* **IGNORE THINGS WRITTEN IN RED**
* **THE PLACE WHERE IT IS HIGHLIGHTED IN YELLOW IS THE CHANGES TO BE DONE IN THE PROGRAMS (I MEAN TO BE ADDED IN THE PROGRAM).**

**PROGRAM 1:**

**Simulate peer-to-peer communication between a client and a server using Point-to-Point protocol. Apply NetAnim software to demonstrate the scenario graphically. Analyze packet parameters by creating trace file usingAscii trace metrics.**

/\* -\*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -\*- \*/

/\*

\* 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/core-module.h"

#include "ns3/network-module.h"

#include "ns3/internet-module.h"

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

#include "ns3/applications-module.h"

#include "ns3/netanim-module.h"

// Default Network Topology

//

// 10.1.1.0

// n0 -------------- n1

// point-to-point

//

using namespace ns3;

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

int

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

{

CommandLine cmd (\_\_FILE\_\_);

cmd.Parse (argc, argv);

Time::SetResolution (Time::NS);

LogComponentEnable ("UdpEchoClientApplication", LOG\_LEVEL\_INFO);

LogComponentEnable ("UdpEchoServerApplication", LOG\_LEVEL\_INFO);

std::string animFile="first.xml";

NodeContainer nodes;

nodes.Create (2);

PointToPointHelper pointToPoint;

pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));

pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));

NetDeviceContainer devices;

devices = pointToPoint.Install (nodes);

InternetStackHelper stack;

stack.Install (nodes);

Ipv4AddressHelper address;

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

Ipv4InterfaceContainer interfaces = address.Assign (devices);

UdpEchoServerHelper echoServer (9);

ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));

serverApps.Start (Seconds (1.0));

serverApps.Stop (Seconds (10.0));

AnimationInterface anim(animFile);

anim.SetConstantPosition(nodes.Get(0),1.0,2.0);

anim.SetConstantPosition(nodes.Get(1),45.0,60.0);

AsciiTraceHelper ascii;

pointToPoint.EnableAsciiAll(ascii.CreateFileStream("first.tr"));

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;

}

**PROGRAM 2:**

**Simulate to implement a bus topology using Point-to-Point protocol betweena client and a LAN with 4 nodes. The LAN use CSMA during packet transmission. Apply NetAnim software to demonstrate the scenario graphically. Analyze packet parameters by creating trace file using Ascii trace metrics.**

/\* -\*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -\*- \*/

/\*

\* 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/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"

// 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);

}

std::string animFile="second.xml";

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 ("second");

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

AnimationInterface anim(animFile);

anim.SetConstantPosition(p2pNodes.Get(0),1.0,2.0);

anim.SetConstantPosition(p2pNodes.Get(1),45.0,60.0);

anim.SetConstantPosition(csmaNodes.Get(1),55.0,60.0);

anim.SetConstantPosition(csmaNodes.Get(2),65.0,60.0);

anim.SetConstantPosition(csmaNodes.Get(3),75.0,60.0);

AsciiTraceHelper ascii;

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

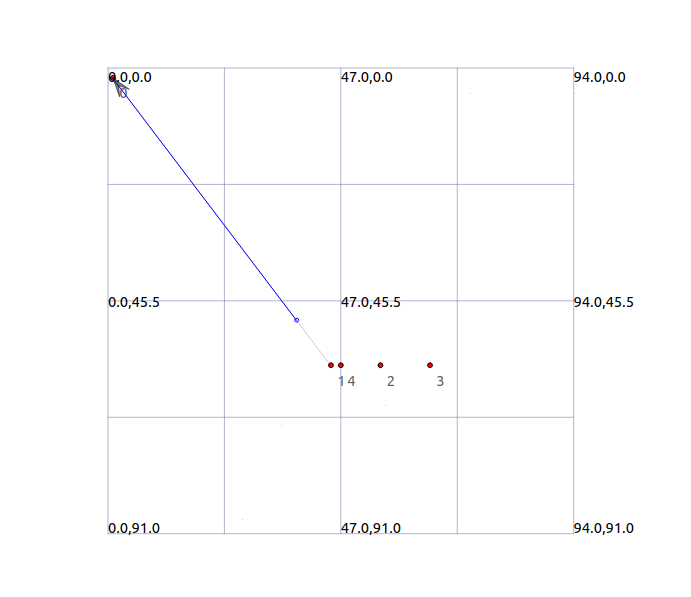
csma.EnableAsciiAll(ascii.CreateFileStream("csma.tr"));

Simulator::Run ();

Simulator::Destroy ();

return 0;

}



**PROGRAM 3:**

**Simulate peer-to-peer communication between a client and a server using CSMA protocol. Apply NetAnim software to demonstrate the scenario graphically. Analyze packet parameters by creating trace file using Ascii trace metrics.**

/\* -\*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -\*- \*/

/\*

\* 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/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"

// 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 = 4;

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);

}

std::string animFile="third.xml";

nCsma = nCsma == 0 ? 1 : nCsma;

//NodeContainer p2pNodes;

//p2pNodes.Create (2);

NodeContainer csmaNodes;

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 (3));

serverApps.Start (Seconds (1.0));

serverApps.Stop (Seconds (10.0));

UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (3), 9);

echoClient.SetAttribute ("MaxPackets", UintegerValue (1));

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

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

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

clientApps.Start (Seconds (2.0));

clientApps.Stop (Seconds (10.0));

//Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

//pointToPoint.EnablePcapAll ("second");

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

AnimationInterface anim(animFile);

//anim.SetConstantPosition(p2pNodes.Get(0),1.0,2.0);

anim.SetConstantPosition(csmaNodes.Get(0),45.0,60.0);

anim.SetConstantPosition(csmaNodes.Get(1),55.0,60.0);

anim.SetConstantPosition(csmaNodes.Get(2),65.0,60.0);

anim.SetConstantPosition(csmaNodes.Get(3),75.0,60.0);

AsciiTraceHelper ascii;

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

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

Simulator::Run ();

Simulator::Destroy ();

return 0;

}

