**PART -A**

1. Implement three nodes point – to – point network with duplex links between them. Set the queue size, vary the bandwidth and find the number of packets dropped.

|  |  |
| --- | --- |
| Network topology |  |
| 10.1.1.0 | 10.1.2.0 |
| n0 -------------- n1.......... | n2 |
| point-to-point |  |

In this program we have created 3 point to point nodes n0, n1, n2. Node n0 has IP address 10.1.1.1 and n3 has 10.1.2.2. Node n1 has 2 interfaces(10.1.1.2 and 10.1.2.1). OnOffHelper application is used to generate the traffic at source node n0. Packets move from n0 to n2 via n1. Acknowledgment is sent from n2 to n0 via n1. Details of the flow(Number of packets sent, received and dropped) can be verified by using tracemetrics(lab1.tr file).

Program

#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/traffic-control-module.h"

using namespace ns3;

NS\_LOG\_COMPONENT\_DEFINE ("Lab-Program-1");

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

{

std::string socketType= "ns3::TcpSocketFactory";;

CommandLine *cmd*;

*cmd*.Parse (argc, argv);

NodeContainer *nodes*;

*nodes*.Create (3); //3 point-to-point nodes are created

InternetStackHelper *stack*;

*stack*.Install (nodes); //TCP-IP layer functionality configured on all nodes

//Bandwidth and delay set for the point-to-point channel. Vary these parameters to //see the variation in number of packets sent/received/dropped.

PointToPointHelper *p2p1*;

*p2p1*.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));

*p2p1*.SetChannelAttribute ("Delay", StringValue ("1ms"));

//Set the base address for the first network(nodes n0 and n1)

Ipv4AddressHelper *address*;

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

NetDeviceContainer *devices*;

*devices* = p2p1.Install (nodes.Get (0), nodes.Get (1));Ipv4InterfaceContainer interfaces = address.Assign (devices);

//Set the base address for the second network(nodes n1 and n2)

devices = *p2p1*.Install (nodes.Get (1), nodes.Get (2)); address.SetBase ("10.1.2.0", "255.255.255.0"); interfaces = address.Assign (devices);

//RateErrorModel allows us to introduce errors into a Channel at a given *rate*. //Vary the error rate value to see the variation in number of packets dropped

Ptr<RateErrorModel>*em* = CreateObject<RateErrorModel> (); em->SetAttribute ("ErrorRate", DoubleValue (0.00002));

devices.Get (1)->SetAttribute ("ReceiveErrorModel", PointerValue (em));

//create routing table at all nodes Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

uint32\_t payloadSize = 1448;

OnOffHelper *onoff* (socketType, Ipv4Address::GetAny ());

//Generate traffic by using OnOff application

*onoff*.SetAttribute ("OnTime", StringValue

("ns3::ConstantRandomVariable[Constant=1]"));

*onoff*.SetAttribute ("OffTime", StringValue

("ns3::ConstantRandomVariable[Constant=0]"));

*onoff*.SetAttribute ("PacketSize", UintegerValue (payloadSize));

*onoff*.SetAttribute ("DataRate", StringValue ("50Mbps")); //bit/s

uint16\_t port = 7;

//Install receiver (for packetsink) on node 2

Address *localAddress1* (InetSocketAddress (Ipv4Address::GetAny (), port));

PacketSinkHelper packetSinkHelper1 (socketType, localAddress1);

ApplicationContainer sinkApp1 = packetSinkHelper1.Install (nodes.Get (2));

sinkApp1.Start (Seconds (0.0));

sinkApp1.Stop (Seconds (10));

//Install sender app on node 0

ApplicationContainer *apps*;

AddressValue remoteAddress (InetSocketAddress (interfaces.GetAddress (1), port)); *onoff*.SetAttribute ("Remote", remoteAddress);

*apps*.Add (onoff.Install (nodes.Get (0))); *apps*.Start (Seconds (1.0)); *apps*.Stop (Seconds (10));

Simulator::Stop (Seconds (10));

AsciiTraceHelper ascii;

*p2p1*.EnableAsciiAll (ascii.CreateFileStream ("lab1.tr"));

//Run the simulator

Simulator::Run ();

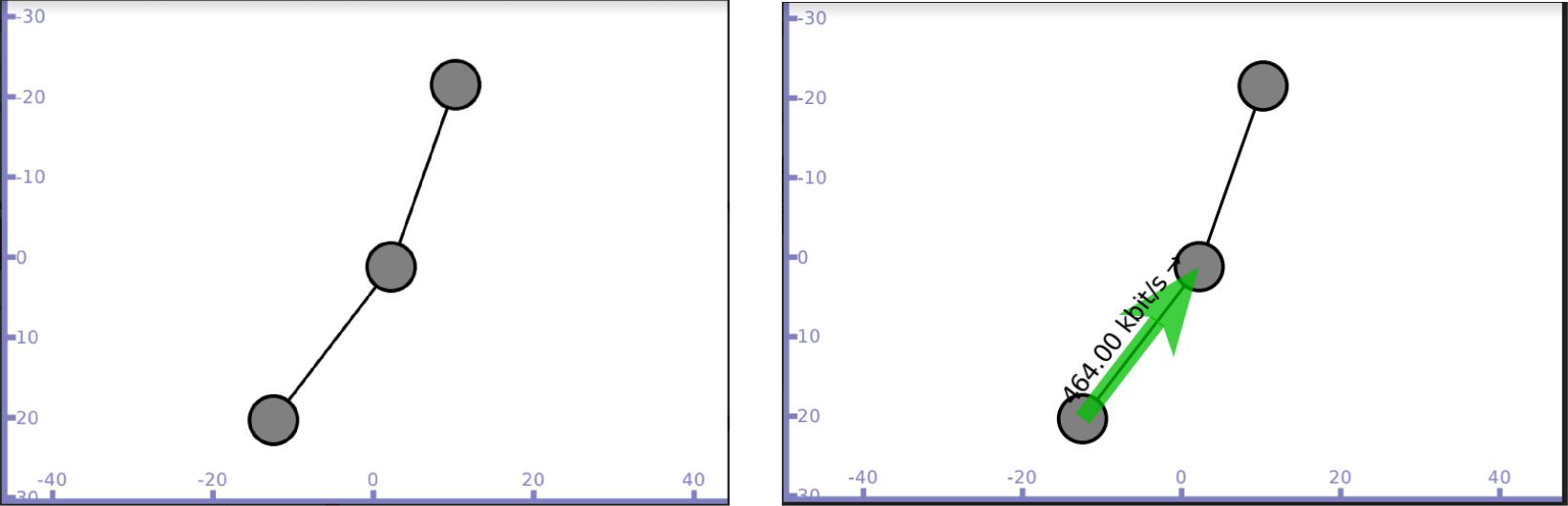
Simulator::Destroy ();

return 0;

