Administració local

Josep Jorba Esteve

P07/M2003/02284

UOC
www.uoc.edu

Índex

Introducció 5

1. Distribucions: particularitats 7

2. Nivells d'arrencada i serveis 9

3. Observar l'estat del sistema 12
3.1. Arrencada del sistema 12
3.2. Nucli: directori /proc 13
3.3. Nucli: */sys* 14
3.4. Processos 14
3.5. *Logs* del sistema 15
3.6. Memòria 16
3.7. Discos i sistemes de fitxers 17

4. Sistema de fitxers 20
4.1. Punts de muntatge 21
4.2. Permisos 24

5. Usuaris i grups 26

6. Servidors d'impressió 31
6.1. BSD LPD 34
6.2. LPRng 36
6.3. CUPS 37

7. Discos i gestió sistemes de fitxers 40
7.1. RAID de programari 42
7.2. Volums Lògics (LVM) 47

8. Programari: actualització 51

9. Treballs no interactius 53

10. Taller: pràctiques combinades dels diferents apartats 55

Activitats 63

Altres fonts de referència i informació 63

Introducció

Una de les primeres tasques amb què s'haurà d'enfrontar l'administrador serà la gestió dels recursos locals presents en la màquina. En el curs d'introducció al GNU/Linux, es van cobrir alguns d'aquests aspectes de manera bàsica. En el present, veurem algunes d'aquestes tasques d'administració amb més profunditat, i alguns dels aspectes de personalització i rendiment dels recursos.

Començarem per analitzar el procés d'arrencada d'un sistema GNU/Linux, que ens farà comprendre l'estructura inicial del sistema i la relació amb els diversos serveis que aquest proporciona.

A continuació, aprendrem a obtenir una visió general de l'estat actual del sistema per mitjà dels diferents procediments i ordres de què es disposa per a avaluar les diverses parts del sistema; d'aquesta manera podrem prendre decisions d'administració si detectem algun error o deficiència de rendiment, o la falta d'algun recurs.

Nota

L'administració local engloba moltes tasques variades, que potser siguin les més utilitzades per l'administrador en el seu treball diari.

Un dels punts principals de l'administració és la gestió d'usuaris, ja que qualsevol configuració de la màquina estarà destinada a ser utilitzada per aquests. Veurem com definir nous usuaris al sistema i controlar-ne el nivell d'accés als recursos.

Quant als perifèrics del sistema, com discos i impressores, disposem de diferents possibilitats de gestió, tant per diferents servidors (cas d'impressió) com per diferents sistemes de fitxers que podem tractar, i algunes tècniques d'optimització del rendiment dels discos.

També examinarem el problema de l'actualització del sistema, com el podem mantenir actualitzat, la nova incorporació de programari d'aplicació i com ferlo accessible als usuaris. A més, analitzarem la problemàtica d'executar treballs temporitzats en el sistema.

En el taller final, examinarem l'avaluació de l'estat d'una màquina –seguint els punts vistos en aquest mòdul– i durem a terme algunes de les tasques d'administració bàsiques descrites. En el transcurs de la unitat, comentarem algunes ordres i, després, en el taller, en veurem algunes amb més detall pel que fa al funcionament i opcions.

1. Distribucions: particularitats

Intentem destacar ara algunes diferències tècniques menors (que cada vegada es redueixen més) en les distribucions (Fedora/Red Hat i Debian) utilitzades [Mor03], que veurem amb més detall al llarg de les unitats, a mesura que apareguin.

Canvis o particularitats de Fedora/Red Hat:

- Ús del gestor d'arrencada *grub* (una utilitat GNU). A diferència de versions passades de la majoria de distribucions que solen usar *lilo*, Fedora utilitza *grub*. GRUB (*grand unified bootloader*) té una configuració en mode text (normalment en */boot/grub/grub.conf*) bastant senzilla, i que es pot modificar en l'arrencada. Potser és més flexible que *lilo*. Últimament les distribucions tendeixen a l'ús de *grub*, Debian també l'inclou com a opcional.

Nota

És important conèixer els detalls d'una distribució, ja que poden ser bàsics per a resoldre una tasca o accelerar-ne la solució (per exemple, si disposa d'eines pròpies).

- Gestió d'alternatives. Quan hi ha més d'un programari equivalent per a una tasca concreta, s'indica l'alternativa que s'utilitza mitjançant un directori (*/etc/alternatives*). Aquest sistema es va manllevar de Debian, que en fa un ús ampli en la seva distribució.

- Programa d'escolta de ports TCP/IP basat en *xinetd*. A */etc/xinetd.d* podem trobar en mòduls els fitxers de configuració per a alguns dels serveis TCP/IP, juntament amb el fitxer de configuració */etc/xinetd.conf*. En els sistemes UNIX clàssics, el programa utilitzat és l'*inetd*, que tenia un únic fitxer de configuració a */etc/inetd.conf*. És el cas, per exemple, de la distribució Debian que utilitza *inetd* i deixa *xinetd* com a opció.

- Alguns directoris de configuració especials. Aquests directoris són el */etc/profile.d*, fitxers que s'executen quan un usuari obre un intèrpret d'ordres; */etc/xinetd.d*, configuració d'alguns serveis de xarxa; */etc/sysconfig*, dades de configuració de diversos aspectes del sistema; */etc/cron.*, diversos directoris en què s'especifiquen treballs per a fer periòdicament (mitjançant *crontab*); */etc/pam.d*, en què PAM són els anomenats *mòduls d'autenticació* –en cada un dels fitxers es configuren permisos per al programa o servei particular–; i */etc/logrotate.d*, configuració de rotació (quan cal netejar, comprimir, etc.) d'alguns dels fitxers de diari per a diferents serveis.

- Programari anomenat *kudzu*. Aquest programari examina el maquinari en arrencada per detectar possibles canvis de configuració i generar els dispositius o configuracions adequades. Encara que migra progressivament a l'API Hal que controla, justament, aquest tema.

En el cas de Debian:

- Sistema d'empaquetat propi. Aquest sistema es basa en els paquets DEB, amb eines de diversos nivells per a treballar amb paquets com *dpkg*, *apt-get*, *dselect*, *tasksel*.

