

Networking with Linux Lab

Module 2 & 3: Client Server Network topology using NS-3 and Animating the Network

Assignment 3: Create a bus topology, assign IPv4 addresses, and animate a simple network using NetAnim in Network Simulator.

1. Change the number of packets to 4 and the size to 1000 using command line arguments.

Aim: To modify the packet count to 4 and packet size to 1000 using command line arguments

Theory: Bus Topology

A bus topology is a network configuration in which nodes are connected in a shared communication line, known as a bus. The bus serves as a central communication pathway that all devices on the network can access. Each device has a unique address, and data transmitted by one device is received by all devices on the bus

Command-Line Arguments

Command-line arguments are parameters supplied to a program when it is executed from the command line or terminal. They allow users to customize the behavior of a program without modifying its source code.

Code:

> second.cc

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/ipv4-global-routing-helper.h"
```

```
using namespace ns3;
```

```
NS_LOG_COMPONENT_DEFINE ("SecondScriptExample");
```

```
int
main (int argc, char *argv[])
{
```

```
bool verbose = true;
uint32_t nCsmas = 3;

CommandLine cmd (__FILE__);
cmd.AddValue ("nCsmas", "Number of \"extra\" CSMA nodes/devices", nCsmas);
cmd.AddValue ("verbose", "Tell echo applications to log if true", verbose);

uint32_t nPackets = 1;
cmd.AddValue("nPkts", "Number of packets to echo", nPackets);
uint32_t nSize = 1024;
cmd.AddValue("nSize", "Size of each packet", nSize);

cmd.Parse (argc,argv);

if (verbose)
{
    LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
    LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
}

nCsmas = nCsmas == 0 ? 1 : nCsmas;

NodeContainer p2pNodes;
p2pNodes.Create (2);

NodeContainer csmaNodes;
csmaNodes.Add (p2pNodes.Get (1));
csmaNodes.Create (nCsmas);

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

NetDeviceContainer p2pDevices;
p2pDevices = pointToPoint.Install (p2pNodes);

CsmaHelper csma;
csma.SetChannelAttribute ("DataRate", StringValue ("100Mbps"));
csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (6560)));

NetDeviceContainer csmaDevices;
csmaDevices = csma.Install (csmaNodes);

InternetStackHelper stack;
stack.Install (p2pNodes.Get (0));
stack.Install (csmaNodes);
```

```

Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer p2pInterfaces;
p2pInterfaces = address.Assign (p2pDevices);

address.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4InterfaceContainer csmaInterfaces;
csmaInterfaces = address.Assign (csmaDevices);

UdpEchoServerHelper echoServer (9);

ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get (nCsma));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (10.0));

UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (nCsma), 9);
echoClient.SetAttribute ("MaxPackets", UIntegerValue (nPackets));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UIntegerValue (nSize));

ApplicationContainer clientApps = echoClient.Install (p2pNodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (10.0));

Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

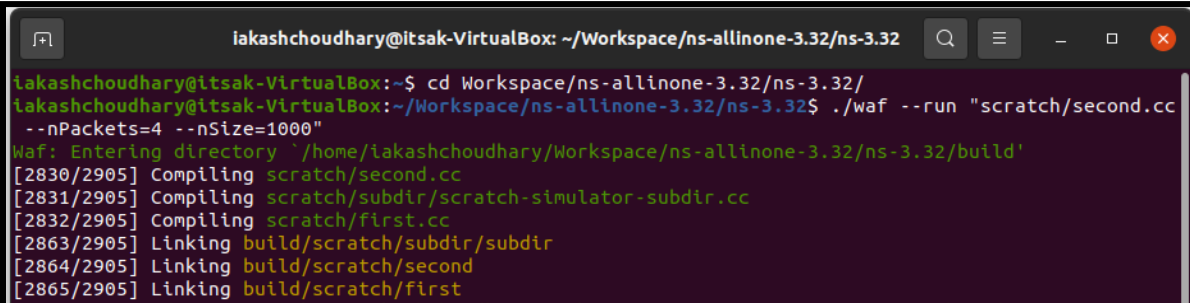
pointToPoint.EnablePcapAll ("second");
csma.EnablePcap ("second", csmaDevices.Get (1), true);

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

```

Command & Screenshot:

> \$./waf --run "scratch/second.cc --nPackets=4 --nSize=1000"



```

iakashchoudhary@itsak-VirtualBox: ~/Workspace/ns-allinone-3.32/ns-3.32
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ ./waf --run "scratch/second.cc
--nPackets=4 --nSize=1000"
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
[2830/2905] Compiling scratch/second.cc
[2831/2905] Compiling scratch/subdir/scratch-simulator-subdir.cc
[2832/2905] Compiling scratch/first.cc
[2863/2905] Linking build/scratch/subdir/subdir
[2864/2905] Linking build/scratch/second
[2865/2905] Linking build/scratch/first

```

