| **Name of Student:** Ajay Karthikesan | | | |
| --- | --- | --- | --- |
| **Roll Number:** 57 | | **Assignment Number:** 8 | |
| **Aim of Assignment:**  Simulate a simple network in Network Simulator | | | |
| **DOP:** 30/05/2023 | | **DOS:** 06/06/2023 | |
| **CO Mapped:**  CO2, CO4 | **PO Mapped:**  PO1, PO2,PO4 PO3, PO5, PO7, PO9, PO11, PSO1, PSO2 | **Faculty Signature:** | **Marks:** |

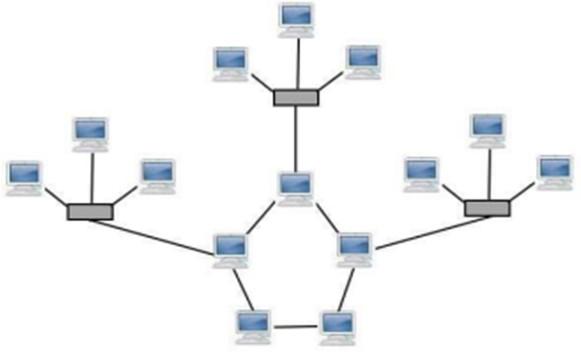
## 

## Practical No. 8

**Aim:** Simulate a simple network in Network Simulator

**Theory:**

**Understanding Hybrid Topology (Wireless Network Topology):**

A hybrid topology is a kind of network topology that is a combination of two or more network topologies, such as mesh topology, bus topology, and ring topology. Its usage and choice are dependent on its deployments and requirements like the performance of the desired network, and the number of computers, their location. The below figure is describing the structure of hybrid topology that contains more than one topology.



However, a variety of technologies are needed for its physical implementation, and it offers a complex structure. Also, it includes an advantage as increasing flexibility; it can increase fault tolerance, and allows new basic topologies to be added or removed easily. The hybrid topology is more useful when you need to fulfil diversity in Computer Network. In this topology, all network sections can include the configuration of different Network Topology.

For instance, you can have a Hybrid network made by two different networks Star Backbone and the Ring Network. You can also use the Star Mesh Hybrid Topology in which if the main backbone gets fail, the entire network will destroy.

**Code:**

/\*

\* This program is free software; you can redistribute it and/or modify

\* it under the terms of the GNU General Public License version 2 as

\* published by the Free Software Foundation;

\*

\* This program is distributed in the hope that it will be useful,

\* but WITHOUT ANY WARRANTY; without even the implied warranty of

\* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

\* GNU General Public License for more details.

\*

\* You should have received a copy of the GNU General Public License

\* along with this program; if not, write to the Free Software

\* Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

\*/

#include "ns3/animation-interface.h"

#include "ns3/applications-module.h"

#include "ns3/core-module.h"

#include "ns3/csma-module.h"

#include "ns3/internet-module.h"

#include "ns3/ipv4-global-routing-helper.h"

#include "ns3/network-module.h"

#include "ns3/point-to-point-module.h"

// Default Network Topology

//

// 10.1.1.0

// n0 -------------- n1 n2 n3 n4

// point-to-point | | | |

// ================

// LAN 10.1.2.0

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);

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(1));

echoClient.SetAttribute("Interval", TimeValue(Seconds(1.0)));

echoClient.SetAttribute("PacketSize", UintegerValue(1024));

ApplicationContainer clientApps = echoClient.Install(p2pNodes.Get(0));

clientApps.Start(Seconds(2.0));

clientApps.Stop(Seconds(10.0));

Ipv4GlobalRoutingHelper::PopulateRoutingTables();

pointToPoint.EnablePcapAll("prac8");

csma.EnablePcap("prac8", csmaDevices.Get(1), true);

AnimationInterface{"prac8\_out.xml"};

