ECE 545 Project 2: In-Depth Investigation of TCP via NS3 Simulation

Objective: The objective of this project is to help you learn the network simulator ns3, and help you to gain the in-depth understanding of TCP through simulation.

Instructions:

- You need to submit a project report, along with your ns3 Tcl scripts for your experiments and the generated trace files and graphs. **We should be able to execute your codes in our machines.**
- You need to type your project report using some editing software, and submit it as a PDF file. Your project should be written with good readability, clearly describing the problem, your codes for the simulation, the generated trace files and the associated graphs, and your analysis.
- Electronically submit your project report and NS3 codes **in one zipped package, titled as "ID_Lastname_prjt2.zip",** through the blackboard system.
- Your report is **due at 23:59 pm, April 30, 2019, Chicago time. LATE submission will not be graded and lead to a "0" grade.**
- Each on-site student will be allocated a time slot during the week **Apr. 24 – Apr. 28 to demonstrate your ns3 codes.** Validity of the ns3 codes from remote students will be checked by TA.
- **In cases that "copy" issues are detected, the involved groups will be reported to the department.**

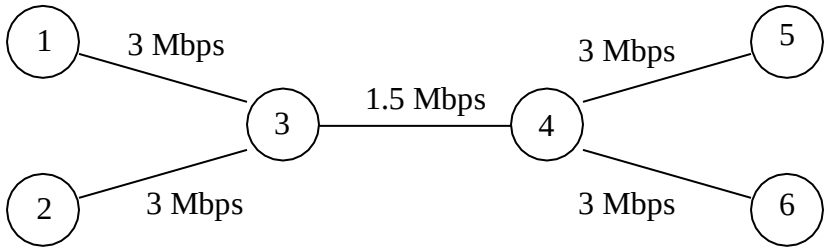
Grade (100 points)

- **Implementation 50%, Graphs: 20%, Analysis: 30%**

Description: NS3 is a widely used discrete event simulator for network analysis. In this project, you will study the implementation of the TCP protocol using ns3 simulation. You will study the various flavors of TCP and how the behavior of TCP changes according to the background traffic. You may need to find and read related RFCs or other references to learn the implementation details of the TCP protocols (i.e., Reno, Tahoe, Westwood) considered in this project.

Consider a network with 6 nodes and 5 links as shown below. The link capacities are given in the figure, where the **link 3-4 is the bottleneck** of the topology. All links have a propagation delay of 10 ms and a **drop-tail queue**. **The following specifications apply to all the experiments:**

- In the experiments, start with default values for all the variables, and then do necessary modifications according the problem specification.
- Each TCP connection is created with an **FTP** flow running on top of it, transferring data from the source node to the destination node.
- When the flavor of TCP is not mentioned, assume **TCP Reno**.
- All your results and observations should be supported by your simulation script and trace files. Generating graphs when necessary or required to better demonstrate your observations.



Part 1: Efficiency of the TCP congestion control, flow control, and reliable data transfer protocols

In the first part, create a single TCP connection between node-1 (source) and node-5 (destination), and no other background traffic is considered. Do experiments to observe the operations of TCP.

1. Use the default parameters, and run your simulation from 0 to 10 seconds. Analyze your simulation results and answer the following questions. What is the total number of segments and the number of bytes successfully transmitted during the 10 seconds? What is the average throughput achieved? How does this compare with the bottleneck bandwidth in your topology? Then, change the queue size used at node 3 to a value much smaller than the default value in one case, and to a value much larger than the default value in the other case. Run simulations to obtain the average throughput in these two cases, respectively. Do you get different average throughputs compared to that under the default queue size? Explain your observations?

2. Set the queue size in the bottleneck link to a proper finite value, so that retransmission and congestion control due to packet dropping and timeout can be observed. Set the receive window large enough to bypass the flow control. You need to write appropriate codes to trace the congestion window, RTT, EstimatedRTT and TimeoutInterval. Run your simulation for long enough time, so that you can capture some packet loss events and timeout events; using multiple simulation runs to capture such events is also fine.
   - Plot the congestion window (cwnd) as a function of time. Use the graphs generated from one or multiple simulation runs to demonstrate the options of slow start, congestion avoidance, the reaction to triple duplicate ACK, and the reaction to timeout.
   - Plot the RTT and EstimatedRTT as a function of time according to one of your trace files, to obtain a similar graph to Fig. 3.32 in our textbook.
   - Use necessary trace data or graphs to illustrate the option of "doubling the timeout interval" upon the retransmission due to timeout events.
   - Run simulations to investigate the relationship between the TCP throughput and the packet loss rate. In this example, set the capacity for all the 5 links to a large value (50 Mbps or larger), i.e., link 3-4 is not a bottleneck link in this experiment. Also use a large queue size at each node to avoid packet drop due to queuing. You need to modify the loss rate at the link 3-4 by using the **rate error model** (refer to https://www.nsnam.org/docs/models/html/error-model.html and https://www.nsnam.org/doxygen/classns3_1_1_rate_error_model.html for implementation details). Run simulations long enough with the loss rates of 10^{-4}, 10^{-3}, 10^{-2}, and 10^{-1} at link 3-4, and calculate the throughput for each scenario, respectively. Plot the TCP throughput vs. packet loss curve. In addition, please calculate and plot the throughput curve according to the equation $averagethroughput = \frac{1.22 \bullet MSS}{RTT\sqrt{L}}$. Compare the throughput curve from the mathematical analysis to the throughput curve obtained from the simulation. Do the two curves have the similar shape? Are they close enough? What is the possible reason if the analytical curve deviates from the simulation results in some degree?

