Networking with Linux Lab
Module 2 & 3: Client Server Network topology using NS-3 and Animating the Network

Assignment 2: Create a simple topology and simulate traffic between two nodes using Pygraphviz and NetAnim.

1. Create a simple topology using a class B address.

Aim: To create a simple topology using a class B address

Theory:

Class B Addressing

In networking, IP addresses are categorized into classes, and Class B addresses fall within the range of 128.0.0.0 to 191.255.255.255. Class B addresses are typically used for medium-sized networks.

A Class B address is divided into network and host portions, with the first two octets representing the network and the last two octets representing the host.

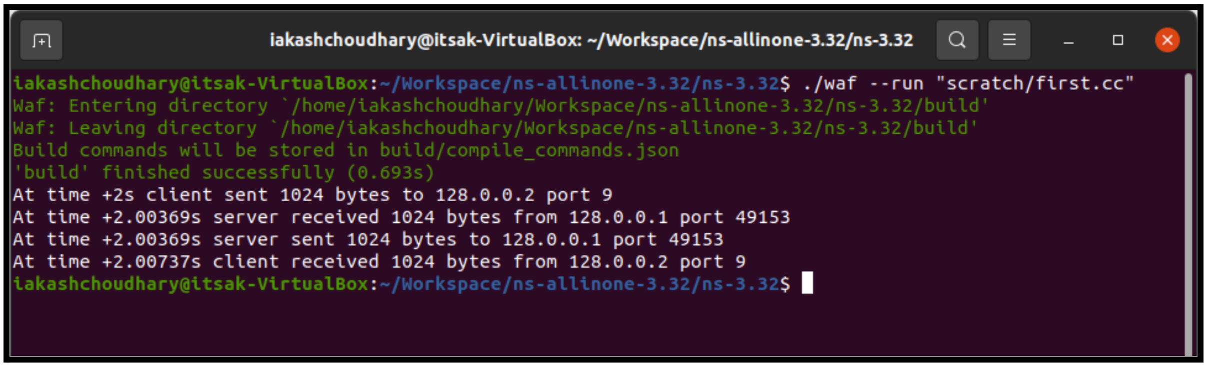
Code:

> first.cc

```
#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"

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 ("128.0.0.0", "255.255.0.0");

  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));

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

Command & Screenshot:

> $ ./waf --run "scratch/first.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/first.cc"
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
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 (0.693s)
At time +2s client sent 1024 bytes to 128.0.0.2 port 9
At time +2.00369s server received 1024 bytes from 128.0.0.1 port 49153
At time +2.00369s server sent 1024 bytes to 128.0.0.1 port 49153
At time +2.00737s client received 1024 bytes from 128.0.0.2 port 9
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ 
```

Conclusion: Designing a simple topology using a class B address provides efficient IP address allocation and network management for medium-sized networks.

2. Use command-line arguments to change the number of packets to 3 and the size to 1000.

Aim: To demonstrate the use of command line arguments to change the number of packets to 3 and the size to 1000

Theory:

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.