- Seguiment de l'FHS sobre l'estructura de directoris. Debian segueix l'FHS, sobre l'estructura de directoris afegint-ne alguns de particulars a */etc* com, per exemple: */etc/default*, fitxers de configuració, i valors per defecte per a alguns programes; */etc/network*, dades i guions de configuració de les interfícies de xarxa; */etc/dpkg* i */etc/apt*, informació de la configuració de les eines de gestió de paquets; */etc/alternatives*, enllaços als programes per defecte, en els que hi ha (o hi pot haver) diverses alternatives disponibles.

- Sistema de configuració de molts paquets de programari per mitjà de l'eina *dpkg-reconfigure*. Per exemple:

  ```
  dpkg-reconfigure gdm
  ```

  permet escollir el gestor d'entrada per a X, o

  ```
  dpkg-reconfigure X-Window-system
  ```

  Que ens permet configurar els diferents elements de X.

- Utilització de configuració de serveis TCP/IP per *inetd*. Que és una configuració en fitxer */etc/inetd.conf*; disposa d'una eina *update-inetd* per a inhabilitar o crear entrades de serveis.

- Alguns directoris de configuració especials. Aquests directoris són */etc/cron.*, diversos directoris en què s'especifiquen treballs per fer periòdicament (mitjançant *crontab*), i */etc/pam.d*, en què PAM són mòduls d'autenticació.

2. Nivells d'arrencada i serveis

Un primer punt important en l'anàlisi del comportament local del sistema és el seu funcionament en els anomenats *nivells d'execució* (o *run levels*), que determinen (en el nivell) el mode actual de treball del sistema, i els serveis que es proporcionen [Wm02].

Un servei és una funcionalitat proporcionada per la màquina, normalment basada en dimonis (o processos en segon pla d'execució, que controlen peticions de xarxa, activitat del maquinari, o altres programes que proveeixin alguna tasca).

L'activació o parada de serveis es realitza utilitzant *scripts*. La majoria dels serveis estàndard –els quals solen tenir la seva configuració en el directori */etc*– se solen controlar mitjançant els *scripts* presents a */etc/init.d/*. En aquest directori, solen aparèixer *scripts* amb noms similars al servei on van destinats, i se solen acceptar paràmetres d'activació o parada. Es realitza de la manera següent:

`/etc/init.d/servei start`	arrencada del servei
`/etc/init.d/servei stop`	parada del servei
`/etc/init.d/servei restart`	aturada i posterior arrencada del servei

Quan un sistema GNU/Linux arrenca, primer es carrega el *kernel* del sistema i, després s'inicia el primer procés, denominat *init*, que és el responsable d'executar i activar la resta del sistema, mitjançant la gestió dels nivells d'execució (o *run levels*).

> Un nivell d'execució és, bàsicament, una configuració de programes i serveis que s'executaran per a un determinat funcionament.

Encara que hi pot haver diferències en l'ordre, en especial en els nivells 2–5 (en la taula, la configuració a Fedora, i la recomanada per l'estàndard LSB), els nivells típics solen ser:

Run level	Funció	Descripció
0	Parada	Finalitza serveis i programes actius, i desmunta sistemes de fitxers actius i per a la CPU.
1	Monousuari	Finalitza la majoria de serveis, i només permet l'entrada de l'administrador (*root*). S'utilitza per a tasques de manteniment i correcció d'errors crítics.
2	Multiusuari sense xarxa	No s'inicien serveis de xarxa, i només permeten entrades locals en el sistema.
3	Multiusuari	Inicia tots els serveis excepte els gràfics associats a l'X Window.
4	Multiusuari	No s'acostuma a fer servir; típicament és igual que el 3.
5	Multiusuari X	Igual que el 3, però amb suport X per a l'entrada d'usuaris (*login* gràfic).
6	Reinici	Per a tots els programes i serveis. Reinicia el sistema.

Per contra, assenyalem que Debian usa un model, en què els nivells 2–5 són pràcticament equivalents, és a dir, realitzen exactament la mateixa funció (encara que podria ser que, en alguna versió, això canviés per coincidir amb l'estàndard LSB).

Aquests nivells solen estar configurats en els sistemes GNU/Linux (i UNIX) per dos sistemes diferents, el BSD o el SystemV (de vegades, abreujat com a *sysV*). En el cas de Fedora i Debian, s'utilitza el sistema SystemV, que és el que mostrarem, però d'altres UNIX i alguna distribució GNU/Linux (com Slackware) utilitzen el model BSD.

En el cas del model *run level* de SystemV, quan el procés *init* arrenca, utilitza un fitxer de configuració anomenat */etc/inittab* per decidir el mode d'execució en què entrarà. En aquest fitxer, es defineixen el *run level* per defecte (*initdefault*) en arrencada (per instal·lació a Fedora el 5, a Debian el 2) i una sèrie de serveis de terminal per activar per a atendre l'entrada de l'usuari.

Després, el sistema, segons el *run level* escollit, consulta els fitxers continguts a */etc/rcn.d*, en què *n* és el número associat al nivell d'execució (nivell escollit), en què es troba una llista de serveis per activar o parar en cas que arrenquem en el *run level,* o l'abandonem. Dins del directori trobarem una sèrie de *scripts* o enllaços als *scripts* que controlen el servei.

Cada *script* té un nom relacionat amb el servei, una S o K inicial que indica si és l'*script* per a iniciar (S) o matar (K) el servei, i un nombre que reflecteix l'ordre en què s'executaran els serveis.

Una sèrie d'ordres de sistema serveixen d'ajuda per a manejar els nivells d'execució. Esmentarem els següents:

- Els *scripts*, que ja hem vist a */etc/init.d/*, ens permeten arrencar, parar o reiniciar serveis individuals.

- *telinit* ens permet canviar de nivell d'execució, només hem d'indicar el nombre. Per exemple, si necessitem fer una tasca crítica en *root* i no hi ha usuaris treballant, podem fer un *telinit* 1 (també es pot utilitzar S), passar a *runlevel* monousuari i, després de la tasca, a un *telinit* 3 per tornar a multiusuari. També es pot utilitzar l'ordre *init* per a la mateixa tasca, encara que *telinit* aporta algun paràmetre extra. Per exemple, el reinici típic d'un sistema UNIX es feia amb *sync; sync; sync; init 6*. L'ordre *sync* força el buidatge dels *buffers* del sistema de fitxers i, després, reiniciem en *runlevel* 6.

