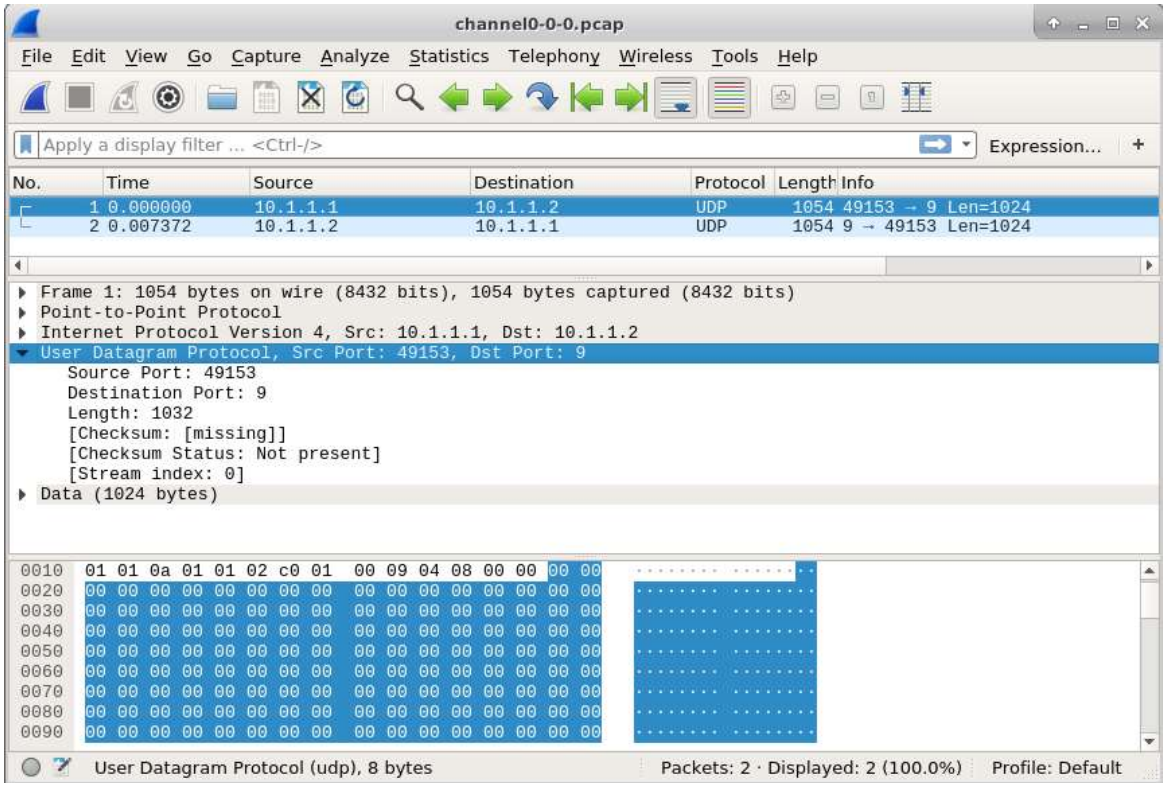
b. Analyse pcap file via Wireshark and tcpdump.

To analyse the pcap files using Wireshark, write wireshark in the terminal, and press enter (as shown below).

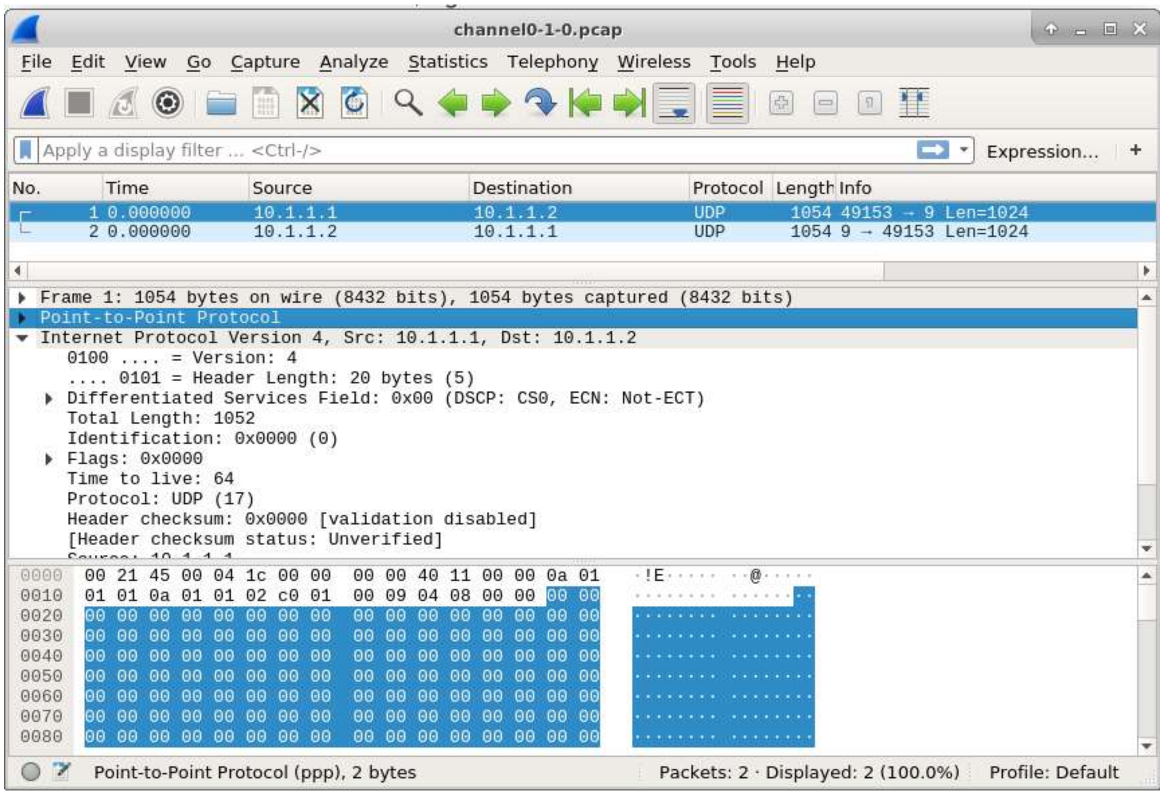


```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ wireshark
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-rajan'
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-rajan'
nl80211 not found.
```

The GUI for Wireshark will open. After that click File > Open File. And, then choose the file from the directory, and press enter.
The Wireshark window will show you different analysis of the respective pcap file, which includes Frame, Point-To-Point Protocol, Internet Protocol, UDP details, and data as shown for both the pcap files below.

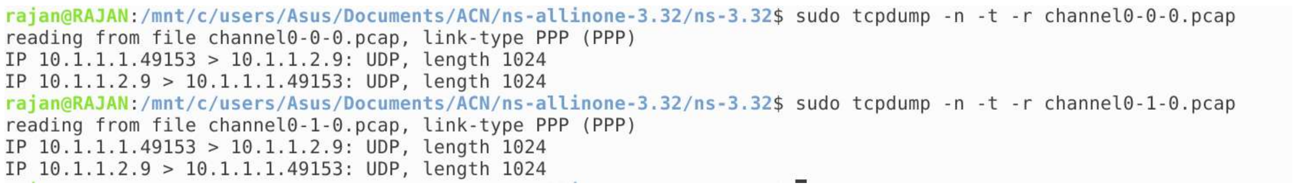

channel0-0-0.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time       Source     Destination  Protocol  Length  Info
  1  0.000000   10.1.1.1   10.1.1.2     UDP       1054    49153 → 9 Len=1024
  2  0.007372   10.1.1.2   10.1.1.1     UDP       1054    9 → 49153 Len=1024

▸ Frame 1: 1054 bytes on wire (8432 bits), 1054 bytes captured (8432 bits)
▸ Point-to-Point Protocol
▸ Internet Protocol Version 4, Src: 10.1.1.1, Dst: 10.1.1.2
▾ User Datagram Protocol, Src Port: 49153, Dst Port: 9
    Source Port: 49153
    Destination Port: 9
    Length: 1032
    [Checksum: [missing]]
    [Checksum Status: Not present]
    [Stream index: 0]
▸ Data (1024 bytes)

0010  01 01 0a 01 01 02 c0 01  00 09 04 08 00 00 00 00   ........ ........
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........

User Datagram Protocol (udp), 8 bytes      Packets: 2 · Displayed: 2 (100.0%)      Profile: Default
```


channel0-1-0.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time       Source     Destination  Protocol  Length  Info
  1  0.000000   10.1.1.1   10.1.1.2     UDP       1054    49153 → 9 Len=1024
  2  0.000000   10.1.1.2   10.1.1.1     UDP       1054    9 → 49153 Len=1024

▸ Frame 1: 1054 bytes on wire (8432 bits), 1054 bytes captured (8432 bits)
▸ Point-to-Point Protocol
▾ Internet Protocol Version 4, Src: 10.1.1.1, Dst: 10.1.1.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1052
    Identification: 0x0000 (0)
  ▸ Flags: 0x0000
    Time to live: 64
    Protocol: UDP (17)
    Header checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]

0000  00 21 45 00 04 1c 00 00  00 00 40 11 00 00 0a 01   .!E..... ..@.....
0010  01 01 0a 01 01 02 c0 01  00 09 04 08 00 00 00 00   ........ ........
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........

Point-to-Point Protocol (ppp), 2 bytes      Packets: 2 · Displayed: 2 (100.0%)      Profile: Default
```

Before the graphical user interface of Wireshark, the pcap files were analysed using tcpdump command as shown below:

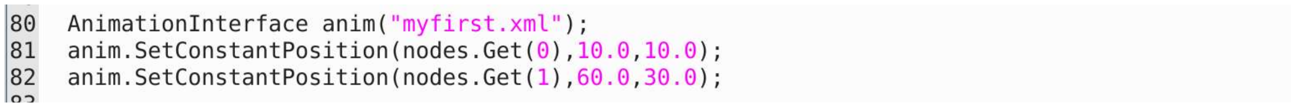


```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ sudo tcpdump -n -t -r channel0-0-0.pcap
reading from file channel0-0-0.pcap, link-type PPP (PPP)
IP 10.1.1.1.49153 > 10.1.1.2.9: UDP, length 1024
IP 10.1.1.2.9 > 10.1.1.1.49153: UDP, length 1024
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ sudo tcpdump -n -t -r channel0-1-0.pcap
reading from file channel0-1-0.pcap, link-type PPP (PPP)
IP 10.1.1.1.49153 > 10.1.1.2.9: UDP, length 1024
IP 10.1.1.2.9 > 10.1.1.1.49153: UDP, length 1024
```

<u>Note:</u> If some error comes, try using sudo in front of tcpdump while writing the command on the terminal.

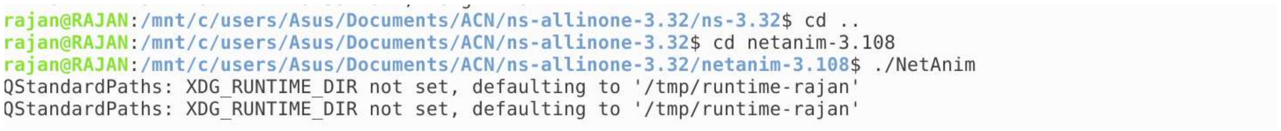
c. Present the node structure and working using Network Animator.

If you want to analyse the node structure using animation, in NetAnim (Network Animator), you need to make xml file for your C++ code in ns-3.
This can be formed using the below written code lines in end of the C++ program as shown. The arguments of SetConstantPosition function show the coordinates of nodes to be shown on the grid in the Network Animator.



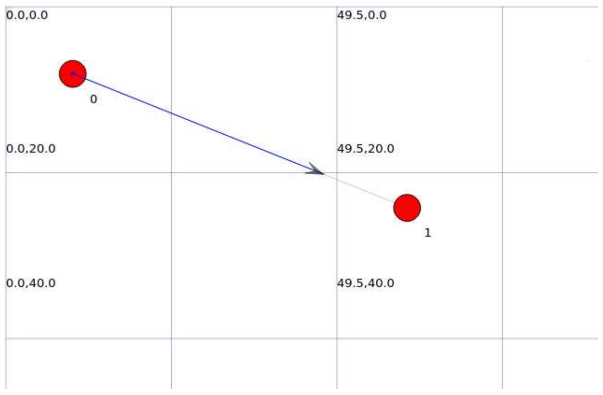
```
AnimationInterface anim("myfirst.xml");
anim.SetConstantPosition(nodes.Get(0),10.0,10.0);
anim.SetConstantPosition(nodes.Get(1),60.0,30.0);
```

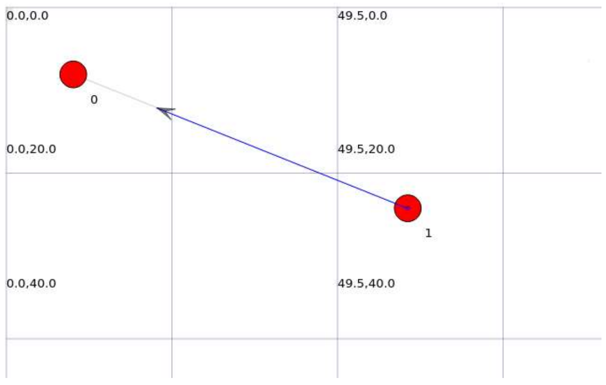
Now to run xml file of your C++ program in NetAnim, follow the below written steps, i.e., go in the netanim-3.108 directory, and write ./NetAnim command as shown:



```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ cd ..
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32$ cd netanim-3.108
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/netanim-3.108$ ./NetAnim
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-rajan'
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-rajan'
```

The NetAnim GUI will open, just select your xml file from the directory, and press play button. The animation will play. The screenshots of node 0 (client) sending packet to the server and server sending acknowledgement back to the client are shown below.





d. Create Ascii Trace file and execute analysis with Tracemetrics.

