Nivell d'usuari

Remo Suppi Boldrito

PID_00167522

Universitat Oberta de Catalunya
www.uoc.edu

Part del material està basat en una versió anterior elaborada pels autors següents: Joaquín López Sánchez-Montañés, Sofia Belles Ramos, Roger Baig i Viñas i Francesc Aulí Llinàs, editat sota dipòsit legal B-1.566-2008 i publicat sota la GNU Free Documentation License, versió 1.2.

Índex

Introducció

Com ja s'ha vist en el mòdul anterior, **GNU/Linux** és un dels termes emprats per a referir-se a la combinació del nucli (*kernel*) –equivalent des del punt de vista de prestacions i funcionalitat (i en alguns casos superior) a Unix– denominat **Linux**, i d'eines de sistema GNU, tot sota llicència GPL (Llicència Pública de GNU) i una altra sèrie de llicències lliures.

Malgrat que Linux és, en sentit estricte, el sistema operatiu, part fonamental de la interacció entre el nucli i l'usuari (o els programes d'aplicació), es maneja usualment amb les eines GNU, com per exemple l'intèrpret d'ordres o instruccions Bash, que permet la comunicació amb el nucli mitjançant un conjunt complet d'ordres i instruccions. Hi ha altres nuclis disponibles per al projecte GNU, com el Hurd, que molts desenvolupadors consideren que és l'autèntic nucli del projecte GNU. A http://www.linuxdriver.co.il/kernel_map es pot consultar un mapa interactiu del nucli de Unix, en què es demostra la complexitat que té un sistema d'aquestes característiques.

En aquest mòdul veurem conceptes de nivell 0 per a aprendre des de l'inici els diferents conceptes de GNU/Linux, i anirem avançant fins a veure aspectes d'inicialització i configuracions per a adequar el sistema operatiu a les nostres necessitats.

1. Introducció al sistema GNU/Linux

El sistema GNU/Linux és un sistema multitasca, és a dir, permet l'execució de centenars de tasques al mateix temps independentment de la quantitat de *cors* o processadors o CPU que tingui per a executar-se, i utilitza una característica comuna de tots els sistemes operatius moderns anomenada *multiprogramació*. Aquesta característica permet executar una tasca durant un determinat temps, suspendre-la, passar a la següent i així successivament, i quan s'arriba al final, tornar a començar per la primera sense afectar el seu comportament o execució.

Normalment aquesta execució es denomina *round robin*, ja que distribueix un quàntum de temps (que oscil·la entre 15 mil·lisegons i 150 mil·lisegons, depenent de l'operatiu) per a cada tasca en espera, i torna a començar per la primera quan s'arriba a l'última de la cua. Si el sistema té més d'un *core* o processador, GNU/Linux té capacitat per a distribuir aquestes tasques en els diferents elements de còmput, i obtenir les conseqüents millores en les prestacions. Pot semblar que 15 mil·lisegons (= 0,015 segons) és un temps petit, però qualsevol processador actual pot executar com a mínim en aquest temps vora 2,2 milions d'instruccions màquina! A més, GNU/Linux és un sistema operatiu multiusuari que permet que més d'un usuari alhora pugui estar treballant amb el sistema, i que aquest amb el seu treball no pugui afectar cap de les tasques dels altres usuaris en el sistema.

> GNU/LINUX és un sistema multitasca i multiusuari que permet (fins i tot amb un sol processador) atendre les necessitats simultànies de múltiples usuaris, i utilitza una tècnica habitual dels sistemes operatius moderns denominada *multiprogramació*.

Tots els sistemes Unix, i GNU/Linux no és una excepció, consideren dos tipus d'usuaris diferenciats: el **superusuari** (anomenat **root**), que té tots els permisos sobre el sistema, i la resta dels usuaris, que disposen d'un directori de treball (*home*), del qual tenen tots els permisos, però en la resta del sistema el que poden fer està en funció de la seva importància i nivell de seguretat per al sistema mateix. Generalment, en qualsevol d'aquests sistemes ***nix** (Unix, Linux...) un usuari pot "mirar" (i en alguns casos executar) tot allò que no impliqui informació confidencial, però té generalment restringides la resta d'accions.

El segon concepte interessant en els sistemes *nix és que es pot treballar interactivament amb dos modes diferenciats: en mode text i en mode gràfic (i en aquest últim es pot obrir una finestra especial anomenada *terminal*, que permet treballar en mode text dins del mode gràfic). Normalment, els modes

gràfics són els més utilitzats en sistemes d'escriptori o d'usuari domèstic, mentre que els de text són adequats per a servidors. No obstant això, com que no s'imposa cap restricció, es pot canviar fàcilment d'un a l'altre amb una seqüència de tecles o fins i tot estar en mode gràfic i desenvolupar codi en mode text sobre un terminal, o connectat amb un terminal a una altra màquina o al disc d'una altra màquina.

El tercer concepte interessant és que la interacció entre l'usuari i el nucli es fa per mitjà d'un intèrpret d'ordres anomenat *shell*, que pot ser escollit per l'usuari entre uns quants. Aquests *shells* permeten l'execució (interpretació, més ben dit, ja que el codi està escrit en llenguatge text ASCII) de petits programes anomenats *shell scripts*, que són molt potents per a executar seqüències d'instruccions (i que poden arribar a ser molt complexes). Òbviament, igual com altres sistemes operatius, GNU/Linux en la seva interfície gràfica suporta la interacció gràfica sobre les diferents accions del sistema, i permet executar *shell scripts* com si es tractés d'altres programes.

Dos conceptes interessants més en els sistemes *nix són la idea de tasca d'usuari o del sistema operatiu i l'estructura del sistema d'arxius. Quant al primer concepte, una tasca és una activitat que ha de fer el sistema operatiu, que pot ser l'execució d'una instrucció, una ordre, editar un arxiu, etc. Per a això, el sistema operatiu ha d'executar un programa adequat per a fer aquesta tasca, que normalment es denomina *executable*, ja que conté les instruccions màquina per a fer-la. Quan el programa executable es carrega en memòria s'anomena *procés* o programa en execució, ja que conté, a més de l'executable, totes les estructures de dades perquè es pugui fer aquesta tasca, i s'allibera tota la memòria quan finalitza l'execució.

Aquest procés es pot suspendre, bloquejar, o se'n pot continuar l'execució d'acord amb les necessitats de l'usuari i amb el que ordeni el sistema operatiu. Una derivació d'aquest concepte de procés és que en els processadors moderns un procés pot ser dividit entre diverses subtasques (anomenats fils d'execució o *threads*). Quins avantatges té un programa multifil? Que el codi que executa cada fil el defineix el programador, i per això es pot tenir un fil atenent una lectura de disc i un altre fent un refresc d'una imatge en pantalla simultàniament dins del mateix procés. Fins i tot tenint un sol processador, aquest tipus de programació és més eficient que no si fa una subtasca primer i una altra després.

Nota

Els fils d'execució o *threads* poden ser executats independentment per diferents processadors.

Finalment, els sistemes *nix disposen d'una estructura d'arxius estàndard en què s'ubiquen els arxius del sistema amb independència total dels dispositius físics. És a dir, a partir d'una arrel (anomenada *root* i definida per la barra /) s'ubiquen els diferents directoris en funció dels seus objectius, i aquí cada usuari disposa d'un directori de treball propi, generalment en el directori /home/nom-usuari, en què el propietari té capacitat de decisió total, mentre que no és així en la resta de l'arbre (el superusuari té el seu directori en /root).

2. Conceptes i ordres bàsiques

Entrant en una mica més de detall, en aquest apartat discutirem les idees bàsiques i les instruccions necessàries per a "moure'ns" en el sistema. La manera més simple és per mitjà d'ordres en l'intèrpret d'ordres (i ja veureu com és el més eficient quan s'adquireix una mica de pràctica, encara que al principi pugui semblar antiquat o complicat, o totes dues coses). Aquest mètode de treball ens permetrà treballar ràpidament amb qualsevol màquina de manera local o remota, ja que hi ha repetició i text predictiu de les instruccions, i es poden executar ordres molt complexes o programar *shell scripts* simplement amb un terminal de text. S'ha de tenir en compte que una interfície gràfica és summament útil per a un usuari novell, però extremadament ineficient per a un usuari avançat.

La majoria de les ordres, o instruccions, que veurem en aquest apartat, formen part de l'estàndard (normes IEEE POSIX) i són comunes a tots els sistemes GNU/Linux i Unix. Encara que cada distribució té les seves aplicacions d'administració i gestió pròpies, generalment totes les accions que s'hi fan també es poden fer amb les ordres que veurem. A partir d'aquestes ordres, podrem manipular gairebé tots els aspectes del sistema i moure'ns eficientment. En aquest apartat tenim per objectiu aprendre a utilitzar correctament aquestes ordres i a navegar per qualsevol sistema basat en GNU/Linux independentment de quina distribució usem. Cada una de les ordres del sistema sol tenir multitud de paràmetres diferents. Amb la utilització dels paràmetres podem, amb una mateixa ordre, executar accions diferents, encara que totes siguin d'un mateix estil. En aquest document no especificarem els diferents paràmetres de cada una de les ordres que veurem, ja que es pot consultar el manual inclòs en tot sistema *nix amb l'ordre **man <nom_ordre>**. És interessant comentar que si no se sap el nom de l'ordre es pot utilitzar la instrucció **apropos acció**, que ens farà una llista de totes les ordres, i en la qual la paraula passada com a acció surt en l'especificació de l'ordre. Per exemple, si posem **apropos copy** ens donarà:

```
cp (1) - copy files and directories
cpgr (8) - copy with locking the given file to the password or gr...
cpio (1) - copy files to and from archives
cppw (8) - copy with locking the given file to the password or gr...
dd (1) - convert and copy a file
...
```

Això indica les instruccions que permeten copiar algun element. El paràmetre d'una ordre està precedit per un espai o moltes vegades per un guionet, com per exemple:

```
cp -dpR /home/juan /usr/local/backup
```

Això permet fer una còpia de suport dels arxius de /home/juan en /usr/local/backup, i amb *-d* indiquem que copiï els enllaços simbòlics tal com són, en lloc de copiar els arxius als quals apunten; amb *-p* preserva els permisos, l'usuari i el grup de l'arxiu per copiar, i amb *-R* copia els directoris recursivament.

2.1. Usuari i grups

Com hem dit en la introducció, tots els *nix són multiusuari i multitasca. Per aquest motiu és molt important que el sistema operatiu mateix incorpori mecanismes per a manipular i controlar correctament els usuaris: el sistema d'entrada i identificació (*login* o connexió), els programes que pot executar, els mecanismes de seguretat per a protegir el maquinari de l'ordinador, protecció per als fitxers dels usuaris, etc.

Per a identificar els usuaris davant del sistema operatiu, generalment s'utilitza una política de noms estàndard, que sol ser posar com a nom d'entrada la primera inicial del nom de l'usuari seguit del seu cognom. Els sistemes *nix organitzen tota aquesta informació per usuaris i grups i cal recordar que es diferencien majúscules i minúscules i, per tant, *abc* és diferent de *ABC*. Per a entrar a treballar interactivament amb el sistema, ens demanarà una connexió i una contrasenya.

L'inici de sessió sol ser un nom que identifica de manera inequívoca l'usuari, i si bé hi ha altres mètodes d'identificació, per exemple, mitjançant certificats digitals, el primer és el mètode més habitual. Per a validar la connexió se sol·licita una paraula que només coneix l'usuari, que s'anomena *contrasenya* (*password*, en anglès). La contrasenya ha de ser una combinació de lletres, nombres i caràcters especials, i no ha de ser cap paraula de diccionari o similars, perquè pot representar un problema de seguretat important.

Exemple

Per exemple, si posem una contrasenya com .MBAqcytvav34 podria semblar impossible de recordar fàcilment, però és un punt i les primeres lletres del tango *Mi Buenos Aires querido cuando yo te vuelva a ver* i l'any que es va escriure, i forma una paraula clau que serà molt més complexa d'esbrinar que una paraula que sigui al diccionari, ja sigui escrita en l'ordre normal o al revés.

Nota

Hi ha diferents tipus d'instal·lacions d'un sistema operatiu en funció de l'objectiu o paper que complirà i les aplicacions que tindrà instal·lades: servidor, escriptori, ofimàtica, lleure, passarel·la, tallafocs, etc. Un **servidor** és aquella màquina que conté programes que s'encarreguen de proporcionar algun tipus de servei (com servir pàgines web, deixar que

els usuaris es connectin remotament, etc.), que en la majoria dels casos estan vinculats a una xarxa de comunicacions, però no necessàriament.

El sistema de contrasenyes és de tipus unidireccional, la qual cosa significa que la nostra contrasenya no és emmagatzemada com a text, sinó que és xifrada i guardada. Quan entrem en el sistema i escrivim la nostra contrasenya, es xifra i es compara amb la que hi ha emmagatzemada. Si coincideixen, la identificació és positiva, si no coincideixen, és negativa. La seguretat del sistema es basa en l'algoritme de xifratge, del qual, si és segur, no es podrà aconseguir la clau original fent el procediment invers. Els programes que intenten trencar les contrasenyes dels usuaris fan milions de xifratges de paraules a partir de diccionaris (amb sistemes automàtics per a derivar-les i buscar variants) i provar si coincideixen amb el xifratge d'alguna de les contrasenyes d'usuari. Per aquest motiu, s'han d'escollir acuradament les contrasenyes, i tenir cura d'on es desen o apunten (cal intentar no fer-ho).

Actualment, en els sistemes GNU/Linux podem seleccionar dos tipus de xifratge possibles per a les contrasenyes d'usuari. El que s'utilitza des dels inicis de Unix és el 3DES. L'únic inconvenient d'aquest tipus de xifratge és que només ens permet contrasenyes de 8 lletres (si escrivim més, s'ignoren), a diferència de l'altre tipus de xifratge, denominat *MD5*, amb el qual podem usar contrasenyes de la longitud que vulguem (de fet, MD5 és un sistema de funció resum o *hashing*, però també es pot utilitzar per a xifrar contrasenyes de manera unidireccional). Com més llarga sigui la contrasenya, resulta més segura, i per això es recomana utilitzar el segon tipus de xifratge en sistemes d'alta seguretat. De tota manera, hem de considerar que, si necessitem usar alguns programes especials per a la gestió d'usuaris, com el NIS, pot ser que no siguin compatibles amb MD5.