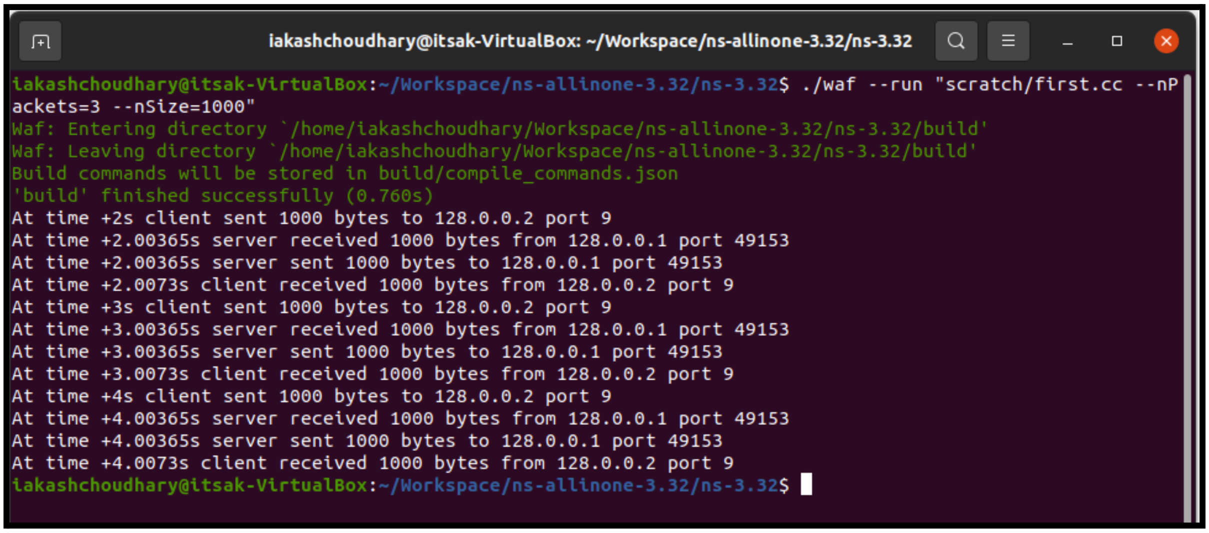
Code:

> first.cc

```
#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"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");

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

  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);

  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 ("128.0.0.0", "255.255.0.0");

  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 (nPackets));
  echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
  echoClient.SetAttribute ("PacketSize", UintegerValue (nSize));

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

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

Command & Screenshot:

> $ ./waf --run "scratch/first.cc --nPackets=3 --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/first.cc --nP
ackets=3 --nSize=1000"
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
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 (0.760s)
At time +2s client sent 1000 bytes to 128.0.0.2 port 9
At time +2.00365s server received 1000 bytes from 128.0.0.1 port 49153
At time +2.00365s server sent 1000 bytes to 128.0.0.1 port 49153
At time +2.0073s client received 1000 bytes from 128.0.0.2 port 9
At time +3s client sent 1000 bytes to 128.0.0.2 port 9
At time +3.00365s server received 1000 bytes from 128.0.0.1 port 49153
At time +3.00365s server sent 1000 bytes to 128.0.0.1 port 49153
At time +3.0073s client received 1000 bytes from 128.0.0.2 port 9
At time +4s client sent 1000 bytes to 128.0.0.2 port 9
At time +4.00365s server received 1000 bytes from 128.0.0.1 port 49153
At time +4.00365s server sent 1000 bytes to 128.0.0.1 port 49153
At time +4.0073s client received 1000 bytes from 128.0.0.2 port 9
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$
```

Conclusion: Hence, command-line arguments were successfully used to alter the number of packets to 3 and the size to 1000.

