The program consists of a client and a server that are run in separate terminals to be connected to each other. The server will be run first, the client will connect. Once the client has successfully connected to the server, it will send a series of ten probe messages with a varying amount of message sizes. For the sake of the experiment, each iteration of messages will have a size of for: 1.) Round-trip time: 1, 100, 200, 400, 800, and 100, 2.) Throughput: 1000, 2000, 4000, 8000, 16000, 32000.

Measuring the round-trip time entails: recording the current time using the time library, sending the probe, receiving the same message back from the server based on the arbitrary server delay value, and then recording the time again. The start will then be subtracted from the end to gather the final round-trip time. Below are the graphs for RTT with 0 server delay and a server delay with one second with an error margin of 5%.

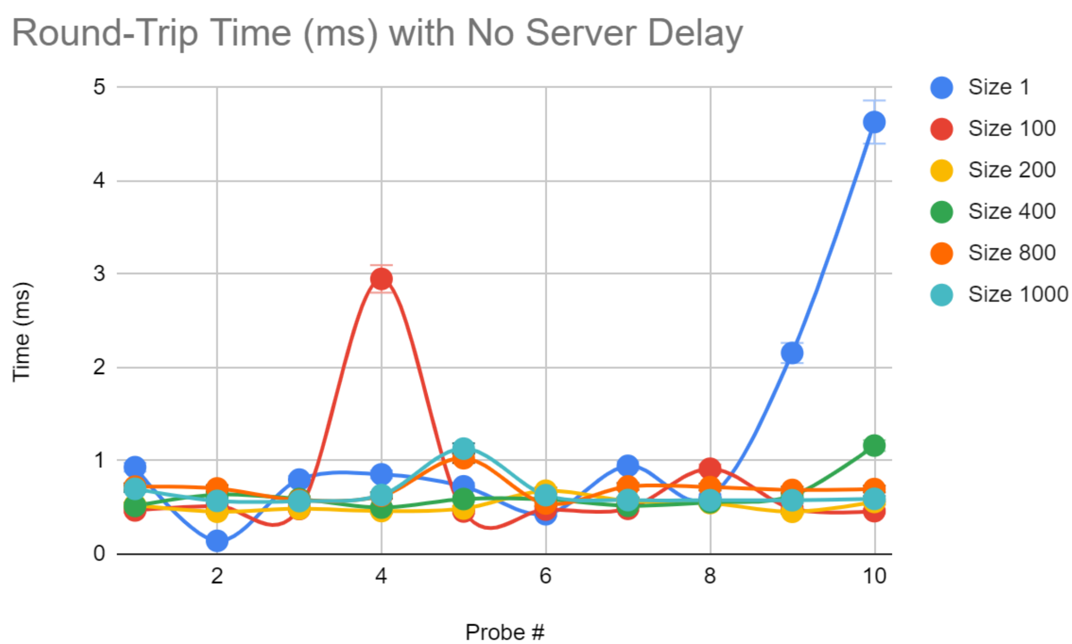

Figure 1.1 Round-Trip Time with No Server Delay

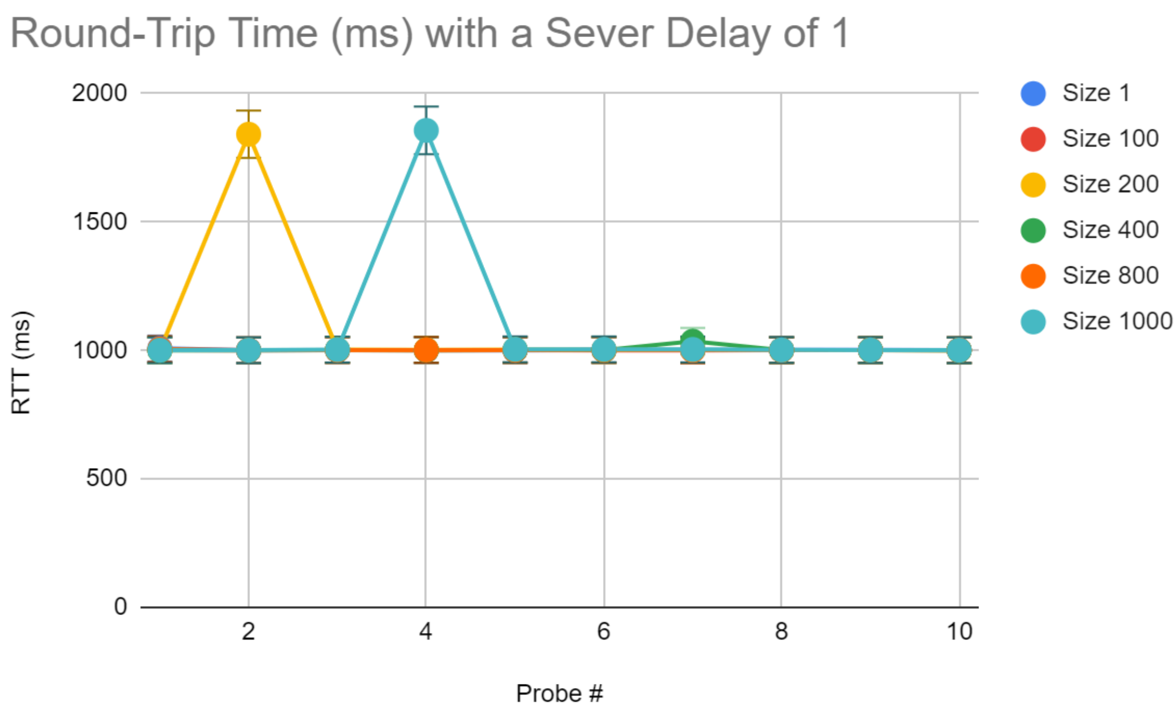

Figure 1.2 Round-Trip Time with Server Delay of 1

As you can see, in Figure 1.2, the results are very similar to Figure 1.1. There is not much difference in packet sizes on a local machine. However, the server delay does reflect the round trip time because the server must wait an extra 1000 ms before echoing back the