Universitatea „Alexandru Ioan Cuza”
Facultatea de Informatiă

Rețele de Calculatoare

MySSH

Autor:
Milea Mihai-Cristian I2A4

Coordonator:
Colab. Stănescu Ioana

Keywords - SSH · TCP · IP · client-server · securitate · criptare · socket · fork

16 ianuarie 2018

Cuprins

1 Introducere

SSH = Secure Socket Shell
Protocoalele SSH1 și SSH-1 au fost developate in 1995 de către Tatu Ylönen, un cercetător de la universitatea de tehnologie in Helsinki, Finlanda.
Modelul său SSH-1 a început să capete din ce in ce mai multă atenție. Cu timpul și-a dat seama că produsul său poate fi utilizat în multe moduri.
In Iulie 1995, SSH-1 a fost disponibil ca software gratuit cu tot cu cod sursă, ca să poate fi copiat și utilizat de oricine și-ar fi dorit.
Până la finalul anului 20.000 de utilizatori din 50 de țări diferite au adoptat SSH-1. Ylönen primea câte 150 de mesaje pe e-mail de la utilizatori cerând suport.
Cu timpul a fost înbunătățit. În acest proiect sunt implementate aceleași idei pe care le-a avut si Ylönen. Scopul este realizarea comunicării mai multor utilizatori cu un server.
Exemplu: PuTTY
Conținutul lucrarii prezintă schimbul de informații criptate între server și utilizatori.

Acest proiect funcționează pe baza protocolului TCP/IP prin care un utilizator se conectează inițial la un server cu credențialele sale de mySSH, având ulterior posibilitatea de a face uz de comenzile pe care le are la dispoziție (date de server).

2 Despre mySSH

Modelul implementat în acest proiect are ca scop interpretarea comenzilor de către server și execuția comenzilor cerute de către utilizatori. Utilizatorii sunt identificați printr-un fișier cu conținut criptat. Utilizatorii pot trimite comenzi către server iar acesta va trimite înapoi execuția.

Cerință: Sa se implementeze o pereche client/server capabila de autentificare si comunicare encriptate. Server-ul va executa comenzile de la client, si va returna output-ul lor clientului. Comenzile sunt executabile din path, cu oricat de multe argumente; cd si pwd vor functiona normal. Se pot executa comenzi multiple legate intre ele sau redirectate prin: |. >, <, 2>, &&, ||, ;.
Această cerință induce la crearea unui server iterativ. Metoda abordată în acest proiect este următoarea: rezervarea a câte un proces pentru fiecare utilizator.

3 Tehnologii utilizate

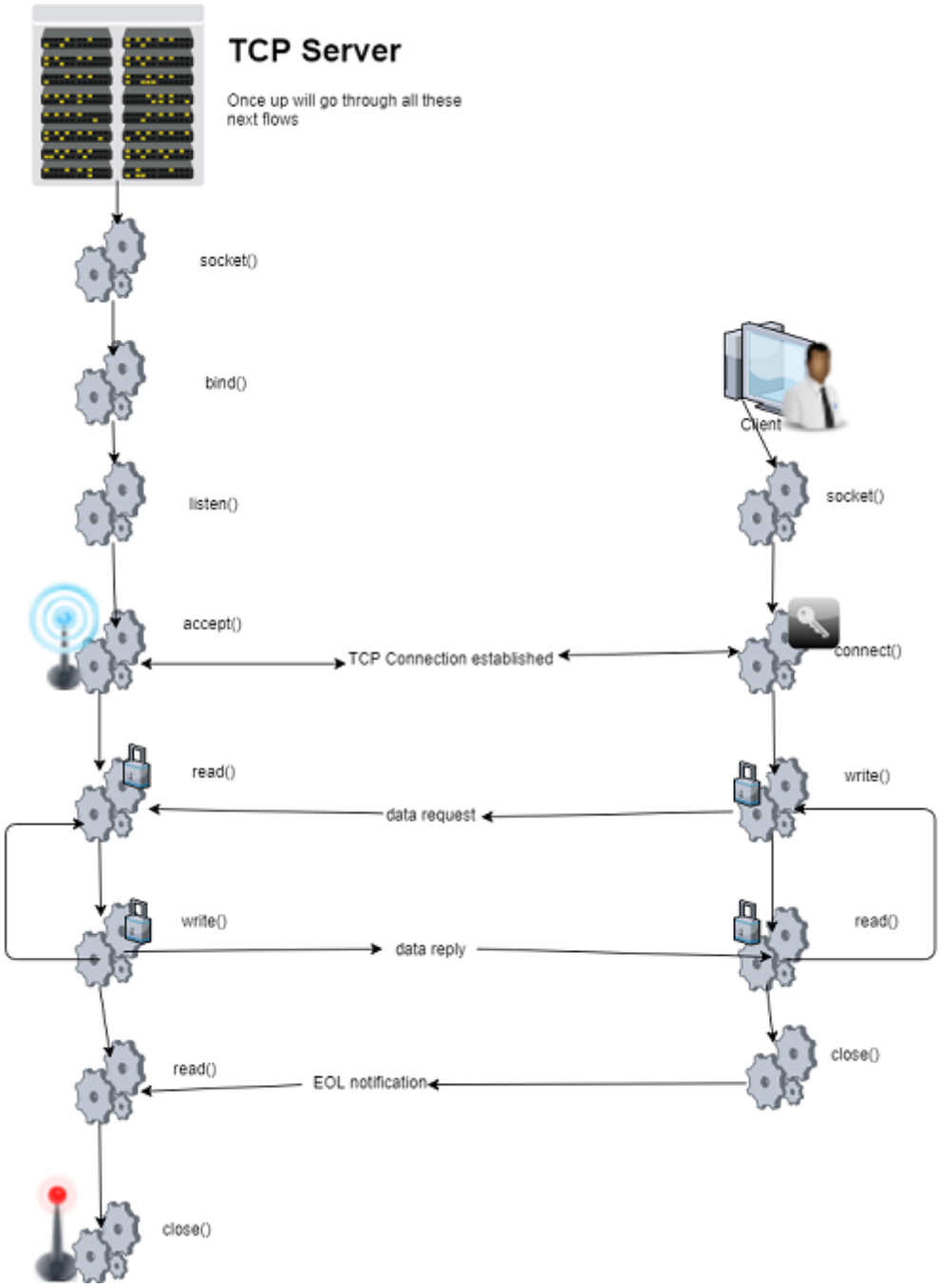
Pentru realizarea acestui proiect au fost utilizate protocolul TCP/IP pentru stabilirea conexiunii între client și server.