3. Change the values of the LogComponentEnable() method to echo 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:

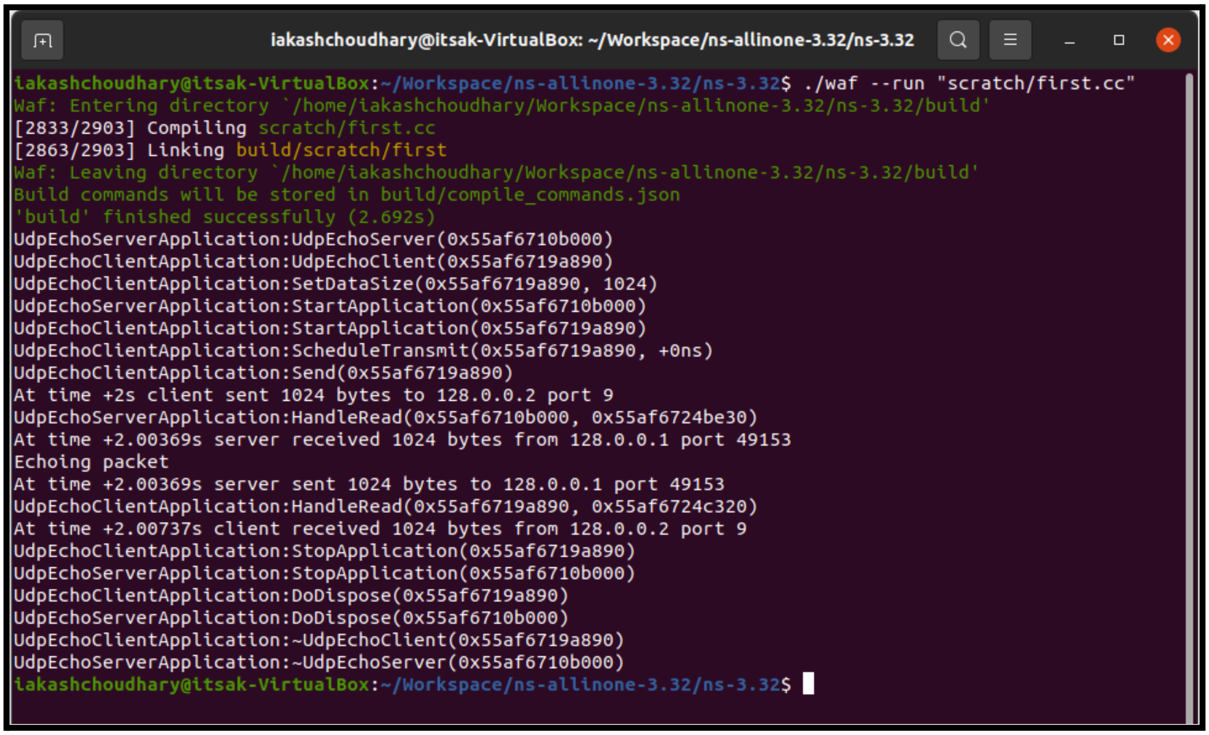
> first.cc

```
#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"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");

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

  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);
  Time::SetResolution (Time::NS);

  // 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);

  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 ("128.0.0.0", "255.255.0.0");

  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 (nPackets));
  echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
  echoClient.SetAttribute ("PacketSize", UintegerValue (nSize));

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

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

}

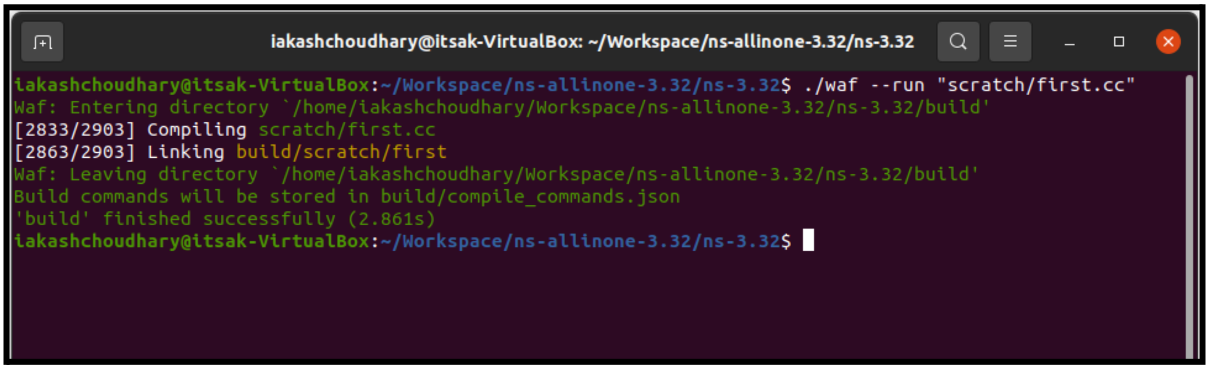
Command & Screenshot:

> For ALL messages: **$ ./waf --run "scratch/first.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/first.cc"
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
[2833/2903] Compiling scratch/first.cc
[2863/2903] 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 (2.692s)
UdpEchoServerApplication:UdpEchoServer(0x55af6710b000)
UdpEchoClientApplication:UdpEchoClient(0x55af6719a890)
UdpEchoClientApplication:SetDataSize(0x55af6719a890, 1024)
UdpEchoServerApplication:StartApplication(0x55af6710b000)
UdpEchoClientApplication:StartApplication(0x55af6719a890)
UdpEchoClientApplication:ScheduleTransmit(0x55af6719a890, +0ns)
UdpEchoClientApplication:Send(0x55af6719a890)
At time +2s client sent 1024 bytes to 128.0.0.2 port 9
UdpEchoServerApplication:HandleRead(0x55af6710b000, 0x55af6724be30)
At time +2.00369s server received 1024 bytes from 128.0.0.1 port 49153
Echoing packet
At time +2.00369s server sent 1024 bytes to 128.0.0.1 port 49153
UdpEchoClientApplication:HandleRead(0x55af6719a890, 0x55af6724c320)
At time +2.00737s client received 1024 bytes from 128.0.0.2 port 9
UdpEchoClientApplication:StopApplication(0x55af6719a890)
UdpEchoServerApplication:StopApplication(0x55af6710b000)
UdpEchoClientApplication:DoDispose(0x55af6719a890)
UdpEchoServerApplication:DoDispose(0x55af6710b000)
UdpEchoClientApplication:~UdpEchoClient(0x55af6719a890)
UdpEchoServerApplication:~UdpEchoServer(0x55af6710b000)
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ 
```

