

# COMPUTER NETWORK PROGRAMMING

CS5283

Spring 2004

## PROJECT

(Last updated: 04/08/04)

Marks: 100

The objective of this project is to give you a better understanding of the concepts you learned in this course about TCP, by implementing them. You will be using UDP sockets to simulate the TCP features that are given below.

**Team Project:** This is a group project with each group containing 2 members, exception of 1 member in special cases. Please mail your group details (team member names, id, and section) to TA before **03/30/2004**

**Description:** The project consists of a simple communication between client and server. The client sends packets of data, while the server acknowledges each packet it accepts. There is only data transfer from the client to server. The client uses the window based flow control as given in chapter 12 of “High-Speed Networks and Internets” William Stallings.

The following Three Retransmit policies need to be used:

- First-only
- Batch
- Individual

The client divides the finite data into packets of 1KB size and transmits to the server. The server may reject the packets it received this simulates the packets dropped by network transmission error or by congestion. The server should send an acknowledgement (Immediate policy) for every packet accepted by it, after a delay. This delay can be any value randomly obtained for each ack during runtime between 50 and 70milli seconds. Server acknowledgments should have sequence numbers corresponding to the packets the server accepted. Until all the n acknowledgements are not sent to client, the server will not be able to send n+1 acknowledgement. The server will simply ignore any acknowledgment that needs to be sent out of order; the client will consider the corresponding packets as lost. In other words the acceptance policy of the server is In-Order. The client has a constant window size of 20 packets. When client receives a successful acknowledgement from server it will increment the leading and trailing window positions and send additional packets.

The project is divided into two phases:

**Phase-1:** The client is implemented with out congestion control. Retransmission Timer (RTT) value is given as 60milli seconds. The simulation should have the finite data starting from 1MB to 10MB with increments of 1MB and resetting all the intermediate values at the beginning of simulation run, for a finite data. The

probability of dropping a packet by server in order to simulate transmission errors starts from 0 to 0.1 with increments of 0.02. Calculate the estimated time taken for total data transfer for all the three retransmit policies along with probabilities of dropping packets. Show the backend working of the scheme that is being used. Also keep track of the throughput, total number of packets transmitted and any other information that you may require for the graphs below (35 Marks).

**Graphs:** Plot graphs for the time taken for the total data transfer, with respect to the finite data sent by the client, for all the probabilities and for a given Retransmission policy (X axis: finite data increments, Y axis time, 6 lines in each graph representing probability value, 3 graphs for each of the Retransmission schemes). Similarly plot graphs for Throughput and total # of packets transmitted (Y axis) (15 Marks).

**Phase-2:** The client is implemented with congestion control using the Slow Start; you need not implement Fast retransmit or Fast recovery. The probability for the server to drop packets is 0.06 (network congestion). Similar to phase 1 finite data starting from 1MB to 10MB with increments of 1MB must be transmitted for each simulation run. The following RTT values should be used.

- Low value: 50milli seconds
- High value: 80milli seconds.
- Medium: 60milli seconds.

Calculate the estimated time taken for the total data transfer for any of the three Retransmission policies and using any of the three RTT values, along with throughput and total number of packets transmitted (20 Marks).

**Graphs:** Plot graphs for time taken for the total data transfer, with respect to the finite data sent by the client, for each of the retransmission schemes and for a given RTT value (X axis: finite data increments, Y axis: time, 3 lines in a graph each representing RTT values, 3 graphs for each of the Retransmission schemes). Similarly plot graphs for Throughput and Total # of packets transmitted (Y axis) (10 Marks).

**Additional Graphs (Bonus 10 Marks):**

From the Data collected in the above simulations, separate the information pertaining to these conditions form both phase-1 and phase-2.

- Probability: 0.06
- RTT value: 60milli seconds.

Plot graphs for the time taken for total data transfer, with respect to the finite data sent by the client in each phase. (X axis: Data increments, Y axis: time, 2 lines in a graph each representing a phase, 3 graphs each representing retransmission scheme). Similarly plot graphs for Throughput and Total # of packets transmitted (Y axis).

**Programming Instructions:**

- The server should accept port number as a command line argument.
- The client should have Server name, Port number and other arguments that can make your life easy, as command line arguments.
- Use print statements to give a clear idea of the process going in both client and server.
- Please visit the specifications regularly as updates to these may be given.

**External Document:** An external document **precisely** explaining how each of the Retransmission policies, Congestion control, and other policies given in project is implemented should be given (**not more than 1 page**). The graphs discussed above must be present in this document along with a plausible **concise** reason for that behavior. **Give the data collected for each graph in a tabular form to verify the authenticity of graphs.** Also give the details of each member's role in handling the project.

**Demo:** The Stillwater and Tulsa students are given the privilege of giving a demo to TA; there may not be any demo for other section students. A hard copy of external document must be submitted during the demo, offsite students can use handin to submit their external document.

**Submission:** Both the client and server programs must be submitted using handin to **project** directory present in the cs5283 class account on CSA. The due date for program submission is **04/27/04**. Demo dates: Stillwater students **04/28/04**, Tulsa students **04/30/04**.

**Additional Instructions:**

- The programs must have both Internal and External Documentation.
- The External documentation as a beginning header block of each program should have
  - Your name and id.
  - The section you belong to (Stillwater, Tulsa etc).
  - Compile instructions for your program.
  - A small description of what the program does.
  - A detailed explanation of all the command line arguments and inputs given to the program.
- Make sure you follow the specifications and give comments.
- The programs will be tested only on the CS department servers and in the MS 222 lab, so see that your programs are not machine dependent. No points will be awarded if the programs are not compiled on the CSA (a.cs.okstate.edu) server.
- Use print statements to display on console, the events that are going behind the scene.
- Don't forget to **close the sockets**.

- If your program doesn't compile then a maximum of **5 points** will be awarded.
- **Documentation, compilation carry 20 Marks.**