Documentation

Distributed Systems – Project I

I have neither given or received unauthorized assistance on this work

Signed: Mitul Magu Date: 02/24/2019

Name/UTA ID: Zeyu Zhang (1001686894), Mitul Magu (1001644213)

Programming Language Used: C++

Working of the Code

Basic socket programming is used to establish the connection between server and client. Client's command will be packaged into a string and sent to the server. After receiving the command string from the client, the server uses split method to split the string into pieces and decode each part of the command. According to different commands, the server runs different procedures.

The upload and download rely on the send and recv methods to transport the data between client and server. Rename and Delete are performed by built-in function 'rename' and 'remove'.

For 'calculate_pi', the server uses a recursive algorithm to compute pi, and it limits the accuracy to 3.141592. For 'add', after receiving two integers from the client, the server just adds them and sends the result back. For 'sort', the server would receive a string of integers and put them into an array, and it calls build-in 'sort' function to sort them, put them into string, and sends it back. For 'matrix_multiply', the client will be asked to input the info of two matrices including dimensions and actual elements in two matrices. All those info will be packaged into a string and send to the server. After receiving it, the server decode the numbers and form two matrices as form of 2-D vectors. It multiplies the two matrices, packaging the result into a single string and sending it back.

Challenges Faced

Many challenges were faced while working with the code. At an initial stage, the client was repeatedly unsuccessful in connecting with the server which made uploading and downloading files to the server very difficult. Modifications had to be made in an attempt to make the connection successful and the issue was later on resolved. Issues were faced at first in creating a docker image but was also resolved later.

Learning Outcome

The project taught us to work with message oriented and rpc based communication and how it can be applied to real life working projects. We also learnt the use of Docker and how to work with Docker images which is a great skill to have.

Converting Single-threaded server to multi-threaded version

Converting the code from a single threaded server to a multi-threaded server will although allow the program to waste less memory by accepting multiple requests on the multi-threaded program, but will waste time for the client if too many requests are sent to the server. The length of the code will also be significantly be increased as the code is modified to allow multiple threads. There is a trade-off between the two options and the one that fits the application requirement must be used.

Why is Distributed locking necessary for the multithreaded servers?

Distributed locking is necessary for multi-threaded servers because in multi-threaded servers we are not only concerned with multiple processes but also multiple clients working on different machines. We need to ensure that these clients that are working on completely separate machines do not interfere with one another while accessing the resource, that is we want only one client to access the resource at a given time. If the resource is shared then two clients that are working with the server concurrently may alter the system state and provide incorrect results. Distributed locking will also ensure the work is done efficiently such that computation is not carried out more than once for the same reason.