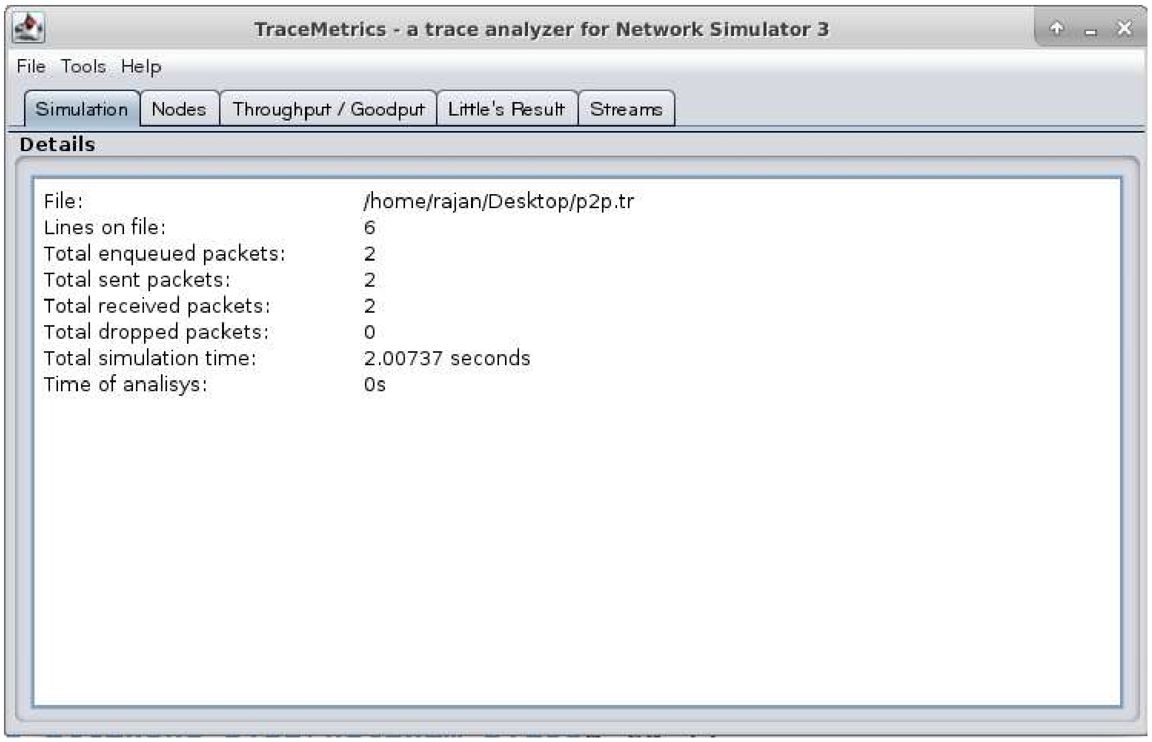
The ASCII trace file is made using the below mentioned command:

```

AsciiTraceHelper ascii;
pointToPoint.EnableAsciiAll(ascii.CreateFileStream("p2p.tr"));

```

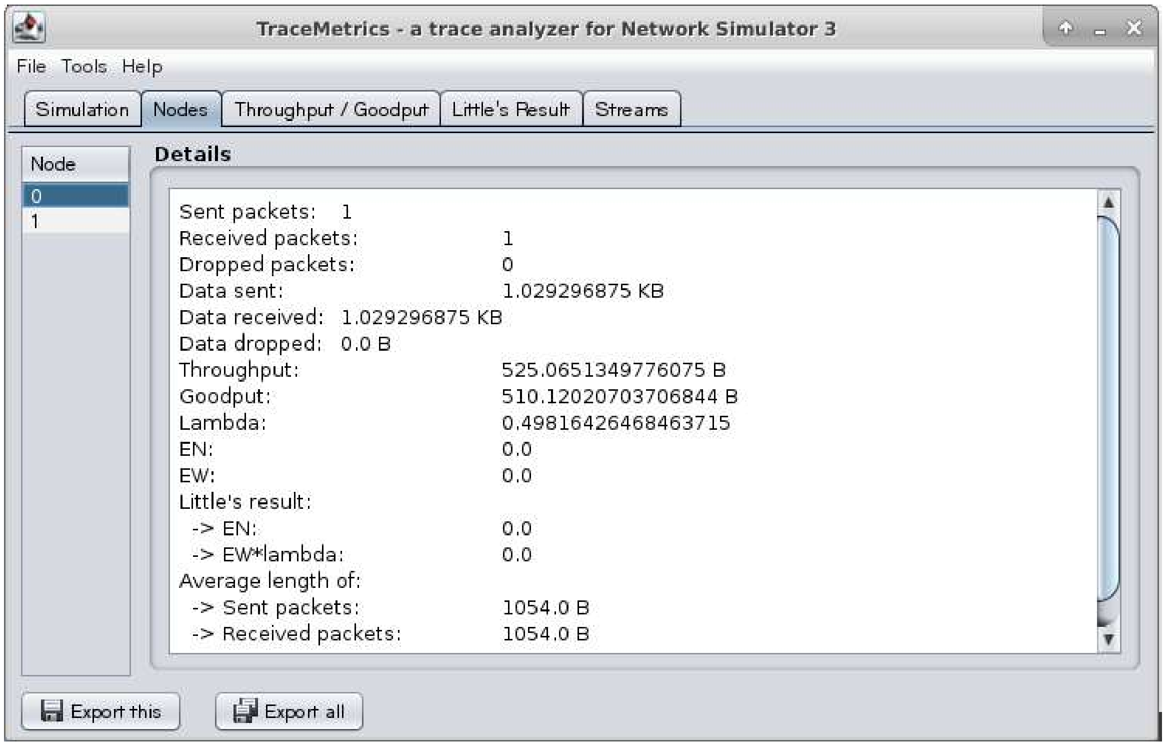
To run TraceMetrics - trace analyzer, run the following command in the directory where you have unzipped/extracted the tracemetrics.zip file.

```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32$ java -jar tracemetrics.jar
TCP size: 0
```

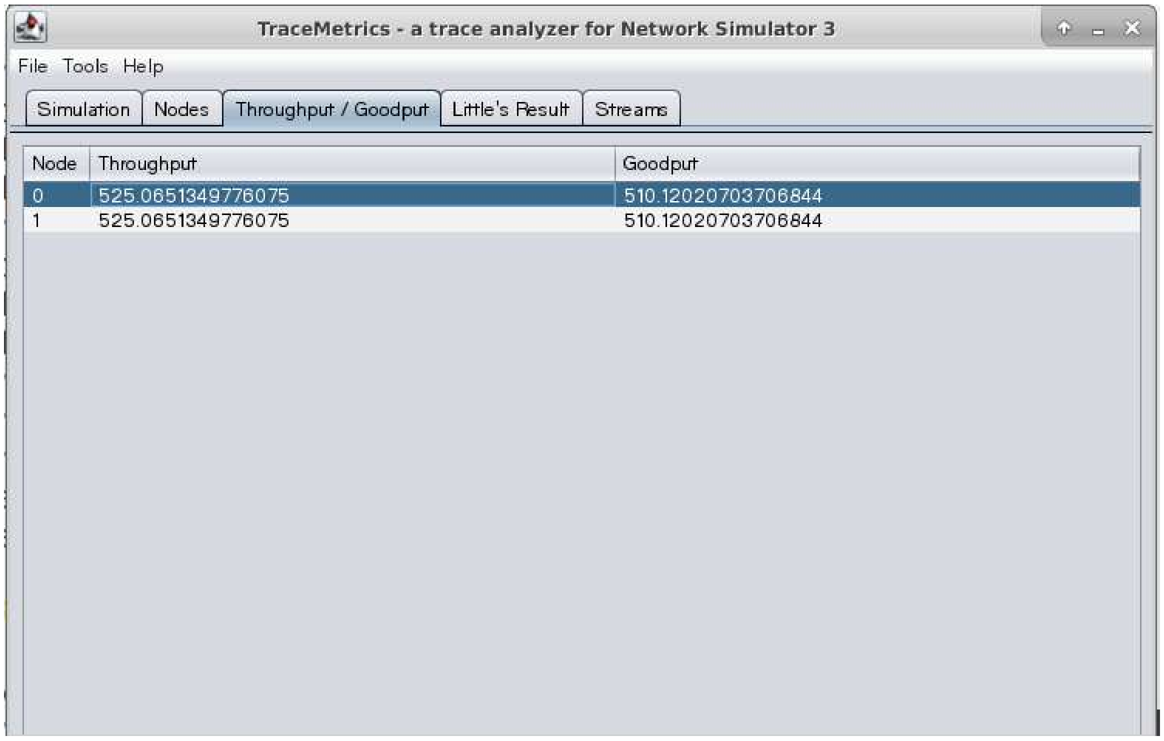
The GUI of TraceMetrics - a trace analyzer for NS3 will open, you will have to select the trace file created by you in the program using File > Open from the Menu bar. And then, all the details of Simulation, Nodes, Throughput/Goodput Little's Result, and Streams will be available in the trace analyzer. You can view that. The screenshots for the above file are attached.



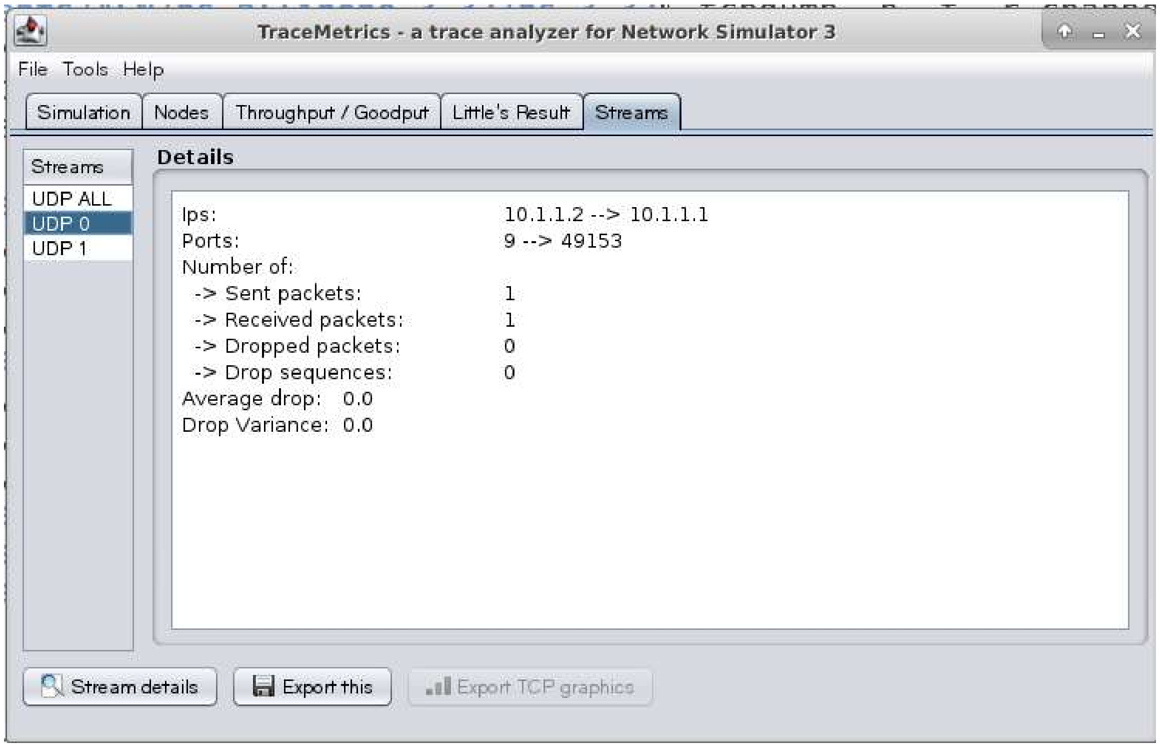
```
File:                        /home/rajan/Desktop/p2p.tr
Lines on file:               6
Total enqueued packets:      2
Total sent packets:          2
Total received packets:      2
Total dropped packets:       0
Total simulation time:       2.00737 seconds
Time of analisys:            0s
```



