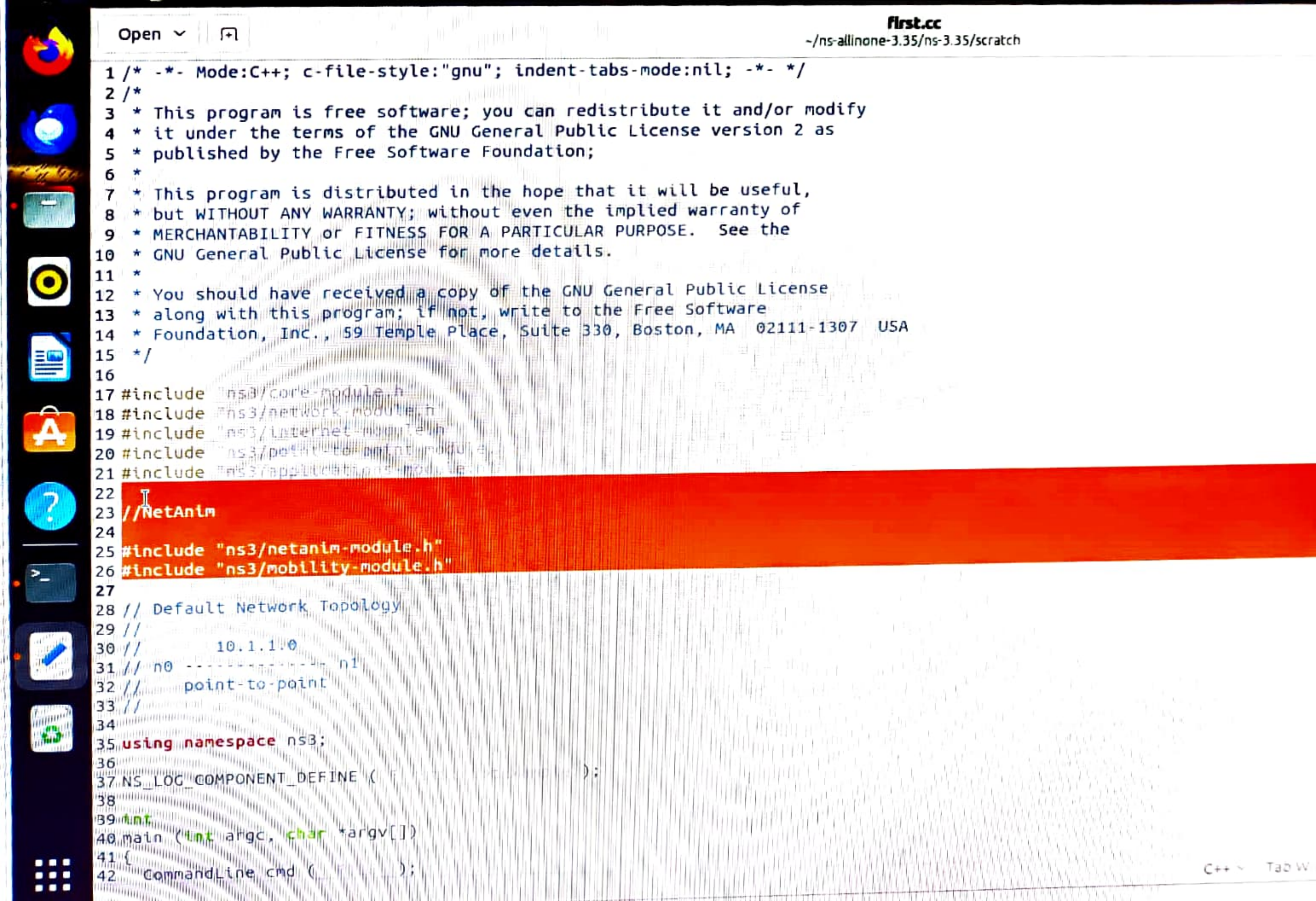
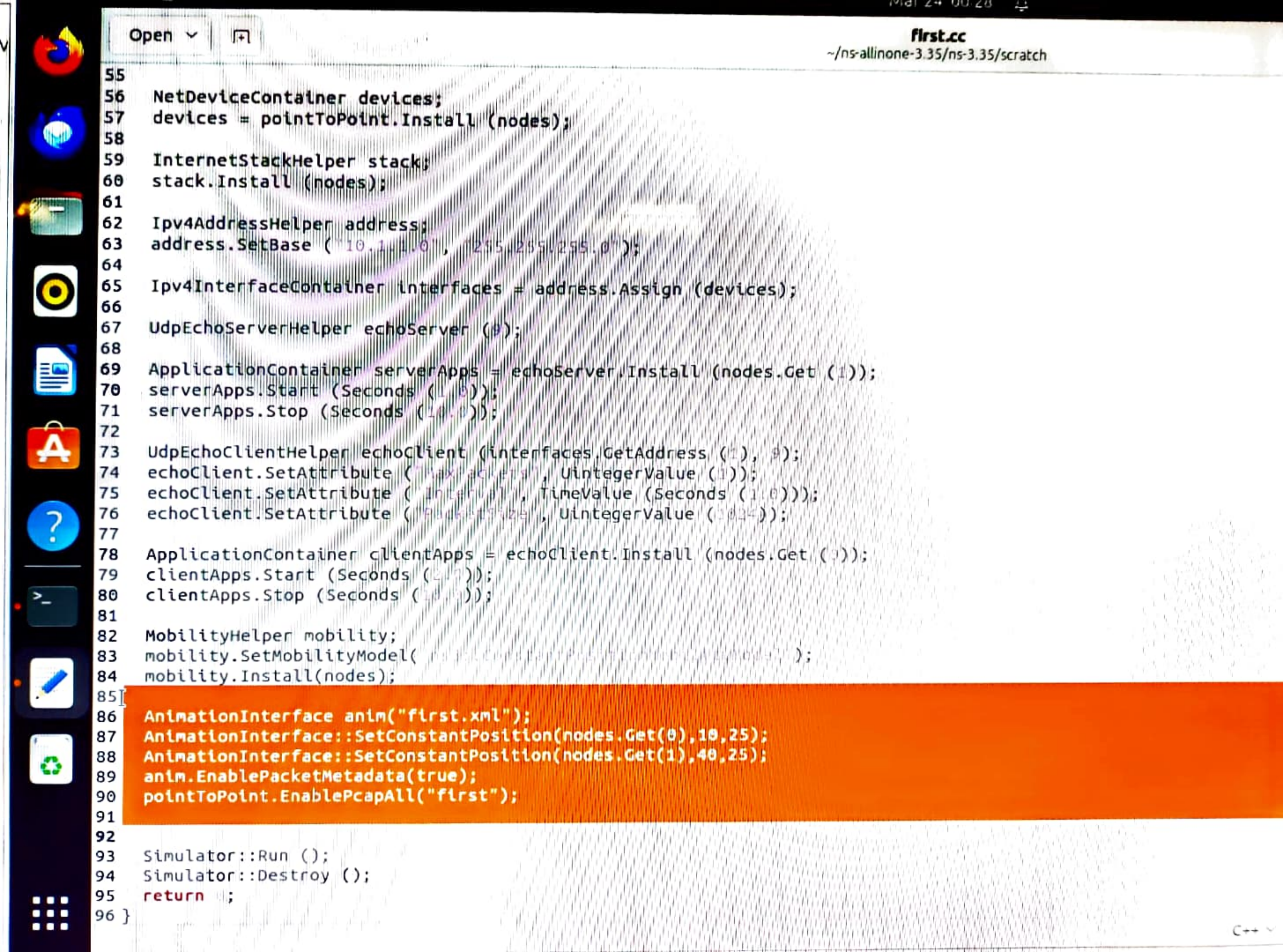
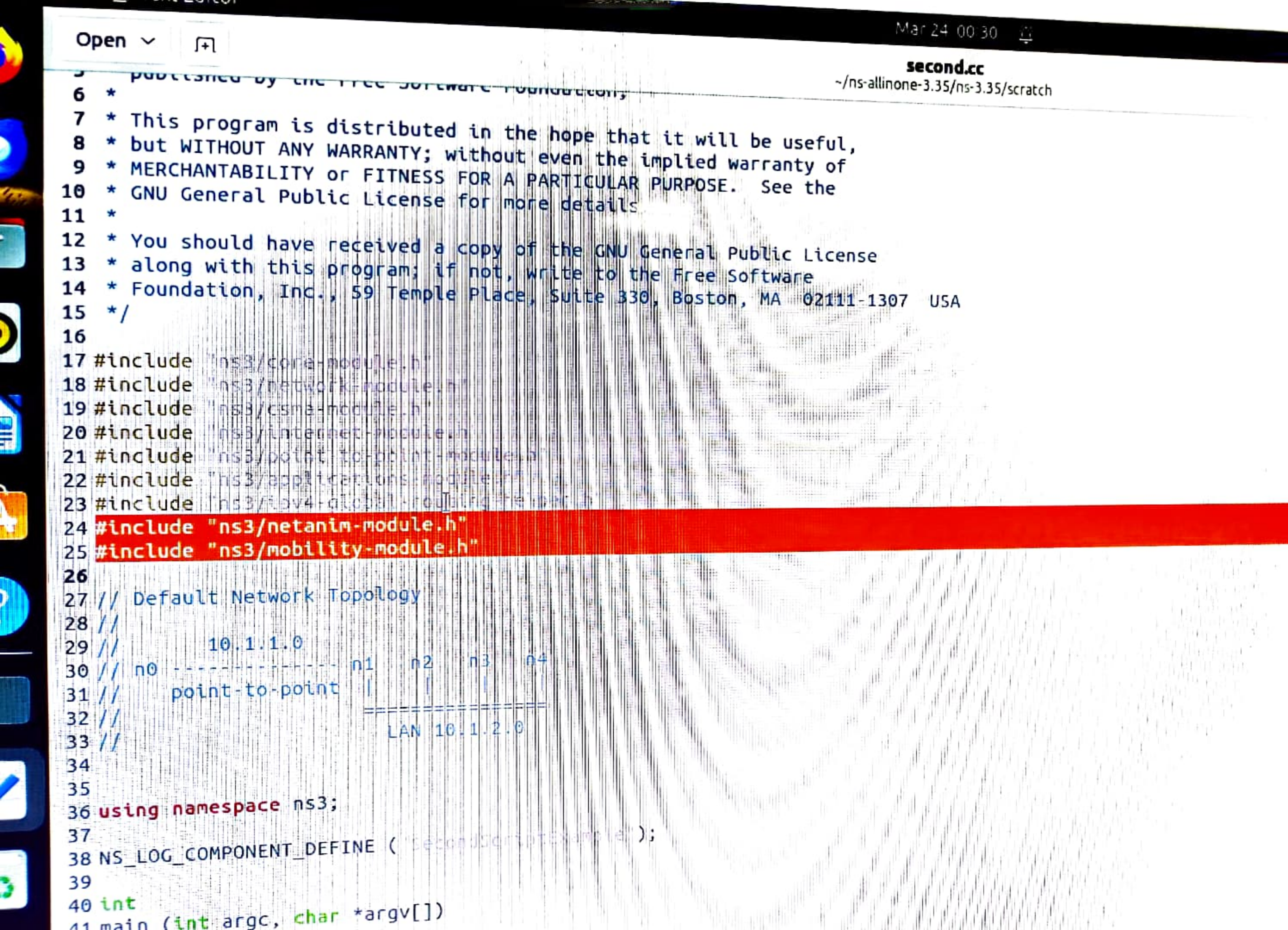


```
/* -*- Mode:C++; c-file-style:"gnu"; indent-tabs-mode:nil; -*- */
/*
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License version 2 as
 * published by the Free Software Foundation;
 *
 * This program is distributed in the 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"

//NetAnim

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

// Default Network Topology
//
//       10.1.1.0
// n0 -------------- n1
//    point-to-point
//

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ([illegible]);

int
main (int argc, char *argv[])
{
  CommandLine cmd ([illegible]);
```