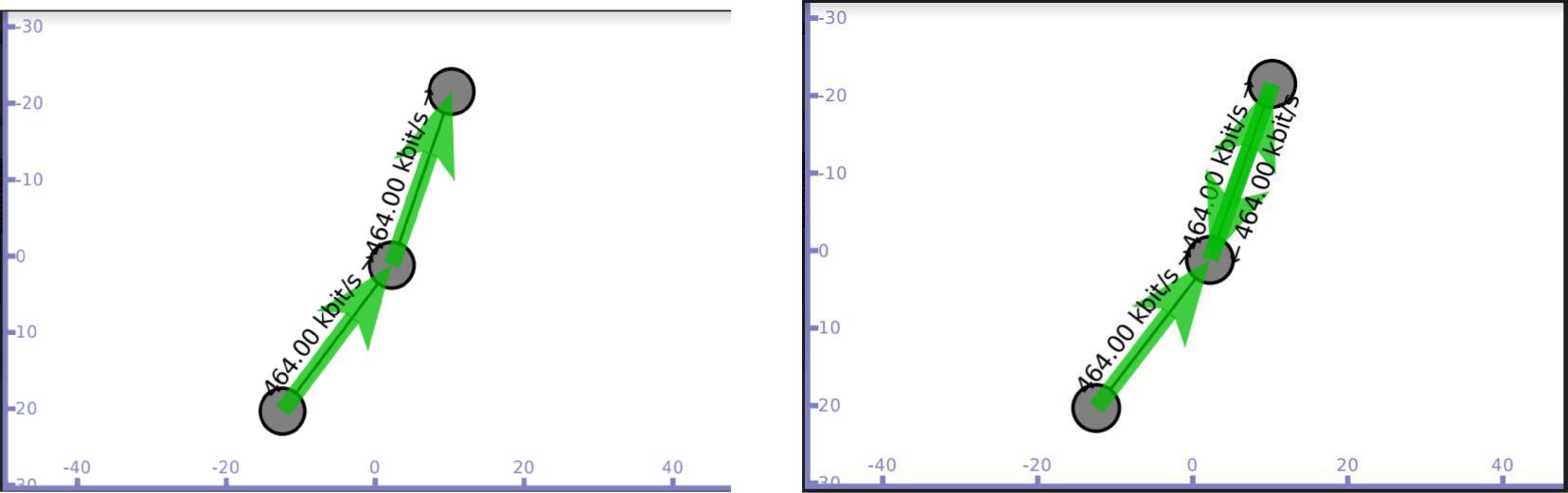
}

./waf - - run scratch/Program1 - -vis

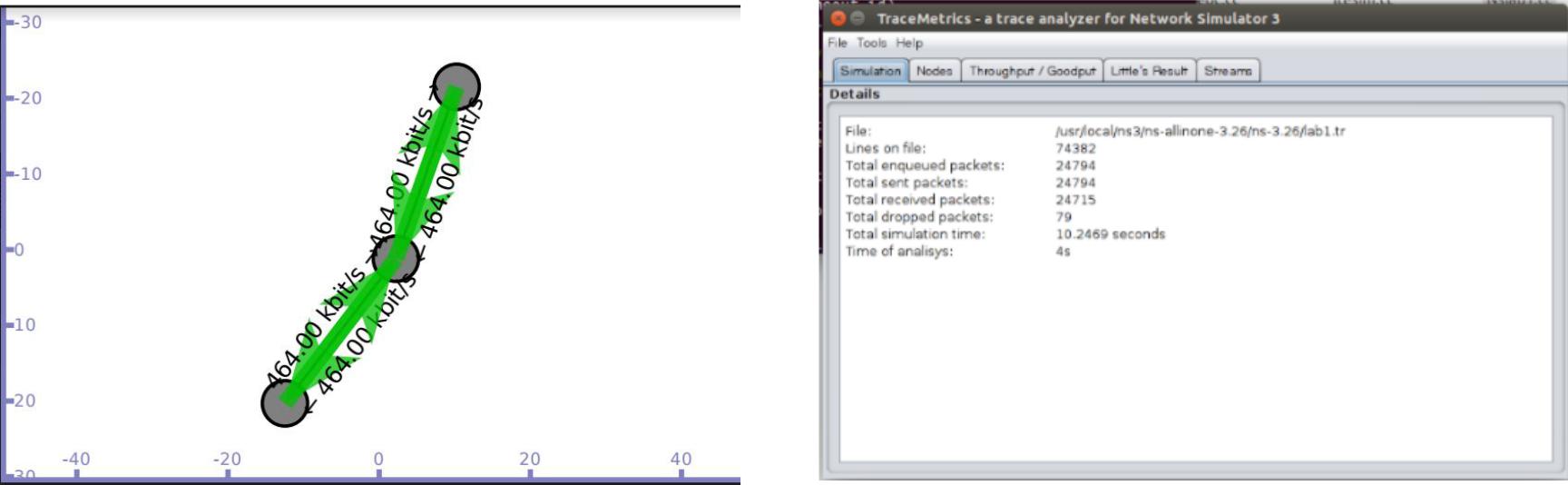
Output



Packet sent from n0 to n1 and then to n2 Acknowledgment sent from n2



Flow details on trace file lab1.tr



1. Implement transmission of ping messages/trace route over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

Network topology

n0 n1 n2 n3 n4 n5

| | | | | |

===========================

CSMA channel with base IP 10.1.1.0

In this program we have created 6 CSMA nodes n0, n1, n2, n3, n4 and n5 with IP addresses 10.1.1.1, 10.1.1.2, 10.1.1.3, 10.1.1.4, 10.1.1.5 and 10.1.1.6 respectively.

Nodes n0 and n1 ping node n2, we can visualize the ping messages transferred between the nodes. Data transfer is also simulated between the nodes n0 and n2 using UdpSocketFactory to generate traffic.

Program

#include <iostream>

#include "ns3/core-module.h"

#include "ns3/network-module.h"

#include "ns3/csma-module.h"

#include "ns3/applications-module.h"

#include "ns3/internet-apps-module.h"

#include "ns3/internet-module.h"

using namespace ns3;

NS\_LOG\_COMPONENT\_DEFINE ("Lab-Program-2");

static void PingRtt (std::string context, Time rtt)

{

std::cout << context <<""<< rtt << std::endl;

}

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

{

CommandLine *cmd*;

*cmd*.Parse (argc, argv);

* Here, we will explicitly create six nodes. NS\_LOG\_INFO ("Create nodes."); NodeContainer *c*;

*c*.Create (6);

* connect all our nodes to a shared channel. NS\_LOG\_INFO ("Build Topology."); CsmaHelper *csma*;

*csma*.SetChannelAttribute ("DataRate", DataRateValue (DataRate (10000))); *csma*.SetChannelAttribute ("Delay", TimeValue (MilliSeconds (0.2)));NetDeviceContainer *devs* = *csma*.Install (c);

* add an ip stack to all nodes.

NS\_LOG\_INFO ("Add ip stack.");

InternetStackHelper *ipStack*;

*ipStack*.Install (c);

// assign ip addresses

NS\_LOG\_INFO ("Assign ip addresses.");

Ipv4AddressHelper *ip*;

*ip*.SetBase ("192.168.1.0", "255.255.255.0");Ipv4InterfaceContainer *addresses* = ip.Assign (devs);

NS\_LOG\_INFO ("Create Sink.");

* Create an OnOff application to send UDP datagrams from node zero to node 1. NS\_LOG\_INFO ("Create Applications.");

uint16\_t port = 9; // Discard port (RFC 863)

OnOffHelper *onoff* ("ns3::UdpSocketFactory",

Address (InetSocketAddress (addresses.GetAddress (2), port))); *onoff*.SetConstantRate (DataRate ("500Mb/s"));

ApplicationContainer *app* = onoff.Install (c.Get (0));

* Start the application *app*.Start (Seconds (6.0)); *app*.Stop (Seconds (10.0));
* Create an optional packet sink to receive these packets PacketSinkHelper *sink* ("ns3::UdpSocketFactory",

Address (InetSocketAddress (Ipv4Address::GetAny (), port))); *app* = *sink*.Install (c.Get (2));

*app*.Start (Seconds (0.0));

NS\_LOG\_INFO ("Create pinger");

V4PingHelper *ping* = V4PingHelper (addresses.GetAddress (2));

NodeContainer *pingers*;

*pingers*.Add (c.Get (0));

*pingers*.Add (c.Get (1));

ApplicationContainer *apps*;

*apps* = *ping*.Install (*pingers*);

*apps*.Start (Seconds (1.0));

*apps*.Stop (Seconds (5.0));

// finally, print the ping rtts.

Config::Connect ("/NodeList/\*/ApplicationList/\*/$ns3::V4Ping/Rtt", MakeCallback (&PingRtt));

NS\_LOG\_INFO ("Run Simulation.");

AsciiTraceHelper ascii;

*csma*.EnableAsciiAll (ascii.CreateFileStream ("ping1.tr"));

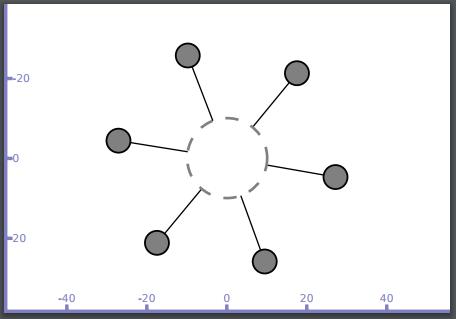
Simulator::Run ();

Simulator::Destroy ();

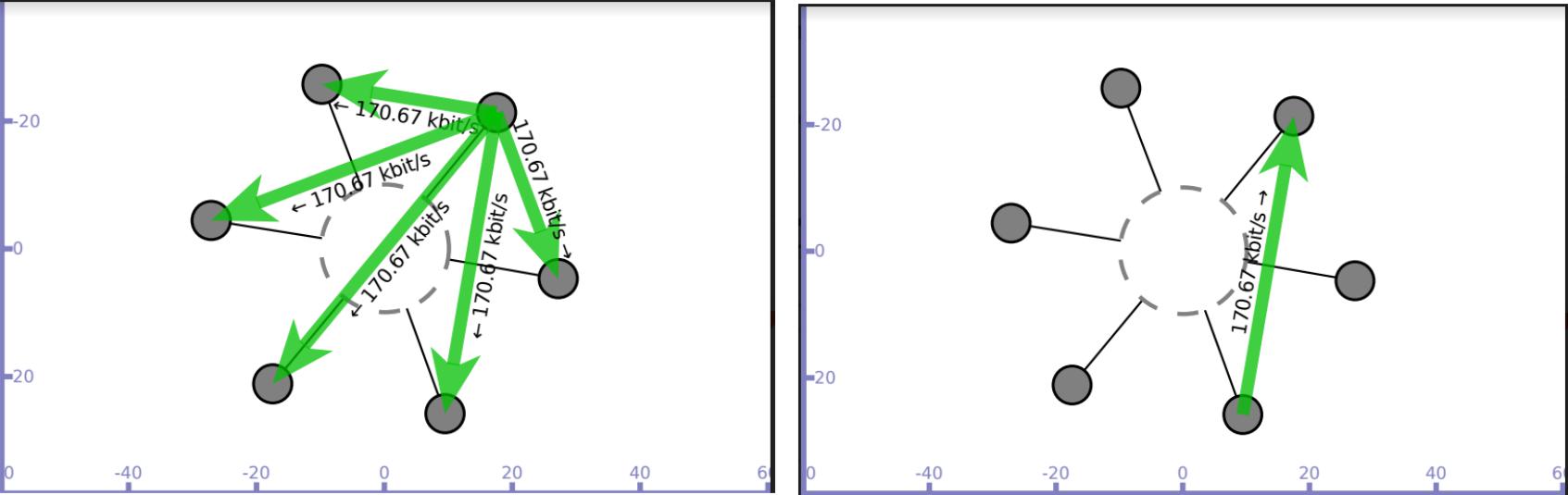
NS\_LOG\_INFO ("Done.");

