

01 - Conceptes generals

Curs 2020 - 2021

ASIX M01-ISO UF1-A01-02 ordres de fitxers

Conceptes generals	1
Descripció	1
La consola / terminal	1
Iniciar / Tancar sessió	1
El prompt i una sessió d'usuari	2
Qui som i on som?	3
Escriure ordres	4
Consultar l'ajuda: man / synopsis	5
Completar rutes amb tab	7
Exercicis d'exemple:	7

Conceptes generals

Descripció

La consola / terminal

Per practicar treballar en mode comanda en el shell bash cal accedir a una terminal de consola. Pot ser una terminal gràfica obrint alguns dels programes tipus `gnome-terminal`, `terminator`, etc.

Però també es poden obrir sessions en les terminals de text que ofereix el sistema apremant `alt+control+Fn`. Una de les consoles té la pantalla de greetings (benvinguda) per iniciar una sessió gràfica. Si obrim sessions gràfiques anirem consumint consoles. En general des de F1 a F7 hi ha consoles que podem usar (si no són gràfiques).

Iniciar / Tancar sessió

Un cop en una consola cal iniciar sessió identificant-se amb l'usuari i la contrassenya. Si l'autenticació és correcta s'inicia una sessió en un shell.

Per tancar una sessió podem realitzar les ordres: *logout*, *exit*, *control+d* (^d).
Per apagar el sistema l'ordre *poweroff* (segurament no ho permetrà si no som l'usuari privilegiat root).

Per reiniciar el sistema disposem de l'ordre *reboot* (altre cop només per a usuaris privilegiats).

El prompt i una sessió d'usuari

Un cop iniciada una sessió d'usuari el sistema mostra el prompt, l'inductor del sistema que ens informa que està llest i a punt per rebre ordres. Cada cop que mostra el prompt és que ha acabat la tasca que realitzava i està esperant una nova instrucció.

```
[ecanet@a36 ~]$ id
uid=1001(ecanet) gid=1001(ecanet) groups=1001(ecanet),10(wheel),971(vagrant),975(docker)

[ecanet@a36 ~]$ sleep 99999
^C
[ecanet@a36 ~]$
```

- En mostrar el prompt es realitza l'ordre id que mostra la informació de identificació de l'usuari. Un cop acabada torna a mostrar el prompt per indicar que està llest per rebre noves ordres.
- En fer l'ordre sleep (compta ovelletes) el prompt no es mostra perquè està comptant i fins que no acabi no el tornarà a mostrar.
- S'ha interromput l'execució del programa amb control+c.
- Un cop interromput ha tornat a mostrar el prompt indicant que torna a estar llest per rebre ordres.

^C permet interrompre l'execució dels programes (control+c).

[ecanet@a36 ~]\$

- user* el primer element del prompt indica el nom de l'usuari que som. En l'exemple l'usuari ecanet.
- @* significa at indica que l'usuari està a tal host.
- host* el hostname o nom de la màquina, en l'exemple a36. Per tant significa que som l'usuari ecanet treballant en la màquina a36.
- ~* el directori actiu. En aquest cas el home de l'usuari. El caràcter ~ és una abreviatura del home dels usuaris.
- \$* indica que es tracta d'un usuari no privilegiat, d'un usuari normal.

[root@server /boot]#

- user* el primer element del prompt indica el nom de l'usuari que som. En l'exemple l'usuari root o superusuari.
- @* significa at indica que l'usuari està a tal host.

- host** el hostname o nom de la màquina, en l'exemple server. Per tant significa que som l'usuari root treballant en la màquina server.
- /boot** el directori actiu. En aquest cas el directori /boot.
- #** indica que es tracta d'un usuari privilegiat, el superusuari **root**.

```
[ecanet@a36 ~]$ id
uid=1001(ecanet) gid=1001(ecanet) groups=1001(ecanet),10(wheel),971(vagrant),975(docker)

[ecanet@a36 ~]$ su -
Password:

[root@a36 ~]# id
uid=0(root) gid=0(root) groups=0(root)

[root@a36 ~]# logout
[ecanet@a36 ~]$
```

- Primerament es mostra la informació d'identificació id del nostre usuari. Podem observar al prompt el caràcter \$.
- Amb l'ordre su - ens convertim en root (indicant el seu password).
- A continuació es mostra la informació id de root. Observem que en el prompt el caràcter ja no és \$ sinó #.
- Finalment finalitzem la sessió de root i tornem a ser l'usuari no privilegiat.

