



L-Università ta' Malta
Faculty of Information &
Communication Technology

Lab 1 Report

Manwel Bugeja

June 1, 2021

Contents

1	Introduction	2
2	Code changes	2
2.1	NodeContainer	2
2.2	PointToPointHelper	2
2.3	InternetStackHelper	3
2.4	Ipv4AddressHelper	3
2.5	UdpEchoServerHelper	3
2.6	UdpEchoClientHelper	3
2.7	PcapHelper	4
3	Investigation	4
3.1	Establishment	4
3.2	Analysis	5

1 Introduction

The aim of this assignment was to replace the helper classes in an NS3 simulation program. Section 2 describes the code replacement of each helper class. On top of that, this said section also describes how the specified code correlates with the Open Systems Interconnection (OSI) Model. Section 3 investigates a packet captured.

2 Code changes

2.1 NodeContainer

The node container was removed by simply creating the nodes separately and not storing them in a container at all.

2.2 PointToPointHelper

Listing 1: Creating a P2P net device

```
Ptr<PointToPointChannel> pointToPointChannel =
    CreateObjectWithAttributes<PointToPointChannel>(
        "Delay", StringValue("3ms"));

Ptr<Node> nodeA = CreateObject<Node>();
//nodeContainer.Add(nodeA);

Ptr<PointToPointNetDevice> pointToPointNetDeviceA =
    CreateObjectWithAttributes<PointToPointNetDevice>("
        DataRate", StringValue("5Mbps"));
nodeA->AddDevice(pointToPointNetDeviceA);

Ptr<Queue<Packet>> > queueA = CreateObject<DropTailQueue
    <Packet>> > ();
pointToPointNetDeviceA->SetQueue(queueA);

pointToPointNetDeviceA->Attach(pointToPointChannel);

netDeviceContainer.Add(pointToPointNetDeviceA);
```

Referring to listing 1, this part of the code starts by creating a point to point channel. This channel acts as a medium between the point to point devices

for example a wire. This means that this corresponds to the first layer of the OSI model.

After that, the first node was created along with its corresponding point to point (P2P) net device. The same attributes as the original code were used for the net device. Then it was attached to the node. A drop tail queue was initiated to be used by the net device. Finally, the previously created channel was also attached to the net device and the device was added to its corresponding container. This process was repeated another time for a second node. The P2P protocol corresponds to the data link layer i.e. layer 2 of the OSI model.

2.3 InternetStackHelper

For the Network layer and Transport layer, several protocols were bundled with the nodes. These are `ArpL3Protocol`, `Ipv4L3Protocol`, `Icmpv4L4Protocol`, `UdpL4Protocol`. The first three being make part of the Network Layer while the `UdpL4Protocol` makes part of the Transport layer. Static routing was used for IPv4. A traffic control layer and packet socket factory were also aggregated to the nodes.

2.4 Ipv4AddressHelper

This part of the code also concerns the Internet layer of the nodes. Here, the devices are each given a unique IPv4 address. The nodes are given the IP addresses 192.168.1.1 and 192.168.1.2 with the default subnet mask for class C i.e. 255.255.255.0. Here, like in `Ipv4AddressHelper`, addresses are only incremented (the code does not take into account a system of IPs).

2.5 UdpEchoServerHelper

As discussed in section 2.3 the nodes already have a transport layer. This section makes part of the application layer for the node. Here, a `UdpEchoServer` object is created and attached to one of the nodes as an application. Henceforth this node starts acting as the server. The start and stop time are specified.

2.6 UdpEchoClientHelper

Similarly to section 2.5, in this part of the code an object of type `UdpEchoClient` is created. The attributes were set with the `PacketSize` being set to 2048.

This is greater than the MTU which is 1500, to fit the required specification. This is then attached to the other node so it starts acting as a client.

2.7 PcapHelper

When doing this section, it was necessary to learn how different classes communicate with each other in NS3. Communication within classes is done via callbacks. Here, function pointers are used to communicate the data. This also applies to communication between the different layers of the system, as most of them are represented by different instances of classes.

This call back feature was used to enable PCAP tracing without the use of the helper. First off, a PCAP file wrapper object was created. This was used to open a file called `firstPcapA`. This is the PCAP tracing file for the first node. Following that, the method `TraceConnectWithoutCallback` is used and a function pointer called `DefaultSink` is passed as one of the parameters as required by the method signature.

