ECE361S Final Lab Project:

# Implementing a reliable file transfer protocol, with routing and congestion control

## Scenario:

* Similar to the previous lab instructions, connect to the ECE361 lab server.
* First, using what you learned in Lab 5, you should find the shortest path to from node no. “0” (which is assumed to be you as the client) to the last node (the node with number “noNodes-1”).
* The number of nodes is a random number generated by the server and sent to you as a String value.
* Find the shortest path and send it to the server.
* Second, you should transfer a file to server (to the node number “noNodes-1”. Therefore, the RTT will be calculated based on the shortest path you found in the previous section). This is similar to Lab 2, but for extra simplicity you should use the same connection for control and data communications.
* Read the file name from the user and send it to the server.
* Use packets of size **1004** bytes for each transfer.
* Use the first **4** bytes of the packet as a header, containing the sequence number of the packet. You can use “ByteBuffer” along with the “getInt()” and “putInt()” methods or any other arbitrary method to convert integers into byte arrays and vice versa.
* It is OK if the last packet is smaller than 1004 bytes since for the last packet, the end of the file may be reached in less than 1000 bytes.
* Send the number of packets (noPackets) to the server as a string value.
* Start transmitting the file contents, along with proper sequence numbers.
* For each transmitted packet, you should receive the ack with the same sequence number in string format. If the ack is not received within the allowed timeoutInterval, you should retransmit.
* For simplicity, use a fixed timeout interval, equal to 2 ×(delay of the shortest path from node “0” to node “noNode-1” that was obtained in previous section)+200 ms. (the timeoutInterval is basically the RTT (=2 × propagation delay plus a small value for considering other factors that cause delay).
* You should adjust the rate of transmission since the server will simulate a limited random bandwidth and will drop packets with rates higher than that (use TCP Tahoe method for adjusting the rate).
* Overall, this section is similar to what you did in Lab 4, except that the sequence numbers and acks are 4 byte integers (instead of one byte integers), and this time not only the data packets contain a sequence number (the first 4 bytes of each packet), but also they contain actual data from a file (the next 1000 bytes of each packet) that is supposed to be transmitted to the server.

## Notes:

1. [Emphasize] Use packets of size 1004 bytes for sending the file. The first 4 bytes of the buffer goes for seq. no. --> 1000 byte data.
2. [Emphasize] All the data transfer is done on a single socket.
3. All control information is send via String format, i.e., use DataOutputStream.writeBytes(str+CRLF) for writing and use BufferedReader.readline() for reading. This includes everything (even the acks) except the 1004 byte packets used for transmitting the file and its sequence number. The format for those packets is byte arrays.
4. All data information is sent as byte streams, i.e., use DataOutputStream.write(byte[],0,len) for sending the packets of size 1004 bytes and use int len=DataInputStream.read(byte[]) for reading the packets.
5. Methods like StringTokenizer.nextInt(), Integer.parseInt(), String.Valueof(), etc. can be useful for converting data from String to Integers or bytes and vice-versa.
6. You can use the latest version of the server for all other lab sections. The server can run different Service types (for different Lab sections) and different modes (for logging purpose). Enter the proper service type and mode as input arguments (for description type in: *$java MainClass help*)
7. Your program should be multi-threaded (at least one for sending the data, which could be the main thread, and one for listening to the acks). You will lose marks if it is single threaded.
8. You will receive marks based on how your code is written, as well as: if you find the correct path, successfully transmit the entire file, and adjust the rate so that the transmission is done within an acceptable time interval. If your program won’t run, you will most likely receive no mark. You may get some mark if the code is written cleanly, commented properly, and prints brief and meaningful information in the console (similar to the sample provided below).
9. Don’t forget to close the Sockets and FileOutputStreams/FileInputStreams after you are done working with them.
10. The server (in verbose mode) prints almost everything it receives from or sends to the client. Use it for debugging your code. The following sample run shows how the server and client should interact.