}

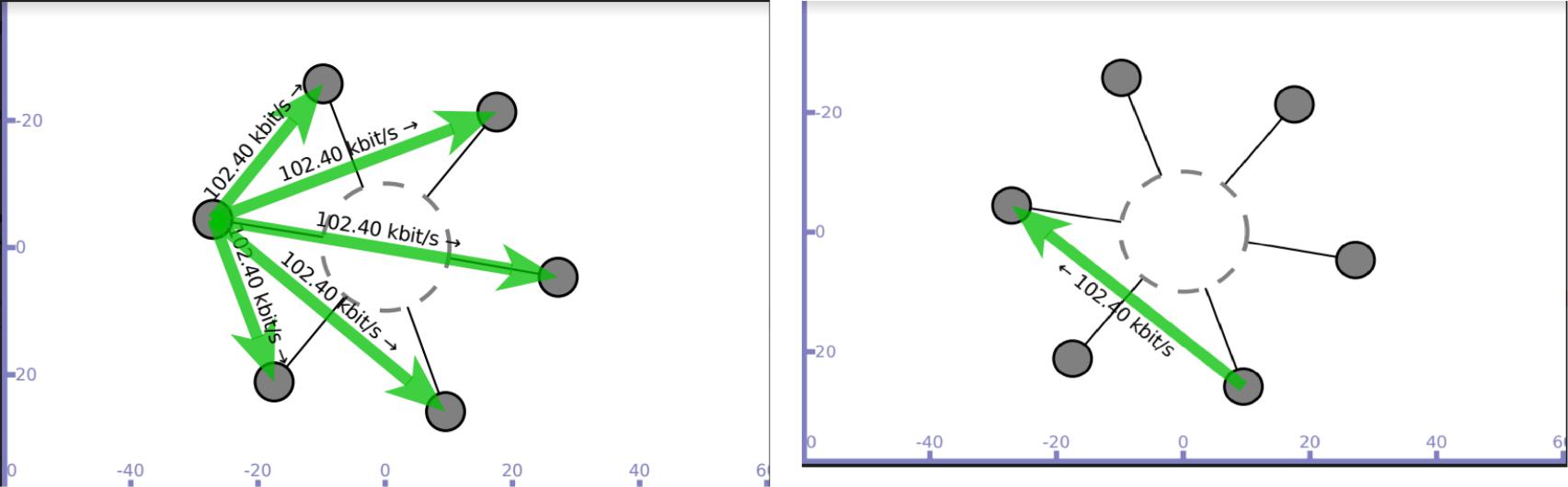
./waf - - run scratch/Program2 - -vis



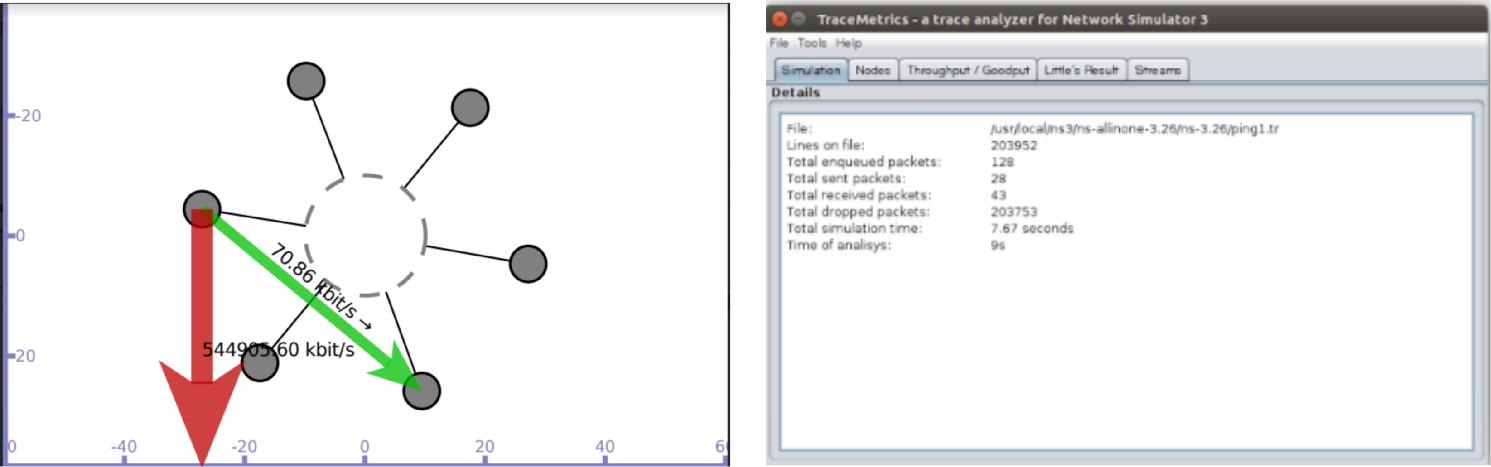
Node n1 sends ping message to n2(Broadcast message is generated) and only n2 responds to n1



Node n0 sends ping message to n2(Broadcast message is generated) and only n2 responds to n0



Data transfer simulated between nodes n0 and n2 Trace file(ping1.tr) generated



1. Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.

Network topology

n0 n1 n2 n3

| | | |

===================== CSMA channel with base IP 10.1.1.0

Source node – n0 sink node - n1

In this program we have created 4 CSMA nodes n0, n1, n2 and n3 with IP addresses 10.1.1.1, 10.1.1.2, 10.1.1.3 and 10.1.1.4 respectively. Data transmission is simulated between nodes n0 and n1. Once the cwnd values are generated, they are exported to .dat file and congestion graph is plot using gnuplot.

Program

#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 <iostream>

#include "ns3/csma-module.h"

using namespace ns3;

NS\_LOG\_COMPONENT\_DEFINE ("3rd Lab Program");

//MyApp class inherits the ns-3 Application class defined in //src/network/model/application.h.

//The MyApp class is obligated to override the StartApplication and //StopApplication methods. These methods are automatically called when MyApp is //required to start and stop sending data during the simulation.

class MyApp : public Application

{

public:

MyApp ();

virtual ~MyApp();

void Setup (Ptr<Socket> socket, Address address, uint32\_t packetSize, uint32\_t nPackets, DataRate dataRate);

private:

virtual void StartApplication (void);

virtual void StopApplication (void);

void ScheduleTx (void);

void SendPacket (void);

Ptr<Socket> m\_socket;

Address m\_peer;

uint32\_t m\_packetSize;

uint32\_t m\_nPackets;

DataRate m\_dataRate;

EventId m\_sendEvent;

bool m\_running;

uint32\_t m\_packetsSent;

};

MyApp::MyApp () // constructor

* m\_socket (0), m\_peer (), m\_packetSize (0), m\_nPackets (0), m\_dataRate (0), m\_sendEvent (), m\_running (false), m\_packetsSent (0)

{

}

MyApp::~MyApp() // destructor

{

m\_socket = 0;

}

// initialize member variables.

void MyApp::Setup (Ptr<Socket> socket, Address address, uint32\_t packetSize,

uint32\_t nPackets, DataRate dataRate)

{

m\_socket = socket;

m\_peer = address;

m\_packetSize = packetSize;

m\_nPackets = nPackets;

m\_dataRate = dataRate;

}

* Below code is the overridden implementation of Application::StartApplication. It //does a socket bind operation and establishes TCP connection with the address //specified in m\_peer.

void MyApp::StartApplication (void)

{

m\_running = true;

m\_packetsSent = 0;

m\_socket->Bind ();

m\_socket->Connect (m\_peer);

SendPacket ();

}

//The next bit of code explains to the Application how to stop creating simulation //events.

void MyApp::StopApplication (void)

{

m\_running = false;

if (m\_sendEvent.IsRunning ())

{

Simulator::Cancel (m\_sendEvent);

}

if (m\_socket)

{

m\_socket->Close ();

}

}

//StartApplication calls SendPacket to start the chain of events that describes the //Application behavior.

void MyApp::SendPacket (void)

{

Ptr<Packet> packet = Create<Packet> (m\_packetSize); m\_socket->Send (packet);

if (++m\_packetsSent < m\_nPackets)

{

ScheduleTx ();

}

}

//It is the responsibility of the Application to keep scheduling the chain of //events, so the next lines call ScheduleTx to schedule another transmit event //(*a SendPacket*) until the Application decides it has sent enough.

void MyApp::ScheduleTx (void)

{

if (m\_running)

{

Time tNext (Seconds (m\_packetSize \* 8 / static\_cast<double> (m\_dataRate.GetBitRate ())));

m\_sendEvent = Simulator::Schedule (tNext, &MyApp::SendPacket, this);

}

}

//Below function logs the current simulation time and the new value of the congestion window every time it is changed.

static void CwndChange (uint32\_t oldCwnd, uint32\_t newCwnd)

{

NS\_LOG\_UNCOND (Simulator::Now ().GetSeconds () <<"\t"<< newCwnd);

}

//trace sink to show where packets are dropped

static void RxDrop (Ptr<const Packet> p)

{

NS\_LOG\_UNCOND ("RxDrop at "<< Simulator::Now ().GetSeconds ());

}

//main function

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

{

CommandLine *cmd*;

*cmd*.Parse (argc, argv);

NS\_LOG\_INFO ("Create nodes.");

NodeContainer *nodes*;

*nodes*.Create (4);//4 csma nodes are created

CsmaHelper *csma*;

*csma*.SetChannelAttribute ("DataRate", StringValue ("5Mbps"));

*csma*.SetChannelAttribute ("Delay", TimeValue (MilliSeconds (0.0001)));

NetDeviceContainer *devices*;

*devices* = *csma*.Install (nodes);

//RateErrorModel allows us to introduce errors into a Channel at a given *rate*.

Ptr<RateErrorModel>*em* = CreateObject<RateErrorModel> (); *em*->SetAttribute ("ErrorRate", DoubleValue (0.00001));

*devices*.Get (1)->SetAttribute ("ReceiveErrorModel", PointerValue (em));

InternetStackHelper *stack*;

*stack*.Install (nodes);

Ipv4AddressHelper *address*;

*address*.SetBase ("10.1.1.0", "255.255.255.0");Ipv4InterfaceContainer *interfaces* = address.Assign (devices);

uint16\_t sinkPort = 8080;

//PacketSink Application is used on the destination node to receive TCP connections //and data.

Address *sinkAddress* (InetSocketAddress (interfaces.GetAddress (1), sinkPort)); PacketSinkHelper *packetSinkHelper* ("ns3::TcpSocketFactory", InetSocketAddress (Ipv4Address::GetAny (), sinkPort));

ApplicationContainer *sinkApps* = packetSinkHelper.Install (nodes.Get (1));

*sinkApps*.Start (Seconds (0.));

*sinkApps*.Stop (Seconds (20.));

//next two lines of code will create the socket and connect the trace source.

Ptr<Socket> ns3TcpSocket = Socket::CreateSocket (nodes.Get (0),

TcpSocketFactory::GetTypeId ());

ns3TcpSocket->TraceConnectWithoutContext ("CongestionWindow", MakeCallback

(&CwndChange));

//creates an Object of type MyApp

Ptr<MyApp> app = CreateObject<MyApp> ();

//tell the Application what Socket to use, what address to connect to, how much //data to send at each send event, how many send events to generate and the rate at //which to produce data from those events.

app->Setup (ns3TcpSocket, sinkAddress, 1040, 1000, DataRate ("50Mbps")); nodes.Get (0)->AddApplication (app); app->SetStartTime (Seconds (1.));

app->SetStopTime (Seconds (20.));

*devices*.Get (1)->TraceConnectWithoutContext ("PhyRxDrop", MakeCallback (&RxDrop));

Simulator::Stop (Seconds (20));

Simulator::Run ();

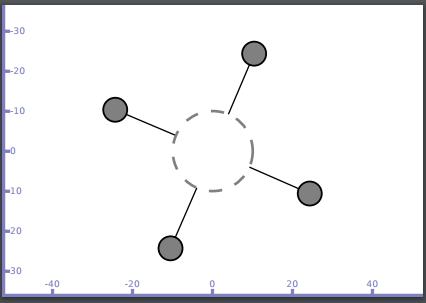
Simulator::Destroy ();

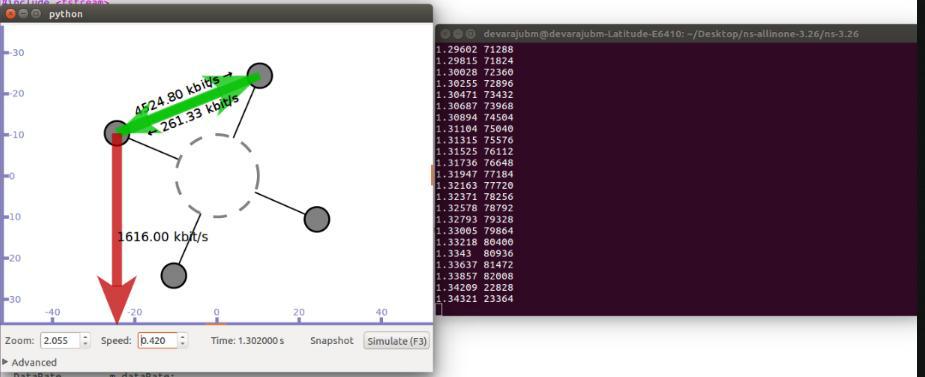
return 0;