- *shutdown* permet parar (*h* de *halt*) o reiniciar el sistema (*r* de *reboot*) en un interval de temps o immediatament. Per a aquestes tasques, també hi ha les ordres *halt* i *reboot*.

- *wall* permet enviar missatges d'advertència als usuaris del sistema. Concretament, l'administrador pot anunciar que es pararà la màquina en un determinat moment. Ordres com *shutdown* els sol utilitzar de manera automàtica.

- *pidof* permet esbrinar el PID (*process* ID) associat a un procés. Amb *ps* obtenim les llistes de processos, i si volem eliminar un servei o procés mitjançant *kill*, necessitarem el seu PID.

Respecte a tot el model d'arrencada, les distribucions presenten algun petit canvi:

- A Fedora/Red Hat, el *runlevel* 4 no té un ús declarat. Els directoris */etc/rcn.d* existeixen com a enllaços cap a subdirectoris de */etc/rc.d*, en què es centralitzen els *scripts* d'arrencada. Els directoris són, així, */etc/rc.d/rcn.d*; però com hi ha els enllaços, és transparent a l'usuari. El *runlevel* per defecte és el 5 amb arrencada amb X.

  Les ordres i fitxers relacionats amb l'arrencada del sistema són en els paquets de programari *sysvinit* i *initscripts*.

  Respecte als canvis de fitxers i guions a Fedora, destaquem les qüestions següents: en */etc/sysconfig*, podem trobar fitxers que especifiquen valors per defecte de la configuració de dispositius o serveis; el guió */etc/rc.d/rc.sysinit* s'invoca una vegada quan el sistema arrenca; el guió */etc/rc.d/rc.local* s'invoca al final del procés de càrrega i serveix per a indicar inicialitzacions específiques de la màquina.

  L'arrencada real dels serveis es fa per mitjà dels guions emmagatzemats a */etc/rc.d/init.d*. També hi ha un enllaç des de /etc/init.d. A més, Fedora proporciona uns *scripts* d'utilitat per a manejar serveis: */sbin/service* per parar o iniciar un servei pel nom i */sbin/chkconfig* per afegir enllaços als fitxers S i K necessaris per a un servei o l'obtenció d'informació sobre els serveis.

- Debian disposa d'ordres de gestió dels *runlevels* com *update-rc.d*, que permet instal·lar o esborrar serveis arrencant-los o parant-los en un o més *runlevels*; *invoke-rc.d* permet les clàssiques accions d'engegar, parar o reiniciar el servei.

  El *runlevel* per defecte a Debian és el 2. L'X Window System no es gestiona des de */etc./inittab*, sinó que hi ha el gestor (per exemple, *gdm* o *kdm*) com si fos un servei més del *runlevel* 2.

3. Observar l'estat del sistema

Una de les tasques principals de l'administrador (*root*), en el seu dia a dia, serà verificar el funcionament correcte del sistema i vigilar l'existència de possibles errors o de saturació dels recursos de la màquina (memòria, discos, etc.). En els subapartats següents, detallarem els mètodes bàsics per a examinar l'estat del sistema en un moment determinat i dur a terme les accions necessàries per evitar problemes.

En el taller final d'aquesta unitat, realitzarem un examen complet d'un sistema exemple, perquè es vegin algunes d'aquestes tècniques.

3.1. Arrencada del sistema

Quan s'arrenca un sistema GNU/Linux, es genera tot un raig d'informació interessant. Quan el sistema arrenca, solen apareixen les dades de detecció de les característiques de la màquina, detecció de dispositius, arrencada de serveis de sistema, etc., i s'esmenten els problemes apareguts.

En la majoria de les distribucions això es pot veure en la consola del sistema directament durant el procés d'arrencada. Tanmateix, o la velocitat dels missatges o algunes distribucions modernes que els oculten al darrere de caràtules gràfiques poden impedir seguir els missatges correctament, per la qual cosa necessitarem una sèrie d'eines per a aquest procés.

Bàsicament, podem utilitzar el següent:

- Ordre *dmesg*. Dóna els missatges de l'última arrencada del *kernel*.

- Fitxer */var/log/messages*. *Log* general del sistema, que conté els missatges generats pel *kernel* i altres dimonis (hi pot haver una gran quantitat de fitxers diferents de log, normalment en /var/log, i depenent de la configuració del servei *syslog*).

- Ordre *uptime*. Indica quant temps fa que el sistema és actiu.

- Sistema */proc*. Pseudosistema de fitxers (*procfs*) que utilitza el *kernel* per emmagatzemar la informació de processos i de sistema.

- Sistema /sys. Pseudosistema de fitxers (*sysfs*) que va aparèixer amb la branca 2.6.x del *kernel*, amb l'objectiu de proporcionar una manera més coherent d'accedir a la informació dels dispositius i els seus controladors (*drivers*).

3.2. Nucli: directori /proc

El *kernel* durant la seva arrencada posa en funcionament un pseudosistema de fitxers anomenat /proc, en què aboca la informació que recopila de la màquina, i moltes de les seves dades internes, durant l'execució. El directori */proc* està implementat sobre memòria i no es desa en disc. Les dades que conté són tant de naturalesa estàtica com dinàmica (varien durant l'execució).

Cal tenir en compte que en ser */proc* molt dependent del *kernel*, afavoreix que la seva estructura depengui del *kernel* del sistema, i l'estructura i els fitxers inclosos poden canviar.

Una de les característiques interessants és que, en el directori /proc, podrem trobar les imatges dels processos en execució, juntament amb la informació que el *kernel* maneja sobre aquests processos. Cada procés del sistema es pot trobar en el directori */proc/<pidproces>*, en què hi ha un directori amb fitxers que representen el seu estat. Aquesta informació és bàsica per a programes de depuració, o bé per a les pròpies ordres del sistema com *ps* o *top*, que la poden utilitzar per a veure l'estat dels processos. En general, moltes de les utilitats del sistema consulten la informació dinàmica del sistema des de /proc (en especial, algunes utilitats proporcionades en el paquet procps).