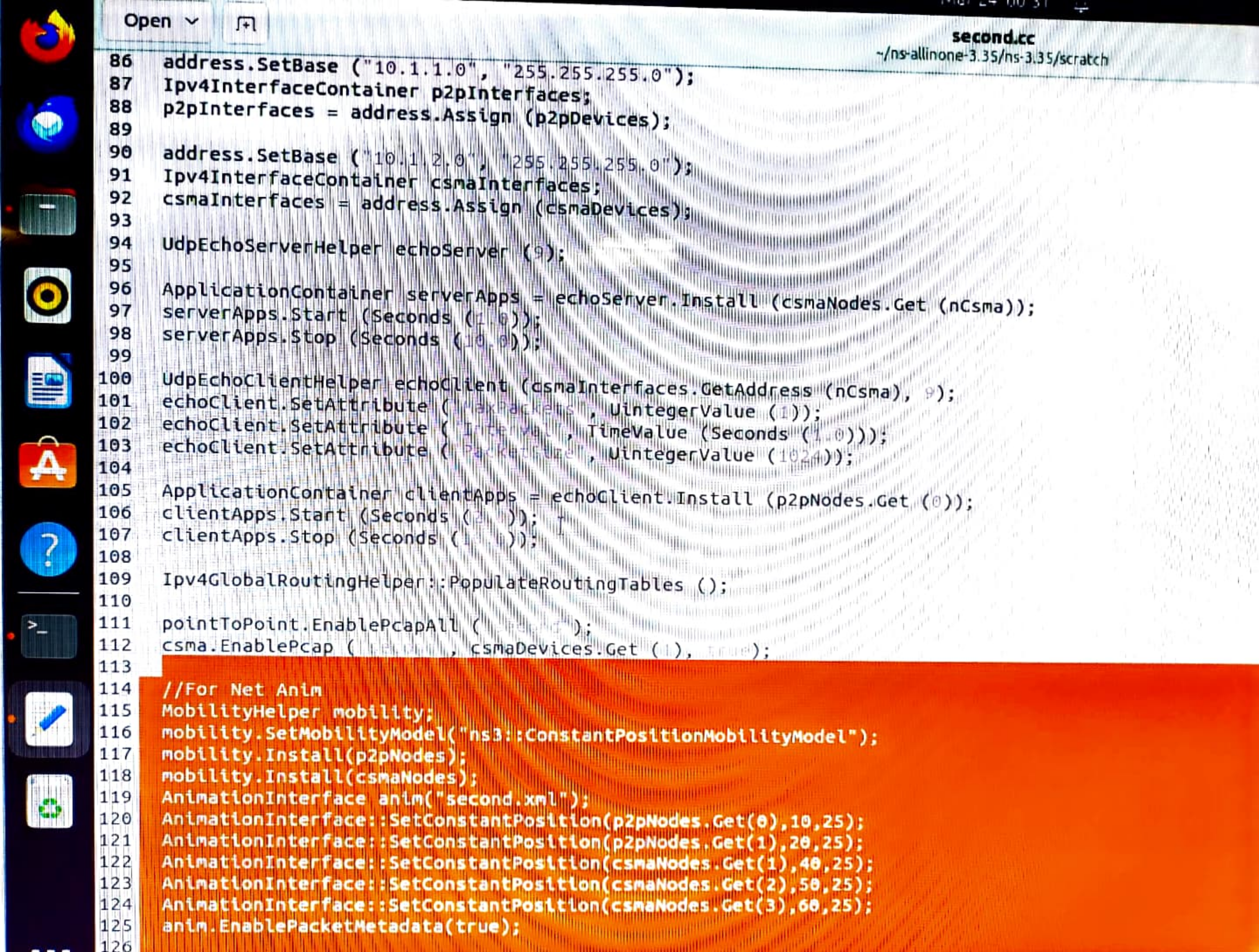


```cpp

  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));

  MobilityHelper mobility;
  mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
  mobility.Install(nodes);

  AnimationInterface anim("first.xml");
  AnimationInterface::SetConstantPosition(nodes.Get(0),10,25);
  AnimationInterface::SetConstantPosition(nodes.Get(1),40,25);
  anim.EnablePacketMetadata(true);
  pointToPoint.EnablePcapAll("first");


  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