}

./waf - - run scratch/Program3 - -vis

Output





Redirect the output to a file called cwnd.dat

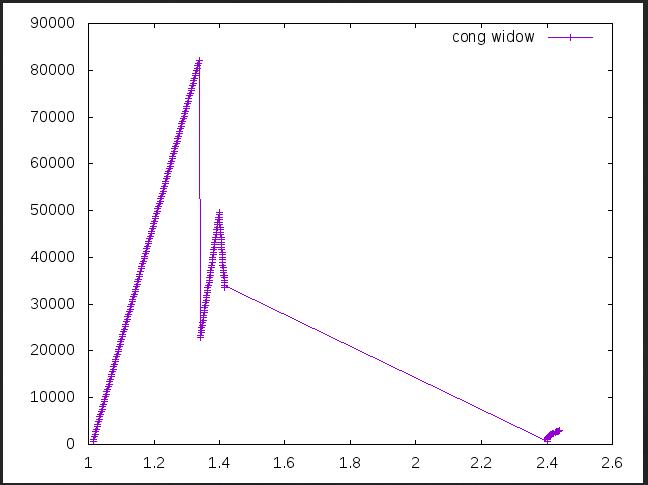
./waf --run scratch/Program3 > cwnd.dat 2>&1

Now run gnuplot

gnuplot> set terminal png size 640,480

gnuplot> set output "cwnd.png"

gnuplot> plot "cwnd.dat" using 1:2 title 'Congestion Window' with linespoints gnuplot> exit

w

n

d

V

a

l

u

e

s

i

n

b

i

ts

Time in seconds

4.Implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to transmission ofpackets.

Default Network Topology

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| | |  |  |  |  |
| Rank | 0 | | Rank 1 | |  |
| ------------------------ | |  |  | |---------------------------- |
| Wifi 10.1.3.0 | | |  |  |
| AP |  |  |  |  |
| \* | \* | \* | \* |  |
| | | | | | | | | 10.1.1.0 |
| n2 | n3 | n4 | n0 | n1 |
| point-to-point | | | | |  |

In this program we have created 3 wifi (STA/mobile)nodes(n2,n3,n4), 2 point to point nodes(n0,n1) where n0 acts as access point n1 is a base station. This program establishes connection between n2(10.1.3.3) and n1(10.1.1.2) through access point(10.1.1.1). The Performance is measured in terms of throughput of the nodes. It can be verified using tracemetrics(Files generated : Tracefilewifides and Tracefilewifisrc).

Program

#include "ns3/core-module.h"

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

#include "ns3/network-module.h"

#include "ns3/applications-module.h"

#include "ns3/wifi-module.h"

#include "ns3/mobility-module.h"

#include "ns3/internet-module.h"

using namespace ns3;

NS\_LOG\_COMPONENT\_DEFINE ("ThirdScriptExample");

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

{

bool verbose = true;

uint32\_t nWifi = 3; // 3 wi-fi nodes are created

CommandLine cmd;

cmd.AddValue ("nWifi", "Number of wifi STA devices", nWifi);

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); }

NodeContainer *p2pNodes*;

*p2pNodes*.Create (2);// 2 nodes are n0,n1 are created

PointToPointHelper *pointToPoint*;

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

NetDeviceContainer *p2pDevices*;

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

NodeContainer *wifiStaNodes*;

*wifiStaNodes*.Create (nWifi);

NodeContainer wifiApNode = *p2pNodes*.Get (0);// 1st node of p2p is also access point

* default PHY layer configuration is used for wifi YansWifiChannelHelper *channel* = YansWifiChannelHelper::Default (); YansWifiPhyHelper *phy* = YansWifiPhyHelper::Default (); *phy*.SetChannel (channel.Create ());

WifiHelper *wifi*;

*wifi*.SetRemoteStationManager ("ns3::AarfWifiManager");//AARF= rate control algorithm

WifiMacHelper *mac*;

Ssid *ssid* = Ssid ("ns-3-ssid");// ssid=service set identifier in 802.11 *mac*.SetType ("ns3::StaWifiMac",

"Ssid", SsidValue (ssid),

"ActiveProbing", BooleanValue (false));

NetDeviceContainer *staDevices*;

*staDevices* = *wifi*.Install (phy, mac, wifiStaNodes);

*mac*.SetType ("ns3::ApWifiMac","Ssid", SsidValue (ssid));

NetDeviceContainer *apDevices*;

*apDevices* = *wifi*.Install (phy, mac, wifiApNode);

MobilityHelper *mobility*;

* 2 dimensional grid to initially place sta(stationary nodes) *mobility*.SetPositionAllocator ("ns3::GridPositionAllocator","MinX", DoubleValue (10.0),

"MinY", DoubleValue (-10.0), "DeltaX", DoubleValue (7.0), "DeltaY", DoubleValue (12.0), "GridWidth", UintegerValue (3), "LayoutType", StringValue ("RowFirst"));

*mobility*.SetMobilityModel ("ns3::RandomWalk2dMobilityModel","Bounds",RectangleValue (Rectangle (-50, 50, -50, 50))); *mobility*.Install (wifiStaNodes);

*mobility*.SetMobilityModel ("ns3::ConstantPositionMobilityModel"); *mobility*.Install (*wifiApNode*);

InternetStackHelper *stack*;

*stack*.Install (*p2pNodes*.Get(1));// stack installed on n1 of p2p *stack*.Install (*wifiApNode*);//stack installed on access point

*stack*.Install (*wifiStaNodes*);//stack installed on mobile nodes

Ipv4AddressHelper *address*;

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

*address*.SetBase ("10.1.3.0", "255.255.255.0"); *address*.Assign (*staDevices*); *address*.Assign (*apDevices*);

//install echo server application on n1 UdpEchoServerHelper *echoServer* (9);

ApplicationContainer *serverApps* = *echoServer*.Install (p2pNodes.Get (1));

*serverApps*.Start (Seconds (1.0));

*serverApps*.Stop (Seconds (10.0));

//install echo client application on n3

UdpEchoClientHelper *echoClient* (p2pInterfaces.GetAddress (1), 9); *echoClient*.SetAttribute ("MaxPackets", UintegerValue (1)); *echoClient*.SetAttribute ("Interval", TimeValue (Seconds (1.0))); *echoClient*.SetAttribute ("PacketSize", UintegerValue (1024));

ApplicationContainer *clientApps* =

*echoClient*.Install (wifiStaNodes.Get (nWifi - 1)); *clientApps*.Start (Seconds (2.0)); *clientApps*.Stop (Seconds (10.0));

Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

Simulator::Stop (Seconds (10.0));

AsciiTraceHelper ascii;

pointToPoint.EnableAsciiAll (ascii.CreateFileStream ("Tracefilewifides.tr")); phy.EnableAsciiAll (ascii.CreateFileStream ("Tracefilewifisrc.tr"));

Simulator::Run ();

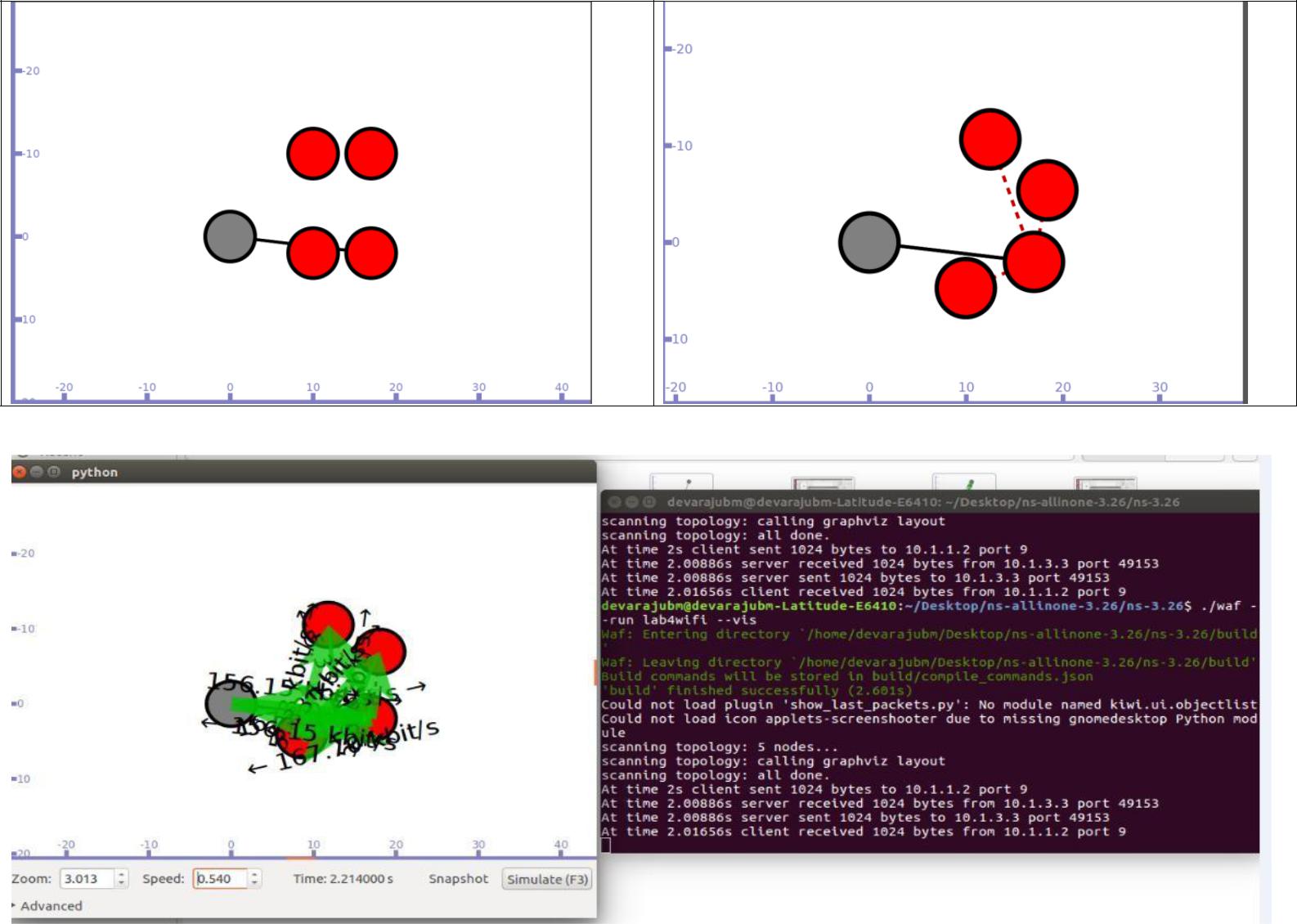
Simulator::Destroy ();

return 0;

