Lab 2
Concurrent I/O multiplexing with event-based driver.

We talked in class about distributed systems that are event-based. Now that we have a little background on sockets and multiplexing we get to look and interact with the code of a simple version of an event-based system for an echo server!

Steps:

1. Read in detail section 12.2 of the provided pdf. This talks about the implementation of the event based driver. This is based on the same echo server code in class, but the class code was just with a single socket-based approach.

2. Download the files. You will have to rearrange them since I provide you with every file from the book, and you only need a handful.

3. Compile and run the event-based echo server and client on a Unix/Linux system. Make sure it works. If you are having trouble, it might be that the system you are using has a port blocked by the firewall.

4. Linux-Lab Required. For the next step, I am requiring that you run this in the Linux lab here are the school. The reason is that I want you to run on different systems (one system running the server and on running the echo). Modify the code so that it prints out the name or ip of the server too when it echoes. Run this in the lab and take a screenshot so I can see it working. Moreover, you might want to try it with multiple clients to see if the server can keep up…. This is a fun experiment.

Turn in a zip file. The zip file should contain a folder with you name. The folder should contain all the files you used in the homework. This includes the files that you modified for part 4. You should also add a Makefile that will compile your client and server. Include screenshots of the client and server running (pdf/png/jpeg) that displays the name of the systems. In a textfile (txt), write one paragraph about what you learned in this lab.