|  |  |
| --- | --- |
| Sample client output: | Sample Server output |
| $ java MainClient  Connected to : localhost:9876  waiting to receive the number of nodes...  number of nodes:7  Adjacency Matrix:  Infinity 693.0 265.0 Infinity 153.0 272.0 540.0  291.0 483.0 398.0 Infinity Infinity 861.0 648.0  394.0 Infinity 115.0 972.0 194.0 620.0 826.0  933.0 Infinity 549.0 898.0 218.0 Infinity 155.0  167.0 148.0 860.0 619.0 480.0 Infinity 635.0  798.0 286.0 254.0 Infinity 302.0 400.0 542.0  786.0 617.0 654.0 128.0 Infinity Infinity 972.0  Node 0  Total time to reach node 0: 0.0 ms, Path: [0]  Total time to reach node 1: 301.0 ms, Path: [0, 4,  Total time to reach node 2: 265.0 ms, Path: [0, 2]  Total time to reach node 3: 668.0 ms, Path: [0, 6,  Total time to reach node 4: 153.0 ms, Path: [0, 4]  Total time to reach node 5: 272.0 ms, Path: [0, 5]  Total time to reach node 6: 540.0 ms, Path: [0, 6]  Enter the name of the file: test.txt  file length: 45326  cwnd= 1  sending packet no:1  received 1  Received ack no:1  last ack:1  # of acks received for cwnd of 1  cwnd= 2  sending packet no:2  sending packet no:3  received 2  Received ack no:2  received 3  Received ack no:3  last ack:3  # of acks received for cwnd of 2  cwnd= 4  sending packet no:4  sending packet no:5  sending packet no:6  sending packet no:7  received 4  Received ack no:4  received 5  Received ack no:5  received 6  Received ack no:6  received 7  Received ack no:7  last ack:7  # of acks received for cwnd of 4  cwnd= 8  sending packet no:8  sending packet no:9  sending packet no:10  sending packet no:11  sending packet no:12  sending packet no:13  sending packet no:14  sending packet no:15  received 8  Received ack no:8  received 9  Received ack no:9  received 10  Received ack no:10  received 11  Received ack no:11  received 12  Received ack no:12  received 13  Received ack no:13  received 14  Received ack no:14  received 15  Received ack no:15  last ack:15  # of acks received for cwnd of 8  cwnd= 16  sending packet no:16  sending packet no:17  sending packet no:18  sending packet no:19  sending packet no:20  sending packet no:21  sending packet no:22  sending packet no:23  sending packet no:24  sending packet no:25  sending packet no:26  sending packet no:27  sending packet no:28  sending packet no:29  sending packet no:30  sending packet no:31  received 16  Received ack no:16  received 17  Received ack no:17  received 18  Received ack no:18  received 19  Received ack no:19  received 20  Received ack no:20  received 21  Received ack no:21  received 22  Received ack no:22  received 23  Received ack no:23  received 24  Received ack no:24  last ack:24  Time Out  cwnd= 1  sending packet no:25  received 25  Received ack no:25  last ack:25  # of acks received for cwnd of 1  cwnd= 2  sending packet no:26  sending packet no:27  received 26  Received ack no:26  received 27  Received ack no:27  last ack:27  # of acks received for cwnd of 2  cwnd= 4  sending packet no:28  sending packet no:29  sending packet no:30  sending packet no:31  received 28  Received ack no:28  received 29  Received ack no:29  received 30  Received ack no:30  received 31  Received ack no:31  last ack:31  # of acks received for cwnd of 4  cwnd= 8  sending packet no:32  sending packet no:33  sending packet no:34  sending packet no:35  sending packet no:36  sending packet no:37  sending packet no:38  sending packet no:39  received 32  Received ack no:32  received 33  Received ack no:33  received 34  Received ack no:34  received 35  Received ack no:35  received 36  Received ack no:36  received 37  Received ack no:37  received 38  Received ack no:38  received 39  Received ack no:39  last ack:39  # of acks received for cwnd of 8  cwnd= 9  sending packet no:40  sending packet no:41  sending packet no:42  sending packet no:43  sending packet no:44  sending packet no:45  sending packet no:46  received 46  Received ack no:46  received  last ack:46  # of acks received for cwnd of 9  Total time to send all packets: 11 seconds.  