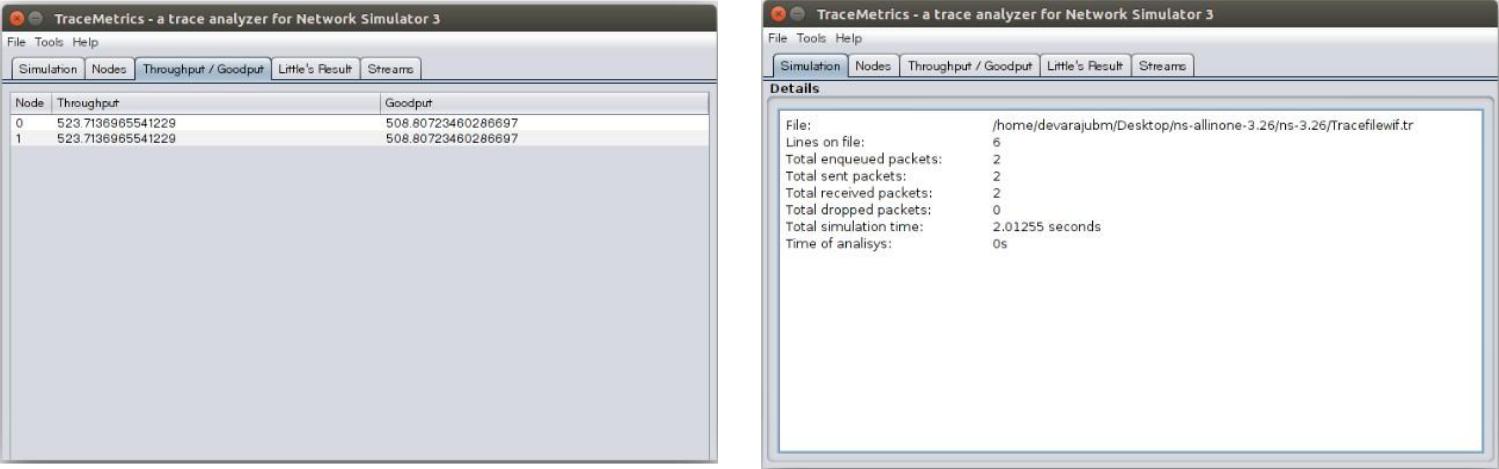
}

./waf - - run scratch/Program4 - -vis

Output



Trace file is used to see the throughput by using TraceMetrics



5. Implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment.

6. Implement and study the performance of CDMA on NS2/NS3 (Using stack called Call net) or equivalent environment.

#include "ns3/lte-helper.h"

#include "ns3/epc-helper.h"

#include "ns3/core-module.h"

#include "ns3/network-module.h"

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

#include "ns3/internet-module.h"

#include "ns3/mobility-module.h"

#include "ns3/lte-module.h"

#include "ns3/applications-module.h"

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

#include "ns3/config-store.h"

//#include "ns3/gtk-config-store.h"

//.............................................................................................................

using namespace ns3;

NS\_LOG\_COMPONENT\_DEFINE ("EpcFirstExample");

int

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

{

uint16\_t numberOfNodes = 2; // numberOfNodes = 6 for CDMA

double simTime = 1.1;

double distance = 60.0;

double interPacketInterval = 100;

// Command line arguments

CommandLine cmd;

cmd.Parse(argc, argv);

Ptr<LteHelper> lteHelper = CreateObject<LteHelper> ();

//This will instantiate some common objects (e.g., the Channel object) and provide the methods to add eNBs and UEs and configure them.

Ptr<PointToPointEpcHelper> epcHelper = CreateObject<PointToPointEpcHelper> ();

//PointToPointEpcHelper, which implements an EPC based on point-to-point links.

//EpcHelper will also automatically create the PGW node and configure it so that it can properly handle traffic from/to the LTE radio access network.

lteHelper->SetEpcHelper (epcHelper);

//Then, you need to tell the LTE helper that the EPC will be used:

ConfigStore inputConfig;

inputConfig.ConfigureDefaults();

//Specify configuration parameters of the objects that are being used for the simulation

// parse again so you can override default values from the command line

cmd.Parse(argc, argv);

Ptr<Node> pgw = epcHelper->GetPgwNode ();

//EpcHelper will also automatically create the PGW node and configure it so that it can properly handle traffic from/to the LTE radio access network.

// Create a single RemoteHost

NodeContainer remoteHostContainer;

remoteHostContainer.Create (1);

Ptr<Node> remoteHost = remoteHostContainer.Get (0);

InternetStackHelper internet;

internet.Install (remoteHostContainer);

// Create the Internet

PointToPointHelper p2ph;

p2ph.SetDeviceAttribute ("DataRate", DataRateValue (DataRate ("100Gb/s")));

p2ph.SetDeviceAttribute ("Mtu", UintegerValue (1500));

p2ph.SetChannelAttribute ("Delay", TimeValue (Seconds (0.010)));

NetDeviceContainer internetDevices = p2ph.Install (pgw, remoteHost);

Ipv4AddressHelper ipv4h;

ipv4h.SetBase ("1.0.0.0", "255.0.0.0");

Ipv4InterfaceContainer internetIpIfaces = ipv4h.Assign (internetDevices);

// interface 0 is localhost, 1 is the p2p device

Ipv4Address remoteHostAddr = internetIpIfaces.GetAddress (1);

Ipv4StaticRoutingHelper ipv4RoutingHelper;

Ptr<Ipv4StaticRouting> remoteHostStaticRouting = ipv4RoutingHelper.GetStaticRouting (remoteHost->GetObject<Ipv4> ());

remoteHostStaticRouting->AddNetworkRouteTo (Ipv4Address ("7.0.0.0"), Ipv4Mask ("255.0.0.0"), 1);

NodeContainer ueNodes;

NodeContainer enbNodes;

enbNodes.Create(numberOfNodes);

ueNodes.Create(numberOfNodes);

// Install Mobility Model

Ptr<ListPositionAllocator> positionAlloc = CreateObject<ListPositionAllocator> ();

for (uint16\_t i = 0; i < numberOfNodes; i++)

{

positionAlloc->Add (Vector(distance \* i, 100, 100));

}

MobilityHelper mobility;

mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");

mobility.SetPositionAllocator(positionAlloc);

mobility.Install(enbNodes);

mobility.Install(ueNodes);

// Install LTE Devices to the nodes

NetDeviceContainer enbLteDevs = lteHelper->InstallEnbDevice (enbNodes);

NetDeviceContainer ueLteDevs = lteHelper->InstallUeDevice (ueNodes);

// Install the IP stack on the UEs

internet.Install (ueNodes);

Ipv4InterfaceContainer ueIpIface;

ueIpIface = epcHelper->AssignUeIpv4Address (NetDeviceContainer (ueLteDevs));

// Assign IP address to UEs, and install applications

for (uint32\_t u = 0; u < ueNodes.GetN (); ++u)

{

Ptr<Node> ueNode = ueNodes.Get (u);

// Set the default gateway for the UE

Ptr<Ipv4StaticRouting> ueStaticRouting = ipv4RoutingHelper.GetStaticRouting (ueNode->GetObject<Ipv4> ());

ueStaticRouting->SetDefaultRoute (epcHelper->GetUeDefaultGatewayAddress (), 1);

}

// Attach one UE per eNodeB

for (uint16\_t i = 0; i < numberOfNodes; i++)

{

lteHelper->Attach (ueLteDevs.Get(i), enbLteDevs.Get(i));

// side effect: the default EPS bearer will be activated

}

// Install and start applications on UEs and remote host

uint16\_t dlPort = 1234;

uint16\_t ulPort = 2000;

uint16\_t otherPort = 3000;

ApplicationContainer clientApps;

ApplicationContainer serverApps;

for (uint32\_t u = 0; u < ueNodes.GetN (); ++u)

{

++ulPort;

++otherPort;

PacketSinkHelper dlPacketSinkHelper ("ns3::UdpSocketFactory", InetSocketAddress (Ipv4Address::GetAny (), dlPort));

PacketSinkHelper ulPacketSinkHelper ("ns3::UdpSocketFactory", InetSocketAddress (Ipv4Address::GetAny (), ulPort));

PacketSinkHelper packetSinkHelper ("ns3::UdpSocketFactory", InetSocketAddress (Ipv4Address::GetAny (), otherPort));

serverApps.Add (dlPacketSinkHelper.Install (ueNodes.Get(u)));

serverApps.Add (ulPacketSinkHelper.Install (remoteHost));

serverApps.Add (packetSinkHelper.Install (ueNodes.Get(u)));

UdpClientHelper dlClient (ueIpIface.GetAddress (u), dlPort);

dlClient.SetAttribute ("Interval", TimeValue (MilliSeconds(interPacketInterval)));

dlClient.SetAttribute ("MaxPackets", UintegerValue(1000000));

UdpClientHelper ulClient (remoteHostAddr, ulPort);

ulClient.SetAttribute ("Interval", TimeValue (MilliSeconds(interPacketInterval)));

ulClient.SetAttribute ("MaxPackets", UintegerValue(1000000));

UdpClientHelper client (ueIpIface.GetAddress (u), otherPort);

client.SetAttribute ("Interval", TimeValue (MilliSeconds(interPacketInterval)));

client.SetAttribute ("MaxPackets", UintegerValue(1000000));

clientApps.Add (dlClient.Install (remoteHost));

clientApps.Add (ulClient.Install (ueNodes.Get(u)));

if (u+1 < ueNodes.GetN ())

{

clientApps.Add (client.Install (ueNodes.Get(u+1)));

}

else

{

clientApps.Add (client.Install (ueNodes.Get(0)));

}

}

serverApps.Start (Seconds (0.01));

clientApps.Start (Seconds (0.01));

lteHelper->EnableTraces ();

// Uncomment to enable PCAP tracing

AsciiTraceHelper ascii;

p2ph.EnableAsciiAll(ascii.CreateFileStream("lab5.tr");

p2ph.EnablePcapAll("lena-epc-first");

Simulator::Stop(Seconds(simTime));

Simulator::Run();

/\*GtkConfigStore config;

