UNIVERSITAT POMPEU FABRA
Facultat d'Economía

SESSIÓ SOBRE UNIX

F. Udina
14 de desembre de 1992

1 Conceptes bàsics (Guió per a una explicació)

1. UNIX es un sistema operatiu implementat en màquines de molts fabricants diferents. Hi ha lleugeres variants dialectals.
2. UNIX es multiusuari i multitasca.
3. Hosts, usernames i passwords. Vies d'accés a UNIX a la UPF.

   Des de un PC correctament connectat a la xarxa de la UPF i amb el software estàndar, ha de ser possible fer
   `telnetd` *nomDeLaMaquina*,
   per exemple, `telnetd sahara` o `telnetd xaloc`.

   Des d'un Mac, cal obrir l'aplicació anomenada "Telnet" o similar, i triar, en el menú File la opció Open Connection.... Caldrà donar com a Session name el de la màquina a la que volguem entrar.

   En qualsevol dels dos casos, pot passar que el software no conegui la màquina pel seu nom. Caldrà alsehores donar el seu número de xarxa complet: `sahara` és `130.206.64.1`, `xaloc` és `130.206.64.2`.
4. Arxius. Directoris. El *home directory* i el *current directory*.
5. Shell. Ordres de shell. Execució de programes.

   Estructura de les ordres de shell UNIX. Opcions i arguments.

2 Ordres de shell més importants

2.1 Com especificar arxius i directoris

En UNIX, tota la informació es guarda en arxius, i aquests poden estar agrupats en directoris. Els conceptes d'arxiu i directori són similars als de DOS o als de Macintosh (un directori és una *carpeta*).

En els noms d'arxius i directoris, **es diferencia entre majúscules i minúscules**.

Per indicar un arxiu `arx23`que està dins d'un directori `dirXX`, teclejem `arx23/dirXX`. Observa que enlloc de la barra `\` de MS-DOS o els dos punts `:` del Mac, aquí s'utilitza la barra `/`. Per indicar el directori pare del que estem, posem `..` (dos punts un darrera l'altre). Per indicar el mateix directori on estem, posem simplement un punt.

Tot i que el noms d'arxius solen tenir la forma *nom.extensió*, la extensió no té massa importància, en tot cas no té la que li dóna MS-DOS.

La majoria d'ordres UNIX que manipulen arxius admeten com a arguments els noms d'arxiu amb comodins. El comodí més usual és l'asterisc (*), que representa *qualsevol combinació de caràcters*

2.2 Ordres de manipulació d'arxius

`pwd` Quin és el directori actual? (*print working directory*)

`ls` Serveix per llistar els arxius d'un(s) directori(s). La ordre `ls` té moltes variants, només en veurem algunes. En qualsevol d'elles, haurem de dir els arxius que ens interessa veure llistats (cosa que indiquem aquí amb el símbol *quinsArxius*. En el seu lloc podem no posar-hi res (voldrem dir que ens interessen tots els arxius del directori on estem), o podem posar-hi el nom d'un directori, o un nom amb comodins.

Els llistats, si no es diu el contrari, surten sempre per odre alfabètic, (les majúscules van totes al davant de les minúscules).

`ls` *quinsArxius* Dóna un llistat compacte dels arxius.
Exemples:

```
ls
ls *ps
ls *92*
```

`ls -l` *quinsArxius* Llistat llarg, com ara:

```
-rw-r--r--   1 udina    users       794 May  5  1992 assigPlaEstudis.tex
-rw-r--r--   1 udina    users       120 Apr  7  1992 assigPlaEstudis.txt
drwxr-xr-x   5 udina    users      1024 Dec  9 14:41 matesI
drwxr-xr-x   2 udina    users      1024 Dec 14 13:30 proves
-rw-rw-rw-   1 udina    users       887 Feb  6  1992 sessinf.tex
-rw-r--r--   1 udina    users       661 Dec 14 14:36 sessioUnix.aux
-rw-r--r--   1 udina    users     11212 Dec 14 14:36 sessioUnix.dvi
-rw-r--r--   1 udina    users      1410 Dec 14 14:36 sessioUnix.log
-rw-r--r--   1 udina    users      9404 Dec 14 14:36 sessioUnix.tex
-rw-r--r--   1 udina    users      1301 Dec 14 11:03 sessioUnix.tex~
```

