**CS4283/5283: Computer Networks**

Vanderbilt University, Fall 2022; Instructor: Aniruddha Gokhale

**Programming Assignment #3**

Handed out 10/06/2022; Due 10/27/2022 (11:59 pm Central in Brightspace)

# **Assignment Theme:** Homegrown Reliable Transport Protocol

**Overview**

This assignment will build on our previous assignments. However, instead of relying on TCP providing reliability end-to-end, we will implement our own homegrown solution for reliable transfer. We are not doing any flow or congestion control in this assignment. For the reliable transport, we can configure our transport layer to adopt one of three configured reliability policies that can be specified in the config.ini file. For example, we could have one of the following entries in our config.ini under the [Transport] category. Accordingly, the transport protocol will implement the corresponding strategy.

[Transport]

AlternatingBit

GoBackN

SelectiveRepeat

**Expected Operation: What remains same as before/What changes?**

In this assignment, we will need a large application-level packet so that it can be divided into many chunks. Unfortunately, our grocery order or health status is a very small packet. Thus, I suggest that after the packet is serialized, we pad the serialized buffer with random number of bytes to make a large application-level packet, say 1 megabyte large. Do this only for the requests (no need for the response).

Padded bytes

Serialized packet

Now suppose we decide to impose a Maximum Transfer Unit (MTU) t 16 bytes, which restricts the subset of the original packet that can be sent at one time. Thus, for a 1MB packet, there will be a total of 64 such 16-byte chunks in a sequence that will make a complete application-level packet. Remember tat these chunks must appear in the same order in which they were sent. For the GoBackN and SelectiveRepeat, say we set the size of our sliding window to 8 chunks at a time. For alternating bit, the size of the window is always 1 chunk.

Summary of specifications:

* Application request packet size = 1024 bytes
* Max chunk size = 16 bytes
* Sliding window size = 8 chunks (for GoBackN and SelectiveRepeat) or 1 chunk for Alternating Bit.

Integrate the layered skeleton code that we have with the Dealer-Router scaffolding code that was provided to showcase the hop-by-hop communication.

**Where/How to start working on the assignment?**

We will keep the same three network topology scenarios as in Assignment #2 to evaluate our solution. In this assignment, we will not let Mininet cause loss or delay of packets but rather we do it ourselves as follows:

In the network layer of our skeleton code, when the REQ socket on the client or the DEALER socket in each intermediate router is ready to forward the chunk to the next hop, using random number logic, either (a) send the chunk to the next hop, or (b) delay it, or (c) just don’t send it at all and drop it.

Naturally, we will need a timer in the transport layer for the end host only (sending client). For selective repeat, we will need buffering on receiving server.

**Milestones (and hence the deliverables and expectations):**

Please use the Slack Channel for programming assignments for discussions/questions.

***Milestone 1:***

* Get the AlternatingBit and GoBackN strategies working
* Submit to Brightspace what all have you accomplished up to this point and what was not accomplished, and difficulties, if any.

***Milestone 2 (final):***

* Get the SelectiveRepeat strategy to work
* Compare the performance of all three strategies on the three evaluation scenarios
* Plot graphs
* Create video for the grader
* Submit the code, plots etc to Brightspace

**Mininet topologies to be used**

We will use exactly the same topologies as in Assignment #2.

**Rubrics (for grading after the final milestone)**

* Correctness (program works): 40%
* Experiments in all scenarios 20%
* Plots of results 20%
* README file explaining how to run the code: 10%
* Zoom-demo or self-explanatory video to grader: 10%