Lab6: Comparing TCP variants + Wireshark

*This lab may be done in groups of size at most 3 persons. Use the **C language** for your socket programs. You can use available code on the Internet for parts of your socket programs, but you must mention the source in comments within the file.*

In this lab we will compare different TCP variants in terms of their throughput. The setup is shown in the Figure below.

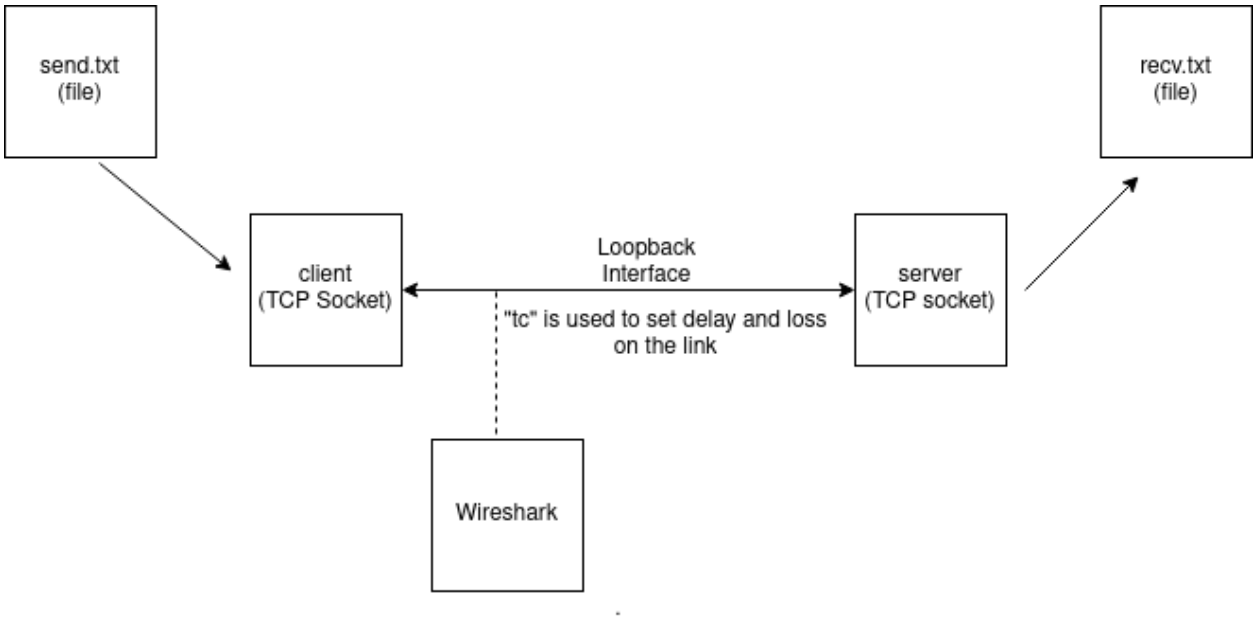

Figure: Lab 6 Setup

You only need a single machine to do this lab. The client transfers a file (send.txt) over the loopback interface to the server. The server writes the received data to a new file (recv.txt). TCP is used for the transfer. We will use different variants of TCP available on your machine. The delay and packet loss on the loopback interface will be set using “tc” similar to Lab3. Wireshark will be used to capture packets on the loopback interface and to study the variation on TCP Window etc.

Preliminaries

1. Install Wireshark (https://www.wireshark.org/download.html). To run wireshark on Ubuntu, type the command “sudo wireshark” in a terminal window. It captures packets on a specified interface (you will have to choose the loopback interface for the lab).
2. Study how Wireshark can be used to study “bytes_in_flight” (unACKed bytes) of a TCP flow: (https://www.youtube.com/watch?v=9lJ0vsA40is)
3. Check which TCP protocols are available on your machine:

