Networked Systems Programming (A.A. 2023/24)

Lab n. 3 – October 30, 2023

Prof. Eugenio Zimeo

Exercise 3.1

Write an application to implement a peer-to-peer chat between two processes.
An application for chatting is intrinsically peer-to-peer (each process can send and receive sequences of characters). The stream-oriented communication channel built atop TCP is full-duplex: communication is bidirectional. On the other hand, the socket API provides some functions to implement the client/server model: a process works with the role of server to listen client connections; the other one starts the interaction. It is worth noting that these two roles characterize the two channel endpoints (sockets) only for building the communication channel. When the channel is built, the two communicating processes can assume different roles during the conversation: when a process (A) sends a message, the other process (B) should be ready to read data and vice versa. Since read() is a blocking function, a half-duplex protocol could be useful to perform bidirectional write/read interactions by suspending one process at a time. According to the half-duplex protocol, while a process writes on the network channel the other ones reads. A state variable and a Finite State Machine can be used to coordinate behavior changes.

Exercise 3.2

Change the chat application implemented with exercise 3.1 to fully exploit the TCP channel for simultaneous bidirectional communication. To this end, we need to overcome the limitations due to the blocking behavior of some sockets functions (e.g. accept(), read(), etc.). I/O based on the select() function is a possible solution. By exploiting this function, differently from exercise 3.1, the state at each iteration will no longer be determined by the special character '-' used to pass the control but will be inferred by the monitored events.

Exercise 3.3 (optional, +0,5 credits)

Write an application to implement the Tic-Tac-Toe game. The game is based on turns between two players who at each turn place a pawn on a location of a 3 x 3 board. The first player that puts three pawns horizontally, vertically or on a diagonal o the board is the winner. Implement the application by using the stream communication model based on TCP sockets. If you use two processes for the game, their behavior should be symmetric, since for each turn they should be able to perform the same actions. However, by using stream-oriented communication sockets, the initial interaction is asymmetric: one player will run a process as server while the other one will run a process as client. When the connection is established, the channel can be used in symmetric way.

Upload the exercises of this lab into your shared folder before November 2nd, 2023.
Upload also the Wireshark capture files and a simple document with screenshots to show the content of the messages.