Assignment 3: Network Simulations using NS-3

Deadline: 12th September 2024, 11:59 PM

Goal:

Study and understand how to set up network topologies and traffic flows using ns-3 and analyze performance metrics with the help of tools like Wireshark and flowMonitor while learning how to configure some error models and observe the variation in metrics thus obtained.

Part 1: Getting Familiar [5 Points]

Using lab_fm.cc as a reference, achieve the following tasks.

1. Create a simple topology of two nodes (Node1, Node2) separated by a point-to-point link.
2. Setup a UdpClient on one Node1 and a UdpServer on Node2. Let it be of a fixed data rate Rate1.
3. Start the client application and measure end-to-end throughput while varying the latency of the link, and submit the screenshots of the outputs. Justify the results.
4. Now add another client application to Node1 and a server instance to Node2. What do you need to configure to ensure that there is no conflict?

Part 2: Advancing to the Next Level [32 Points]

Using demo.cc as a reference, try and understand how to set up different data rates for different links, how to print routing tables of routers and how to finally visualize the topology and simulation visually through NetAnim.

PART A

Create a custom network topology as defined below: **[5 Points]**

The topology consists of multiple nodes connected via routers. Here's the structure:

1. **n0**, **n1**, and **n2** are connected to **router0**.
5. **router0** is connected to both **router1** and **router2**.

6. **n3**, **n4**, and **n5** are connected to **router1**.
7. **n6** and **n7** are connected to **router2**.

Once you have created the topology, it's time to set up the traffic flows:

1. Set up a TCP flow between **n7 and n4**.
2. Set up a UDP flow between **n1 and n5**.
3. Set up additional TCP flows between **n2,n6** and **n0,n3**.

** Clearly mention the **delays** and **data rates** of the links you choose and ensure that the flows between **n2 and n6** and **n0 and n3** have **different** values set for these than the rest of the traffic flows.

Tasks to be done:

1. Plot the flow vs throughput graph. **[3 Points]**
2. Plot the flow vs total packets lost graph. **[3 Points]**
3. Generate a file '**routing-tables.txt**' wherein you print the routing tables of the 3 routers. **[3 Points]**
4. Generate the **NetAnim trace file** and append that as well. **[1 Point]**

**** Points to be taken note of**:

- Generate the plots using gnuplot or matplotlib or any other framework and share the script.
- Append Screenshots and mark clearly the routers and nodes and the flows.
- Support your FlowMon statistics with screenshots of results wherever possible.
- The flow graphs indicate the results obtained from flowMon statistics and each flow corresponds to the ones set in the topology like between **n7 and n4** or **n1 and n5** so there should be **4** such flows overall, **3 of them TCP and one UDP** so they come on the X-axis and ensure to record values of flows from client to sink app i.e the downlink flow as flowMon also records the reverse flows, **make a note of this**.

PART B

Modify your source code by including an error model.

For this part you have to set up **UdpEchoClient and **UdpEchoServer** Application flows **in place of all the flows** established in **PART A**.

Error models are used to indicate that a packet should be errored, according to the underlying (possibly stochastic or empirical) error model. NS-3 offers the following error models (refer References section for more details).

- RateErrorModel
- ListErrorModel
- ReceiveListErrorModel
- BurstErrorModel

E.g. Rate Error Model has these parameters

- ErrorRate (0,1)
- ErrorUnit [Packet, Bit, Byte]

Run the following commands to know the available attributes of these error models and their default values.

./ns3 --run "scratch/myfirst --PrintAttributes=ns3::RateErrorModel"
./ns3 --run "scratch/myfirst --PrintAttributes=ns3::BurstErrorModel"

Though there is supposed to be an error rate for a given link (e.g., point-to-point channel), NS-3 requires you to set an error rate for each receiver (e.g., NetDevices like PointToPointNetDevice).

The following command shows the default error model for PointToPointNetDevice which is “0” meaning error-free channel.

./ns3 --run "scratch/myfirst --PrintAttributes=ns3::PointToPointNetDevice"

You need to select the error model as RateErrorModel and tune its parameters to measure Packet Delivery Ratio (PDR) in your code by transmitting at least 100 packets in each run of the simulation. As we are interested in the average behavior of the network, repeat each experiment by varying the RngRun value 10 times (i.e. 10 seed values) as one of the command line arguments. Refer to simple-error-model.cc and fifth.cc (refer to the Reference file paths which follows to know about their path in your NS-3 installation) to know how to set the error model for a NetDevice. To ensure the same error rate for both devices connecting to a point-to-point channel, set the same ErrorRate value for both in your program.

PDR is the ratio of successfully delivered packets at the receiver to the total transmitted packets by the sender. In this task, you need to measure the PDR of the network by summing up the total no. of packets received at UdpEchoServer and UdpEchoClient (say R_Total) and summing up the total no. of packets transmitted by UdpEchoClient and UdpEchoServer (say T_Total). Then PDR=R_Total/T_Total

[5 Points]

Fill in the table below with PDR by transmitting at least 100 packets in each run of your code

RngRun/Error Rate	**ER1=0.2**	**ER2=0.4**	**ER3=0.6**	**ER4=0.8**	**ER5=1**
RngRun1=					
RngRun2=					
...					

RngRun10=					
AVG. PDR					

Tasks to be done:

- Plot variation in average PDR (on the Y-axis) by varying ErrorRate on the scale of 0 to 1 (with a step value of 0.2 on the X-axis) for Rate Error Model by using tools like Gnuplot. Generate such tables for each of the 3 error units [Packet, Bit, Byte], so 3 tables and corresponding plots become one of the deliverables (You have to plot avg. PDR Vs ER by using values from the above table)... **[9 Points]**
- You need to write a script that gets the data and populates the tables for the 10 seed values, and you are free to use any language for automation, bash/python, but attach the script used. **[3 points]**
- **Make sure to automate all your simulation tasks and attach the scripts used.**
- How to select RngRun values for the above table?
    - RngRun1 = "Last TWO DIGITS of your ROLL NUMBER"
    - RngRun2 = RngRun1+1
    - ….
    - RngRun10=RngRun1+9

**** ns-3 version should be >=3.36**
**** Reference File Paths:**

- ns-allinone-3.36.1/ns-3.36.1/examples/error-model/simple-error-model.cc
- ns-allinone-3.36.1/ns-3.36.1/examples/tutorial/fifth.cc

Deliverables in a tar ball on GC:

- **Submission Guidelines**: For each part, create a separate folder. Upload all the folders and Assignment Report in GC as a tarball with the file name as <your roll no>_<your name>.tar
- Readable Report [**3 Marks for report quality**] enumerating steps followed with screenshots for each of the important steps.
    - Put the screenshots in the report for better clarity
    - Your report should have screenshots of your final results **necessarily,** and you are expected to **explain** in the report which function was used or which line of code set the data rate or a TCP flow or a UDP flow or how you fetch the address of a

particular node, so everything needs to be carefully explained in your report.

- Explain everything that has been asked in your report and highlight the important parts, and submit every codefile, script or anything that you used to complete the tasks.
- For all the plots, write the inferences that you have observed.

Check Web sources for additional information