Lucrarea de laborator nr. 10

Command Line Interface

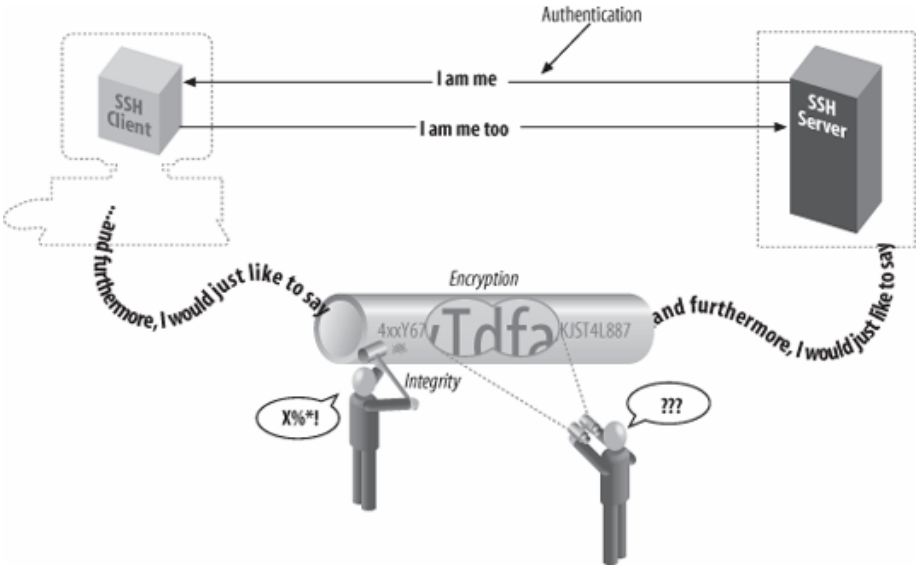
Curs: Utilizarea Sistemelor de Operare

Autor: Gabriel Noaje

SSH - Secure SHell

SSH (Secure SHell) este o abordare populara si puternica a securitatii in retea. Orice informatie trimisa de catre un calculator catre retea este automat criptata, apoi cand ajunge la destinatie este decriptata tot automat. Ceea ce rezulta este o criptare transparenta in sensul ca utilizatorul nu este afecatat in nici un fel, el lucrand normal fara sa stie faptul ca datele sunt criptate in timpul transmisiei in retea.

SSH este un protocol nu o aplicatie. Este o specificatie a modului in care trebuie securizata comunicatia intr-o retea. Protocolul SSH asigura autentificarea, criptarea si integritatea datelor transmise intr-o conexiune.
Autentificare: este realizata o identificare precisa a celui care initiaza conexiunea
Criptare: datele sunt complet criptate astfel incat doar destinatarul autentificat le poate decripta
Integritate: garanteaza ca datele care circula in timpul conexiunii ajung nealterate



Istoric

SSH1 (programul) si SSH-1 (protocolul) au fost dezvoltate in anul 1995 de catre Tatu Ylönen, cercetator la Universitatea Tehnica din Helsinki. Totul a pornit de la faptul ca in acel an universitatea a suferit un atac prin care au fost furate diverse parole. Ceea ce pornise ca o aplicatie de uz personal, in iulie 1995, SSH1 este facut public sub licenta open-source. In urma feedback-ului primit de la utilizatori Ylönen pune bazele SSH Communications Security Corporation (SCS).
Ulterior au fost descoperite o serie de limitari ale protocolului si in 1998 SCS a realizat SSH2, bazat pe protocolul SSH-2 care era mult mai evoluat si beneficia de o securitate sporita. Totusi acesta nu a inlocuit imediat SSH1 deoarece acestea nu erau compatibile, iar licenta sub care a fost publicat era mult mai restrictiva.
Situatia s-a schimbat o data cu aparitia OpenSSH, o implementare opensource a protoclului SSH-2. Aceasta era bazata pe ultima versiune originala SSH opensource (1.2.12). Proiectul s-a dezvoltat extrem de rapid devenind in scurt timp cea mai cunoscuta si utilizata implementare a protocolului SSH-2.