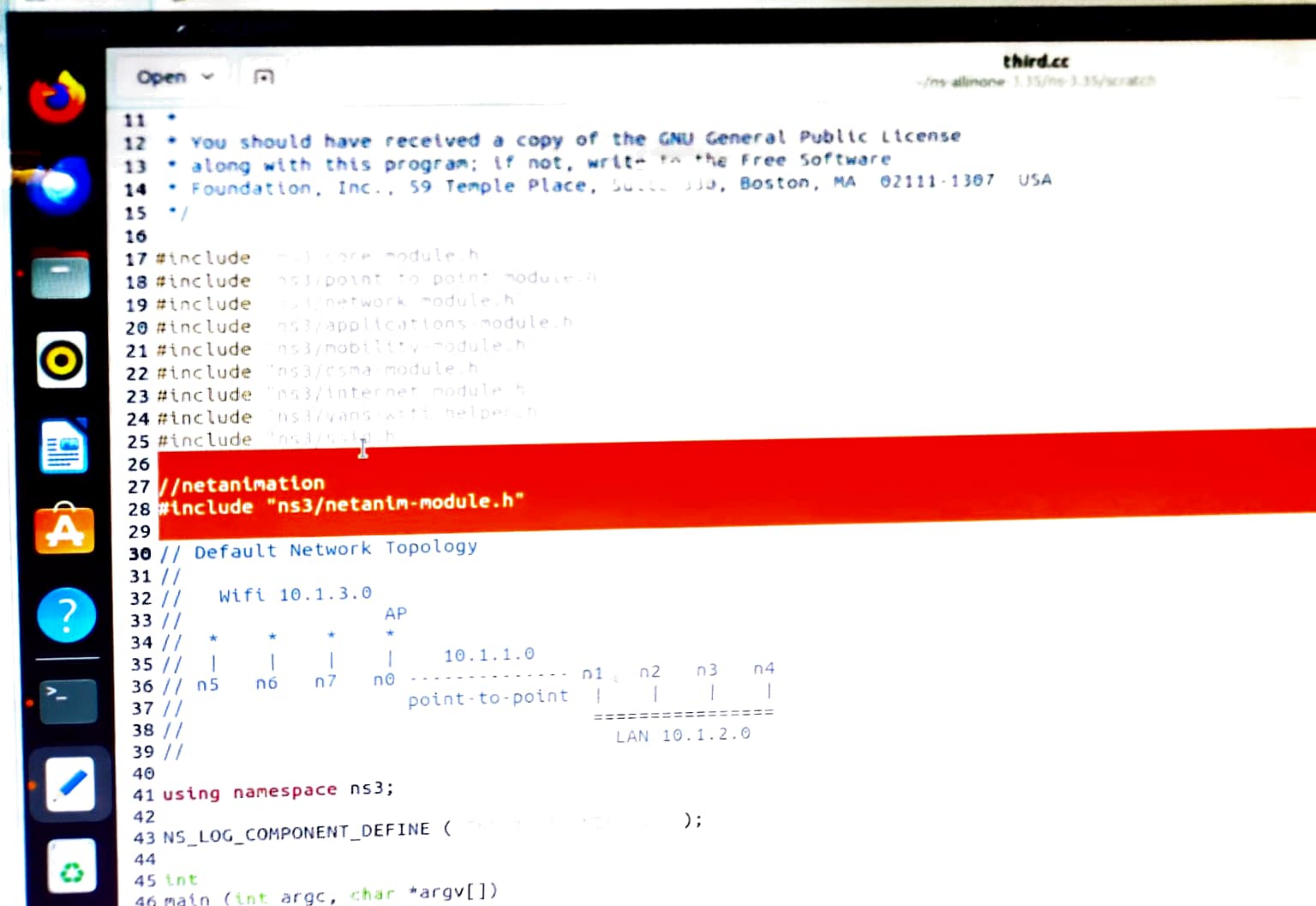


second.cc
~/ns-allinone-3.35/ns-3.35/scratch

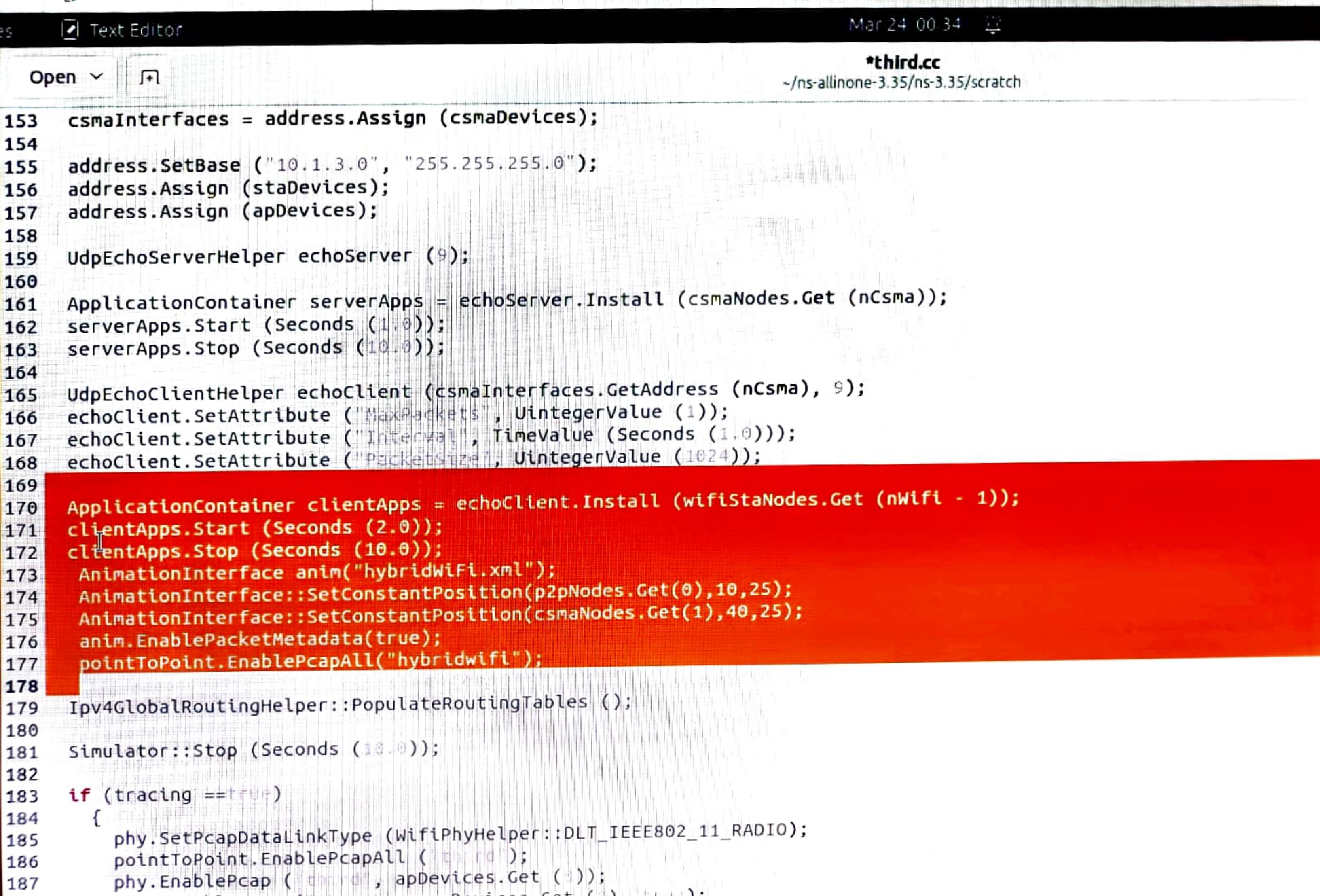
```
 * published by the Free Software Foundation;
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * 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"
#include "ns3/mobility-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[])
```



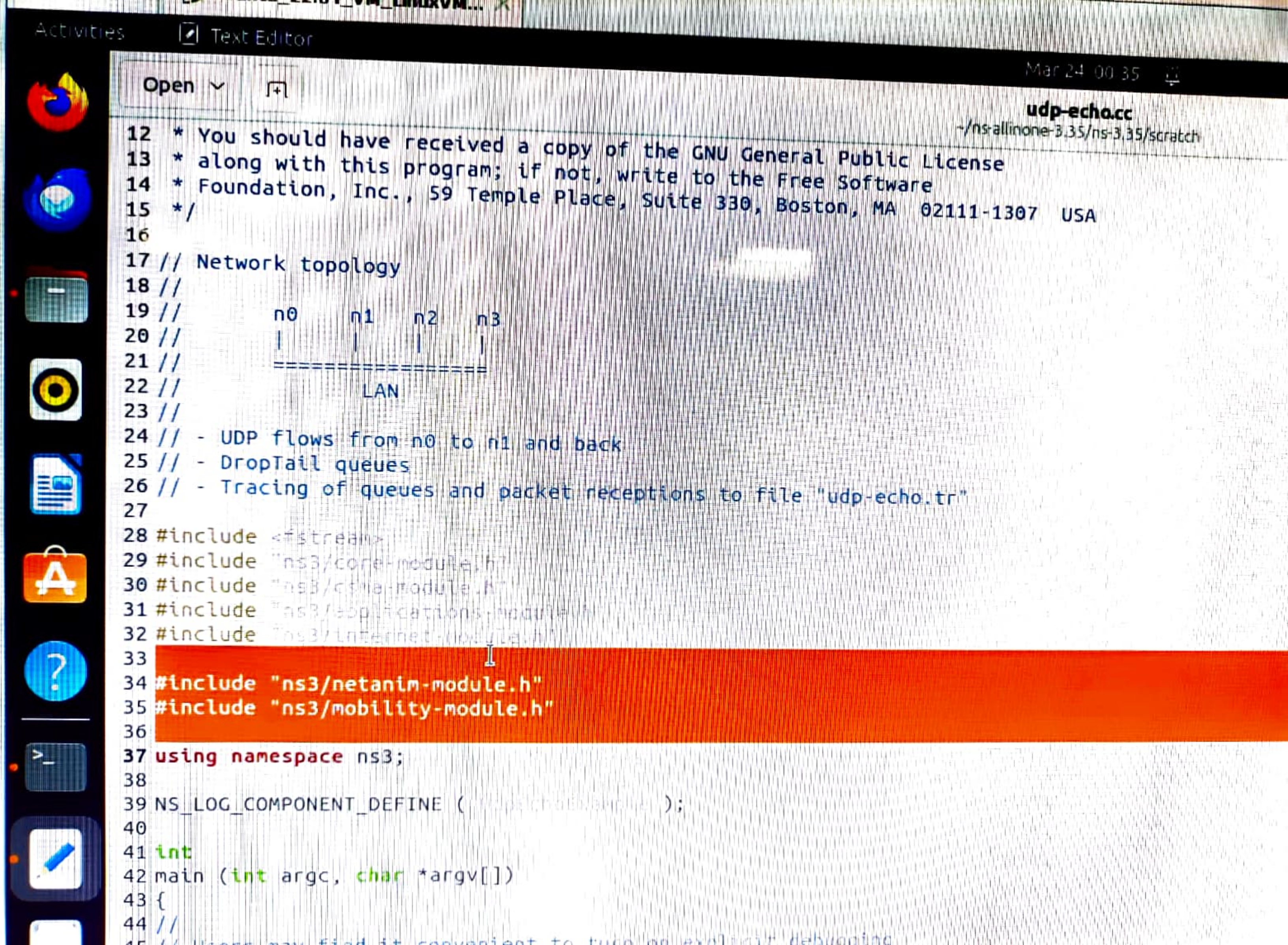
```
  address.SetBase ("10.1.1.0", "255.255.255.0");
  Ipv4InterfaceContainer p2pInterfaces;
  p2pInterfaces = address.Assign (p2pDevices);