Nota

NIS són una sèrie d'aplicacions que ens permeten gestionar tots els usuaris d'una mateixa xarxa de manera centralitzada en un sol servidor.

Els grups d'usuaris són un conjunt d'usuaris amb accés al sistema que comparteixen unes mateixes característiques, de manera que ens és útil agrupar-los per a poder donar-los una sèrie de permisos especials en el sistema. Un usuari ha de pertànyer, almenys, a un grup, encara que pot ser de més d'un. El sistema també utilitza tot aquest mecanisme d'usuaris i grups per a gestionar els servidors d'aplicacions instal·lats i altres mecanismes. Per aquesta raó, a més dels usuaris reals, en un sistema hi haurà usuaris i grups que no tenen accés interactiu però estan vinculats a altres tasques que s'han de fer en l'operatiu.

Exemple

Per exemple, l'usuari Debian-Exim és un usuari definit per a les funcionalitats del gestor de correu Exim que té ID d'usuari (101) i de grup (105), però que no es pot connectar interactivament (tal com ho indica el /bin/false de la definició de l'usuari en l'arxiu /etc/passwd).

```
Debian-exim:x:101:105::/var/spool/exim4:/bin/false.
```

Com ja hem explicat, en tot sistema operatiu hi ha un superusuari (*root*) que té privilegis màxims per a efectuar qualsevol operació sobre el sistema. És necessari que existeixi, ja que serà qui s'encarregarà de tota l'administració i gestió

de servidors, grups, etc. Aquest usuari no s'ha d'utilitzar per a treballar normalment en el sistema. Només haurem d'entrar com a *root* quan sigui realment necessari (actualment, la majoria de les distribucions no permeten entrar com a *root*, sinó entrar com a usuari normal i després "elevar-se" com a *root* per mitjà de la instrucció **su** o executar instruccions amb l'ordre **sudo**) i utilitzarem altres usuaris per al treball normal. D'aquesta manera, mai no podrem danyar el sistema amb operacions errònies o provant programes que poden ser maliciosos, etc.

Tota la informació d'usuaris i grups es troba als arxius següents:

- /etc/passwd: informació (nom, directori *home*, etc.) de l'usuari.
- /etc/group: informació sobre els grups d'usuaris.
- /etc/shadow: contrasenyes xifrades dels usuaris i configuració per a la validació, canvi, etc.

Utilitzar l'arxiu *shadow* és opcional però recomanable, ja que és un arxiu que només pot llegir *root;* per tant, la resta d'usuaris no tindrà accés a les contrasenyes xifrades i es redueix així la probabilitat que un usuari normal pugui utilitzar eines de força bruta per a esbrinar les contrasenyes d'altres usuaris o de *root*.

Nota

Informació d'usuaris i grups:
- /etc/passwd: informació dels usuaris.
- /etc/group: informació sobre els grups d'usuaris.
- /etc/shadow: contrasenyes xifrades dels usuaris i configuració per a la validació, canvi, etc.

Tots aquests fitxers estan organitzats per línies, cada una de les quals identifica un usuari o grup (depenent del fitxer). En cada línia hi ha diversos camps separats pel caràcter ":", i és important saber què són aquests camps, per la qual cosa els explorarem amb una mica més de detall:

1) ***/etc/passwd***. Per exemple, *games:x:5:60:games:/usr/games:/bin/sh*.

- Camp 1. Connexió: el nom de l'usuari. No hi pot haver dos noms iguals, encara que algun pot coincidir amb un grup del sistema.

- Camp 2. Contrasenya xifrada: si no s'utilitza el fitxer *shadow*, les contrasenyes xifrades s'emmagatzemen en aquest camp. Si utilitzem el fitxer *shadow*, tots els usuaris existents han d'estar també en el *shadow* i en aquest camp s'identifica amb el caràcter *x*.

- Camp 3. ID d'usuari: número d'identificació de l'usuari. És el número amb el qual el sistema identifica l'usuari. El 0 és el reservat per al *root*.

- Camp 4. ID del grup: el número de grup al qual pertany l'usuari. Com que un usuari pot pertànyer a més d'un grup, aquest grup es denomina *primari*.

- Camp 5. Comentaris: camp reservat per a introduir els comentaris sobre l'usuari. Se sol utilitzar per a posar el nom complet o algun tipus d'identificació personal.

- Camp 6. Directori d'usuari: el directori de l'usuari, que és on pot deixar tots els seus fitxers. Se solen posar en una carpeta del sistema (generalment /home/login-usuari).

- Camp 7. Intèrpret d'ordres: un intèrpret d'ordres (*shell*) és un programa que s'encarrega de llegir tot el que escrivim amb el teclat i executar els programes o ordres que indiquem. En el món GNU/linux el més utilitzat és el *Bash* (GNU Bourne again shell), si bé n'hi ha altres (*csh*, *ksh*, etc.). Si en aquest camp escrivim /bin/false, no permetrem que l'usuari executi cap ordre en el sistema, encara que estigui donat d'alta.

2) /etc/group. Per exemple, *netdev:x:110:Debian*.

- Camp 1. Nom del grup.

- Camp 2. Contrasenya xifrada: la contrasenya d'un grup s'utilitza per a permetre que els usuaris d'un determinat grup es puguin canviar a un altre o per a executar alguns programes amb permisos d'un altre grup (sempre que es disposi de la contrasenya).

- Camp 3. ID de grup: número d'identificació del grup. És el número amb el qual el sistema identifica internament els grups. El 0 està reservat per al grup de *root* (els administradors).

- Camp 4. Llista d'usuaris: els noms dels usuaris que pertanyen al grup, separats per comes. Encara que tots els usuaris han de pertànyer a un grup determinat (especificat en el quart camp del fitxer passwd), aquest camp es pot utilitzar perquè usuaris d'altres grups també disposin dels mateixos permisos que té l'usuari a qui s'està fent referència.

3) /etc/shadow. Per exemple,
root:1oE9iKzko$n9imGERnPDayiyat0XQjm.:14409:0:99999:7:::

- Camp 1. Connexió: ha de ser el mateix nom que s'utilitza en el fitxer /etc/passwd.
- Camp 2. Contrasenya xifrada.
- Camp 3. Dies que han passat, des de l'1/1/1970, fins que la contrasenya ha estat canviada per última vegada.
- Camp 4. Dies que han de passar fins que la contrasenya es pugui canviar.
- Camp 5. Dies que han de passar fins que la contrasenya s'hagi de canviar.
- Camp 6. Dies abans que caduqui la contrasenya s'avisarà l'usuari que l'ha de canviar.
- Camp 7. Dies que poden passar després que la contrasenya caduqui abans de deshabilitar el compte de l'usuari (si no es canvia la contrasenya).
- Camp 8. Dies, des de l'1/1/1970, que el compte és vàlid, i passats aquests dies serà deshabilitat.

- Camp 9. Reservat.

Quan un usuari entra en el sistema, se situa en el seu directori personal i s'executa l'intèrpret d'ordres (*shell*) configurat en /etc/passwd en la línia corresponent a l'usuari. D'aquesta manera, ja pot començar a treballar. Només el *root* del sistema (o els usuaris del seu grup) tenen permís per a manipular la informació dels usuaris i grups, donar-los d'alta, de baixa, etc. Cada ordre per a manejar els usuaris té diversos paràmetres diferents per a gestionar tots els camps que hem vist anteriorment.

A tall d'exemple, podem esmentar:

- **adduser:** ens serveix per a afegir un nou usuari al sistema. La manera com s'afegeix (si no li especifiquem res) es pot configurar en el fitxer /etc/adduser.conf. Admet un conjunt d'opcions diferents per a especificar el directori d'usuari, l'intèrpret que utilitzarà, etc.

- **useradd:** crea un nou usuari o canvia la configuració per defecte. Aquesta ordre i l'anterior ens poden servir per a fer les mateixes accions, encara que amb diferents paràmetres.

- **usermod:** amb aquesta ordre podem modificar la majoria dels camps que es troben en els fitxers passwd i shadow, com el directori personal, l'intèrpret, la caducitat de la contrasenya, etc.

- **chfn:** canvia la informació personal de l'usuari, continguda en el camp de comentaris del fitxer passwd (campos).

- **chsh:** canvia l'intèrpret d'ordres de l'usuari.

- **deluser:** elimina un usuari del sistema, i esborra tots els seus fitxers segons els paràmetres que li passem, fa còpia de seguretat o no, etc. La configuració que s'utilitzarà per defecte amb aquesta ordre s'especifica en el fitxer /etc/deluser.conf.

- **userdel:** ordre amb les mateixes possibilitats que l'anterior.

- **passwd:** serveix per a canviar la contrasenya d'un usuari, la informació de caducitat, o per a bloquejar o desbloquejar un determinat compte.

- **addgroup:** permet afegir un grup al sistema.

- **groupadd:** el mateix que l'ordre anterior, però amb diferents paràmetres.

- **groupmod:** permet modificar la informació (nom i GID) d'un grup determinat.

Nota

Ordres bàsiques de gestió d'usuaris i grups:

- useradd
- usermod
- chfn
- chsh
- deluser
- userdel
- passwd
- addgroup
- grouppadd
- groupmodd
- delgroup
- groupdel
- gpasswd
- groups

- **delgroup:** elimina un grup determinat. Si algun usuari el té com a primari, no es podrà eliminar.

- **groupdel:** igual que en el cas anterior.

- **gpasswd:** serveix per a canviar la contrasenya del grup. Per a saber quin usuari som, podem utilitzar l'ordre **whoami**, que ens mostrarà el nostre nom d'entrada.

- **groups:** serveix per a saber a quins grups pertanyem, i **id** ens mostrarà l'usuari i els grups.

També és interessant poder convertir-nos en un altre usuari sense sortir de la sessió (ordre **login** o **su**) o canviar-nos de grup amb l'ordre **newgrp.** Aquesta última ordre s'ha d'utilitzar només quan no es pertany al grup en qüestió i se sap la seva contrasenya (que ha d'estar activada en el fitxer group). Si només necessitem els permisos del grup en qüestió per a executar una ordre determinada, també podem utilitzar **sg.**

Com veiem, en GNU/Linux tenim més d'una manera per a executar una acció determinada. Aquesta és la tònica general que se segueix en el sistema: podem editar directament els fitxers i modificar-los, utilitzar algunes de les ordres que hi ha, crear-les nosaltres mateixos, etc. En definitiva, tenim la possibilitat de seleccionar quina és l'opció que més ens satisfà.

D'altra banda, i com dèiem anteriorment, GNU/Linux és un sistema operatiu multiusuari, per la qual cosa en un mateix moment hi pot haver diversos usuaris connectats al sistema de manera simultània. Per a esbrinar qui són, es pot utilitzar l'ordre **who**, que ens mostra la llista d'usuaris dins del sistema; **w**, a més, ens mostra què és el que estan fent. Ens podem comunicar amb un altre usuari utilitzant l'ordre **write**, amb la qual apareix el missatge que hem escrit a la pantalla de l'usuari indicat o **wall**, que escriu el contingut del fitxer que hem especificat en tots els usuaris dins del sistema. Per a activar o desactivar l'opció de rebre missatges, tenim l'ordre **mesg.** També podem fer un xat personal amb algun usuari a partir de l'ordre **talk.**

2.2. El sistema de fitxers i la jerarquia

Tot sistema operatiu necessita desar multitud d'arxius: configuració del sistema, registre d'activitats, d'usuaris, etc. Hi ha diferents sistemes d'arxius caracteritzats per la seva estructura, fiabilitat, arquitectura, rendiment, etc., i GNU/

Linux és capaç de llegir i escriure arxius en la gairebé totalitat dels sistemes d'arxius, encara que té els seus sistemes propis optimitzats per a les seves funcionalitats, com per exemple ext3 o ReiserFS.

L'ext4 és una millora compatible amb *ext3* que, al seu torn, és una evolució de l'ext2 que és el més típic i estès. El seu rendiment és molt bo, incorpora tot tipus de mecanismes de seguretat i adaptació, és molt fiable i incorpora una tecnologia denominada *journaling*, que permet recuperar fàcilment errors en el sistema quan, per exemple, hi ha un tall de llum o l'ordinador sofreix una parada no prevista. *ext4* suporta volums de fins a 1024 PiB (PiB pebibyte = 250 bytes ≈ 1.000.000.000.000.000 bytes), millora l'ús de la CPU i el temps de lectura i escriptura..

ReiserFS és un altre dels sistemes utilitzats en Linux que incorpora noves tecnologies de disseny, que li permeten obtenir prestacions i utilització de l'espai lliure més adequades. Qualsevol d'aquests tipus de sistemes d'arxius pot ser seleccionat en el procés d'instal·lació, i és recomanable ext3 per a la majoria de les instal·lacions.

Una característica molt important de tots els sistemes *nix és que tots els dispositius del sistema es poden tractar com si fossin arxius (independència del dispositiu físic). És a dir, hi ha el procediment de "muntar el dispositiu" per mitjà de l'ordre **mount** en un directori del sistema i després, accedint a aquest directori, s'accedeix al dispositiu (fins i tot si el dispositiu és remot), per la qual cosa no hi ha una identificació del dispositiu físic per a treballar amb els fitxers com en altres sistemes operatius, com A:, C:, etc.).