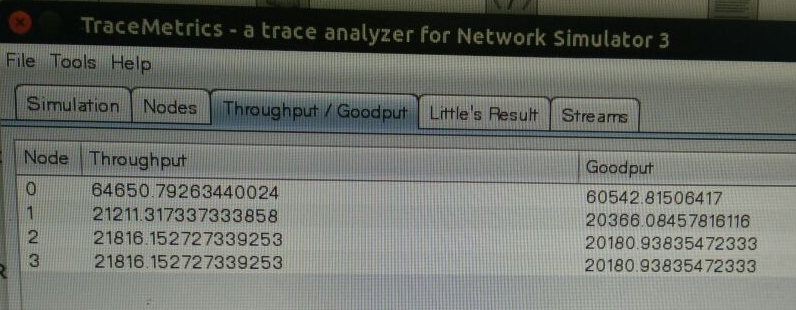
config.ConfigureAttributes();\*/

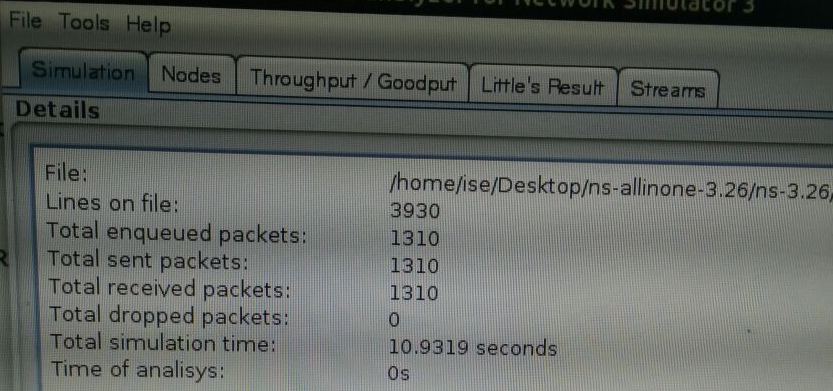
Simulator::Destroy();

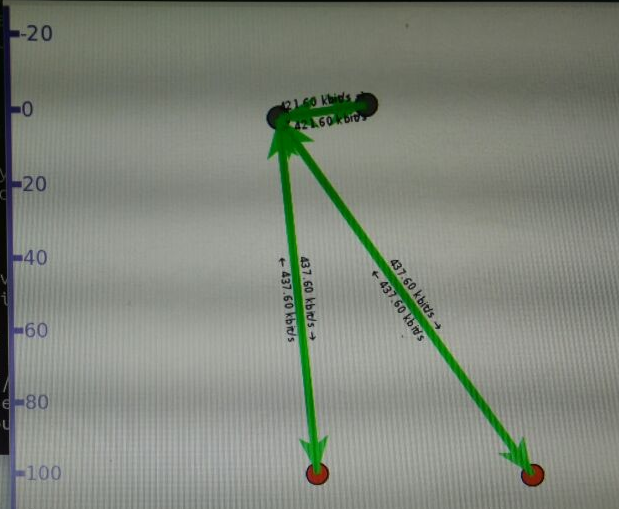
return 0;

}

Output







**Part –B**

1. **Write a program for error detecting code using CRC-CCITT (16- bits).**

import java.io.\*;  
import java.util.Scanner;  
  
class crcscanner  
{  
    public static void main(String a[]) throws IOException  
    {  
          
          Scanner sc=new Scanner(System.in);  
          int[] message;  
          int[] gen;  
          int[] app\_message;  
          int[] rem;  
          int[] trans\_message;  
          int message\_bits,gen\_bits, total\_bits;  
          
         System.out.println("\n Enter number of bits in message : ");  
         message\_bits=sc.nextInt();  
   
         message=new int[message\_bits];  
         System.out.println("\n Enter message bits : ");  
         for(int i=0; i<message\_bits; i++)  
         message[i]=sc.nextInt();  
        System.out.println("\n Enter number of bits in gen : ");  
        gen\_bits=sc.nextInt();  
          
        gen=new int[gen\_bits];  
        System.out.println("\n Enter gen bits : ");  
        for(int i=0; i<gen\_bits; i++)  
       {  
         gen[i]=sc.nextInt();  
       }  
  
  
      total\_bits=message\_bits+gen\_bits-1;   
     
      app\_message=new int[total\_bits];  
      rem=new int[total\_bits];  
      trans\_message=new int[total\_bits];  
          
      for(int i=0;i<message.length;i++)  
      {  
     app\_message[i]=message[i];  
      }  
             
      System.out.print("\n Message bits are : ");  
      for(int i=0; i< message\_bits; i++)  
      {  
  System.out.print(message[i]);          
      }   
      System.out.print("\n Generators bits are : ");  
      for(int i=0; i< gen\_bits; i++)  
      {  
    System.out.print(gen[i]);          
      }  
                  
      System.out.print("\n Appended message is : ");  
      for(int i=0; i< app\_message.length; i++)  
      {  
 System.out.print(app\_message[i]);          
      }  
          
          
      for(int j=0; j<app\_message.length; j++)  
     {  
              rem[j] = app\_message[j];  
     }  
      
     rem=computecrc(app\_message, gen, rem);  
          
     for(int i=0;i<app\_message.length;i++)          
     {  
           trans\_message[i]=(app\_message[i]^rem[i]);  
     }  
          
     System.out.println("\n Transmitted message from the transmitter is : ");      
     for(int i=0;i<trans\_message.length;i++)  
     {   
    System.out.print(trans\_message[i]);  
     }  
              
     System.out.println("\n Enter received message of "+total\_bits+" bits at receiver end : ");  
     for(int i=0; i<trans\_message.length; i++)  
     {  
      trans\_message[i]=sc.nextInt();;  
     }  
     System.out.println("\n Received message is :");    
     for(int i=0; i< trans\_message.length; i++)  
     {  
     System.out.print(trans\_message[i]);          
     }  
             
     for(int j=0; j<trans\_message.length; j++)  
    {  
              rem[j] = trans\_message[j];  
    }  
    rem=computecrc(trans\_message, gen, rem);  
     for(int i=0; i< rem.length; i++)  
    {  
          if(rem[i]!=0)   
   
                
          {  
                System.out.println("\n There is Error in the received me            ");  
                break;  
          }  
          if(i==rem.length-1)  
            
     System.out.println("\n There is No Error in the received m ");  
     }  
  }  
  
   static int[] computecrc(int app\_message[],int gen[], int rem[])  
 {  
      int current=0;  
      while(true)  
     {  
            for(int i=0;i<gen.length;i++)  
           {  
      rem[current+i]=(rem[current+i]^gen[i]);  
           }  
            while(rem[current]==0 && current!=rem.length-1)  
          {  
      current++;  
 }  
            if((rem.length-current)<gen.length)  
      {  
           break;  
 }  
     }  
     return rem;  
 }  
}

8**. Write a program to find the shortest path between vertices using bellman-ford algorithm.**

import java.util.Scanner;

public class BellmanFord

{

private int D[]; private int num\_ver;

public static final int MAX\_VALUE = 999;

public BellmanFord(int num\_ver)

{

this.num\_ver = num\_ver; D = new int[num\_ver + 1];

}

public void BellmanFordEvaluation(int source, int A[][])

{

for (int node = 1; node <= num\_ver; node++)

{

D[node] = MAX\_VALUE;

}

D[source] = 0;

for (int node = 1; node <= num\_ver - 1; node++)

{

for (int sn = 1; sn <= num\_ver; sn++)

{

for (int dn = 1; dn <= num\_ver; dn++)

{

if (A[sn][dn] != MAX\_VALUE)

{

if (D[dn] > D[sn]+ A[sn][dn])

D[dn] = D[sn] + A[sn][dn];

}

}

}

}

for (int sn = 1; sn <= num\_ver; sn++)

{

for (int dn = 1; dn <= num\_ver; dn++)

{

if (A[sn][dn] != MAX\_VALUE)

{

if (D[dn] > D[sn]+ A[sn][dn])

System.out.println("The Graph contains negative egde cycle");

}

}

}

for (int vertex = 1; vertex <= num\_ver; vertex++)

{

System.out.println("distance of source " + source + " to "+ vertex + " is " + D[vertex]);

}

}

public static void main(String[ ] args)

{

int num\_ver = 0; int source;

Scanner scanner = new Scanner(System.in); System.out.println("Enter the number of vertices"); num\_ver = scanner.nextInt();

int A[][] = new int[num\_ver + 1][num\_ver + 1]; System.out.println("Enter the adjacency matrix"); for (int sn = 1; sn <= num\_ver; sn++)

{

for (int dn = 1; dn <= num\_ver; dn++)

{

A[sn][dn] = scanner.nextInt(); if (sn == dn)

{

A[sn][dn] = 0; continue;

}

if (A[sn][dn] == 0)

{

A[sn][dn] = MAX\_VALUE;

}

}

}

System.out.println("Enter the source vertex"); source = scanner.nextInt();

BellmanFord b = new BellmanFord (num\_ver); b.BellmanFordEvaluation(source, A); scanner.close();

}

}

**Input graph:**

****

Output:



**9.Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.**

**Server Program**

import java.io.BufferedInputStream;

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

import java.io.OutputStream;

import java.net.ServerSocket;

import java.net.Socket;

public class SimpleFileServer {

public final static int SOCKET\_PORT = 13267; // you may change this

public final static String FILE\_TO\_SEND = "e:/source1.txt"; // you may change this

public static void main (String [] args ) throws IOException {

FileInputStream fis = null;

BufferedInputStream bis = null;

OutputStream os = null;

ServerSocket servsock = null;

Socket sock = null;

try {

servsock = new ServerSocket(SOCKET\_PORT);

while (true) {

System.out.println("Waiting...");

try {

sock = servsock.accept();

System.out.println("Accepted connection : " + sock);

// send file

File myFile = new File (FILE\_TO\_SEND);

byte [] mybytearray = new byte [(int)myFile.length()];

fis = new FileInputStream(myFile);

bis = new BufferedInputStream(fis);

bis.read(mybytearray,0,mybytearray.length);

os = sock.getOutputStream();

System.out.println("Sending " + FILE\_TO\_SEND + "(" + mybytearray.length + " bytes)");

os.write(mybytearray,0,mybytearray.length);

os.flush();

System.out.println("Done.");

}

finally {

if (bis != null) bis.close();

if (os != null) os.close();

if (sock!=null) sock.close();

}

}

}

finally {

if (servsock != null) servsock.close();

}

}

}

**Client Program**

import java.io.BufferedOutputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.net.Socket;

public class SimpleFileClient {

public final static int SOCKET\_PORT = 13267; // you may change this

public final static String SERVER = "127.0.0.1"; // localhost

public final static String

FILE\_TO\_RECEIVED = "e:/source-downloaded.txt"; // you may change this, I give a