> For WARN messages: **$ ./waf --run "scratch/first.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/first.cc"
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
[2833/2903] Compiling scratch/first.cc
[2863/2903] 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 (2.861s)
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.

4. Simulate traffic between two nodes using pygraphviz + NetAnim. If pyviz is not working, kindly use NetAnim.

Aim: To simulate traffic between two nodes using pygraphviz or NetAnim

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:

> first.cc

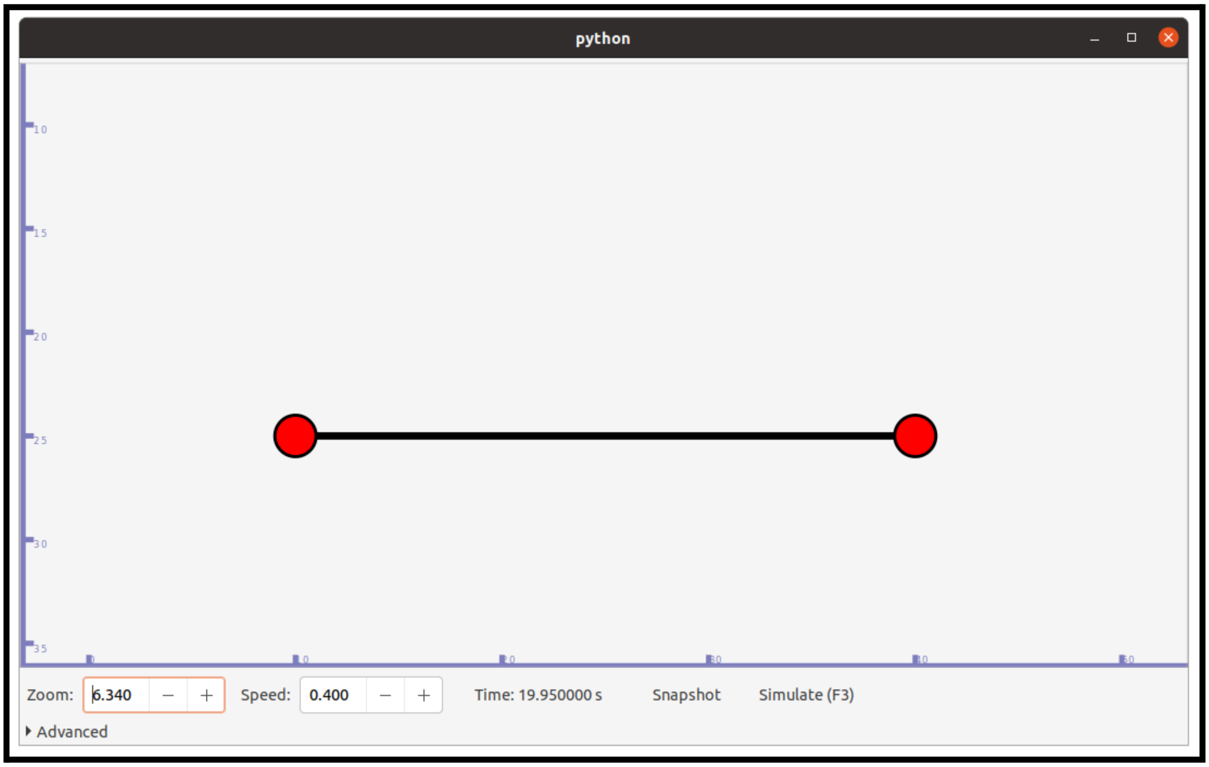
```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/netanim-module.h"
#include "ns3/mobility-helper.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");

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

  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);
```

