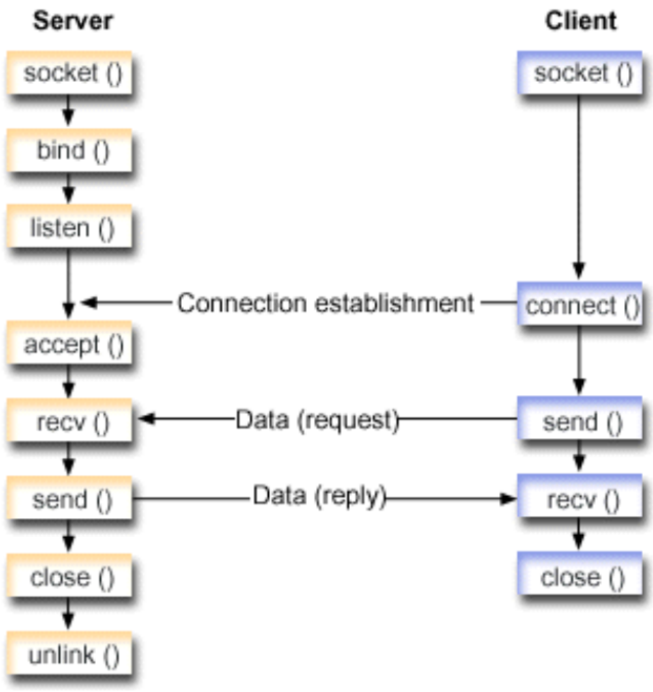
Yu Yin Guan
4/11/2014
CE 156
Section: Wednesday 4-6pm

# Lab #1 Documentation

**Introduction:**

For this first assignment, I followed the diagram shown in the class notes about the design of TCP, I will discuss the protocol later. The only difference in my program is an additional check that I have added to make a network with a server that loops until it is shut down by the client. To do so, the client must enter the -s argument command with a correct password. The client will be able to get the current time off from the server with a request call using the server's IP and port number.

**Protocol:**



For the protocol that I have used in my design, I followed the diagram on the left. I left out the unlink section on the server side because it did not have any files linked to it. There are still some differences between this implementation and mine.

**Client:**

One the client side, it starts off with check argc to see the number of arguments the user have inputed. If it is not 3 or 4, it means it is neither the arguments need for request of time or request for shutdown. In this case, exit_fail the program. Otherwise, go through and initialize variables and array. If the argument is length 4 and the 2nd argument is '-s', then set flag for a shutdown procedure. If this flag is 0 or 1, it will set the send buffer to be "Time?" or to request a password from user that stores in PW and sets send buffer to "Shut!". Socket is then created, the information of the server address is set. The server port number and IP is set in servAddr depending on if the command was a shutdown or time request command, which has different location on argv array. The client then connects using that socket and server information and sends the send buffer which contain "Time?" or "Shut!". Then the client waits for a response from the server. If the response is Password, then it goes through the procedure of sending the previous stored user entered password and wait for another response from server. If the response is “Right”, that means the server has shutdown and prints out a message. If “Wrong”, it prints out message saying so. Then I check if shutdown flag is off, if it is, I will not get the response of “Password” because the send buffer is set to be “Timer”. In this case, I just print out the time I received and close the socket.

**Server:**
The first thing the server does is to check argument to see if it is 2. Then it initializes variable, create socket for listen, bind information on the socket, and listen on that socket. After it received a connection, it will enter an infinite loop where is accepts the connection. After accepting, it checks for the first received argument, if the request was “Time?”, it returns the current time and date back to the client. If the request was a “Shut!”, it sends over the request “Password” and listens for the reply. After receiving the new reply, it checks against a currently stored password. If the password matches, it replies “Right”, exits the loop and closes the sockets. If the password did not match, it replies “Wrong” and continues the loop.

**MakeFile:**
The makefile compiles everything and outputs 2 files named Client and Server and some object files. The Client file is for the client side and the Server file is for the server side. Make clean will clean all the files created from the make command.

**Result:**
The command “`Client Server_IP_address Port_number`” will return “The time now is” with the current time. When the command “`Client -s Server_IP_address Port_number`” is used, it will prompt the user for a password, then returns if the password is wrong or the server has shutdown. Each time the client side runs, it will close itself afterwards and close all sockets used. The server side will remain in loop until it is told to shut down with the right password from the client.