CS550 - Advanced OS

Fall 2020

Amit Nikam

A20470263

anikam@hawk.iit.edu

EVALUATION REPORT

1) One client to the server connection:

Establishing a successful server and client connection by defining communication protocol was the first challenge in implementing multithreaded client and server.

For this test, my client was able to connect to the server and download a .jpg format image of the size 48422 Bytes in just 0.0022 seconds.

2) 8 clients connected to the server:

The next step was to implement a server which can handle multiple clients at the same time. To achieve this, threading was implemented on the server. Each connection to the server is handled through a handler which is threaded. This way multiple connections are handled at the same time.

For 8 clients downloading the same .jpg image file:

- Average Transfer Times were 0.0045925 seconds
- Average Downloads = 1.125
- Total Failed Downloads = 1

It is observed that the transfer time doubled i.e. transfer speed was halved compared to case 1.

3) Multiple clients downloading multiple files:

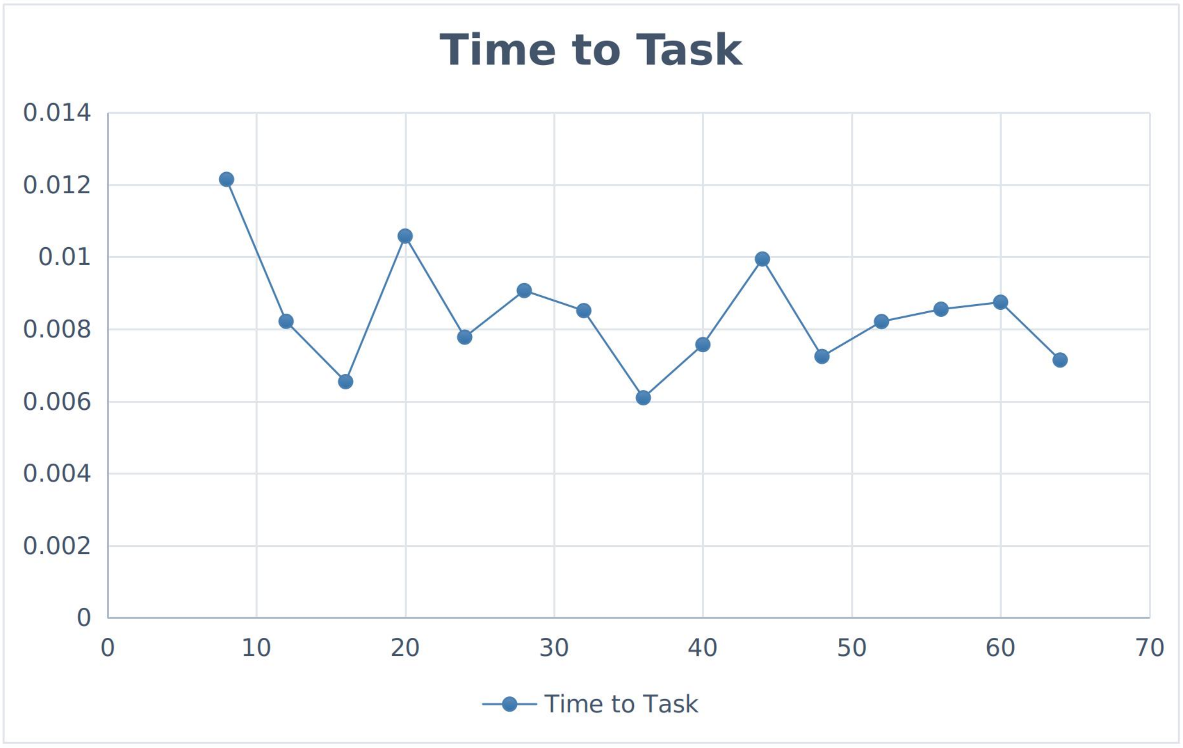
For this evaluation, 3 files of which 2 are images and 1 is text of sizes 28794, 48422 and 231 Bytes were used. The same process of downloading these three files was repeated for incrementing number of client connections.

The following table is based on the implementation outcomes

# Client	Average Transfer Time per File	Average Downloads	Average Total Time each client spent downloading
8	0.002700057	4.5	0.012150258
12	0.002464616	3.333333333	0.008215388
16	0.001517047	4.3125	0.006542265
20	0.002938314	3.6	0.010577929
24	0.001985314	3.916666667	0.007775813
28	0.002395358	3.785714286	0.00906814
32	0.002368367	3.59375	0.00851132
36	0.001769735	3.444444444	0.006095754
40	0.002005096	3.775	0.007569236
44	0.002877995	3.454545455	0.009942163
48	0.002145566	3.375	0.007241284
52	0.002307235	3.557692308	0.008208431
56	0.002370425	3.607142857	0.008550461
60	0.002498435	3.5	0.008744522
64	0.002004627	3.5625	0.007141482

It can be observed from the above table that clients have re-downloaded the files leading to more than 3 file downloads. Reason for a re-download can be either a lost connection / timeout / no service from server. By multiplying the average file transfer times for each of the cases with the average downloads we can get the average total time each client spend downloading the same 3 files from the server.

We would be plotting this outcome with respect to the number of clients and would be referring to it as 'time to task' to get better visual understanding.



It is observed that the server performed really well at 36 concurrent client connections although the performance was really poor at 8 connections.

In the end, a multithreaded client and multithreaded server implementation was achieved. Even under heavy loads the server performance was consistent with just an average Time To Task deviation of +/-0.0025 seconds.