3.1 TCP/IP

Modelul TCP/IP este cel mai utilizat model pentru comunicarea dintre calculatoare. Denumirea sa provine de la cele două protocoale fundamentale utilizate: TCP (Transmission Control Protocol) şi IP (Internet Protocol).

Protocolul TCP este un protocol orientat-conexiune sigur ce permite transmiterea unui flux de octeți de la o mașina la alta fara erori, astfel vor exista pierderi de informație. Pentru fiecare pachet IP trimis, se așteaptă o confirmare că pachetul a fost trimis. Dacă dupa un anumit interval de timp acest răspuns nu apare, atunci pachetul va fi retrimis.

Deasemenea, la recepţionare se verifică integritatea datelor, în cazul în care datele au fost afectate se cere retransmiterea mesajului. Unele pachete pot ajunge înaintea altora și pot avea alte topologii de a-și face drum la destinație.


TCP Server
Once up will go through all these next flows
socket()
bind()
listen()
accept()
read()
write()
read()
close()
Client
socket()
connect()
write()
read()
close()
TCP Connection established
data request
data reply
EOL notification

4 Arhitectura Aplicatiei

Proiectul este împărțit în doua aplicații ce pot fi configurate independet. De pilda, se pot modifica proturile pe care se realizează comunicarea între server si clienți. De menționat este că numărul de utilizatori pe care îî poate susține aplicația server depinde de capacitatea mașinii. De asemenea, numarul maxim de utilizatori din fișierul de utilizatori este de 64, și el modificabil.

Odata ce serverul va porni, se va alfa intr-o stare blocantă până la stabilirea conexiunii cu un client. Această conexiune se realizează într-un proces copil. După care procesul rădăcina va reintra iar in aceeași stare. Transferul de date se face cu o cheie de criptare privată. Dacă comanda introdusă de client este diferită, va fi tratată într-un mod diferit. Pentru manevrarea mai ușoară a comenzilor primite de către clienți, odată ce au drepturi, ele sunt sparte în cuvinte, utilizate ulterior. Un utilizator are la dispoziție 3 greșeli de autentificare. Dacă datele introduse sunt greșite, la a 3-a incercare nereușită se va încheia legatura dintre server și client.

4.1 Client

Prima parte a acestui proiect este aplicația Client. Aceasta îi va permite utilizatorului să se conecteze la server printr-un IP si un port, inserate de către utilizator. Utilizatorul se va afla in 3 etape de-a lungul utilizării aplicației Client din perspectiva aplicației Server.