3 Investigation

3.1 Establishment

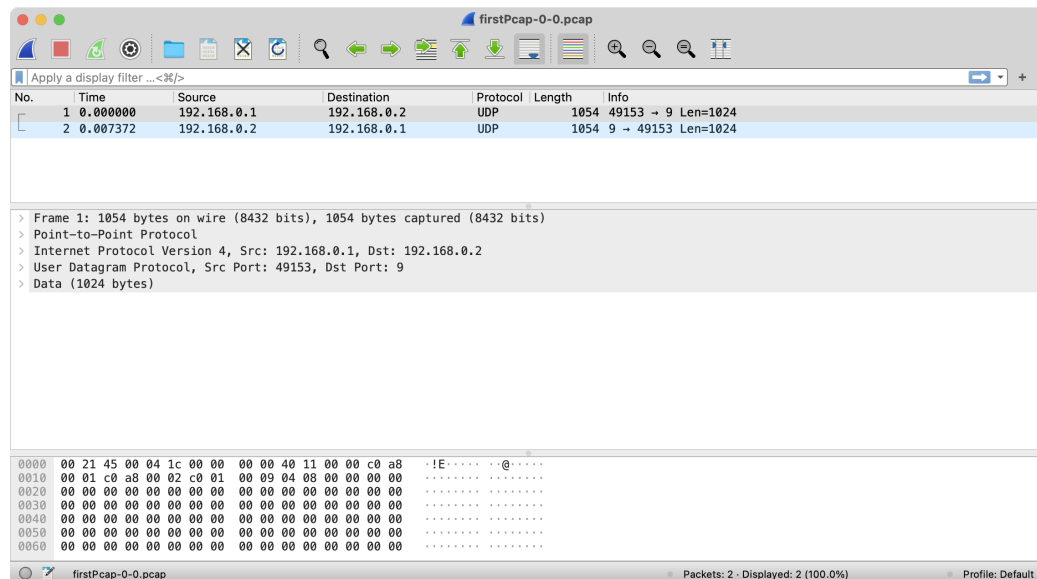


Figure 1: Original packet capture file

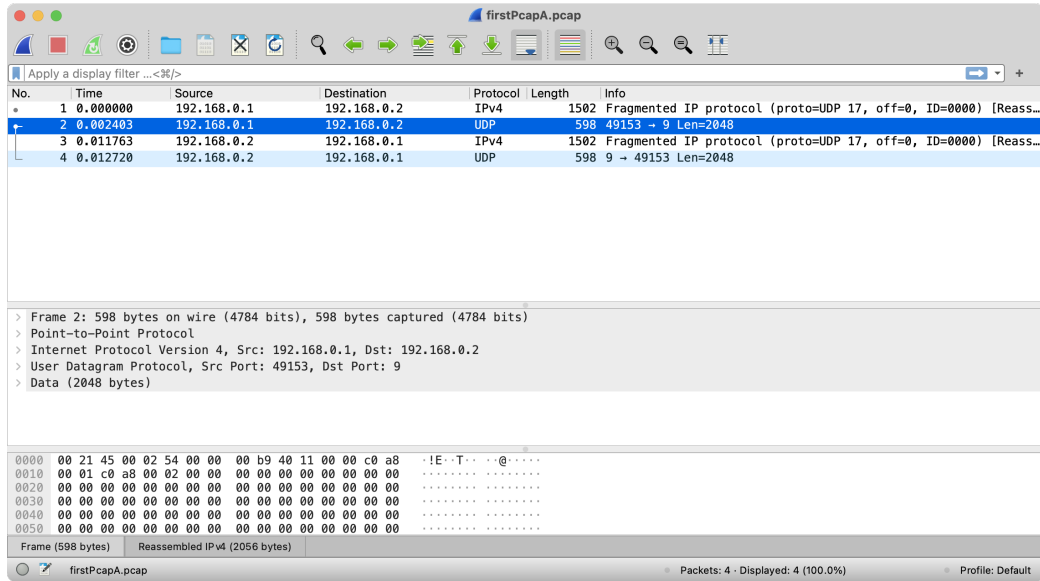


Figure 2: Packet capture file without helper

3.2 Analysis

Figure 1 show the pcap file captured on the given simulation, while figure 2 shows the one capture without the use of any helpers. These figures confirm that the UDP protocol was used in the transport layer. The figures also show the IP addresses of each node. Moreover, one can see the bytes of data that were transferred; the data link protocol (P2P); the Internet protocol (version 4). These confirm that the situation is working as intended.