Multi-threaded C programming

A Crash Course
(condensed into one lecture)

HW3 Question 3

- Modify the client/server source code discussed/presented in class to allow a server to accept and service multiple client connections simultaneously.
- This means:
    - Make the server multi-threaded
    - Do it in C

Threads

- Lightweight processes
  - Think of every thread as a separate, concurrently running program, except...
  - Every thread shares one memory space
    - Means all threads can access same variables
- Threads are the easiest way to let multiple “programs” use the same socket
  - Normally, when you bind to a socket, no other program can use it

Threads in C

- Pthreads
  - POSIX Threads are a standard (originally for UNIX)
  - http://www.llnl.gov/computing/tutorials/workshops/workshop/pthreads/MAIN.html
- This means you're best off using one of the Linux machines for this question

Creating a pthread

- `int pthread_create(pthread_t * thread, pthread_attr_t * attr, void *(*start_routine)(void *), void * arg);`
- `thread` points to space allocated for the thread handle (remember file or socket handles)
- `attr` can be NULL
- `start_routine` is a function
- `arg` is the only argument to start_routine (think `start_routine(arg)`)
- Remember to `malloc thread` and `arg`!
  - And then `free` them when you're done!

Example

Development tools

- Editor: your favorite editor
- Compiler: gcc
- Debugger: gdb
- Confusion reducer: Google

Compiling

- gcc -g server.c -o server -lpthread -Wall
- -g adds debugging symbols. gdb uses these
- -o server says make the binary server
- -lpthread links it to the pthreads library
- -Wall warns you about a lot of stuff

Debugging with gdb

- `gdb program`
- `help` is very useful
  - `help running`
  - `help info`
- Common commands:
  - `b line#` / `b function` - sets breakpoints
  - `run arguments -` starts running the program
  - `n` - goes to the next line of instruction (into subroutines)
  - `cont` - continues until the next breakpoint/stdin read
  - `print expression` – prints the value of something
  - `info threads` – tells you what threads are around
  - `thread thread#` - switches the current threads
  - `list` – shows you source code

Example

Good Luck!