Caracteristicile SSH

Control la distanta

Se poate realiza logarea pe orice calculator dintr-o retea (in general din Internet) daca pe acel calculator ruleaza un server SSH. Exista si alte programe care au aceasta facilitate (telnet), dar toate datele sunt transmise in plain-text (necriptat). SSH asigura comunicatia pe un canal complet securizat si criptat. O data realizata logarea este permisa executia oricarei comenzi pe server ca si cum utilizatorul s-ar afla in fata acelui calculator.

Transfer de fisiere

Se poate realiza transfer de fisiere intre server si host folosind criptarea oferita de protocolul SSH. Transferul prin ftp nu este securizat !

Port forwarding

Protocolul SSH poate fi utilizat pentru a securiza comunicatia altor aplicatii bazate pe TCP/IP cum ar fi telnet, ftp, etc. Tehnica denumita port forwarding sau tunneling ruteaza conexiunea TCP/IP prin cea de SSH. Aceasta tehnica poate permite accesul anumitor aplicatii dincolo de un firewall care in mod normal le-ar bloca.

Clienti SSH

Unix

- Lsh – client si server dezvoltate in cadrul proiectului GNU (http://www.lysator.liu.se/~nisse/lsh/)
- OpenSSH – client si server pentru protocoalele SSH1, 1.45 si 2 dezvoltat de OpenBSD (http://www.openssh.org/)

Windows

PuTTY – client SSH ce suporta scp, ssh, sftp si telnet (http://www.chiark.greenend.org.uk/~sgtatham/putty/)

Conectarea la un server SSH

Presupunand un calculator ce are pornit serverul SSH cu numele `cs.pub.ro` si are unul dintre useri `student` conexiunea SSH cu acesta se initializeaza astfel:

```
$ ssh -l student cs.pub.ro
user's password: ******

Last login: Mon Dec 5 19:32:51 2005 from 82.78.105.203
You have new mail.
user@cs>
```

Paramentrul **–l** specifica userul cu care se va face logarea, in cazul de fata `student`. Acesta poate sa lipseasca in cazul in care userul care a initiat conexiunea este identic cu cel de pe server atunci poate fi omis aces parametru.
O alta modalitate de conexiunea este folosind user@domain:

```
$ ssh student@cs.pub.ro
```

Programului SSH ii poate fi adaugata optiunea **–v** (“verbose”) pentru a afisa informatii detaliate despre conexiune. Acestea pot fi utile in cazul depanarii unor probleme aparute in timpul conexiunii.

Atunci cand se initiaza o conexiune SSH pentru prima data catre un nou server, clientul poate afisa un mesaj de felul:

```
$ ssh student@cs.pub.ro

The authenticity of host 'cs.pub.ro (141.151.0.11)' can't be
established.
RSA key fingerprint is
77:a5:69:81:9b:eb:40:76:7b:13:04:a9:6c:f4:9c:5d.

Are you sure you want to continue connecting (yes/no)?
```

Presupunand ca raspunsul este afirmativ este afisat un mesaj de confirmare:

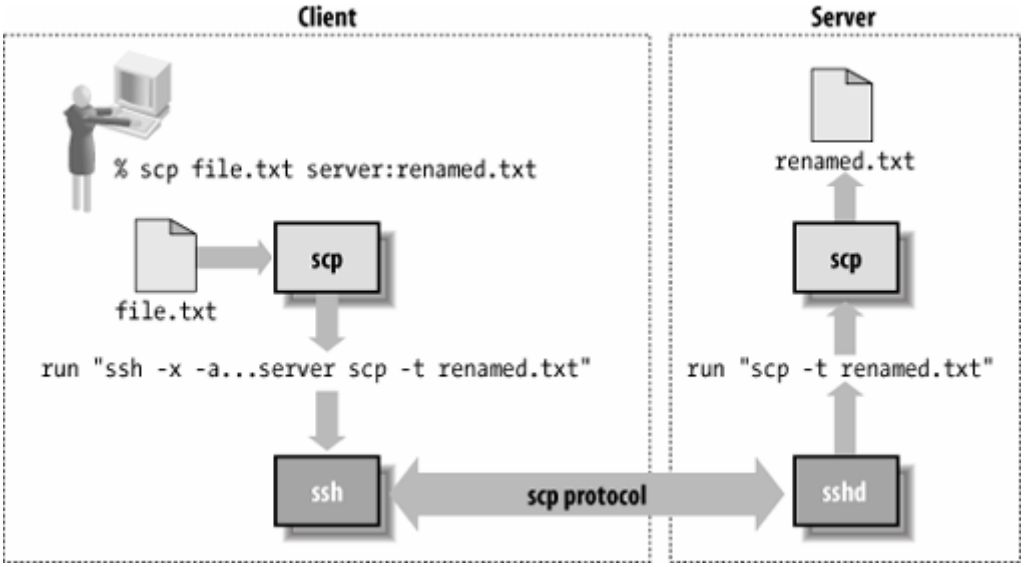
```
Warning: Permanently added 'cs.pub.ro, 141.151.0.11' (RSA) to the
list of known hosts.
```

