

MyFileTransferProtocol

Trifan Nicoleta-Tatiana, grupa B3

Universitatea Alexandru Ioan Cuza din Iași

1 Introducere

În cadrul proiectului am avut de implementat o aplicație client/server care asigură comunicarea și transmiterea fișierelor între aceștia. De asemenea, serverul (concurrent) pune la dispoziție câteva opțiuni pentru clienți de operare cu fișiere, ca de exemplu : vizualizare, încărcare, descărcare și ștergere. Înainte de a putea fi executată vreo operație asupra fișierelor, clientul trebuie să se logheze cu un nume de utilizator și parolă, iar în cazul în care acesta se află în *blacklist* – logarea nu este reușită, altfel (*whitelist*) – utilizatorul este logat și poate opera cu directoare/ fișiere. De asemenea e necesar realizarea unui mecanism de transmitere a parolei cât mai securizat; un mecanism care ar permite păstrarea și transmiterea parolelor cu intenția de a asigura securitatea datelor utilizatorilor.

2 Tehnologii utilizate

2.1 TCP

Ca protocol de transmitere am ales TCP/IP, deoarece acesta asigură atât securitatea datelor în cadrul comunicării, cât și integritatea informației. Față de UDP, TCP mai întâi stabilește conexiunea cu receiver-ul înainte de a transmite pachetele de date, astfel este garantat faptul ca informația va ajunge la destinatar. La fel, față de UDP, primirea pachetelor de date are loc în aceeași ordine în care au fost trimise, iar pe lângă această ordonare mai este asigurat și controlul fluxului de informație. [2]

2.2 Sqlite3

Pentru proiectul meu am ales implementarea unei baze de date cu ajutorul sqlite3, deoarece îmi asigură portabilitatea- ceea ce e o caracteristică importantă pentru aplicație. La fel nu necesită multe configurări pentru utilizare. Am un tabel utilizatori cu următoarele atribute: ID, USERNAME, PASSWORD, LIST. ID – este un identificator unic pentru fiecare utilizator; câmpurile USERNAME și PASSWORD conțin datele cu care se loghează utilizatorul, iar câmpul LIST – înregistrează informații despre statutul clientului *whitelist/blacklist*.

2.3 Mecanism securizat de transmitere a datelor

Mecanismul de transmitere securizată a datelor are intenția de a stoca datele despre utilizatori într-un mod cât mai securizat. După cum am menționat în subsecțiunea anterioară câmpul PASSWORD, din baza de date inclusă în proiect, trebuie să conțină date despre parola utilizatorului. După mai multe documentări, am decis să implementez o funcție de criptare care să codifice parolele pentru de baza de date. Codificarea are loc în felul urmator – în baza de date sunt memorate caracterele pentru fiecare parola de forma: codul ASCII – 1.

Utilizatorul când se loghează își scrie username-ul și parola în format normal, după care la nivel de aplicație intervine funcția de criptare care criptează parola înregistrată de utilizator, în cazul în care aceasta coincide cu cea din baza de date, iar utilizatorul satisface și ceilalți parametri(LIST) logarea este reușită, altfel – nu.

2.4 Whitelist/Blacklist

Acest câmp al bazei de date presupune asignarea unui statut pentru fiecare client de *white* ori *black*, care indică dacă acesta se poate sau nu loga la server. Chiar dacă clientul își indica parola corectă, dacă acesta are un statut de *black* nu îi reușește logarea, deoarece se prezintă a fi un client restricționat de server .

