Jordan University of Science & Technology
Department of Network Engineering and Security
NES416- Network Programming
Programming Assignment 4

Due Date: see elearning

Description:

Use UDP socket programming in C to implement a pair of **iterative** client and server programs. The client sends a request to the server (based on a menu item), retrieves the response from the server and print it locally, and waits for more user-input (see below). The server receives the request from the client, performs the requested operation, and sends the result to the client, and waits for more requests from the client side (see below)

Client-Side:

Once the client starts, it asks the user to select an option from a simple menu. Specifically, the client should be able to ask the server to return the current date, the current time, or both the date and time together. For example, if the client sends the character d (or D) , the server replies back with the current date only. If the client sends the character t or (T), the server replies back with the current time only. Finally, if the client sends the character a (or A), the server replies with the current date and time. Note that the connection between the client and server should stay open, so that the client can repeat the operation again until the user asks for termination by entering the appropriate option (s or S). At this point, also the server terminates. For example, if the user enters the characters s, the **server and client** programs terminate.

In order to make sure that you don't face a buffer limitation problem, your client code should get and report the sending and receiving buffer sizes for a UDP and TCP sockets (SO_SNDBUF and SO_RCVBUF). After printing the results on the screen, the program should change the send and receive buffer sizes to some value entered by the user and report the new values again into the screen. This step needs to be done at the beginning before start sending requests to the server. Furthermore, ***please report the kernel behavior of updating the socket options***.

you should display a menu for the client from which a command is chosen as follows:

```
**********welcome to NES 411 HW #1 **********
1- Insert t|T for time
2- Insert d|D for date
3- Insert a|A for date+time
4- Insert s|S to quit
***********************************************
```

The client should use **command line arguments** to read the IP and port number of the server, and leave it for the kernel to assign its local socket address.

Server-side

Once a message is received from the client, the server prints the received message, then the server will perform the requires operations, and send the result back to the client. Then, it waits for more request until it receives s , at which time it exits. To avoid conflict when working on the course server, each student is required to use a port number for the server using the following format: ***55abc*** for its connection, where ***abc*** is the least significant 3 digits of your students ID. For example, a student whose ID is 12345 will have the server listening port = 55345

Submission:

- You files should follow the following naming convention: yourID_HW#
- Submit a zipped file containing only the course code (following the naming convention) and a screen shot of compiling and running your code (both client and server)

Hints/Notes:

- ☐ DO NOT use the header file "unp.h" from the book
- ☐ Each student should use a port number for the server following the rule described above
- ☐ Your program for client needs to take two arguments that specify the IP address of the server and the port that it is trying to connect to. Your program for server needs to take an argument that specifies the port that it is listening to (the same one provided to the client)
- ☐ Ask questions as early as possible.
- ☐ Your programs should be compiled and run without any single error or warning.
- ☐ This is a group assignment, no exception
- ☐ **Comment and error-check you code**