System Programming Assignment 4

**Announced: Wednesday December 4, 2013.**

**Due: Wednesday December 18, 2013 18:00.**

**Process and Thread Control**

**Introduction**

In previous homework, we use “select” to improve the throughput of server. In addition, we also enable server handling multiple clients. However, when a client uploads an extreme large file, the server will be blocked by this client for a long period. To solve this problem, you will revise the server to be multiple-threaded so as to avoid the blocking to the clients.

Last, **file\_merger,** which can be either provided executable or your own, to merge the files which are uploaded from multiple clients and are conflicted with each other.

**Requirements**

**1. Server thread pool**

I. Implement a thread pool function to create given number of threads and block newly created threads until dispatch.

II. Add a config “thread=[num]” to server.cfg

III. When receiving a request, server picks a free thread from thread pool and assigns the thread to handle the request. If there are no free threads, server returns “busy” to client. The client shall submit the request again. (Server does not queue the requests.)

**2. Client process**

I. Revise the client process to respond to ‘Busy’ message from server: the client outputs “Server busy” on stdout, sleeps a while and retry after a random delay time.

**3. Apply file\_merger**

I. Apply exclusive file lock when server writes a file.

II. If a thread is blocked by file lock, it forks a child process and executes file\_merger to merge the two file

\* only allow “fork” and “exec\*”

\* use wait to wait file\_merger return and prevent zombie process

\* only allow the current thread in the child process to execute and call file\_merger, via stopping other threads in child process. (Note that you may use pthread\_kill() to send SIGSTOP/SIGCONT to stop/continue other threads in the same process.)

III. Download the file to clients if the file is updated via file\_merger on the server.

**Grade**

**1. Server thread pool**

I. Check whether the number of created thread matches the number specified in config file. (2 pt)

II. Use threads to handle the requests from multiple clients. (3 pts)

III. Client outputs “Server busy” on stdout if there are no available thread on server side.

(1 pt)

**2. Fork (bonus)**

I. Fork a child process to merge conflict file. (2 pts)  
II. Check whether the conflict file on different clients, with same account, are the same as the one on server side. (1 pt)

III. Stop/continue other threads in child process (1 pt)

**Late assignment**: Only the assignments submitted before the deadline will receive the full credit. 5% of your credits will be deducted for every single day delay.

**Notes**: Please use your student id as the title of assignment, and use tgz as the type of compressed file. (e.g., tar cvzf STUDENT\_ID.tgz STUDENT\_ID)

**Contact**

* Email: [box0922480107@gmail.com](mailto:box0922480107@gmail.com)
* Ptt2 : SysProgram, TA account: f771213
* facebook club: <https://www.facebook.com/groups/1430382077184035/>