Ghid de utilizare Linux (II) :

Comenzi UNIX. *Sisteme de fișiere* UNIX

Cristian Vidrașcu

vidrascu@info.uaic.ro

Sumar

- Introducere
- Comenzi de *help*
- Editoare de texte
- Compilatoare, depanatoare, ș.a.
- Sistemul de fișiere
- Structura arborescentă a sistemului de fișiere
- Montarea volumelor în structura arborescentă
- Protecția fișierelor prin drepturi de acces

Sumar (cont.)

- Comenzi de bază în lucrul cu fișiere și directoare
- Comenzi pentru prelucrarea conținutului fișierelor
- Alte comenzi utile pentru fișiere
- Comenzi ce oferă diverse informații
- Alte categorii de comenzi
- *Troubleshooting* (Ce să faceți dacă vi se “blochează” o comandă)

Introducere

În UNIX există două categorii de comenzi:

- *comenzi interne*: sunt implementate în interpretorul de comenzi. Exemple: `cd`, `help`, ș.a.
- *comenzi externe*: sunt implementate de sine stătător (*i.e.*, se găsesc fiecare în câte un fișier, având același nume cu comanda respectivă), în:
  - fișiere executabile (*i.e.*, programe executabile obținute prin compilare din programe sursă scrise în C sau alte limbaje). Exemple: `passwd`, `ls`, ș.a.
  - fișiere text cu comenzi, numite *script*-uri. Exemple: `.profile`, `.bashrc`, ș.a.

Introducere (cont.)

Forma generală de lansare în execuție a unei comenzi:

`UNIX>` *`comanda`* [*`opțiuni`*] [*`argumente`*]

Opțiunile și argumentele pot lipsi, după caz.
Prin convenție, opțiunile sunt precedate de caracterul '-' .
Argumentele sunt cel mai adesea nume de fișiere.
Separatorul între numele comenzii și parametrii ei, precum și între fiecare dintre parametri, este caracterul SPACE sau TAB.
Comenzile externe pot fi specificate și prin calea, absolută sau relativă, a fișierului respectiv.
O comandă poate fi scrisă pe mai multe linii, caz în care fiecare linie trebuie terminată cu caracterul '\' , cu excepția ultimei linii.

Comenzi de *help*

`UNIX>` `help`

afișează lista tuturor comenzilor interne disponibile.

`UNIX>` `help` *`nume_comandă_internă`*

afișează pagina de *help* pentru comanda internă specificată.

Pentru a obține *help* despre comenzile externe (și despre funcțiile C) sunt disponibile următoarele comenzi: `man`, `whatis`, `whereis`, `apropos`, `info`.

Observație: paginile de manual sunt organizate în 8 secțiuni. **Secțiunea 1** conține documentația despre comenzile uzuale, *i.e.* accesibile utilizatorilor obișnuiți, iar **secțiunea 8** pe cea despre comenzile accesibile doar administratorului de sistem. **Secțiunea 2** conține documentația despre apelurile de sistem din Linux, iar **secțiunea 3** conține funcțiile din biblioteca standard de C. Despre restul secțiunilor și alte detalii puteți citi aici și aici .

Comenzi de *help* (cont.)

`UNIX>` `whatis` *`nume`*

afișează lista comenzilor și funcțiilor existente cu numele specificat, precum și secțiunile de manual ce le conțin.

`UNIX>` `man [`*`secțiune`*`] `*`nume`*

afișează pagina de manual (din secțiunea specificată) pentru comanda sau funcția specificată.

`UNIX>` `whereis` *`nume`*

afișează locația comenzii și a paginii de manual asociate.

`UNIX>` `apropos` *`cuvânt`*

caută cuvântul specificat în descrierile comenzilor.

`UNIX>` `info [`*`opțiuni`*`] [`*`cuvânt`*` ...]`

afișează documentația în format INFO pentru cuvintele specificate.

Editoare de texte

- editoare de texte obișnuite:
  a) în mod text: `mcedit`, `joe`, `pico`, `nano`, `vi`, `vim`, ș.a.
  b) în mod grafic: `gedit`, `kedit`, ș.a.
- editorul `emacs`: este un editor în mod text, ce face parte din proiectul GNU, cu facilități puternice, utilizabil ca și mediu integrat de programare
- editorul TEX/LATEX: este un editor (de fapt, un mediu de lucru) ce permite tehnoredactarea de documente științifice în limbajul TEX / LATEX
- suite de birou: OPENOFFICE, LIBREOFFICE (pentru mediul grafic)

Compilatoare, depanatoare, ș.a.

Sub UNIX există compilatoare și interpretoare pentru majoritatea limbajelor de programare existente, mai noi sau mai vechi: C, C++, Pascal, Fortran, Java, Ada, ș.a.