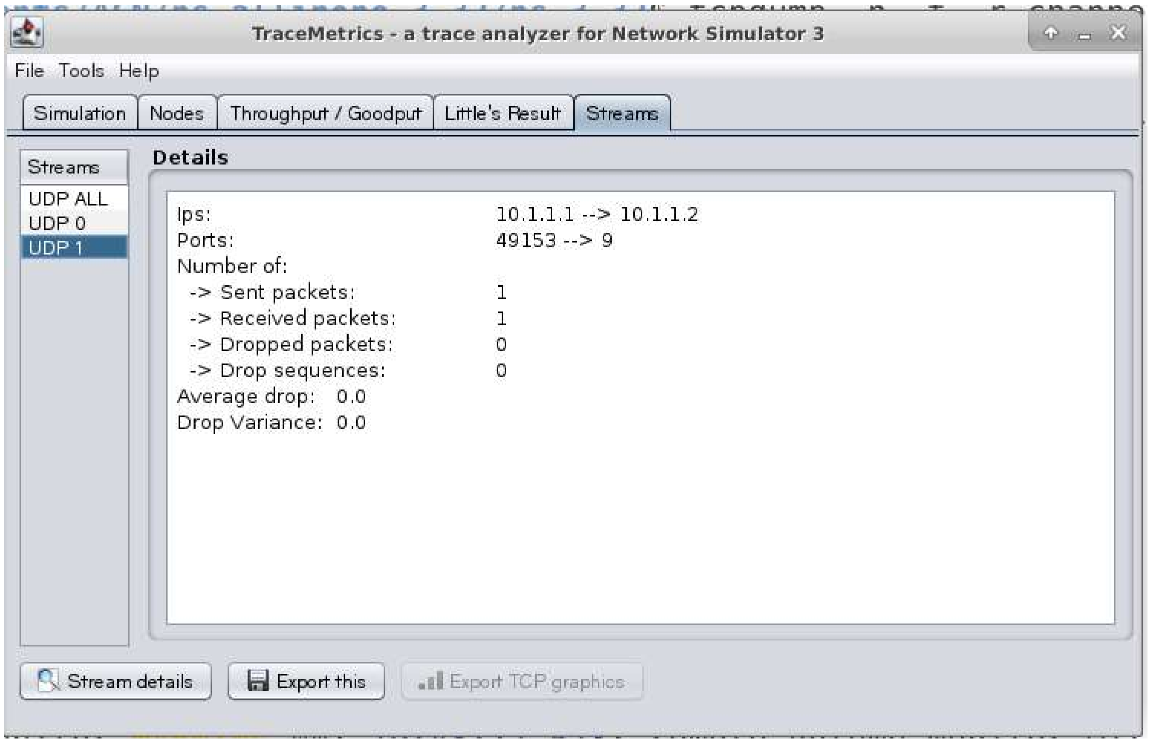
```
Sent packets:     1
Received packets:            1
Dropped packets:             0
Data sent:                   1.029296875 KB
Data received:  1.029296875 KB
Data dropped:   0.0 B
Throughput:                  525.0651349776075 B
Goodput:                     510.12020703706844 B
Lambda:                      0.49816426468463715
EN:                          0.0
EW:                          0.0
Little's result:
  -> EN:                     0.0
  -> EW*lambda:              0.0
Average length of:
  -> Sent packets:           1054.0 B
  -> Received packets:       1054.0 B
```



Node	Throughput	Goodput
0	525.0651349776075	510.12020703706844
1	525.0651349776075	510.12020703706844



```
Ips:                        10.1.1.2 --> 10.1.1.1
Ports:                      9 --> 49153
Number of:
  -> Sent packets:          1
  -> Received packets:      1
  -> Dropped packets:       0
  -> Drop sequences:        0
Average drop:   0.0
Drop Variance:  0.0
```



```
Ips:                        10.1.1.1 --> 10.1.1.2
Ports:                      49153 --> 9
Number of:
  -> Sent packets:          1
  -> Received packets:      1
  -> Dropped packets:       0
  -> Drop sequences:        0
Average drop:   0.0
Drop Variance:  0.0
```

6.2 Create a star topology of 7 nodes (One node will act as server and the rest will act as a client) separated by a point-to-point link using client-server Architecture.

Code:

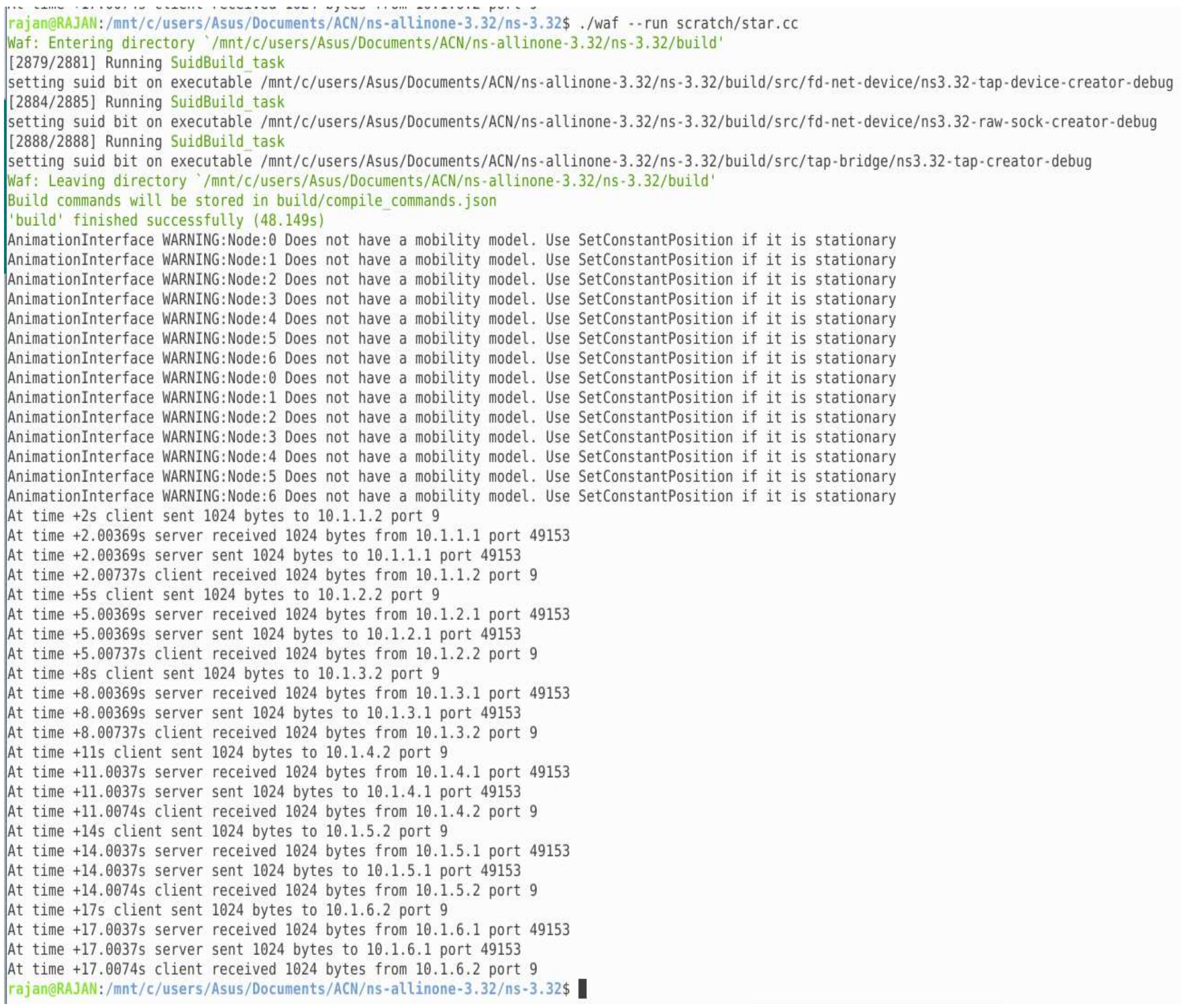
```
 #include "ns3/core-module.h"
 #include "ns3/network-module.h"
 #include "ns3/netanim-module.h"
 #include "ns3/internet-module.h"
 #include "ns3/point-to-point-module.h"
 #include "ns3/applications-module.h"
 #include "ns3/point-to-point-layout-module.h"

 // Network Topology
 // Star Toloplogy with 1 server and 6 hosts
 // Point-to-point connections with UDP

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("FirstTopology");

int main (int argc, char *argv[])
{
  CommandLine cmd (__FILE__);
  cmd.Parse (argc, argv);

  Time::SetResolution (Time::NS);
  LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
  LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

  NodeContainer nodes;
  nodes.Create (7);

  /*******************************************************/

  PointToPointHelper pointToPoint;
  pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
  pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
  PointToPointHelper pointToPoint1;
  pointToPoint1.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
  pointToPoint1.SetChannelAttribute ("Delay", StringValue ("2ms"));
  PointToPointHelper pointToPoint2;
  pointToPoint2.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
  pointToPoint2.SetChannelAttribute ("Delay", StringValue ("2ms"));
  PointToPointHelper pointToPoint3;
  pointToPoint3.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
```

```
pointToPoint3.SetChannelAttribute ("Delay", StringValue ("2ms"));
PointToPointHelper pointToPoint4;
pointToPoint4.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
pointToPoint4.SetChannelAttribute ("Delay", StringValue ("2ms"));
PointToPointHelper pointToPoint5;
pointToPoint5.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
pointToPoint5.SetChannelAttribute ("Delay", StringValue ("2ms"));

/*********************************************************/

NetDeviceContainer devices;
devices = pointToPoint.Install (nodes.Get(1), nodes.Get(0));
NetDeviceContainer devices1;
devices1 = pointToPoint1.Install (nodes.Get(2), nodes.Get(0));
NetDeviceContainer devices2;
devices2 = pointToPoint2.Install (nodes.Get(3), nodes.Get(0));
NetDeviceContainer devices3;
devices3 = pointToPoint3.Install (nodes.Get(4), nodes.Get(0));
NetDeviceContainer devices4;
devices4 = pointToPoint4.Install (nodes.Get(5), nodes.Get(0));
NetDeviceContainer devices5;
devices5 = pointToPoint5.Install (nodes.Get(6), nodes.Get(0));

/*********************************************************/

InternetStackHelper stack;
stack.Install (nodes);

/*********************************************************/

Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4AddressHelper address1;
address1.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4AddressHelper address2;
address2.SetBase ("10.1.3.0", "255.255.255.0");
Ipv4AddressHelper address3;
address3.SetBase ("10.1.4.0", "255.255.255.0");
Ipv4AddressHelper address4;
address4.SetBase ("10.1.5.0", "255.255.255.0");
Ipv4AddressHelper address5;
address5.SetBase ("10.1.6.0", "255.255.255.0");

/*********************************************************/

Ipv4InterfaceContainer interfaces = address.Assign (devices);
Ipv4InterfaceContainer interfaces1 = address1.Assign (devices1);
```

```
Ipv4InterfaceContainer interfaces2 = address2.Assign (devices2);
Ipv4InterfaceContainer interfaces3 = address3.Assign (devices3);
Ipv4InterfaceContainer interfaces4 = address4.Assign (devices4);
Ipv4InterfaceContainer interfaces5 = address5.Assign (devices5);

/*******************************************************/

UdpEchoServerHelper echoServer (9);
ApplicationContainer serverApps = echoServer.Install (nodes.Get (0));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (50.0));

/*******************************************************/

UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UintegerValue (1024));
UdpEchoClientHelper echoClient1 (interfaces1.GetAddress (1), 9);
echoClient1.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient1.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient1.SetAttribute ("PacketSize", UintegerValue (1024));
UdpEchoClientHelper echoClient2 (interfaces2.GetAddress (1), 9);
echoClient2.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient2.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient2.SetAttribute ("PacketSize", UintegerValue (1024));
UdpEchoClientHelper echoClient3 (interfaces3.GetAddress (1), 9);
echoClient3.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient3.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient3.SetAttribute ("PacketSize", UintegerValue (1024));
UdpEchoClientHelper echoClient4 (interfaces4.GetAddress (1), 9);
echoClient4.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient4.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient4.SetAttribute ("PacketSize", UintegerValue (1024));
UdpEchoClientHelper echoClient5 (interfaces5.GetAddress (1), 9);
echoClient5.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient5.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient5.SetAttribute ("PacketSize", UintegerValue (1024));

/*******************************************************/

ApplicationContainer clientApps = echoClient.Install(nodes.Get (1));
ApplicationContainer clientApps1 = echoClient1.Install(nodes.Get (2));
ApplicationContainer clientApps2 = echoClient2.Install(nodes.Get (3));
ApplicationContainer clientApps3 = echoClient3.Install(nodes.Get (4));
ApplicationContainer clientApps4 = echoClient4.Install(nodes.Get (5));
ApplicationContainer clientApps5 = echoClient5.Install(nodes.Get (6));
```

```
 /*********************************************************/

 clientApps.Start (Seconds (2.0));
 clientApps.Stop (Seconds (4.0));
 clientApps1.Start (Seconds (5.0));
 clientApps1.Stop (Seconds (7.0));
 clientApps2.Start (Seconds (8.0));
 clientApps2.Stop (Seconds (10.0));
 clientApps3.Start (Seconds (11.0));
 clientApps3.Stop (Seconds (13.0));
 clientApps4.Start (Seconds (14.0));
 clientApps4.Stop (Seconds (16.0));
 clientApps5.Start (Seconds (17.0));
 clientApps5.Stop (Seconds (19.0));

 /*********************************************************/

 pointToPoint.EnablePcapAll("channel01");

 /*********************************************************/

 AnimationInterface anim("mystar.xml");
 anim.SetConstantPosition(nodes.Get(0),49.0,37.5);
 anim.SetConstantPosition(nodes.Get(1),0.0,0.0);
 anim.SetConstantPosition(nodes.Get(2),0.0,75.0);
 anim.SetConstantPosition(nodes.Get(3),40.0,75.0);
 anim.SetConstantPosition(nodes.Get(4),49.0,0.0);
 anim.SetConstantPosition(nodes.Get(5),98.0,0.0);
 anim.SetConstantPosition(nodes.Get(6),98.0,75.0);

 /*********************************************************/

 AsciiTraceHelper ascii;
 pointToPoint.EnableAsciiAll(ascii.CreateFileStream("p2pstar0.tr"));

 /*********************************************************/

 Simulator::Run ();
 Simulator::Destroy ();
 return 0;

}
```