Pasul I - Introducerea username-ului. Dacă va fi găsit se va trece la pasul II, daca nu, se repetă pasul I;

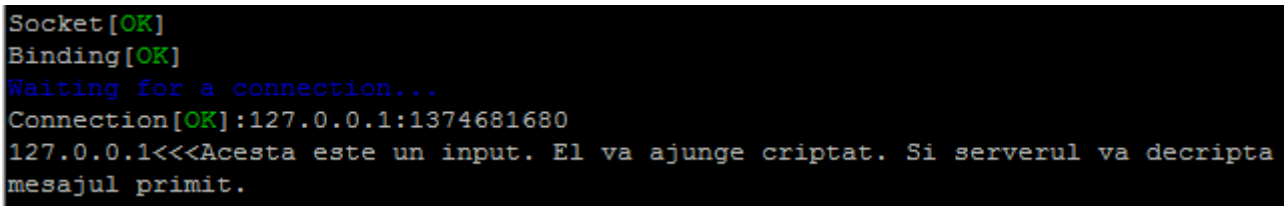
Pasul II - Introducerea parolei username-ului. Daca parola introdusă coincide cu cea a username-ului atunci se va trece la pasul III, daca nu, se repetă pasul II;

Pasul III - Clientul poate utiliza toate funcțiile pe care i le pune la dispoziție serverul;

Printre caracteristicile aplicației Client se numară:

- linie de comandă;
- utilizarea comenzilor specifice;

O listă a acțiunilor ce pot fi executate de către utilizator in cadrul serverului u o scurtă descriere, este reprezentată în tabelul de mai jos



Comamndă	*Descriere*
help	Afișează o listă a comenzilor.
cd	Modifica directorul pe care se afla utilizatorul.
disconnect	Încheie conexiunea cu serverul.
dir	Afișează conținutul folderului aflat pe server.
pwd	Afișează path-ul curent.
mv	Mutarea unui fișier dintr-un loc in altul.
mkdir	Crează un director nou.
rmdir	Ștergerea unui folder.
rm	Ștergerea unui fișier aflat pe server.
ls	Afișează o listă a fișierelor și a folderelor aflate în folderul curent.
wget	Descarcă un fișier.
echo	afișează o secvență de caractere.
who	Listează utilizatorii conectați, de când sunt conectați si adresele fiecăruia.
mcedit	Permite editarea unui fișier cu editorul McEdit.
nslookup	Afișează informatii despre un hostname.
pwd	Afișează folderul curent de lucru de pe server.
touch	Creează un fișier.
zip/unzip	Arhivează sau dezarhivează un set de fișiere respectiv o arhivă.
chmod	Modifică atributele unui fișier (drepturi).
cp	Copiază un fișier intr-o altă locație.

