Distributed Programming I

Solution of laboratory 2

Exercise 2.1 (perseverant UDP client)

Use the same approach of the previous exercise, by inserting send tries in a proper loop.

Exercise 2.2 (limiting UDP server)

Extend the UDP server developed for the exercise 1.3 by memorizing, for every received message, the source address in a 10 elements vector that simulates a circular buffer. To store the address use a vector of sockaddr_storage structures in order to simplify portability to IPv6 (that can be implemented as a useful extra exercise)

Exercise 2.3 (iterative TCP server)

Use Socket(), Bind() and Listen() functions. Put the Accept() function in an infinite loop in order to be able to (sequentially) handle any client requests. After the Accept(), an inner infinite loop must receive the client commands and, for each command, sent the requested file or an error message. This second infinite loop must be interrupted when the client sends the QUIT command (connection must be closed by using the Close() command on Server side), or when it is not possible anymore to use the socket. In this last case print a message stating that the connection has been closed by the client, then use the Close() command to the Server side. Pay attention not to use the (buffered) Readline() function to read client commands, but use the unbuffered version of Readline or create your own reading loop able to scan the input character by character, until the '\n' is read: this is because Readline() buffers input, which may cause problems to the next Read() or Recv() calls. Pay attention to server robustness by avoiding to terminate the server in case of error (do not use err_quit() either).

Exercise 2.4 (XDR standard data)

Proceed as in the development of a regular TCP client. Create an XDR buffer through xdr_create(), insert numbers into the buffer through xdr_int(), then get the amount of bytes that needs to be sent through xdr_getpos(). When receiving data, use the same procedure: initialize the buffer through xdr_create() and then call as many xdr_int() as necessary.