Lab Assignment 6
(Due on 04/01/2013)

Objective. This lab assignment is designed to develop a deeper understanding on congestion control and flow control.

Team. You can work in a team of no more than three students. Only one copy of the code needs to be submitted to the online grading system. The IDs and names of all team members need to be provided to the system during the code submission.

Background. JDK installation in DETERlab. By default, the OS images on DETERlab do not provide JDK. To install JDK on the experiment nodes, use the following command:

```
sudo apt-get install openjdk-7-jdk
```

To run your experiment, you may need to install JDK on both the server and the client nodes.

Assignment.

Experiment I. (6 pts) Measure the throughput of a TCP connection between nodeA and nodeC over the topology given in Lab6-Exp1.tcl using `iperf` (please refer to Lab1 instructions for installation/usage of `iperf`). Change the bandwidth of the duplex link to 50Mbps and 100Mbps. Perform the measurement again. Record your measurement results in the online submission system. Do you think TCP fully exploits the expanded link capacity? Now change the delay of the link to 50msec and 500msec. Perform the measurement again, record your measurement results in the online submission system. What is the impact of link delay on the TCP performance? Explain what you have observed.
Experiment II. (4 pts) Load topology Lab6-Exp2.tcl. Setup two simultaneous TCP connections between nodeA and nodeC, the other between nodeB and nodeD (the two connections transfer at roughly the same time). Now measure the throughput of each TCP connection respectively, and record their throughputs.

Experiment III. (4 pts) Now setup one TCP connection between nodeA and nodeC and one UDP flow sent from nodeB to nodeD on Lab6-Exp2.tcl. The two connections transfer roughly at the same time. Change the UDP traffic rate to the following values: 100Kbps, 500Kbps, 5Mbps, 10Mbps. Measure the throughput of TCP connection in each scenario. Record and compare the throughput of TCP and UDP (UDP’s throughput may be different from its sending rate, as packets may get lost).

Now revise or tune the program that you’ve developed in Lab 5 which performs reliable data transfer (uni-directional) using UDP socket and measure its performance under the following scenarios.

Experiment IV. (16 pts + bonus pts) Run your code on the same topology as in Experiment I (Lab6-Exp1.tcl). Measure the throughput of your reliable data delivery protocol, log your measurement results in a file named Exp4-thoughput.log in plaintext (e.g., throughput: 100Kbps) and report it on the online grading system. Compare it with the TCP throughput you measured in Experiment I. Your will get 16 pts, if the performance of your protocol is comparable to TCP performance. Here is the algorithm for grading.
Ratio = (throughput of your RDT)/(throughput of TCP)
Ratio<0.7: Grade = (Ratio/0.7)* 16 pts;
0.7<=Ratio<=1: Grade = 16 pts;
Ratio>1: Grade = 16 + (Ratio-1)*10;

**Experiment V.** (10 pts + bonus pts) On the topology as in Lab6-Exp2.tcl, setup two simultaneous connections of your RDT implementation between nodeA and nodeC, the other between nodeB and nodeD (the two connections transfer at the same time). Now measure the throughput of each connection and record their throughputs. Log your measurement results in a file named Exp5-thoughput.log in plaintext (e.g., A-C throughput 100Kbps; B-D throughput 200Kbps). Are they similar to each other (i.e., fairness) (5pts)? How does the throughput compare with the TCP(5pts)? Here is the algorithm for grading.

Fairness =
(throughput of connection1 + throughput of connection2)^2/
2*(throughput of connection1^2 + throughput of connection^2)

Fairness <0.7: Grade = (Fairness /0.7)* 5;
0.7<= Fairness <=1: Grade =5;

avg_throughput = (throughput of connection1+ throughput of
connection2)/2

Ratio = (avg_throughput of your RDT)/(avg_throughput of TCP)
Ratio<0.7: Grade = (Ratio/0.7)* 5;
0.7<=Ratio<=1: Grade = 5;
Ratio>1: Grade = 5 + (Ratio-1)*10;

**Experiment VI.** (10 pts + bonus pts) Now setup two simultaneous connections – one is your RDT implementation between nodeA and nodeC, the other is TCP between nodeB and nodeD. Now measure the throughput of each connection and record their throughputs in a file named Exp6-thoughput.log in plaintext (e.g., RDT throughput 100Kbps; TCP throughput 200Kbps). Are they similar to each other (i.e., TCP friendliness) (5pts)?

Ratio = (avg_throughput of your RDT)/(avg_throughput of TCP)
Ratio<0.7: Grade = (Ratio/0.7)* 5;
0.7<=Ratio<=1: Grade = 5;
Ratio>1: Grade = 5 + (Ratio-1)*10;

Run the experiment again with your RDT protocol and the UDP protocol with a sending rate of 5Mbps. Compare the throughput of your RDT protocol with the TCP throughput you measured in Experiment III.

Ratio = (avg_throughput of your RDT)/(avg_throughput of TCP)
Ratio<0.7: Grade = (Ratio/0.7)* 5;
0.7<=Ratio<=1: Grade = 5;
Ratio>1: Grade = 5 + (Ratio-1)*10;

Submission.

Please create a new directory with name “Lab6” under your home directory in DETERlab, and put all the associated files with this assignment under ~/Lab6/.

Once you are happy with the code and the experiment result, please go to the online grading system, provide necessary information to the system as requested, so that your code will be automatically compiled and executed, and the results be automatically logged. (the logged information will be used for manual validation). Specifically, this is what the online grade system will do:

Make sure your Experiment setup is swapped in and JDK is installed, when you request the online grading system to grade. The online grade system will:

1) Run “make clean” and “rm bigfile_RDTreceived.txt” to clean up the executable code and the previous experiment result;
2) Run “make” to compile your code (so make sure your files are named correctly);
3) Run your protocol implementation to get all the log files;

Manual validation will be performed to check whether the logged information is consistent with your self-report information in the online grading system. Your code will also be validated manually to check whether logged information accurately reflects the performance of your protocol.