// different name because i don't want to

// overwrite the one used by server...

public final static int FILE\_SIZE = 6022386; // file size temporary hard coded

// should bigger than the file to be downloaded

public static void main (String [] args ) throws IOException {

int bytesRead;

int current = 0;

FileOutputStream fos = null;

BufferedOutputStream bos = null;

Socket sock = null;

try {

sock = new Socket(SERVER, SOCKET\_PORT);

System.out.println("Connecting...");

// receive file

byte [] mybytearray = new byte [FILE\_SIZE];

InputStream is = sock.getInputStream();

fos = new FileOutputStream(FILE\_TO\_RECEIVED);

bos = new BufferedOutputStream(fos);

bytesRead = is.read(mybytearray,0,mybytearray.length);

current = bytesRead;

do {

bytesRead =

is.read(mybytearray, current, (mybytearray.length-current));

if(bytesRead >= 0) current += bytesRead;

} while(bytesRead > -1);

bos.write(mybytearray, 0 , current);

bos.flush();

System.out.println("File " + FILE\_TO\_RECEIVED

+ " downloaded (" + current + " bytes read)");

}

finally {

if (fos != null) fos.close();

if (bos != null) bos.close();

if (sock != null) sock.close();

}

}

}

**10. Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.**

**UDP Client**

import java.io.\*;

import java.net.\*;

public class UDPC

{

public static void main(String[] args)

{

DatagramSocket skt;

try {

skt=new DatagramSocket(); String msg= "text message "; byte[] b = msg.getBytes();

InetAddress host=InetAddress.getByName("127.0.0.1"); int serverSocket=6788;

DatagramPacket request =new DatagramPacket (b,b.length,host,serverSocket); skt.send(request);

byte[] buffer =new byte[1000];

DatagramPacket reply= new DatagramPacket(buffer,buffer.length); skt.receive(reply);

System.out.println("client received:" +new String(reply.getData())); skt.close();

}

catch(Exception ex)

{

}

}

}

**UDP Server**

import java.io.\*; import java.net.\*;

public class UDPS

{

public static void main(String[] args)

{

DatagramSocket skt=null;

try

{

skt=new DatagramSocket(6788); byte[] buffer = new byte[1000];

while(true)

{

DatagramPacket request = new DatagramPacket(buffer,buffer.length);

skt.receive(request);

String[] message = (new String(request.getData())).split("");

byte[] sendMsg= (message[1]+ " server processed").getBytes();

DatagramPacket reply = new DatagramPacket(sendMsg,sendMsg.length,request.getAddress(),request.getPort());

skt.send(reply);

}

}

catch(Exception ex)

{

}

}

}

1. **Write a program for simple RSA algorithm to encrypt and decrypt the data.**

Implementation of RSA Algorithm(Encryption and Decryption) in Java

import java.math.BigInteger;

import java.util.Random;

import java.io.\*;

public class RSA {

private BigInteger p;

private BigInteger q;

private BigInteger N;

private BigInteger phi;

private BigInteger e;

private BigInteger d;

private int bitlength = 1024;

private int blocksize = 256;

//blocksize in byte

private Random r;

public RSA() {

r = new Random();

p = BigInteger.probablePrime(bitlength, r);

q = BigInteger.probablePrime(bitlength, r);

N = p.multiply(q);

phi = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));

e = BigInteger.probablePrime(bitlength/2, r);

while (phi.gcd(e).compareTo(BigInteger.ONE) > 0 && e.compareTo(phi) < 0 ) {

e.add(BigInteger.ONE);

}

d = e.modInverse(phi);

}

public RSA(BigInteger e, BigInteger d, BigInteger N) {

this.e = e;

this.d = d;

this.N = N;

}

public static void main (String[] args) throws IOException {

RSA rsa = new RSA();

DataInputStream in=new DataInputStream(System.in);

String teststring ;

System.out.println("Enter the plain text:");

teststring=in.readLine();

System.out.println("Encrypting String: " + teststring);

System.out.println("String in Bytes: " + bytesToString(teststring.getBytes()));

// encrypt

byte[] encrypted = rsa.encrypt(teststring.getBytes());

System.out.println("Encrypted String in Bytes: " + bytesToString(encrypted));

// decrypt

byte[] decrypted = rsa.decrypt(encrypted);

System.out.println("Decrypted String in Bytes: " + bytesToString(decrypted));

System.out.println("Decrypted String: " + new String(decrypted));

}

private static String bytesToString(byte[] encrypted) {

String test = "";

for (byte b : encrypted) {

test += Byte.toString(b);

}

return test;

}

//Encrypt message

public byte[] encrypt(byte[] message) {

return (new BigInteger(message)).modPow(e, N).toByteArray();

}

// Decrypt message

public byte[] decrypt(byte[] message) {

return (new BigInteger(message)).modPow(d, N).toByteArray();

}

}

1. **Write a program for congestion control using leaky bucket algorithm.**

filename:Licky.java

import java.io.\*;

import java.util.\*;

class Queue

{

int q[],f=0,r=0,size;

void insert(int n)

{

Scanner in = new Scanner(System.in);

q=new int[10];

for(int i=0;i<n;i++)

{

System.out.print("\nEnter " + i + " element: ");

int ele=in.nextInt();

if(r+1>10)

{

System.out.println("\nQueue is full \nLost Packet: "+ele);

break;

}

else

{

r++;

q[i]=ele;

}

}

}

void delete()

{

Scanner in = new Scanner(System.in);

Thread t=new Thread();

if(r==0)

System.out.print("\nQueue empty ");

else

{

for(int i=f;i<r;i++)

{

try

{

t.sleep(1000);

}

catch(Exception e){}

System.out.print("\nLeaked Packet: "+q[i]);

f++;

}

}

System.out.println();

}

}

class Licky extends Thread

{

public static void main(String ar[])throws Exception

{

Queue q=new Queue();

Scanner src=new Scanner(System.in);

System.out.println("\nEnter the packets to be sent:");

int size=src.nextInt();

q.insert(size);

q.delete();

}

}

/\*

OUTPUT

bash-3.00$ javac Licky.java

bash-3.00$ java Licky

Enter the packets to be sent:

11

Enter 0 element: 1

Enter 1 element: 0

Enter 2 element: 2

Enter 3 element: 3

Enter 4 element: 4

Enter 5 element: 5

Enter 6 element: 6

Enter 7 element: 7

Enter 8 element: 8

Enter 9 element: 9

Enter 10 element: 10

Queue is full

Lost Packet: 10

Leaked Packet: 1

Leaked Packet: 0

Leaked Packet: 2

Leaked Packet: 3

Leaked Packet: 4

Leaked Packet: 5

Leaked Packet: 6

Leaked Packet: 7

Leaked Packet: 8

Leaked Packet: 9

**Part –B**

1. **Write a program for error detecting code using CRC-CCITT (16- bits).**

import java.io.\*;  
import java.util.Scanner;  
  
class crcscanner  
{  
    public static void main(String a[]) throws IOException  
    {  
          
          Scanner sc=new Scanner(System.in);  
          int[] message;  
          int[] gen;  
          int[] app\_message;  
          int[] rem;  
          int[] trans\_message;  
          int message\_bits,gen\_bits, total\_bits;  
          
         System.out.println("\n Enter number of bits in message : ");  
         message\_bits=sc.nextInt();  
   
         message=new int[message\_bits];  
         System.out.println("\n Enter message bits : ");  
         for(int i=0; i<message\_bits; i++)  
         message[i]=sc.nextInt();  
        System.out.println("\n Enter number of bits in gen : ");  
        gen\_bits=sc.nextInt();  
          
        gen=new int[gen\_bits];  
        System.out.println("\n Enter gen bits : ");  
        for(int i=0; i<gen\_bits; i++)  
       {  
         gen[i]=sc.nextInt();  
       }  
  
  
      total\_bits=message\_bits+gen\_bits-1;   
     
      app\_message=new int[total\_bits];  
      rem=new int[total\_bits];  
      trans\_message=new int[total\_bits];  
          
      for(int i=0;i<message.length;i++)  
      {  
     app\_message[i]=message[i];  
      }  
             
      System.out.print("\n Message bits are : ");  
      for(int i=0; i< message\_bits; i++)  
      {  
  System.out.print(message[i]);          
      }   
      System.out.print("\n Generators bits are : ");  
      for(int i=0; i< gen\_bits; i++)  
      {  
    System.out.print(gen[i]);          
      }  
                  
      System.out.print("\n Appended message is : ");  
      for(int i=0; i< app\_message.length; i++)  
      {  
 System.out.print(app\_message[i]);          
      }  
          
          
      for(int j=0; j<app\_message.length; j++)  
     {  
              rem[j] = app\_message[j];  
     }  
      
     rem=computecrc(app\_message, gen, rem);  
          
     for(int i=0;i<app\_message.length;i++)          
     {  
           trans\_message[i]=(app\_message[i]^rem[i]);  
     }  
          
     System.out.println("\n Transmitted message from the transmitter is : ");      
     for(int i=0;i<trans\_message.length;i++)  
     {   
    System.out.print(trans\_message[i]);  
     }  
              
     System.out.println("\n Enter received message of "+total\_bits+" bits at receiver end : ");  
     for(int i=0; i<trans\_message.length; i++)  
     {  
      trans\_message[i]=sc.nextInt();;  
     }  
     System.out.println("\n Received message is :");    
     for(int i=0; i< trans\_message.length; i++)  
     {  
     System.out.print(trans\_message[i]);          
     }  
             
     for(int j=0; j<trans\_message.length; j++)  
    {  
              rem[j] = trans\_message[j];  
    }  
    rem=computecrc(trans\_message, gen, rem);  
     for(int i=0; i< rem.length; i++)  
    {  
          if(rem[i]!=0)   
   
                
          {  
                System.out.println("\n There is Error in the received me            ");  
                break;  
          }  
          if(i==rem.length-1)  
            
     System.out.println("\n There is No Error in the received m ");  
     }  
  }  
  
   static int[] computecrc(int app\_message[],int gen[], int rem[])  
 {  
      int current=0;  
      while(true)  
     {  
            for(int i=0;i<gen.length;i++)  
           {  
      rem[current+i]=(rem[current+i]^gen[i]);  
           }  
            while(rem[current]==0 && current!=rem.length-1)  
          {  
      current++;  
 }  
            if((rem.length-current)<gen.length)  
      {  
           break;  
 }  
     }  
     return rem;  
 }  
}