El sistema de fitxers ext2/3/4 ha estat dissenyat per a manejar de manera òptima fitxers petits (els més comuns) i de manera acceptable els fitxers grans (per exemple, arxius multimèdia) si bé es poden configurar els paràmetres del sistema de fitxers per a optimitzar el treball amb aquest tipus d'arxius. Com hem esmentat anteriorment, el sistema d'arxius parteix d'una arrel indicada amb /, i les carpetes (directoris) i subcarpetes (subdirectoris) s'organitzen a partir d'aquesta, de manera que presenta una organització jeràrquica (visualitzada amb l'ordre *tree -L 1*) com:

```
/
|-- bin
|-- boot
|-- cdrom -> media/cdrom
|-- dev
|-- etc
|-- home
|-- initrd.img -> boot/initrd.img-2.6.26-2-686
|-- lib
|-- lost+found
```

```
|-- media
|-- mnt
|-- opt
|-- proc
|-- root
|-- sbin
|-- selinux
|-- srv
|-- sys
|-- tmp
|-- usr
|-- var
`-- vmlinuz -> boot/vmlinuz-2.6.26-2-686
```

Aquí es mostra el primer nivell de directoris a partir del /. Tots són directoris, excepte els que figuren amb el caràcter "->", que són enllaços a arxius (l'arxiu original es troba a la dreta de la fletxa).

2.3. Directoris del sistema

En les distribucions GNU/Linux se segueix l'estàndard FHS, i tenim com a directoris en l'arrel (els més importants):

- **/bin:** ordres bàsiques per a tots els usuaris del sistema.
- **/boot:** arxius necessaris per a l'arrencada del sistema.
- **/dev:** dispositius del sistema.
- **/etc:** arxius de configuració del sistema i de les aplicacions que hi ha instal·lades.
- **/home:** directoris personals dels usuaris.
- **/lib:** biblioteques essencials per al nucli del sistema i els seus mòduls.
- **/mnt:** punt de muntatge temporal per a dispositius.
- **/proc:** processos i variables del nucli del sistema.
- **/root:** directori personal per al *root* del sistema.
- **/sbin:** instruccions especials per al *root* del sistema.
- **/tmp:** arxius temporals.
- **/usr:** segona estructura jeràrquica, utilitzada per a emmagatzemar tot el programari instal·lat en el sistema.
- **/var:** directori per als gestors de cues o *spoolers* d'impressió, arxius de registre (*logs*), etc.

No s'han d'esborrar aquests directoris, malgrat que sembli que no s'utilitzen, per al bon funcionament del sistema (moltes aplicacions poden no instal·lar-se o donar errors si els directoris estàndard no estan definits).

Per a moure'ns per l'estructura de directoris hem d'utilitzar les ordres per a fer una llista dels continguts i canviar de carpeta. Quan entrem en el sistema, és usual que la connexió ens situï en el nostre directori personal, que generalment se sol indicar amb el caràcter "~". Si volem veure el que hi ha en el directori en el qual estem situats, podem fer una llista dels continguts utilitzant l'ordre **ls** o, en la seva versió més completa, **ls -la**, que implica tots els fitxers (incloent-hi els que comencen per un ".", que no es mostren normalment) amb *-a*, i en format llarg amb *-l*. En tots els directoris hi ha dues entrades indicades amb "." i ".."; la primera fa referència al directori actual i la segona al directori superior.

Exemple

Per exemple, si fem **ls .** mostrarà la llista del directori actual i si fem **ls ..** mostrarà la llista del directori immediatament superior.

Per a canviar de directori podem utilitzar l'ordre **cd**, la qual, si no li passem cap paràmetre, ens situarà en el directori personal de l'usuari que l'ha executada. Totes les ordres accepten adreces relatives; per exemple, *ls ./grub* si estem en el directori /boot ens mostrarà el contingut del directori /boot/grub (forma relativa) o *ls /boot/grub* també ens mostrarà el contingut del directori /boot/grub, però des de qualsevol lloc on estiguem (forma absoluta). Una altra ordre útil per a saber on estem aturats és **pwd**, que ens indicarà en quin directori som.

2.4. Enllaços

Un element molt utilitzat en els arxius són els enllaços o vincles. Un enllaç és un pont a un arxiu o directori i representa una referència que podem posar en qualsevol lloc que ens interessi, i actua com un accés directe a qualsevol altre.

Aquest mecanisme ens permet accedir a carpetes o fitxers de manera segura i còmoda, sense haver-nos de desplaçar per la jerarquia de directoris.

Exemple

Si necessitem accedir freqüentment a l'arxiu /etc/network/if-up/mountnfs, per exemple, podem utilitzar un enllaç en el nostre directori amb l'ordre *ln -s /etc/network/if-up/mountnfs nfs-conf*, i fent un *cat nfs-conf* tindrem el mateix resultat que fent *cat /etc/network/if-up/mountnfs*, i així evitem haver d'introduir cada vegada tota la ruta completa.

En aquest cas hem creat un enllaç simbòlic paràmetre (-s) és a dir, que si esborrem l'arxiu l'enllaç quedarà apuntant a res i donarà un error quan executem l'ordre cat nfs-conf, però aquest tipus d'enllaç es pot fer en qualsevol recurs i en qualsevol de les particions del disc. L'altra possibilitat és fer un enllaç fort (*hard link*), permès només per a recursos en la mateixa partició; en aquest cas, si esborrem l'arxiu, l'enllaç queda actiu fins que no hi hagi més enllaços apuntant a aquest arxiu (moment en el qual s'esborrarà l'arxiu de destinació). Aquest recurs s'ha d'utilitzar amb compte (només el *root* pot fer enllaços forts a directoris), ja que permet ocultar a quin arxiu està apuntant, mentre que amb un enllaç simbòlic es pot veure l'arxiu de destinació.

2.5. Permisos

Ja que els sistemes *nix són multiusuari, necessitem que els arxius emmagatzemats tinguin una sèrie de propietats que permetin llegir, escriure i executar (paràmetres *r, w, x –read, write, execute*). Per a això GNU/Linux pot treballar amb un mètode simplificat (anomenat *access control list reduïdes*) en què per a cada element en el sistema d'arxius es consideren tres bits (que representen els atributs per a *rwx*) per al propietari de l'element (*owner*), tres per al grup i tres per a la resta d'usuaris. Llavors, per a cada element (arxiu, directori, dispositiu, enllaç, etc.) hi ha 9 bits a més de la identificació de qui és l'amo de l'element (uid) i a quin grup pertany (gid). Quan fem *ls -l* tindrem per a cada element una sortida com la següent:

```
-rwxr-xr-x 1 root root 4297 2008-01-18 06:09 mountnfs
```

Els primers deu caràcters (començant per l'esquerra) ens indiquen els permisos del fitxer de la manera següent:

- Caràcter 1: indica el tipus d'arxiu; els més comuns són "-" per a un arxiu, *d* per a un directori, i *l* per a un enllaç.

- Caràcters 2, 3, 4: ens indiquen, respectivament, els permisos de lectura, escriptura i execució per al propietari del fitxer. En cas de no tenir el permís corresponent activat, hi ha el caràcter "-" i si no, *r, w* o *x*. En el tercer caràcter, a més, ens podem trobar una *s*, que ens indica si l'arxiu és de tipus SetUserId, que significa que en executar-lo obtindrà els permisos del propietari del fitxer. Si només té el permís *x*, quan el programa s'executa ho fa amb els permisos de qui l'hagi llançat.

- Caràcters 5, 6, 7: aquests caràcters tenen exactament el mateix significat que els anteriors, però fan referència als permisos concedits als usuaris del grup a què pertany l'arxiu.

- Caràcters 8, 9, 10: igual que en el cas anterior, però per als altres usuaris del sistema.

La xifra següent (1, en aquest cas) ens indica el nombre d'enllaços forts que té l'arxiu. Per als directoris, aquest nombre indica quantes carpetes hi ha en l'interior, a més dels enllaços forts que té. A continuació es troba el propietari i el grup de l'arxiu, seguit de la mida (en bytes) que ocupa i la data de l'última modificació. En tots els arxius es desa la data de creació, de l'últim accés i de l'última modificació, que podem manipular amb l'ordre **touch.** Al final es troba el nom del fitxer, en el qual es diferencien minúscules de majúscules, i aquí podem tenir tot tipus de caràcters sense cap problema (encara que després

serà més complicat executar ordres amb aquests caràcters, ja que s'haurà de posar entre cometes ("") perquè no s'interpretin). Es recomana utilitzar: a-z A-Z . - _ 0-9.

El mecanisme de SetUserId és molt útil quan un programa necessita tenir els permisos del seu propietari per a accedir a certs arxius o fer algun tipus d'operació en el sistema. S'ha de vigilar aquest tipus d'arxius perquè poden generar problemes de seguretat en el sistema si són mal utilitzats. Per a canviar els permisos d'un arxiu determinat podem utilitzar l'ordre **chmod**, i aquesta acció només la pot fer el propietari de l'arxiu. Hi ha dues maneres comunes d'utilitzar *chmod*. Una és: *chmod XXX nom_arxiu*, en què XXX està comprès entre 0 i 7, que corresponen al valor en octal dels permisos *rwx* per al propietari (primer número), el grup (segon) i públic (tercer). Per a treure el número hem de considerar que 1 vol dir *amb permís concedit*, i 0, *sense;* per exemple, *r-x* es tradueix com 101, que en octal (o binari) és 5. Per això, r-x--xr-- es tradueix com 101001100, que queda com 514.

Proteccions

Per a canviar les proteccions de file a *rwxr--r-*:
chmod 744 file
Per a canviar de propietari i grup:
chown user file
chgrp group file

L'altra manera d'utilitzar l'ordre és indicar de manera explícita quin permís volem donar o eliminar en l'arxiu, indicant amb les lletres *u, g, o* l'usuari, el grup o la resta, respectivament, un + per a agregar el permís i un – per a treure'l, i *r, w, x* o *s* (aquest últim per al SetUserId) per al permís en concret. Per exemple, *chmod go +r mountfs* concediria el permís de lectura al grup i als altres usuaris per a l'arxiu mountnfs. Perquè canviï el propietari d'un fitxer, s'utilitza l'ordre **chown**[1], i per a canviar el grup d'un arxiu es pot utilitzar l'ordre **chgrp.** Els permisos per defecte per als arxius es poden definir amb l'ordre *umask*, amb la mateixa notació que per a la primera forma del **chmod** però complementat (és a dir, si volem *rw-r- -r--* el valor hauria de ser 133).

[(1)]Aquesta ordre només la pot utilitzar el *root*, ja que un usuari podria fer una acció maliciosa i després canviar el propietari de l'arxiu i responsabilitzar un altre usuari de l'acció feta.

2.6. Manipulació, patrons, cerques i continguts

L'ordre **rm** permet eliminar els arxius, i per a eliminar un directori podem utilitzar l'ordre **rmdir**, encara que només l'esborrarà quan sigui buit (si volguéssim esborrar completament un directori i tot el seu contingut, podem utilitzar *rm -r*, que funciona de manera recursiva). Per a copiar arxius d'un lloc a un altre tenim l'ordre **cp**, indicant el fitxer o directori origen i el lloc o nom de destinació, encara que sigui en el directori actual (si volem moure un arxiu o directori, es pot utilitzar l'ordre **mv**).

Podem utilitzar modificadors en els noms dels arxius o directoris, com per exemple "*", per a referir-nos a qualsevol cadena, i "?" per a indicar un caràcter qualsevol. Per exemple, si volem mostrar tots els arxius que comencin per *a* i acabin per *x* caldrà executar *ls a*x*, però si volem mostrar tots els arxius de tres lletres que comencin per *a* i acabin per *x*, serà *ls a?x*. Es pot utilitzar "[]" per a una selecció de caràcters; per exemple *ls [Yy]** indicaria tots els arxius que comencin per *Y* o per *y*, i després qualsevol cosa. Podem agregar també

Caràcters modificadors

*: qualsevol cosa,
?: un caràcter
[Yy]: un o l'altre
[A-Z]: un rang
[:classe:]: una classe

"!", que significa la negació del que indiquem, com en *[!Yy]*, és a dir, que no comencin per *Y* o *y*. Finalment, per a facilitar certes cerques, dins de "[]" podem especificar classes de caràcters com [:classe:], que pot ser una de les següents:

- *alnum:* [A-Za-z0-9]
- *alpha:* [A-Za-z]
- *blank:* [\]
- *cntrl:* caràcters de control
- *digit:* [0-9A-Fae-f]
- *graph:* caràcters imprimibles (sense espais)
- *lower:* [a-z]
- *print:* caràcters imprimibles (amb espais)
- *punct:* [.,¡!¿?:;]
- *space:* []
- *upper:* [A-Z]
- *xdigit:* [0-9A-Fa-f]

Un altre tipus d'operació molt útil és la cerca d'arxius. Hi ha diverses ordres per a fer cerques de diferents tipus: **find** és l'ordre més versàtil per a buscar informació sobre els arxius o directoris (per nom, mida, data, proteccions, etc.), **locate** permet utilitzar una base de dades del sistema per a fer cerques més ràpides que el **find**, però s'ha de tenir en compte que pot donar informació no actualitzada i que es pot actualitzar amb **updatedb**, i **whereis** indica on es troba l'arxiu especificat.

Com els arxius poden ser de molts tipus (executables, text, dades, música, etc.) en els sistemes *nix no s'utilitza l'extensió per a identificar el tipus d'arxiu sinó un nombre intern a la capçalera de l'arxiu anomenat *magic number*, que determina el tipus d'arxiu segons les seves dades (es pot utilitzar l'ordre **file** per a llegir i determinar el tipus d'arxiu). Si necessitem veure el contingut d'un arxiu, una de les ordres bàsiques és **cat**, o **more** si l'arxiu és ASCII; llavors el mostrarà paginat. Si fem un **cat** d'un arxiu executable alguns dels caràcters poden desconfigurar el terminal i es pot reinicialitzar amb l'ordre **reset**, i **clear** per a esborrar la pantalla. L'ordre **less** ens permet moure'ns de manera més eficient (endavant i endarrere per l'arxiu). Si l'arxiu és binari i volem veure què conté, podem utilitzar les ordres **hexdump** per a veure el contingut de forma hexadecimal o **strings** per a buscar les cadenes de caràcters. A l'ordre **grep** li podem passar com a segon paràmetre el nom de l'arxiu, i com a primer, el patró que vulguem buscar (segons la sintaxi que hem vist anteriorment, estesa a altres opcions). A més, l'ordre ens permet múltiples accions més, com comptar el nombre de línies en les quals apareix el patró (paràmetre *-c*), etc. Amb **cut** podem separar en camps el contingut de cada línia del fitxer especificant quin caràcter és el separador, molt útil en tasques d'administració. També podem visualitzar un determinat nombre de línies del començament o del final d'un arxiu amb les ordres **head** i **tail**, respectivament, i amb **wc** podem comptar el nombre de línies o paraules, la màxima longitud de línia d'un fitxer, etc. Finalment, per a comparar diferents arxius hi ha diverses ordres per a fer-ho:

diff, **cmp** i **comm** fan comparacions de diferents maneres i mètodes en els fitxers que indiquem; **sdiff** a més permet barrejar les dades d'acord amb els paràmetres indicats.