```
Waf: Leaving directory '/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (9.139s)
At time +2s client sent 1000 bytes to 10.1.2.4 port 9
At time +2.00776s server received 1000 bytes from 10.1.1.1 port 49153
At time +2.00776s server sent 1000 bytes to 10.1.1.1 port 49153
At time +2.01753s client received 1000 bytes from 10.1.2.4 port 9
At time +3s client sent 1000 bytes to 10.1.2.4 port 9
At time +3.00374s server received 1000 bytes from 10.1.1.1 port 49153
At time +3.00374s server sent 1000 bytes to 10.1.1.1 port 49153
At time +3.00748s client received 1000 bytes from 10.1.2.4 port 9
At time +4s client sent 1000 bytes to 10.1.2.4 port 9
At time +4.00374s server received 1000 bytes from 10.1.1.1 port 49153
At time +4.00374s server sent 1000 bytes to 10.1.1.1 port 49153
At time +4.00748s client received 1000 bytes from 10.1.2.4 port 9
At time +5s client sent 1000 bytes to 10.1.2.4 port 9
At time +5.00374s server received 1000 bytes from 10.1.1.1 port 49153
At time +5.00374s server sent 1000 bytes to 10.1.1.1 port 49153
At time +5.00748s client received 1000 bytes from 10.1.2.4 port 9
lakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$
```

Conclusion: The successful design of a bus topology using command-line arguments to modify the number of packets to 4 and the size to 1000 was achieved.

2. Modify the LogComponentEnable() method to output ALL and WARN messages.

Aim: To configure the LogComponentEnable() method to print both ALL and WARN log messages

Theory:

Changing LogComponentEnable() values

The LogComponentEnable() method is likely a part of a logging system in a software application. By changing its values to echo "ALL" and "WARN" messages, you are adjusting the logging level.

- "ALL" typically means that all log messages, including DEBUG, INFO, WARN, and ERROR, will be logged.
- "WARN" means that only WARNING and ERROR messages will be logged.

This change suggests that you want to capture a broader range of log messages, including informational (INFO) and debugging (DEBUG) messages, as well as warnings and errors.

Code:

> second.cc

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/ipv4-global-routing-helper.h"
```

```
using namespace ns3;
```

```
NS_LOG_COMPONENT_DEFINE ("SecondScriptExample");
```

```
int
```

```
main (int argc, char *argv[])
```

```
{
```

```
    bool verbose = true;
```

```
    uint32_t nCsma = 3;
```

```
    CommandLine cmd (__FILE__);
```

```
    cmd.AddValue ("nCsma", "Number of \"extra\" CSMA nodes/devices", nCsma);
```

```
    cmd.AddValue ("verbose", "Tell echo applications to log if true", verbose);
```

```
uint32_t nPackets = 1;
cmd.AddValue("nPackets", "Number of packets to echo", nPackets);
uint32_t nSize = 1024;
cmd.AddValue("nSize", "Size of each packet", nSize);

cmd.Parse (argc,argv);

if (verbose)
{
    // Uncomment the following lines as needed based on the requirements
    //LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_ALL);
    //LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_ALL);
    //LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_WARN);
    //LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_WARN);
}

nCsma = nCsma == 0 ? 1 : nCsma;

NodeContainer p2pNodes;
p2pNodes.Create (2);

NodeContainer csmaNodes;
csmaNodes.Add (p2pNodes.Get (1));
csmaNodes.Create (nCsma);

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

NetDeviceContainer p2pDevices;
p2pDevices = pointToPoint.Install (p2pNodes);

CsmaHelper csma;
csma.SetChannelAttribute ("DataRate", StringValue ("100Mbps"));
csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (6560)));

NetDeviceContainer csmaDevices;
csmaDevices = csma.Install (csmaNodes);

InternetStackHelper stack;
stack.Install (p2pNodes.Get (0));
stack.Install (csmaNodes);

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

```

Ipv4InterfaceContainer p2pInterfaces;
p2pInterfaces = address.Assign (p2pDevices);

address.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4InterfaceContainer csmaInterfaces;
csmaInterfaces = address.Assign (csmaDevices);

UdpEchoServerHelper echoServer (9);

ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get (nCsma));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (10.0));

UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (nCsma), 9);
echoClient.SetAttribute ("MaxPackets", UintegerValue (nPackets));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UintegerValue (nSize));

ApplicationContainer clientApps = echoClient.Install (p2pNodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (10.0));

Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

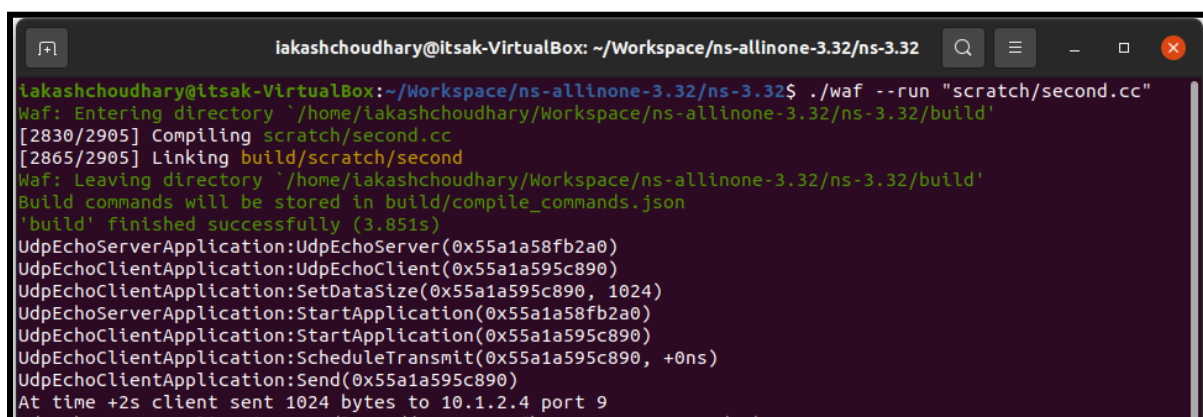
pointToPoint.EnablePcapAll ("second");
csma.EnablePcap ("second", csmaDevices.Get (1), true);

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

```

Command & Screenshot:

> For ALL messages: **\$./waf --run "scratch/second.cc"**



```

iakashchoudhary@itsak-VirtualBox: ~/Workspace/ns-allinone-3.32/ns-3.32
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ ./waf --run "scratch/second.cc"
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
[2830/2905] Compiling scratch/second.cc
[2865/2905] Linking build/scratch/second
Waf: Leaving directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (3.851s)
UdpEchoServerApplication:UdpEchoServer(0x55a1a58fb2a0)
UdpEchoClientApplication:UdpEchoClient(0x55a1a595c890)
UdpEchoClientApplication:SetDataSize(0x55a1a595c890, 1024)
UdpEchoServerApplication:StartApplication(0x55a1a58fb2a0)
UdpEchoClientApplication:StartApplication(0x55a1a595c890)
UdpEchoClientApplication:ScheduleTransmit(0x55a1a595c890, +0ns)
UdpEchoClientApplication:Send(0x55a1a595c890)
At time +2s client sent 1024 bytes to 10.1.2.4 port 9

```