8**. Write a program to find the shortest path between vertices using bellman-ford algorithm.**

import java.util.Scanner;

public class BellmanFord

{

private int D[]; private int num\_ver;

public static final int MAX\_VALUE = 999;

public BellmanFord(int num\_ver)

{

this.num\_ver = num\_ver; D = new int[num\_ver + 1];

}

public void BellmanFordEvaluation(int source, int A[][])

{

for (int node = 1; node <= num\_ver; node++)

{

D[node] = MAX\_VALUE;

}

D[source] = 0;

for (int node = 1; node <= num\_ver - 1; node++)

{

for (int sn = 1; sn <= num\_ver; sn++)

{

for (int dn = 1; dn <= num\_ver; dn++)

{

if (A[sn][dn] != MAX\_VALUE)

{

if (D[dn] > D[sn]+ A[sn][dn])

D[dn] = D[sn] + A[sn][dn];

}

}

}

}

for (int sn = 1; sn <= num\_ver; sn++)

{

for (int dn = 1; dn <= num\_ver; dn++)

{

if (A[sn][dn] != MAX\_VALUE)

{

if (D[dn] > D[sn]+ A[sn][dn])

System.out.println("The Graph contains negative egde cycle");

}

}

}

for (int vertex = 1; vertex <= num\_ver; vertex++)

{

System.out.println("distance of source " + source + " to "+ vertex + " is " + D[vertex]);

}

}

public static void main(String[ ] args)

{

int num\_ver = 0; int source;

Scanner scanner = new Scanner(System.in); System.out.println("Enter the number of vertices"); num\_ver = scanner.nextInt();

int A[][] = new int[num\_ver + 1][num\_ver + 1]; System.out.println("Enter the adjacency matrix"); for (int sn = 1; sn <= num\_ver; sn++)

{

for (int dn = 1; dn <= num\_ver; dn++)

{

A[sn][dn] = scanner.nextInt(); if (sn == dn)

{

A[sn][dn] = 0; continue;

}

if (A[sn][dn] == 0)

{

A[sn][dn] = MAX\_VALUE;

}

}

}

System.out.println("Enter the source vertex"); source = scanner.nextInt();

BellmanFord b = new BellmanFord (num\_ver); b.BellmanFordEvaluation(source, A); scanner.close();

}

}

**Input graph:**

****

Output:



**9.Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.**

**Server Program**

import java.io.BufferedInputStream;

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

import java.io.OutputStream;

import java.net.ServerSocket;

import java.net.Socket;

public class SimpleFileServer {

public final static int SOCKET\_PORT = 13267; // you may change this

public final static String FILE\_TO\_SEND = "e:/source1.txt"; // you may change this

public static void main (String [] args ) throws IOException {

FileInputStream fis = null;

BufferedInputStream bis = null;

OutputStream os = null;

ServerSocket servsock = null;

Socket sock = null;

try {

servsock = new ServerSocket(SOCKET\_PORT);

while (true) {

System.out.println("Waiting...");

try {

sock = servsock.accept();

System.out.println("Accepted connection : " + sock);

// send file

File myFile = new File (FILE\_TO\_SEND);

byte [] mybytearray = new byte [(int)myFile.length()];

fis = new FileInputStream(myFile);

bis = new BufferedInputStream(fis);

bis.read(mybytearray,0,mybytearray.length);

os = sock.getOutputStream();

System.out.println("Sending " + FILE\_TO\_SEND + "(" + mybytearray.length + " bytes)");

os.write(mybytearray,0,mybytearray.length);

os.flush();

System.out.println("Done.");

}

finally {

if (bis != null) bis.close();

if (os != null) os.close();

if (sock!=null) sock.close();

}

}

}

finally {

if (servsock != null) servsock.close();

}

}

}

**Client Program**

import java.io.BufferedOutputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.net.Socket;

public class SimpleFileClient {

public final static int SOCKET\_PORT = 13267; // you may change this

public final static String SERVER = "127.0.0.1"; // localhost

public final static String

FILE\_TO\_RECEIVED = "e:/source-downloaded.txt"; // you may change this, I give a

// different name because i don't want to

// overwrite the one used by server...

public final static int FILE\_SIZE = 6022386; // file size temporary hard coded

// should bigger than the file to be downloaded

public static void main (String [] args ) throws IOException {

int bytesRead;

int current = 0;

FileOutputStream fos = null;

BufferedOutputStream bos = null;

Socket sock = null;

try {

sock = new Socket(SERVER, SOCKET\_PORT);

System.out.println("Connecting...");

// receive file

byte [] mybytearray = new byte [FILE\_SIZE];

InputStream is = sock.getInputStream();

fos = new FileOutputStream(FILE\_TO\_RECEIVED);

bos = new BufferedOutputStream(fos);

bytesRead = is.read(mybytearray,0,mybytearray.length);

current = bytesRead;

do {

bytesRead =

is.read(mybytearray, current, (mybytearray.length-current));

if(bytesRead >= 0) current += bytesRead;

} while(bytesRead > -1);

bos.write(mybytearray, 0 , current);

bos.flush();

System.out.println("File " + FILE\_TO\_RECEIVED

+ " downloaded (" + current + " bytes read)");

}

finally {

if (fos != null) fos.close();

if (bos != null) bos.close();

if (sock != null) sock.close();

}

}

}

**10. Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.**

**UDP Client**

import java.io.\*;

import java.net.\*;

public class UDPC

{

public static void main(String[] args)

{

DatagramSocket skt;

try {

skt=new DatagramSocket(); String msg= "text message "; byte[] b = msg.getBytes();

InetAddress host=InetAddress.getByName("127.0.0.1"); int serverSocket=6788;

DatagramPacket request =new DatagramPacket (b,b.length,host,serverSocket); skt.send(request);

byte[] buffer =new byte[1000];

DatagramPacket reply= new DatagramPacket(buffer,buffer.length); skt.receive(reply);

System.out.println("client received:" +new String(reply.getData())); skt.close();

}

catch(Exception ex)

{

}

}

}

**UDP Server**

import java.io.\*; import java.net.\*;

public class UDPS

{

public static void main(String[] args)

{

DatagramSocket skt=null;

try

{

skt=new DatagramSocket(6788); byte[] buffer = new byte[1000];

while(true)

{

DatagramPacket request = new DatagramPacket(buffer,buffer.length);

skt.receive(request);

String[] message = (new String(request.getData())).split("");

byte[] sendMsg= (message[1]+ " server processed").getBytes();

DatagramPacket reply = new DatagramPacket(sendMsg,sendMsg.length,request.getAddress(),request.getPort());

skt.send(reply);

}

}

catch(Exception ex)

{

}

}

}

1. **Write a program for simple RSA algorithm to encrypt and decrypt the data.**

Implementation of RSA Algorithm(Encryption and Decryption) in Java

import java.math.BigInteger;

import java.util.Random;

import java.io.\*;

public class RSA {

private BigInteger p;

private BigInteger q;

private BigInteger N;

private BigInteger phi;

private BigInteger e;

private BigInteger d;

private int bitlength = 1024;

private int blocksize = 256;

//blocksize in byte

private Random r;

public RSA() {

r = new Random();

p = BigInteger.probablePrime(bitlength, r);

q = BigInteger.probablePrime(bitlength, r);

N = p.multiply(q);

phi = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));

e = BigInteger.probablePrime(bitlength/2, r);

while (phi.gcd(e).compareTo(BigInteger.ONE) > 0 && e.compareTo(phi) < 0 ) {

e.add(BigInteger.ONE);

}

d = e.modInverse(phi);

}

public RSA(BigInteger e, BigInteger d, BigInteger N) {

this.e = e;

this.d = d;

this.N = N;

}

public static void main (String[] args) throws IOException {

RSA rsa = new RSA();

DataInputStream in=new DataInputStream(System.in);

String teststring ;

System.out.println("Enter the plain text:");

teststring=in.readLine();

System.out.println("Encrypting String: " + teststring);

System.out.println("String in Bytes: " + bytesToString(teststring.getBytes()));

// encrypt

byte[] encrypted = rsa.encrypt(teststring.getBytes());

System.out.println("Encrypted String in Bytes: " + bytesToString(encrypted));

// decrypt

byte[] decrypted = rsa.decrypt(encrypted);

System.out.println("Decrypted String in Bytes: " + bytesToString(decrypted));

System.out.println("Decrypted String: " + new String(decrypted));

}

private static String bytesToString(byte[] encrypted) {

String test = "";

for (byte b : encrypted) {

test += Byte.toString(b);

}

return test;

}

//Encrypt message

public byte[] encrypt(byte[] message) {

return (new BigInteger(message)).modPow(e, N).toByteArray();

}

// Decrypt message

public byte[] decrypt(byte[] message) {

return (new BigInteger(message)).modPow(d, N).toByteArray();

}

}

1. **Write a program for congestion control using leaky bucket algorithm.**

filename:Licky.java

import java.io.\*;

import java.util.\*;

class Queue

{

int q[],f=0,r=0,size;

void insert(int n)

{

Scanner in = new Scanner(System.in);

q=new int[10];

for(int i=0;i<n;i++)

{

System.out.print("\nEnter " + i + " element: ");

int ele=in.nextInt();

if(r+1>10)

{

System.out.println("\nQueue is full \nLost Packet: "+ele);

break;

}

else

{

r++;

q[i]=ele;

}

}

}

void delete()

{

Scanner in = new Scanner(System.in);

Thread t=new Thread();

if(r==0)

System.out.print("\nQueue empty ");

else

{

for(int i=f;i<r;i++)

{

try

{

t.sleep(1000);

}

catch(Exception e){}

System.out.print("\nLeaked Packet: "+q[i]);

f++;

}

}

System.out.println();

}

}

class Licky extends Thread

{

public static void main(String ar[])throws Exception

{

Queue q=new Queue();

Scanner src=new Scanner(System.in);

System.out.println("\nEnter the packets to be sent:");

int size=src.nextInt();

q.insert(size);

q.delete();

}

}

/\*

OUTPUT

bash-3.00$ javac Licky.java

bash-3.00$ java Licky

Enter the packets to be sent:

11

Enter 0 element: 1

Enter 1 element: 0

Enter 2 element: 2

Enter 3 element: 3

Enter 4 element: 4

Enter 5 element: 5

Enter 6 element: 6

Enter 7 element: 7

Enter 8 element: 8

Enter 9 element: 9

Enter 10 element: 10

Queue is full

Lost Packet: 10

Leaked Packet: 1

Leaked Packet: 0

Leaked Packet: 2

Leaked Packet: 3

Leaked Packet: 4

Leaked Packet: 5

Leaked Packet: 6

Leaked Packet: 7

Leaked Packet: 8

Leaked Packet: 9