  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);

  //For Net Anim
  MobilityHelper mobility;
  mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
  mobility.Install(p2pNodes);
  mobility.Install(csmaNodes);
  AnimationInterface anim("second.xml");
  AnimationInterface::SetConstantPosition(p2pNodes.Get(0),10,25);
  AnimationInterface::SetConstantPosition(p2pNodes.Get(1),20,25);
  AnimationInterface::SetConstantPosition(csmaNodes.Get(1),40,25);
  AnimationInterface::SetConstantPosition(csmaNodes.Get(2),50,25);
  AnimationInterface::SetConstantPosition(csmaNodes.Get(3),60,25);
  anim.EnablePacketMetadata(true);
```



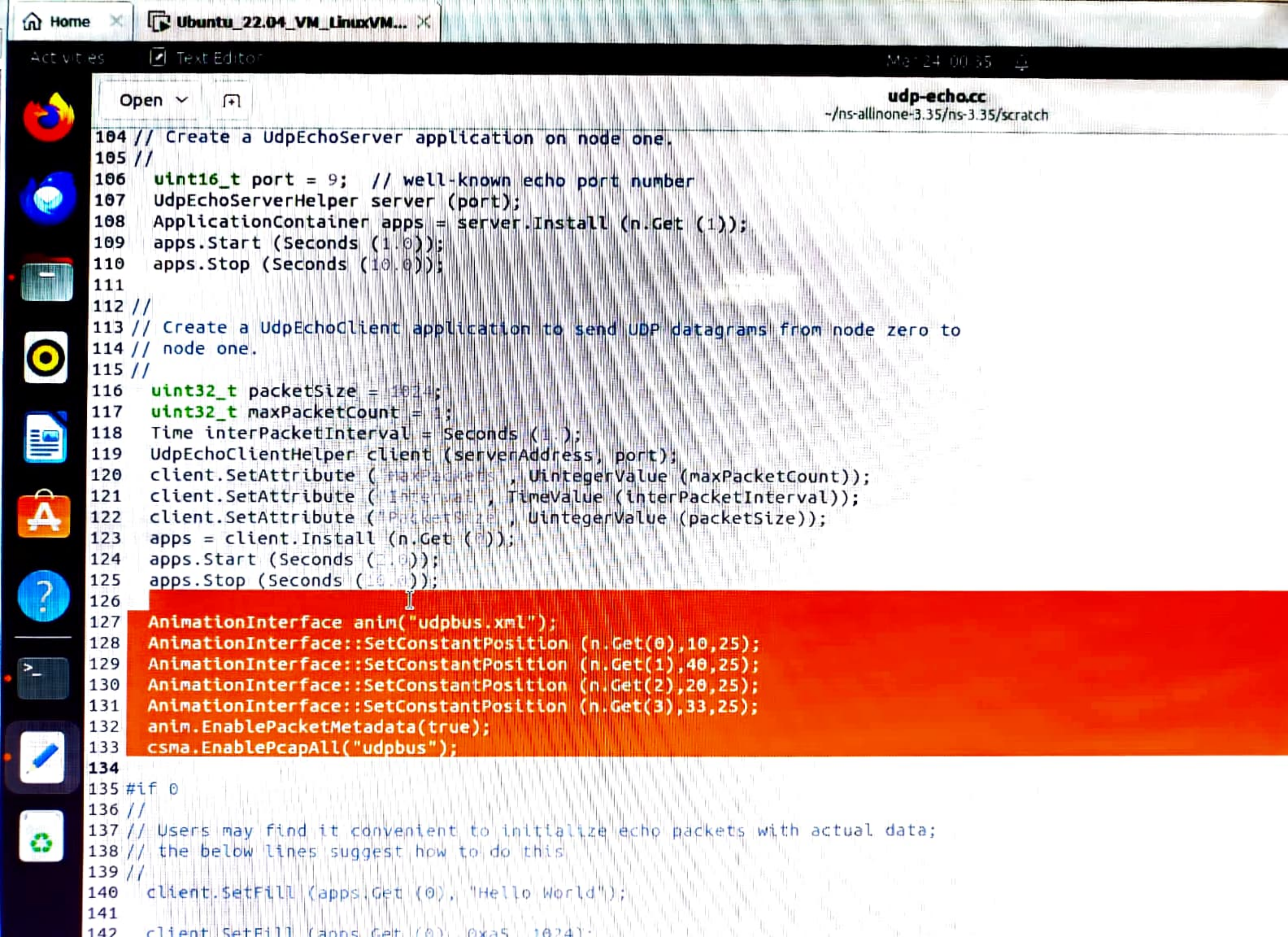
```
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 */