```
UdpEchoServerApplication:HandleRead(0x55a1a58fb2a0, 0x55a1a5a24d20)
At time +2.0078s server received 1024 bytes from 10.1.1.1 port 49153
Echoing packet
At time +2.0078s server sent 1024 bytes to 10.1.1.1 port 49153
UdpEchoClientApplication:HandleRead(0x55a1a595c890, 0x55a1a5a25460)
At time +2.01761s client received 1024 bytes from 10.1.2.4 port 9
UdpEchoClientApplication:StopApplication(0x55a1a595c890)
UdpEchoServerApplication:StopApplication(0x55a1a58fb2a0)
UdpEchoClientApplication:DoDispose(0x55a1a595c890)
UdpEchoServerApplication:DoDispose(0x55a1a58fb2a0)
UdpEchoClientApplication::~UdpEchoClient(0x55a1a595c890)
UdpEchoServerApplication::~UdpEchoServer(0x55a1a58fb2a0)
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$
```

> For WARN messages: `$./waf --run "scratch/second.cc"`

```
iakashchoudhary@itsak-VirtualBox: ~/Workspace/ns-allinone-3.32/ns-3.32
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ ./waf --run "scratch/second.cc"
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
[2835/2905] Compiling scratch/second.cc
Waf: Leaving directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (3.129s)
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$
```

Conclusion: Enabling the LogComponent to echo ALL and WARN messages provides comprehensive logging while specifically highlighting warning messages for a thorough understanding of system behavior.

3. Perform animation using PyGraphviz and NetAnim on a bus topology.

Aim: To demonstrate and visualize network communication in a bus topology using PyGraphviz and NetAnim through animation

Theory:

PyGraphviz is used to visualize network topologies. Network topology refers to the arrangement of nodes and connections in a network. Graph theory helps in modeling and understanding these topologies.

NetAnim is a network animation tool used with ns-3 for simulating and visualizing network behavior. It employs Discrete Event Simulation (DES) to model events like packet transmissions and node movements at discrete time points. The tool focuses on packet-level visualization, providing insights into data flow and network issues. Its real-time analysis feature aids in debugging and optimizing network protocols by offering a dynamic visual representation of the simulation.

Code:

> **second.cc**

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/ipv4-global-routing-helper.h"
#include "ns3/netanim-module.h"
#include "ns3/mobility-helper.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("SecondScriptExample");

int
main (int argc, char *argv[])
{
    bool verbose = true;
    uint32_t nCsmas = 3;

    CommandLine cmd (__FILE__);
    cmd.AddValue ("nCsmas", "Number of \"extra\" CSMA nodes/devices", nCsmas);
    cmd.AddValue ("verbose", "Tell echo applications to log if true", verbose);
```

```
uint32_t nPackets = 1;
cmd.AddValue("nPackets", "Number of packets to echo", nPackets);
uint32_t nSize = 1024;
cmd.AddValue("nSize", "Size of each packet", nSize);

cmd.Parse (argc,argv);

if (verbose)
{
    LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
    LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
}

nCsma = nCsma == 0 ? 1 : nCsma;

NodeContainer p2pNodes;
p2pNodes.Create (2);