Terminal Output:



```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ ./waf --run scratch/star.cc
Waf: Entering directory `/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build'
[2879/2881] Running SuidBuild_task
setting suid bit on executable /mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build/src/fd-net-device/ns3.32-tap-device-creator-debug
[2884/2885] Running SuidBuild_task
setting suid bit on executable /mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build/src/fd-net-device/ns3.32-raw-sock-creator-debug
[2888/2888] Running SuidBuild_task
setting suid bit on executable /mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build/src/tap-bridge/ns3.32-tap-creator-debug
Waf: Leaving directory `/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (48.149s)
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:2 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:5 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:6 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
AnimationInterface WARNING:Node:2 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:5 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:6 Does not have a mobility model. Use SetConstantPosition if it is stationary
At time +2s client sent 1024 bytes to 10.1.1.2 port 9
At time +2.00369s server received 1024 bytes from 10.1.1.1 port 49153
At time +2.00369s server sent 1024 bytes to 10.1.1.1 port 49153
At time +2.00737s client received 1024 bytes from 10.1.1.2 port 9
At time +5s client sent 1024 bytes to 10.1.2.2 port 9
At time +5.00369s server received 1024 bytes from 10.1.2.1 port 49153
At time +5.00369s server sent 1024 bytes to 10.1.2.1 port 49153
At time +5.00737s client received 1024 bytes from 10.1.2.2 port 9
At time +8s client sent 1024 bytes to 10.1.3.2 port 9
At time +8.00369s server received 1024 bytes from 10.1.3.1 port 49153
At time +8.00369s server sent 1024 bytes to 10.1.3.1 port 49153
At time +8.00737s client received 1024 bytes from 10.1.3.2 port 9
At time +11s client sent 1024 bytes to 10.1.4.2 port 9
At time +11.0037s server received 1024 bytes from 10.1.4.1 port 49153
At time +11.0037s server sent 1024 bytes to 10.1.4.1 port 49153
At time +11.0074s client received 1024 bytes from 10.1.4.2 port 9
At time +14s client sent 1024 bytes to 10.1.5.2 port 9
At time +14.0037s server received 1024 bytes from 10.1.5.1 port 49153
At time +14.0037s server sent 1024 bytes to 10.1.5.1 port 49153
At time +14.0074s client received 1024 bytes from 10.1.5.2 port 9
At time +17s client sent 1024 bytes to 10.1.6.2 port 9
At time +17.0037s server received 1024 bytes from 10.1.6.1 port 49153
At time +17.0037s server sent 1024 bytes to 10.1.6.1 port 49153
At time +17.0074s client received 1024 bytes from 10.1.6.2 port 9
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$
```

a. Create pcap file for each node.

The below written command has been used in the above program to create pcap files for all the nodes.

pointToPoint.EnablePcapAll("channel01");

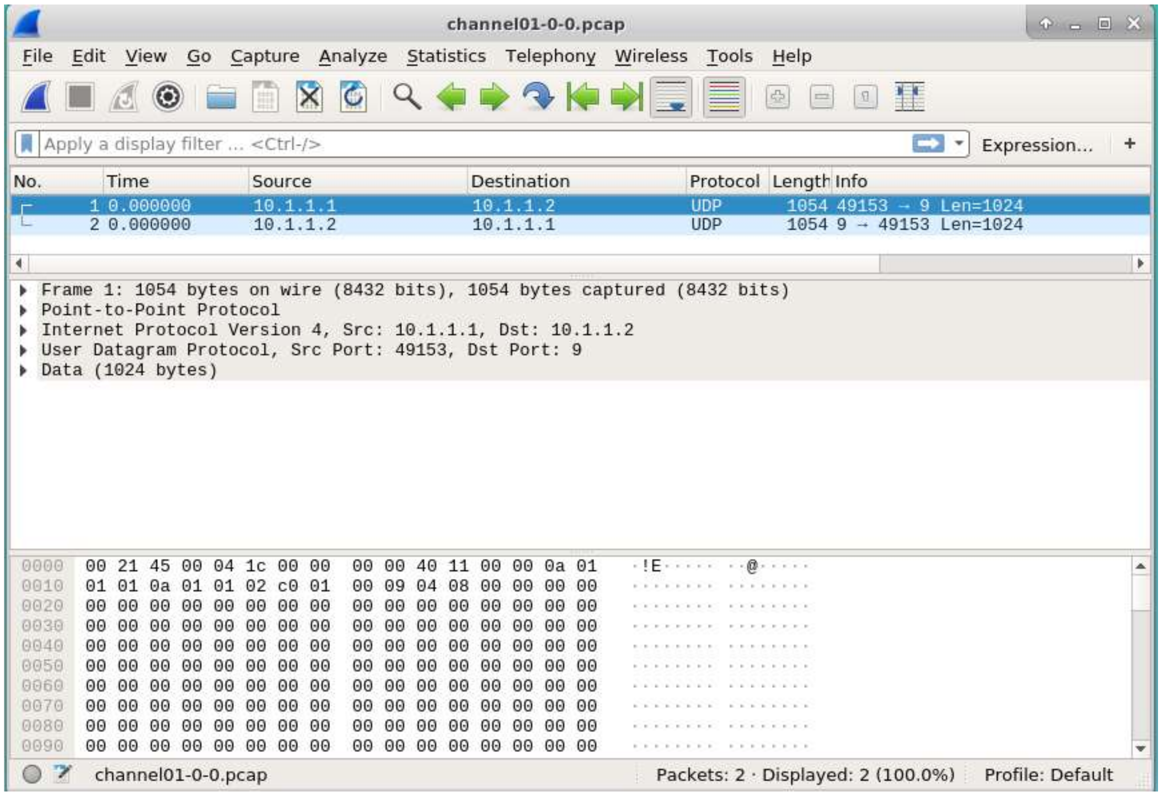
This will create six .pcap files, channel01-0-0.pcap, channel01-0-1.pcap, channel01-0-2.pcap, channel01-0-3.pcap, channel01-0-4.pcap, channel01-0-5.pcap and channel01-0-6.pcap.

b. Analyse pcap file via Wireshark and tcpdump.

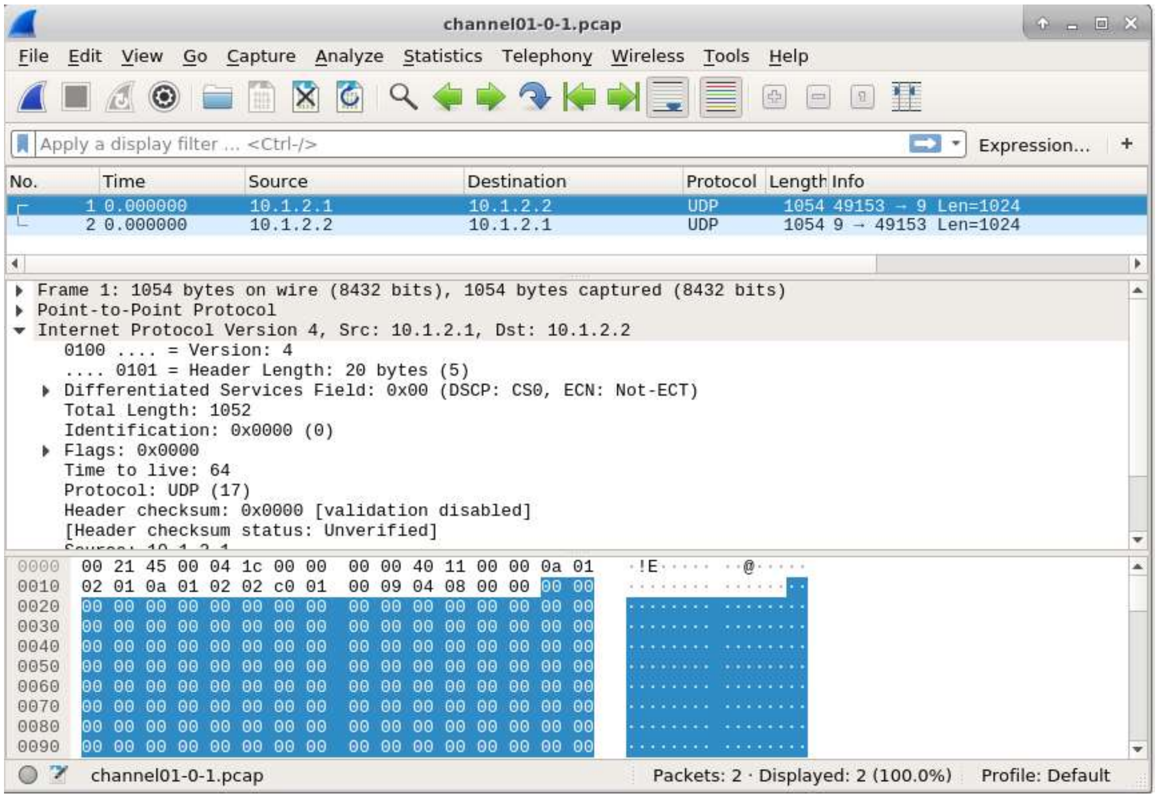
To analyse the pcap files using Wireshark, write wireshark in the terminal, and press enter (as shown below).

```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ wireshark
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-rajan'
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-rajan'
nl80211 not found.
```

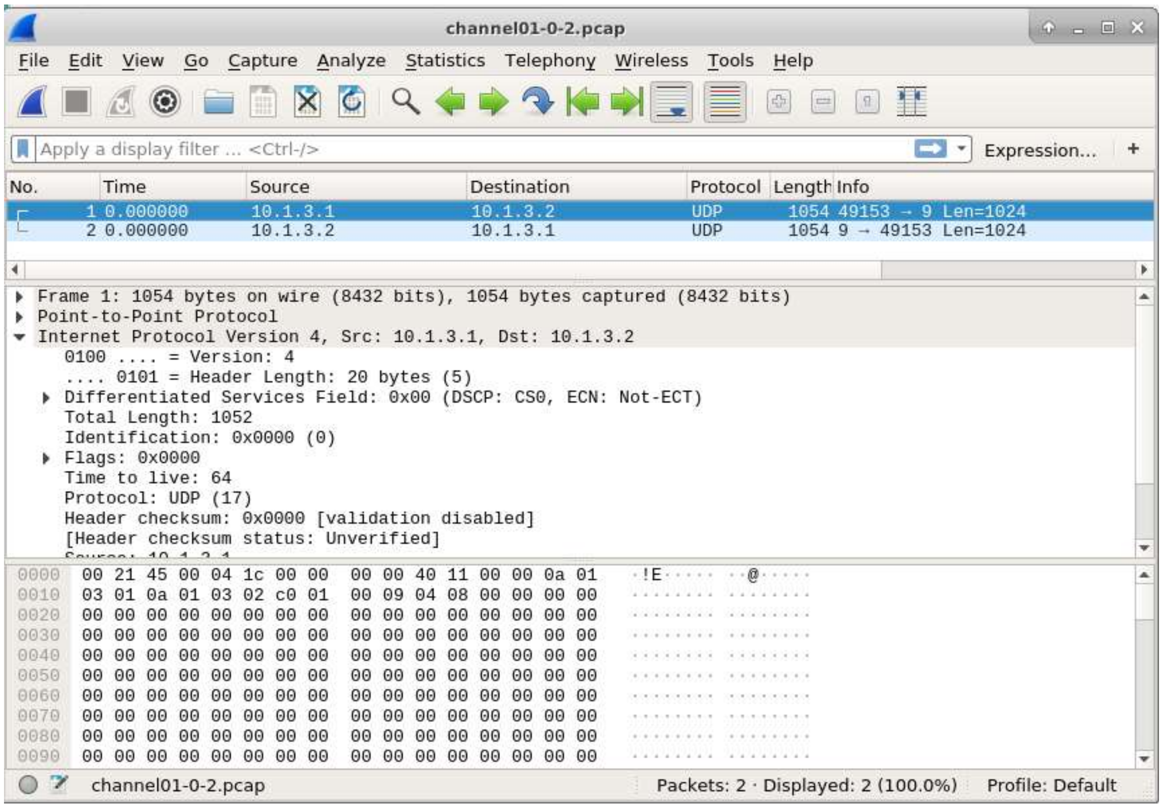
The GUI for Wireshark will open. After that click File > Open File. And, then choose the file from the directory, and press enter.
The Wireshark window will show you different analysis of the respective pcap file, which includes Frame, Point-To-Point Protocol, Internet Protocol, UDP details, and data as shown for both the pcap files below.