#include "ns3/core-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/network-module.h"
#include "ns3/applications-module.h"
#include "ns3/mobility-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-module.h"
#include "ns3/yans-wifi-helper.h"
#include "ns3/ssid.h"

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

// Default Network Topology
//
//   Wifi 10.1.3.0
//                 AP
//  *    *    *    *
//  |    |    |    |    10.1.1.0
// n5   n6   n7   n0 -------------- n1   n2   n3   n4
//                   point-to-point  |    |    |    |
//                                   ================
//                                     LAN 10.1.2.0
//

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ( [illegible] );

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



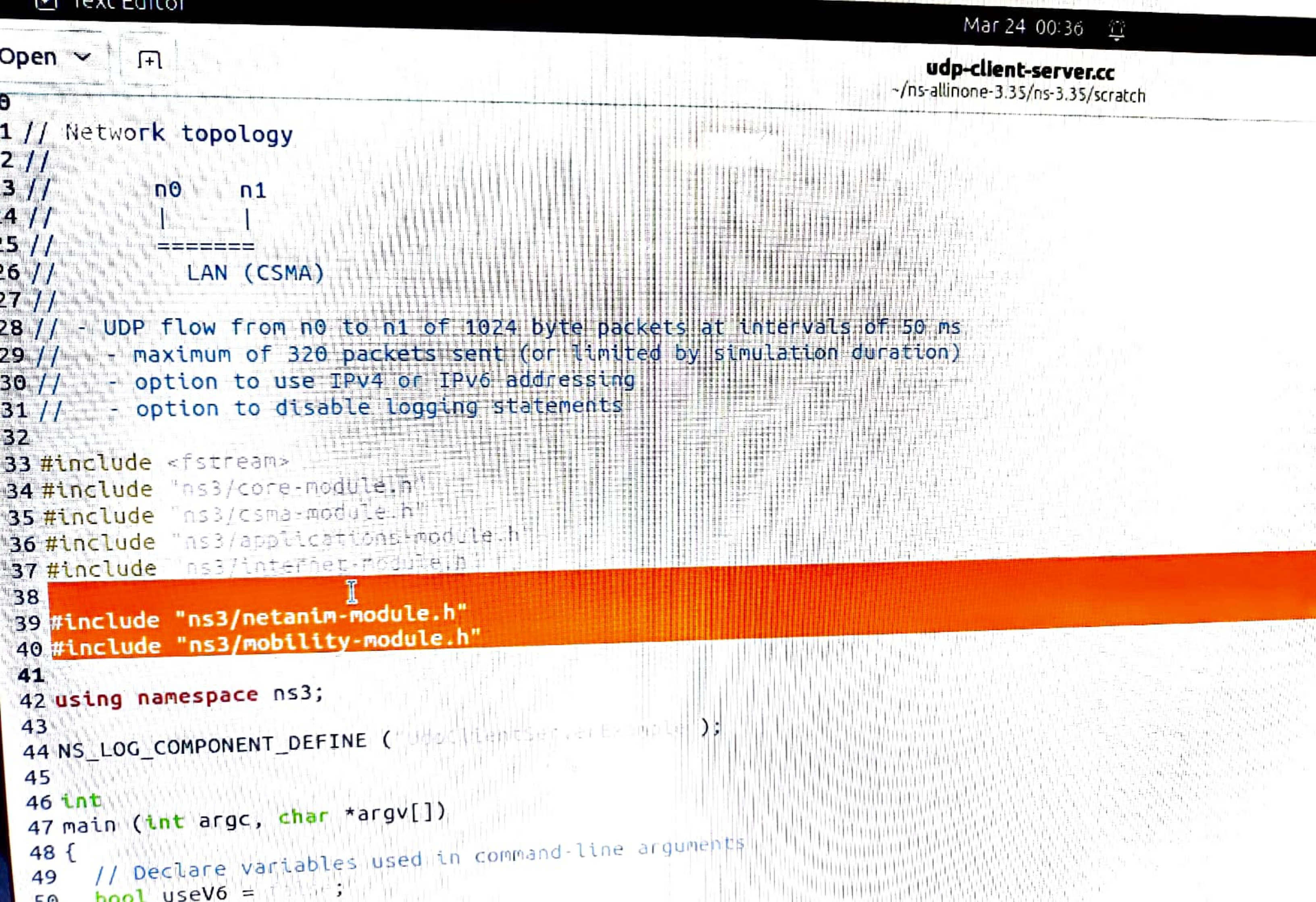
```
  csmaInterfaces = address.Assign (csmaDevices);

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

  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 (wifiStaNodes.Get (nWifi - 1));
  clientApps.Start (Seconds (2.0));
  clientApps.Stop (Seconds (10.0));
   AnimationInterface anim("hybridWiFi.xml");
   AnimationInterface::SetConstantPosition(p2pNodes.Get(0),10,25);
   AnimationInterface::SetConstantPosition(csmaNodes.Get(1),40,25);
   anim.EnablePacketMetadata(true);
   pointToPoint.EnablePcapAll("hybridwifi");

  Ipv4GlobalRoutingHelper::PopulateRoutingTables ();

  Simulator::Stop (Seconds (10.0));

  if (tracing ==true)
    {
      phy.SetPcapDataLinkType (WifiPhyHelper::DLT_IEEE802_11_RADIO);
      pointToPoint.EnablePcapAll ("third");
      phy.EnablePcap ("third", apDevices.Get (0));
```



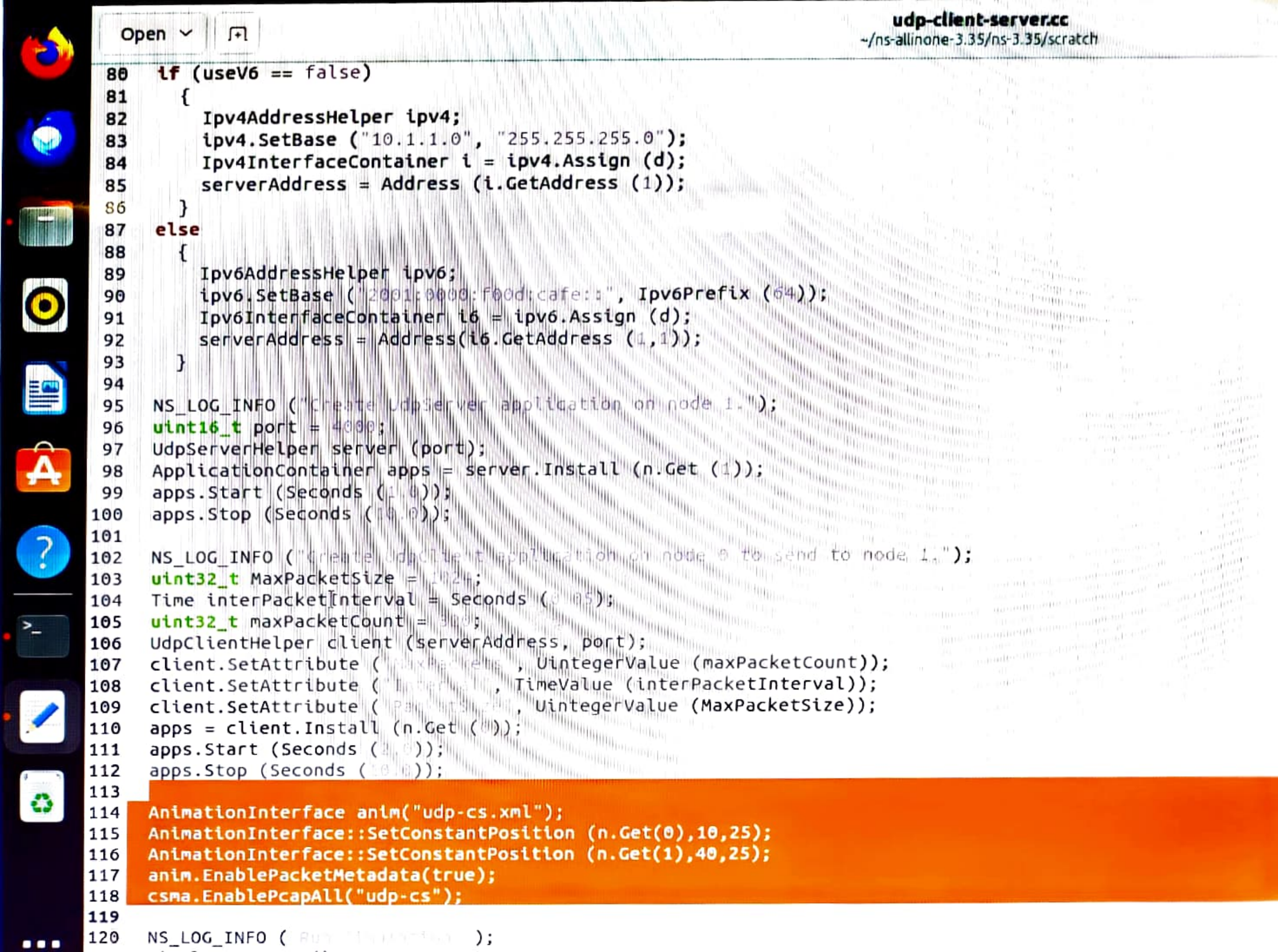
```
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 */

// 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>
#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"
#include "ns3/mobility-module.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("UdpEchoExample");

int
main (int argc, char *argv[])
{
//
// Users may find it convenient to turn on explicit debugging
```