NodeContainer csmaNodes;
csmaNodes.Add (p2pNodes.Get (1));
csmaNodes.Create (nCsma);

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

NetDeviceContainer p2pDevices;
p2pDevices = pointToPoint.Install (p2pNodes);

CsmaHelper csma;
csma.SetChannelAttribute ("DataRate", StringValue ("100Mbps"));
csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (6560)));

NetDeviceContainer csmaDevices;
csmaDevices = csma.Install (csmaNodes);

InternetStackHelper stack;
stack.Install (p2pNodes.Get (0));
stack.Install (csmaNodes);

Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer p2pInterfaces;
p2pInterfaces = address.Assign (p2pDevices);
```

```
address.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4InterfaceContainer csmaInterfaces;
csmaInterfaces = address.Assign (csmaDevices);

UdpEchoServerHelper echoServer (9);

ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get (nCsma));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (10.0));

UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (nCsma), 9);
echoClient.SetAttribute ("MaxPackets", UIntegerValue (nPackets));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UIntegerValue (nSize));

ApplicationContainer clientApps = echoClient.Install (p2pNodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (10.0));

Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

// Creating pcap files for Wireshark-pcap: packet capture information
pointToPoint.EnablePcapAll ("second");
csma.EnablePcap ("second", csmaDevices.Get (1), true);

// NetAnimation ---before simulator run
MobilityHelper mobility;
mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
mobility.Install(p2pNodes);
mobility.Install(csmaNodes);

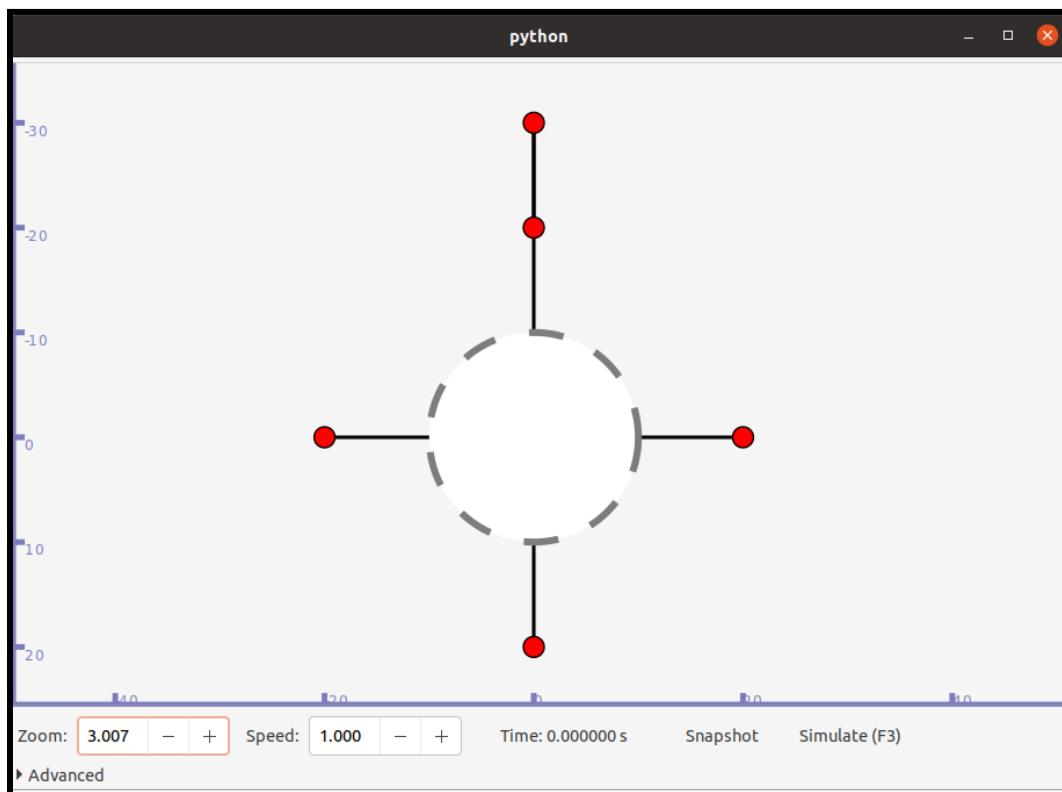
AnimationInterface anim("second.xml");
AnimationInterface::SetConstantPosition(p2pNodes.Get(0), 0, -30);
AnimationInterface::SetConstantPosition(p2pNodes.Get(1), 0, -20);
AnimationInterface::SetConstantPosition(csmaNodes.Get(1), 0, 20);
AnimationInterface::SetConstantPosition(csmaNodes.Get(2), 20, 0);
AnimationInterface::SetConstantPosition(csmaNodes.Get(3), -20, 0);
anim.EnablePacketMetadata(true);

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

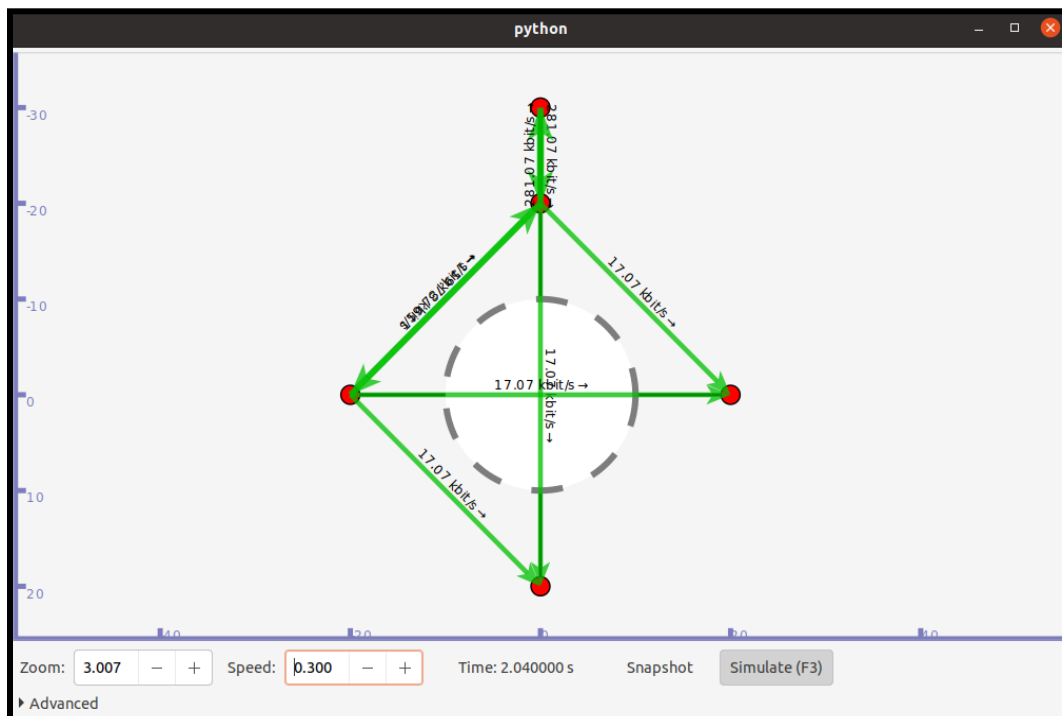
Command & Screenshot:

> Perform animation using pygraphviz: `$./waf --run "scratch/second.cc" --vis`

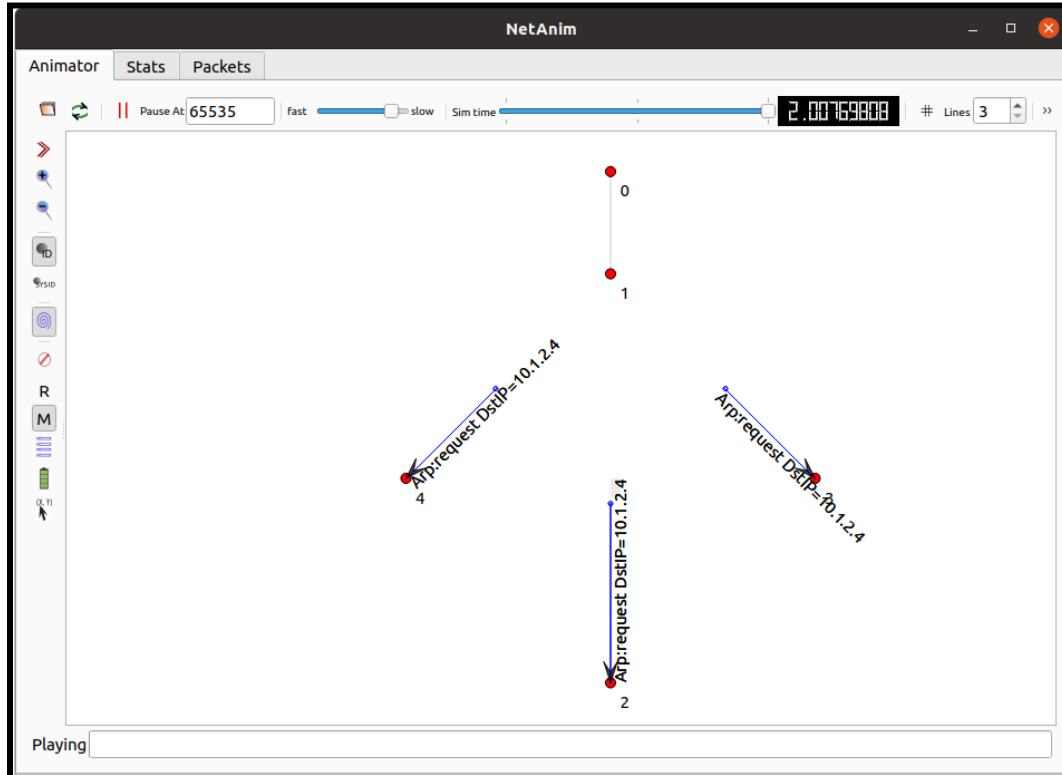
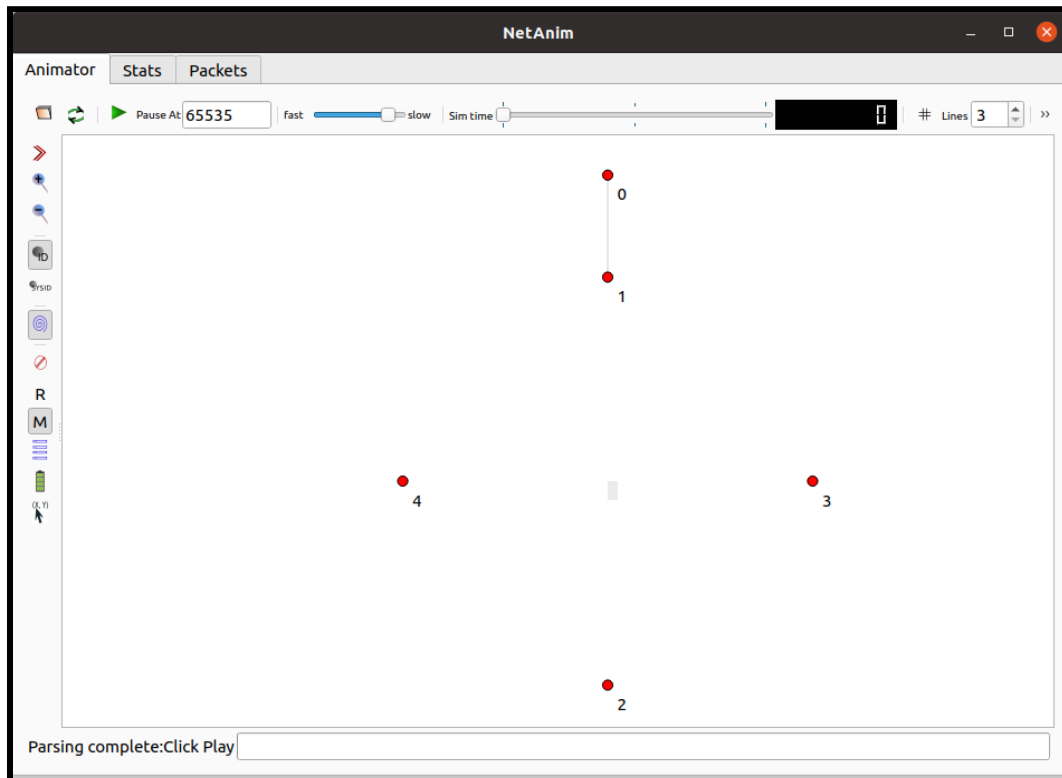
```
iakashchoudhary@itsak-VirtualBox: ~/Workspace/ns-allinone-3.32/ns-3...  
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ sudo ./waf --run  
scratch/second.cc --vis  
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'  
[2846/2905] Compiling scratch/second.cc  
[2865/2905] Linking build/scratch/second  
Waf: Leaving directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'  
Build commands will be stored in build/compile_commands.json  
'build' finished successfully (3.659s)  
Could not load plugin 'show_last_packets.py': No module named 'kiwi'  
Could not load icon applets-screenshooter due to missing gnomedesktop Python module  
scanning topology: 5 nodes...  
scanning topology: calling graphviz layout  
scanning topology: all done.  
█
```



