Networks Assignment 8 Report

-Suchintan Pati (18CS10064), Dudyala Vamshidhar Reddy (18CS30017)

Documentation:

Taken help from: http://beej.us/guide/bgnet/

At first, the user name entered in the terminal is parsed and a listening socket is bound. The listening socket is added to the 'master', which contains all the read file descriptors. 'fdmax' stores the maximum file descriptor. This is followed by two nested for loops. The outer loop is an infinite loop, where the read_fds is updated (from "master") and select() function is called. The select function times out after 60 sec. This function returns when a particular file descriptor in read_fds is ready to be read from. The inner for loop iterates over all the read_fds and checks if that particular file descriptor is ready. Three types of file descriptors are present: listener, stdin and sockets from existing connections. The three cases are handled:

1. Listener ready: This means a new connection is trying to establish connection. So the server accepts (using accept() call) the connection and adds the socket to "master".
2. stdin ready: The user has entered a message to be delivered to a recipient user and is read by read() call. The message is parsed and the username of the sending user is appended at the end of the message. These are the two cases depending on recipient user:
    a. A connection exists with recipient user i.e. "user[u] ==1": Using the info stored in the user_fd array, the message is sent (using send()) to the correct socket.
    b. No connection exists with recipient user i.e. "user[u] ==0": A new connection is attempted to be established (using connect() call) with the recipient user, followed by sending the message using send() call. This socket is added to the "master" and user and user_fd arrays are updated.
3. Socket from existing connection ready: The message is read using recv() call and is displayed on stdout after being parsed.

Compilation Procedure:

In order to compile, run: gcc -o chat chat.c , or just run: make.

Running Procedure:

The user_info table is as follows:

User Name	IP Address	Port no.
0	localhost	100
1	localhost	101
2	localhost	102
3	localhost	103
4	localhost	104

Note that above user_info table is hardcoded, i.e, one needn't specify the port no. and IP address in the terminal. One needs to only give user name (i.e. 0,1,2,3,4) in the terminal. Format to execute:" sudo ./chat <user name> ". One can open up to 5 terminals and enter this command with different usernames. Then, in order to send a message to a user, enter in the following format : " <recipient user name>/<msg> ". The intended user will receive the message on its terminal in the format: "<sender user name> : <msg>".

Sample Input and Output:

Teminal 1:

sudo ./chat 1

2/How are you doing?

NEW Connection to User 2

3/Hello, did you do networks assignment?

NEW Connection to User 3

User 3: No, I am yet to start :(

User 2: Fine, wbu ?

2/ I am good.

3/ Bro I need some help in networks assignment.

User 2: My partner is yet to start :(

Terminal 2:

sudo ./chat 2

selectserver: new connection from 127.0.0.1 on socket 4

User 1: How are you doing?

1/Fine, wbu ?

selectserver: new connection from 127.0.0.1 on socket 5

User 3: Bro I have completed the networks assignment. Lets get started on os assignment now.

User 1: I am good.

3/Ok Cool!

User 1: Bro I need some help in networks assignment.

1/ My partner is yet to start :(

Terminal 3:

sudo ./chat 3

selectserver: new connection from 127.0.0.1 on socket 4

User 1: Hello, did you do networks assignment?

1/No, I am yet to start :(

2/ Bro I have completed the networks assignment. Lets get started on os assignment now.

NEW Connection to User 2

User 2: Ok Cool!