3. **Go back to the original link-capacity configuration as given in the figure on page-1, and use it in all the following experiments (including Part 2).** Adjust the value of the receive window and other related parameters if necessary to create a scenario for demonstrating the effect of **flow control**. Use necessary trace data or graph to illustrate your observations of flow control option.

Part 2: Resource sharing under the transport layer protocol

In the second part, create two transport-layer flows. Flow-1 is between node-1 (source) and node-5 (destination). Flow-2 is between node-2 (source) and node-6 (destination). Do experiments to observe the resource sharing between these two flows.

1. TCP + UDP: Flow-1 is TCP and flow-2 is UDP attached with a **CBR source**. Run the experiment multiple times by modifying the traffic generation rate of the UDP flow, and observe the throughput achieved by the two flows. Plot a graph showing the UDP throughput on the x-axis and TCP throughput on the y-axis. Determine the UDP throughput for which the two flows achieve a fair share of the link. Fix the UDP rate at the fair-share rate you just found, and run simulations to calculate the loss rate of the TCP connection. Replace the drop-tail queue with another queuing scheme provided by ns3 and run simulation again. Do you get different throughput under a new queuing scheme? If yes, do some extra reading to find possible reasons leading to the TCP throughput increase or decrease under a certain queuing scheme.

2. TCP + UDP: Flow-1 is TCP and flow-2 is UDP attached with an **exponential on-off type source**. Set large average "on" and "off" times for the UDP traffic generator. Run simulations with long enough time, and plot on one graph the throughput of both the TCP and the UDP flows vs. time. Explain how TCP elastically adjust its rate to fit the available bandwidth.

3. TCP + TCP: Both flows are TCP, with a maximum window size (MWS) of 30 packets for flow-1 and 6 packets for flow-2, respectively. Plot, in one graph, the throughput of both flows vs. time. Do the flows get a fair share of the link? Explain.

4. TCP + TCP: Both flows are TCP, with a packet size of 1000 bytes for flow-1 and 500 bytes for flow-2, respectively. Choose the same MWS for the two flows. Plot, in one graph, the throughput of both flows vs. time. Do the flows get a fair share of the link? Explain.

5. TCP Tahoe + TCP Reno: Read more about TCP Tahoe. Flow-1 is TCP Tahoe, and TCP-2 is TCP Reno. Choose the same MWS for the two flows. Plot the throughput achieved by the two flows, flow-1 on x-axis and flow-2 on y-axis. Explain your observations from the graph.

6. TCP Tahoe + TCP Westwood: Read about TCP Westwood. Flow-1 is TCP Tahoe, and TCP-2 is TCP Westwood. Choose the same MWS for the two flows. Plot the throughput achieved by the two flows, flow-1 on x-axis and flow-2 on y-axis. Explain your observations from the graph.

7. TCP Reno + TCP Westwood: Flow-1 is TCP Reno, and TCP-2 is TCP Westwood. Choose the same MWS for the two flows. Plot the throughput achieved by the two flows, flow-1 on x-axis and flow-2 on y-axis. Explain your observations from the graph.

8. Compare the last three graphs and explain.

Tips: When plot flow throughput on both axes, you should use **instantaneous throughput** taken at a sequence of timestamps. For example, denote $x_{t=1}$ and $y_{t=1}$ as the instantaneous throughput of two flows at t = 1sec, respectively. Then the graph is generated by plotting dots: $(x_{t=1}, y_{t=1})$, $(x_{t=2}, y_{t=2})$, … . Instantaneous throughput at t is obtained by calculating the throughput in a narrow window around t. For example, $x_{t=1}$ can be obtained by calculating the throughput from 0.9sec to 1sec (with a window size of 100ms).