Total time in terms of RTT: 10 RTT.  46 out of 46 packets have been sent successfully  Quitting... | $ java MainClass 9876 PROJECT\_SERVER VERBOSE  Server online.  Host name: Shervin-VAIO  Host Address: 192.168.1.102:9876  waiting for requests.  request received. request number: 1 client: /127.0.0.1:57136  connection established:  service type:PROJECT\_SERVER  mode:VERBOSE  client id:1  socket: Socket[addr=/127.0.0.1,port=57136,localport=9876]  [07:31:50] Sending the number of nodes to client:  [07:31:50] to 1: 7  [07:31:50] Adjacency Matrix  Infinity 693.0 265.0 Infinity 153.0 272.0 540.0  291.0 483.0 398.0 Infinity Infinity 861.0 648.0  394.0 Infinity 115.0 972.0 194.0 620.0 826.0  933.0 Infinity 549.0 898.0 218.0 Infinity 155.0  167.0 148.0 860.0 619.0 480.0 Infinity 635.0  798.0 286.0 254.0 Infinity 302.0 400.0 542.0  786.0 617.0 654.0 128.0 Infinity Infinity 972.0  [07:31:50] to 1: Infinity 693.0 265.0 Infinity 153.0 272.0 540.0 291.0 483.0 398  .0 Infinity Infinity 861.0 648.0 394.0 Infinity 115.0 972.0 194.0 620.0 826.0 93  3.0 Infinity 549.0 898.0 218.0 Infinity 155.0 167.0 148.0 860.0 619.0 480.0 Infi  nity 635.0 798.0 286.0 254.0 Infinity 302.0 400.0 542.0 786.0 617.0 654.0 128.0  Infinity Infinity 972.0  [07:31:50] From Node 0  Total time to reach node 0: 0.0 ms, Path: [0]  Total time to reach node 1: 301.0 ms, Path: [0, 4, 1]  Total time to reach node 2: 265.0 ms, Path: [0, 2]  Total time to reach node 3: 668.0 ms, Path: [0, 6, 3]  Total time to reach node 4: 153.0 ms, Path: [0, 4]  Total time to reach node 5: 272.0 ms, Path: [0, 5]  Total time to reach node 6: 540.0 ms, Path: [0, 6]  [07:31:50] waiting to receive the selected path from client...  [07:31:50] from 1: [0, 6]  (0, 6) delay: 540.0  [07:31:50] CORRECT path selected.  [07:31:50] waiting to receive the file name (in string format)...  [07:31:54] from 1: test.txt  [07:31:54] waiting to receive the number of packets (in string format)...  [07:31:54] from 1: 46  [07:31:54] Waiting to receive data...  [07:31:54] 1: Received packet#1  [07:31:54] 1: Acknowledging packet #1  [07:31:55] 1: Received packet#2  [07:31:55] 1: Acknowledging packet #2  [07:31:55] 1: Received packet#3  [07:31:55] 1: Acknowledging packet #3  [07:31:56] 1: Received packet#4  [07:31:56] 1: Acknowledging packet #4  [07:31:56] 1: Received packet#5  [07:31:56] 1: Acknowledging packet #5  [07:31:56] 1: Received packet#6  [07:31:56] 1: Acknowledging packet #6  [07:31:56] 1: Received packet#7  [07:31:56] 1: Acknowledging packet #7  [07:31:57] 1: Received packet#8  [07:31:57] 1: Acknowledging packet #8  [07:31:57] 1: Received packet#9  [07:31:57] 1: Acknowledging packet #9  [07:31:57] 1: Received packet#10  [07:31:57] 1: Acknowledging packet #10  [07:31:57] 1: Received packet#11  [07:31:57] 1: Acknowledging packet #11  [07:31:57] 1: Received packet#12  [07:31:57] 1: Acknowledging packet #12  [07:31:57] 1: Received packet#13  [07:31:57] 1: Acknowledging packet #13  [07:31:57] 1: Received packet#14  [07:31:57] 1: Acknowledging packet #14  [07:31:57] 