2.7. Processos

Com hem dit en la introducció, els *nix són sistemes operatius multitasca que executen processos i fils (*threads*) mitjançant una tècnica anomenada *multiprogramació*, que permet executar més d'un procés o fil alhora de manera concurrent i més eficient que si l'executéssim seqüencialment, ja que es pot encavalcar l'execució d'entrada o sortida d'un procés amb l'execució en la CPU d'un altre procés. Per a identificar de manera inequívoca cada procés, el nucli del sistema els assigna un número denominat *PID* (*process identification*), necessari per a administrar i referenciar el procés.

Per a saber quins processos s'estan executant, podem utilitzar l'ordre **ps.** Una altra ordre interessant per a mirar l'execució interactiva dels processos és **top**, que mostra la càrrega i l'estat de sistema de manera dinàmica (cal prémer *q* –*quit*– per a sortir de la instrucció).

Ordre *ps*

Per exemple, *ps -edaf* mostra un conjunt d'informació sobre tots els processos en execució i en diferents estats.

A més d'això, podem enviar senyals als processos a fi d'informar-los d'algun esdeveniment, els podem treure de la cua d'execució, eliminar-los, donar-los més prioritat, etc. Saber manipular correctament tots aquests aspectes també és molt important, ja que ens permetrà utilitzar el nostre ordinador de manera més eficient. L'ordre **kill** ens permet enviar senyals als processos que ens interessin.

En general, tots els programes es dissenyen perquè puguin rebre aquest tipus de senyals (fins i tot els scripts poden capturar els senyals). D'aquesta manera, segons el tipus de senyal rebut saben que han de fer unes operacions o d'altres (per exemple, suspendre l'execució quan un usuari fa un Ctrl-D). Per a veure els tipus de senyals, consulteu *man* **kill**. **killall** és una ordre per a fer el mateix, però utilitza el nom del procés en lloc del PID, i **skill** és similar, però amb una sintaxi diferent: per exemple, per a detenir totes les execucions d'un usuari determinat, podríem utilitzar *skill -STOP -u login*, amb la qual cosa s'acabarà l'execució dels processos d'aquest usuari. A més de **Ctrl-D** o **Ctrl-C** per a finalitzar un procés (la primera és amb espera fins que el procés acabi el seu E/S i la segona és en el moment que la rep), amb **Ctrl-Z** podem interrompre un programa i reviure'l amb **fg.**

Nota

Ordres per a processos:

- *kill -9 <pid>*
- *skill -STOP -u <login>*
- *ps -edaf*
- *top*
- *pstree*
- *nice [-n increment] <ordre>*

Kill

Per exemple, amb el senyal TERM –que és 15– si fem *kill -15 PID*, que és equivalent a fer Ctrl-C en un procés interactiu sobre un terminal, indiquem al procés que volem que acabi, de manera que en rebre el senyal haurà de desar tot el necessari i acabar l'execució; si el programa no està preparat per a rebre aquest tipus de senyal, podem utilitzar la *-9* (que obeixen tots els processos) i acabar independentment del que estiguin fent amb *kill -9 PID*.

L'ordre **pstree** permet veure aquesta jerarquia de manera gràfica, i veurem que el pare de tots els processos s'anomena *init*, ja que és el procés inicial per a posar en marxa els processos restants del sistema, i a partir d'aquest neixen tots els altres, que al seu torn poden tenir més fills. Aquesta estructura és molt útil per a identificar d'on vénen els processos (qui els ha posat en marxa) i per a eliminar-ne un conjunt, ja que, en eliminar un procés pare, també s'eliminen tots els seus fills.

Un paràmetre important dels processos és un valor anomenat *prioritat*, que està relacionat amb la CPU que rebrà aquest procés (com més prioritat més temps de CPU). El rang de prioritats va des del -20 fins al 19 (les negatives només les pot utilitzar el *root*), de major a menor. Per a llançar un procés amb una prioritat determinada, podem utilitzar l'ordre **nice** i **renice**, si volem donar una prioritat diferent d'un procés que ja estigui en execució. Per defecte, la prioritat amb què s'executen els programes és la 0. L'ordre **time** permet calcular el temps que utilitza un procés per a executar-se (generalment amb finalitats comptables, per exemple si hem de facturar pel temps de CPU que gasta un procés).

2.8. Ordres complementàries

Totes les ordres disposen en GNU/Linux (i en tots els *nix) d'un manual complet que indica tots els paràmetres i opcions (***man ordre***), i s'utilitza l'ordre *less* per a visualitzar-les, i per això podem anar endavant i enrere amb les tecles d'avançar i retrocedir pàgina, buscar una paraula amb el caràcter "/" seguit de la paraula (*n* ens serveix per a buscar les aparicions següents i *N*, per a les anteriors), *q* per a sortir, etc. Els manuals del sistema es divideixen en diferents seccions segons la seva naturalesa:

1) Programes executables (aplicacions, ordres, etc.).
2) Crides al sistema proporcionades per l'intèrpret d'ordres.
3) Crides a biblioteques del sistema.
4) Arxius especials (generalment els de dispositiu).
5) Format dels arxius de configuració.
6) Jocs.
7) Paquets de macros.
8) Ordres d'administració del sistema (generalment les que només el *root* pot utilitzar).
9) Rutines del nucli.

Si hi ha més d'un manual disponible per a una mateixa paraula, el podem especificar indicant el número corresponent de la secció que ens interessa abans de la paraula; per exemple, *man 3 printf* (**man -k paraula** buscarà entre les pàgines del manual les que tinguin la "paraula" passada com a argument, equivalent a **apropos**, i **mandb** permetrà actualitzar la base de dades dels manuals).

Nota

Ordres complementàries:

a) Manuals:
- man <ordre>
- man -k <paraula>

b) Comprimir:
- tar cvf <destinació> <origen>
- tar zcvf <destinació> <origen>
- gzip <file>
- bzip <file>

c) Espai de disc:
- df -k
- du -k <directori/file>

d) Paràmetres de sistema d'arxius:
- dumpe2fs <partició>

e) Sincronitzar sistema d'arxius:
- sync

Si el manual no ens proporciona tota la informació que necessitem, podem utilitzar la instrucció **info**, que és el mateix que el manual però amb informació estesa.

Una altra ordre útil és la utilitzada per a comprimir un arxiu, agrupar-ne diversos en un de sol o veure què conté un arxiu comprimit. Si bé hi ha desenes de programes diferents en tots els sistemes GNU/Linux trobarem l'ordre **tar.** Aquest programa ens permet manipular de qualsevol manera un o diversos arxius per a comprimir-los, agrupar-los, etc. La seva sintaxi és ***tar opcions arxiu-destinació arxius-origen***, en què si l'arxiu origen és una carpeta hi treballarà de manera recursiva i desarà a l'arxiu de destinació el contingut de tota la carpeta. Els paràmetres comuns són *c* per a crear i *f* si ho volem desar en un arxiu (*tar cf arxiu.tar o**** empaquetarà tots els arxius del directori actual que comencin per *o*). Si a més volguéssim comprimir, podríem utilitzar *czf*, i llavors s'utilitzaria el programa *gzip* després d'empaquetar-los. Per a desempaquetar un arxiu determinat, el paràmetre necessari és *x*, de manera que hauríem d'escriure *tar xf* per a indicar l'arxiu empaquetat. Si estigués comprimit, hauríem de passar *xzf*.

El programa **gzip** (utilitzat pel *tar* per a comprimir) usa un format de compressió propi i diferent del popular *zip* o del *compress* (estàndard en els *nix però obsolet) i que es pot utilitzar per a comprimir un arxiu de manera independent (**gunzip** per a fer el procés invers o *gzip -u*). Una altra aplicació de compressió bastant utilitzada i que proporciona molt bons resultats és el **bzip2.**

La gestió i manipulació dels discos durs de l'ordinador és un altre aspecte fonamental en les tasques d'administració del sistema. Més endavant es veurà tot un apartat per a discos, però ara tractarem les ordres útils per a obtenir informació dels discos. El disc dur es divideix en particions, a les quals podem accedir com si es tractés d'un dispositiu independent, i les denominarem *unitat*. Això és molt útil perquè ens permet separar de manera adequada la informació que tinguem en el sistema, tenir més d'un sistema operatiu instal·lat al mateix disc, etc. L'ordre **df** ens mostrarà, de cada unitat muntada en el sistema, l'espai que s'ha utilitzat i el que queda lliure, i l'ordre **du** ens mostra realment el que ens ocupa un fitxer en disc o un directori (ens ho mostrarà en blocs de discos però amb el paràmetre *-k* en kilobytes).

Un disc és organitzat en pistes i dins de les pistes en sectors (zones on es desarà la informació) i com aquest últim valor és configurable, quan es crea el sistema d'arxius amb finalitats d'optimitzar les prestacions de disc o espai utilitzat ens pot interessar veure aquests paràmetres (per a sistemes ext2/3/4), per la qual cosa podem utilitzar l'ordre **dumpe2fs partició** i esbrinar els paràmetres amb els quals ha estat creat el disc.

Un altre concepte molt estès és la desfragmentació d'un disc, que no és més que la reorganització dels blocs dels fitxers perquè quedin en llocs consecutius i l'accés sigui més ràpid. En els sistemes de fitxers que utilitzem amb GNU/Linux no és necessari desfragmentar els discos (encara que hi ha programes amb

aquesta finalitat), ja que el sistema s'encarrega automàticament de tenir el disc sempre desfragmentat, i a més, en el procés d'arrencada sempre es comproven els errors i es fa una desfragmentació total si és necessària.

L'optimització del sistema d'arxius de GNU/Linux prové del fet que totes les funcions del nucli que s'encarreguen de la gestió de fitxers utilitzen uns mètodes per a agilitar els processos de lectura i escriptura. Un és la utilització d'una memòria cau de disc per a evitar estar llegint constantment i escrivint al disc físic (procés lent i costós). Això pot representar problemes si tenim un tall d'alimentació, ja que les últimes operacions de lectura/escriptura no s'hauran desat perquè són en memòria. El programa **fsck** comprova i arregla un sistema de fitxers que hagi quedat en aquest estat. Encara que el podem executar quan vulguem, el sistema operatiu mateix l'executa quan en el procés d'arrencada detecta que el sistema no es va tancar adequadament. Per això, per a apagar l'ordinador correctament hem d'executar l'ordre **shutdown**, que s'encarrega de llançar tots els processos necessaris perquè els programes acabin, es desmunti el sistema de fitxers, etc. En aquest sentit, el sistema de fitxers ext3/4, és més eficaç que l'ext2, ja que el *journaling* li permet recuperar més informació dels arxius perduts i de manera més eficient (la integritat física d'una partició es pot comprovar amb l'ordre **badblocks**). Si bé no és aconsellable, es pot desactivar la memòria cau de disc i si volem en algun moment bolcar de cau a disc, per a evitar problemes podem executar l'ordre **sync.**

S'ha de tenir en compte que la majoria de les ordres esmentades s'han d'executar com a *root* (o alguns dels usuaris que formin part del grup de *root*).

3. Instal·lació i arrencada de GNU/Linux (conceptes bàsics)

En aquest apartat veurem els passos essencials que se segueixen en la majoria dels processos d'instal·lació de GNU/Linux, i que seran complementats posteriorment amb els tallers d'instal·lació. Si bé cada distribució té el seu entorn d'instal·lació propi, en totes hi ha uns passos bàsics per a instal·lar el sistema operatiu, i que es descriuran de manera resumida. És important notar que avui dia qualsevol de les distribucions té una instal·lació molt optimitzada que necessita molt poca atenció de l'usuari, ja que obté informació del maquinari subjacent, per la qual cosa, generalment, per a un usuari novell no és necessari prendre decisions importants (distribucions com Ubuntu o Fedora, per exemple, són instal·lacions pràcticament automàtiques).

També hem de tenir en compte que un usuari novell pot iniciar el seu camí en el món Linux amb un altre tipus d'execucions de GNU/Linux que no modifiquen l'ordinador i permeten treballar en el sistema operatiu sense haver-ne d'instal·lar un altre. És altament recomanable iniciar els primers passos sobre un **GNU/Linux live:** l'usuari s'ha de baixar la imatge del sistema operatiu, crear-hi un CD o DVD i arrencar des d'aquest dispositiu sense tocar el disc dur de la màquina. Aquest tipus de distribucions (és recomanable utilitzar Knoppix, per la seva eficiència i versatilitat en les seves versions per a CD, DVD o USB) tenen "l'inconvenient" que per a desar el treball de l'usuari s'ha de fer sobre un dispositiu de disc USB, ja que si es desa sobre el sistema d'arxiu, com que és a la RAM, es perdran les dades.

VirtualBox

VirtualBox que permet arrencar una imatge o instal·lar un SO sobre un altre SO amfitrió (*host*) tant en 32 bits com en 64. Si no volem fer una instal·lació des de 0 es pot trobar una gran quantitat de distribucions amb les seves imatges ja fetes com per exemple en http://virtualboxes.org/images/.