Cand un client SSH initializeaza o conexiune, atat clientul trebuie sa isi dovedeasca identitatea (fie prin parola, fie printr-o cheie), cat si serverul trebuie sa se autentifice printr-un mecanism bazat pe o cheie publica. Fiecare server SSH are un ID unic (host key) cu care se identifica catre clienti. La o prima conexiunea catre un server, cheia publica asociate cheii private a serverului este copiata pe calculatorul local. La conexiunile ulterioare este verificata identitatea serverului pe baza cheii publice.

Exercitii
Sa se realizeze o conexiune SSH si sa se execute diferite comenzi: copiere, mutare, monitorizare procese, etc

Transfer de fisiere folosind *scp*

Programul *scp* are sintaxa asemanatoare cu comanda traditionala *cp* pentru copierea de fisiere. De fapt ideea este de a copia fisierul folosind canalul securizat oferit de SSH




```
scp sursa destinatie
```

- Daca *sursa* este un fisier, *destinatia* poate fi un fisier (existent sau nu) sau un director (existent). Adica un singur fisier poate fi copiat intr-un alt fisier sau intr-un director.
- Daca *sursa* este reprezentata de doua sau mai multe fisiere, unul sau mai multe directoare sau o combinatie, *destinatia* trebuie sa fie un director existent. Adica mai multe fisiere sau directoare pot fi copiate numai intr-un director.

Atat sursa cat si destinatia pot avea urmatoare forma de la stanga la dreapta:

- username-ul contului care contine fisierul sau directorul urmat de @. Aceasta parte este optionala si daca este omisa, valoarea este username-ul userului care a invocat comanda scp.
- hostname-ul este host-ul ce contine fisierul sau directorul urmat de semnul doua puncte (:). Aceasta parte este optionala, daca calea este prezenta, iar userul lipseste; daca este omisa, se considera implicit ***localhost***
- calea catre fisier sau director (optionala daca hostname-ul este prezent); caile relative sunt presupuse relative la directorul curent (pentru caile locale) sau la directorul ***home*** al utilizatorului (pentru caile remote); daca este omisa complet, calea este presupusa a fi directorul default

Desi fiecare camp este optional, nu pot fi omise toate in acelasi timp. Fie hostname-ul fie calea trebuie sa fie prezente.

Exemple

MyFile	Fisierul *./MyFile* pe localhost
MyDirectory	Directorul *./MyDirectory* pe localhost
server.example.com:	Directorul */home/user* pe *server.example.com* unde user este acelasi username atat pe calculatorul local cat si pe cel remote
student@server.example.com:MyFile	Fisierul *MyFile* din directorul */home/student* pe *server.example.com*

student@server.example.com:/dir/MyFile	Fisierul *MyFile* din directorul */dir* pe *server.example.com* (desi autentificarea se face cu userul *student* este specificata o cale absoluta catre fisier)

Copiaza fisierul *myfile* sub numele *myfile2* local (identic cu comanda cp)

```
$ scp myfile myfile2
```

Copiaza *./myfile* in */home/bob* pe *host1*

```
$ scp myfile bob@host1:
```

Copiaza */home/bob/myfile* in directorul curent pe *localhost*

```
$ scp student@host1:myfile .
```

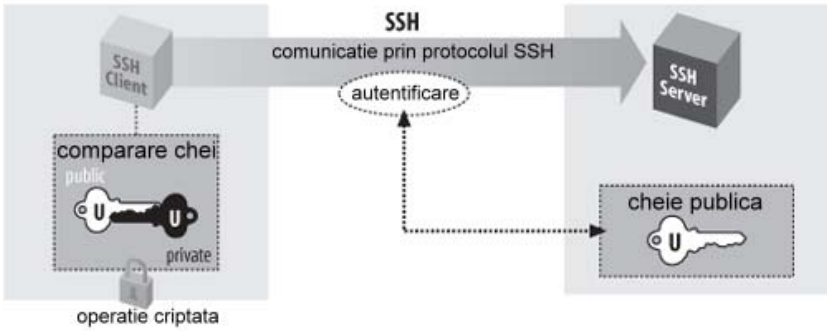
Exercitii
Sa se copieze un fisier de pe calculatorul local pe calculatorul remote si invers folosind scp.

