**#include "ns3/core-module.h"**

**#include "ns3/global-route-manager.h"**

**#include "ns3/network-module.h"**

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

**#include "ns3/bridge-module.h"**

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

**#include "ns3/applications-module.h"**

**#include "ns3/netanim-module.h"**

using namespace ns3;

**NS\_LOG\_COMPONENT\_DEFINE("Firstscript");**

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

{

CommandLine cmd;

cmd.Parse (argc, argv);

/\* Configuration. \*/

**Config::SetDefault ("ns3::OnOffApplication::PacketSize", UintegerValue (1024));**

**Config::SetDefault ("ns3::OnOffApplication::DataRate", StringValue ("5Mbps"));**

**std::string animFile = "exp1.xml";**

/\* Build nodes. \*/

NodeContainer term\_0;

term\_0.Create (1);

NodeContainer term\_1;

term\_1.Create (1);

NodeContainer term\_2;

term\_2.Create (1);

NodeContainer term\_3;

term\_3.Create (1);

/\* Build link. \*/

PointToPointHelper p2p\_p2p\_0;

p2p\_p2p\_0.SetDeviceAttribute ("DataRate", DataRateValue (**5000000**));

p2p\_p2p\_0.SetChannelAttribute ("Delay", TimeValue (MilliSeconds (**2**)));

PointToPointHelper p2p\_p2p\_1;

p2p\_p2p\_1.SetDeviceAttribute ("DataRate", DataRateValue (**5000000**));

p2p\_p2p\_1.SetChannelAttribute ("Delay", TimeValue (MilliSeconds (**2**)));

PointToPointHelper p2p\_p2p\_2;

p2p\_p2p\_2.SetDeviceAttribute ("DataRate", DataRateValue (**5000000**));

p2p\_p2p\_2.SetChannelAttribute ("Delay", TimeValue (MilliSeconds (**2**)));

/\* Build link net device container. \*/

NodeContainer all\_p2p\_0;

all\_p2p\_0.Add (term\_0);

all\_p2p\_0.Add (term\_2);

NetDeviceContainer ndc\_p2p\_0 = p2p\_p2p\_0.Install (all\_p2p\_0);

NodeContainer all\_p2p\_1;

all\_p2p\_1.Add (term\_1);

all\_p2p\_1.Add (term\_2);

NetDeviceContainer ndc\_p2p\_1 = p2p\_p2p\_1.Install (all\_p2p\_1);

NodeContainer all\_p2p\_2;

all\_p2p\_2.Add (term\_2);

all\_p2p\_2.Add (term\_3);

NetDeviceContainer ndc\_p2p\_2 = p2p\_p2p\_2.Install (all\_p2p\_2);

/\* Install the IP stack. \*/

InternetStackHelper internetStackH;

internetStackH.Install (term\_0);

internetStackH.Install (term\_1);

internetStackH.Install (term\_2);

internetStackH.Install (term\_3);

/\* IP assign. \*/

Ipv4AddressHelper ipv4;

ipv4.SetBase ("10.0.0.0", "255.255.255.0");

Ipv4InterfaceContainer iface\_ndc\_p2p\_0 = ipv4.Assign (ndc\_p2p\_0);

ipv4.SetBase ("10.0.1.0", "255.255.255.0");

Ipv4InterfaceContainer iface\_ndc\_p2p\_1 = ipv4.Assign (ndc\_p2p\_1);

ipv4.SetBase ("10.0.2.0", "255.255.255.0");

Ipv4InterfaceContainer iface\_ndc\_p2p\_2 = ipv4.Assign (ndc\_p2p\_2);

/\* Generate Route. \*/

Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

/\* Generate Application. \*/

uint16\_t port\_udpEcho\_0 = 123;

UdpEchoServerHelper server\_udpEcho\_0 (port\_udpEcho\_0);

ApplicationContainer apps\_udpEcho\_0 = server\_udpEcho\_0.Install (term\_3.Get(0));

apps\_udpEcho\_0.Start (Seconds (0.0));

apps\_udpEcho\_0.Stop (Seconds (10.0));

Time interPacketInterval\_udpEcho\_0 = Seconds (1.0);

UdpEchoClientHelper client\_udpEcho\_0 (iface\_ndc\_p2p\_2.GetAddress(1), 123);

client\_udpEcho\_0.SetAttribute ("MaxPackets", UintegerValue (**20**));

client\_udpEcho\_0.SetAttribute ("Interval", TimeValue (interPacketInterval\_udpEcho\_0));

client\_udpEcho\_0.SetAttribute ("PacketSize", UintegerValue (1024));

apps\_udpEcho\_0 = client\_udpEcho\_0.Install (term\_1.Get (0));

apps\_udpEcho\_0.Start (Seconds (0.1));

apps\_udpEcho\_0.Stop (Seconds (10.0));

uint16\_t port\_tcp\_0 = 224;

Address sinkLocalAddress\_tcp\_0 (InetSocketAddress (Ipv4Address::GetAny (), port\_tcp\_0));

PacketSinkHelper sinkHelper\_tcp\_0 ("ns3::TcpSocketFactory", sinkLocalAddress\_tcp\_0);

ApplicationContainer sinkApp\_tcp\_0 = sinkHelper\_tcp\_0.Install (term\_3);

sinkApp\_tcp\_0.Start (Seconds (0.0));

sinkApp\_tcp\_0.Stop (Seconds (10.0));

OnOffHelper clientHelper\_tcp\_0 ("ns3::TcpSocketFactory", Address ());

**clientHelper\_tcp\_0.SetAttribute ("OnTime", StringValue ("ns3::ConstantRandomVariable[Constant=1]"));**

**clientHelper\_tcp\_0.SetAttribute ("OffTime", StringValue ("ns3::ConstantRandomVariable[Constant=0]"));**

ApplicationContainer clientApps\_tcp\_0;

AddressValue remoteAddress\_tcp\_0 (InetSocketAddress (iface\_ndc\_p2p\_2.GetAddress (1), port\_tcp\_0));

clientHelper\_tcp\_0.SetAttribute ("Remote", remoteAddress\_tcp\_0);

clientApps\_tcp\_0.Add (clientHelper\_tcp\_0.Install (term\_0));

clientApps\_tcp\_0.Start (Seconds (0.0));

clientApps\_tcp\_0.Stop (Seconds (10.0));

/\* Simulation. \*/

**AnimationInterface anim (animFile);**

**Ptr<Node> n = term\_0.Get (0);**

**anim.SetConstantPosition (n, 100, 100);**

**n = term\_1.Get (0);**

**anim.SetConstantPosition (n, 100, 200);**

**n = term\_2.Get (0);**

**anim.SetConstantPosition (n, 200, 150);**

**n = term\_3.Get (0);**

**anim.SetConstantPosition (n, 300, 150);**

/\* Pcap output. \*/

/\* Stop the simulation after x seconds. \*/

uint32\_t stopTime = 11;

Simulator::Stop (Seconds (stopTime));

/\* Start and clean simulation. \*/

Simulator::Run ();

Simulator::Destroy ();

}