Una altra opció totalment recomanable com a introducció (o com a forma de treball habitual) sense haver de tocar el sistema operatiu d'una màquina és treballar amb màquines virtualitzades. Per a això, es recomana utilitzar **VirtualBox**, que permet arrencar una imatge o instal·lar-ne una sobre un sistema operatiu amfitrió (*host*); tant en 32 bits com en 64 bits és altament configurable i si no volem fer una instal·lació es poden trobar gran quantitat de distribucions amb les seves imatges ja fetes, com per exemple a http://virtualboxes.org/images/, que té aproximadament 30 distribucions de Linux i 15 distribucions d'altres sistemes *nix o fins i tot d'Android o sistemes no *nix.

És important que abans d'instal·lar un nou sistema coneguem adequadament els components maquinari que tenim instal·lats en el nostre ordinador per a poder configurar-lo adequadament, encara que la distribució que utilitzem incorpori detecció de maquinari. És possible que en un sol disc dur tinguem instal·lats dos sistemes operatius (*dualboot*) o més totalment independents, i si

bé el procés d'instal·lació d'un altre sistema operatiu al mateix disc no hauria d'interferir amb les particions dels altres, és aconsellable fer còpies de seguretat de tots els documents importants.

És necessari, abans de començar, tenir informació de la marca i el model de la targeta gràfica, la de so, la de xarxa, la marca, el tipus i les característiques del monitor, i també qualsevol altre maquinari especial que tinguem (la resta del maquinari serà detectat pel sistema: placa base, la CPU i la memòria RAM).

Generalment, totes les distribucions de GNU/Linux proporcionen algun tipus de mitjà per a l'arrencada del procés d'instal·lació, i el més comú és un CD o DVD d'arrencada, per la qual cosa és necessari configurar la BIOS perquè pugui arrencar (*boot*) des de CD/DVD. Les instal·lacions són autoguiades i és important parar atenció a la selecció de l'idioma i del teclat per a evitar problemes des de l'inici (si bé es podrà configurar posteriorment).

Per a usuaris ja avançats, hi ha alguna altra manera d'instal·lar GNU/Linux que permet fer la instal·lació des de qualsevol mitjà: FTP, HTTP, disc dur, NFS, USB, però és recomanable no utilitzar-les com a primera experiència.

La partició del disc dur és una de les parts més crítiques de tot el procés, ja que implica dividir el disc dur en diverses seccions que seran considerades independents. Si ja tenim un sistema operatiu instal·lat al nostre ordinador, el disc estarà particionat en una o diverses particions, però si el disc és nou, tindrà una única partició. Per a instal·lar GNU/Linux hem de disposar, almenys, d'una partició per a ús propi (si bé és possible instal·lar-lo sobre altres sistemes d'arxius, no és recomanable aquesta opció per qüestions de rendiment i fiabilitat) i una altra de més petita per a una extensió de la memòria RAM de l'ordinador, anomenada *partició de swap* (generalment del doble de la memòria RAM instal·lada).

El procediment més comú per a reduir, crear o canviar la mida de les particions és utilitzar eines com les disponibles a Windows Vista, 7 (Administració de discos o l'aplicació **fips** amb llicència GPL i per a sistemes FAT) o en qualsevol Linux *live*. Es pot usar **gparted** per a modificar la mida d'una partició ja creada sense perdre'n el contingut (si bé es recomana fer còpies de seguretat dels arxius més importants). El procediment recomanable és engegar amb una Linux *live* i utilitzar la instrucció *gparted*, que és molt eficient i segur.

MBR

La informació de particions i el programa de càrrega d'un sistema operatiu es desen en una zona de dades reservada anomenada *MBR* (*master boot record*) sobre el primer disc.

Com a primer pas és recomanable que GNU/Linux utilitzi dues particions al disc dur (una per al sistema de fitxers i l'altra per a la *swap*). Si bé totes les distribucions tenen un particionament guiat, es pot fer de manera manual amb diferents utilitats (*fdisk, cfdisk, diskDruid*, etc.). La manera com GNU/Linux identifica els discos és amb */dev/hdX* per als discos IDE i */dev/sdX* per als

SCSI i Serial ATA, en els quals X és una lletra, corresponent al disc al qual ens vulguem referir: */dev/hda* és el mestre del primer canal IDE, */dev/hdb* el segon i així successivament (o */dev/sda* el primer disc SCSI o SATA i */dev/sdb* el segon, etc.). L'aplicació d'instal·lació ens farà una llista dels discos i haurem d'escollir sobre quin volem fer la instal·lació.

Quan creem una partició podrem escollir entre primària o lògica. En un disc dur podem tenir fins a 4 particions primàries i fins a 64 lògiques. Si no necessitem més de 4 particions, podem elegir qualsevol dels dos tipus. Si en necessitem més, haurem de tenir en compte que les lògiques se situen dins d'una de primària (fins a un màxim de 16 per a cada una), de manera que no podem tenir 4 particions primàries creades i després afegir-ne de lògiques. En aquest cas, n'hauríem de crear 3 de primàries i fins a 16 lògiques en la quarta partició primària.

Partició activa

De totes les particions d'un disc dur en podem elegir una perquè sigui l'activa. Aquest indicador serveix per a indicar a la BIOS o a l'EFI del sistema (sistema d'inicialització i càrrega del sistema operatiu) quina és la partició que ha d'iniciar si en l'MBR no troba cap programa d'arrencada.

Quan es crea una partició s'ha d'indicar quin sistema de fitxers utilitzarà (Linux ext3, Linux ext4, Linux *swap* o un altre) i una vegada fetes les particions, desarem la configuració i hem d'indicar al procés d'instal·lació on volem situar l'arrel del sistema de fitxers (*root filesystem*) i la *swap* del sistema, i a partir d'aquest moment es podrà continuar amb la instal·lació.

Una part important de la instal·lació són els mòduls del nucli, que són parts de programari especialitzades que treballen amb alguna part del maquinari o del sistema. En les distribucions actuals simplement s'ha de seleccionar quin dispositiu tenim (monitor, xarxa, so, gràfics) i la instal·lació carregarà pràcticament tots els mòduls necessaris, encara que en la majoria hi ha processos d'autodetecció, per la qual cosa no serà necessari seleccionar pràcticament res. Si algun mòdul no s'inclou durant la instal·lació, és possible fer-ho després amb ordres com ***insmod*** o ***modprobe*** (per a afegir un nou mòdul), ***lsmod*** (per a fer una llista dels instal·lats), ***rmmod*** (per a eliminar-ne algun) i també ***modprobe*** (per a provar-ne algun i, si funciona correctament, incloure'l en el nucli). Tots aquests mòduls són fitxers binaris que solem trobar en el directori /lib/modules/versió-del-sistema-operatiu/.

Després de configurar els mòduls que s'inclouran en el nucli del sistema operatiu, haurem de configurar la xarxa (si tenim la targeta necessària). Encara que en aquest document no entrarem en detall sobre xarxes, descriurem els conceptes necessaris per a poder fer aquest pas de manera bàsica. La primera dada que sol·licitarà la instal·lació és el nom del sistema (per a referir-nos-hi de manera amigable) i a continuació demanarà si a la nostra xarxa utilitzem un mecanisme anomenat DHCP (consisteix a tenir un servidor especial que s'encarrega d'assignar automàticament les IP als ordinadors que engeguen). Si utilitzem aquest mecanisme, ho hem d'indicar, i si no, ens preguntarà la IP (quatre nombres entre 0 i 255 separats per punts) i la màscara del nostre ordinador (quatre nombres entre 0 i 255, i és comú utilitzar 255.255.0.0). Si no coneixem aquestes dades, ens hem de dirigir a l'administrador de la nostra

xarxa. Seguidament haurem d'introduir l'IP de la passarel·la de la nostra xarxa (dispositiu o ordinador que actua de pont entre la nostra xarxa local i Internet; si no tenim cap dispositiu d'aquest tipus, podem deixar en blanc aquest camp).

A continuació, hem d'especificar el servidor (o servidors) de noms que utilitzem, anomenat DNS, que és una màquina que ens proporciona l'equivalència entre un nom i una adreça IP (és a dir, ens permetrà conèixer per exemple la IP de www.uoc.es de manera transparent). Si no sabem quins són, haurem de recórrer a l'administrador de la xarxa.

Adreça IP

Una adreça IP és la identificació d'un ordinador dins d'una xarxa quan utilitzem el protocol TCP/IP (protocol utilitzat en Internet).

Si som en una xarxa local podem consultar l'administrador perquè ens proporcioni tota la informació necessària o, si tenim un altre sistema operatiu instal·lat a l'ordinador, en podrem obtenir aquesta informació, però en cap cas no hem d'inventar aquests valors, ja que si l'ordinador està connectat a una xarxa local pot generar problemes a altres ordinadors.

Una vegada configurats aquests aspectes, haurem de seleccionar si volem instal·lar un petit programa al disc dur perquè en el procés d'arrencada de l'ordinador puguem elegir quin sistema operatiu dels que tenim instal·lats volem arrencar (fins i tot si només hem instal·lat GNU/Linux). Les aplicacions més usuals són el **lilo** (*Linux loader*) o el **grub** (recomanat) (*GNU, grand unified bootloader*), que tenen per objectiu iniciar el procés de càrrega i execució del nucli del sistema operatiu que li indiquem interactivament o per defecte després d'un temps d'espera. Totes les distribucions (si no hi ha problemes) detecten si tenim algun altre sistema operatiu instal·lat en el disc dur i configuren automàticament el sistema d'arrencada. Aquest programa generalment s'instal·la en l'MBR del disc mestre del primer canal IDE o SCSI, que és el primer lloc que la BIOS o EFI de l'ordinador inspecciona buscant un programa d'aquestes característiques.

L'últim pas de la instal·lació és la selecció de paquets per instal·lar a més dels estrictament necessaris per al funcionament bàsic del sistema operatiu. La majoria dels processos d'instal·lació inclouen dues maneres de seleccionar els programes del sistema: bàsic o expert. Amb el procés de selecció bàsic, s'agrupen els paquets disponibles per a grans grups de programes: administració, desenvolupament de programari, ofimàtica, matemàtiques, etc.; és una opció recomanable per a fer els primers passos. Si no seleccionem un paquet després es podrà fer una instal·lació posterior amb l'eina de la qual disposen totes les distribucions per a instal·lar o desinstal·lar paquets. Debian GNU/Linux va ser una de les primeres a incloure aplicacions per a gestionar els paquets; s'anomena **apt** i és molt útil per a fer el manteniment i actualització de tots els paquets instal·lats fins i tot en el sistema operatiu mateix.

Si la instal·lació no ha funcionat correctament, pot passar que no puguem arrencar cap dels sistemes operatius instal·lats (ni el nou ni l'anterior), però totes les distribucions tenen un mode de rescat en l'arrencada (*rescue mode*), que ens permetrà arrencar el sistema GNU/Linux des del CD/DVD, accedir al disc

dur i arreglar aquelles coses que no han funcionat o recuperar el sistema operatiu inicial, si bé per a alguns casos són necessaris una sèrie de coneixements avançats en funció de quina hagi estat la causa d'error.

4. Configuracions bàsiques

4.1. El sistema d'entrada

Tant en mode gràfic com en mode text, el procediment d'identificació i entrada es denomina connexió (*login*). Generalment, el sistema arrencarà en mode gràfic, però podem passar a mode text seleccionant el tipus de sessió al plafó d'entrada o amb Ctrl-Alt-Esborrar (que acaba l'execució del servidor gràfic).

En mode text es llancen 5 terminals independents, als quals es pot accedir mitjançant Alt-F1, Alt-F2, etc., que permetran treballar simultàniament amb diferents comptes alhora. El procés de connexió posa en pantalla, primer, un missatge que es pot modificar des de l'arxiu /etc/issue i que admet diferents variables (*d*, data actual, *s*, nom del SO, *t*, hora actual, etc.) i en segon lloc el missatge del dia des de /etc/motd (creant un fitxer buit anomenat *.hushlogin* en el nostre directori d'usuari s'anul·la aquest missatge). A continuació el procés de connexió llança l'intèrpret per defecte per a l'usuari (indicat en l'últim camp a /etc/passwd).

L'intèrpret d'ordres executa l'arxiu .profile del directori de l'usuari per a les opcions per defecte d'aquest mateix usuari, que es complementa amb /etc/profile, que configura opcions per defecte per a tots els usuaris. Cada intèrpret a més té arxius de configuració propis, com per exemple el *shell Bash*, que executa, a més, dos fitxers més anomenats .bashprofile (que s'executa en cada connexió) i .bashrc (que s'executa cada vegada que s'obre un nou terminal). Veurem algunes de les instruccions que podem trobar en aquests arxius:

```
# ~/.profile: executat per l'intèrpret a l'entrada
# No serà llegit per bash si hi ha ~/.bash_profile o ~/.bash_login
# Vegeu-ne exemples a /usr/share/doc/bash/examples/startup-files

# El valor per defecte d'umask
umask 022

# Si està executant bash executa .bashrc si hi és
if [ -n "$BASH_VERSION" ]; then
if [ -f "$HOME/.bashrc" ]; then
. "$HOME/.bashrc"
fi
```

```
fi

# Inclou en PATH un directori bin de l'usuari
if [ -d "$HOME/bin" ] ; then
PATH="$HOME/bin:$PATH"
fi
# Canvia l'indicador (prompt)
export PS1='\h:\w\$ '

# ~/.bashrc: executat per bash(1) per a terminals no d'entrada.
# Si no s'executa interactivament no fa res.
[ -z "$PS1" ] && return

# Habilita el suport de color per a l'ordre ls
if [ -x /usr/bin/dircolors ]; then
eval "`dircolors -b`"
alias ls='ls --color=auto'
alias dir='dir --color=auto'
fi

# altres àlies
alias ll='ls -l'

# habilita programmable completion features
if [ -f /etc/bash_completion ]; then
. /etc/bash_completion
fi
```

Com es pot observar en aquests fitxers (que són un exemple de *shell script* i que veurem més endavant), s'inclouen diferents definicions de variables (PATH, per exemple, que és la variable on es buscaran els programes per a executar-los sense haver d'incloure tot el camí; PS1, que és la variable que emmagatzema l'indicador *–prompt–*, que és el caràcter que surt a l'esquerra de la línia d'ordres acabat en $ o % per a l'usuari normal i en # per al *root*). També tenim àlies d'ordres o execució condicional d'altres arxius (en el primer si s'està executant el *bash* executa el $HOME/.bashrc). Si es volen executar programes del directori des del qual estem situats sense necessitat de posar ". /" al principi, podríem afegir aquesta entrada en la declaració del PATH, però generalment no s'inclou, ja que pot representar un forat de seguretat. Per al **prompt** hem utilitzat l'ordre **export** per a definir les anomenades *variables d'entorn*, que es mantenen durant tota la sessió i es poden consultar amb la mateixa ordre.