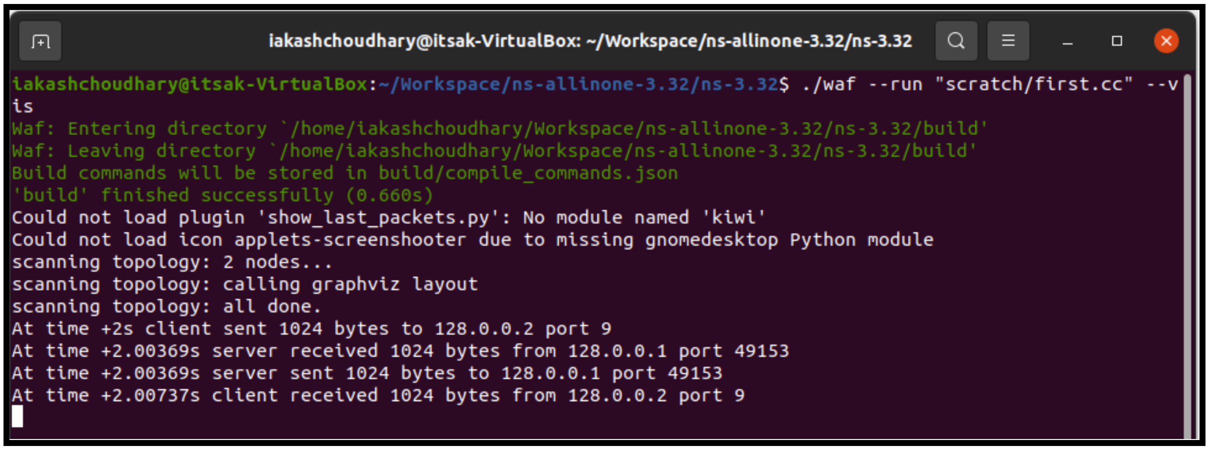
```
Time::SetResolution (Time::NS);
LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

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 ("128.0.0.0", "255.255.0.0");

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 (nPackets));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UintegerValue (nSize));

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

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

AnimationInterface anim("first.xml");
AnimationInterface::SetConstantPosition(nodes.Get(0), 10, 25);
