

Ghid de utilizare Linux (III) :

Interpretoare de comenzi UNIX, partea I-a: Prezentare generală

Cristian Vidrașcu

`vidrascu@info.uaic.ro`

Sumar

- Introducere
- Comenzi simple. Lansarea lor în execuție
- Redirectări I/O
- Înlănțuiri de comenzi simple
- Comenzi compuse - execuția secvențială, paralelă sau condițională de comenzi simple
- Specificarea numelor de fișiere
- Fișierele de configurare
- Istoricul comenzilor tastate

Introducere

Interpretorul de comenzi (numit uneori și *shell*) este un program executabil ce are, în UNIX, același scop ca și în MS-DOS sau Windows, realizând două funcții de bază:

- preia comenzile introduse de utilizator și afișează rezultatele execuției acestora, realizând astfel interfața dintre utilizator și sistemul de operare;
- oferă facilități de programare într-un limbaj propriu, cu ajutorul căruia se pot scrie *script*-uri, adică fișiere text ce conțin secvențe de comenzi UNIX.

Comenzi simple

În UNIX există două categorii de comenzi simple:

- *comenzi interne*: sunt implementate în interpretorul respectiv. 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.

Lansarea în execuție a comenzilor simple

Forma generală de lansare în execuție a unei comenzi simple este:

```
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.

Comenzile externe pot fi specificate și prin calea, absolută sau relativă, a fișierului respectiv.

Lansarea în execuție a comenzilor simple (cont.)

O altă posibilitate de a lansa în execuție o comandă simplă este prin apelul unui anumit *shell*:

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

O a treia posibilitate, dar numai pentru comenzi ce sunt *script-uri*, ar fi:

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

sau

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

Execuția comenzilor simple în *background*

În toate cazurile anterioare, spunem că acea comandă simplă este executată în *foreground* (*i.e.*, în “planul din față”), deoarece interpretorul așteaptă terminarea execuției acelei comenzi și abia apoi oferă utilizatorului posibilitatea să execute o nouă comandă.

O altă manieră de execuție a comenzilor ar fi în *background* (*i.e.*, în “planul din spate”), adică interpretorul să nu mai aștepte terminarea execuției acelei comenzi, ci să-i ofere imediat utilizatorului posibilitatea să execute o nouă comandă.

Lansarea comenzii în *background* se face adăugând caracterul ‘&’ la sfârșitul liniei:

```
UNIX> comanda [parametri] &
```

Redirectări I/O

Există trei dispozitive logice I/O standard:

- *intrarea standard* (`stdin`), de la care se citesc datele de intrare în timpul execuției unei comenzi
- *iesirea normală standard* (`stdout`), la care sunt scrise datele de ieșire în timpul execuției unei comenzi
- *iesirea de eroare standard* (`stderr`), la care sunt scrise mesajele de eroare în timpul execuției unei comenzi

În mod implicit, dispozitivul logic `stdin` este atașat dispozitivului fizic tastatură (= terminalul de intrare), iar dispozitivele logice `stdout` și `stderr` sunt atașate dispozitivului fizic ecran (= terminalul de ieșire).

Redirectări I/O (cont.)

Interpretorul de comenzi poate “forța” o comandă ca, pe parcursul execuției sale, să primească datele de intrare dintr-un fișier specificat, în locul tastaturii, și/sau să trimită rezultatele într-un fișier specificat, în locul ecranului. Realizarea redirectărilor se face în modul urmator:

- *redirectarea intrării standard* (stdin):

```
UNIX> comanda < fișier_intrare
```

- *redirectarea ieșirii normale standard* (stdout):

```
UNIX> comanda > fișier_iesire
```

```
UNIX> comanda >> fișier_iesire
```

- *redirectarea ieșirii de eroare standard* (stderr):

```
UNIX> comanda 2> fișier_iesire_err
```

```
UNIX> comanda 2>> fișier_iesire_err
```

Înlănțuiri de comenzi simple

Înlănțuirea de comenzi se face prin simbolul *pipe*:

```
UNIX> comanda_1 | comanda_2 | ... | comanda_N
```

Simbolul '|' (*pipe*) marchează conectarea ieșirii normale standard a unei comenzi la intrarea standard a comenzii următoare din lanțul de comenzi. Comunicația se face printr-un canal anonim; se elimină astfel necesitatea comunicării prin intermediul unor fișiere temporare.

Toate comenzile sunt executate simultan, în paralel (nu secvențial!).

Exemple:

```
UNIX> ls -Al | wc -l
```

Efect: afișează numărul fișierelor (și subdirectoarelor) din directorul curent.

```
UNIX> who | cut -f1 -d" " | sort -u
```

Efect: afișează lista ordonată a numelor utilizatorilor conectați la sistem.