Nota

El directori /proc és un recurs extraordinari per a obtenir informació de baix nivell sobre el funcionament del sistema, moltes ordres de sistema hi troben suport per a les seves tasques.

D'altra banda, a */proc*, podem trobar altres fitxers d'estat global del sistema. Comentem breument alguns fitxers que podrem examinar per obtenir informació important:

Fitxer	Descripció
/proc/bus	Directori amb informació dels busos PCI i USB.
/proc/cmdline	Línia d'arrencada del *kernel*.
/proc/cpuinfo	Informació de la CPU.
/proc/devices	Llista de dispositius del sistema de caràcters o blocs.
/proc/drive	Informació d'alguns mòduls *kernel* de maquinari.
/proc/filesystems	Sistemes de fitxers habilitats en el kerne.l
/proc/ide	Directori d'informació del bus IDE, característiques de discos.
/proc/interrups	Mapa d'interrupcions de maquinari (IRQ) utilitzades.
/proc/ioports	Ports d'E/S utilitzats.
/proc/meminfo	Dades de l'ús de la memòria.
/proc/modules	Mòduls del nucli.
/proc/mounts	Sistemes de fitxers muntats actualment.
/proc/net	Directori amb tota la informació de xarxa.
/proc/scsi	Directori de dispositius SCSI, o IDE emulats per SCSI.
/proc/sys	Accés a paràmetres del *kernel* configurables dinàmicament.
/proc/version	Versió i data del nucli.

A partir de la branca 2.6 del nucli, s'ha iniciat una transició progressiva de *procfs* (/proc) a *sysfs* (/sys) amb l'objectiu de moure, cap al sistema /sys, tota la informació que no estigui relacionada amb processos, en especial, dispositius i els seus controladors (mòduls del nucli).

3.3. Nucli: */sys*

El sistema Sys s'encarrega de fer disponible la informació de dispositius i controladors, informació de la qual disposa el *kernel*, en l'espai d'usuari de manera que altres API o aplicacions puguin accedir d'una manera flexible a la informació dels dispositius (o els seus controladors). Sol ser utilitzada per capes AL i el servei *udev* per al monitoratge i configuració dinàmica dels dispositius.

Dins del concepte de *sys* hi ha una estructura de dades en arbre dels dispositius i controladors (diguem el model conceptual fix), i de com després s'hi accedeix pel sistema de fitxers *sysfs* (l'estructura del qual pot canviar entre versions).

Quan, en el sistema, es detecta o apareix un objecte afegit, en l'arbre del model de controladors (controladors, dispositius incloent les seves diferents classes) es crea un directori a *sysfs*. La relació pare/fill es reflecteix amb subdirectoris sota */sys/devices/* (reflectint la capa física i els seus identificadors). En el subdirectori */sys/bus* es col·loquen enllaços simbòlics que reflecteixen la manera en què els dispositius pertanyen als diferents busos físics del sistema. I a */sys/class* mostra els dispositius agrupats d'acord amb la seva classe com, per exemple, xarxa, mentre que */sys/block/* conté els dispositius de blocs.

Part de la informació proporcionada per */sys* també es pot trobar a */proc*, però es va considerar que aquest barrejava diferents coses (dispositius, processos, dades de maquinari, paràmetres *kernel*) de manera incoherent, i aquest va ser un dels motius per a crear */sys*. S'espera que, progressivament, es migri informació de */proc* a */sys* per a centralitzar la informació dels dispositius.

3.4. Processos

Els processos que es trobin en execució en un determinat moment seran, en general, de diferent naturalesa:

- **Processos de sistema**, tant si són processos associats al funcionament local de la màquina, *kernel*, com si són processos (denominats *dimonis*) associats al control de diferents serveis. D'altra banda, poden ser locals o de xarxa, depèn de si s'ofereix el servei (actuem de servidor) o rebem els resultats del servei (actuem de client). La majoria d'aquests processos apareixeran associats a l'usuari arrel (*root*), encara que no siguem presents en aquell moment com a usuaris. Hi pot haver alguns serveis associats a altres usuaris de

sistema (*lp*, *bin*, *www*, correu, etc.), que són usuaris "virtuals", no interactius, que utilitza el sistema per executar certs processos.

- **Processos de l'usuari administrador**: en cas d'actuar com a arrel, els nostres processos interactius o aplicacions llançades també apareixeran com a processos associats a l'usuari arrel.

- **Processos d'usuaris del sistema**: associats a l'execució de les seves aplicacions, que poden ser tasques interactives en mode text o en mode gràfic.

Com a ordres ràpides i més útils podem utilitzar les següents:

- *ps*. L'ordre estàndard llista els processos amb les seves dades d'usuari, temps, identificador de procés i línia d'ordres utilitzada. Una de les opcions més utilitzada és *ps -ef* (o *-ax*), però hi ha moltes opcions disponibles (vegeu *man*).

- *top*. Una versió que ens dóna una llista actualitzada a intervals, monitoritzant dinàmicament els canvis. Ens permet ordenar la llista de processos per diferents *ítem* com despesa de memòria, d'ús CPU, amb el propòsit d'obtenir un rànquing dels processos que acaparen els recursos. És molt útil per a donar indicis en situacions extremes de saturació d'ús de recursos, de la possible font de problemes.

- *kill*. Ens permet eliminar processos del sistema mitjançant la tramesa de senyals al procés com, per exemple, la d'acabament *kill -9 pid_del_proces* (9 correspon a SIGKILL), en què indiquem l'identificador del procés. És útil per a processos amb comportament inestable o programes interactius que han deixat de respondre. Podem veure una llista dels senyals vàlids en el sistema amb *man 7 signal*.

3.5. *Logs* del sistema

Tant el *kernel* com molts dels dimonis de serveis, i diferents aplicacions o subsistemes del GNU/Linux, poden generar missatges que vagin a parar a fitxers *log*, per tenir-ne una traça del funcionament o bé per detectar errors o advertències de mal funcionament, o situacions crítiques. Aquest tipus de *logs* són imprescindibles, en molts casos, per a les tasques d'administració i se sol emprar bastant temps d'administració en el processament i anàlisi dels seus continguts.