response. Again, like 1.1, there are a few outliers notably packet 2 with size 800 bytes and packet four with a size of 1000. These could be caused by a number of different reasons, but are not reflective of the final number with an average RTT per packet size of:

Size 1 Average: 1008.3694458007812 s
Size 100 Average: 1001.0912418365479 s
Size 200 Average: 1001.3682842254639 s
Size 400 Average: 1001.0416507720947 s
Size 800 Average: 1000.9219646453857 s
Size 1000 Average: 1002.0036697387695 s

Similar to the round-trip measurement, the start time is recorded and stored in a ‘start’ variable. A message of given payload size is sent, it is received back based on the server delay and then end time is then recorded. However, an additional step is to calculate the throughput rate and then calculate by dividing the size of the message by the duration (RTT).

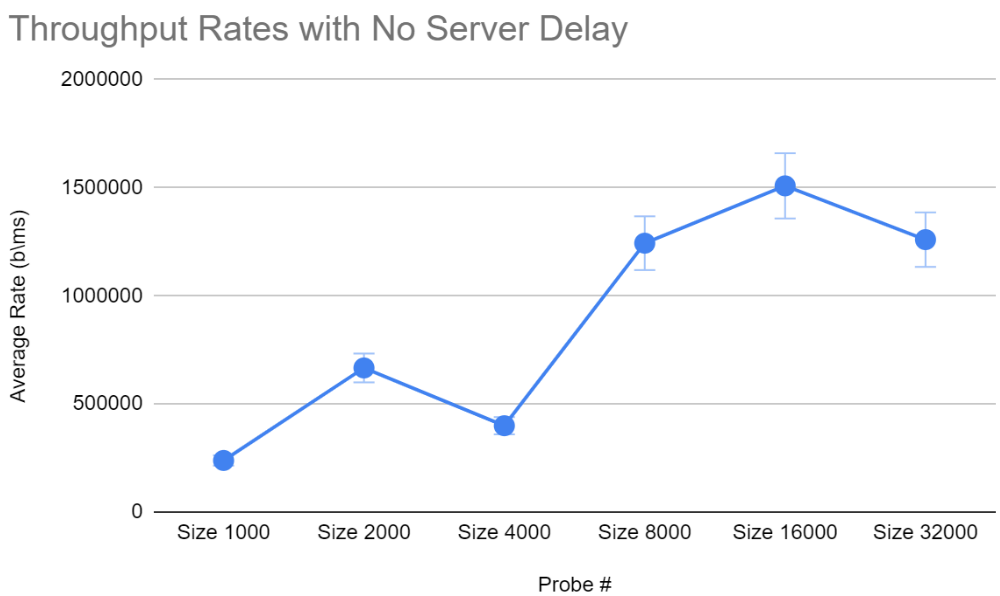

Figure 1.1 Throughput with No Server Delay

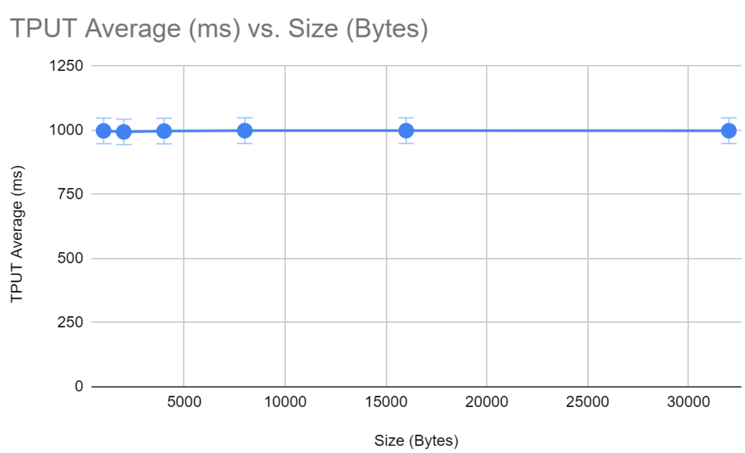

Figure 1.2 Throughput with Server Delay of 1s

Reading the graphs, it is obvious to see the difference between graphs. With server delay, the rate of throughput in bits per second is significantly decreased since less data is being sent per second. Overall 1.2 remains relatively uniform despite packet size. Surprisingly 1.1, experiencing a uptick of data transfer with the last three larger payload sizes.