cat /proc/sys/net/ipv4/tcp_available_congestion_control

4. Check which is the default TCP variant:
   cat /proc/sys/net/ipv4/tcp_congestion_control
   Note: you are likely to find Reno and Cubic on the latest Ubuntu OS. Do the experiment on a machine which has more than one TCP variant.
5. How to setup a TCP connection between two sockets on the same machine:
   Peterson & Davie Textbook Example:
   https://book.systemsapproach.org/foundation/software.html?highlight=sockets#example-application
   Beej's guide: https://beej.us/guide/bgnet/
6. Trick to create a text file of any length: you can use the "yes" command on Ubuntu.
7. Lookup examples online on how to read a text file and write to a TCP socket (required at the client), and also how to read data from a TCP socket and write this data to an output file (required at the server).
   Example:
   https://stackoverflow.com/questions/2014033/send-and-receive-a-file-in-socket-programming-in-linux-with-c-c-gcc-g
8. How to set which TCP variant the socket should use:
   https://stackoverflow.com/questions/59265004/how-to-change-tcp-congestion-control-algorithm-using-setsockopt-call-from-c
9. How to set delay and loss on the loopback interface using "tc":
   https://netbeez.net/blog/how-to-use-the-linux-traffic-control/
   https://wiki.linuxfoundation.org/networking/netem
   Note: you will have to use the loopback interface in the command as in Lab3. You may have to "sudo" the command.
10. Study how to use "gettimeofday" to read the local clock. You will need this to find the time taken to transfer a file, which will be used to compute throughput.
11. To automate the large number of experimental runs you have to do, it is always a good idea to use a script. You should learn about scripting. Example:
    https://ma.ttias.be/bash-loop-first-step-automation-linux/
12. Setting Maximum Transmission Unit (MTU) on the loopback interface to 1500B. The MTU may be much larger on the loopback interface than on say an Ethernet interface. To mimic the Internet, reduce the MTU to 1500B.
    To find out the MTU on all interfaces: netstat -i
    To set the MTU: sudo ifconfig <loopback interface> mtu 1500
    Run "netstat -i" to check that the MTU has been updated.
13. Confidence intervals for graphs:
    https://www.mathsisfun.com/data/confidence-interval.html (see Step 2 on this page)

EXPERIMENTS

In all experiments, you will transfer a text file (send.txt) of size 5MB from client to server. Recall that the server writes the data received to recv.txt. You should check if indeed send.txt and recv.txt are the same, to ensure that the code is working properly. For each experiment below, calculate the mean and standard deviation of the obtained throughput [throughput (in bits/sec) = fileSize (in bits)/transfer time (secs)) for 20 runs. Write scripts to automate the process. You should not have to manually run the experiment 20 times. Also, the calculated throughput information etc. should be automatically appended to a file each time you run. You should not do this manually.

Each experiment uses a combination of Delay, Loss (set using tc) and TCP variant. We will have a total of 18 experiments (3x3x2). Note that each of these must be run 20 times. The different choices for Delay, Loss, and TCP variant are given below.

Delay=10ms, 50ms, 100ms

Loss=0.1%, 0.5%, 1%

TCP variant: Reno, Cubic (if Cubic is not installed, you can use any other TCP variant besides Reno)

In your report give the results in the form of plots and comment on them. Draw error bars of 90% confidence intervals around the mean. Feel free to use any graphic tool (The online tool Desmos allows plots of error bars).

Plot 1: (Loss=0.1%) Throughput vs. Delay for both Reno and Cubic

Plot 2: (Loss=0.5%) Throughput vs. Delay for both Reno and Cubic

Plot 3: (Loss=1%) Throughput vs. Delay for both Reno and Cubic

Plot 4: (Delay=10ms) Throughput vs. Loss for both Reno and Cubic

Plot 5: (Delay = 50ms) Throughput vs. Loss for both Reno and Cubic

Plot 6: (Delay = 100ms) Throughput vs. Loss for both Reno and Cubic

Cubic is supposed to be more aggressive (has faster window increase during congestion avoidance) than Reno. **In your report** comment on whether the observed throughputs confirm this or not. Comment on how throughput changes with increasing Delay. Comment on how it increases with increasing packet loss.

In your report show the Window Scaling graphs (which shows bytes_in_flight) for both TCP variants from Wireshark for one run when Delay = 10ms, Loss=0.1%. Do the same for the case Delay = 100ms, Loss=1%. Annotate the plots for Reno to show at least one region where the TCP was in Slow Start and at least one region where TCP was in Congestion avoidance. Mark any obvious locations of packet loss on the graphs. Do the same for the plots of the Cubic variant.

Per group, upload only one tar file to Moodle. The file name should have roll numbers of all group members. The file should contain:

1. (5 marks) README file saying how to compile code and how to run the experiments.

2. (10 marks) Scripts to automate running of experiments (the more automated, the better. For example, if you give TCP variant as command line input to the client program, set delay/loss in the script itself, then expect more marks)

3. (20 marks) Well commented code for client and server (marks will be given for comments)

4. (20 marks) Report