AnimationInterface::SetConstantPosition(nodes.Get(1), 40, 25);
```

```
  anim.EnablePacketMetadata(true);

  // Creating pcap files for Wireshark-pcap: packet capture information
  pointToPoint.EnablePcapAll("first");

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

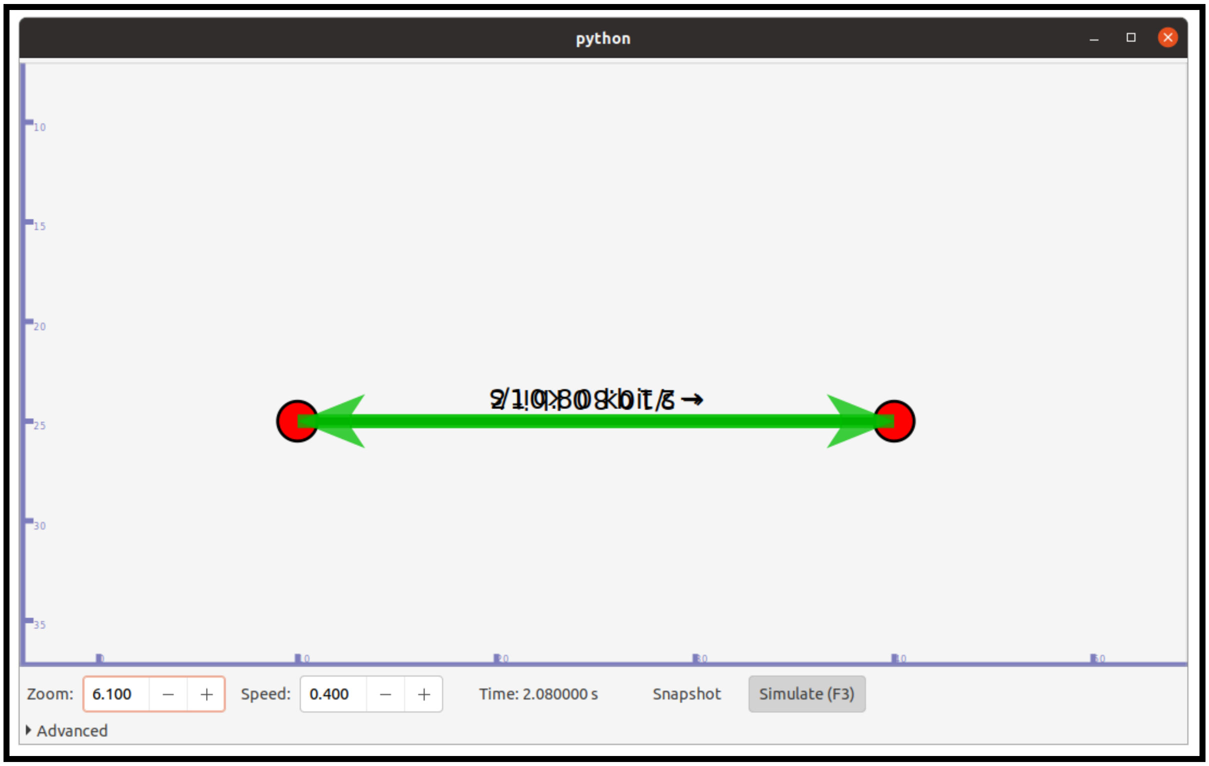
Command & Screenshot:

> Simulate using pygraphviz: **$ ./waf --run "scratch/first.cc" --vis**



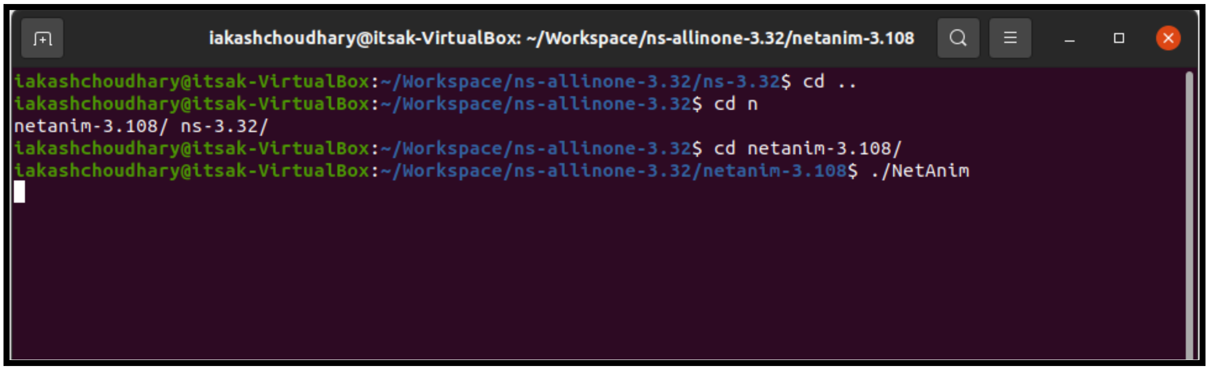

iakashchoudhary@itsak-VirtualBox: ~/Workspace/ns-allinone-3.32/ns-3.32

