Lab Exercise 6: Routing, Throughput and IP Fragmentation

Objectives:

- gain insights into routing dynamics and IP fragmentation
- Set up simulation in NS2 for TCP throughput measurement

Prerequisites and Links:

- Week 7 and 8 Lectures
- Relevant Parts of Chapter 4 and Chapter 5 of the textbook
- Introduction to Tools of the Trade
- Basic understanding of Linux. A good resource is here but there are several other resources online.
- Introduction to ns-2 from Labs 4 and 5.
- tp_routing.tcl
- exercise2.tcl
- TCPThroughput.png
- ip_frag

Marks: 10 marks.

- We expect the students to go through as much of the lab exercises as they can at home and come to the lab for clarifying any doubts in procedure/specifications.

Deadline:

23:59:59 Sunday 14th April. You can submit as many times as you wish before the deadline. A later submission will override the earlier submission, so make sure you submit the correct file. Do not leave until the last moment to submit, as there may be technical, or communications error and you will not have time to rectify it.

Late Report Submission Penalty:

Late penalty will be applied as follows:

- 1 day after deadline: 20% reduction
- 2 days after deadline: 40% reduction
- 3 days after deadline: 60% reduction
- 4 or more days late: NOT accepted

Note that the above penalty is applied to your final mark in report. For example, if you submit your lab work report 2 days late and your score on the lab report is 4, then your final mark will be 4 - 1.6 (40% penalty) = 2.4.

Submission Instructions:

Submit a PDF document **Lab6.pdf** with answers to all questions for all exercises. To include all supporting files, create a tar archive of all the files called **Lab6.tar.** Submit the archive using give. You can submit from a lab machine or ssh into the CSE login server.

Original Work Only:

You are strongly encouraged to discuss the questions with other students in your lab. However, each student must submit his or her own work. You may need to refer to the material indicated above (particularly Tools of the Trade document) and also conduct your own research to answer the questions.

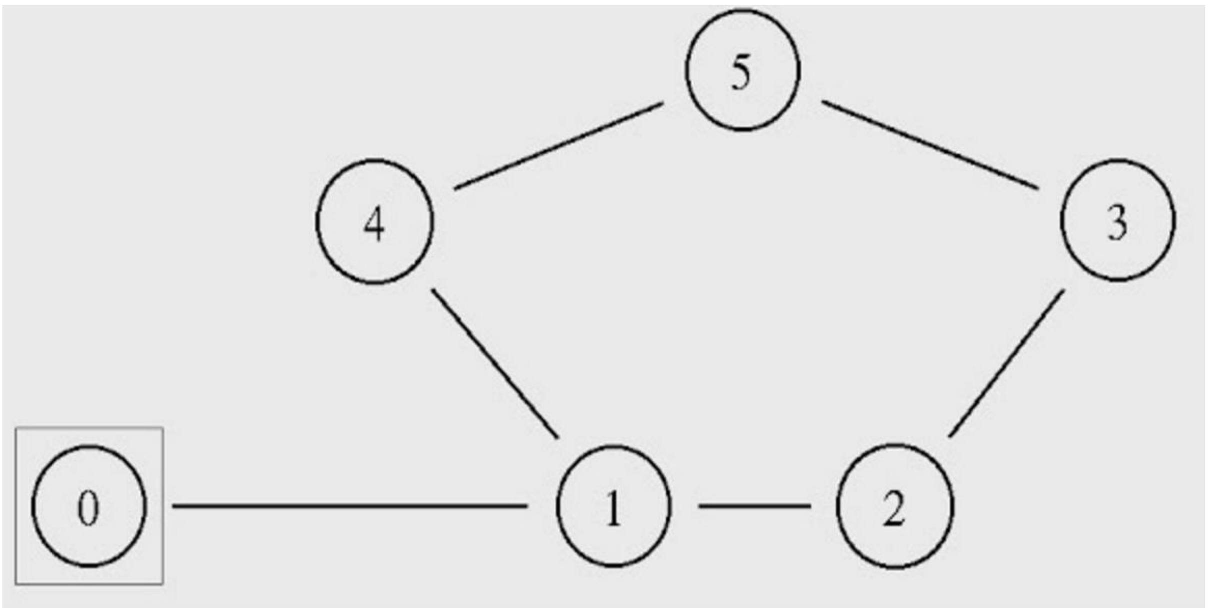
OS Compatibility:

<o:p></o:p> The provided script (for ns-2) have been tested on CSE Linux machines. They may not work on your personal machine even if you have installed ns-2. As such, we suggest that you work on a CSE machine to complete these lab exercises. You can do so by going to a lab in person or via ssh/vlab.