on podem veure els permisos i altres peculiaritats de l'arxiu, el nom del seu propietari, el grup al que pertany, el tamany en bytes de l'arxiu, la data i hora de la darrera modificació que ha sofert, i finalment el seu nom.

`ls -F` *quinsArxius* Llistat en el que es veu el tipus dels arxius: els directoris acabaran en /, els executables en *, ... Exemple:

```
exfin.aux        exfin.ps         exfin92fig2.ps   telf*
exfin.dvi        exfin.tex        subdir/          texput.log
exfin.log        exfin92fig1.ps   telefons
```

`ls -a` *quinsArxius* Llistat en el que es mostren també els arxius el nom dels quals comença amb un punt, que normalment no es llisten.

Es poden combinar aquestes opcions en una mateixa ordre. Per exemple, `ls -alF` demana un llistat de tots els arxius, (inclosos els que comencen amb un punt) llarg i amb senyals de tipus. Una combinació que m'agrada és `ls -altrF` que dona un llistat com aquest, ordenat segons la data de modificació, amb els més nous al final.

`cd` Canviar de directori actual.

`cd` a seques ens porta al *home directory*.
`cd ..` (atenció a l'espai) ens porta al directori pare.
`cd` *nom* ens porta al directori *nom*.

`cat` i `more` Per veure un arxiu. `grep` per mirar-hi a dins.

`cat` *nomDarxiu* ens mostra l'arxiu en pantalla. Si és molt llarg no tindrem temps de veure res.
`more` *nomDarxiu* ens el mostra, aturant-se a cada pantalla. Quan s'atura podem picar ESPAI per veure la següent pantalla, o `q` per sortir (quit).
Si volem saber quin és l'arxiu que conté tal *paraula*, podem picar
`grep` *paraula quinsarxius*
Exemples:

```
grep show *ps
grep Rosa tele*
grep -i rosa tele*
```

Amb la opció `-i`, majúsules i minúscules són equivalents.

`cp` Per copiar (duplicar) un arxiu. $\mathcal{PERILL}$!

`mv` Per moure (canviar de lloc o de nom) un arxiu.$\mathcal{PERILL}$!

`rm` Per eliminar (al tanto!) un arxiu (o diversos, o tots). `rm -r` elimina un directori i tot el que hi ha a dins (!!).

Atenció Les tres ordres anteriors poden ser perilloses. UNIX no avisa abans de trepitjar un arxiu que ja existeix amb una nova còpia produïda per `cp` o `mv`, ni abans de eliminar-lo amb `rm`. Pot ser bo acostumar-se a utilitzar sempre la opció `-i` amb qualsevol de les tres ordres. Amb aquesta opció, se'ns avisarà abans de fer malbé un arxiu exsitent. Per exemple:

```
$ rm -i pats.ps
pats.ps: ? (y/n) n
```

`mkdir` Per crear un nou directori.

`chmod` Per canviar els permisos d'un arxiu. En UNIX, hi ha tres nivells de control de seguretat per als arxius: propietari, grup i altres. La situació més normal és que l'arxiu sigui propietat d'algún usuari, que pertany a un grup determinat (normalment, els usuaris normals pertanyen tots al grup `users`). Quan demanem `ls -l` i obtenim

```
-rw-------   1 udina    users          8 Dec 14 13:07 exfin.aux
-rw-------   1 udina    users       5336 Dec 14 13:07 exfin.dvi
-rw-------   1 udina    users       1147 Dec 14 13:07 exfin.log
-rw-r--r--   1 udina    users      72947 Dec 14 13:07 exfin.ps
-rw-------   1 udina    users       3969 Dec 14 13:07 exfin.tex
-rw-rw-rw-   1 udina    users      22621 Dec 14 13:07 exfin92fig1.ps
-rw-rw-rw-   1 udina    users      19336 Dec 14 13:07 exfin92fig2.ps
```

```
-rwxr-xr-x   1 udina    users         17 Dec 14 13:09 tel
-rw-r--r--   1 udina    users         63 Dec 14 13:11 telefons
-rw-------   1 udina    users        179 Dec 14 13:07 texput.log
```