Qui som i on som?

Hi ha diverses ordres per saber amb quin usuari estem treballant (podem tenir sessions obertes amb usuaris diferents) com per exemple: **id**, **whoami**, **who**. També hi ha ordres per veure quin és el directori actiu, llistar-ne el contingut i canviar de directori: **pwd** i **ls** per exemple.

```
[ecanet@a36 ~]$ id
uid=1001(ecanet) gid=1001(ecanet) groups=1001(ecanet),10(wheel),971(vagrant),975(docker)

[ecanet@a36 ~]$ whoami
ecanet

[ecanet@a36 ~]$ who
ecanet  tty2      2020-10-10 09:37 (:0)
[ecanet@a36 ~]$
```

- Primerament mostra la informació d'identificació id del usuari.
- La segona ordre mostra el nom de l'usuari.
- La tercera llista els usuaris que hi ha connectats al sistema. En aquest exemple només un en el terminal tty2.

Per defecte en iniciar sessió els usuaris es troben en el seu home com a directori actiu. Cada usuari té definit un directori del disc dur com el seu directori, tot el què hi fa dins li pertany. Aquest directori s'anomena el seu HOME.

Generalment els usuaris tenen homes del tipus `/home/pere` (per l'usuari pere), `/home/guest` (per l'usuari guest), etc. Però això no és sempre així ni té perquè ser-ho. En una organització els usuaris del vendes poden tenir el home a `/home/vendes/pere` per exemple. Els alumnes de primer d'asix per exemple a `/home/users/inf/hisx1`.

L'usuari `root` sempre té el seu home a `/root`.

```
[ecanet@a36 ~]$ pwd
/home/ecanet

[ecanet@a36 ~]$ ls
Desktop Documents Downloads escola.20200922.ldiff memories.txt Music Pictures pt Public Templates
Videos

[ecanet@a36 ~]$ cd /tmp/
[ecanet@a36 tmp]$ pwd
/tmp

[ecanet@a36 tmp]$ cd
[ecanet@a36 ~]$ pwd
/home/ecanet
```

- Mostra quin és el directori actiu, el home de l'usuari és `/home/ecanet`.
- Observar que el prompt no el mostra tal qual sinó la abreviació `~` que significa el home de l'usuari actual.
- La segona ordre llista el contingut del directori home de l'usuari.
- Amb l'ordre `cd` es canvia el directori actiu per el directori `/tmp`. Ara aquest és el directori actiu.
- L'ordre `pwd` mostra que ara `/tmp` és el directori actiu.
- Amb l'ordre `cd` sense arguments es torna a fer actiu el directori home de l'usuari.

Escriure ordres

Un cop es mostra el prompt l'usuari pot escriure una ordre i prémer enter per tal de que sigui executada. Les ordres poden tenir opcions i arguments.

\$ *ordre [opcions]... [arguments]...*

[opcions] les opcions modifiquen com funciona l'ordre. Una ordre que per exemple llista fitxers li podem demanar que llisti només els noms, o tota la informació per a cada fitxer o els fitxers ocults, etc. Les opcions li indiquen a l'ordre aspectes de com volem que funcioni.

Pot no haver-hi opcions, una o múltiples. En format *short-option* es poden combinar i generalment l'ordre no és significatiu.

Les opcions poden ser:

- ❑ ***short-option*** s'indiquen per un guió i una lletra, per exemple `-l`.
- ❑ ***long-option*** s'indiquen per dos guions i una paraula, per exemple `--all`.

Per exemple:

- `$ ls -la`
indica llistar fent servir les opcions `-l` (l·listat llarg) i `-a` (l·listar-ho tot).
- `$ ls --all`
indica llistar mostrant-ho tot (els fitxers ocults també).
`$ ls -l -a`
indica fer un l·listat llarg de tot. En aquest cas no s'han agrupat les opcions.
- `$ ls -l --all`
indica fer un l·listat llarg de tot. Es combina una short i una long option.

