Jordan University of Science & Technology
Department of Network Engineering and Security
NES411- Network Programming
Programming Assignment 4 (CLO2, CLO3, CLO4)
Due Date: See E-Learning

Goal: investigate TCP/UDP socket programming using I/O multiplexing and utilizing DNS service

Description:

In this assignment, you will implement a character count service which would respond to both TCP and UDP requests. When the serve runs, it listens to both the TCP port and the UDP port (it will take service_name as an argument). When a TCP connection is made to the port, the server (actually the child, i.e the TCP server is concurrent) will reply to that connection with how many a given is repeated in a string, and then the client will print out the result and close the connection. When a UDP packet is received on that port, the server itself (i.e the UDP server is iterative) will reply with the appropriate response as stated above. The input in both cases is a string whose first character represents the one that needs to be counted. For example, if the user enters "ahello NES students" , it should count the count of character a in the given string which is zero in this case. If the user enters "Ehello NES students", it should return 1. Note that since you can get a request from both a TCP or UDP clients, you have to use I/O multiplexing using Select at the SERVER side. Please see section 8.15 for more information

To test your server, you will also implement a client that can send either TCP or UDP request. The user is asked/prompted whether to send the request using UDP or TCP. The client will work in a loop until you enter the value -1, at which time the clients and the servers exit. For example, the following menu is displayed to the user once the client starts

```
**********welcome to NES 416 HW #5 *****************
Please select your choice
1- Enter a string to count for a given character using TCP
2- Enter a string to count for a given character using UDP
3- Exit (by entering -1)
************************************************
```

Note that your client can accept/receive input from multiple sources, so you need to use I/O multiplexing for enhanced performance.

Your client should take hostname and port number as command arguments. However, you need to use server name in your code rather that its IP address. That is, you need to edit the `/etc/hosts` file in your machine and add a line that specifies the server name as ***nes416_server*** and assign it the IP address of your machine. In your code you should use the appropriate functions to convert between hostnames and IP addresses. Furthermore, you need to use the service name in the command line rather than the port number. Edit the appropriate file(s) and use the ***count_service*** as your service name.

Submission:

- You files should follow the following naming convention: your_group#_HW#, and contains your ID's as comments at the beginning of the code

- 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:

- ☐ Don't use the file "unp.h"
- ☐ Submit your source code using the link of the elearning site
- ☐ Your programs should be compiled and run without any single error or warning. TA will run your code assuming **Ubuntu-16.04.3**
- ☐ Your program for client needs to take two arguments that specify the server's hostname and the service name that it is trying to connect to. Your program for server needs to take an argument that specifies the service namet that it is listening to (the same one provided to the client)
- ☐ Test your code by running the two clients simultaneously.
- ☐ **Comment and error-check you code**
- ☐ **Make sure zombies and interrupted system calls are handled correctly**
- ☐ **Make sure you have valid input before sending the data to the server**
- ☐ **Make sure necessary socket options are enabled**
- ☐ Ask questions as early as possible.
- ☐ Your programs should be compiled and run without any single error or warning.
- ☐ Feel free to use any port in this assignment (ofcourse, above 1024)