```
iakashchoudhary@itsak-VirtualBox: ~/Workspace/ns-allinone-3.32/ns-3...  
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ sudo ./waf --run  
scratch/second.cc --vis  
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'  
[2846/2905] Compiling scratch/second.cc  
[2865/2905] Linking build/scratch/second  
Waf: Leaving directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'  
Build commands will be stored in build/compile_commands.json  
'build' finished successfully (3.659s)  
Could not load plugin 'show_last_packets.py': No module named 'kiwi'  
Could not load icon applets-screenshooter due to missing gnomedesktop Python module  
scanning topology: 5 nodes...  
scanning topology: calling graphviz layout  
scanning topology: all done.  
At time +2s client sent 1024 bytes to 10.1.2.4 port 9  
At time +2.0078s server received 1024 bytes from 10.1.1.1 port 49153  
At time +2.0078s server sent 1024 bytes to 10.1.1.1 port 49153  
At time +2.01761s client received 1024 bytes from 10.1.2.4 port 9  
█
```



> Perform animation using NetAnim: **\$./NetAnim**



Conclusion: PyGraphviz and NetAnim can be effectively used to animate and visualize a bus topology in a network simulation.

4. Create a Trace file (.tr file) and then retrieve information using the TraceMetric Tool/tcpdump.

Aim: To generate a trace file (.tr file) and analyze network traffic using the TraceMetric Tool/tcpdump for performance evaluation and debugging purposes

Theory:

TraceMetric Tool

TraceMetric tools are software utilities designed to analyze and extract specific metrics or information from trace files. These tools offer functionalities to parse trace files, filter data based on criteria, calculate various network metrics (e.g., throughput, packet loss, delay), and present results in a readable format.

Depending on the tool, it may have a graphical user interface (GUI) or operate through a command-line interface (CLI).

tcpdump

tcpdump is a widely-used command-line packet analyzer for Unix-like systems. It captures packets traversing a network interface and saves them to a file or displays them in real-time.

tcpdump allows users to apply filters based on protocol, source/destination IP addresses, port numbers, etc., to capture specific network traffic. By analyzing the captured packets, users can gain insights into network behavior, diagnose problems, and troubleshoot network issues.

Code:

```
> second.cc
```

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/ipv4-global-routing-helper.h"
#include "ns3/netanim-module.h"
#include "ns3/mobility-helper.h"
```

```
using namespace ns3;
```

```
NS_LOG_COMPONENT_DEFINE ("SecondScriptExample");
```

```
int
```



```
main (int argc, char *argv[])
{
    bool verbose = true;
    uint32_t nCsma = 3;

    CommandLine cmd (__FILE__);
    cmd.AddValue ("nCsma", "Number of \"extra\" CSMA nodes/devices", nCsma);
    cmd.AddValue ("verbose", "Tell echo applications to log if true", verbose);

    uint32_t nPackets = 1;
    cmd.AddValue("nPackets", "Number of packets to echo", nPackets);
    uint32_t nSize = 1024;
    cmd.AddValue("nSize", "Size of each packet", nSize);

    cmd.Parse (argc,argv);

    if (verbose)
    {
        LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
        LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
    }

    nCsma = nCsma == 0 ? 1 : nCsma;

    NodeContainer p2pNodes;
    p2pNodes.Create (2);

    NodeContainer csmaNodes;
    csmaNodes.Add (p2pNodes.Get (1));
    csmaNodes.Create (nCsma);

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

    NetDeviceContainer p2pDevices;
    p2pDevices = pointToPoint.Install (p2pNodes);

    CsmaHelper csma;
    csma.SetChannelAttribute ("DataRate", StringValue ("100Mbps"));
    csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (6560)));

    NetDeviceContainer csmaDevices;
    csmaDevices = csma.Install (csmaNodes);

    InternetStackHelper stack;
```

```
stack.Install (p2pNodes.Get (0));
stack.Install (csmaNodes);

Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer p2pInterfaces;
p2pInterfaces = address.Assign (p2pDevices);

address.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4InterfaceContainer csmaInterfaces;
csmaInterfaces = address.Assign (csmaDevices);

UdpEchoServerHelper echoServer (9);

ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get (nCsma));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (10.0));

UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (nCsma), 9);
echoClient.SetAttribute ("MaxPackets", UIntegerValue (nPackets));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UIntegerValue (nSize));