[arguments] Els arguments són paràmetres que necessita la ordre per funcionar. Hi ha ordres que no necessiten arguments però d'altres que si. Per exemple l'ordre `pwd` que indica quin és el directori actiu no necessita res més per funcionar. L'ordre que esborra fitxers necessita com a mínim un argument, què cal esborrar. L'ordre que copia fitxers necessita almenys dos arguments què es vol copiar (origen) i a on es vol copiar (destí).

Per exemple:

- `cat /tmp/carta.txt`
mostra el contingut del fitxer `carta.txt`. Cal almenys un argument que és què volem mostrar.
- `cp carta.txt /tmp/treball.txt`
copia el fitxer `carta.txt` al directori `tmp` amb el nom `treball.txt`. Calen dos arguments què es vol copiar i a on es vol copiar.
- `cat fitxer.txt /etc/fstab /boot/grub2/grub.cfg`
mostra el contingut de tres fitxers, per tant calen tres arguments.

Recordeu:

opcions: modifiquen com actua l'ordre però no són necessàries per fer-la funcionar.

arguments: són necessaris per indicar a l'ordre sobre què ha d'actuar.

Consultar l'ajuda: `man` / synopsis

Per saber com funciona una ordre, quines opcions té i quins arguments requereix és recomanable consultar el `man` de la ordre.

L'ordre **man** prové de **manual** i és la ordre que mostra l'ajuda de les ordres del sistema:

`$ man <ordre>`

- ❑ Quan s'entra al `man` es pot navegar amb `pgup` i `pgdown`, `inici`, `fi` i també l'espai.

- ❑ Per sortir prémer la 1 de quit.

Cal saber interpretar la *synopsis* de les ordres, és la gramàtica, la manera d'escriure-les correctament.

Llistat del man de l'ordre ls

LS(1)	User Commands	LS(1)
NAME		
ls - list directory contents		
SYNOPSIS		
ls [OPTION]... [FILE]...		
DESCRIPTION		
List information about the FILES (the current directory by default). Sort entries alphabetically if none of -ctuvSUX nor --sort is specified.		
Mandatory arguments to long options are mandatory for short options too.		
-a, --all		
do not ignore entries starting with .		
-A, --almost-all		
do not list implied . and ..		
--author		
with -l, print the author of each file		

Exemple de la sinopsi de l'ordre ls:

ls [option]... [file]...

- Significa que pot tenir cap o múltiples opcions (tipus -l, -a, etc).
- També pot tenir cap o múltiples arguments. Quan és cap llista el directori actiu, si hi ha arguments llista cada un dels arguments.

Llistat del man de l'ordre cp

CP(1)	User Commands	CP(1)
NAME		
cp - copy files and directories		
SYNOPSIS		
cp [OPTION]... [-T] SOURCE DEST		
cp [OPTION]... SOURCE... DIRECTORY		
cp [OPTION]... -t DIRECTORY SOURCE...		
DESCRIPTION		
Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.		
Mandatory arguments to long options are mandatory for short options too.		
-a, --archive		
same as -dR --preserve=all		
--attributes-only		
don't copy the file data, just the attributes		
--backup[=CONTROL]		
make a backup of each existing destination file		
-b like --backup but does not accept an argument		

Exemple d'una de les sinopsi de cp:

cp [options]... SOURCE DEST

- Pot haver-hi cap o múltiples options.
- Hi ha com a mínim dos arguments (origen i destí)
- El primer argument és l'origen SOURCE, què volem copiar.
- El segon argument és el destí, DEST, on ho volem copiar.

Completar rutes amb tab

Un dels problemes més habituals és que els usuaris cometen errors tipogràfics en escriure rutes, fins i tot en escriure el nom de les ordres. Un gran avantatge de GNU/Linux és la compleció automàtica dels noms prement la tecla [tab].

[tab] el primer cop completa o intenta completar. Si només hi ha una opció per completar la completa. Si n'hi ha múltiples no fa res.