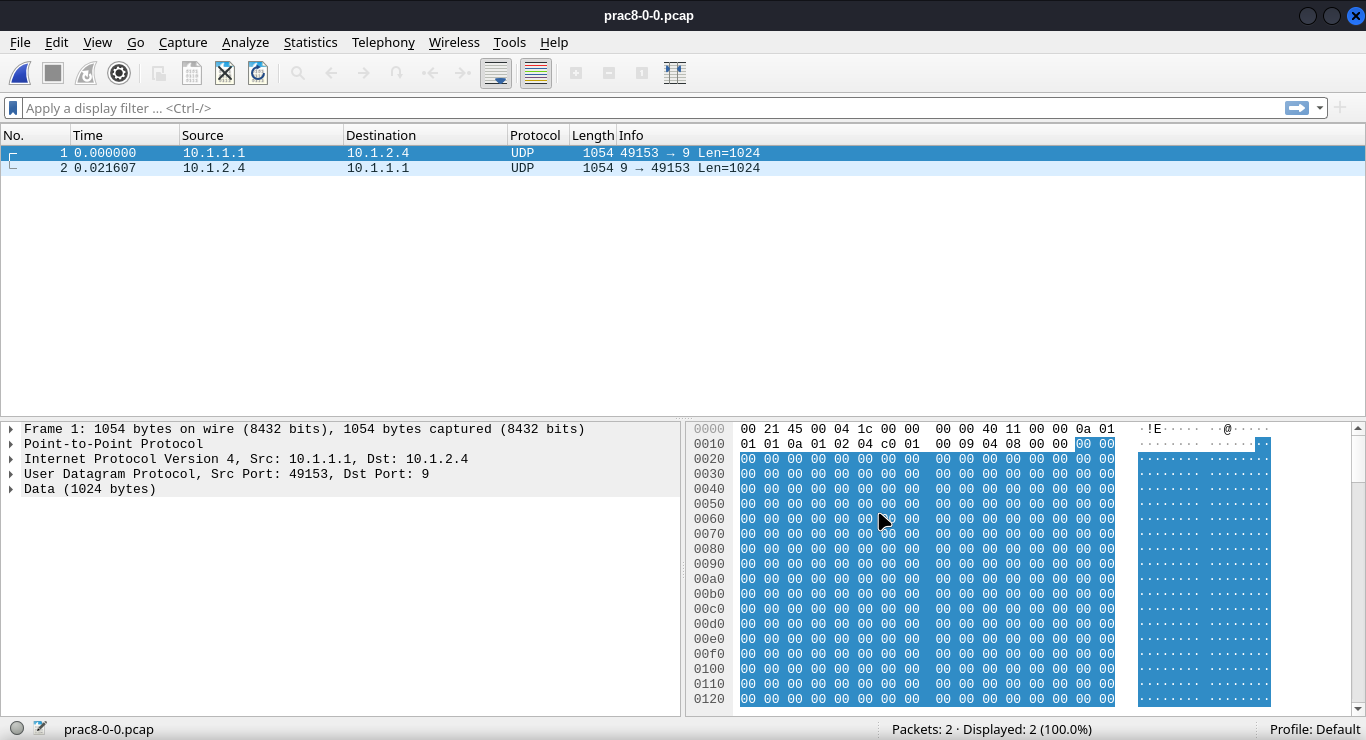
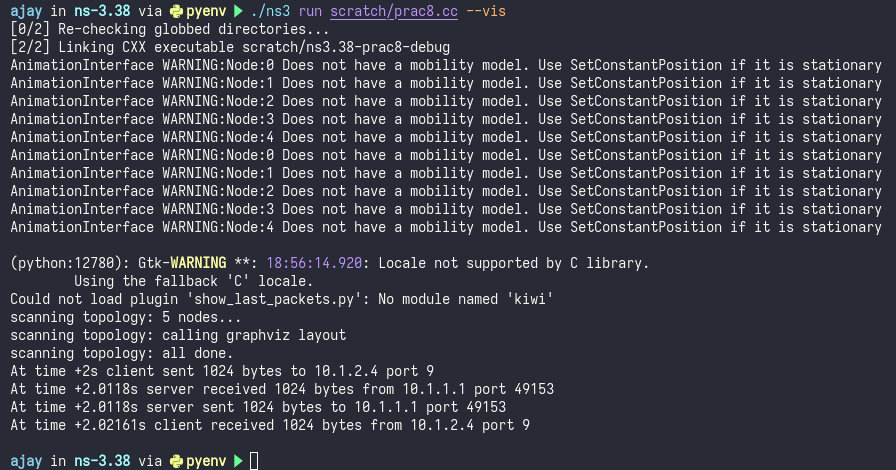
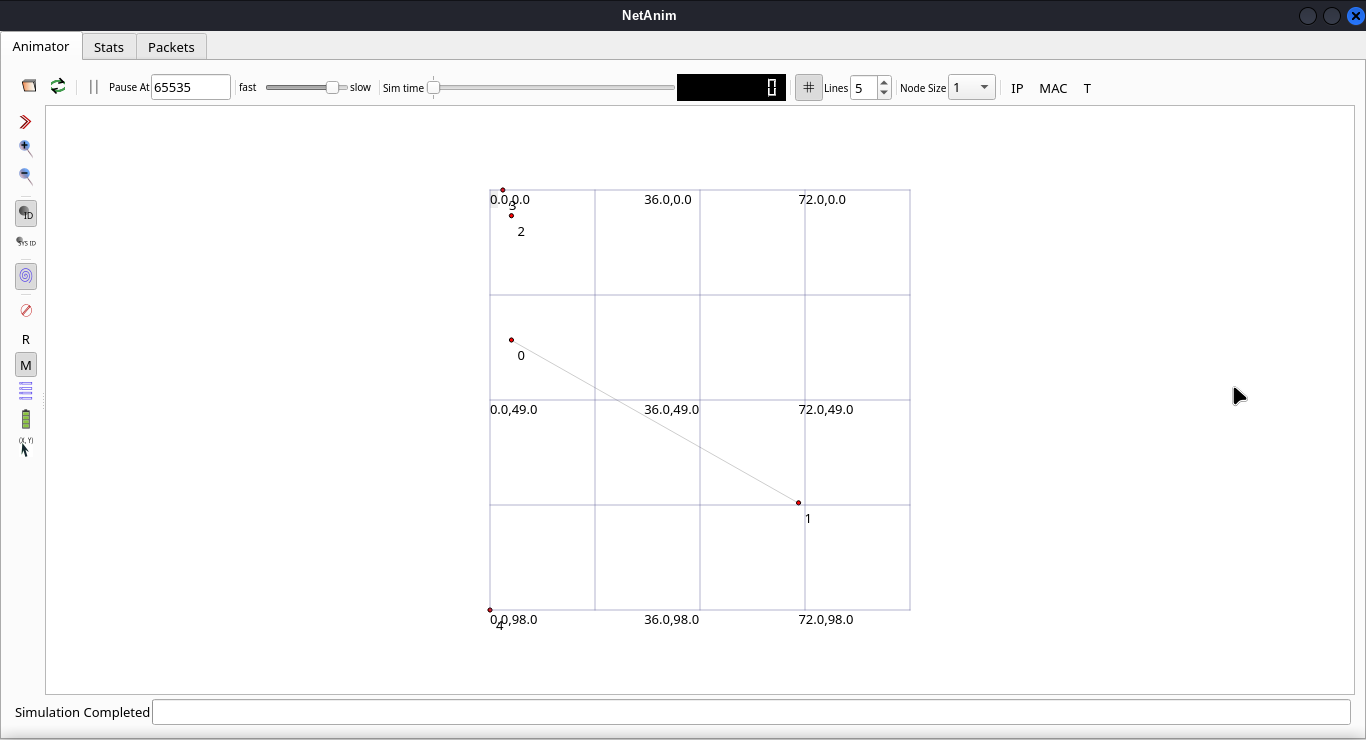
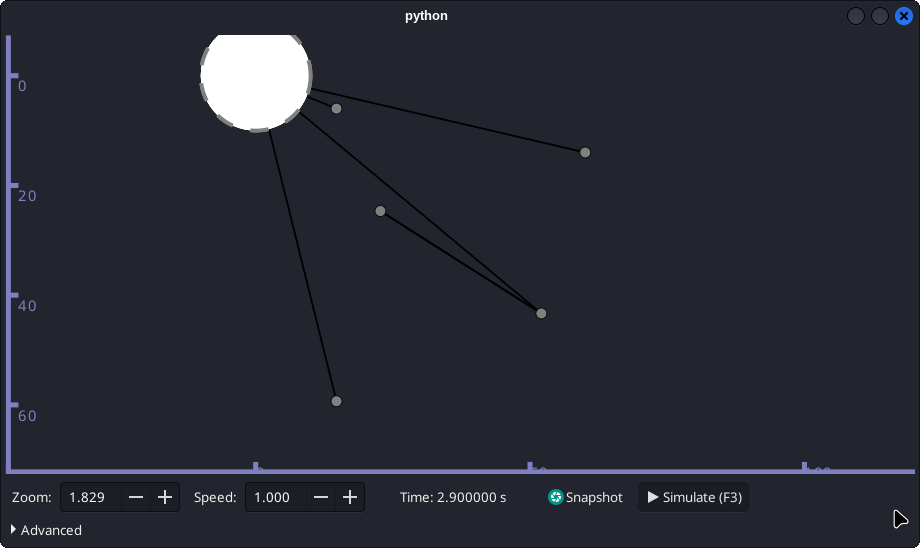
Simulator::Run();

Simulator::Destroy();

return 0;

}

**Output:**



**Conclusion:**

I learnt how to simulate a simple network in Network Simulator.