ApplicationContainer clientApps = echoClient.Install (p2pNodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (10.0));

Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

// Creating pcap files for Wireshark-pcap: packet capture information
pointToPoint.EnablePcapAll ("second");
csma.EnablePcap ("second", csmaDevices.Get (1), true);

// NetAnimation ---before simulator run
MobilityHelper mobility;
mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
mobility.Install(p2pNodes);
mobility.Install(csmaNodes);

AnimationInterface anim("second.xml");
AnimationInterface::SetConstantPosition(p2pNodes.Get(0), 0, -30);
AnimationInterface::SetConstantPosition(p2pNodes.Get(1), 0, -20);
AnimationInterface::SetConstantPosition(csmaNodes.Get(1), 0, 20);
AnimationInterface::SetConstantPosition(csmaNodes.Get(2), 20, 0);
AnimationInterface::SetConstantPosition(csmaNodes.Get(3), -20, 0);
anim.EnablePacketMetadata(true);
```

```
// Create the ASCII trace file stream
AsciiTraceHelper ascii;
Ptr<OutputStreamWrapper> stream = ascii.CreateFileStream("second.tr");

// Install the trace sinks for all interfaces
pointToPoint.EnableAsciiAll (stream);
csma.EnableAsciiAll (stream);

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

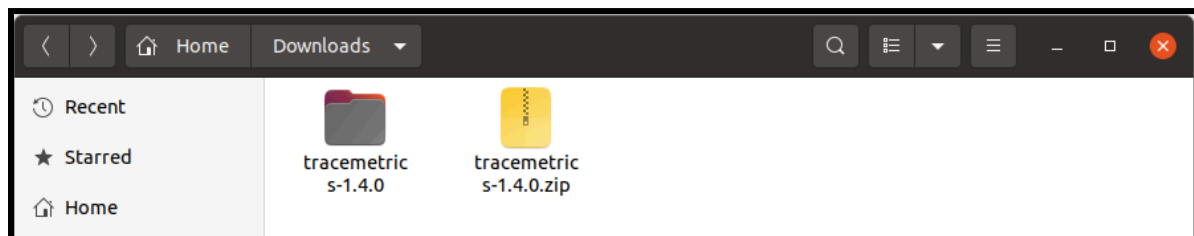
Command & Screenshot:

Step 1: Download TraceMetric from SourceForge using the link below.

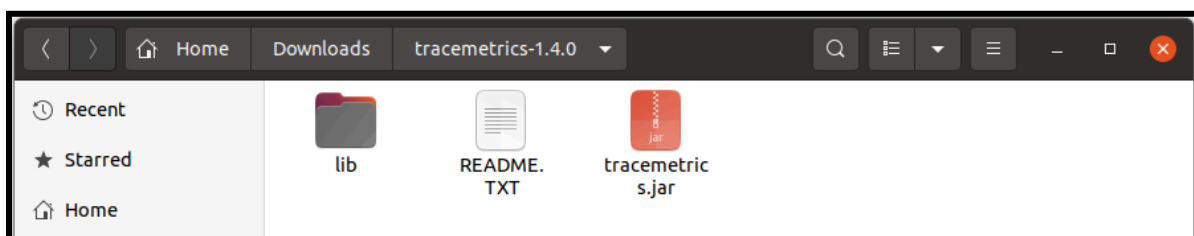
<https://sourceforge.net/projects/tracemetrics/>



Step 2: Extract the contents of tracemetrics-1.4.0.zip either using the terminal or manually.

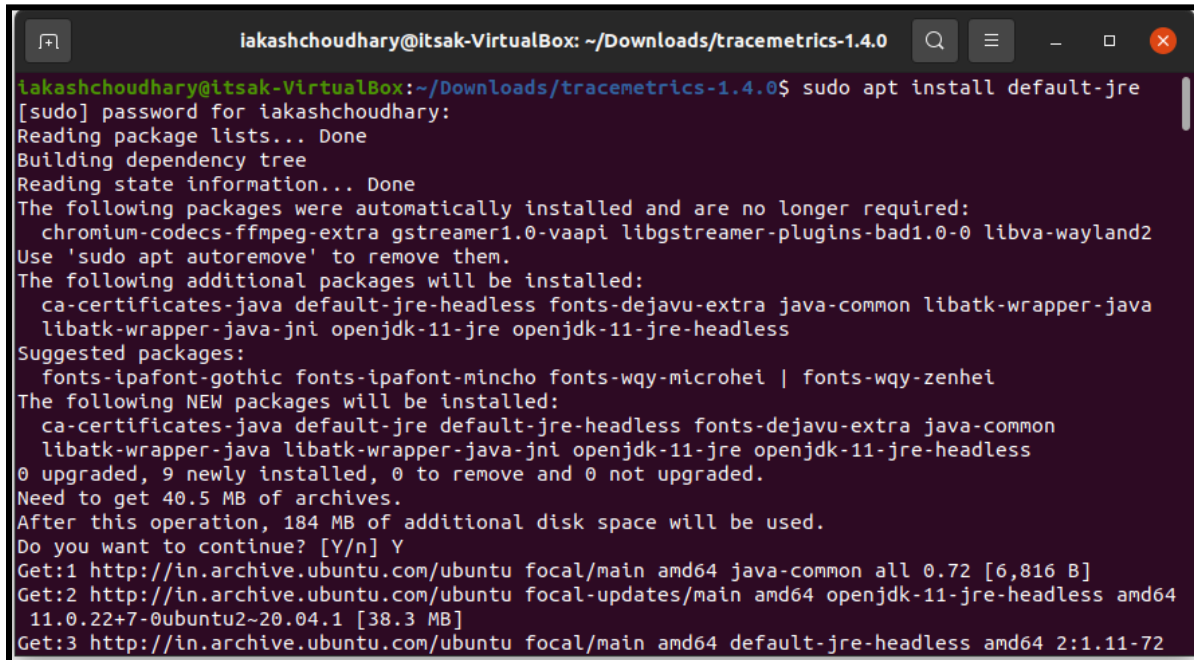


Step 3: Open the extracted folder, then right-click on a blank space and choose 'Open in Terminal'.



Step 4: Execute the following command to install default-jre.

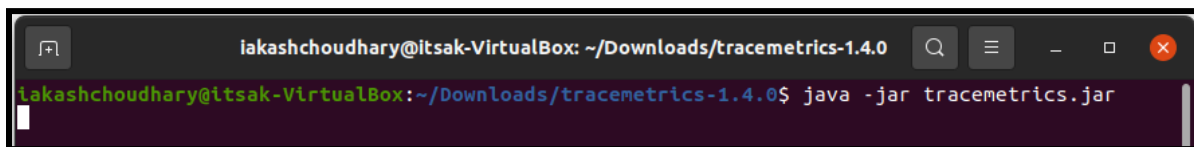
> **\$ sudo apt install default-jre**



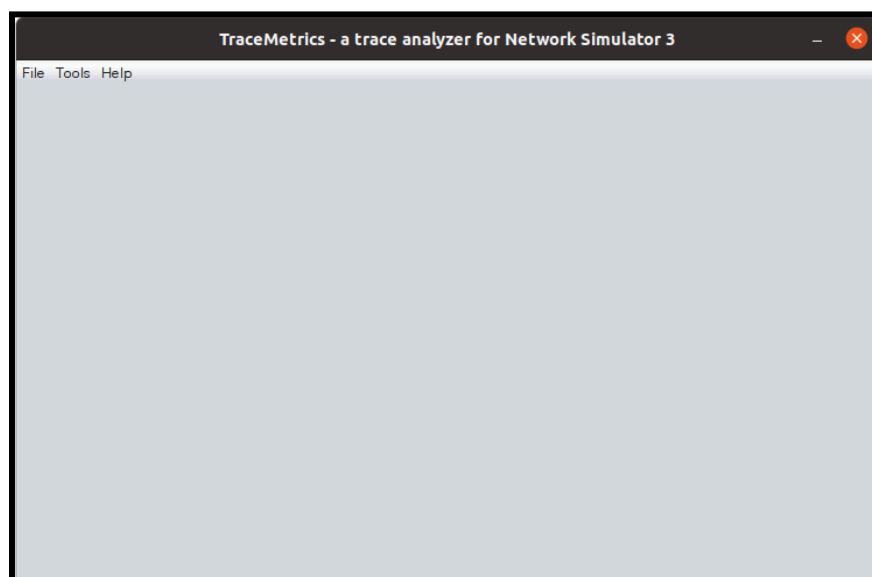
```
iakashchoudhary@itsak-VirtualBox: ~/Downloads/tracemetrics-1.4.0
iakashchoudhary@itsak-VirtualBox:~/Downloads/tracemetrics-1.4.0$ sudo apt install default-jre
[sudo] password for iakashchoudhary:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libgstreamer-plugins-bad1.0-0 libva-wayland2