channel01-0-0.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

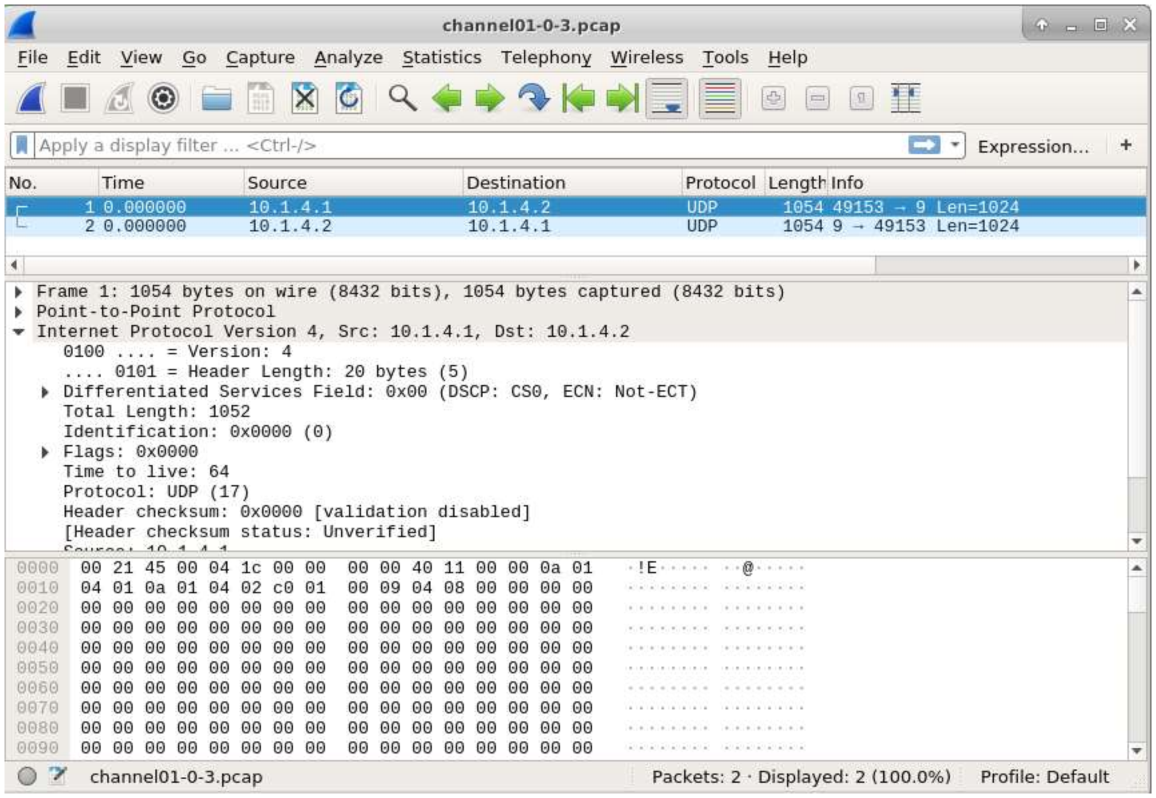
```
No.  Time      Source    Destination  Protocol Length Info
   1 0.000000  10.1.1.1  10.1.1.2     UDP      1054 49153 → 9 Len=1024
   2 0.000000  10.1.1.2  10.1.1.1     UDP      1054 9 → 49153 Len=1024

▸ Frame 1: 1054 bytes on wire (8432 bits), 1054 bytes captured (8432 bits)
▸ Point-to-Point Protocol
▸ Internet Protocol Version 4, Src: 10.1.1.1, Dst: 10.1.1.2
▸ User Datagram Protocol, Src Port: 49153, Dst Port: 9
▸ Data (1024 bytes)

0000  00 21 45 00 04 1c 00 00  00 00 40 11 00 00 0a 01   ·!E·····  ··@·····
0010  01 01 0a 01 01 02 c0 01  00 09 04 08 00 00 00 00   ········  ········
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········

channel01-0-0.pcap          Packets: 2 · Displayed: 2 (100.0%)    Profile: Default
```



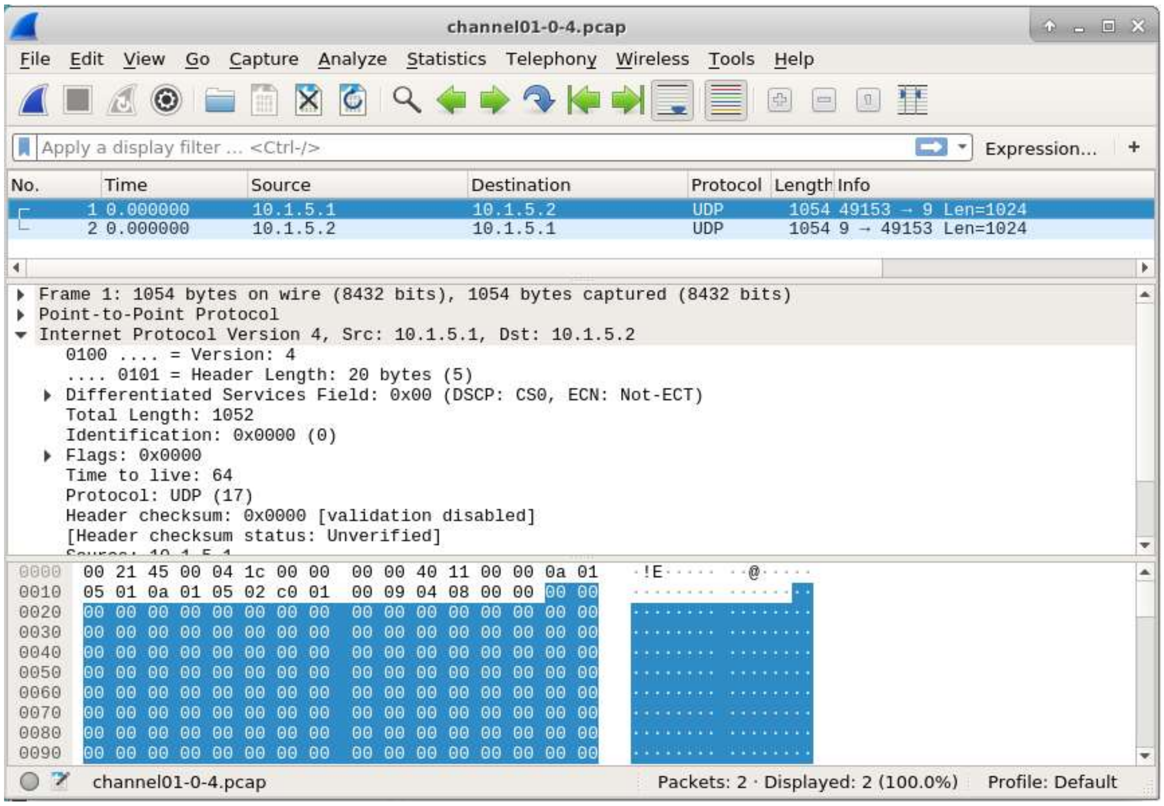
```
channel01-0-1.pcap

No.  Time      Source    Destination  Protocol Length Info
1    0.000000  10.1.2.1  10.1.2.2     UDP      1054   49153 → 9 Len=1024
2    0.000000  10.1.2.2  10.1.2.1     UDP      1054   9 → 49153 Len=1024

▸ Frame 1: 1054 bytes on wire (8432 bits), 1054 bytes captured (8432 bits)
▸ Point-to-Point Protocol
▾ Internet Protocol Version 4, Src: 10.1.2.1, Dst: 10.1.2.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1052
    Identification: 0x0000 (0)
  ▸ Flags: 0x0000
    Time to live: 64
    Protocol: UDP (17)
    Header checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]

0000  00 21 45 00 04 1c 00 00  00 00 40 11 00 00 0a 01   ·!E·····  ··@·····
0010  02 01 0a 01 02 02 c0 01  00 09 04 08 00 00 00 00   ········  ········
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········

channel01-0-1.pcap    Packets: 2 · Displayed: 2 (100.0%)    Profile: Default
```



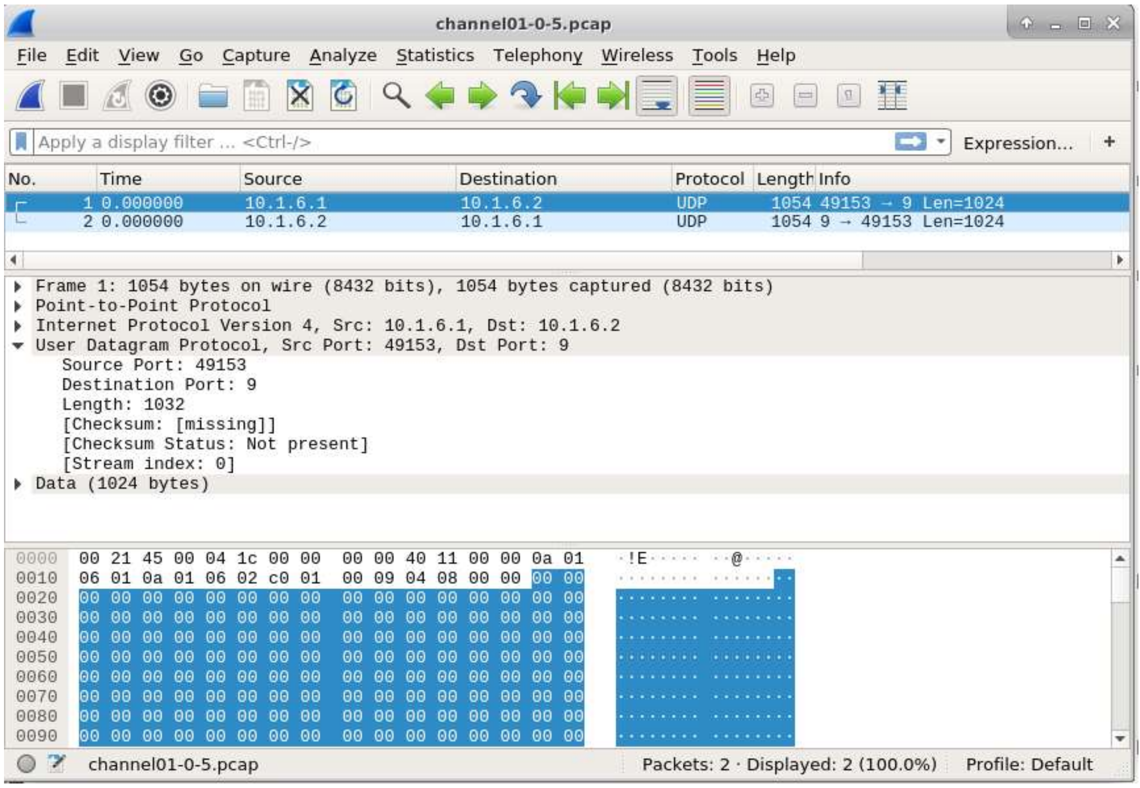
```
channel01-0-2.pcap

No.  Time      Source    Destination  Protocol Length Info
1    0.000000  10.1.3.1  10.1.3.2     UDP      1054   49153 → 9 Len=1024
2    0.000000  10.1.3.2  10.1.3.1     UDP      1054   9 → 49153 Len=1024

▸ Frame 1: 1054 bytes on wire (8432 bits), 1054 bytes captured (8432 bits)
▸ Point-to-Point Protocol
▾ Internet Protocol Version 4, Src: 10.1.3.1, Dst: 10.1.3.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1052
    Identification: 0x0000 (0)
  ▸ Flags: 0x0000
    Time to live: 64
    Protocol: UDP (17)
    Header checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]

0000  00 21 45 00 04 1c 00 00  00 00 40 11 00 00 0a 01   ·!E·····  ··@·····
0010  03 01 0a 01 03 02 c0 01  00 09 04 08 00 00 00 00   ········  ········
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········

channel01-0-2.pcap    Packets: 2 · Displayed: 2 (100.0%)    Profile: Default
```


channel01-0-3.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.   Time       Source      Destination   Protocol Length Info
    1 0.000000   10.1.4.1    10.1.4.2      UDP      1054 49153 → 9 Len=1024
    2 0.000000   10.1.4.2    10.1.4.1      UDP      1054 9 → 49153 Len=1024

▸ Frame 1: 1054 bytes on wire (8432 bits), 1054 bytes captured (8432 bits)
▸ Point-to-Point Protocol
▾ Internet Protocol Version 4, Src: 10.1.4.1, Dst: 10.1.4.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1052
    Identification: 0x0000 (0)
  ▸ Flags: 0x0000
    Time to live: 64
    Protocol: UDP (17)
    Header checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]

