1. Point to Point UDP

```
/*
 * 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/netanim-module.h"
#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;
int main(int argc, char *argv[])
{
  Time::SetResolution(Time::NS);
  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));

  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));
  AnimationInterface anim("first.xml");
  Simulator::Run();
  Simulator::Destroy();
  return 0;
}
```

2. LAN Using UDP

```
/* -*- 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
 */

// Network topology
//
//       n0    n1   n2   n3
//       |     |    |    |
//       =================
//              LAN
//
// - UDP flows from n0 to n1 and back
// - DropTail queues
// - Tracing of queues and packet receptions to file "udp-echo.tr"

#include <fstream>
```

```cpp
#include "ns3/core-module.h"
#include "ns3/csma-module.h"
#include "ns3/applications-module.h"
#include "ns3/internet-module.h"
#include "ns3/netanim-module.h"

using namespace ns3;

int main(int argc, char *argv[])
{
  Address serverAddress;
  NodeContainer n;
  n.Create(4);
  InternetStackHelper internet;
  internet.Install(n);
  CsmaHelper csma;
  csma.SetChannelAttribute("DataRate", DataRateValue(DataRate(5000000)));
  csma.SetChannelAttribute("Delay", TimeValue(MilliSeconds(2)));
  csma.SetDeviceAttribute("Mtu", UintegerValue(1400));
  NetDeviceContainer d = csma.Install(n);
  Ipv4AddressHelper ipv4;
  ipv4.SetBase("10.1.1.0", "255.255.255.0");
  Ipv4InterfaceContainer i = ipv4.Assign(d);
  serverAddress = Address(i.GetAddress(1));

  uint16_t port = 9; // well-known echo port number
  UdpEchoServerHelper server(port);
  ApplicationContainer apps = server.Install(n.Get(1));
  apps.Start(Seconds(1.0));
  apps.Stop(Seconds(10.0));

  uint32_t packetSize = 1024;
  uint32_t maxPacketCount = 1;
  Time interPacketInterval = Seconds(1.);
  UdpEchoClientHelper client(serverAddress, port);
  client.SetAttribute("MaxPackets", UintegerValue(maxPacketCount));
  client.SetAttribute("Interval", TimeValue(interPacketInterval));
  client.SetAttribute("PacketSize", UintegerValue(packetSize));
  apps = client.Install(n.Get(0));
  apps.Start(Seconds(2.0));
  apps.Stop(Seconds(10.0));

#if 0
client.SetFill (apps.Get (0), "Hello World");
client.SetFill (apps.Get (0), 0xa5, 1024);
uint8_t fill[] = { 0, 1, 2, 3, 4, 5, 6};
  client.SetFill (apps.Get (0), fill, sizeof(fill), 1024);
#endif
```

```
  AnimationInterface anim("second.xml");
  Simulator::Run();
  Simulator::Destroy();
}
```

3.

```
#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/csma-module.h"
#include "ns3/internet-module.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;
int main(int argc, char *argv[])
{

  uint32_t nCsma = 3;
  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("third.xml");
  Simulator::Run();
  Simulator::Destroy();
  return 0;
}
```

4. Point to point TCP

```
#include <string>
#include <fstream>
#include "ns3/core-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/network-module.h"
#include "ns3/packet-sink.h"
#include "ns3/netanim-module.h"

using namespace ns3;
int main(int argc, char *argv[])
{

  uint32_t maxBytes = 0;
  NodeContainer nodes;
  nodes.Create(2);
  PointToPointHelper pointToPoint;
  pointToPoint.SetDeviceAttribute("DataRate", StringValue("500Kbps"));
  pointToPoint.SetChannelAttribute("Delay", StringValue("5ms"));
  NetDeviceContainer devices;
  devices = pointToPoint.Install(nodes);
  InternetStackHelper internet;
  internet.Install(nodes);
  Ipv4AddressHelper ipv4;
  ipv4.SetBase("10.1.1.0", "255.255.255.0");
  Ipv4InterfaceContainer i = ipv4.Assign(devices);
  uint16_t port = 9; // well-known echo port number
  BulkSendHelper source("ns3::TcpSocketFactory",
                        InetSocketAddress(i.GetAddress(1), port));
  source.SetAttribute("MaxBytes", UintegerValue(maxBytes));
  ApplicationContainer sourceApps = source.Install(nodes.Get(0));
  sourceApps.Start(Seconds(0.0));
  sourceApps.Stop(Seconds(10.0));
  PacketSinkHelper sink("ns3::TcpSocketFactory",
                        InetSocketAddress(Ipv4Address::GetAny(), port));
  ApplicationContainer sinkApps = sink.Install(nodes.Get(1));
  sinkApps.Start(Seconds(0.0));
  sinkApps.Stop(Seconds(10.0));
  Simulator::Stop(Seconds(10.0));
  AnimationInterface anim("fourth.xml");
  anim.EnablePacketMetadata(true);
  Simulator::Run();
  Simulator::Destroy();
}
```