1 Description

In this assignment you will develop a UDP client. The client will communicate with a server we provide to you.

2 Protocol

This section describes the protocol your UDP client should follow:

1. HELLO: The hello message is sent from the client to the server (denoted by client → server)
   The HELLO messages have exactly two fields. Your username and a random integer number. The fields in the message are separated by ‘#’. An example of a HELLO message might be lex#87. The message is sent to the server as string; use `sprintf` to create the HELLO message. Your username field must not be longer than 20 characters. The random integer $i \in [-127, 127]$. The server ignores any HELLO message that does not follow these specifications.

2. RESPONSE (Server → Client): The RESPONSE message has exactly two fields. Your username and a binary number. The two fields are separated by ‘#’. Your username will be reversed thus, if you have sent ‘lex’ the correct answer is ‘xel’. The binary number is the 2’s complement of the number you sent. An example of a correct RESPONSE message for the HELLO lex#87 is xel#10101001. The 2’s complement is always encoded as a 8 bit string of zeros and ones.

Your client should always check the correctness of the server’s response. You should:

1. Check if the first field you received is in fact your reversed username. If it’s not, you should send a new HELLO message to the server.

2. Check if the 2’s complement you received from the server is valid. If it is invalid you should send a new hello message to the server.

3. When you send new HELLO messages you should generate a new random integer inside the pre-specified space.

4. For every request you make, your program should print the following:

   USERNAME:<u> $ INT:<i> $ RESULT:<outcome> $ REASON:<r>

   • ‘u’: username you sent
   • ‘i’: integer you sent in the hello message
• ‘outcome’: success or failure; you must use only these words.
• ‘r’: if the outcome is failure you should specify the reason. The reason can be either username or complement (always in lowercase). In the case of success you don’t need to specify the reason.

2.1 Remarks

1. You must use the C programming language.
2. You must submit code that compiles, otherwise your assignment will not be graded.
3. Your code must be -Wall clean on linux-lab’s gcc otherwise your assignment will not be graded.
4. You must submit a tar.gz file that contains a directory with your solution.
5. The name of the directory that includes your solution should follow the format: <username>_assignment0. The username should be the same as it appears on the grades server (grades.cs.umd.edu).
6. The filename of your solution should be <username>_client.c.
7. You must provide a Makefile.
8. The TA will post information about the IP address and the listening port of the server.
9. You are not allowed to work in teams or to copy code any source.