Compilarea unui program C se face cu comanda:

```
UNIX> gcc sursa.c [-o executabil] [-Wall]
```

Compilarea unui program C++ se face cu comanda:

```
UNIX> g++ sursa.cpp [-o executabil] [-Wall]
```

Pentru depanarea programelor se poate utiliza depanatorul GNU DeBugger, accesibil prin comanda `gdb`.

Sistemul de fișiere

În sistemul de operare UNIX, datele și programele sunt păstrate în fișiere, identificate prin *nume*.
Numele fișierelor pot avea până la 255 de caractere și pot conține oricâte caractere '.' (nu sunt împărțite sub forma 8.3, *nume.extensie*, ca în sistemul de fișiere FAT din MS-DOS sau Windows), singurele restricții fiind nefolosirea caracterelor neprintabile, sau a caracterelor NULL, '/' și a spațiilor albe TAB și SPACE.
Numele fișierelor în UNIX sunt *case-sensitive* (spre deosebire de sistemele de fișiere FAT și NTFS din Windows).

Deși UNIX-ul nu impune nici o convenție privitoare la numirea fișierelor, există totuși sufixe utilizate în mod standard, precum ar fi: `.c` și `.h` pentru fișiere sursă în limbajul C, `.tar` pentru arhive tar, ș.a.

Sistemul de fișiere (cont.)

Fișierele în UNIX pot fi de următoarele tipuri:

- normale – sunt fișiere ordinare/obișnuite (iar conținutul lor este stocat pe disc)
- directoare – sunt "cataloage" de fișiere
- *link*-uri (legături), *hard* sau simbolice – sunt un fel de *alias*-uri, i.e. alte nume pentru fișiere deja existente
- fișiere speciale, în mod bloc sau în mod caracter – sunt drivere de periferice fizice sau logice
- fișiere de tip *fifo* – sunt mijloace de comunicație locală între procese
- fișiere de tip *socket* – sunt folosite pentru comunicația la distanță între procese, prin rețea (i.e., între procese rulate pe sisteme diferite)

Structura arborescentă a sistemului de fișiere

Sistemul de fișiere în `UNIX` este *ierarhizat* (arborescent), adică este ca un arbore: avem directoare ce conțin subdirectoare și fișiere propriu-zise (la fel ca în `MS-DOS` sau `Windows`).
Însă acest arbore are o singură rădăcină, referită prin "`/`" (*i.e.*, nu avem mai multe unități de discuri logice `C:`, `D:`, ..., ca în `MS-DOS` sau `Windows`), iar ca separator pentru căile de subdirectoare se utilizează caracterul '/' (în locul caracterului '\' folosit în `MS-DOS` sau `Windows`).

Fișierele pot fi accesate (specificate) fie relativ la rădăcina "`/`" sistemului de fișiere (*i.e.*, specificare prin **cale absolută**), fie relativ la directorul curent de lucru (*i.e.*, specificare prin **cale relativă la directorul curent**).

Montarea volumelor în structura arborescentă

Sistemele de fișiere UNIX se pot afla pe mai multe dispozitive fizice sau în mai multe partiții ale aceluiași disc fizic.

Fiecare dintre ele are un director root "/" și poate fi navigat prin încărcarea sistemului de operare de pe dispozitivul respectiv.
Dacă totuși sistemul de fișiere de pe un dispozitiv trebuie folosit fără încărcarea sistemului de operare de pe acel dispozitiv, există soluția de **a monta** structura arborescentă de fișiere de pe acel dispozitiv în structura dispozitivului de pe care s-a încărcat sistemul de operare.

Montarea unui sistem de fișiere se face cu comanda `mount`, iar apoi accesarea conținutului său se face prin intermediul punctului de montare. Iar operația inversă, de *demontare* a unui sistem de fișiere, se face cu comanda `umount`.

Fișiere de configurare: `/etc/fstab`, `/etc/mtab`, `/etc/filesystems`.
(Pentru detalii consultați paginile `man 1 mount` și `man 5 fstab` .)

Protecția fișierelor prin drepturi de acces

În UNIX fiecare fișier are asociat un anumit utilizator drept **utilizator proprietar**, și un anumit grup de utilizatori drept **grup proprietar** al fișierului.

Utilizatorii pot fi clasificați astfel în trei categorii în raport cu un fișier:

- proprietarul fișierului (*owner*)
- colegii de grup ai proprietarului (*group*)
- ceilalți utilizatori (*others*)

Protecția fișierelor prin drepturi de acces (cont.)