Autentificare: parola vs. cheie criptografica

Neajunsurile unei parole:

- Pentru ca o parola sa fie sigura, ar trebui sa contina cat mai multe caractere alese aleator, dar astfel de parola este greu de memorat
- parola trimisa prin retea, chiar si protejata de un canal securizat SSH, poate fi capturata atunci cand ajunge pe calcultorul remote daca acesta a fost compromis
- Majoritatea sistemelor de operare suporta o singura parola per cont. Pentru conturile partajate apar doua probleme:
  - schimbarea parolei unui cont trebuie anuntata tuturor utilizatorilor care au acces la acel cont
  - urmarirea utilizarii unui cont este dificila deoarece sistemul de operare nu poate face diferenta intre utilizatorii aceluiasi cont

SSH rezolva aceste probleme prin utilizarea *autentificarii prin cheie publica*. O cheie reprezinta identitatea digitala a celui care o foloseste. Cheia publica este bazata pe un algoritm de criptare asimetric si consta in generarea a doua chei pereche: una publica si un privata. Exista o legatura intre cele doua chei: datele criptate cu o pereche a cheii nu pot fi decriptate fara a stii cealalta pereche si este imposibil sa se obtina cheia privata avand doar cheia publica. Astfel cheia publica este disponibila tututor facand posibila decriptarea, pe cand cheia privata trebuie pastrata cu mare atentie. Exista doi algoritmi majori care sunt folositi de catre SSH pentru generarea cheilor publice si pentru autentificare: Rivest-Shamir-Adleman (RSA) si Digital Signature Algorithm (DSA).



Generarea de chei publice folosind *ssh-keygen*

Utilizarea autentificarii criptografice presupune generearea unei perechi de chei, constand in cheia privata (identitatea digitala a userului care se gaseste pe calculatorul client) si o cheie publica (care se gaseste pe server). Programul poate genera atat chei DSA cat si RSA.
In principal sunt folosite chei de tip RSA deoarece se considera a fi mai sigure.

```
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key
(/home/student/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in
/home/student/.ssh/id_rsa.
Your public key has been saved in
/home/student/.ssh/id_rsa.pub.
The key fingerprint is:
b6:e8:e9:2c:aa:85:33:19:61:22:d1:61:d0:cf:75:ce
student@debian
```

Programul genereaza doua fisiere: unui id_rsa sau id_dsa ce contine cheia privata si inca unul avand aceasi denumire cu primul la care se adauga extensia .pub ce contine cheia publica. In cursul generarii perechii de chei publice este ceruta o fraza de trecere (*passphrase*). Aceasta va inlocui parola clasica pentru un anumit cont, fiind asociata cheii generate.

In cazul utilizarii parolelor pentru autentificare sistemul de operare mentine o asociere intre numele utilizatorului si parola. Pentru cheile criptografice, este obligatia utilizatorului de a realiza o astfel de asociere manual. Astfel dupa ce este generata, cheia publica trebuie instalata pe calculatorul remote.
Se creaza sau se modifica fisierul *~/.ssh/authorized_keys* de pe calculatorul remote si se adauga cheia publica (adica continutul fisierului id_rsa.pub generat pe calculatorul local). Un fisier *authorized_keys* contine o lista de chei publica, cate una pe un rand. OpenSSH include un program, *ssh-copy-id*, care instaleaza automat o cheie publica pe un server, folosind o singura comanda, plasand cheia in fisierul *authorized_keys*

```
$ ssh-copy-id -i key_file [user@]server_name
```

