Lab Assignment 6

NS-3 Simulator

Experiment 1:

1. Create a simple topology of Three nodes (Node0, Node1, Node2), Create two point-to-point links between Node0 --> Node1 and Node1 --> Node2. Let it be of a fixed data rate Rate1 1. Setup a UdpClient on one Node0 and a UdpServer on Node2.
2. Start the client application, and measure end to end throughput whilst varying the latency of the link.
3. Now add another client application to Node0 and a server instance to Node2. What do you need to configure to ensure that there is no conflict?
4. Repeat step 3 with the extra client and server application instances. Show screenshots of pcap traces which indicate that delivery is made to the appropriate server instance.

Experiment 2:

n0 n1 n2 n3 ----------n0 n1 n2 n3
| | | | point-to-point | | | |
========== 10.1.3.0 ==========
LAN 10.1.1.0 LAN 10.1.2.0

1. Create two bus topology(LAN) of four nodes each
2. Connect the last node(n3) of first LAN with first node of second LAN using point-to-point link 3. Set DataRate of point-to-point link to 5Mbps and Delay to 2ms
4. Set DataRate of LAN to 100Mbps and Delay to 6560ns (for both the LAN) 5. Set up UdpClient on second node(n1) of first LAN and UdpServer on third node(n2) of second

LAN

6. Generate .pcap(wireshark) , .tr(TraceMatrices) and .xml(FlowMonitor) file and analyze the file.

Sample solution for Experiment 1:

This simulation will include two point-to-point links, UDP client-server applications, and the analysis of end-to-end throughput while varying latency.

1. Topology Setup

We'll create three nodes and connect them with two point-to-point links. Each link can have different latency parameters that you can vary to measure the impact on throughput.

2. UDP Client and Server Setup

A UDP client will be set up on Node0, and a UDP server will be on Node2. We will start with one client-server pair and later add another pair to the same nodes.

3. Configuration for Multiple Clients and Servers

To avoid conflicts when running multiple clients and servers on the same nodes, we will use different port numbers for each server and configure the clients to target these ports accordingly.

4. Packet Capture and Analysis

We will enable pcap tracing to capture packets and use Wireshark to analyze these traces to confirm that packets are being delivered to the correct server instances.

```
#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"
#include "ns3/flow-monitor-module.h"


#include "ns3/flow-monitor-helper.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("ThreeNodeTopology");

int main () {
    // Log level setting
    LogComponentEnable("UdpEchoClientApplication", LOG_LEVEL_INFO);
    LogComponentEnable("UdpEchoServerApplication", LOG_LEVEL_INFO);

    // Create nodes
    NodeContainer nodes;
    nodes.Create(3);

    // Create point-to-point link attributes
    PointToPointHelper pointToPoint;
    pointToPoint.SetDeviceAttribute("DataRate", StringValue("5Mbps"));
    pointToPoint.SetChannelAttribute("Delay", StringValue("1ms"));

    // Install devices and channels between Node0 and Node1, and Node1
and Node2
    NetDeviceContainer device0_1 = pointToPoint.Install(nodes.Get(0),
nodes.Get(1));
    NetDeviceContainer device1_2 = pointToPoint.Install(nodes.Get(1),
nodes.Get(2));
```

```
    // Install Internet Stack
    InternetStackHelper stack;
    stack.Install(nodes);

    // Assign IP addresses
    Ipv4AddressHelper address;
    address.SetBase("10.1.1.0", "255.255.255.0");
    Ipv4InterfaceContainer interfaces0_1 = address.Assign(device0_1);
    address.SetBase("10.1.2.0", "255.255.255.0");
    Ipv4InterfaceContainer interfaces1_2 = address.Assign(device1_2);

    // Create UDP echo server on Node2
    uint16_t serverPort = 9;
    UdpEchoServerHelper server1(serverPort);
    ApplicationContainer serverApp1 = server1.Install(nodes.Get(2));
    serverApp1.Start(Seconds(1.0));
    serverApp1.Stop(Seconds(10.0));

    // Create UDP echo client on Node0
    UdpEchoClientHelper client1(interfaces1_2.GetAddress(1),
serverPort);
    client1.SetAttribute("MaxPackets", UintegerValue(10));
    client1.SetAttribute("Interval", TimeValue(Seconds(1.0)));
    client1.SetAttribute("PacketSize", UintegerValue(1024));

    ApplicationContainer clientApp1 = client1.Install(nodes.Get(0));
    clientApp1.Start(Seconds(2.0));
    clientApp1.Stop(Seconds(10.0));

    // Adding another UDP echo server on Node2 on a different port
```

```
    uint16_t serverPort2 = 10;
    UdpEchoServerHelper server2(serverPort2);
    ApplicationContainer serverApp2 = server2.Install(nodes.Get(2));
    serverApp2.Start(Seconds(1.0));
    serverApp2.Stop(Seconds(10.0));

    // Adding another UDP echo client on Node0
    UdpEchoClientHelper client2(interfaces1_2.GetAddress(1),
serverPort2);
    client2.SetAttribute("MaxPackets", UintegerValue(10));
    client2.SetAttribute("Interval", TimeValue(Seconds(1.0)));
    client2.SetAttribute("PacketSize", UintegerValue(1024));

    ApplicationContainer clientApp2 = client2.Install(nodes.Get(0));
    clientApp2.Start(Seconds(2.0));
    clientApp2.Stop(Seconds(10.0));

    // Enable pcap tracing
    pointToPoint.EnablePcapAll("three-node-topology");

    // Run the simulation
    Simulator::Run();
    Simulator::Destroy();

    return 0;
}
```