Laboratorul 5

Lucrul în rețea

1 Domenii

Numele unui domeniu este legat de o adresă IP. Pentru a obține adresa de IP putem folosi mai multe metode. Comanda `nslookup(1)` (*name server look-up*) primește ca prim argument numele *host*-ului și, opțional, un al doilea argument care specifică ce server DNS să folosească pentru a căuta informația.

```
$ nslookup fmi.unibuc.ro
Server:         213.154.124.1
Address:        213.154.124.1#53

Non-authoritative answer:
Name:   fmi.unibuc.ro
Address: 193.226.51.6
```

În prima parte sunt afișate date legate de server-ul DNS folosit. A doua parte oferă informațiile cerute: numele și adresa. Dacă dorim să întrebăm un server anume (în exemplul de mai jos server-ul Google) îi punem adresa în al doilea argument:

```
$ nslookup fmi.unibuc.ro 8.8.8.8
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   fmi.unibuc.ro
Address: 193.226.51.6
```

Alte comenzi utile care funcționează similar sunt:

- `dig(1)`: `$ dig @8.8.8.8 fmi.unibuc.ro` – server-ul DNS trebuie prefixat cu `@`

- `whois(1)`: `$ whois unibuc.ro` – informații despre domeniul principal, nu despre subdomeniul fmi)

Pentru a vedea dacă un *host* este conectat la rețea putem folosi comanda `ping(1)`.

```
$ ping fmi.unibuc.ro
PING fmi.unibuc.ro (193.226.51.6): 56 data bytes
64 bytes from 193.226.51.6: icmp_seq=0 ttl=50 time=7.317 ms
64 bytes from 193.226.51.6: icmp_seq=1 ttl=50 time=7.053 ms
64 bytes from 193.226.51.6: icmp_seq=2 ttl=50 time=6.925 ms
^C
--- fmi.unibuc.ro ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 6.925/7.098/7.317/0.163 ms
```

Observați că această comandă întâi găsește adresa *host*-ului și apoi comunică direct cu IP-ul acestuia. In Unix, dacă nu se specifică un număr de încercări, comanda va încerca până când utilizatorul o oprește cu `Ctrl-C`. În Windows, comanda încearcă de 4 ori implicit după care se oprește.

Fișierul `/etc/hosts` este folosit pentru a defini manual perechi adresă IP – nume. Acest fișier are prioritatea în fața serverelor DNS. Formatul este simplu:

```
$ cat /etc/hosts
127.0.0.1       localhost
192.168.1.1     myserver
5.2.14.244      alex.unibuc.ro
```

2 Acces la distanță

2.1 Transfer de date prin FTP

Pentru a accesa un *host* ce servește date prin protocolul `FTP` se folosește cumanda `ftp(1)`.

```
$ ftp alex@fmi.unibuc.ro
$ ftp ftp://fmi.unibuc.ro
$ ftp fmi.unibuc.ro -P 2121
```

Multe servere oferă informații legate de acces și structura datelor pe server la momentul conectării. Este important de văzut dacă este oferit acces anonim, fără autentificare. În acest caz de obicei se folosește utilizatorul `anonymous` și, politicos, se trece ca parola adresa de email la care puteți fi contactați. Dacă nu doriți acest lucru, apăsați pur și simplu `Enter` când se cere parola.

O dată conectați va apărea promptul `ftp>` care indică faptul că vă aflați într-un `shell` specializat protocolului `FTP` . Comenzile de navigare și manipulare a fișierelor (dacă aveți dreptul) sunt aceleași ca cele învățate până acum în `shell`: `ls`, `cd`, `pwd`, `rmdir`, `chmod` etc. Pentru a vedea toate comenzile disponibile apelați la comanda `help`.

Pentru a urca sau coborî un fișier folosiți comenzile `put`, respectiv, `get`. Dacă aveți nevoie să efectuați operația pentru mai multe fișiere puteți folosi `mput` și `mget` (`m` de la *multiple*). Pentru a ieși folosiți `quit`.

Server-ele `FTP` sunt din ce în ce mai rare în spațiul public, dar comenzile și modul de lucru este comun cu înlocuitorii lor moderni (ex. `sftp`.

2.2 Administrare prin `SSH`

Pentru a executa anumite comenzi sau a configura servicii de pe un *host* se folosește comanda `ssh(1)`.

```
$ ssh fmi.unibuc.ro
$ ssh alex@fmi.unibuc.ro
$ ssh alex@fmi.unibuc.ro -p 2222
```

Întâi se încearcă autentificarea prin chei asimetrice, iar dacă aceasta eșuează se revine la cea clasică: utilizator și parolă. O dată autentificați, suntem întâmpinați de un `shell` ca cel cu care am lucrat până acum doar că toate comenzile sunt executate pe *host*-ul la distanță nu pe mașina proprie.