Observatii:
1) nu trebuie specificata extensia .pub pentru fisierul ce contine cheia, deoarece programul detecteaza automat care este cheia publica si pe aceea o va copia
2) daca pe calculatorul remote nu exista fisierul *authorized_keys* acesta este creat si este copiata in el cheia publica; daca fisierul exista atunci cheia publica este adaugata la sfarsitul fisierului
3) daca fisierul *authorized_keys* exista dar nu se termina cu caracterul linie noua (new line) atunci noua cheie este adaugata imediat dupa ultima cheie, lucru ce compromite ultimele doua chei

Dupa ce a fost copiata cheia publica pe server conexiunea la SSH se initializeaza la fel ca mai inainte singura diferenta fiind modalitatea de autentificare. In locul parolei este ceruta fraza de trecere.

```
$ ssh student@cs.pub.ro
Enter passphrase for key '/home/student/.ssh/id_dsa':
************
Last login: Mon Dec 5 19:44:21 2005 from 82.79.58.2
```

In cazul in care pe calculatorul local se genereaza o noua cheie privata trebuie inlocuite pe toate calculatoarele remote cheia publica veche cu cea noua.

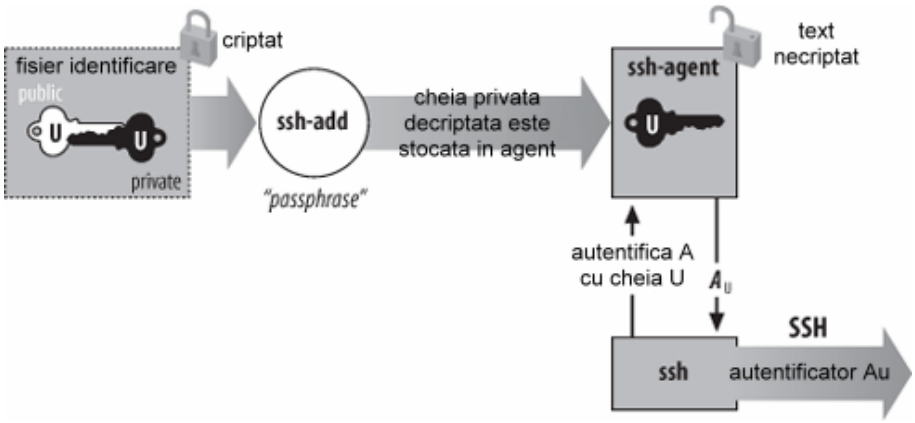
De fiecare data cand se va realiza o noua conexiunea folosind autentificarea prin cheie publica va trebui introdusa fraza de trecere. Acest lucru poate deveni suparator la un moment dat. Astfel se poate folosi un agent care sa memoreze fraza de trecere si ulterior sa comunice cu aplicatia SSH ori de cate ori aceasta solicita o autentificare. Programul se numeste *ssh-agent* si este rulat inaintea stabilirii unei conexiunii SSH, incarcadu-l cu cheia privata si cu fraza de trecere pentru aceasta.
Pornirea agentului se face folosind comanda:

```
$ ssh-agent $SHELL
```

O data agentul activat se poate incarca cheia privata.

```
$ ssh-add
Enter passphrase for /home/student/.ssh/id_dsa: ********
Identity added: /home/student/.ssh/id_dsa
(/home/student/.ssh/id_dsa)
```

In continuare toate conexiunile SSH care folosesc cheia incarcata in agent nu vor mai solicita fraza de trecere.



Exercitii
Fiecare student isi va genera propria cheie publica, o va instala pe server, o va incarca intr-un agent SSH si va realiza o conexiune SSH.

Optional: Conectarea la un server SSH folosind utilitarul PuTTy pe o platforma Windows.

Links - web browser for Unix systems

Descriere: Browser web open source cu capabilitati de afisare a paginilor complexe

Caracteristici:

- suporta partial standardul HTML 4.0 : tabele,frame-uri, diferite seturi de caractere
- afiseaza doar partea de text a site-urilor dar nu si elementele grafice
- versiunea originala de Links a fost dezvoltata de Mikuláš Patočka din Cehia
- mai tarziu grupul acestuia [Twibright Labs] a dezvoltat Links2 care poate afisa imagini, poate renderiza fonturi de diferite dimensiuni si suporta JavaScript
- Links2 poate afisa elementele grafice chiar in lipsa serverului X, dar trebuie sa fie prezenta libraria SVGAlib

Web:
Links (Wikipedia- http://en.wikipedia.org/wiki/Links_(web_browser))
Links (official homepage - http://artax.karlin.mff.cuni.cz/~mikulas/links/)