La majoria dels *logs* es generen en el directori */var/log*. Encara que algunes aplicacions poden modificar aquest comportament, la majoria de *logs* del propi sistema sí que es troben en aquest directori.

Un dimoni particular del sistema (important) és el dimoni *Syslogd*. Aquest dimoni s'encarrega de rebre els missatges que envien el *kernel* i altres dimonis de serveis i els envia a un fitxer *log* que es troba en */var/log/messages*. Aquest és el fitxer per defecte, però Syslogd també és configurable (en el fitxer */etc/syslog.conf*), de manera que es poden generar altres fitxers depenent de la font, segons el dimoni que envia el missatge, i així dirigir-lo a un *log* o a un altre (així classifica per font), o també classificar els missatges per importància (nivell de prioritat): *alarm, warning, error, critical*, etc.

Nota

El dimoni Syslogd és el servei més important d'obtenció d'informació dinàmica de la màquina. El procés d'anàlisi dels logs ens ajuda a entendre el funcionament, els possibles errors i el rendiment del sistema.

Depenent de la distribució, pot estar configurat de diferents maneres per defecte. En */var/log* sol generar (per exemple), a Debian, fitxers com *kern.log, mail.err, mail.info*..., que són els *logs* de diferents serveis. Podem examinar la configuració per a determinar d'on provenen els missatges i en quins fitxers els desa. Una opció que sol ser útil és enviar els missatges a una consola virtual de text (en */etc/syslog.conf* s'especifica per al tipus, o tipus, de missatge una consola de destinació, com */dev/tty8 o /dev/xconsole*), de manera que podrem veure els missatges a mesura que es produeixin. Això sol ser útil per a monitoritzar l'execució del sistema sense haver de mirar els fitxers de *log* a cada moment. Una modificació simple d'aquest mètode podria consistir a introduir, des d'un terminal, la instrucció següent (per al *log* general):

```
tail -f /var/log/messages
```

Aquesta sentència ens permet deixar el terminal o finestra de terminal, de manera que vagin apareixent els canvis que es produeixin en el fitxer.

Altres ordres relacionades:

- *uptime*. Temps que fa que el sistema està actiu. És útil per a comprovar que no hi ha hagut cap reiniciació del sistema inesperada.

- *last*. Analitza el *log* d'entrades i sortides del sistema *(/var/log/wtmp*) dels usuaris, i les reiniciacions del sistema. El *lastlog* controla l'última vegada que els usuaris han estat vistos en el sistema (informació en */var/log/lastlog*).

- Diverses utilitats per al processament combinat de *logs,* que emeten resums (o alarmes) del que ha passat en el sistema com, per exemple, *logwatch*, *logcheck* (Debian), *log_analysis* (Debian)...

3.6. Memòria

Respecte a la memòria del sistema, haurem de tenir en compte que disposem de la memòria física de la pròpia màquina i de memòria virtual, que pot ser encaminada pels processos. Normalment (tret que tractem amb servidors empresarials), no disposarem de quantitats gaire grans, de manera que la memòria física serà menor que la mida de memòria virtual necessària (4 GB en

sistemes de 32 bits). Això obligarà a utilitzar una zona d'intercanvi (*swap*) sobre disc per implementar els processos associats a memòria virtual.

Aquesta zona d'intercanvi (*swap*) es pot implementar com un fitxer en el sistema de fitxers, però és més habitual trobar-la com una partició d'intercanvi (trucada de *swap*) creada durant la instal·lació del sistema. En el moment de particionar el disc, es declara de tipus Linux Swap.

Per a examinar la informació sobre memòria tenim diversos mètodes i ordres útils:

- Fitxer */etc/fstab*. Apareix la partició *swap* (si existeix). Amb una ordre *fdisk* podem esbrinar la seva mida (o consulta a */proc/swaps*).

- Ordre *ps*. Permet saber quins processos tenim, amb les opcions de percentatge i memòria usada.

- Ordre *top*. És una versió *ps* dinàmica actualitzable per períodes de temps. Pot classificar els processos segons la memòria que usen o el temps de CPU.

- Ordre *free*. Informa sobre l'estat global de la memòria. També dóna la mida de la memòria virtual.

- Ordre *vmstat*. Informa sobre l'estat de la memòria virtual i l'ús que se li dóna.

- Alguns paquets com *dstat* permeten recollir dades dels diferents paràmetres (memòria, *swap*, i d'altres) a intervals de temps (de manera semblant a *top*).

3.7. Discos i sistemes de fitxers

Examinarem quins discos tenim disponibles, com estan organitzats, i de quines particions i sistema de fitxers (*file systems*) disposem.

Quan disposem d'una partició, i disposem d'un sistema de fitxers determinat accessible, haurem que realitzar un procés de muntatge per a integrar-la en el sistema, explícitament o bé programada en arrencada. En el procés de muntatge, es connecta el sistema de fitxers associat a la partició a un punt de l'arbre de directoris.

Per a conèixer els discos (o dispositius d'emmagatzematge) que tenim en el sistema, ens podem basar en la informació d'arrencada del sistema (*dmesg*), en què es detecten els presents, com els */dev/hdx* per als dispositius IDE o els SCSI amb dispositius */dev/sdx*. Altres dispositius, com discos durs connectats per USB, discos flaix (els de tipus *pen drive*), unitats amovibles i CD-ROM externs, solen ser

dispositius amb algun tipus d'emulació SCSI, per la qual cosa es veuran com a dispositiu d'aquest tipus.

Qualsevol dispositiu d'emmagatzematge presentarà una sèrie de particions de l'espai. Típicament, un disc IDE suporta un màxim de quatre particions físiques, o més si són lògiques (que permeten col·locar diverses particions d'aquest tipus sobre una de física). Cada partició pot contenir tipus de sistemes de fitxers diferents, d'un mateix operatiu o d'operatius diferents.