Fiecare fișier are asociate câte trei **drepturi de acces**, pentru fiecare dintre cele trei categorii de utilizatori ale acelui fișier:

- **r** (*read*) – drept de citire a fișierului
- **w** (*write*) – drept de scriere a fișierului
- **x** (*execute*) – drept de execuție a fișierului

Modificarea drepturilor de acces ale unui fișier se poate face cu comanda `chmod`. Iar schimbarea proprietarului, respectiv a grupului proprietar, al unui fișier se poate face folosind comanda `chown`, respectiv `chgrp`.

Protecția fișierelor prin drepturi de acces (cont.)

Pentru directoare, drepturile au o semnificație aparte:

- **r** (*read*) – drept de citire a conținutului directorului (*i.e.*, drept de aflare a numelor fișierelor din director)
- **w** (*write*) – drept de scriere a conținutului directorului (*i.e.*, drept de adăugare/ștergere de fișiere din director)
- **x** (*execute*) – drept de inspectare a conținutului directorului (*i.e.*, drept de acces la fișierele din director)

Protecția fișierelor prin drepturi de acces (cont.)

Verificarea dreptului de acces este efectuată de sistemul de operare la momentul inițierii lucrului cu un fișier (e.g., la momentul apelului `open()` sau `exec()`), astfel:

Presupunem că fișierul este situat la calea absolută:
`/d1/d2/d3/.../file1`,
iar proprietarul comenzii ce dorește să-l acceseze este *userul* Y.
Pentru fiecare director din lista `/`, `d1`, `d2`, `d3`, ..., sistemul de operare verifică prezența dreptului **x** pentru categoria din care face parte Y în raport cu proprietarul acelui director,
iar pentru fișierul propriu-zis, `file1`, se verifică prezența dreptului (sau drepturilor) corespunzătoare tipului de operație declarată în acel apel că se dorește a se efectua asupra fișierului.

Comenzi de bază în lucrul cu fișiere și directoare

- Comenzi de bază pentru directoare:
  - `mkdir` – pentru crearea unui director
  - `rmdir` – pentru ștergerea unui director
  - `ls` – pentru afișarea conținutului unui director
  - `pwd` – pentru aflarea directorului curent de lucru
  - `cd` – pentru schimbarea directorului curent de lucru

Comenzi de bază în lucrul cu fișiere și directoare (cont.)

- Comenzi de bază pentru fișiere:
  - `touch` – pentru crearea unui fișier obișnuit
  - `mkfifo` – pentru crearea unui fișier de tip *fifo*
  - `mknod` – pentru crearea unui fișier de tip *device*
  - `ln` – pentru crearea unui fișier de tip *alias*
  - `rm` – pentru ștergerea unui fișier
  - `cp` – pentru copierea de fișiere
  - `mv` – pentru mutarea (sau redenumirea) de fișiere
  - `mc` – utilitarul GNU Midnight Commander

Alte comenzi pentru lucrul cu fișiere

- Comenzi pentru prelucrarea conținutului fișierelor:
  - `cat`, `tac`, `more`, `less`, `head`, `tail`, `mcview` – pentru afișarea conținutului unui fișier, în diverse formate
  - `file` – oferă informații despre "tipul" unui fișier pe baza conținutului acestuia
  - `stat` – pentru afișarea atributelor (informațiile asociate) despre un fișier (*i.e.*, o comandă mai puternică decât `ls -l`)
  - `grep` – pentru selectarea liniilor de text dintr-un fișier, ce conțin un anumit șablon (*i.e.*, face "selecție pe orizontală")
  - `cut` – pentru selectarea anumitor coloane de text dintr-un fișier (*i.e.*, face "selecție pe verticală")

Alte comenzi pentru lucrul cu fișiere (cont.)

- Alte comenzi utile pentru fișiere:
  - `find` – pentru căutarea de fișiere, după diverse criterii
  - `sort` – pentru sortarea unui fișier, după diverse criterii
  - `wc` – oferă anumite statistici despre conținutul fișierelor text (*e.g.*, numărul de caractere, de cuvinte sau de linii de text)
  - `cmp`, `comm`, `diff` – pentru compararea a două fișiere, după diverse criterii
  - `sum`, `md5sum`, `sha1sum`, `sha256sum` – calculează diverse sume de control pentru fișiere
  - `tr` – filtru pentru "translatarea" anumitor caractere
  - `uniq` – filtru ce elimină liniile duplicat consecutive
  - `rev` – filtru pentru "inversarea" cuvintelor de pe fiecare linie

