MyFileTransferProtocol

Vlaișan Maria Flavia

1 Introducere

Pe lângă faptul că realizez un proiect de la zero, am fost determinată să aleg MyFileTransferProtocol și din dorința de a acumula informații noi, proaspete pe care, probabil, le credeam imposibil de implementat înainte. Autentificarea pe baza unei parole și a unui nume de utilizator, tranferul de fișiere între client si server, implementarea unui mecanism de autorizare whitelist/blacklist și a unui mecanism de transmitere securizată a parolei la autentificare vor fi un start pentru mine în ceea ce privește scrierea de cod.

2 Tehnologii utilizate

2.1 TCP

Proiectul pe care îl voi realiza va fi o aplicație pe modelul client-server, având la bază protocolul TCP. Folosirea TCP-ul îmi va asigura un tranfer garantat, sigur și corect al fișierelor. Totodată în cadrul acestui protocol este nevoie de recunoașterea datelor și are capacitatea de a retransmite, în cazul în care utilizatorul cere. În ceea ce privește comunicarea între procese se vor folosi socket-uri, iar limbajul folosit este C.

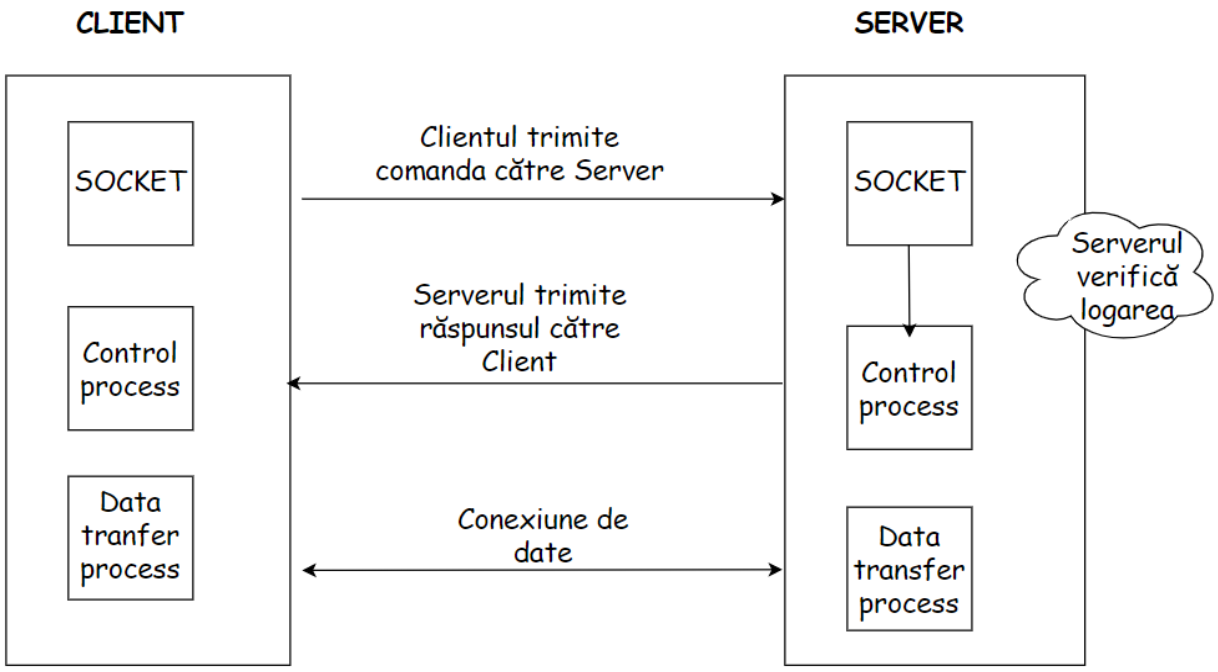
3 Arhitectura aplicației

3.1 Concepte implicate

Baza arhitecturii aplicației este susținută de această comunicare dintre client și server. Clientul va trimite comenzi către server, iar acesta le va procesa și va trimite înapoi către server un răspuns.

Așa cum am mentionat și mai sus, comunicarea client-server se va face cu ajutorul socket-rilor prin funcțiile **read** și **write**. Pentru fiecare client nou conectat se va crea un proces-copil din procesul principal. Conexiunea se poate realiza doar pe bază de logare cu succes.

3.2 Diagrama aplicației



4 Detalii de implementare

4.1 Scenarii de utilizare

CLIENTUL: Clientul va fi utilizatorul, adică noi cei care încercăm funcționalitatea aplicației. Utilizatorul va putea folosi procesul de transfer de fișiere doar după autentificare (sub forma unui cont de utilizator). Acest lucru se poate efectua prin introducerea în linia de comandă a comenzii "login". După acest pas, clientului i se va cere introducerea unei noi comenzi de tipul "username". Username-ul este verificat, iar dacă acesta are drept de acces (adică se află în whitelist, nu în blacklist) i se va cere și parola. Parola se va trimite securizat, prin criptare, la introducerea acesteia în linia de comandă, caracterele obișnuite sunt înlocuite cu "*". Dacă logarea s-a realizat cu succes, procesul este pregătit de tranfer de fișiere.

SERVERUL: Serverul va fi cel care procesează toate comenzile primite de la client. Presupunând că logarea s-a realizat cu succes, el va citi din socket operațiile pe care trebuie să le execute (ex: remove,create,insert) asupra fișierelor. În cazul în care comenzile transmise de client sunt scrise greșit sau au alte greșeli, serverul va semna aceste erori, trimițând posibile rezolvări. Aceste raspunsuri ale serverului vor fi scrise in socket-ul clientul si citite de către utilizator.

RESURSE: Avem trei fișiere pe care le vom folosi în proiect: whitelist.txt, blacklist.txt și login.txt.
whitelist.txt = conține numele utilizatorilor care au acces de execuție, care pot trimite comenzi către server
black.txt = conține numele utilizatorilor care NU au acces de execuție și NU pot trimite comenzi către server. Dacă un astfel de client încearcă să se conecteze, va fi atenționat de către server. La trei astfel de încercări, acel utilizator va fi șters definitiv, fără a mai putea vreodata să se conecteze. **login.txt** = numele si parola ("username" - "parola") tuturor utilizatorilor care un cont în aplicație/care s-au mai logat înainte

5 Concluzii

Aplicația ar putea funcționa mult mai simplu și ușor dacă utilizatorii care își fac cont ar fi introduși într-o bază de date.

6 Bibliografie

1 springer.com
2 TCP