Serverul acceptă și comenzi înlănțuite: ls -a >intro.txt

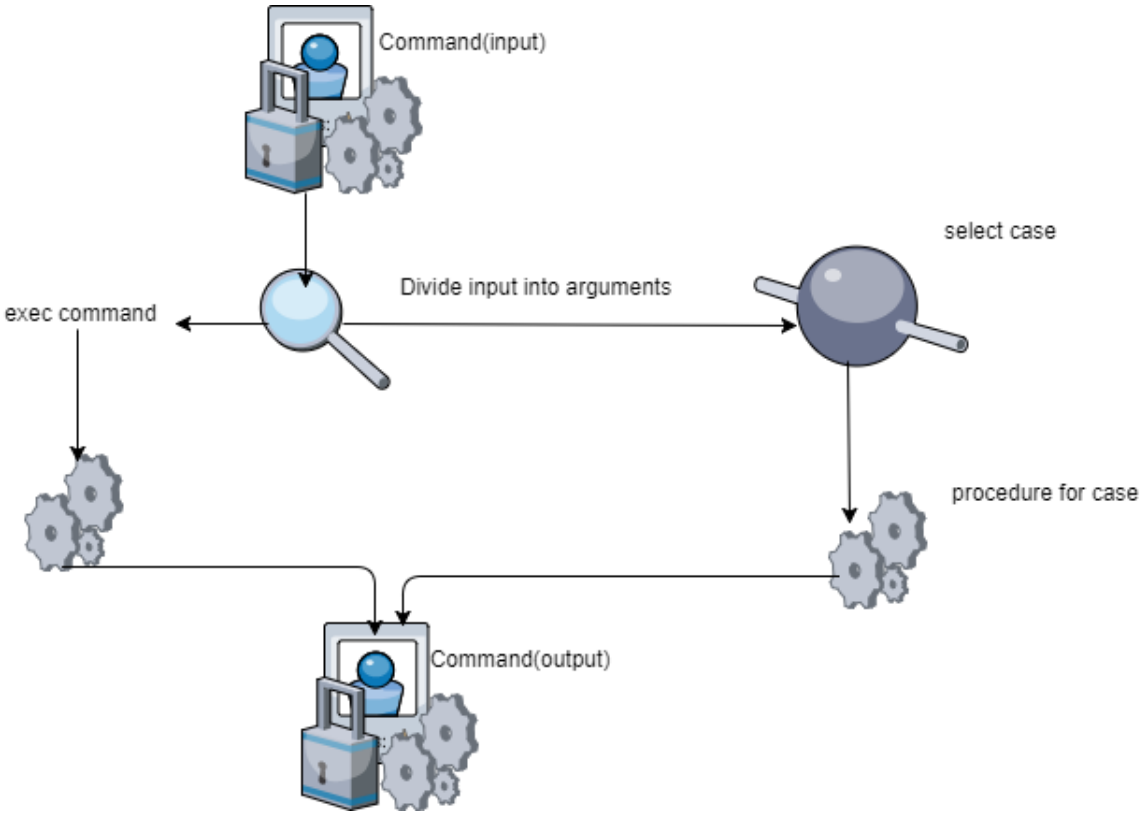
4.2 Server

A doua parte a acestui proiect constă în aplicația Server, care se va afla pe calculatorul ce se dorește a fi utilizat drept gazdă pentru Clienți. Câteva din funcțiile pe care le dispune această aplicație:

- criptarea traficului dintre client și server;
- implementarea comenzilor ce permit administrarea serverului de la distanță;
- filtrarea IP-urilor, permite verificarea adresei IP de la care se conectează utilizatorul și stabilește dacă aceasta este una autorizată sau nu;
- implementarea unui sistem de autorizare de tipul whitelist\blacklist pentru conturile utilizatorilor.

4.3 Stabilirea Conexiunii

În continuare vom trage o ochiadă asupra conexiunii ce urmează să se stabilească între calculatorul nostru - *Clientul* - și calculatorul gazdă, cu care dorim să comunicăm - *Serverul*.



În momentul în care s-a realizat conexiunea clientul este pus in pasul I. Conexiunea se va face printr-un port prestabilit pe portul 6636, iar daca se dorește se poate face și pe alt port.

4.4 Procesarea Comenzilor

```
1  struct mySSH_user{
2  char username[20];  //numele utilizatorului
3  char passwd[20];    //parola utilizatorului
4  int id;             //id-ul utilizatorului
5  };
6
```

Informațiile expuse mai sus sunt salvate si păstrate criptate. De exemplu

Procesarea comenzilor se va realiza atât de partea aplicația client cât și de aplicația server, ambele părți efectuând acțiunile aferente comenzii inițiate. Unele comenzi trebuiesc tratate în mai multe runde. De exemplu dacă se dorește editarea unui fișier acesta va fi salvat local, editat local și încărcat inapoi in calculatorul gazdă. Altele vor fi tratate intr-o singura rundă. De exemplu afișarea fișierelor din directorul curent. Fiecare utilizator va fi stocat in server cu informațiile acestuia.

4.5 Securitate

Orice transfer de date între server si client se face protejat. Pentru că se bazează pe clienți, există vulnerabilitate. Un exemplu ar fi un SMBbomb, unde serverul va primi nenumărate conexiuni până nu mai face față.
Unele din atacurile pe care le poate suferi serverul:

- Atacurile de tip Brute force
- Captura de pachete
- Atac de tip Spoofing

O metodă bonus pentru a securiza transferul de date ar fi:

- Utilizarea unei rețele private (VPN).

5 Server-core

Serverul va primi de la client o comandă și serverul trebuie sa interpreteze comanda și să îi returneze ce va afișa execuția. Pentru acest lucru se va crea un nou proces copil. Se va duplifica afișarea execuției într-un buffer trimis mai apoi la client.

6 Concluzii

Acest proiect va dispune un set de aplicații (client/server) de comunicare a mai multor clienți cu un server. Aceasta soluție poate fi înbunătățită printr-un algoritm de criptare de genul RSA, SSL și o interfată grafică prietenoasă.

7 Bibliografie

1. http://slacksite.com/other/ftp.html
2. http://www.unixinside.org/projects/linux-ro/RFC/rfc959-ftp.html
3. https://docstore.mik.ua/orelly/networking_2ndEd/ssh/ch01_05.htm
4. https://profs.info.uaic.ro/ computernetworks/files/NetEx/S9/servTcpCSel.c
5. http://www.cs.rpi.edu/ moorthy/Courses/os98/Pgms/socket.html