CS 421 – Computer Networks
Programming Assignment II

Implementing a Reliable Transfer Protocol over UDP

Fall 2017

Due: Dec. 18, 2017 at 11:59PM

In this programming assignment, you are asked to write the Java code for simulating the operation of a Go-Back-N (GBN) sender over a lossy network layer and the code for transferring a specified file as an application-layer program. The GBN receiver entity is supplied to you as a Java program for your convenience. The GBN protocol will be implemented over the User Datagram Protocol (UDP), which on its own does not provide a reliable service to applications.

The GBN Sender: This is the part you are asked to implement. Basically, at the application layer, you will send a specified file that resides on the disk. You can get this file of size 10 MBytes from http://wlab.cs.bilkent.edu.tr/test.bin. Your program must accept the path of the file to be sent as a command-line argument. And after sending this single file and receiving the acknowledgement, the program should terminate. You should use threads to implement sending data and receiving acknowledgments (ACKs) concurrently.

For reliable data transfer, you should implement the GBN sender over UDP. The details of the GBN sender entity can be found in the course textbook. You should implement *every* aspect of the GBN sender including the countdown retransmission timer. Your program must accept the window size, N, and the timeout interval in milliseconds as command-line parameters. Your GBN sender must divide the file to be transferred into chunks (segments) each containing a maximum of 1 Kbytes (1024 Bytes) of data.

The packet losses are implemented by the receiver code that is provided to you. The receiver drops a data packet with some specified probability p or it does not send an acknowledgment for a data packet with the same probability p.

Finally, since the sender and the receiver will be run over a local area network with a relatively small delay, the receiver randomly delays the receiving of the data packets and the sending of the acknowledgments for uniformly distributed time values in the interval $[0, D_{max}]$. This will emulate the effects of network delay. D_{max} is in milliseconds.

To summarize the command-line arguments to your program, the usage of the GBN sender should be as follows:
java GBNSender <file_path> <receiver_ip_address> <receiver_port> <window_size_N> <retransmission_timeout>

The GBN Receiver: The receiver implementation is supplied to you as a Java program in the Java archive file receiver.java. You should compile and run it. You may run the GBN receiver as follows:
java -jar receiver.jar <bind_ip_address> <bind_port> <packet_loss_probability_p> <max_packet_delay_D_{max}>

The receiver binds to the specified local IP address at the specified port and implements the GBN protocol receiver entity. It also contains the implementation of the receiver side of the file transfer application. GBN receiver maintains no receive buffer and any out-of-order packets successfully received will be discarded as discussed in the class.

The sequence number of each UDP segment is expected to be the first two bytes of the payload of the UDP packet in big-endian form. In big endian, you store the most significant byte in the smallest address.

Similarly, cumulative acknowledgment numbers will be the first (and only) two bytes of the payload of the acknowledgment packets carried as UDP segments. The receiver assumes that sequence numbers start from 1 and it is incremented by 1 for each segment.

At the application level, the receiver entity just dumps the data it extracts from the data packets into a file named receiver_out.bin. You should validate that the file you send and the receiver_out.bin are identical.

Our application level protocol for transferring files is overly simplified and the receiver does not know the size of the file being sent nor does it know when the transfer ends. You should terminate the sender program after receiving the final acknowledgment. You should manually interrupt the receiver program after the file transfer is completed. The sender program should also report the total file transfer time before it terminates.

In your report, you need to provide the following plots and include a discussion about how your results relate with what you have learned in class.

In your experiments, use the following values for D_{max}, p, retransmission timeout and window size:

- D_{max} = 5 ms, 10 ms, 20 ms
- p = 0, 0.01, 0.1, 0.2
- Timeout =150 ms, 300 ms, 600 ms
- N = 1, 5, 10, 15, 20, 25

A total of 4 plots are required in your report. When plotting a single plot, you should choose a variable and fix the value of all other variables to values from the above list. Then, you should plot transfer time vs. the chosen variable. For example, you can choose D_{max} as 10 ms, p as 0.1, N as 10 and plot transfer time vs. Timeout.

Important

When preparing your project please mind the formatting as this project will be auto-graded by a computer program. Any problems in the formatting can create problems in the grading. While you will not get a zero from your project, you might need to have an appointment with us for a manual grading. Problems caused by Windows/Linux incompatibility will have no effects on your grade.

Submission Rules

You can do this assignment either individually or together with another student from the **same course section.**

You need to apply all the following rules in your submission. **You will lose points if you do not obey the submission rules below or your program does not run as described in the assignment above.**

- The assignment should be submitted as an e-mail attachment sent to yarkin.cetin[at]bilkent.edu.tr. Any other methods (Disk/CD/DVD) of submission will not be accepted.
- The subject of the e-mail should start with [CS421_PA2], and include your name and student ID. For example, the subject line must be

  [CS421_PA2]AliVelioglu20111222

  if your name and ID are Ali Velioglu and 20111222. If you are submitting an assignment done by two students, the subject line should include the names and IDs of both group members. The subject of the e-mail should be

  [CS421_PA2]AliVelioglu20111222AyseFatmaoglu20255666

  if group members are Ali Velioglu and Ayse Fatmaoglu with IDs 20111222 and 20255666, respectively.
- All the files must be submitted in a zip file whose name is the same as the subject line **except the [CS421_PA2] part**. The file must be a .zip file, not a .rar file or any other compressed file.
- All of the files must be in **the root of the zip file**; directory structures are not allowed. Please note that this also disallows organizing your code into Java packages. The archive should not contain:

  – Any class files or other executables,

  – Any third-party library archives (i.e. jar files),

  – Any text files **except the report in pdf format**,

  – Project files used by IDEs (e.g., JCreator, JBuilder, SunOne, Eclipse, Idea or NetBeans etc.). You may, and are encouraged to, use these programs while developing, but the end result must be a clean, IDE-independent program.

- The standard rules for plagiarism and academic honesty apply. For a discussion of academic integrity, please read this document. Refer to 'YOK Student Disciplinary Rules and Regulations', items 7.e and 7.f, and 'Bilkent University Undergraduate Education Regulations' item 4.9 for possible consequences of plagiarism and cheating.