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

/*

* Copyright (c) 2011 UPB

* Copyright (c) 2017 NITK Surathkal *

* 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 layout: *

* R0 is a DHCP server. The DHCP server announced R1 as the default router. * Nodes N1 will send UDP Echo packets to node A. *

*

* ┌-----------------------------------------------┐

* | DHCP Clients | * | 172.30.0.14 | * | DHCP static | * | ┌─────────┐ ┌─────────┐ ┌─────────┐ |

* | │ N0 │ │ N1 │ │ N2 │ | ┌─────────┐

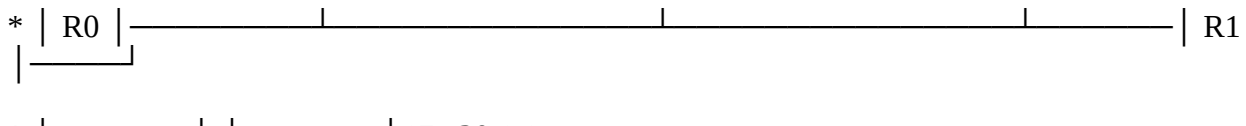
* | └─────────┘ └─────────┘ └─────────┘ | ┌──────│ A │

* | │ │ │ | │ └─────────┘

* └-------│--------------│---------------│----------┘ │ 172.30.1.2

* DHCP Server │ │ │ │

* ┌─────────┐ │ │ │ ┌─────────┐ │




* 172.30.0.12 172.30.0.17

*

* Things to notice: * 1) The routes in A are manually set to have R1 as the default router, * just because using a dynamic outing in this example is an overkill. * 2) R1's address is set statically though the DHCP server helper interface. * This is useful to prevent address conflicts with the dynamic pool. * Not necessary if the DHCP pool is not conflicting with static addresses. * 3) N2 has a dynamically-assigned, static address (i.e., a fixed address assigned via DHCP). *

*/

```
#include "ns3/core-module.h"
#include "ns3/internet-apps-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/netanim-module.h"
#include "ns3/mobility-module.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("DhcpExample");

int

main (int argc, char *argv[])

{

CommandLine cmd (__FILE__);

bool verbose = false;

bool tracing = false;

cmd.AddValue ("verbose", "turn on the logs", verbose);

cmd.AddValue ("tracing", "turn on the tracing", tracing);

cmd.Parse (argc, argv);

// GlobalValue::Bind ("ChecksumEnabled", BooleanValue (true));

if (verbose)
```

```
{
LogComponentEnable ("DhcpServer", LOG_LEVEL_ALL);
LogComponentEnable ("DhcpClient", LOG_LEVEL_ALL);
LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
}
Time stopTime = Seconds (20);
NS_LOG_INFO ("Create nodes.");
NodeContainer nodes;
NodeContainer router;
nodes.Create (3);
router.Create (2);
NodeContainer net (nodes, router);
NS_LOG_INFO ("Create channels.");
CsmaHelper csma;
csma.SetChannelAttribute ("DataRate", StringValue ("5Mbps"));
csma.SetChannelAttribute ("Delay", StringValue ("2ms"));
csma.SetDeviceAttribute ("Mtu", UintegerValue (1500));
NetDeviceContainer devNet = csma.Install (net);
NodeContainer p2pNodes;
p2pNodes.Add (net.Get (4));
p2pNodes.Create (1);
PointToPointHelper pointToPoint;
pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
NetDeviceContainer p2pDevices;
```

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

InternetStackHelper tcpip;

tcpip.Install (nodes);

tcpip.Install (router);

tcpip.Install (p2pNodes.Get (1));

Ipv4AddressHelper address;

address.SetBase ("172.30.1.0", "255.255.255.0");

Ipv4InterfaceContainer p2pInterfaces;

p2pInterfaces = address.Assign (p2pDevices);

// manually add a routing entry because we don't want to add a dynamic routing

Ipv4StaticRoutingHelper ipv4RoutingHelper;

Ptr<Ipv4> ipv4Ptr = p2pNodes.Get (1)->GetObject<Ipv4> ();

Ptr<Ipv4StaticRouting> staticRoutingA = ipv4RoutingHelper.GetStaticRouting (ipv4Ptr);

staticRoutingA->AddNetworkRouteTo (Ipv4Address ("172.30.0.0"), Ipv4Mask ("/24"),
Ipv4Address ("172.30.1.1"), 1);

NS_LOG_INFO ("Setup the IP addresses and create DHCP applications.");

DhcpHelper dhcpHelper;

// The router must have a fixed IP.

Ipv4InterfaceContainer fixedNodes = dhcpHelper.InstallFixedAddress (devNet.Get (4),
Ipv4Address ("172.30.0.17"), Ipv4Mask ("/24"));

// Not really necessary, IP forwarding is enabled by default in IPv4. fixedNodes.Get (0).first-
>SetAttribute ("IpForward", BooleanValue (true));

// DHCP server

ApplicationContainer dhcpServerApp = dhcpHelper.InstallDhcpServer (devNet.Get (3),
Ipv4Address ("172.30.0.12"), Ipv4Address ("172.30.0.0"), Ipv4Mask ("/24"), Ipv4Address
("172.30.0.10"), Ipv4Address

("172.30.0.15"), Ipv4Address ("172.30.0.17"));

// This is just to show how it can be done. DynamicCast<DhcpServer> (dhcpServerApp.Get (0))-
>AddStaticDhcpEntry (devNet.Get (2)- >GetAddress (), Ipv4Address ("172.30.0.14"));
```

```
dhcpServerApp.Start (Seconds (0.0));

dhcpServerApp.Stop (stopTime);

// DHCP clients

NetDeviceContainer dhcpClientNetDevs;

dhcpClientNetDevs.Add (devNet.Get (0));

dhcpClientNetDevs.Add (devNet.Get (1));

dhcpClientNetDevs.Add (devNet.Get (2));

ApplicationContainer dhcpClients = dhcpHelper.InstallDhcpClient (dhcpClientNetDevs);

dhcpClients.Start (Seconds (1.0));

dhcpClients.Stop (stopTime);

UdpEchoServerHelper echoServer (9);

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

serverApps.Start (Seconds (0.0));

serverApps.Stop (stopTime);

UdpEchoClientHelper echoClient (p2pInterfaces.GetAddress (1), 9);

echoClient.SetAttribute ("MaxPackets", UintegerValue (100));

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

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

ApplicationContainer clientApps = echoClient.Install (nodes.Get (1));

clientApps.Start (Seconds (10.0));

clientApps.Stop (stopTime);

MobilityHelper mobility;
mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
mobility.Install(nodes);
AnimationInterface anim("dhcp.xml");
AnimationInterface::SetConstantPosition(nodes.Get(0),10,25);
AnimationInterface::SetConstantPosition(nodes.Get(1),40,25);

anim.EnablePacketMetadata(true);
pointToPoint.EnablePcapAll("dhcp");
```

```
Simulator::Stop (stopTime + Seconds (10.0));
if (tracing)
{
csma.EnablePcapAll ("dhcp-csma");
pointToPoint.EnablePcapAll ("dhcp-p2p");
}
NS_LOG_INFO ("Run Simulation.");
Simulator::Run ();
Simulator::Destroy ();
NS_LOG_INFO ("Done.");
}
```