estem veient els permisos de cada arxiu.

El propietari pot canviar els permisos amb la ordre
`chmod` *qui*$\pm$*què quinsArxius*
on *qui* es una combinació de cap o varies de les lletres `u` (user owner), `g` (group), `o` (other), i on *què* es una combinació de cap o diverses de les lletres `r` (read), `w` (write), `x` (execute). En lloc de $\pm$ pot anar $+$ (donar permís) o $-$ (treure permís).

Exemples:

```
chmod u+x tel
chmod ug+rw *ps
chmod o-r *tex
chmod -w *
```

Per imprimir. La ordre bàsica és `lp` Teclejant
`lp` *nomDarxiu*
hem d'aconseguir una còpia en paper de l'arxiu. Un altre problema és per quina impressora sortirà...

3 Altres qüestions d'interés

1. Connexions amb d'altres hosts. `telnet` i `ftp`.

   Quan estem treballant amb una màquina UNIX, podem saber el seu nom teclejant `hostname`.

   Si volem entrar a treballar amb un altre host, hem de fer

   `telnet` *nomDelHost*, on el nom pot ser senzill si la màquina es de la xarxa local de la UPF, per exemple `telnet sahara`, o haura de ser més complert si volem accedir a màquines d'altres llocs, per exemple `telnet ccuab1.uab.es`. Evidentment, haurem de tenir compte obert al host al que volem accedir.

   La manera estàndar de transferir arxius de una màquina a una altra es a través del programa `ftp` (*file transfer protocol*). A la nostra xarxa, funciona ftp per transferir arxius en les següents direccions:

   ```
   unix host <--> unix host
   unix host <--> macs
   unix host <--> pc
   ```

   Des de un PC, teclejeu, per exemple, `ftpd sahara`. Amb el Mac, heu d'obrir primer una sessió de telnet amb el host i des d'allà fer `ftp` *n.n.n.n*, on aquestes *n*'s són el número de xarxa del Mac (el podeu trobar en el menú ..., opció Send ip number.

   Quan esteu fent `ftp`, hi ha dos *hosts* involucrats, el *local host* i el *remote host*. Per exemple, si feu `ftpd` des d'un PC, el *local host* és el PC. Si feu ftp al Mac des de una finestra de telnet, el *remote host* és el Mac.

   Les principals ordres que podeu teclejar des de `ftp` són:

`dir` Veure el llistat del directori actual del *remote host.*

`cd` Canviar de directori en el *remote host.*

`lcd` Caviar de directori en el *local host.*

`get` Enviar un arxiu del remot al local.

`put` Enviar un arxiu del local al remot.