1: Received packet#15  [07:31:57] 1: Acknowledging packet #15  [07:31:59] 1: Received packet#16  [07:31:59] 1: Acknowledging packet #16  [07:31:59] 1: Received packet#17  [07:31:59] 1: Acknowledging packet #17  [07:31:59] 1: Received packet#18  [07:31:59] 1: Acknowledging packet #18  [07:31:59] 1: Received packet#19  [07:31:59] 1: Acknowledging packet #19  [07:31:59] 1: Received packet#20  [07:31:59] 1: Acknowledging packet #20  [07:31:59] 1: Received packet#21  [07:31:59] 1: Acknowledging packet #21  [07:31:59] 1: Received packet#22  [07:31:59] 1: Acknowledging packet #22  [07:31:59] 1: Received packet#23  [07:31:59] 1: Acknowledging packet #23  [07:31:59] 1: Received packet#24  [07:31:59] 1: Acknowledging packet #24  [07:31:59] 1: Received packet#25  [07:31:59] 1: Dropped packet #25  [07:31:59] 1: Received packet#26  [07:31:59] 1: Dropped packet #26  [07:31:59] 1: Received packet#27  [07:31:59] 1: Dropped packet #27  [07:31:59] 1: Received packet#28  [07:31:59] 1: Dropped packet #28  [07:31:59] 1: Received packet#29  [07:31:59] 1: Dropped packet #29  [07:31:59] 1: Received packet#30  [07:31:59] 1: Dropped packet #30  [07:31:59] 1: Received packet#31  [07:31:59] 1: Dropped packet #31  [07:32:00] 1: Received packet#25  [07:32:00] 1: Acknowledging packet #25  [07:32:01] 1: Received packet#26  [07:32:01] 1: Acknowledging packet #26  [07:32:01] 1: Received packet#27  [07:32:01] 1: Acknowledging packet #27  [07:32:02] 1: Received packet#28  [07:32:02] 1: Acknowledging packet #28  [07:32:02] 1: Received packet#29  [07:32:02] 1: Acknowledging packet #29  [07:32:02] 1: Received packet#30  [07:32:02] 1: Acknowledging packet #30  [07:32:02] 1: Received packet#31  [07:32:02] 1: Acknowledging packet #31  [07:32:03] 1: Received packet#32  [07:32:03] 1: Acknowledging packet #32  [07:32:03] 1: Received packet#33  [07:32:03] 1: Acknowledging packet #33  [07:32:03] 1: Received packet#34  [07:32:03] 1: Acknowledging packet #34  [07:32:03] 1: Received packet#35  [07:32:03] 1: Acknowledging packet #35  [07:32:03] 1: Received packet#36  [07:32:03] 1: Acknowledging packet #36  [07:32:03] 1: Received packet#37  [07:32:03] 1: Acknowledging packet #37  [07:32:03] 1: Received packet#38  [07:32:03] 1: Acknowledging packet #38  [07:32:03] 1: Received packet#39  [07:32:03] 1: Acknowledging packet #39  [07:32:04] 1: Received packet#40  [07:32:04] 1: Acknowledging packet #40  [07:32:04] 1: Received packet#41  [07:32:04] 1: Acknowledging packet #41  [07:32:04] 1: Received packet#42  [07:32:04] 1: Acknowledging packet #42  [07:32:04] 1: Received packet#43  [07:32:04] 1: Acknowledging packet #43  [07:32:04] 1: Received packet#44  [07:32:04] 1: Acknowledging packet #44  [07:32:04] 1: Received packet#45  [07:32:04] 1: Acknowledging packet #45  [07:32:04] 1: Received packet#46  [07:32:04] 1: Acknowledging packet #46  [07:32:05] 46 out of 46 packets have been received.  [07:32:05] CORRECT, file transmission was reliable  [07:32:05]  Total bandwidth: 9 MSS/RTT  Total number of packets: 46  Total transmission time: 11176 seconds.  Average round trip time: 1080  Minimum possible transmission time: 5.111111111111111 RTT.  Expected transmission time using TCP Tahoe (for initial ssthres= [big value]):10  RTT.  Actual transmission time: ~10.348148148148148 RTT.  Actual throughput: 0.49391553328561205  [07:32:05] CORRECT answer. data received within the allowed time.  --------------------------------------------------  [07:32:05] FINAL SCORE (out of 3): 3  connection to 1 closed. |

Goodluck. ☺