Project One Design Report

Design and Structure

For minimum latency, every command from the client is sent to the server once and the server will grab the argument in this command layer by layer and finally server operates the specific command with the corresponding opcode.
Once the server run, it will never stop and wait for the coming command from the client. The pipeline of the execution at server side is as follow:

1) The server receives a message from the client, immediately **fork** and for this specific client and wait for the next consecutive commands.
2) Grab the opcode from message and go into specific a function(selected by function select_action()) according to the opcode.
3) Execute corresponding instructions at the server-side and send back return values and errnos.
4) The client receives return values and errnos.

Protocol

General protocol of the message is like:

Header+Opcode+Argument1+Argument2+(buffer)
(8bytes) (8bytes) (8bytes) (8bytes)

The value of the header is the final length of the whole string sent to the server, the judgment of the finish of the server's recv-while loop is whether the number of bytes received is equal to the length grabbed from the received string.
Every recv in both client and server is in a while loop. The procedure of the recv is: Grab the first 8 bytes(header) from TCP and store it as length value. After the sum of the received bytes is equal or larger than length, break out the loop.

Example

Let me explain the write function in detail as an example.
The format of the string sent from the client is: **LengthOpcodeFdCount(32 bytes)**
1) Once the server receives this string, grab the length and know that the length of the message is 32 bytes and store it into a buffer.
3) Grab the next 8 bytes as opcode in **select_action** (the opcode of write is 5) and go into **do_write**.
4） Execute write in **do_write** at the server-side and send back errnos and return value.
5） Client gets a message from the server and set errnos and return value.

Zihao ZHOU