[tab][tab] quan hi ha múltiples opcions prement una segona vegada [tab] es llisten totes les opcions disponibles.

```
[ecanet@a36 ~]$ user [tab]
useradd  userdel  userhelper  usermod  usernetctl  users

[ecanet@a36 ~]$ ls /boot/[tab][tab]
06f1b134f5ac451a823ad828b1047db0/  initramfs-4.18.19-100.fc27.x86_64.img
config-4.18.19-100.fc27.x86_64      System.map-4.18.19-100.fc27.x86_64
efi/                                vmlinuz-0-rescue-06f1b134f5ac451a823ad828b1047db0
extlinux/                          vmlinuz-4.18.19-100.fc27.x86_64
grub2/                             vmlinuz-4.18.19-100.fc27.x86_64.hmac
initramfs-0-rescue-06f1b134f5ac451a823ad828b1047db0.img
```

Exercicis d'exemple:

Primerament heu de fer l'exercici **00-Creació_de_estructura** per generar els directoris i els fitxers necessaris per fer aquest exercici.

1. Fer actiu el directori **/tmp/mp1**.

Des d'aquest directori realitzarem tots els exercicis, tots. No és permès de canviar de directori amb l'ordre cd. Totes les ordres del sistema es poden realitzar des de qualsevol directori actiu.

Després de fer aquest directori actiu verificar-ho amb l'ordre **pwd**.

```
$ cd /tmp/m01/

$ pwd
/tmp/m01
```

Recordeu quina és l'estructura de directoris i fitxers que estem utilitzant:

```
$ tree /tmp/m01
/tmp/m01
├── operatiu
│   ├── apunts
│   │   ├── carta.txt
│   │   ├── dades.pdf
│   │   ├── dossier.odt
│   │   ├── informe.pdf
│   │   ├── projecte.odt
│   │   └── treball.txt
│   └── exercicis
└── xarxes

$ tree /var/tmp/prova/
```

```
/var/tmp/prova/
```

Observeu i identifiqueu quins elements són fitxers i quins directoris.

2. Mostrar el *directori actiu*.

```
$ pwd  
/tmp/m01
```

3. Què mostra el *prompt*?

```
[ecanet@a36 m01]$
```

4. Inicia en una segona consola de text una sessió de *root*

```
Prémer Alt+Control+Fnombre  
user: root  
passwd: <el password de root>
```

5. Consulta l'ajuda de l'ordre *who*.

```
$ man who  
WHO(1) User Commands WHO(1)  
  
NAME  
  who - show who is logged on  
  
SYNOPSIS  
  who [OPTION]... [ FILE | ARG1 ARG2 ]  
  
DESCRIPTION  
  Print information about users who are currently logged in.  
  
  -a, --all  
      same as -b -d --login -p -r -t -T -u  
  
  -b, --boot  
      time of last system boot
```

6. Executa l'ordre *who* per observar quins usuaris hi ha connectats al sistema i a quines terminals.

```
$ who  
ecanet tty2 2020-10-15 07:59 (:0)  
root tty3 2020-10-15 08:25
```

- Observeu que la columna tty indica a quina terminal està connectat cada usuari.

7. Canvia de terminal i torna a la sessió d'usuari no privilegiat (la d'alumne no la de root).

```
Prémer alt+ctrl+Fnombre
```


- Podem anar canviant de terminal de text variant la tecla de funció que usem, a cada tecla de funció hi ha una terminal de text o gràfics (a les sis primeres).

8. Verifica que ets realment el teu usuari amb l'ordre *whoami*.

```
$ whoami
ecanet
```

9. Executa l'ordre *id* que mostra les dades identificatives de l'usuari.

```
$ id
uid=1001(ecanet) gid=1001(ecanet) groups=1001(ecanet),10(wheel),971(vagrant),975(docker)
```

- Podem observar que es mostra el *UID* User Identifier (valor numèric identificador de l'usuari) i el *GID* (Group ID) grup principal de l'usuari, també la llista d'altres grups als que pertany.

10. Executa l'ordre *id root* (des de la sessió del teu usuari).

```
$ id root
uid=0(root) gid=0(root) groups=0(root)
```

- Observeu que tot i ser un altre usuari (ecanet) li podem demanar al sistema que mostri les dades identificatives de qualsevol usuari.
- Quan no li passem arguments a l'ordre *id* interpreta que ha de mostrar les dades de l'usuari actual.