2.5 FTP

Schimbarea de date prezintă o mare importanță în prezent. Protocolul FTP destinat aplicațiilor server/client se caracterizează prin conectarea clientului/clientilor la un server cu posibilitatea de a *încărca*, *descărca*, *șterge* și *vizualiza* fișierele de pe un server. Clientul după ce se conectează cu succes poate realiza una din opțiunile enumerate anterior. Serverul este conectat la un *server file system* - care prezintă un sistem de fișiere la care au acces clienții și cu care pot opera, la fel clienții odată cu conectarea li se creează un dosar personal, iar în cazul în care acesta există se accesează și e posibilă operarea cu fișierele din interiorul acestuia. [1]

3 Arhitectura aplicatiei

În cadrul diagramei de mai jos am încercat să prezint cât mai detaliat structura proiectului. Observam că doar serverul are acces la baza de date cu informații despre clienți (nume utilizatori, parole etc.) și este conectat la sistemul de fișiere pe care îl distribuie și clientilor. Clienții pentru a putea realiza anumite operațiuni asupra fișierelor(de a *încărca*, *descărca*, *șterge*, *redenumi* și *vizualiza*) au nevoie de logare, deci în comunicarea client/server clientul se poate loga, ulterior să aleaga ce vrea să execute cu fișierele la care are acces. Fiecare client e conectat la propriul spațiu de memorie.

Observam că serverul concurent permite conectarea mai multor clienți concomitent și oferirea unui răspuns instant relativ la cererea clientului.

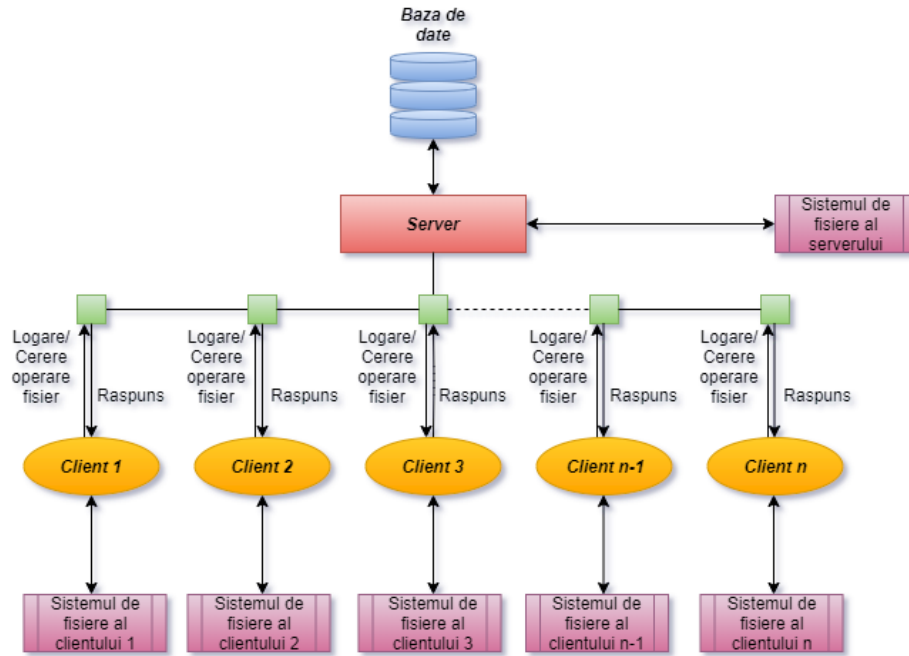


Fig. 1. Diagrama proiectului MyFileTransferProtocol

4 Detalii de implementare

4.1 Thread

Aplicația client/server pe care am implementat-o în primul rând trebuia să asigure concurența în timpul utilizării, acest lucru l-am implementat cu ajutorul threadurilor. Acestea garantează paralelismul, prin repartizarea a câte unui thread fiecărui client conectat, astfel ceea ce face fiecare client prezintă un proces independent de ceilalți fără a interfera. Implicit asigură și o eficiență temporală prin răspunsuri instantanee la cererile clienților. [3]

4.2 Scenarii de utilizare

Logarea am introdus-o într-o buclă, astfel încât opțiunea de logare se afișează la ecran până nu este introdus username-ul și parola corectă.