Un exemple de sessió `ftp`, on s'ha assenyalat amb `<<<<<<` el que tecleja l'usuari:

```
$ ftp xaloc                                                 <<<<<<<<<<<<
Connected to xaloc.upf.es.
220 xaloc FTP server (Version 16.2 Wed Oct 16 23:04:42 GMT 1991) ready.
Name (xaloc:udina): udina                                   <<<<<<<<<<<<
331 Password required for udina.
Password:                                                   <<<<<<<<<<<<

230 User udina logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> dir                                                    <<<<<<<<<<<<
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 512
-rw-rw-rw-   1 udina    users        502 Sep 22 13:21 .profile
-rw-------   1 udina    users       5134 Dec 15 15:10 .sh_history
drwx------   2 udina    mail          24 Sep 22 13:22 Mail
-rw-rw-rw-   1 udina    users       1535 Nov 13 11:18 file.m
-rw-rw-rw-   1 udina    users       7236 Nov 13 11:20 file.ps
-rw-rw-rw-   1 udina    users      19204 Nov 13 11:21 file2.ps
-rw-r--r--   1 udina    users       4128 Dec  9 11:04 sfk-menu.el
-rwxrwxrwx   1 udina    users        111 Nov 24 13:31 xemacs
226 Transfer complete.
ftp> cd npe                                                 <<<<<<<<<<<<
250 CWD command successful.
ftp> dir                                                    <<<<<<<<<<<<
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 198
drwxrwxrwx   2 udina    users       1024 Dec 15 15:07 dades
-rwxrwxrwx   1 udina    users      98304 Dec 15 15:07 npe
drwxrwxrwx   2 udina    users       1024 Dec 15 15:08 outs
-rw-r--r--   1 udina    users        962 Dec 15 15:07 script.kde1
226 Transfer complete.
ftp> get script.kde1                                       <<<<<<<<<<<<
200 PORT command successful.
150 Opening BINARY mode data connection for script.kde1 (962 bytes).
226 Transfer complete.
962 bytes received in 0.03 seconds (29.22 Kbytes/s)
```

```
ftp> put sessioUnix.tex                                    <<<<<<<<<<<<
200 PORT command successful.
150 Opening BINARY mode data connection for sessioUnix.tex.
226 Transfer complete.
13462 bytes sent in 0.03 seconds (501.70 Kbytes/s)
ftp> dir                                                   <<<<<<<<<<<<
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 222
drwxrwxrwx   2 udina    users       1024 Dec 15 15:07 dades
-rwxrwxrwx   1 udina    users      98304 Dec 15 15:07 npe
drwxrwxrwx   2 udina    users       1024 Dec 15 15:08 outs
-rw-r--r--   1 udina    users        962 Dec 15 15:07 script.kde1
-rw-r-----   1 udina    users      13462 Dec 17 12:50 sessioUnix.tex
226 Transfer complete.
ftp> quit                                                 <<<<<<<<<<<<
221 Goodbye.
$
```

2. X Windows, finestres en UNIX. Només es pot fer servir des de X terminals.

3. Scripts de shell. Connexions entre ordres de shell.

   Amb scripts de shell es poden crear petits programes que el shell executarà.

   Exemple: Tenim un arxiu anomenat `telefons` que conté les linies

```
Pepet 234 56 78
Lluis 456 93 02
Joana 345 02 93
Rosa 214 00 99
```

   Podem consultar-lo picant

```
grep Rosa telefons
```

   Si creem un arxiu que es digui `tel` i que contingui la línia

```
grep -i $1 telefons
```

   i li donem permisos d'execució amb la ordre

```
chmod +x tel
```

   podrem consultar els telèfons més còmodament picant

```
tel rosa
```

UNIX té mecanismes de interconnexió molt poderosos i versàtils entre els programes que executa. Només comentarem un exemple:

La ordre `cat` *arxiu* produeix un llistat de l'arxiu, que normalment va a parar a pantalla.

La ordre `mailx -s "missatge per a tu" pepet@upf.es` espera que teclejem el missatge i l'envia per e-mail a l'usuari pepet.

Si empalmem les dues ordres amb una *tuberia* de UNIX, (*unix pipe*), el que produeix la ordre `cat` alimenta a l'altra ordre:

```
cat sessioUnix.tex | mailx -s "un arxiu per tu" pepet@upf.es
```

i aconseguim que l'arxiu s'envii per e-mail a l'usuari pepet.

4. L'arxiu d'inicialització automàtica: `.profile`

   Quan un usuari entra en una màquina UNIX, el shell executa un arxiu anomenat `.profile` que està al *home directory* de l'usuari. Allà s'hi poden tenir una sèrie de ordres per fer que el shell es comporti segons les preferències personals o locals. No toqueu aquest arxiu si no sabeu bé el què esteu fent.

