My solution is to use a pool of threads. The initial thread creates the socket and makes the file available in a global scope. Next it spawns the threads each of which will be running the function that listens and handles requests. Finally the main thread also calls this function. The only overhead in this architecture is the socket file descriptor. The actual balancing between threads happens implicitly, multiple threads maybe be waiting in the receive state, but only one will be able to respond to the next incoming packet and during this time the rest of the threads are available to receive and handle the next incoming packets.