Exercise 1: Understanding the Impact of Network Dynamics on Routing

In this exercise, we will observe how routing protocols react when network conditions change (e.g., a network link fails) using a ns-2 simulation.

The provided script, tp_routing.tcl takes no arguments and generates the network topology shown in the figure below.



You can run the simulation with the following command:

```
$ns tp_routing.tcl
```

Step 1: Run the script and observe the NAM window output.

Question 1. Which nodes communicate with which other nodes? Which route do the packets follow? Does it change over time?

Note: You can also answer the above question by examining the simulation setting in the script file.

Step 2: Modify the script by uncommenting the following two lines (line No 84 and 85):

```
$ns rtmodel-at 1.0 down $n1 $n4

$ns rtmodel-at 1.2 up $n1 $n4
```

Step 3: Rerun the simulation and observe the NAM window output.

NOTE: Ignore the NAM syntax warnings on the terminal. These will not affect the simulation.

Question 2: What happens at time 1.0 and at time 1.2? Does the route between the communicating nodes change as a result of that?

Step 4: The nodes in the simulation above use a static routing protocol (i.e., preferred routes do not change over time). We are going to change that, so that they use a Distance-Vector routing protocol. Modify the script and uncomment the following line (Line No 16) before the definition of the `finish` procedure.

```
$ns rtproto DV
```

Step 5: Rerun the simulation and observe the NAM window output.

Question 3: Did you observe any additional traffic as compared to Step 3 above? How does the network react to the changes that take place at time 1.0 and time 1.2 now?

Step 6: Comment the two lines (Lines 84 and 85) that you had added to the script in Step 2 and uncomment the following line (Line 87) instead:

```
$ns cost $n1 $n4 3
```

Step 7: Rerun the simulation and observe the NAM window output.

Question 4: How does this change affect the routing? Explain why.

Step 8: Comment line 87 and Uncomment the following lines (Lines 89 and 90):

```
$ns cost $n1 $n4 2

$ns cost $n3 $n5 3
```

and uncomment the following (Line 29), which is located right after the finish procedure definition:

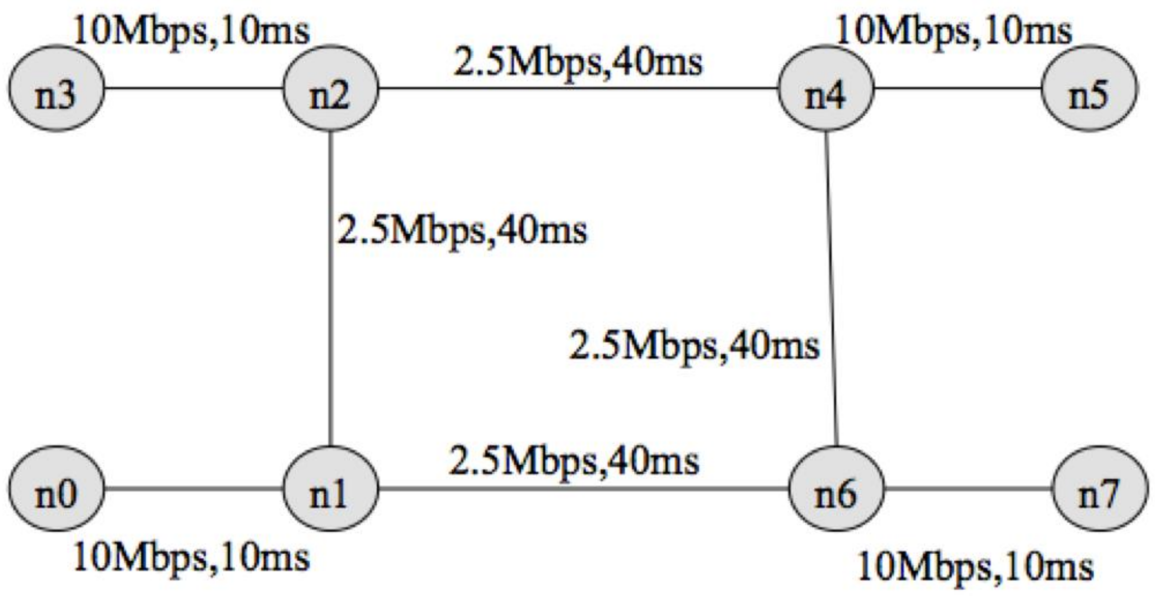
```
Node set multiPath_ 1
```

Step 9: Rerun the simulation and observe th e NAM window output.