5. L'editor `vi`

   L'editor `vi` és un editor bastant infumable que té l'aventatge que és estàndar i està disponible a tots els sistemes UNIX.

   Per entrar en l'editor, piqueu `vi` *nomDarxiu*. El *nomDarxiu* pot ser un de nou que volem crear o un existent. Un cop en l'editor us podeu bellugar pel text amb les tecles de fletxa. Si no funcionen per algún problema de configuració de la terminal, podeu bellugar-vos amb les tecles `h`, `j`, `k`, `l`. Podeu esborrar caràcters amb la tecla `x`.

   Si voleu inserir caràcters nous en l'arxiu, heu de picar primer la tecla `i` per entrar en el *mode d'inserció*. Després, podeu teclejar tot el que voleu inserir. Quan acabeu d'inserir el nou text, **això és important!**, heu de sortir de *inserir* picant la tecla Escape.

   Si us cal, podreu aleshores moure el cursor a un altre lloc, esborrar el que calgui, etc...

   Per sortir de l'editor `vi`, heu de estar fora de *inserir* i teclejar `:wq` (si voleu guardar els canvis) o `:q` si no hi ha canvis. Si hi ha canvis que no voleu guardar en l'arxiu, heu de teclejar `:q!` per sortir.

   A la majoria de màquines UNIX de la Pompeu hi ha un altre editor, anomenat `emacs`, bastant més agradable de fer servir, però això ja serà tema d'una altra sessió.

6. Les variables d'entorn. La ordre `env`

   Quan s'està executant un programa UNIX, hi ha una sèrie de varibales que controlen el comportament d'alguns aspectes del sistema. No podem entrar ara en detalls, l'únic que val la pena saber de moment és que, picant `env` podeu obtenir-ne una llista com aquesta:

```
MANPATH=/usr/man:/usr/contrib/man:/usr/local/man
LANG=C
PATH=/bin:/usr/bin:/usr/contrib/bin:/usr/local/bin:/usr/bin/X11:.
COLUMNS=80
EDITOR=emacs
```

```
LOGNAME=udina
MAIL=/usr/mail/udina
USER=udina
DISPLAY=libiya:0.0
SHELL=/bin/ksh
HOME=/users/udina
TERM=hp
PWD=/users/udina/facultat/proves
TZ=WET0WETDST
LINES=48
```

7. Referències i fonts d'informació

   La principal font d'informació i ajuda sobre les ordres UNIX és la ordre `man`, de manual. Si teclejeu `man rm` obtindreu una sèrie de pantalles que podeu anar passant amb la barra d'espai (vegeu la ordre `more` en la secció 2.2). Això és el que obtindreu:

```
$ man rm
Reformatting entry.  Wait...
 rm(1)                                                                  rm(1)

 NAME
      rm - remove files or directories

 SYNOPSIS
      rm [-f|-i] [-Rr] file ...

 DESCRIPTION
      rm removes the entries for one or more files from a directory.  If an
      entry was the last link to the file, the file is destroyed.  Removal
      of a file requires write and search (execute) permission in its
      directory, but no permissions on the file itself.



... ...


    Options
      The following options are recognized:

           -f   Force each file or directory to be removed without prompting
                for confirmation, regardless of the permissions of the
                entry.  This option also suppresses diagnostic messages
                regarding non-existent operands.

                This option does not suppress any diagnostic messages other
                than those regarding non-existent operands.  To suppress all
                error message and interactive prompts, the -f option should
```

```
                be used while redirecting the standard error output to
                /dev/null.

           -i   Write a prompt to the standard error output requesting
                confirmation before removing each entry.  This option is
                ignored when used in conjunction with the -f option.

...
```

Realment, les *manual pages* de UNIX serveixen sobre tot quan ja saps de què va la cosa. . . .