0000  00 21 45 00 04 1c 00 00  00 00 40 11 00 00 0a 01   ·!E·····  ··@·····
0010  04 01 0a 01 04 02 c0 01  00 09 04 08 00 00 00 00   ········  ········
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········

channel01-0-3.pcap        Packets: 2 · Displayed: 2 (100.0%)     Profile: Default
```

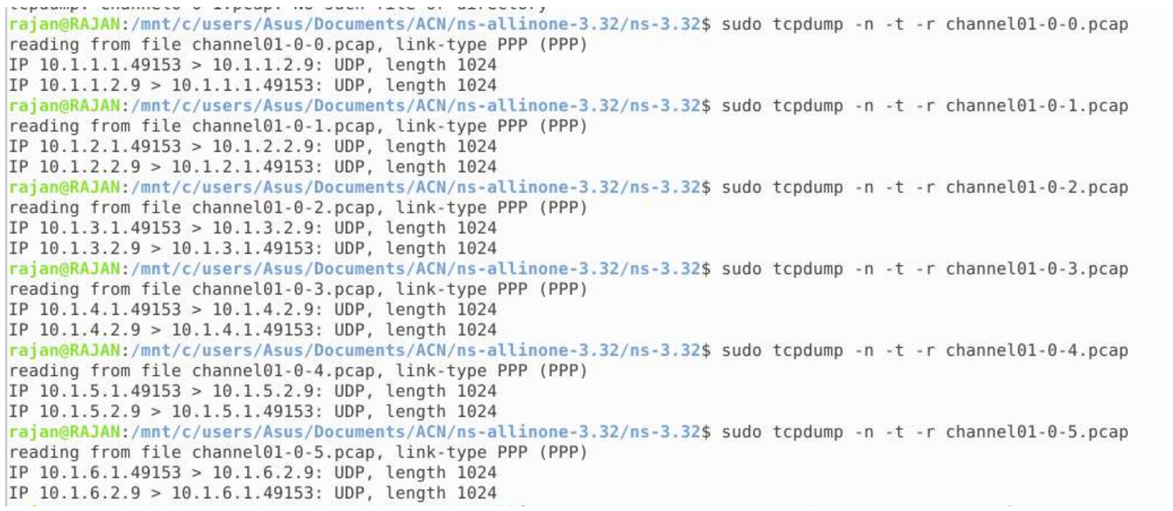

channel01-0-4.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.   Time       Source      Destination   Protocol Length Info
    1 0.000000   10.1.5.1    10.1.5.2      UDP      1054 49153 → 9 Len=1024
    2 0.000000   10.1.5.2    10.1.5.1      UDP      1054 9 → 49153 Len=1024

▸ Frame 1: 1054 bytes on wire (8432 bits), 1054 bytes captured (8432 bits)
▸ Point-to-Point Protocol
▾ Internet Protocol Version 4, Src: 10.1.5.1, Dst: 10.1.5.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1052
    Identification: 0x0000 (0)
  ▸ Flags: 0x0000
    Time to live: 64
    Protocol: UDP (17)
    Header checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]

0000  00 21 45 00 04 1c 00 00  00 00 40 11 00 00 0a 01   ·!E·····  ··@·····
0010  05 01 0a 01 05 02 c0 01  00 09 04 08 00 00 00 00   ········  ········
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········

channel01-0-4.pcap        Packets: 2 · Displayed: 2 (100.0%)     Profile: Default
```


channel01-0-5.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time      Source     Destination  Protocol  Length Info
  1 0.000000   10.1.6.1   10.1.6.2     UDP       1054 49153 → 9 Len=1024
  2 0.000000   10.1.6.2   10.1.6.1     UDP       1054 9 → 49153 Len=1024

▸ Frame 1: 1054 bytes on wire (8432 bits), 1054 bytes captured (8432 bits)
▸ Point-to-Point Protocol
▸ Internet Protocol Version 4, Src: 10.1.6.1, Dst: 10.1.6.2
▾ User Datagram Protocol, Src Port: 49153, Dst Port: 9
    Source Port: 49153
    Destination Port: 9
    Length: 1032
    [Checksum: [missing]]
    [Checksum Status: Not present]
    [Stream index: 0]
▸ Data (1024 bytes)

0000  00 21 45 00 04 1c 00 00  00 00 40 11 00 00 0a 01   ·!E····· ··@·····
0010  06 01 0a 01 06 02 c0 01  00 09 04 08 00 00 00 00   ········ ········
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········

channel01-0-5.pcap        Packets: 2 · Displayed: 2 (100.0%)    Profile: Default
```

Before the graphical user interface of Wireshark, the pcap files were analysed using tcpdump command as shown below:



```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ sudo tcpdump -n -t -r channel01-0-0.pcap
reading from file channel01-0-0.pcap, link-type PPP (PPP)
IP 10.1.1.1.49153 > 10.1.1.2.9: UDP, length 1024
IP 10.1.1.2.9 > 10.1.1.1.49153: UDP, length 1024
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ sudo tcpdump -n -t -r channel01-0-1.pcap
reading from file channel01-0-1.pcap, link-type PPP (PPP)
IP 10.1.2.1.49153 > 10.1.2.2.9: UDP, length 1024
IP 10.1.2.2.9 > 10.1.2.1.49153: UDP, length 1024
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ sudo tcpdump -n -t -r channel01-0-2.pcap
reading from file channel01-0-2.pcap, link-type PPP (PPP)
IP 10.1.3.1.49153 > 10.1.3.2.9: UDP, length 1024
IP 10.1.3.2.9 > 10.1.3.1.49153: UDP, length 1024
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ sudo tcpdump -n -t -r channel01-0-3.pcap
reading from file channel01-0-3.pcap, link-type PPP (PPP)
IP 10.1.4.1.49153 > 10.1.4.2.9: UDP, length 1024
IP 10.1.4.2.9 > 10.1.4.1.49153: UDP, length 1024
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ sudo tcpdump -n -t -r channel01-0-4.pcap
reading from file channel01-0-4.pcap, link-type PPP (PPP)
IP 10.1.5.1.49153 > 10.1.5.2.9: UDP, length 1024
IP 10.1.5.2.9 > 10.1.5.1.49153: UDP, length 1024
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ sudo tcpdump -n -t -r channel01-0-5.pcap
reading from file channel01-0-5.pcap, link-type PPP (PPP)
IP 10.1.6.1.49153 > 10.1.6.2.9: UDP, length 1024
IP 10.1.6.2.9 > 10.1.6.1.49153: UDP, length 1024
```

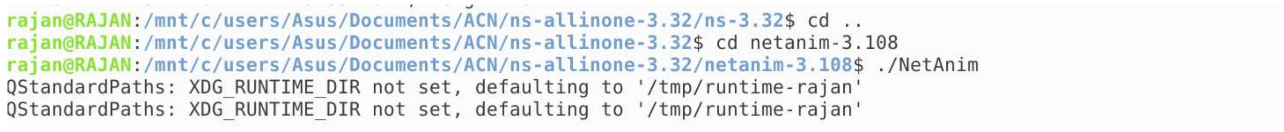
<u>Note:</u> If some error comes, try using sudo in front of tcpdump while writing the command on the terminal.

c. Present the node structure and working using Network Animator.

If you want to analyse the node structure using animation, in NetAnim (Network Animator), you need to make xml file for your C++ code in ns-3.
This can be formed using the below written code lines in end of the C++ program as shown. The arguments of SetConstantPosition function show the coordinates of nodes to be shown on the grid in the Network Animator.

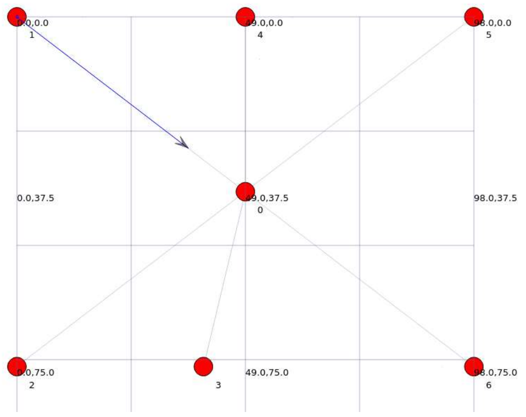
```
AnimationInterface anim("mystar.xml");
anim.SetConstantPosition(nodes.Get(0),49.0,37.5);
anim.SetConstantPosition(nodes.Get(1),0.0,0.0);
anim.SetConstantPosition(nodes.Get(2),0.0,75.0);
anim.SetConstantPosition(nodes.Get(3),40.0,75.0);
anim.SetConstantPosition(nodes.Get(4),49.0,0.0);
anim.SetConstantPosition(nodes.Get(5),98.0,0.0);
anim.SetConstantPosition(nodes.Get(6),98.0,75.0);
```

Now to run xml file of your C++ program in NetAnim, follow the below written steps, i.e., go in the netanim-3.108 directory, and write ./NetAnim command as shown:

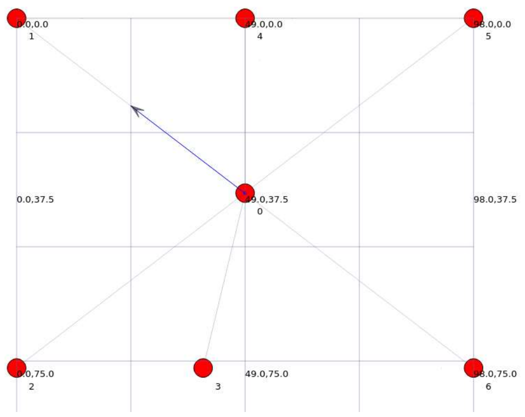


```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32$ cd ..
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32$ cd netanim-3.108
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32/netanim-3.108$ ./NetAnim
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-rajan'
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-rajan'
```

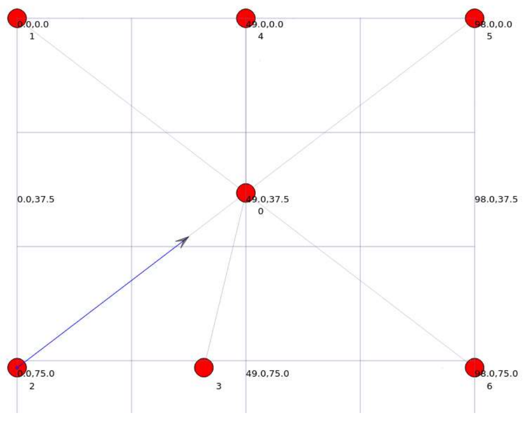
The NetAnim GUI will open, just select your xml file ("mystar.xml" here) from the directory, and press play button. The animation will play. The screenshots of all the 6 clients sending packet to the server and server sending acknowledgement back to each the client are shown below.



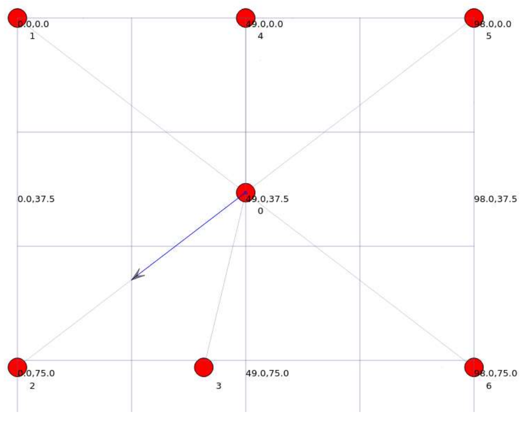
Node 1 sending packet to Node 0



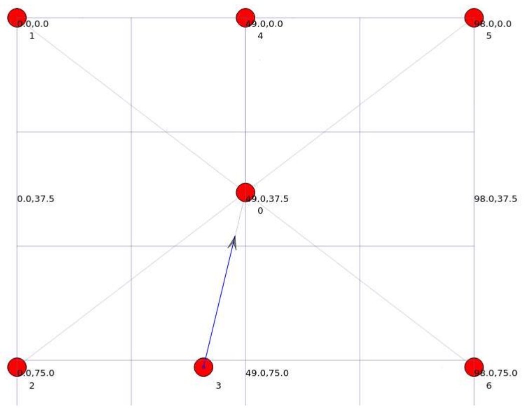
Node 0 sending acknowledgement to Node 1



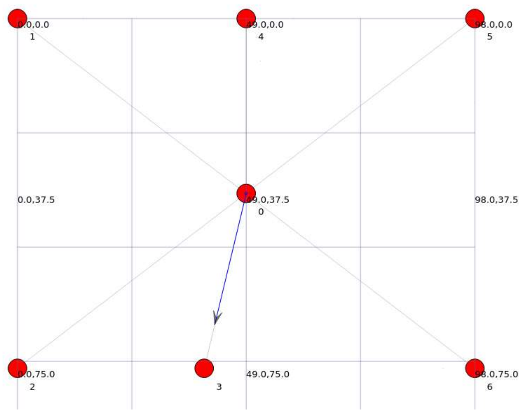
Node 2 sending packet to Node 0



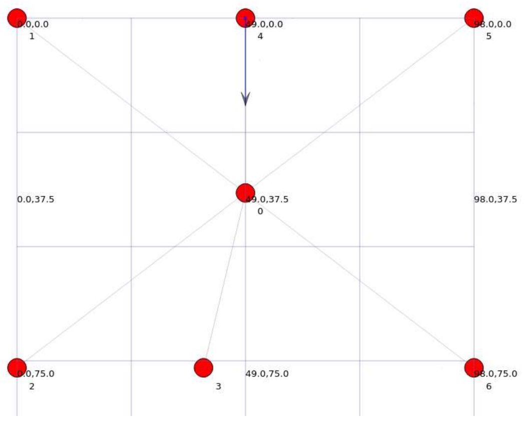
Node 0 sending acknowledgement to Node 2



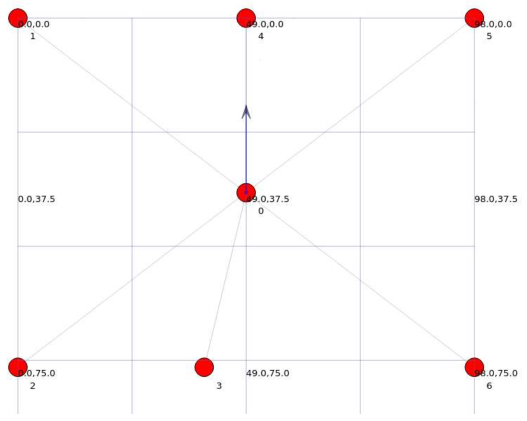
Node 3 sending packet to Node 0



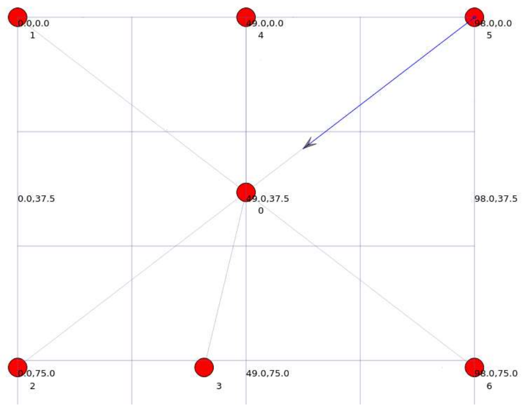
Node 0 sending acknowledgement to Node 3



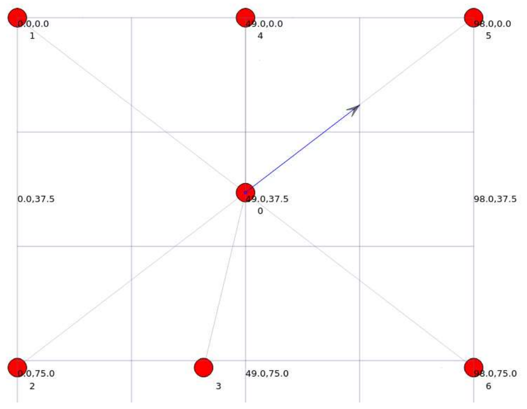
Node 4 sending packet to Node 0



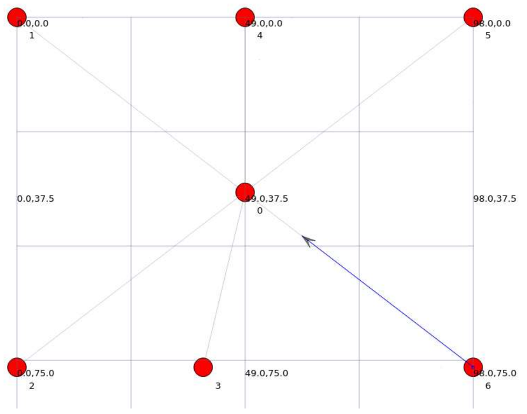
Node 0 sending acknowledgement to Node 4



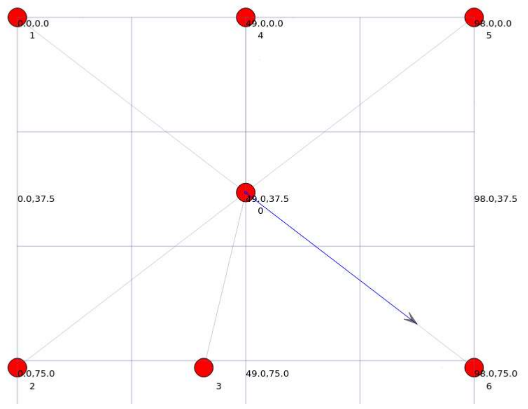
Node 5 sending packet to Node 0



Node 0 sending acknowledgement to Node 5

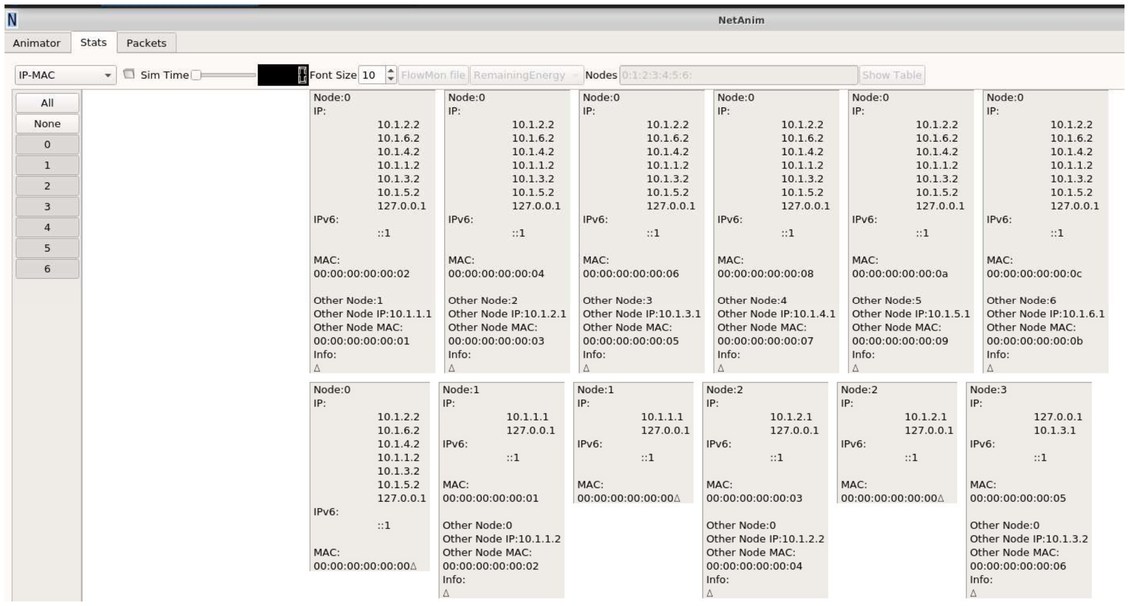


Node 6 sending packet to Node 0



Node 0 sending acknowledgement to Node 6

After Animator tab in NetAnim, we have Stats tab, which tells us the statistics of each of the node as shown in screenshot below:



```
Node:0
IP:
          10.1.2.2
          10.1.6.2
          10.1.4.2
          10.1.1.2
          10.1.3.2
          10.1.5.2
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:02

Other Node:1
Other Node IP:10.1.1.1
Other Node MAC:
00:00:00:00:00:01
Info:
Δ
```

```
Node:0
IP:
          10.1.2.2
          10.1.6.2
          10.1.4.2
          10.1.1.2
          10.1.3.2
          10.1.5.2
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:04

Other Node:2
Other Node IP:10.1.2.1
Other Node MAC:
00:00:00:00:00:03
Info:
Δ
```

```
Node:0
IP:
          10.1.2.2
          10.1.6.2
          10.1.4.2
          10.1.1.2
          10.1.3.2
          10.1.5.2
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:06

Other Node:3
Other Node IP:10.1.3.1
Other Node MAC:
00:00:00:00:00:05
Info:
Δ
```

```
Node:0
IP:
          10.1.2.2
          10.1.6.2
          10.1.4.2
          10.1.1.2
          10.1.3.2
          10.1.5.2
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:08

Other Node:4
Other Node IP:10.1.4.1
Other Node MAC:
00:00:00:00:00:07
Info:
Δ
```

```
Node:0
IP:
          10.1.2.2
          10.1.6.2
          10.1.4.2
          10.1.1.2
          10.1.3.2
          10.1.5.2
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:0a

Other Node:5
Other Node IP:10.1.5.1
Other Node MAC:
00:00:00:00:00:09
Info:
Δ
```

```
Node:0
IP:
          10.1.2.2
          10.1.6.2
          10.1.4.2
          10.1.1.2
          10.1.3.2
          10.1.5.2
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:0c

Other Node:6
Other Node IP:10.1.6.1
Other Node MAC:
00:00:00:00:00:0b
Info:
Δ
```

```
Node:0
IP:
          10.1.2.2
          10.1.6.2
          10.1.4.2
          10.1.1.2
          10.1.3.2
          10.1.5.2
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:00Δ
```

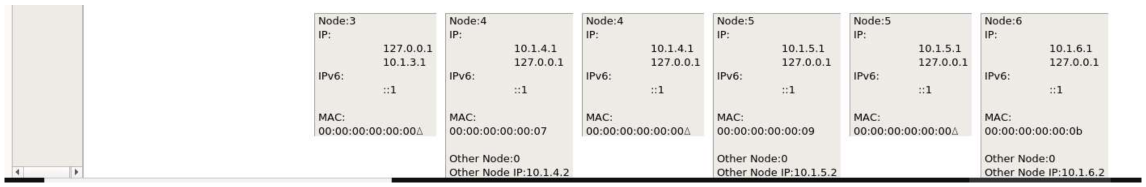
```
Node:1
IP:
          10.1.1.1
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:01

Other Node:0
Other Node IP:10.1.1.2
Other Node MAC:
00:00:00:00:00:02
Info:
Δ
```

```
Node:1
IP:
          10.1.1.1
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:00Δ
```

```
Node:2
IP:
          10.1.2.1
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:03

Other Node:0
Other Node IP:10.1.2.2
Other Node MAC:
00:00:00:00:00:04
Info:
Δ
```

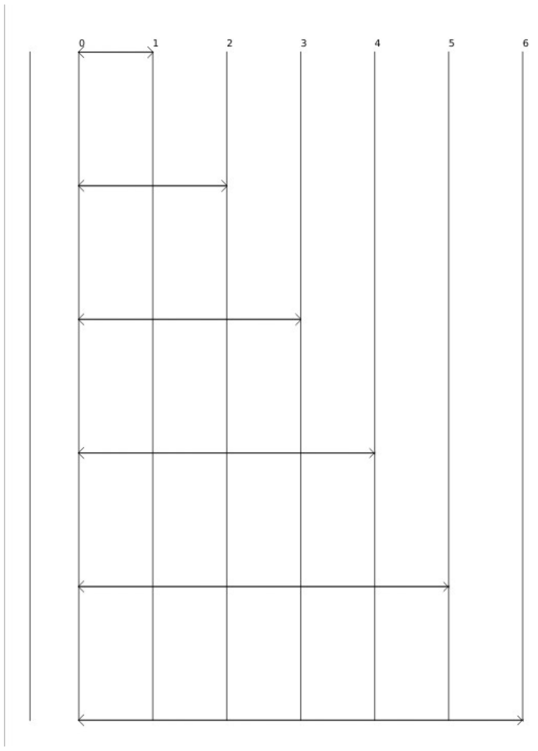
```
Node:2
IP:
          10.1.2.1
          127.0.0.1
IPv6:
          ::1

MAC:
00:00:00:00:00:00Δ
```

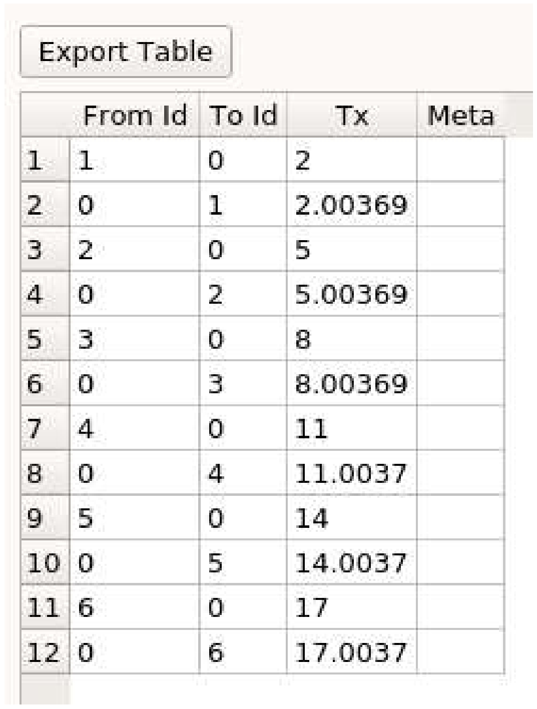
```
Node:3
IP:
          127.0.0.1
          10.1.3.1
IPv6:
          ::1

MAC:
00:00:00:00:00:05

Other Node:0
Other Node IP:10.1.3.2
Other Node MAC:
00:00:00:00:00:06
Info:
Δ
```



```
Node:3              Node:4                    Node:4              Node:5                    Node:5              Node:6
IP:                 IP:                       IP:                 IP:                       IP:                 IP:
        127.0.0.1           10.1.4.1                  10.1.4.1            10.1.5.1                  10.1.5.1            10.1.6.1
        10.1.3.1            127.0.0.1                 127.0.0.1           127.0.0.1                 127.0.0.1           127.0.0.1
IPv6:               IPv6:                     IPv6:               IPv6:                     IPv6:               IPv6:
        ::1                 ::1                       ::1                 ::1                       ::1                 ::1

MAC:                MAC:                      MAC:                MAC:                      MAC:                MAC:
00:00:00:00:00:00Δ  00:00:00:00:00:07         00:00:00:00:00:00Δ  00:00:00:00:00:09         00:00:00:00:00:00Δ  00:00:00:00:00:0b