```
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ ./waf --run "scratch/first.cc" --v
is
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'
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 (0.660s)
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: 2 nodes...
scanning topology: calling graphviz layout
scanning topology: all done.
At time +2s client sent 1024 bytes to 128.0.0.2 port 9
At time +2.00369s server received 1024 bytes from 128.0.0.1 port 49153
At time +2.00369s server sent 1024 bytes to 128.0.0.1 port 49153
At time +2.00737s client received 1024 bytes from 128.0.0.2 port 9
```

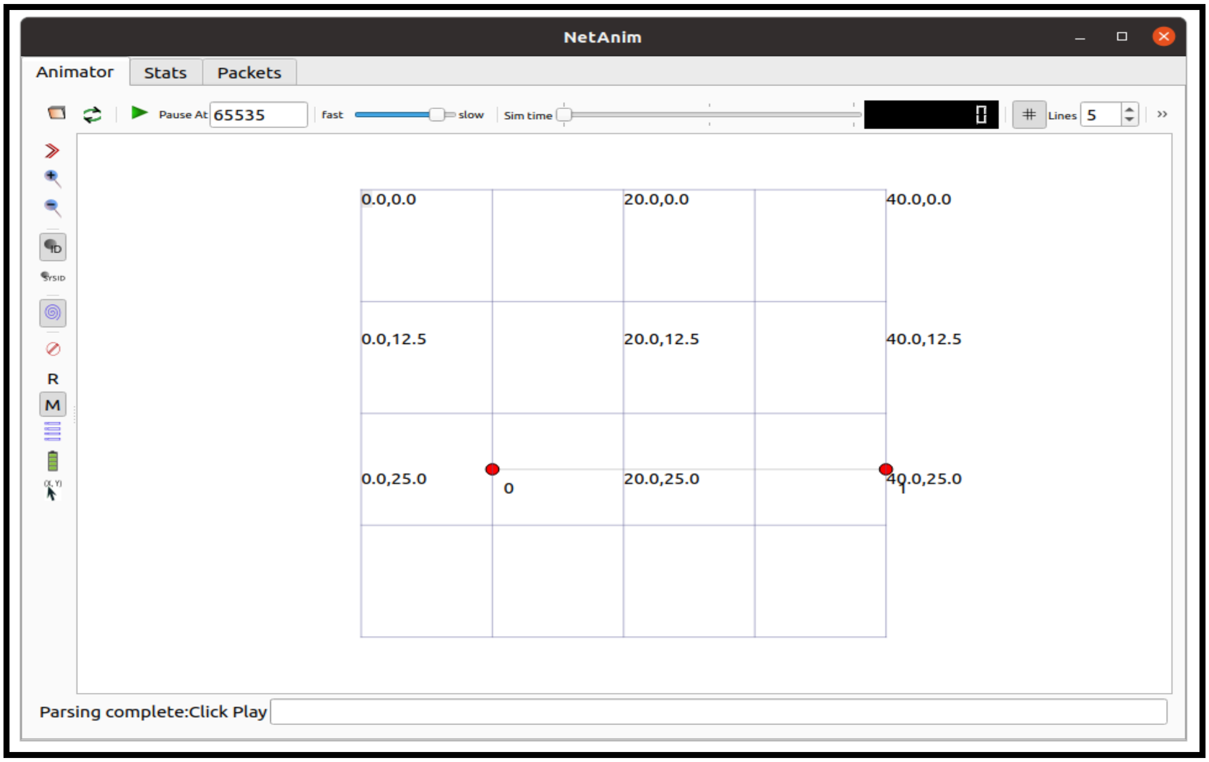


Zoom: 6.100 Speed: 0.400 Time: 2.080000 s

> Simulate using NetAnim: **$ ./NetAnim**


iakashchoudhary@itsak-VirtualBox: ~/Workspace/ns-allinone-3.32/netanim-3.108