Lansarea in executie:

```
# links
```

Porneste links ca o pagina alba; se apasa ESC pt a afisa meniul

```
# links [options] URL
```

Optiunile sunt:
async-dns <0>/<1> Asynchronous DNS resolver on(1)/off(0)
max-connections <max> Maximum number of concurrent connections (default: 10)

max-connections-to-host <max> Maximum number of concurrent connection to a given host (default: 2)
retries <retry> Number of retries (default: 3)
receive-timeout <sec> Timeout on receive (default: 120)
unrestartable-receive-timeout <sec> Timeout on non restartable connections (default: 600)
format-cache-size <num> Number of formatted document pages cached (default: 5)
memory-cache-size <Kbytes> Cache memory in Kilobytes (default: 1024)
http-proxy <host:port> Host and port number of the HTTP proxy, or blank (default: blank)
ftp-proxy <host:port> Host and port number of the FTP proxy, or blank (default: blank)
download-dir <path> Default download directory (default: actual dir)
assume-codepage <codepage> Use the given codepage when the webpage did not specify its codepage (default: ISO 8859-1)
anonymous Restrict links so that it can run on an anonymous account No local file browsing. No downloads. Executing of viewers is allowed, but user can't add or modify entries in association table.
force-html Treat file as if it had an .html extension
dump Write a plain-text version of the given HTML document to stdout.
width <size> Size of screen in characters, used in combination with -dump
no-connect Runs links as a separate instance - instead of connecting to existing instance.
source Write the given HTML document in source form to stdout.
version Prints the links version number and exit.
help Prints this help screen

Meniuri si tastele de navigatie
ESC display menu
^C quit
^P, ^N scroll up, down
[,] scroll left, right
up, down select link
→ follow link
← go back
g go to url
G go to url based on current url
/ search
? search back
n find next
N find previous
= document info
**** document source
d download
c quit

Exercitii
1. Acomodarea cu interfata Links
2. Meniurile si tastele de navigatie

3. Incarcarea de pagini care cuprind diferite elemente de web-design pt a observa modul de renderizare
4. Comparatie intre afisarea acelorasi pagini web intr-un browser tip window si Links

Netcat - "TCP/IP Swiss Army knife"

Descriere:Utilitar pt retea care citeste si scrie date printr-un protocol TCP/IP

Caracteristici:

- Conexiuni TCP sau UDP catre si dinspre orice port
- mod de tunneling cu posibilitatea specificarii tuturor parametrilor conexiunii
- capabilitati de scanare de porturi

Web:
Netcat (Wikipedia - http://en.wikipedia.org/wiki/Netcat)
Netcat (official homepage - http://netcat.sourceforge.net/)

Optiuni:
c shell commands as `-e'; use /bin/sh to exec [dangerous!!]
e filename program to exec after connect [dangerous!!]
b allow broadcasts
g gateway source-routing hop point[s], up to 8
G num source-routing pointer: 4, 8, 12, ...
h this cruft
i secs delay interval for lines sent, ports scanned
l listen mode, for inbound connects
n numeric-only IP addresses, no DNS
o file hex dump of traffic
p port local port number
r randomize local and remote ports
q secs quit after EOF on stdin and delay of secs
s addr local source address
t answer TELNET negotiation
u UDP mode
v verbose [use twice to be more verbose]
w secs timeout for connects and final net reads
z zero-I/O mode [used for scanning]

Exercitii:

1. Transfer simplu de fisier

```
netcat -l -p 1111
```

```
netcat 127.0.0.1 1111
```

Pe primul calculator se porneste netcat in mod listen (’-l’) pe portul 1111 (’-p’). Pe cel de-al 2 lea calculator se initiaza o conexiune catre IP si portul specificate. Se pot scrie texte ce pot fi trimise intre cele 2 sesiuni

2. Redirectari

```
netcat -l -p 1111 > outputfile
```

```
cat infile | netcat 127.0.0.1 1111 –q 10
```

Pe primul calc se porneste netcat a.i. tot ce primeste este redirectat catre fisierul outputfile. Pe calc 2 iesirea comenzii cat aplicata fis infile este redirectata prin netcat catre calc 1. Optiunea -q 10 intrerupe conexiunea dupa 10 secunde