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  ca-certificates-java default-jre-headless fonts-dejavu-extra java-common libatk-wrapper-java
  libatk-wrapper-java-jni openjdk-11-jre openjdk-11-jre-headless
Suggested packages:
  fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei
The following NEW packages will be installed:
  ca-certificates-java default-jre default-jre-headless fonts-dejavu-extra java-common
  libatk-wrapper-java libatk-wrapper-java-jni openjdk-11-jre openjdk-11-jre-headless
0 upgraded, 9 newly installed, 0 to remove and 0 not upgraded.
Need to get 40.5 MB of archives.
After this operation, 184 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://in.archive.ubuntu.com/ubuntu focal/main amd64 java-common all 0.72 [6,816 B]
Get:2 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 openjdk-11-jre-headless amd64
  11.0.22+7-0ubuntu2~20.04.1 [38.3 MB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal/main amd64 default-jre-headless amd64 2:1.11-72
```

Step 5: To run the TraceMetrics Tool, use the following command.

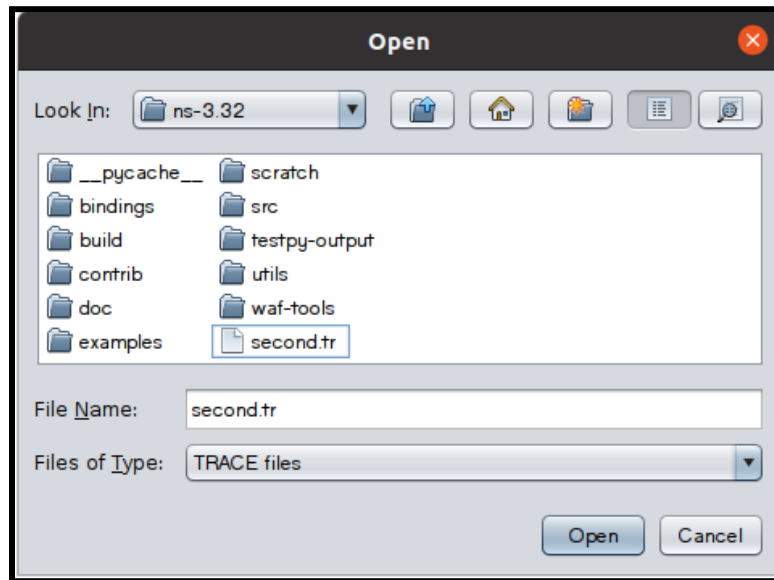
> **\$ java -jar tracemetrics.jar**



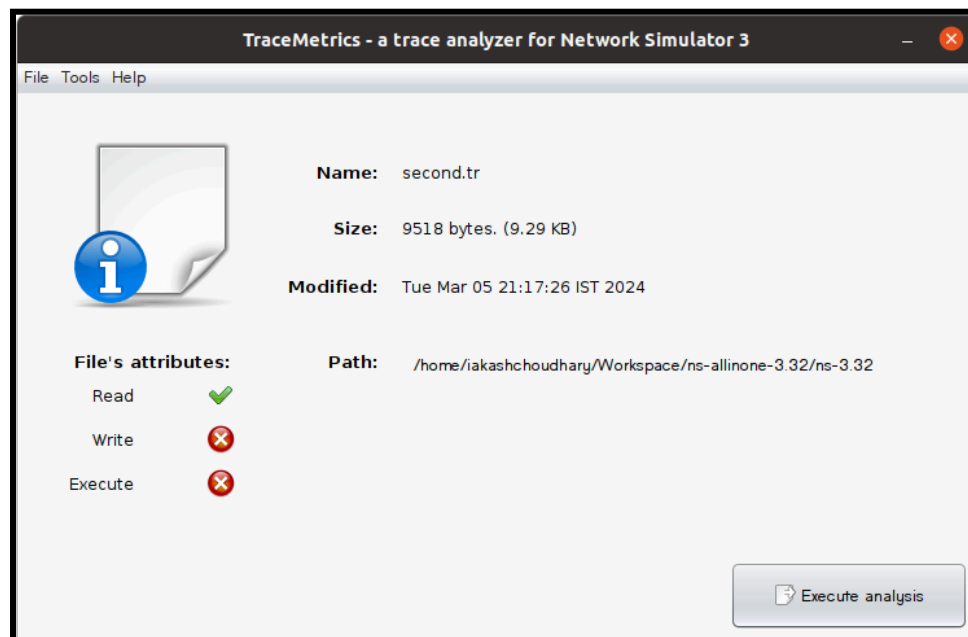
```
iakashchoudhary@itsak-VirtualBox: ~/Downloads/tracemetrics-1.4.0
iakashchoudhary@itsak-VirtualBox:~/Downloads/tracemetrics-1.4.0$ java -jar tracemetrics.jar
```



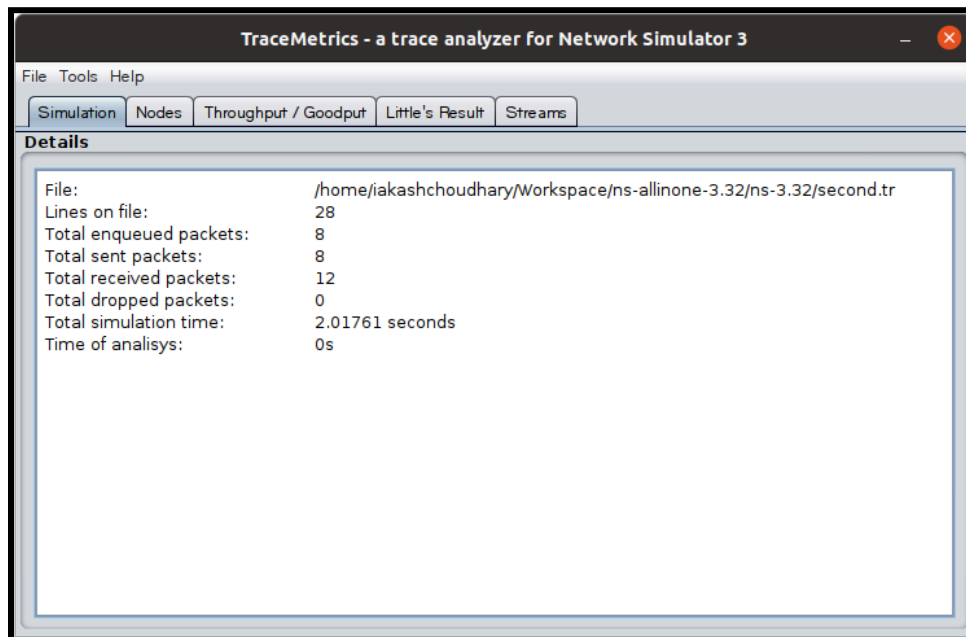
Step 6: Click on File. > Choose Open File. > Browse to the file location. > Click Open.



Step 7: Now, click on the Execute analysis.



Step 8: Finally, you can check the trace metrics of the bus topology.



The screenshot shows the 'TraceMetrics - a trace analyzer for Network Simulator 3' window. The 'Throughput / Goodput' tab is selected. The table displays the following data:

Node	Throughput	Goodput
0	522.4002656608562	507.5311878906231
1	1061.6521527946431	1032.9052691055258
2	0.0	0.0
3	0.0	0.0
4	539.251887133787	525.3740812149028

Conclusion: Creating a Trace file (.tr file) and analyzing it using TraceMetric Tool/tcpdump provides insights into network traffic patterns and allows for detailed examination of network communication.