Amb **set** i **unset** també podem inicialitzar o treure altres variables o atributs representats per defecte (amb **echo $variable** podrem consultar el valor de cada variable). Algunes de Bash són:

PWD: directori actual.

LOGNAME: nom de l'usuari connectat.
BASH_VERSION: versió del Bash que utilitzem.
SHELL: intèrpret d'ordres utilitzat.
RANDOM: genera un nombre aleatori diferent cada vegada que en mostrem el contingut.
SECONDS: nombre de segons que han passat des que hem obert l'intèrpret d'ordres.
HOSTNAME: nom del sistema.
OSTYPE: tipus de sistema operatiu que estem utilitzant.
MACHTYPE: arquitectura de l'ordinador.
HOME: directori personal de l'usuari.
HISTFILESIZE: mida de l'arxiu d'història (nombre d'ordres que es desen).
HISTCMD: número d'ordre actual en la història.
HISTFILE: fitxer en el qual es desa la història d'ordres.

La configuració addicional de la connexió gràfica es farà per mitjà de l'entorn gràfic i dependrà del tipus d'entorn.

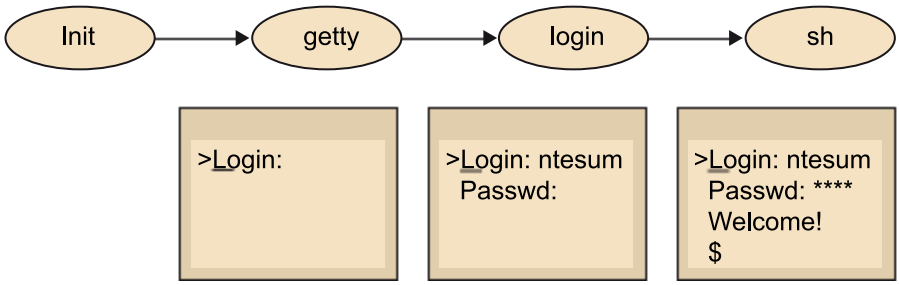
4.2. L'intèrpret d'ordres (*shell*)

Com ja hem avançat, el terme genèric *shell* s'utilitza per a denominar un programa que serveix d'interfície entre l'usuari i el nucli del sistema GNU/Linux. En aquest apartat veurem algunes característiques bàsiques dels intèrpretsinteractius de text, que és el programa que veurà l'usuari (i l'atendrà) una vegada fet el procediment de connexió.

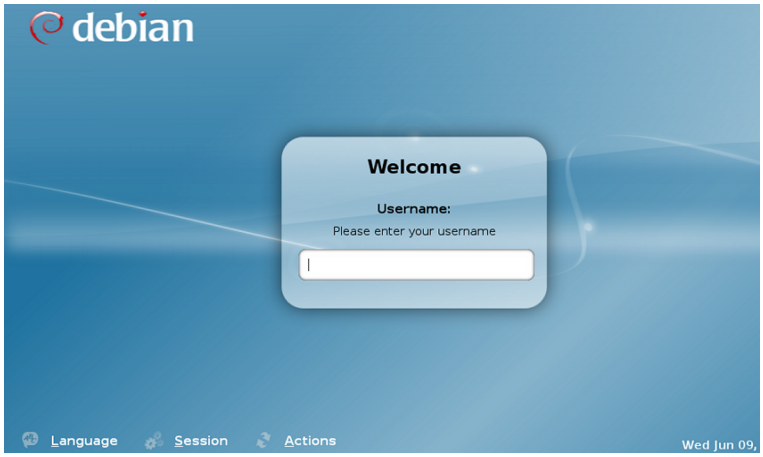
L'intèrpretés el que els usuaris veuen del sistema, ja que la resta del sistema operatiu roman ocult als seus ulls. L'intèrpret està escrit de la mateixa manera que un procés (programa) d'usuari; no està integrat en el nucliis'executa com un programa més de l'usuari.

Quan el sistema GNU/Linux arrenca, sol presentar als usuaris una interfície d'entrada que pot ser gràfica o en mode text. Aquestes diferents entrades dependran dels modes (o nivells) d'arrencada del sistema (*runlevels*), que permetran configurar els modes d'arrencada del sistema operatiu.

La figura següent mostra l'arrencada *shell* en mode text i els processos de sistema involucrats (OKe).



En el mode d'arrencada gràfica, la interfície està composta per algun gestor d'accés que administra el procés de connexióde l'usuari des d'una pantalla (caràtula) gràfica, en la qual se sol·licita la informació d'entrada corresponent: el seu identificador com a usuari i la seva paraula de pas (contrasenya). En GNU/Linux solen ser habituals els gestors d'accés: **xdm** (propi d'X Window), **gdm** (Gnome) i **kdm** (KDE), i també algun altre associat a diferents gestors de finestres (*window managers*) com mostra la figura següent:



Una vegada validat l'accés, l'usuari trobarà una interfície gràfica d'X Window amb algun gestor de finestres, com Gnome o KDE (vegeu l'últim punt del capítol). Des del mode gràfic també és possible "interactuar" amb el sistema per mitjà d'un entorn de treball en mode text simplement obrint un emulador de terminal o simplement terminal (p. ex., Xterm) des dels menús de la interfície gràfica (en Gnome: Aplicacions → Accessoris → Terminal).

Si l'accés és en mode text (anomenat també *modeconsola*), una vegada identificats obtindrem l'accés a l'intèrpret de manera interactiva (es pot passar del mode gràfic al text amb Crtl+Alt+F1 a F5, que permetrà tenir 5 consoles diferents i tornar-hi amb Crtl+Alt+F7). Un altre mode de treball amb un intèrpret interactiu és per mitjà d'una connexió remota des d'una altra màquina connectada en xarxa i amb aplicacions com ara Telnet o *rlogin* (poc utilitzades per insegures), *ssh,* o gràfiques com els emuladors X Window.

Una vegada iniciat l'intèrpret interactiu [Qui01], es mostra un indicador d'ordres(símbol o seqüència de caràcters com $, %, # –utilitzat generalment per a identificar l'usuari arrel o *root*– o també quelcom configurable per l'usuari com *MySys*) que indica a l'usuari que pot introduir una línia d'ordres.

Després de la introducció, l'intèrpret assumeix la responsabilitat de validar la sintaxi i posar els processos necessaris en execució, mitjançant una sèrie de seqüències o fases:

1) Llegir i interpretar la línia d'ordres.

2) Avaluar els caràcters comodí com $ * ? o d'altres.

3) Gestionar les redireccions E/S necessàries, les canonades i els processos en segon pla (*background*) necessaris (&).

4) Manejar senyals.

5) Preparar l'execució dels programes.

Normalment, les línies d'ordres podran ser execucions d'ordres del sistema, ordres pròpies de l'intèrpret interactiu, l'engegada d'aplicacions o *shell scripts* (seqüències d'ordres, variables, sentències de control, etc., que generalment es troben en un arxiu ASCII).

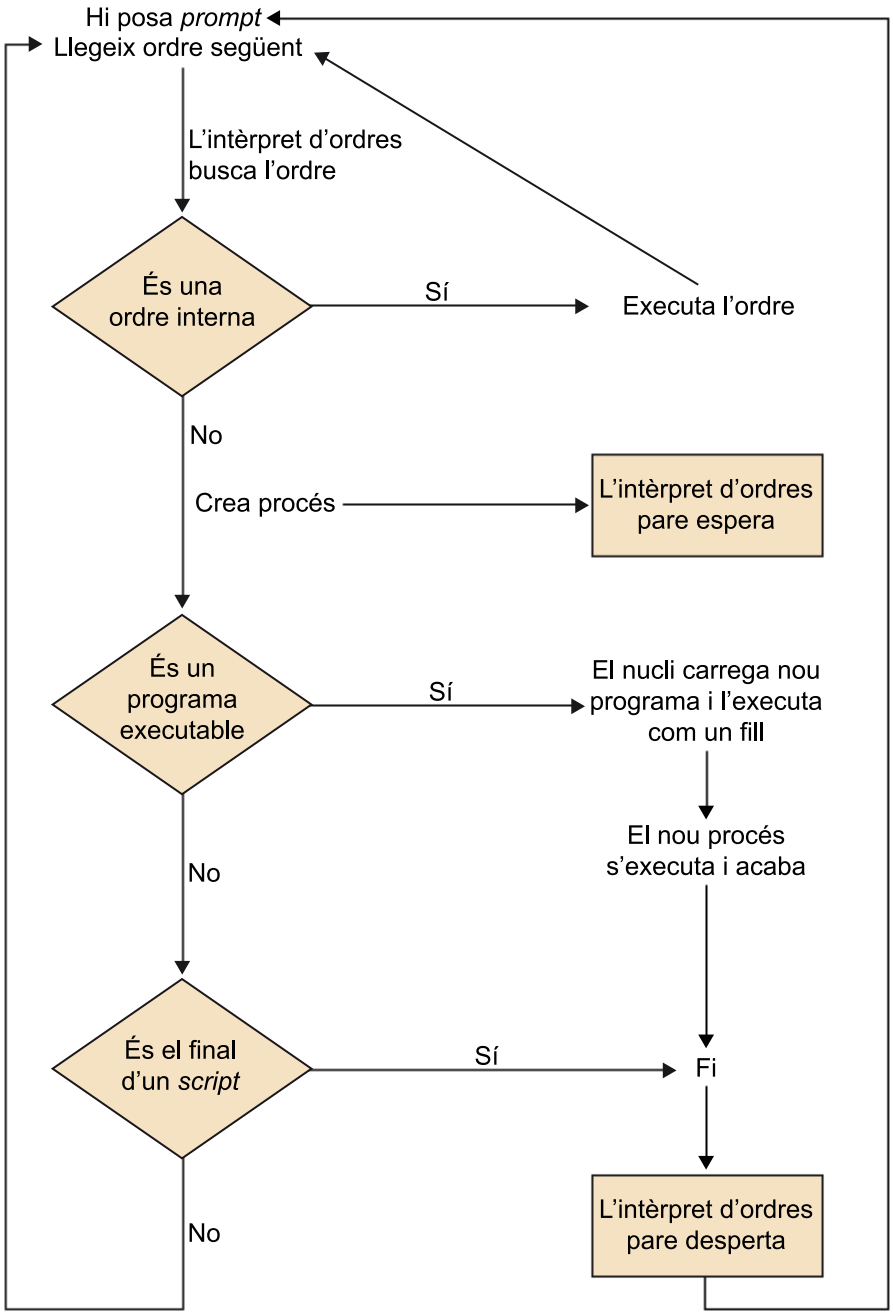
Els fitxers de *script* són directament executables pel sistema sota el nom que s'hagi donat al fitxer. Per a executar-los, s'invoca l'intèrpret juntament amb el nom del fitxer, o bé es donen permisos d'execució al *shell script*.

En certa manera, podem veure el *shell script* com a codi d'un llenguatge interpretat que s'executa sobre l'intèrpret interactiu corresponent. Per a l'administrador, els *shell scripts* són molt importants bàsicament per dues raons:

1) La configuració del sistema i de la majoria dels serveis proporcionats es fan mitjançant eines en formade *shell scripts*.

2) La manera principal d'automatitzar processos d'administració és mitjançant la creació de *shell scripts* per part de l'administrador.

La figura següent mostra el flux de control bàsic d'un intèrpret d'ordres:



Tots els programes invocats mitjançant un intèrpret posseeixen tres fitxers predefinits, especificats pels descriptors de fitxers corresponents (*file handles*). Per defecte, aquests fitxers són:

1) ***standard input*** (entrada estàndard): normalment assignada al teclat del terminal (consola); usa el descriptor número 0 (en UNIX els fitxers utilitzen descriptors sencers).

2) ***standard output*** (sortida estàndard): normalment assignada a la pantalla del terminal; usa el descriptor 1.

3) ***standard error*** (sortida estàndard d'errors): normalment assignada a la pantalla del terminal; utilitza el descriptor 2.

Això ens indica que qualsevol programa executat des de l'intèrpret tindrà per defecte l'entrada associada al teclat del terminal, la seva sortida cap a la pantalla i, en el cas de produir-se errors, també els envia a la pantalla.

A més, els intèrprets poden proporcionar tres mecanismes següents de gestió de l'E/S:

1) Redirecció: atès que els sistemes UNIX tracten de la mateixa manera els dispositius d'E/S i els fitxers, l'intèrpretels tracta a tots simplement com a fitxers. Des del punt de vista de l'usuari, es poden reassignar els descriptors dels fitxers perquè els fluxos de dades d'un descriptor vagin a qualsevol altre descriptor; això s'anomena *redirecció*. Per exemple, ens referirem a la redirecció dels descriptors 0 o 1 com a la redirecció de l'E/S estàndard. Per a això s'utilitzen els símbols > >> < <<; per exemple *ls* > *dir.txt* redirecciona la sortida de la instrucció *ls* a un fitxer anomenat *dir.txt*, que si existeix s'esborra i desa la sortida de l'ordre, i si no, es crea i es desa la sortida. Si fos *ls* >> *dir.txt* la sortida s'afegeix al final si el fitxer existeix.

2) Canonades (*pipes*): la sortida estàndard d'un programa es pot usar com a entrada estàndard d'un altre per mitjà de canonades. Diversos programes poden ser connectats entre si mitjançant canonades per a formar el que es denomina un *pipeline*. Per exemple *ls* | *sort* | *more*, en què la sortida de la instrucció *ls* s'ordena (*sort*) i la sortida d'aquest es mostra per pantalla en forma paginada.