Per a examinar l'estructura d'un dispositiu conegut o canviar-ne l'estructura particionant el disc, podem utilitzar l'ordre *fdisk,* o qualssevol de les seves variants més o menys interactives (*cfdisk*, *sfdisk)*. Per exemple, en examinar un disc exemple *ide /dev/hda*, ens dóna la informació següent:

```
# fdisk /dev/hda         (dins opció p)
Disk /dev/hda: 20.5 GB, 20520493056 bytes 255 heads, 63
sectors/track, 2494 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Device     Boot  Start  End   Blocks Id System
/dev/hda1       *      1     1305   10482381   7 HPFS/NTFS
/dev/hda2       *      1306  2429   9028530    83 Linux
/dev/hda3             2430  2494   522112+    82 Linux swap
```

Disc de 20 GB amb tres particions (s'identifiquen amb el número afegit al nom del dispositiu), en què veiem dues particions amb arrencada (columna Boot amb *) de tipus NTFS i Linux, la qual cosa implica l'existència d'un Windows NT/2000/XP/Vista juntament amb una distribució GNU/Linux, i l'última partició que ha utilitzat *swap* per a Linux. A més, tenim informació de l'estructura del disc i de les mides de cada partició.

Alguns discos i particions que tinguem es trobaran muntats en el nostre sistema de fitxers, o estaran preparats per muntar sota demanda o bé muntar en el moment en què es disposi de mitjà (en el cas de dispositius extraïbles).

Aquesta informació la podem obtenir de diferents maneres (veurem més detalls en el taller final):

- Fitxer */etc/fstab*. Indica dispositius que estan preparats per a muntar-se en l'arrencada o els extraïbles que es podran muntar. No sempre hi estan tots els del sistema, sinó sols els que vulguem tenir en arrencada. Els altres els podem muntar sota demanda amb l'ordre *mount*, o desmuntar-los amb *umount*.

- Ordre *mount*. Ens informa dels sistemes de fitxers muntats en aquell moment (tant si són dispositius reals, com sistemes de fitxers virtuals, com */proc*). També podem obtenir aquesta informació des del fitxer */etc/mtab*.

- Ordre *df -k*. Ens informa dels sistemes de fitxers d'emmagatzematge, i ens permet verificar l'espai usat i disponible. És una ordre bàsica per a controlar espai de disc disponible.

Respecte a aquesta darrera ordre *df -k*, una de les nostres tasques bàsiques d'administració de la màquina és controlar els recursos de la màquina i, en aquest cas, l'espai disponible en els sistemes de fitxers utilitzats. Aquests espais s'han de monitoritzar amb certa freqüència per a evitar la caiguda del sistema; mai no s'hauria de deixar un sistema de fitxers (i més si és el /) per sota d'un 10% o 15%, ja que hi ha molts processos dimonis que estan escrivint normalment informació temporal o *logs,* que poden generar molta informació. Un cas particular el formen els fitxers *core,* ja comentats, que poden implicar (depenent del procés) mides de fitxer molt grans. Normalment, caldrà prendre algunes precaucions de "neteja del sistema" si es detecten situacions de saturació del sistema de fitxers:

- S'han d'eliminar els temporals antics. Els directoris */tmp* i */var/tmp* solen acumular molts fitxers generats per diferents usuaris o aplicacions. Alguns sistemes o distribucions ja prenen mesures de neteja, com netejar */tmp* en cada arrencada del sistema.

- S'ha d'evitar el creixement excessiu dels *logs*: segons la configuració del sistema (per exemple de Syslogd) la informació generada de missatges pot ser molt gran. Normalment, caldrà fer neteja periòdicament en arribar a determinades mides i, en tot cas, si necessitem la informació per a anàlisis posteriors, podem realitzar còpies de seguretat en medis extraïbles. Aquest procés es pot automatitzar mitjançant l'ús de *scripts cron* o d'eines especialitzades com *logrotate.*

- Hi ha altres punts del sistema que solen créixer molt, com ara els següents: a) els fitxers *core* dels usuaris –els podem eliminar periòdicament o eliminar-ne la generació–; b) el sistema de correu electrònic, que emmagatzema tots els correus enviats i rebuts –podem demanar als usuaris que facin neteja periòdicament, o bé posar sistemes de quotes–; c) les *cachés* dels navegadors o altres aplicacions –també solen tenir mides grans, i s'haurà de fer neteja periòdicament–; d) els comptes dels usuaris –poden tenir quotes per a no excedir les mides prefixades, etc.

4. Sistema de fitxers

En cada màquina amb un sistema GNU/Linux podem trobar sistemes de fitxers de diferents tipus [Hin].

Per començar, és habitual trobar-se amb els propis sistemes de fitxers Linux creats en diverses particions dels discos [Koe]. La configuració habitual sol ser de dues particions: la corresponent a la barra inclinada (*root file system*) i la corresponent al fitxer d'intercanvi o de *swap*. Encara que, en configuracions més professionals, és habitual separar particions amb parts "diferenciades" del sistema, una tècnica habitual consisteix, per exemple (veurem més opcions després), a crear particions diferents per al següent:

/ /boot /home /opt /tmp /usr /var swap

Que segurament es trobaran muntades des de diferents orígens (diferents discos o, fins i tot, xarxa en alguns casos). La idea és separar clarament parts estàtiques i dinàmiques del sistema, com permetre d'una manera més fàcil, davant de problemes de saturació, estendre les particions, o aïllar més fàcilment parts per a la realització de còpies de seguretat (per exemple els comptes dels usuaris en la partició */home*).

El tipus de particions *swap* és de tipus Linux *swap*, i la corresponent a / sol ser d'algun dels sistemes de fitxers estàndard, que pot ser *ext2* (el tipus per defecte fins als *kernels* 2.4) o el nou *ext3*, que és una millora de l'*ext2*, compatible però amb *journaling*. Això permet tenir un *log* del que passa en el sistema de fitxers, per a recuperacions més ràpides en cas d'error. També són habituals altres sistemes de fitxers com Reiser o XFS.

Una altra configuració habitual és de tres particions: /, *swap*, */home*, en què la */home* es dedicarà als comptes dels usuaris. Això permet separar els comptes dels usuaris del sistema, aïllant-les en dues particions separades, i podem donar l'espai necessari per als comptes en una altra partició.