Comenzi ce oferă diverse informații

- informații despre utilizatori: `id`, `whoami`, `groups`, `finger`

  Fișiere de configurare: `/etc/passwd`, `/etc/group`, `/etc/(g)shadow`.
- schimbarea datelor unui cont: `passwd`, `chsh`, `chage`
- informații despre utilizatorii contectați la sistem: `users`, `w`, `who`, `finger`, `last`
- informații despre dată / timp și calendar: `date`, `cal` / `ncal`
- informații despre terminale: `tty`, `stty`
- informații despre procese: `ps`, `pstree`, `top`, `jobs`, `fg` / `bg`
- comenzi pentru planificarea/gestiunea execuției proceselor: `kill` / `killall`, `at`, `nice`, `time` / `times`

Alte categorii de comenzi

- informații diverse despre sistem: `uptime`, `hostname`, `hostnamectl`, `uname`, `lsb_release`
- scrierea de mesaje pe ecran: `echo`, `printf`, `write`, `talk`
- editoare și limbaje de *scripting*: `ed`, `sed`, `awk`, `perl`, ș.a.
- arhivare, comprimare, codificare de fișiere: `gzip` / `gunzip`, `tar`, `zip` / `unzip`, `compress`, `encode`, ș.a.
- conectarea la / deconectarea de la un sistem UNIX: `telnet`, `ssh`, `login` / `logout`
- programe pentru diverse protocoale INTERNET: `ftp`, `sftp`, `scp`, `mail` / `pine`, `lynx` / `links`, `finger`, `host`, ș.a.

Troubleshooting (Ce să faceți dacă vi se “blochează” o comandă)

În caz de blocare a unei comenzi, urmați pașii de mai jos:

- Mai întâi, așteptați un timp rezonabil, poate totuși programul nu este blocat, ci doar ocupat cu calcule laborioase. Dacă totuși nu apare prompterul, atunci:

Troubleshooting (Ce să faceți dacă vi se “blochează” o comandă)

În caz de blocare a unei comenzi, urmați pașii de mai jos:

- Mai întâi, așteptați un timp rezonabil, poate totuși programul nu este blocat, ci doar ocupat cu calcule laborioase. Dacă totuși nu apare prompterul, atunci:
- Apăsați (simultan) tastele `CTRL + C`. Aceasta determină trimiterea semnalului de întrerupere `SIGINT` programului respectiv. Dacă tot nu apare prompterul, atunci:

Troubleshooting (Ce să faceți dacă vi se “blochează” o comandă)

În caz de blocare a unei comenzi, urmați pașii de mai jos:

- Mai întâi, așteptați un timp rezonabil, poate totuși programul nu este blocat, ci doar ocupat cu calcule laborioase. Dacă totuși nu apare prompterul, atunci:
- Apăsați (simultan) tastele `CTRL + C`. Aceasta determină trimiterea semnalului de întrerupere `SIGINT` programului respectiv. Dacă tot nu apare prompterul, atunci:
- Apăsați (simultan) tastele `CTRL + \`. Aceasta determină trimiterea semnalului de terminare `SIGQUIT` programului respectiv. Dacă tot nu apare prompterul, atunci:

Troubleshooting (Ce să faceți dacă vi se “blochează” o comandă)

În caz de blocare a unei comenzi, urmați pașii de mai jos:

- Mai întâi, așteptați un timp rezonabil, poate totuși programul nu este blocat, ci doar ocupat cu calcule laborioase. Dacă totuși nu apare prompterul, atunci:
- Apăsați (simultan) tastele `CTRL + C`. Aceasta determină trimiterea semnalului de întrerupere `SIGINT` programului respectiv. Dacă tot nu apare prompterul, atunci:
- Apăsați (simultan) tastele `CTRL + \`. Aceasta determină trimiterea semnalului de terminare `SIGQUIT` programului respectiv. Dacă tot nu apare prompterul, atunci:
- Apăsați (simultan) tastele `CTRL + Z`. Aceasta determină suspendarea programului respectiv (*i.e.*, trecerea lui în starea *SUSPENDED*) și afișarea prompterului.
  Mai departe, pentru a opri acel program (el este doar suspendat, nu și terminat), procedați în felul următor. Folosind comanda `ps` aflați `PID`-ul acelui program care vi se blocase, iar apoi tastați comanda
  `UNIX> kill -9 pid`
  unde *pid* este `PID`-ul aflat anterior.
  Ca urmare a acestei comenzi procesul în cauză este omorât (*i.e.*, terminat forțat).

Bibliografie obligatorie

Cap.2, §2.1 și §2.2 din manualul, în format PDF, accesibil din pagina disciplinei “Sisteme de operare”:

- `http://profs.info.uaic.ro/~vidrascu/SO/books/ManualID-SO.pdf`

Plus documentația comenzilor uzuale, accesibilă cu comanda `man`.