3) Concurrència de programes d'usuari: els usuaris poden executar diversos programes simultàniament, tot indicant que l'execució es produirà en segon terme (*background*), oposat al primer terme (o *foreground*), en què es té un control exclusiu de pantalla. Una altra utilització consisteix a permetre tasques llargues en segon terme quan interactuem l'intèrpret amb altres programes en primer terme. Per exemple, *ls >dir.txt &* s'executarà en segon termei permetrà a l'usuari continuar interactuant amb l'intèrpretmentre l'ordre es va executant.

L'intèrpret d'ordres per defecte és el que s'indica en l'últim camp del fitxer /etc/passwd per a l'usuari, i es pot esbrinar des de la línia d'ordres pel valor de la variable d'entorn amb el següent:

```
echo $SHELL
```

Algunes consideracions comunes a tots els intèrprets d'ordres:

- Tots permeten l'escriptura de *shell scripts*, que després són interpretats executant-los o bé pel nom (si el fitxer té permís d'execució) o bé passant-lo com a paràmetre a la instrucció de l'intèrpret.

- Els usuaris del sistema tenen un intèrpret associat per defecte. Aquesta informació es proporciona en crear els comptes dels usuaris. L'administrador assigna un intèrpret a cada usuari, o si no s'assigna l'intèrpret per defecte (Bash en GNU/Linux). Aquesta informació es desa en el fitxer /etc/passwd, i es pot canviar amb l'ordre **chsh**;aquesta mateixa ordre amb l'opció *-l* ens fa una llista els intèrprets disponibles en el sistema (o vegeu també /etc/shells).

- Cada intèrpret és en realitat una instrucció executable, normalment present en els directoris /bin en GNU/Linux (o /usr/bin).

- Es poden escriure *shell scripts* en qualsevol intèrpret, però ajustant-se a la sintaxi de cadascun, que normalment és diferent (de vegades hi ha només petites diferències). La sintaxi de les construccions, i també les ordres internes, estan documentats a la pàgina *man* de cada intèrpret (*man bash*, per exemple).

- Cada intèrpret té alguns fitxers d'arrencada associats (fitxers d'inicialització), i cada usuari els pot adaptar a les seves necessitats, incloent-hi codi, variables, rutes (*path*)...

- La potència en la programació es troba a combinar la sintaxi de cada intèrpret (de les seves construccions), amb les ordres internes de cada intèrpret, i una sèrie d'ordres UNIX molt utilitzades en els *scripts*, com per exemple *cut, sort, cat, more, echo, grep, wc, awk, sed, mv, ls, cp*...

- Si com a usuaris estem utilitzant un intèrpret determinat, res no ens impedeix arrencar una còpia nova de l'intèrpret (anomenat *subshell*), tant si és el mateix com un altre de diferent. Senzillament, l'invoquem pel nom de l'executable, ja sigui *sh, bash, csh o ksh*. També quan executem un *shell script* es llança un *subshell* amb l'intèrpret que correspongui per a executar l'*script* demanat.

Atesa la importància dels *shell scripts* en les tasques d'administració d'un sistema UNIX, en el capítol següent es veuran alguns detalls més sobre l'intèrpret i exemples de sintaxi i de programació.

4.3. El sistema d'arrencada

Ja hem vist en el punt anterior la manera de configurar durant la instal·lació de l'arrencada del sistema per mitjà d'un gestor com *lilo* o *grub* (recomanable). Com ja sabem, a partir de la BIOS o EFI, l'ordinador llegeix l'MBR del primer disc mestre i executa el gestor d'arrencada (si no es troba aquest programa, s'inspecciona el sector d'arrencada de la partició activa del disc) encara que recomanem instal·lar *grub* a l'MBR, que és el primer lloc que s'inspecciona.

El *grub* ens permet múltiples configuracions, permet tenir un petit intèrpret d'ordres en arrencar l'ordinador, i accedir als arxius de les particions del disc sense carregar cap sistema operatiu, etc. En aquest subapartat només veurem algunes configuracions bàsiques, però si es necessiten opcions avançades es pot consultar la documentació existent a http://www.gnu.org/software/grub/manual/grub.html.

El *grub* es carrega en dues fases i durant el procediment d'instal·lació ja es carreguen dos fitxers corresponents a cada fase. Si volem instal·lar el *grub* perquè no ens mostri cap menú per a seleccionar el sistema operatiu que volem carregar, només hem d'executar l'ordre **grub** i des de l'indicador, executar:

```
install (hd0, 0)/boot/grub/stage1 d (hd0) (hd0, 0)/boot/grub/stage2
```

Aquesta instrucció instal·la el *grub* en l'MBR del disc mestre i, com podem observar, la manera com es refereixen els discos difereix de la de GNU/Linux (o *lilo*). A *hdX* la *X*, en lloc de *a, b* ..., és 0, 1..., i per a les particions també es comença amb el 0 per a la primera, per la qual cosa *hda1* serà en *grub (hd0, 0)*. El paràmetre **d (hd0)** indica que la primera fase del *grub* s'instal·larà en l'MBR del primer disc. L'última opció especifica on se situa el fitxer per a la segona fase de càrrega, que és executada per la primera (una altra manera de fer la instal·lació del *grub* és amb l'ordre **grub-install**).

El pas de paràmetres al nucli Linux en el moment de l'arrencada és molt simple. Per exemple, passant *single* o *1* iniciaria el sistema en el *runlevel 1*, amb *root=/dev/hda3* especificarem l'arrel del sistema de fitxers, etc. Si bé es poden introduir les ordres interactivament, durant l'arrencada és recomanable utilitzar un arxiu de configuració com el següent (els comentaris comencen per " # "):

```
# Arxiu menu.lst en /boot/grub
# Especificació del sistema operatiu que es carrega per defecte
default 0
# Espera de 10 segons abans de carregar el sistema per defecte
timeout 10
# Configuració d'arrencada per a un sistema GNU/Linux
title Debian GNU/Linux
kernel (hd0, 0) /vmlinuz root=/dev/hda1
# Configuració d'arrencada per a un sistema Windows
title Windows
root (hd0, 2)
makeactive
```

Per a instal·lar *grub* amb aquest menú d'arrencada, hauríem d'executar la mateixa instrucció que anteriorment, però afegint el paràmetre *p (hd0, 0)/boot/grub/menu.lst*.

Normalment és la manera habitual de treballar, i per això només haurem d'entrar a /boot/grub i modificar aquest arxiu. L'arxiu de configuració accepta diversos paràmetres, a més dels esmentats, com els següents:

- **Colors:**

```
color cyan/blue white/blue
```

- ***passwd*** (per a evitar que es pugui editar l'arxiu durant l'arrencada):

```
password ['--md5'] passwd
```

- **Càrrega d'un nucli específic amb *initramfs:***

```
title Debian GNU/Linux, kernel 2.6.26-2-686
root (hd0,0)
kernel /boot/vmlinuz-2.6.26-2-686 root=/dev/hda1 ro quiet
initrd /boot/initrd.img-2.6.26-2-686
```

- **Càrrega d'un nucli específic amb *initramfs* i en mode *single user:***

```
title Debian GNU/Linux, kernel2.6.26-2-686 (single-user mode)
root (hd0,0)
kernel /boot/vmlinuz-2.6.26-2-686 root=/dev/hda1 ro single
initrd /boot/initrd.img-2.6.26-2-686
```

La càrrega d'un nucli (*kernel*) amb una imatge anomenada *initrd* fa referència a un disc RAM (generat amb **mkinitramfs**), que es carrega en el moment de l'arrencada per a accedir a diferents mòduls que s'annexaran al nucli. Aquest tipus de configuració permet tenir un nucli general i agregar els dispositius en el moment de la càrrega (mòduls), per la qual cosa podríem utilitzar el nucli en diferents màquines i amb diferent maquinari només canviant la imatge dels mòduls de l'*initrd*.

4.4. Accés a particions i dispositius

Els sistemes tipus Unix tracten tots els dispositius de l'ordinador com si fossin arxius. Això permet total flexibilitat, ja que es poden aprofitar tots els mecanismes i les funcions que s'utilitzen amb fitxers per als dispositius. En el direc-

tori /dev es tenen tots els dispositius reconeguts pel sistema. Si el sistema no reconeix adequadament un dispositiu o volem crear-ne un d'especial, es pot utilitzar l'ordre **mknod**, però s'ha d'utilitzar amb compte, ja que l'ús incorrecte podria danyar parts del sistema.

Per a les unitats d'emmagatzemament, el sistema proveeix instruccions com *mount* i *umount*, que situen (munten) o desmunten tot el sistema d'arxius d'un determinat dispositiu o unitat en un directori existent del sistema. La manera bàsica d'utilitzar l'ordre és **mount dispositiu directori**, en què el *dispositiu* pot ser qualsevol del canal IDE o SCSI (*/dev/hdXX*, */dev/sdXX*), la disquetera (/dev/FDX), memòries USB, etc., i *directori* és la ubicació en la qual muntarem l'estructura de fitxers del dispositiu, i si el sistema d'arxius no és el mateix amb el qual estem treballant, s'haurà d'agregar el paràmetre *-t filesystem*, en què *filesystem* és una sigla que es pot consultar a la pàgina del manual de *mount*. És recomanable que el directori en el qual muntem aquests dispositius sigui buit, ja que quan s'utilitza com a punt de muntatge no s'hi pot accedir. Per a desmuntar un d'aquests dispositius, podem utilitzar **umount directori**, en què el directori ha de ser el punt de muntatge utilitzat. Si muntem dispositius mòbils com CD o USB, és important no treure el dispositiu del suport, ja que abans hem d'avisar al sistema perquè actualitzi la memòria cau del sistema de fitxers del dispositiu (o actualitzar les taules internes en el cas que el dispositiu solament sigui de lectura). Igualment, tampoc no podem desmuntar el dispositiu si algun usuari o aplicació està utilitzant algun dels arxius o directoris (en intentar-ho, el sistema donaria un missatge d'error).

Per defecte, per a poder muntar i desmuntar sistemes d'arxiu, es necessiten privilegis de superusuari, però es poden cedir aquests permisos a usuaris "administradors" afegint una entrada en el fitxer /etc/sudoers, de manera que amb l'ordre *sudo mount ...* l'usuari habilitat podrà fer tasques permeses només al *root* (consulteu el manual de **mount** per a altres opcions en el treball amb dispositius i particions, com per exemple *rw, suid, dev, exec, auto, nouser, async*, que són les considerades per defecte).

Tot el muntatge de particions i dispositius es pot fer durant la inicialització del sistema operatiu per mitjà d'entrades en l'arxiu /etc/fstab. Un exemple típic d'aquest arxiu és:

```
# /etc/fstab: static file system information.
#
# <file system> <mount point>  <type>       <options>         <dump> <pass>
  proc          /proc          proc         defaults             0      0
  /dev/hda1     /              ext3         errors=remount-ro    0      1
  /dev/hda5     none           swap         sw                   0      0
  /dev/hdc      /media/cdrom0  udf,iso9660  user,noauto          0      0
```

Cada línia és un *mount* i els paràmetres són (en ordre): dispositiu per muntar, on es munta, tipus de sistema d'arxius (*auto* perquè ho detecti automàticament), opcions de muntatge, especificació de si volem fer còpies de seguretat (*dump*), i l'últim camp serveix per a indicar l'ordre de muntatge (0, indica que l'ordre no és important). L'arrel del sistema d'arxius és el primer que s'ha de muntar, i per això en aquest camp hi hauria d'haver un 1. Una entrada que sempre veurem en aquest fitxer i que ens pot sorprendre és el directori /proc, que té un significat especial. Realment, el que hi ha en aquest directori no són fitxers, sinó el valor de moltes de les variables que utilitza el nucli del sistema, que permetrà ajustar paràmetres del nucli com si es tractessin d'un arxiu.

És interessant consultar l'ordre **autofs**, que permet muntar automàticament un sistema d'arxiu quan es detecta la inserció d'un dispositiu en el sistema.

4.5. Configuració de dispositius

És important, abans d'intentar configurar algun dispositiu, buscar-ne informació, i fins i tot abans de comprar-ne un, assegurar-se que disposa de controladors compatibles amb la versió amb què pretenem treballar.

1) Teclat

Si comencem pel teclat, és important que funcioni d'acord amb el mapa de caràcters configurat en el procés d'instal·lació. Aquest mapa de caràcters es troba a /etc/console/boottime.kmap.gz, i si canviem de teclat, només s'ha de canviar aquest fitxer per l'adequat, que es troba a /usr/share/keymaps/ ordenat per arquitectures d'ordinadors i països (d'aquests arxius es pot canviar alguna de les seves entrades per a adaptar-les a les nostres necessitats). Es pot veure el número que correspon a cada tecla executant com a *root* l'ordre **showkey.** Si no volem reiniciar el sistema en canviar l'arxiu de configuració de teclat, podem utilitzar l'ordre **loadkeys; dumpkeys** ens mostra les que estan configurades i **consolechars** permet carregar la lletra que volem per al terminal. Un altre aspecte relacionat amb el teclat és el tema dels accents, les dièresis, que es poden configurar a partir del fitxer /etc/inputrc (totes les directives possibles d'aquest fitxer les tenim especificades al manual de **readline**). La que pot ser més útil és la de **convert-meta**, que en desactivar-la (*set convert-meta off*) ens permet utilitzar els accents i les dièresis.

Finalment, una altra configuració important (indirectament relacionada amb el teclat) és la *local*, en què es pot configurar la zona geogràfica en la qual ens trobem per a poder utilitzar tecles especials del teclat, veure les dates en el format correcte, etc. Aquesta configuració és utilitzada per moltes de les biblioteques del sistema, de manera que en moltes ordres i aplicacions s'utilitzarà aquesta configuració per a adaptar algunes funcions de l'entorn local. Aquesta configuració es troba a /etc/locale.gen i es poden utilitzar les ordres **locale-gen** i **locale**, veure-les o actualitzar-les.