Operare cu fișiere - odată clientul fiind conectat acestuia i se afișează meniul pentru a alege una din opțiuni. Indiferent dacă se conectează 1 sau mai mulți clienți, aceștia ușor pot opera cu fișierele atât din dosarul personal cât și din spațiul de memorie distribuit de server. Niciunul dintre clienți nu poate accesa dosarul personal al altui client. Orice opțiune aleasă de client este trimisă la server, iar în cazul în care se solicită vreun răspuns de la server acesta îl transmite.

Posibilitatea de a tasta opțiuni e accesibilă până în momentul în care clientul decide să se deconecteze.

Meniul care îi este afișat clientului este prezentat mai jos :

Alegeti urmatoarea optiune pe care doriti sa o executati:

Vizualizeaza fisierele/directoarele (1)

Descarca fisier/director (2)

Sterge fisier/director (3)

Incarca fisier/director (4)

Deconectare (5)

Pentru a putea executa una din optiunile de mai sus, tastati numarul corespunzator din paranteza

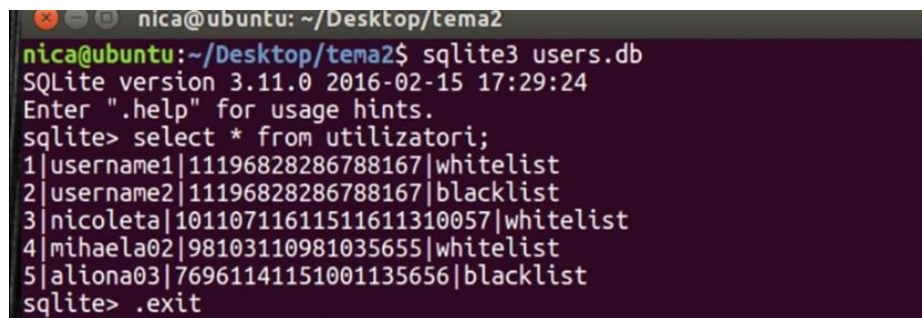
4.3 Reguli de utilizare

Dacă clientul alege opțiunea de descărcare acesta trebuie să scrie fișierele în felul următor : *server_file_system/nume_fișier*.

Dacă acesta alege orice altă opțiune ștergere/încărcare poate scrie doar numele fișierului cu care dorește să opereze.

4.4 Baza de date

Un model de tabel **utilizatori** descris anterior. Unde prima coloană este ID, a doua - USERNAME, ulterior fiind PASSWORD și LIST. Observăm mai jos că parola este deja criptată, conform algoritmului descris la secțiunea **2.3**.



```
nica@ubuntu: ~/Desktop/tema2
nica@ubuntu:~/Desktop/tema2$ sqlite3 users.db
SQLite version 3.11.0 2016-02-15 17:29:24
Enter ".help" for usage hints.
sqlite> select * from utilizatori;
1|username1|11196828286788167|whitelist
2|username2|11196828286788167|blacklist
3|nicoleta|10110711611511611310057|whitelist
4|mihaela02|98103110981035655|whitelist
5|aliona03|76961141151001135656|blacklist
sqlite> .exit
```

Fig. 2. Model baza de date pentru clienți

5 Concluzii

Proiectul poate fi îmbunătățit prin adăugarea unei interfețe grafice pentru un aspect mai user-friendly. La fel ar putea fi analizat mai atent cazul în care doi clienți în același moment vor să opereze cu același fișier.

Bibliografie

1. FTP
<https://searchnetworking.techtarget.com/definition/File-Transfer-Protocol-FTP>
2. TCP
<https://www.privateinternetaccess.com/blog/2018/12/tcp-vs-udp-understanding-the-difference/>
3. Documentație cod
<https://profs.info.uaic.ro/~computernetworks/files/NetEx/S12/ServerConcThread/servTcpConcTh2.c>
<https://www.includehelp.com/c-programs/find-size-of-file.aspx>
<https://www.includehelp.com/c-programs/find-size-of-file.aspx>
<https://github.com/omair18/Socket-Programming-in-C>