Question 5: Describe what happens and deduce the effect of the line you just uncommented.

Exercise 2: Setting up NS2 simulation for measuring TCP throughput

Consider the topology shown in the following figure where bandwidth and delay for each link is shown.



You have been provided with a stub tcl file exercise2.tcl . Your task is to complete the stub file so that it runs with ns and produces two trace files tcp1.tr and tcp2.tr and nam.out. Check the animation for the simulation using nam.out file. Next write a script named "throughput.plot" (referenced from within exercise2.tcl in procedure finish( )) to plot the throughput received by host n5 for two flows terminating at n5. Uncomment the line (#exec gnuplot throughput.plot &) to execute gnuplot. You have been provided with the throughput plot TCPThroughput.png produced by gnuplot for comparing your final output.

">>" in the stub file indicates that one (or more) lines need to be added. Remove the ">>" and insert the required code.

Consider the following traffic pattern for your simulation.

FTP/TCP Source n0 -> TCP Sink n5 : start time: 0.5 sec End time: 8.5 sec

FTP/TCP Source n3 -> TCP Sink n5 : start time: 2.0 sec End time: 9.5 sec

FTP/TCP Source n7 -> TCP Sink n0 : start time: 3.0 sec End time: 9.5 sec

FTP/TCP Source n7 -> TCP Sink n3 : start time: 4.0 sec End time: 7.0 sec

You have to submit your completed tcl file (exercise2.tcl) and the script (throughput.plot) for producing the throughput plot. Please also answer the following questions:

Question 1) Why the throughput achieved by flow tcp2 is higher than tcp1 between time span 6 sec to 8 sec?

Question 2) Why the throughput for flow tcp1 is fluctuating between time span 0.5 sec to 2 sec?
Question 3) Why is the maximum throughput achieved by any one flow capped at around 1.5Mbps?

Exercise 3: Understanding IP Fragmentation

We will try to find out what happens when IP fragments a datagram by increasing the size of a datagram until fragmentation occurs. You are provided with a Wireshark trace file ip_frag that contains trace of sending pings with specific payloads to 8.8.8.8. We have used ping with option (– s option on Linux) to set the size of data to be carried in the ICMP echo request message. Note that the default packet size is 64 bytes in Linux (56 bytes data + 8 bytes ICMP header). Also note that Linux implementation for ping also uses 8 bytes of ICMP time stamp option leaving 48 bytes for the user data in the default mode. Once you have send a series of packets with the increasing data sizes, IP will start fragmenting packets that it cannot handle. We have used the following commands to generate this trace file.

Step 1: Ping with default packet size to the target destination as 8.8.8.8

```
ping -c 10 8.8.8.8
```

Step 2: Repeat by sending a set of ICMP requests with data of 2000.

```
ping -s 2000 -c 10 8.8.8.8
```

Step 3: Repeat again with data size set as 3500

```
ping -s 3500 -c 10 8.8.8.8
```

Load this trace file in Wireshark, filter on protocol field ICMP (you may need to clear the filter to see the fragments) and answer the following questions.

Question 1: Which data size has caused fragmentation and why? Which host/router has fragmented the original datagram? How many fragments have been created when data size is specified as 2000?

Question 2: Did the reply from the destination 8.8.8.8. for 3500-byte data size also get fragmented? Why and why not?

Question 3: Give the ID, length, flag and offset values for all the fragments of the first packet sent by 192.168.1.103 with data size of 3500 bytes?

Question 4: Has fragmentation of fragments occurred when data of size 3500 bytes has been used? Why and why not?

Question 5: What will happen if for our example one fragment of the original datagram from 192.168.1.103 is lost?