CSC/ECE 573 – Internet Protocols

Project #2- Report

Team Members:

Nikhila Nathani (nnathan)

Rithish Koneru (rkoneru)

Overview:

A point to multipoint reliable data transfer is shown in this project. The underlay is on the UDP network, and the protocol used to implement and send & receive data packets is stop-and-wait protocol, with ARQ scheme.

Here, the client become the sender, and several servers become the receivers. The sender to receiver part handles the files or byte stream transfer with the MSS in store, that is broadcasting to all the receivers in accordance to stop-and-wait protocol, that is to send a packet to all the receivers, and wait till an acknowledgement is received from every single one of them, before transferring the next one. Hence, the receiver to sender part, is to send the ACK as soon as the file is received without being corrupt.

In the protocol, the initial sequence number is always set to zero, therefore having only one data segment that is outstanding at any point of time. And the header of the file is 8 bytes, with a 32-bit sequence number, and 16-bit checksum, and a 16-bit field, that is predefined. This segment is received by the server on the default port that is 7735 in this case. Then the checksum is calculated and an in-sequence check is done, if everything is verified to be true, server sends an ACK, and writes the data to output file. If not in sequence, the ACK for the last received packet is sent. And the header for the ACK is also 8 bytes, which has 32-bit sequence number, a 16-bit field of 0's and a 16-bit field of predefined value.

Implementation:

P2MP UDP Client:

The client here, creates an individual thread for each server, so that it can have an individual connection for transmission, and to wait for ACK of the sent packet. There will be at least one segment of MSS size.

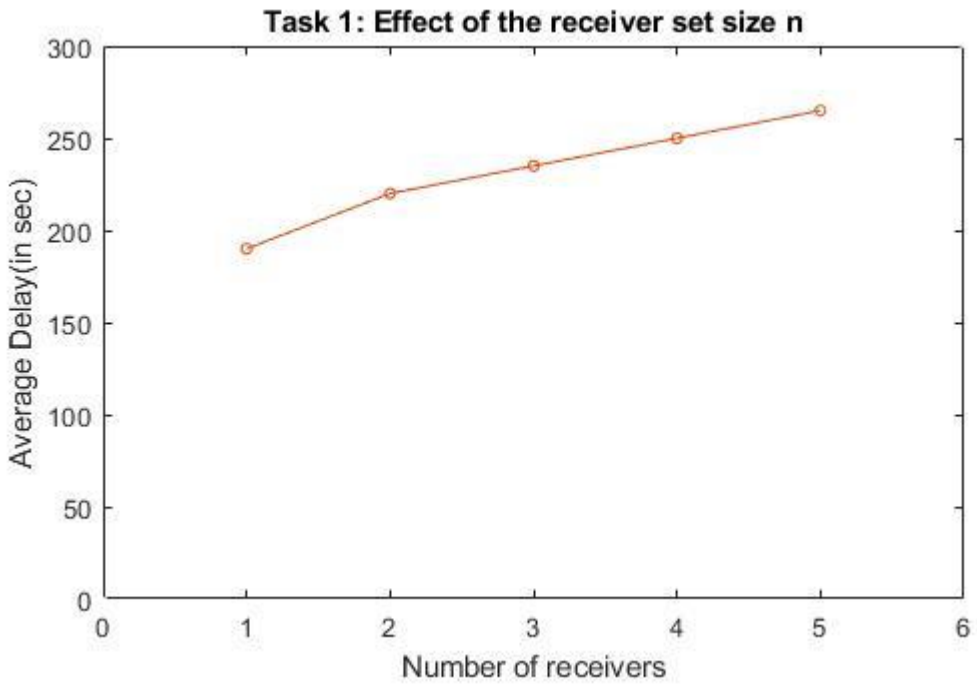
Each packet of MSS size, is sent through a broadcast mechanism, and the threads used are synchronized, before going to the next packet of transfer. Since each file starts with the sequence number as zero, it helps in identifying the start of a new file each time. The time taken to build the segments to computing the checksum and adding the header, transmitting and then the time for ACK's to get back, make up for the total transfer delay. Even the retransmitting accounts into the delay time.

P2MP UDP Server:

The server listens on the port 7735, for the packet, and the receiver side of the stop-and -wait protocol is implemented on this, where the loss probability can be specified, and checked, using a random number that is generated. The condition is such that, if the probabilistic loss is less than the random number, it is ignored, else the packet is discarded.

So, if a duplicate packet is received, and the ACK for that was already sent, the packet is discarded and an ACK is resent. And when a file is sent more than once, to get an average transfer delay, the file is overwritten so that the contents are not accumulated, those many times.