Un altre esquema molt utilitzat consisteix a separar en particions les parts estàtiques del sistema de les dinàmiques: per exemple, una partició en què es col·locar / amb la part estàtica *(/bin /sbin* i */usr* en alguns casos) que s'espera que no creixerà o ho farà molt poc, i una altra o diverses amb la part dinàmica *(/var /tmp /opt*), suposant que */opt*, per exemple, és el punt d'instal·lació del programari nou. Això permet ajustar millor l'espai de disc i deixar més espai per a les parts del sistema que ho necessitin.

Respecte als sistemes de fitxers suportats, n'hem de destacar la gran varietat. Actualment, podem trobar (entre d'altres) els següents:

- Sistemes associats al GNU/Linux, com l'estàndard *ext2*, i l'*ext3*, evolució de l'anterior amb concepte de *journaling* (suport de *log* d'operacions realitzades en el sistema de fitxers que en pot permetre la recuperació en cas d'algun desastre que el faci inconsistent).

Nota

El document *Filesystems Howto*, dóna breus explicacions dels diversos sistemes de fitxers i adreces web d'interès per a cada un.

- Compatibilitat amb entorns no GNU/Linux: *msdos*, *vfat*, *ntfs*, accés als diferents sistemes de fat16, fat32 i *ntfs*. En particular, destaquem que el suport *kernel*, en el cas del *kernel*, es limita a lectura. Però, com ja hem comentat, hi ha solucions en espai d'usuari (mitjançant FUSE, un component que permet escriure sistemes de fitxers en espai d'usuari) que la permeten, com el *ntfs-ntfs-3g* ja esmentat. També hi ha compatibilitat en altres entorns com Mac amb *hfs* i *hfsplus*.

- Sistemes associats a suports físics, com CD/DVD o els iso9660, i *udf*.

- Sistemes utilitzats en diferents UNIX, que generalment ofereixen millor rendiment (de vegades a costa de més consum de recursos, en CPU per exemple), com JFS2 (IBM), XFS (SGI) o ReiserFS.

- Sistemes de fitxers en xarxa (més tradicionals): NFS, Samba (*smbfs*, *cifs*), que permeten accedir a sistemes de fitxers disponibles en altres màquines de manera transparent per xarxa.

- Sistemes distribuïts en xarxa, com GFS, Coda.

- Pseudosistemes de fitxers, com *procfs* (*/proc*) o *sysfs* (*/sys*).

En la majoria (excepte algun cas especial) d'aquests sistemes de fitxers, el GNU/Linux ens permetrà crear particions d'aquests tipus, construir el sistema de fitxers del tipus requerit i muntar-les com a part integrant de l'arbre de directoris, temporalment o permanentment.

4.1. Punts de muntatge

A part del sistema de fitxers principal / i de les seves possibles divisions en particions extres *(/usr*, */var*, */tmp* i */home*), es pot tenir en compte la possibilitat de deixar punts de muntatge preparats per al muntatge d'altres sistemes de fitxers, que poden ser particions de disc o altres dispositius d'emmagatzematge.

En les màquines en què GNU/Linux comparteix la partició amb altres sistemes operatius mitjançant algun sistema d'arrencada (*lilo* o *grub*), hi pot haver diverses particions assignades als diferents operatius. Moltes vegades, és interessant compartir dades amb aquests sistemes, per llegir-ne els fitxers o

modificar-los. A diferència d'altres sistemes (que només tenen en compte les seves pròpies dades i sistemes de fitxers, i que en algunes versions no se suporten alguns dels seus propis sistemes de fitxers), el GNU/Linux és capaç de tractar, com hem vist, amb una quantitat enorme de sistemes de fitxers de diferents operatius i compartir la informació.

Exemple

Si en els PC personals hem instal·lat el GNU/Linux, segurament trobarem més d'un operatiu, per exemple, una altra versió del GNU/Linux amb *ext2* o 3 de sistema de fitxers, podríem trobar un antic *msdos* amb el seu sistema de fitxers FAT, un Windows98/ME/XP Home amb FAT32 (o *vfat* per al Linux), o un Windows NT/2000/XP/Vista amb sistemes NTFS (*ntfs* per al Linux) i FAT32 (*vfat*) alhora.

El nostre sistema GNU/Linux pot llegir dades (és a dir fitxers i directoris) de tots aquests sistemes de fitxers i escriure en la majoria.

En el cas de l'NTFS, fins a certs moments, hi va haver problemes en l'escriptura que estava en forma experimental en la majoria de *drivers* de *kernels* apareguts. Això va ser degut, principalment, a les diferents versions que van apareixent del sistema de fitxers, ja que hi ha dues versions principals anomenades NTFS i NTFS2, i algunes extensions com els anomenats *volums dinàmics*, o els sistemes de fitxers encriptats. I accedir amb segons quins *drivers* presentava certes incompatibilitats, que podrien causar corrupcions de dades o errors en el sistema de fitxers.

Gràcies al FUSE, un mòdul integrat en el *kernel* (a partir de 2.6.11), s'ha permès un desenvolupament més flexible de sistemes de fitxers, directament en espai d'usuari (de fet, el FUSE actua com un "pont" entre les peticions del *kernel* i l'accés que es fa des del *driver*).

Gràcies a les possibilitats del FUSE, es té un suport més o menys complet de l'NTFS, (mentre Microsoft no faci més canvis en l'especificació), en especial, des de l'aparició del *driver* (basat en FUSE) *ntfs-3g* i la combinació amb les utilitats *ntfsprogs*.

Nota

http://www.ntfs-3g.org

Perquè es puguin llegir o escriure les dades, la partició ha d'estar disponible dins del nostre sistema de fitxers arrel (/). Per tant, cal dur a terme un procés de "muntatge" del sistema de fitxers en algun punt del nostre arbre de directoris. Se seguirà el mateix procés si es tracta d'un dispositiu d'emmagatzematge, tant si és disquet com *floppy*.

Depenent de la distribució, se n'usen uns o altres, o també els podem crear nosaltres. Normalment existeixen o bé com subdirectoris de l'arrel –per exemple */cdrom /win /floppy*– o bé com a subdirectoris dins de */mnt*, el punt estàndard de muntatge (apareixen com a */mnt/cdrom /mnt/floppy*...), o bé el directori */media* preferit últimament per les distribucions. Segons l'estàndard FHS, */mnt* s'hauria d'usar per a muntatges temporals de sistemes de fitxer mentre */media* s'utilitzaria per a muntar dispositius amovibles.