                    Other Node:0                                  Other Node:0                                  Other Node:0
                    Other Node IP:10.1.4.2                        Other Node IP:10.1.5.2                        Other Node IP:10.1.6.2
```

Next to the Stats tab, we have Packet analyzer, which has the time diagram and the time table for the packets transferred between the nodes as shown below:





Export Table

	From Id	To Id	Tx	Meta
1	1	0	2	
2	0	1	2.00369	
3	2	0	5	
4	0	2	5.00369	
5	3	0	8	
6	0	3	8.00369	
7	4	0	11	
8	0	4	11.0037	
9	5	0	14	
10	0	5	14.0037	
11	6	0	17	
12	0	6	17.0037	

d. Create Ascii Trace file and execute analysis with Tracemetrics.

The ASCII trace file is made using the below mentioned command:

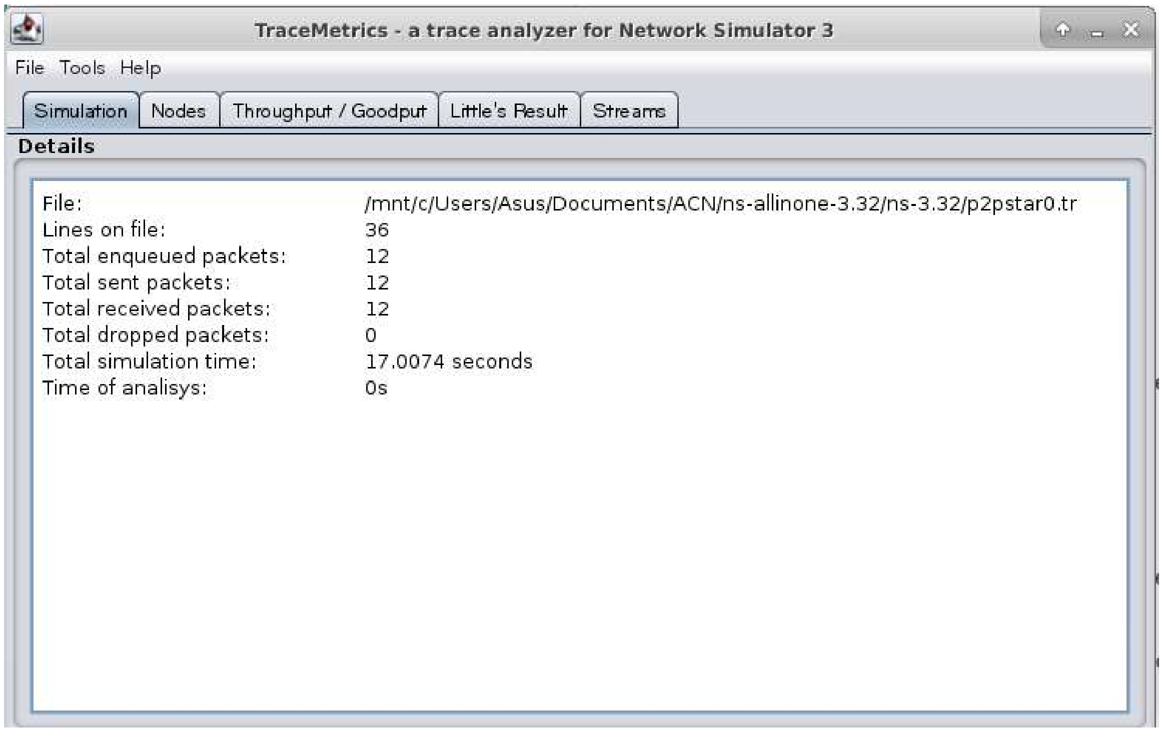
```
AsciiTraceHelper ascii;
pointToPoint.EnableAsciiAll(ascii.CreateFileStream("p2pstar0.tr"));
```

To run TraceMetrics - trace analyzer, run the following command in the directory where you have unzipped/extracted the tracemetrics.zip file.

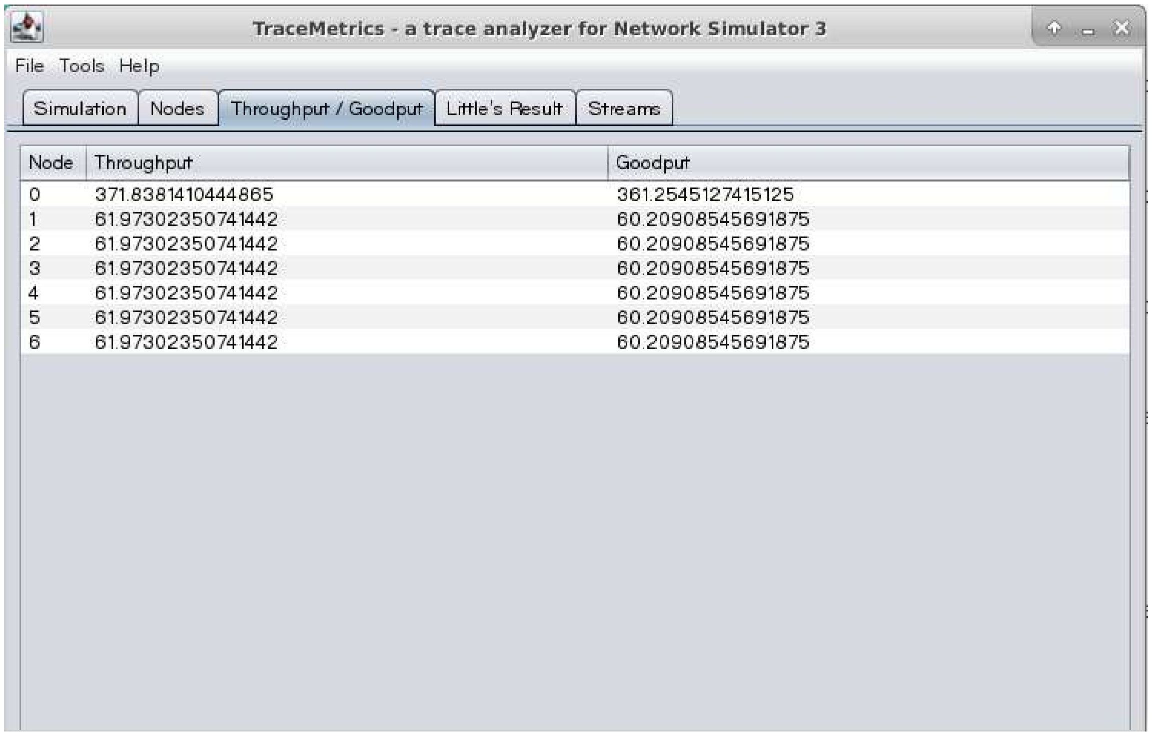
```
rajan@RAJAN:/mnt/c/users/Asus/Documents/ACN/ns-allinone-3.32$ java -jar tracemetrics.jar
TCP size: 0
```

The GUI of TraceMetrics - a trace analyzer for NS3 will open, you will have to select the trace file created by you in the program using File > Open from the Menu bar. And then, all the details of

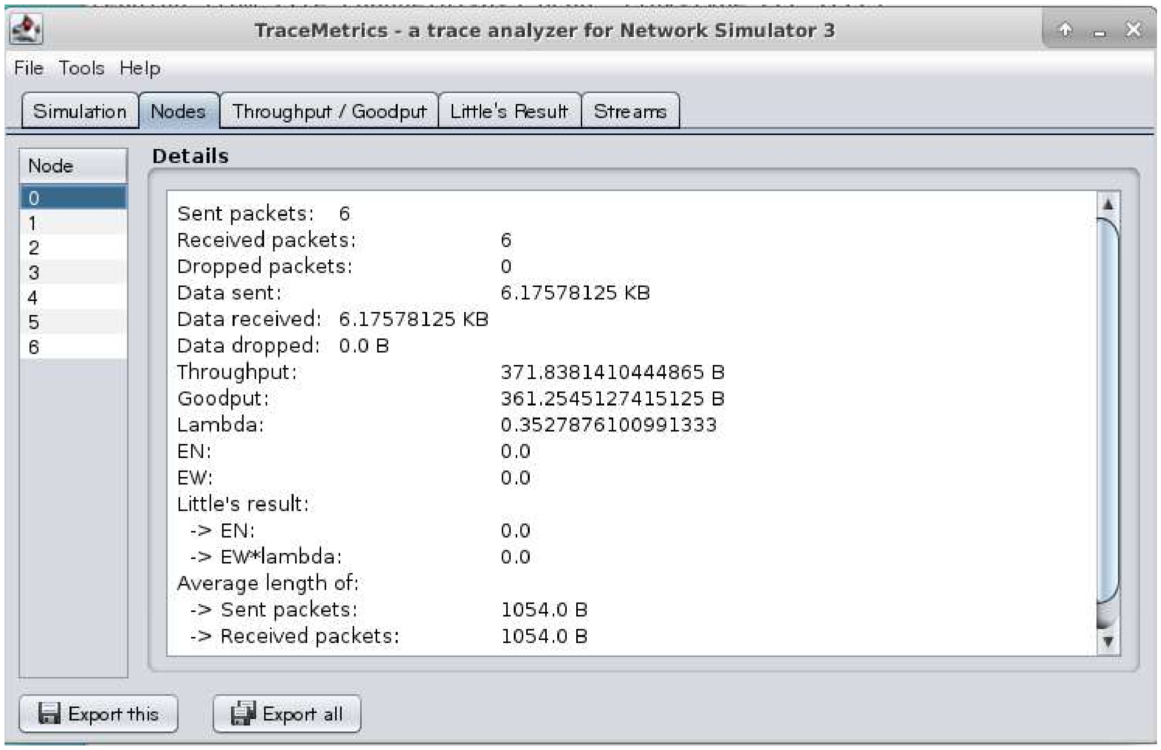
Simulation, Nodes, Throughput/Goodput Little's Result, and Streams will be available in the trace analyzer. You can view that. The screenshots for the above file are attached.


TraceMetrics - a trace analyzer for Network Simulator 3
File Tools Help
Simulation Nodes Throughput / Goodput Little's Result Streams
Details

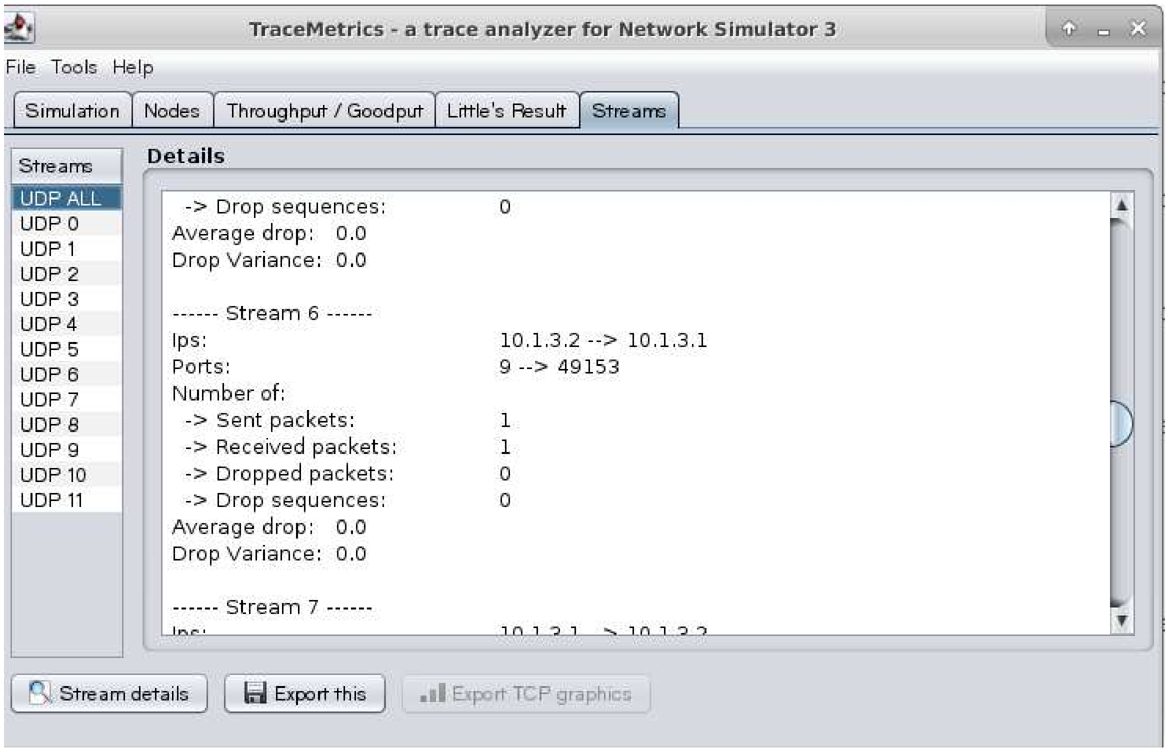
```
File:                      /mnt/c/Users/Asus/Documents/ACN/ns-allinone-3.32/ns-3.32/p2pstar0.tr
Lines on file:             36
Total enqueued packets:    12
Total sent packets:        12
Total received packets:    12
Total dropped packets:     0
Total simulation time:     17.0074 seconds
Time of analisys:          0s
```


TraceMetrics - a trace analyzer for Network Simulator 3
File Tools Help
Simulation Nodes Throughput / Goodput Little's Result Streams

Node	Throughput	Goodput
0	371.8381410444865	361.2545127415125
1	61.97302350741442	60.20908545691875
2	61.97302350741442	60.20908545691875
3	61.97302350741442	60.20908545691875
4	61.97302350741442	60.20908545691875
5	61.97302350741442	60.20908545691875
6	61.97302350741442	60.20908545691875



```
Sent packets:     6
Received packets:            6
Dropped packets:             0
Data sent:                   6.17578125 KB
Data received:  6.17578125 KB
Data dropped:   0.0 B
Throughput:                  371.8381410444865 B
Goodput:                     361.2545127415125 B
Lambda:                      0.3527876100991333
EN:                          0.0
EW:                          0.0
Little's result:
  -> EN:                     0.0
  -> EW*lambda:              0.0
Average length of:
  -> Sent packets:           1054.0 B
  -> Received packets:       1054.0 B
```



```
  -> Drop sequences:          0
Average drop:   0.0
Drop Variance:  0.0

------ Stream 6 ------
Ips:                          10.1.3.2 --> 10.1.3.1
Ports:                        9 --> 49153
Number of:
  -> Sent packets:            1
  -> Received packets:        1
  -> Dropped packets:         0
  -> Drop sequences:          0
Average drop:   0.0
Drop Variance:  0.0

------ Stream 7 ------
```