```
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ cd ..
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32$ cd n
netanim-3.108/ ns-3.32/
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32$ cd netanim-3.108/
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/netanim-3.108$ ./NetAnim
```

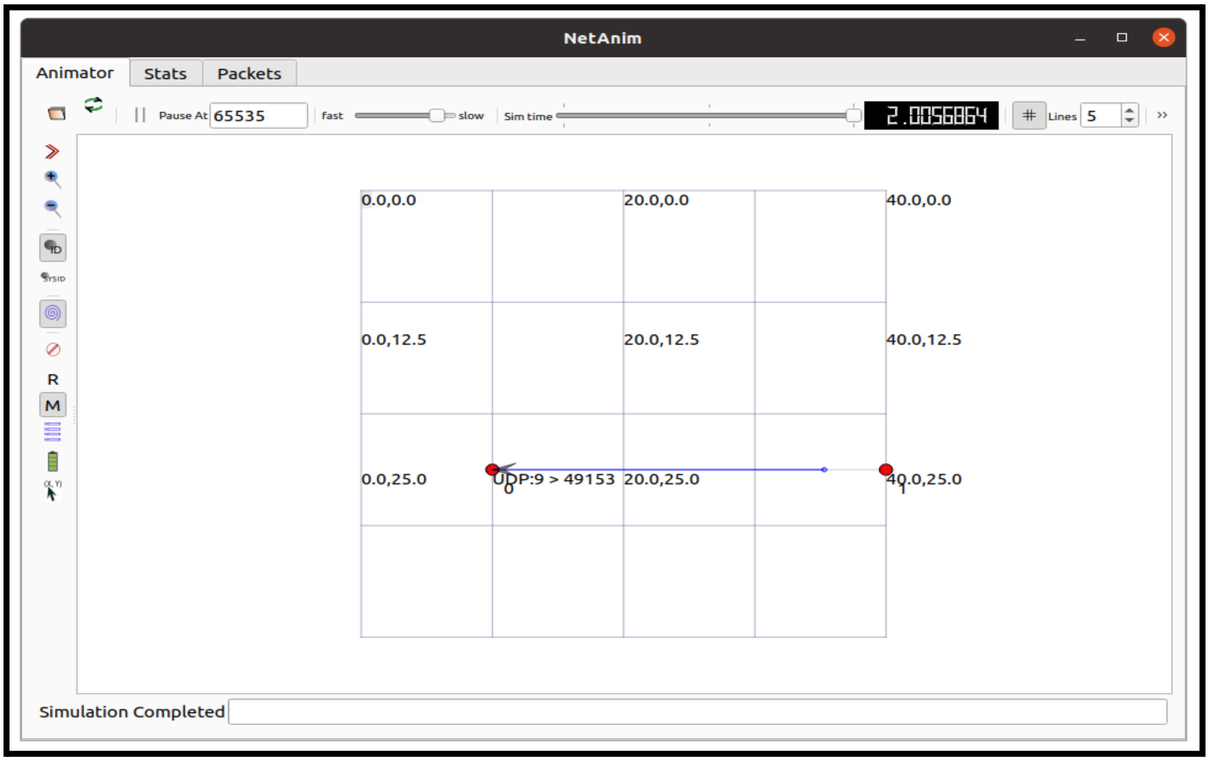


Pause At 65535

Sim time 0

Lines 5

Parsing complete:Click Play



Pause At 65535

Sim time 2.0056864

Lines 5

Simulation Completed

Conclusion: Simulating traffic between two nodes using pygraphviz or NetAnim alone provides valuable insights into network behavior and performance.