Pentru a executa o simplă comandă fără a mai intra în `shell`, putem specifica comanda imediat după *host*:

```
$ ssh fmi.unibuc.ro ls
```

Generarea unei chei asimetrice se face cu comanda `ssh-keygen(1)`.

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/alex/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/alex/.ssh/id_rsa.
Your public key has been saved in /home/alex/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:ixhRJaJimfffiqTED8+ZGSp+ZMGtqwC/V7qsmxPTtAU alex@fmi
The key's randomart image is:
+---[RSA 2048]----+
|      . o..      |
|   o.Eo .        |
|.+o.o.           |
|...ooo.          |
|.   o++  S       |
|..++ooo...       |
|. +*o=....       |
|..o*@ * .        |
|.oOBoX .         |
+----[SHA256]-----+
```

Cheia publică are sufix `.pub` și se găsește în `/home/alex/.ssh/id_rsa.pub`. Cea privată se găsește în același loc dar fără sufix.

Implicit comanda `ssh-keygen(1)` generează chei `RSA`. Este recomandat să folosiți un algoritm mai nou cum ar fi `ed25519` sau `ecdsa`. Pentru aceasta folosiți argumentul `-t` *algoritm*:

```
$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
...
```

Cheile publice pentru cei care doriți să aibă acces pe contul dumneavoastră de pe un *host* (ex. calculatorul propriu, server web etc.) se pun în fișierul `.ssh/authorized_keys` din `$HOME`.

Pentru a adăuga o cheie publică `id_rsa.pub` folosiți

```
$ cat id_rsa.pub >> .ssh/authorized_keys
```

Pentru a transfera date prin `SSH` se folosește comanda `scp(1)` care se comportă aproape identic cu `cp(1)`. Diferența apare în specificarea sursei și destinației. Acestea sunt prefixate cu date legate de *host*.

```
$ scp hello.c fmi.unibuc.ro:
$ scp hello.c alex@fmi.unibuc.ro:code/
$ scp -r project/ alex@fmi.unibuc.ro:
```

Implicit, dacă nu este specificată nici o cale după `:`, transferul se face din/în directorul `$HOME` al utilizatorului. Dacă este specificată o cale, aceasta poate fi relativă `fmi.unibuc.ro:catalog` sau absolută `fmi.unibuc.ro:/etc/passwd`.

O implementare similară `FTP` folosind protocolul `SSH` este `SFTP`. Pentru a accesa un server se folosește comanda `sftp(1)` în același mod în care folosim comanda `ssh(1)`. O dată autentificați, comenzile și modul de lucru sunt aproape identice cu cazul `FTP`. Excepție face faptul că modul anonim nu mai este disponibil.

3 Sarcini de laborator

1. Găsiți adresele IP pentru `google.com`, `fmi.unibuc.ro`, `wikipedia.org`. Adăugați câte o intrare pentru fiecare în `/etc/hosts`.

2. Inter-schimbați adresele de IP pentru `google.com` și `fmi.unibuc.ro`. Folosiți `ping(1)` pentru cele două *host*-uri înainte și după modificare. Apare vreo schimbare?

3. Accesați `ftp://ftp.vim.org/`, navigați în directorul `pub/vim/pc/` și obțineți fișierul `vimXXsrc.zip` unde `XX` este cea mai recentă versiune pe care o găsiți în acel director (indiciu: folosiți `ls`).

4. Intrați pe contul Google Cloud de pe `https://cloud.google.com/` și folosiți cuponul primit pe Moodle. Urmați ghidul `https://cloud.google.com/compute/docs/tutorials/basic-webserver-apache` pentru a vă crea o mașină virtuală în cloud care servește pagini web. Folosiți `ssh-keygen(1)` pentru a genera o cheie asimetrică. Copiați cheia publică (cea cu extensia

`.pub`) pe contul Google Cloud în dreptul mașinii virtuale. Conectați-vă prin `ssh(1)` la noua mașină virtuală.

5. Pe tablă veți găsi datele de conectare la o mașină virtuală a profesorul de la laborator. Pe această mașină virtuală veți găsi un director care conține un mic site web. Copiați acest director pe mașina voastră din cloud folosind `scp(1)`. Noul director trebuie pus în `/var/www/html/prof/`.