Home
Ubuntu_22.04_VM_LinuxVM...
Activities
Text Editor
Mar 24 00:35
Open
udp-echo.cc
~/ns-allinone-3.35/ns-3.35/scratch

```
// Create a UdpEchoServer application on node one.
//
  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));

//
// Create a UdpEchoClient application to send UDP datagrams from node zero to
// node one.
//
  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));

 AnimationInterface anim("udpbus.xml");
 AnimationInterface::SetConstantPosition (n.Get(0),10,25);
 AnimationInterface::SetConstantPosition (n.Get(1),40,25);
 AnimationInterface::SetConstantPosition (n.Get(2),20,25);
 AnimationInterface::SetConstantPosition (n.Get(3),33,25);
 anim.EnablePacketMetadata(true);
 csma.EnablePcapAll("udpbus");

#if 0
//
// Users may find it convenient to initialize echo packets with actual data;
// the below lines suggest how to do this
//
  client.SetFill (apps.Get (0), "Hello World");

  client.SetFill (apps.Get (0), 0xa5, 1024);
```

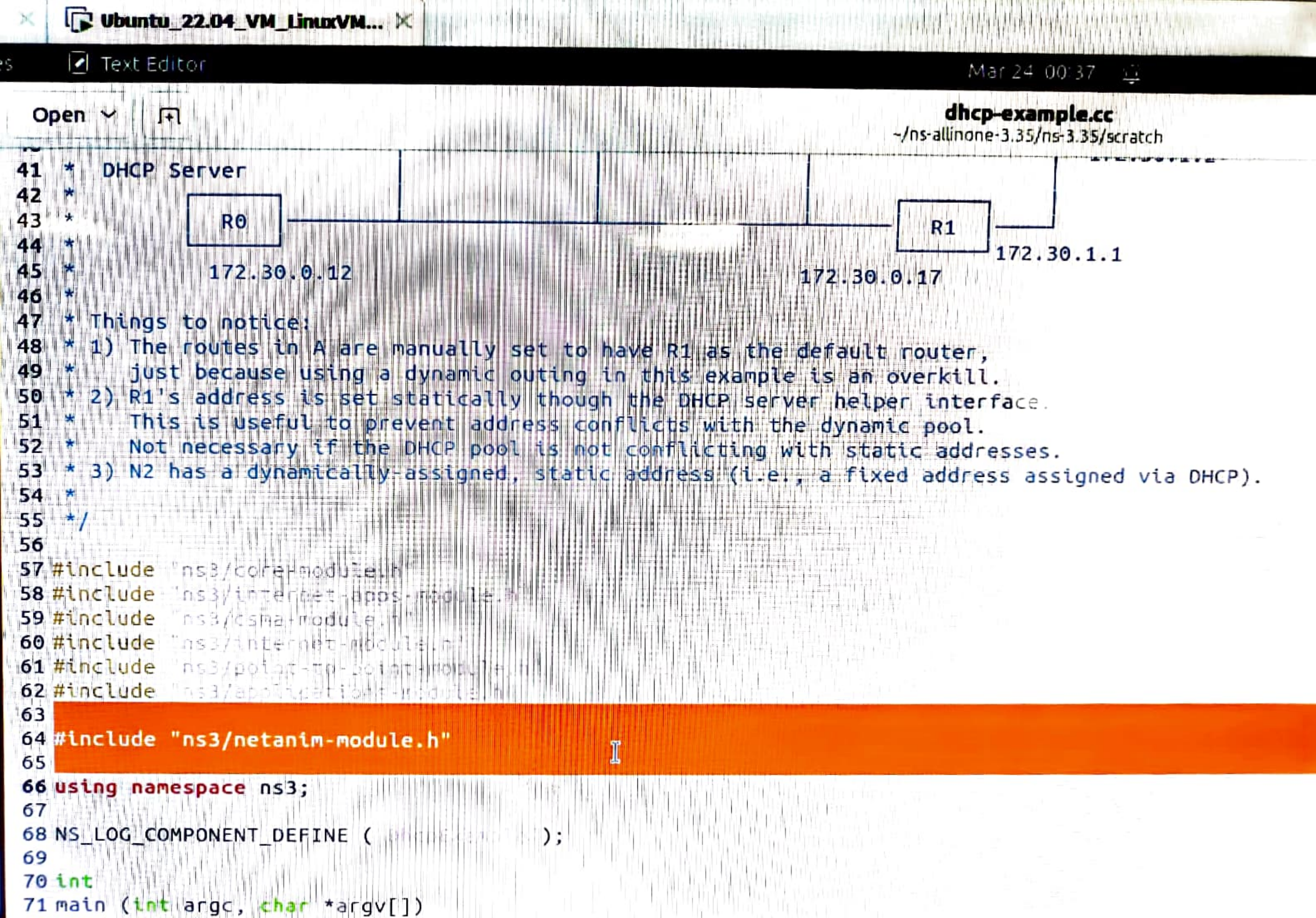


```
// Network topology
//
//       n0    n1
//       |     |
//       =======
//         LAN (CSMA)
//
// - UDP flow from n0 to n1 of 1024 byte packets at intervals of 50 ms
//   - maximum of 320 packets sent (or limited by simulation duration)
//   - option to use IPv4 or IPv6 addressing
//   - option to disable logging statements

#include <fstream>
#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"
#include "ns3/mobility-module.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("UdpClientServerExample");

int
main (int argc, char *argv[])
{
  // Declare variables used in command-line arguments
  bool useV6 = [illegible];
```