```
UNIX> cat fis1 fis2 > fis3
```

Efect: concatenează primele două fișiere în cel de-al treilea.

Comenzi compuse - execuția secvențială, paralelă sau condițională de comenzi simple

Două (sau mai multe) comenzi simple se pot grupa într-o comandă compusă, prin tastarea lor pe un singur rând, separate prin:

- caracterul ';' – *execuție secvențială*

Comenzile simple separate prin caracterul ';' sunt executate secvențial, în ordinea în care apar în linia de comandă.

Exemplu: `UNIX> ls -A ; cd html ; ls -l`

- caracterul '|' (*pipe*) – *execuție paralelă, înlănțuită*

Comenzile simple separate prin caracterul '|' sunt executate simultan (*i.e.*, în paralel), fiind înlănțuite prin *pipe* (a se vedea slide-ul anterior).

Exemplu: `UNIX> cat /etc/passwd | grep -w so`

- caracterul '&' – *execuție paralelă, neînlănțuită*

Comenzile simple separate prin caracterul '&' sunt executate (aproape) simultan, fiind lansate în *background* pe rând, dar nu sunt înlănțuite prin *pipe* (*i.e.*, fiecare rulează în mod independent de cealaltă).

Exemplu: `UNIX> cat /etc/passwd & cat /etc/group [&]`

Execuția secvențială, paralelă sau condițională de comenzi (cont.)

Execuția unei comenzi poate fi condiționată de rezultatul execuției unei alte comenzi:

- caracterele '&&' – *conjuncția a două comenzi*:

UNIX> *comanda_1* && *comanda_2*

Efect: mai întâi se execută prima comandă, iar apoi se va executa a doua comandă numai dacă execuția primei comenzi se termină cu succes (*i.e.*, *comanda_1* întoarce codul de retur 0).

- caracterele '||' – *disjuncția a două comenzi*:

UNIX> *comanda_1* || *comanda_2*

Efect: mai întâi se execută prima comandă, iar apoi se va executa a doua comandă numai dacă execuția primei comenzi se termină cu eroare (*i.e.*, *comanda_1* întoarce un cod de retur nenul).

Specificarea numelor de fișiere

Specificarea numelor de fișiere ca argumente pentru comenzi se poate face:

- prin *calea absolută* (*i.e.*, pornind din directorul rădăcină)
Exemplu: `/home/vidrascu/so/file0003.txt`
- prin *calea relativă* la directorul curent de lucru (*i.e.*, pornind din directorul curent)
Exemplu (presupunând că directorul curent este `/home/vidrascu`):
`so/file0003.txt`
- prin *calea relativă la un director home* al unui anumit utilizator (*i.e.*, pornind din directorul *home* al acestuia)
Exemplu: `~vidrascu/so/file0003.txt`

Specificarea numelor de fișiere (cont.)

Se poate specifica o listă de fișiere (un “șablon”) ca argumente pentru comenzi prin folosirea următoarelor caracterele speciale:

- caracterul '*': înlocuiește orice șir de caractere, inclusiv șirul vid

Exemplu: `~vidrascu/so/file*.txt`

- caracterul '?': înlocuiește orice caracter (exact un caracter)

Exemplu: `~vidrascu/so/file000?.txt`

Specificarea numelor de fișiere (cont.)

Se poate specifica o listă de fișiere (un “șablon”) ca argumente pentru comenzi prin folosirea următoarelor caracterele speciale:

- **specificatorul mulțime de caractere [...]** : înlocuiește exact un caracter, dar nu cu orice caracter posibil, ci doar cu cele specificate între parantezele '[' și ']', sub formă de enumerare (separate prin ',' sau nimic) și/sau interval (dat prin capetele intervalului, separate prin '-')

Exemple: `~vidrascu/so/file000[1,3,579].txt` ,
`~vidrascu/so/file00[0-9][3-9].txt` ,
`~vidrascu/so/file000[1-3,57-9].txt`

Specificarea numelor de fișiere (cont.)

Se poate specifica o listă de fișiere (un “șablon”) ca argumente pentru comenzi prin folosirea următoarelor caracterele speciale:

- **specificatorul mulțime exclusă de caractere `[^...]` :**
înlocuiește exact un caracter, dar nu cu orice caracter posibil, ci doar cu cele din complementara mulțimii specificate între parantezele '[' și ']' similar ca mai sus, exceptând faptul că primul caracter de după '[' trebuie să fie '^' pentru a indica complementariere (excludere)

Exemplu: `~vidrascu/so/file000[^1-3].txt`

Specificarea numelor de fișiere (cont.)

Se poate specifica o listă de fișiere (un “șablon”) ca argumente pentru comenzi prin folosirea următoarelor caracterele speciale:

- caracterul `'\'` : se folosește pentru a inhiba interpretarea operator a caracterelor speciale anterioare, și anume `\c` (unde `c` este unul dintre caracterele `'*'`, `'?'`, `'['`, `']'`, `'^'`, `'\'`) va interpreta acel caracter `c` ca text (*i.e.*, prin el însuși) și nu ca operator (*i.e.*, prin “șablonul” asociat lui în felul descris mai sus)

Exemple: `ce_mai_faci\?.txt` , `lectie\[lesson].txt`

Fișierele de configurare

Fișiere de configurare ale interpretorului bash:

- *script-uri de inițializare a proceselor shell de login:*
locale, specifice unui utilizator (`~/ .profile` sau `~/ .bash_profile`)
și globale (`/etc/profile` și `/etc/environment`)
- *script de finalizare a acestora:*
local (`~/ .bash_logout`)
- *script-uri de inițializare a proceselor shell interactive, cu excepția celor de login:*
local (`~/ .bashrc`) și global (`/etc/bashrc`)

Istoricul comenzilor tastate

- *fișier de istoric* al comenzilor tastate:
local (`~/ .bash_history`)

Poate fi vizualizat cu comanda `history`.

Numerele afișate de această comandă pot fi folosite pentru a executa în mod repetat comenzile din istoric, fără a fi nevoie să mai fie re-tastate. Și anume, pentru a invoca din nou o comandă, se tastează la prompter caracterul `!` urmat imediat de numărul asociat comenzii respective.

O altă facilitate a interpretorului `bash` ce permite invocarea comenzilor tastate anterior, precum și editarea lor, constă în navigarea prin istoric prin apăsarea, la prompter, a tastelor `UP` și `DOWN` (*i.e.*, tastele săgeată-sus și săgeată-jos).

Bibliografie obligatorie

Cap.2, §2.3 din manualul, în format PDF, accesibil din pagina disciplinei “Sisteme de operare”:

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