11. Consulta l'ajuda de l'ordre *id*.

```
$ man id
ID(1)                                User Commands                                ID(1)

NAME
  id - print real and effective user and group IDs

SYNOPSIS
  id [OPTION]... [USER]

DESCRIPTION
  Print user and group information for the specified USER, or (when USER omitted) for the current user.

  -a  ignore, for compatibility with other versions
  -Z, --context
      print only the security context of the process
  -g, --group
      print only the effective group ID
  -G, --groups
      print all group IDs
```

12. Fes l'ordre *ls /* que llista l'arrel del sistema

```
$ ls /
bin boot dades dev etc home lib lib64 lost+found media mnt opt proc root run sbin srv sys tmp usr var
```

13. Torna a fer l'ordre anterior però afegint l'opció -l per fer un llistat llarg. Cal fer `ls -l /`.

```
$ ls -l /
total 72
lrwxrwxrwx. 1 root root 7 3 ago 2017 bin -> usr/bin
dr-xr-xr-x. 6 root root 4096 20 abr 19:59 boot
drwxrwxrwx. 2 root root 4096 25 set 2019 dades
drwxr-xr-x. 22 root root 4200 15 oct 07:58 dev
drwxr-xr-x. 150 root root 12288 15 oct 07:58 etc
drwxr-xr-x. 25 root root 4096 19 set 00:59 home
lrwxrwxrwx. 1 root root 7 3 ago 2017 lib -> usr/lib
lrwxrwxrwx. 1 root root 9 3 ago 2017 lib64 -> usr/lib64
drwx----- 2 root root 16384 17 set 2019 lost+found
drwxr-xr-x. 2 root root 4096 3 ago 2017 media
drwxr-xr-x. 2 root root 4096 3 ago 2017 mnt
drwxr-xr-x. 6 root root 4096 20 jun 17:54 opt
dr-xr-xr-x. 311 root root 0 15 oct 2020 proc
dr-xr-xr-x. 9 root root 4096 10 oct 20:54 root
drwxr-xr-x. 44 root root 1280 15 oct 08:25 run
lrwxrwxrwx. 1 root root 8 3 ago 2017/sbin -> usr/sbin
drwxr-xr-x. 2 root root 4096 3 ago 2017 srv
dr-xr-xr-x. 13 root root 0 15 oct 07:58 sys
drwxrwxrwt. 15 root root 380 15 oct 08:26 tmp
drwxr-xr-x. 12 root root 4096 17 set 2019 usr
drwxr-xr-x. 22 root root 4096 19 set 2019 var
```

14. Escriu `ls /` i prem *dos* cops *tabulador*, què passa?

```
$ ls /[tab][tab]
.autorelabel dades/ home/ lost+found/ opt/ run/ sys/ var/
bin/ dev/ lib/ media/ proc/ sbin/ tmp/ virtualbox/
boot/ etc/ lib64/ mnt/ root/ srv/ usr/
```

- En prémer per primera vegada la tecla tab no fa res perquè dins de l'arrel hi ha múltiples coses i no sap amb quina completar la ruta.
- En prémer per segon cop mostra totes les rutes possibles amb que pot completar, continuar el camí des de l'arrel.

15. Escriu `ls /b` i prem *dos* cops *tabulador*, què passa?

```
$ ls /b[tab][tab]
bin/ boot/
```

- En prémer tab per primer cop no expandeix (complet) cap ruta perquè hi ha múltiples opcions dins de l'arrel que comencen per b.
- En prémer tab per segon cop completa amb totes les rutes possibles que tinguin un com a continuació del que hem començat a escriure. Completa els noms que comencen per b.

16. Escriu `ls /e` i prem *un* cop *tab*. Què passa?

```
$ ls /etc/
```

- En prémer la e i tab com que només hi ha un element que pugui completar, el directori /etc/, el completa.

17. Escriu cat `/etc/pas` i prem *un* cop *tabulador*, què passa?

```
$ cat /etc/passwd
```

- Com que només hi ha un element dins de /etc que comenci per pass completa el nom del fitxer /etc/passwd.

18. Consultant *l'ajuda* de l'ordre *cp* indica com es diu la *long-option* equivalent a la *short-option -r*.

```
-R, -r, --recursive  
copy directories recursively
```

- Podem observar que les opcions -R i -r i --recursive són la mateixa opció.

19. Consultant *l'ajuda* de l'ordre *mv* indica com es diu la *short-option* equivalent a la *long-option --interactive*.

```
-i, --interactive  
prompt before overwrite
```

- Podem observar que les opcions -i i --interactive són la mateixa opció