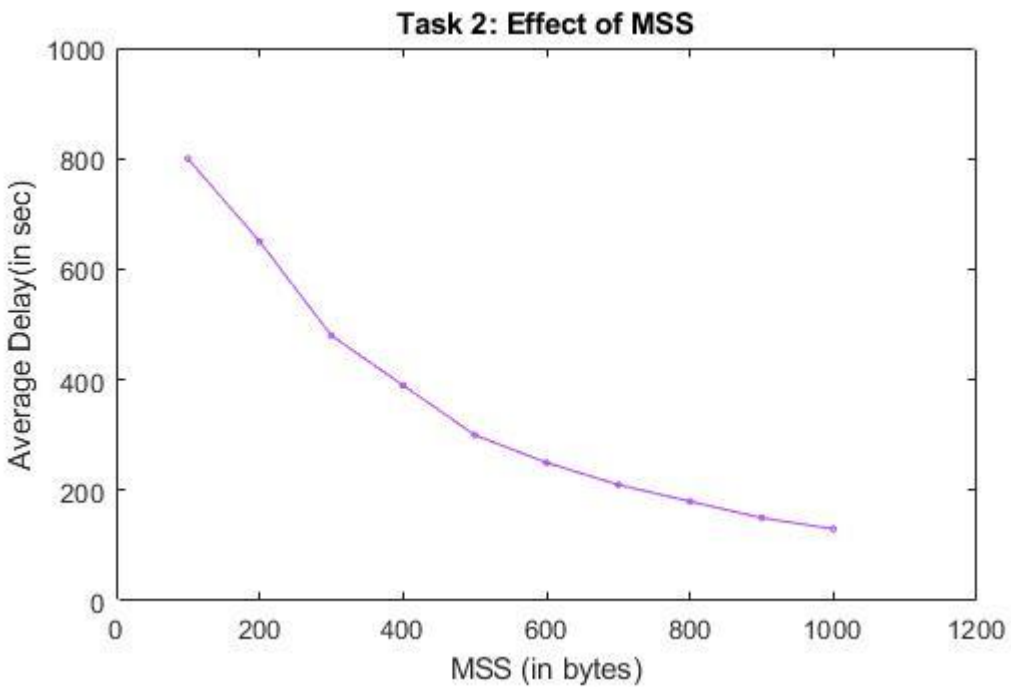
Task 1: Effect of the receiver set size n



The MSS was set to 500 bytes, and the loss probability to 0.05. The experiment was varied between 5 receivers, and to compute average time delays, the file was transmitted to each server, 5 times, overall accounting to 25 transmissions. Each time, the 1 MB text file was broadcasted to those servers.

The observation showed that the average delay increased linearly, as the number of receivers increased. This can be because the loss probability of each server is a constant, since the loss probability effects the delay time, to a certain extent, the curve increases linearly. If the loss probability differs for each server, then the graph might not show linearity.

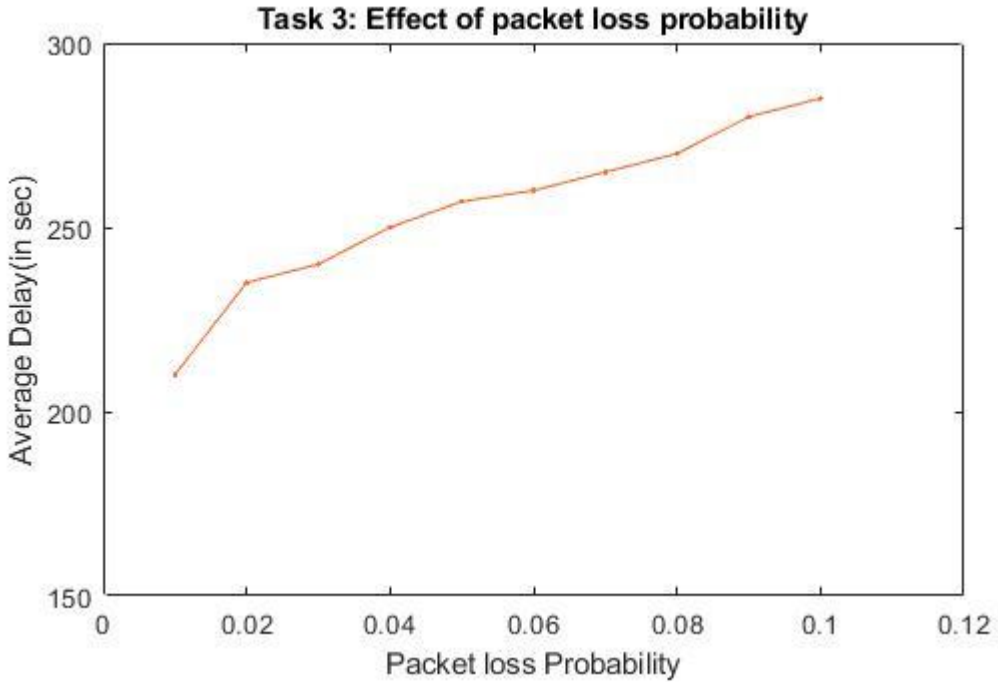
Task 2: Effect of MSS



The simulation for this is done on 3 receivers, with a loss probability set to 0.05 and the maximum segment size varying from 100 to 1000, in increments of 100. Each simulation was done 5 times on each MSS size of packet. Thereby getting 50 in total.

Here it can be observed that the Delay decreases, as the MSS increases, it is because the number of packets that are being trasnmitted decrease due to increase in size and also the number of retransmissions will take effect in a decrease, proportionally, hence the file transfer delay decreases exponentially.

Task 3: Effect of packet loss probability



The simulation here, also has 3 servers, and the MSS to be 500, with the loss probability, changing by 0.01 from 0.01 to 0.1. The simulations are done 5 times at each probability set, hence overall 50 are done.

The average delay increases with the increase in packet loss probability, since the delay also accounts to the retransmission time, and when the packets are lost, retransmission rate increases, and with the increase in loss probability, the average delay also increases. Since the number of servers are the same, there will be a linear increase in retransmission, which leads to linear increase in file transfer delay.

Conclusion:

The stop-and-wait protocol, doesn't provide reliable data transfer on UDP very efficiently. The main disadvantage of this protocol is that the sender waits till all the ACK's are received, before transmitting the next segment. This pertains to increase the delay time in file transfer, and performance degradation. Better scalability can be obtained, if the file is transferred without the synchronization for each receiver, as opposed to broadcasting.