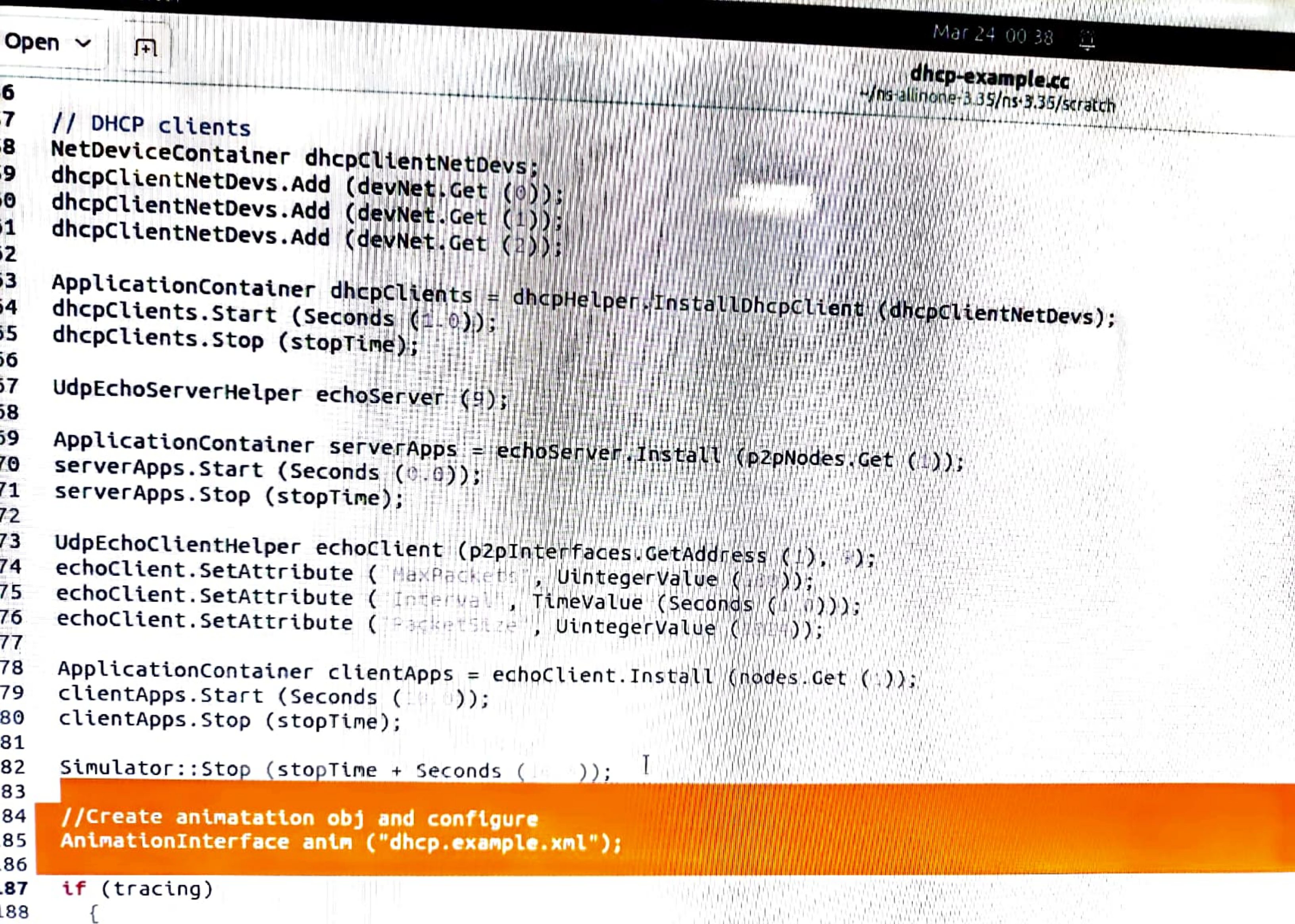
udp-client-server.cc
~/ns-allinone-3.35/ns-3.35/scratch

```
 if (useV6 == false)
   {
     Ipv4AddressHelper ipv4;
     ipv4.SetBase ("10.1.1.0", "255.255.255.0");
     Ipv4InterfaceContainer i = ipv4.Assign (d);
     serverAddress = Address (i.GetAddress (1));
   }
 else
   {
     Ipv6AddressHelper ipv6;
     ipv6.SetBase ("2001:0000:f00d:cafe::", Ipv6Prefix (64));
     Ipv6InterfaceContainer i6 = ipv6.Assign (d);
     serverAddress = Address(i6.GetAddress (1,1));
   }

 NS_LOG_INFO ("Create UdpServer application on node 1.");
 uint16_t port = 4000;
 UdpServerHelper server (port);
 ApplicationContainer apps = server.Install (n.Get (1));
 apps.Start (Seconds (1.0));
 apps.Stop (Seconds (10.0));

 NS_LOG_INFO ("Create UdpClient application on node 0 to send to node 1.");
 uint32_t MaxPacketSize = 1024;
 Time interPacketInterval = Seconds (0.05);
 uint32_t maxPacketCount = 320;
 UdpClientHelper client (serverAddress, port);
 client.SetAttribute ("MaxPackets", UintegerValue (maxPacketCount));
 client.SetAttribute ("Interval", TimeValue (interPacketInterval));
 client.SetAttribute ("PacketSize", UintegerValue (MaxPacketSize));
 apps = client.Install (n.Get (0));
 apps.Start (Seconds (2.0));
 apps.Stop (Seconds (10.0));

 AnimationInterface anim("udp-cs.xml");
 AnimationInterface::SetConstantPosition (n.Get(0),10,25);
 AnimationInterface::SetConstantPosition (n.Get(1),40,25);
 anim.EnablePacketMetadata(true);
 csma.EnablePcapAll("udp-cs");

 NS_LOG_INFO ("Run Simulation.");
```



Ubuntu_22.04_VM_LinuxVM...
Text Editor
Mar 24 00:37
Open
dhcp-example.cc
~/ns-allinone-3.35/ns-3.35/scratch

```
 *  DHCP Server
 *
 *       ┌──────┐                                                   ┌──────┐
 *       │  R0  │─────────────────────────────────────────────────── │  R1  │────┘
 *       └──────┘                                                   └──────┘ 172.30.1.1
 *        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"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ([illegible]);

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



```
  // 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);

  Simulator::Stop (stopTime + Seconds (10.0));

  //Create animatation obj and configure
  AnimationInterface anim ("dhcp.example.xml");

  if (tracing)
    {
```