CSE 473 – Introduction to Computer Networks

Lab 4

John DeHart *Due 11/9/2015*

Review and follow the general instructions included in the Lab 1 writeup.

In this lab, you will be implementing a reliable data transfer protocol and testing it in ONL. The protocol is a sliding window protocol, similar to those discussed in Kurose and Ross, and includes the selective repeat feature. You will be provided with the basic framework for the protocol, but you will need to implement the mechanisms for detecting lost packets and retransmitting them, as needed. Unlike the descriptions in Kurose and Ross, your protocol will support data transfer in both directions. That is, both endpoints can operate as both "senders" and "receivers".

The provided framework is structured in three layers, an application layer, called *SrcSnk*, a transport layer called *Rdt* that implements the reliable data transfer protocol, and a lower layer, called the *Substrate* which deals with the socket communication. These three layers are each implemented as a separate java modules, each with their own threads, and they communicate with each other through queues. You will be provided with the *SrcSnk* and *Substrate* modules, as well as a skeleton of the *Rdt* module. There is also a "main" module called *TestRdt* that handles command-line arguments and launches the others. Finally, there is a *Packet* module that provides methods for accessing and setting packet fields and packing the packet fields into a buffer that can be sent over the network. You will find additional details in the provided source code, which is located in the *lab4* folder in your repository.

To complete the *Rdt* module, you will need to add a *send buffer* that stores packets that have been sent, but not yet acknowledged. The send buffer can be implemented as an array, with the index of each entry being its sequence number. In addition, you will need a second array (also indexed by sequence number) containing the *resend times* of the packets in the resend buffer. You will also find it useful to include a *resend list* containing the sequence numbers of the packets that have not yet been acknowledged, listed in the order of their resend times. The resend list can be implemented efficiently using a *LinkedList* data structure. You will also need to add a *receive buffer* for packets whose payloads have not yet been passed up to the application layer. Because packets can be lost and the payloads must be delivered in order, the receive buffer may need to hold a number of packets equal to the window size. You can implement the receive buffer as an array, indexed by sequence numbers.

The lab includes completing the *Rdt* module, demonstrating that it works correctly and then performing a series of experiments to get a better understanding of its performance characteristics. Detailed instructions are contained in the lab report template.

As on the previous lab, you have the option to work with a lab partner. You need not work with the same partner as before, but if you do decide to switch partners, make sure you let your

former partner know of your intention ahead of time, so that they can make alternate plans. Also send me email to let me know of the change.

If you are going to work with the same partner as you did on Lab3 then you do not need to let me know.

IF YOU DECIDE TO WORK ON THIS LAB WITH A **NEW** PARTNER, YOU NEED TO RECORD YOUR TEAM BY NOON WEDNESDAY 10/28/2015. SEND ME AN EMAIL (WITH COPY TO YOUR PARTNER) WITH THE NAMES AND STUDENT IDs OF THE TWO PARTNERS.