Nota

Una altra manera de reconfigurar el mapa de tecles i la configuració local en Debian és *apt-reconfigure console-data* i *apt-reconfigure locales*, respectivament.

2) Targeta de xarxa (Ethernet)

Per a configurar una nova targeta de xarxa (de tipus Ethernet), en primer lloc, cal afegir el mòdul necessari perquè es reconegui adequadament. Si bé no és necessari per a algunes targetes, ens hem d'assegurar (abans de comprar la targeta) que tenim el controlador o mòdul necessari.

Amb l'ordre **discover** podem saber quin tipus de maquinari tenim i trobar el mòdul corresponent. Si es vol deixar configurat perquè es carregui, s'haurà d'incloure a /etc/modules (o també es pot utilitzar **modprobe** o **insmode**). Per a configurar-la a /etc/network/interfaces, es podrà especificar (en Debian) la configuració de les interfícies del sistema. Una interfície és un dispositiu (real o lògic) relacionat amb la xarxa a partir del qual el sistema es podrà comunicar. Un exemple podria ser:

```
# Interfície de loopback
auto lo
iface lo inet loopback
# NIC auto eth0
iface eth0 inet static
address 192.168.0.10
netmask 255.255.255.0
network 192.168.0.0 # opcional
broadcast 192.168.0.255 # opcional
gateway 192.168.0.1 # opcional
```

La primera entrada d'aquest arxiu és la interfície de *loopback*. Aquesta interfície no es correspon amb cap targeta ni dispositiu real de l'ordinador i permet utilitzar els mecanismes de comunicació internament. La directiva *auto*, davant del dispositiu, indica que es pot muntar automàticament quan el sistema arrenca. La directiva *iface* especifica el tipus de targeta i protocol que s'hi utilitzarà mitjançant la sintaxi *iface dispositiu protocol configuració*. Per a les targetes Ethernet, el dispositiu serà *ethX*, en què la *X* serà un nombre, començant per 0, que indica el número de targeta instal·lada a l'ordinador. La família del protocol de comunicació utilitzat amb la targeta sol ser *inet* per al protocol IPv4 utilitzat a Internet i *inet6* per a la nova versió d'IPv6, i en l'últim camp s'indica com s'obté la configuració de xarxa de la targeta (la seva adreça IP, la xarxa on es troba, la passarel·la que cal utilitzar, etc.), amb les possibilitats següents (bàsiques):

- **static:** haurem de proveir *address* (adreça IP de la interfície), *netmask* (màscara de l'adreça IP), *gateway* (adreça IP de la passarel·la que utilitzem per a aquesta interfície).

- **dhcp:** per a configurar de manera remota la IP dels ordinadors d'una xarxa local (*dynamic host configuration protocol*).

Per a manejar la configuració de xarxa tenim diferents ordres, com **ifconfig** (visualitza i configura els dispositius), **ifdown** i **ifup** (per a apagar o encendre la interfície) o **route** (que ens mostra la taula d'encaminament).

3) Targeta sense fil (Wi-Fi)

Per a configurar la xarxa sense fil (*wireless LAN* o Wi-Fi) haurem d'afegir al nucli els mòduls necessaris o utilitzar els que ja estan compilats en el nucli mateix. Se segueix la mateixa norma de nom de dispositius que per a les targetes Ethernet, i en /etc/network/interfaces (en Debian) haurem d'afegir la configuració necessària perquè s'assigni una IP com en l'exemple següent (considerem que és el segon dispositiu, per la qual cosa serà *eth1*):

```
# Wireless network
auto eth1
iface eth1 inet dhcp
wireless_essid miwifi
wireless_channel 6
wireless_mode managed
wireless_keymode open
wireless_key1 millavehexadecimal
wireless_key2 s: millaveascii
wireless_defaultkey 1
```

Els paràmetres subsegüents a *iface* els podrem trobar a la pàgina del manual de l'ordre **iwconfig**, que és la que s'utilitza per a configurar els dispositius sense fil.

4) Targeta de so

Igual com en els casos anteriors, la targeta de so també necessita el mòdul del nucli per a poder funcionar correctament. Amb l'aplicació **discover**, podem descobrir quin mòdul és el que es correspon amb la nostra targeta o amb l'ordre **lspci | grep audio.** Per a instal·lar el mòdul, procedirem com amb la targeta de xarxa amb ***insmod*** o ***modprobe***, per a configurar-lo permanentment a /etc/modules. Si bé amb el mòdul corresponent ja podrem utilitzar la targeta de so adequadament, generalment també se sol instal·lar la infraestructura de so ALSA (Advanced Linux Sound Architecture) inclosa per defecte en la majoria de les distribucions.

5) Impressora

En GNU/Linux la configuració d'impressores es pot fer de diferents maneres, si bé **lpd** (*line printer daemon*) va ser un dels primers programes de gestió d'impressió, actualment n'hi ha d'altres més fàcils de configurar i gestionar. Els més bàsics són:

- **lpd:** un dels primers dimonis d'impressió dels sistemes tipus Unix. La configuració s'ha de fer manualment.

- **lpr:** la versió de BSD de l'*lpd*. És molt recomanable utilitzar algun tipus de filtre automàtic com *magicfilter* o *apsfilter* per a configurar-lo.

- **LPRng:** aplicacions basades en *lpr*, amb l'avantatge que incorporen una eina de configuració denominada *lprngtool*, que permet configurar-la de manera gràfica i senzilla.

- **gnulpr:** la versió de GNU del sistema d'impressió *lpr*. També incorpora eines gràfiques de configuració, gestió dels serveis, etc.

- **CUPS (recomanat):** de *Common UNIX Printing System*, aquest conjunt d'aplicacions és compatible amb les ordres d'*lpr* i també serveix per a xarxes Windows.

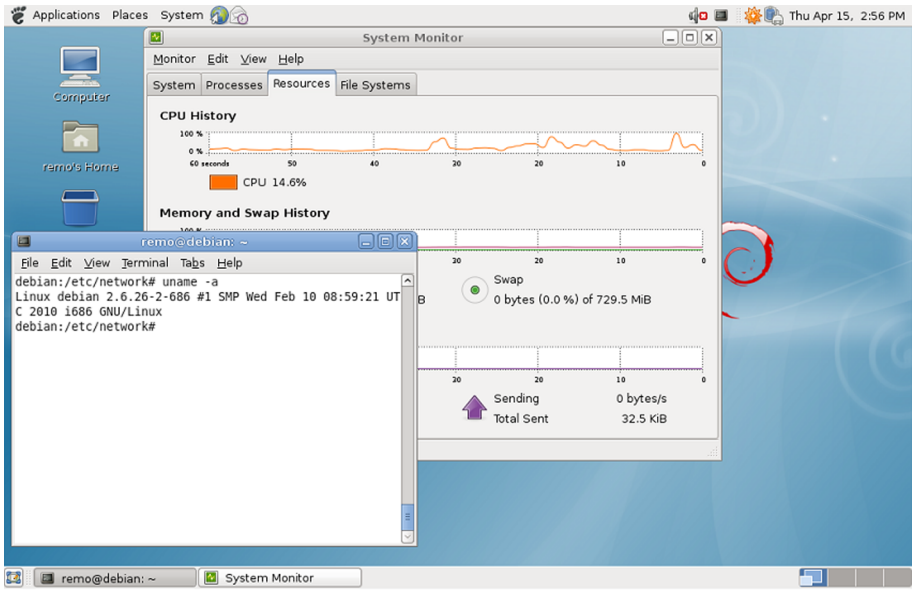
Normalment, totes aquestes ordres tenen els seus mètodes de configuració propis, però utilitzen el fitxer /etc/printcap per a desar-la i utilitzen un dimoni (procés que s'executa indefinidament) perquè el sistema d'impressió sigui operatiu i fins i tot es pugui imprimir des d'altres ordinadors.

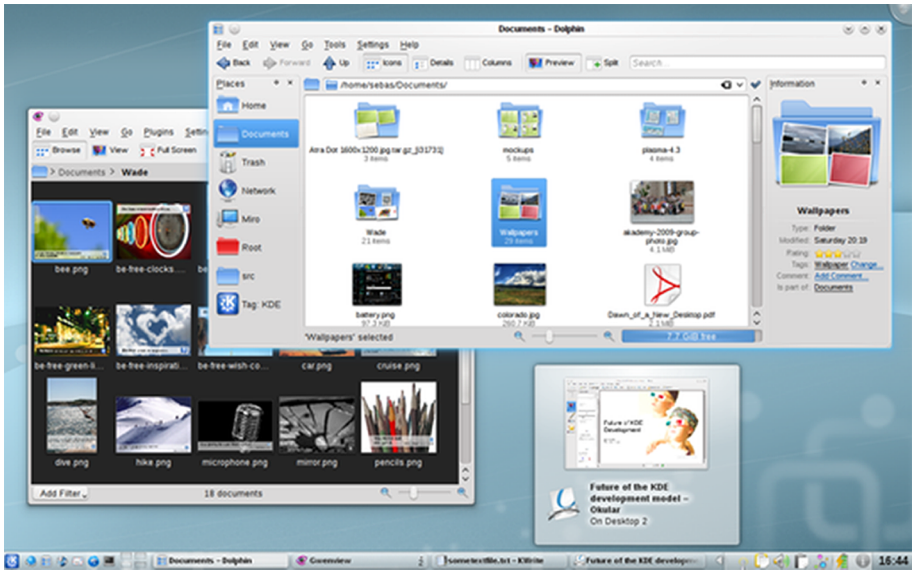
5. L'entorn gràfic

Els sistemes *nix utilitzen una arquitectura d'entorn gràfic anomenada *X-Window*, dissenyada en la dècada dels vuitanta, que és independent de la plataforma i serveix per a qualsevol tipus de *nix. X.Org és una implementació de codi obert del sistema X-Window (que sorgeix com a bifurcació de projecte XFree86) i funciona en mode client/servidor, de manera que no podem connectar gràficament un servidor remot o executar sobre un ordinador local aplicacions gràfiques d'un ordinador remot. Avui dia es treballa juntament amb una infraestructura de DRI (Direct Rendering Infrastructure), que permet aprofitar els xips de processament de les targetes per a estalviar treball de visualització al client X-Window.

El sistema X-Window (basat en una biblioteca anomenada *xlibs*) proporciona els mètodes per a crear les interfícies gràfiques d'usuaris (GUI), però no n'implementa cap sinó que aquestes són proporcionades per jocs d'eines (*toolkits*), que són biblioteques generalment implementades amb *xlibs* i que proporcionen un GUI particular. El gestor de finestres és un servidor especial de X-Window, que s'encarrega de gestionar totes les finestres, els escriptoris, les pantalles virtuals, etc. Òbviament, totes les aplicacions poden funcionar amb qualsevol gestor de finestres, ja que aquest només s'encarrega de gestionar la finestra on està ubicat el programa, i n'hi ha desenes (WMaker, *sawmill, olvwm*, etc.), de manera que l'usuari pot elegir el que més li agradi.

Un altre tipus de programari molt relacionat amb X-Window és el que s'encarrega de proporcionar un entorn integrat per a les aplicacions, l'escriptori, les eines d'administració del sistema, etc. Els més populars actualment són KDE (K Desktop Environment) i GNOME (GNU Network Object Model Environment). Tots dos proporcionen un joc d'eines particular, un entorn d'escriptori amb moltes funcionalitats i configuracions diferents i una llista d'aplicacions integrades, que com més va creix més, i són els més usats en totes les distribucions GNU/Linux. A les figures següents podem veure en primer lloc l'aspecte de KDE i en segon el GNOME (a http://www.xwinman.org/ es poden veure diferents gestors de finestres i entorns):





Actualment, la majoria de les targetes gràfiques del mercat estan suportades i molts fabricants ja donen suport per a GNU/Linux i proporcionen els seus controladors propis (*drivers*).

Per a instal·lar X.Org al nostre ordinador, és necessari baixar els paquets que contenen les eines bàsiques i el programari per al client i el servidor. Generalment, aquests paquets se solen denominar *xorg, xserver-xorg*, etc., i porten implícites diverses dependències de fonts i algunes utilitats bàsiques per al maneig d'X-Window (es pot utilitzar en Debian **apt-cache search Xorg** per a veure els servidors disponibles i **apt-get install <Xorg.server>** per a instal·lar-ne un en particular). Una vegada instal·lats aquests paquets, hem de configurar adequadament els dispositius de què disposem per a poder arrencar correctament el client i el servidor X-Window. En Debian podem utilitzar l'ordre **dexconf**, que utilitzarà la base de dades **debconf** per a generar l'arxiu de configuració de l'entorn gràfic (normalment ubicat a /etc/X11/xorg.conf).

Per a provar la configuració podem executar ***startx*** i una vegada que tenim la pantalla en mode gràfic, en podem canviar la resolució amb les tecles Ctrl-Alt+ Ctrl-Alt o tornar als terminals de text amb Ctrl-Alt-F1-F5, i amb Ctrl-Alt-F7 tornaríem a la gràfica. Amb Ctrl-Alt-Esborrar acabaríem la sessió gràfica i tornaríem al terminal text.

Activitats

1. Analitzeu les ordres següents i els seus paràmetres principals:

passwd, pwd, cd, ls, chmod, chown, chgrp, vi, man, cat, cp, date, df, du, find, mail, mkdir, more, page, mv, rm, rmdir, who, w, ps, cal, clear, zip, gzip, compress, uncompress, dc, diff, env, expr, exit, file, grep, kill, ln, lp, mesg, nice, nohup, quota, sed, awk, ftp, telnet, sleep, sort, su, sudo, tail, tar, tee, pstree, test, users, wc, write, cut, head.

2. Instal·leu VirtualVox sobre un sistema operatiu amfitrió del qual disposeu i carregueu una imatge ja configurada, tot verificant que el sistema funcioni (incloent-hi sistema gràfic, xarxa, accés a disc de l'amfitrió, etc.).

3. Repetiu el punt anterior instal·lant el sistema des dels CD o DVD enviats per la UOC.