El procés de muntatge es realitza mitjançant l'ordre *mount* amb el format següent:

```
mount -t filesystem-type device mount-point
```

El tipus de sistema de fitxers pot ser *msdos* (*fat*), *vfat* (*fat32*), *ntfs* (*ntfs* lectura), iso9660 (per a *cdrom*)... (dels possibles).

El dispositiu és l'entrada corresponent en el directori */dev* a la localització del dispositiu. Els IDE tenien */dev/hdxy*, en què x és *a,b,c,* o *d* (1 *master*, 1 *slave*, 2 *master*, 2 *slave*) i *y*, el número de partició, els SCSI *(/dev/sdx*) en què *x* és *a,b,c,d*... (segons l'ID SCSI associat 0,1,2,3,4...).

Veurem alguns exemples:

```
mount -t iso9660 /dev/hdc /mnt/cdrom
```

muntaria el CD-ROM (si és l'IDE que està en el segon IDE de forma màster) en el punt */mnt/cdrom*.

```
mount -t iso9660 /dev/cdrom /mnt/cdrom
```

muntaria el CD-ROM. */dev/cdrom* s'usa com a sinònim (és un enllaç) del dispositiu on està connectat.

```
mount -t vfat /dev/fd0H1440 /mnt/floppy
```

muntaria el disquet, */dev/fd0H1440*. Seria la disquetera A en alta densitat (1.44 MB). També es pot utilitzar */dev/fd0*.

```
mount -t ntfs /dev/hda2 /mnt/winXP
```

muntaria la segona partició del primer dispositiu IDE (la C:), de tipus NTFS (per exemple, un Windows XP).

Si aquestes particions són més o menys estables en el sistema (és a dir, no canvien freqüentment) i les volem utilitzar, el millor serà incloure els muntatges perquè es facin en temps d'execució, en iniciar el sistema, mitjançant la configuració del fitxer */etc./fstab*:

```
# /etc/fstab: Informació estàtica del sistema de fitxers
#
#<Sis. fitxers><Punt muntatge><Tipus><Opcions>
<abocament><passada>
/dev/hda2   /         ext3      errors = remountro   0  1
/dev/hdb3   none      swap      sw                   0  0
proc        /proc     proc      defaults             0  0
/dev/fd0    /floppy   auto      user,noauto          0  0
/dev/cdrom  /cdrom    iso9660   ro,user,noauto       0  0
/dev/sdb1   /mnt/usb  vfat      user,noauto          0  0
```

Per exemple, aquesta configuració inclou alguns dels sistemes estàndard com l'arrel a */dev/hda2*, la partició de *swap* que està a *hdb3*, i el sistema *proc* (que utilitza el *kernel* per desar la seva informació). També inclou el disquet, el CD-ROM, i en aquest cas un disc USB de tipus Flash (que es detecta com un dispositiu SCSI). En alguns casos, s'especifica *auto* com a tipus de sistema de fitxers. Això permet que s'autodetecti el sistema de fitxers i, si es coneix, és millor indicar-ho en la configuració. D'altra banda, el *noauto* en les opcions permet que no es munti automàticament sempre, sinó a petició (o accés).

Si tenim aquesta informació en el fitxer, el procés de muntatge se simplifica molt, ja que es farà o en execució, o en arrencada, o a petició (per als *noauto*). I es pot fer ara simplement demanant que es munti el punt de muntatge o el dispositiu:

```
mount /mnt/cdrom
mount /dev/fd0
```

ja que el sistema ja té la resta de la informació.

El procés contrari, el desmuntatge, és bastant senzill, i realitza mitjançant l'ordre *umount* amb punt o dispositiu:

```
umount /mnt/cdrom
umount /dev/fd0
```

En el cas de mitjans extraïbles, de tipus CD-ROM (o d'altres), es pot utilitzar *eject* per a l'extracció del suport físic:

```
eject /dev/cdrom
```

o, en aquest cas, només:

```
eject
```

Les ordres *mount* i *umount* munten o desmunten tots els sistemes disponibles. En el fitxer */etc/mtab*, es manté una llista dels sistemes muntats en un moment concret, que es pot consultar. També es pot executar *mount* sense paràmetres per a obtenir aquesta informació.

4.2. Permisos

Un altre tema que caldrà controlar en el cas dels fitxers i directoris és el dels permisos que volem establir en cada un, recordant que cada fitxer pot disposar de la sèrie de permisos: *rwxrwxrwx* en què es corresponen amb *rwx* del propietari, *rwx* del grup a què l'usuari pertany, i *rwx* per a altres usuaris. En cada un es pot establir el permís de lectura (*r*), escriptura (*w*) o execució (*x*). En el cas

d'un directori, *x* denota el permís per poder entrar en aquest directori (amb l'ordre *cd*, per exemple).

Per modificar els drets sobre un directori o fitxer, hi ha les ordres següents:

- *chown*: canviar propietari dels fitxers
- *chgrp*: canviar grup propietari dels fitxers
- *chmod*: canviar permisos específics (*rwx*) dels fitxers

Aquestes ordres també permeten l'opció -R, que és recursiva si es tracta d'un directori.

5. Usuaris i grups

Els usuaris d'un sistema GNU/Linux disposen normalment d'un compte associat (definit amb algunes de les seves dades i preferències) juntament amb l'espai en disc perquè puguin desenvolupar els seus fitxers i directoris. Aquest espai està assignat a l'usuari, i només ell el pot fer servir (tret que els permisos especifiquin coses diferents).

Dins dels comptes associats a usuaris, en podem trobar de diferents tipus:

- El de l'administrador. Amb identificador *root*, que només és (o hauria de ser) utilitzat per a les operacions d'administració. L'usuari *root* és el que disposa de més permisos, i accés complet a la màquina i als fitxers de configuració. Per tant, també és el que més pot perjudicar per errors o omissions. És millor evitar utilitzar el compte de *root* com si fos un usuari més, per la qual cosa es recomana deixar-lo només per a operacions d'administració.

- Comptes d'usuaris. Els comptes normals per a qualsevol usuari de la màquina tenen els permisos restringits a l'ús de fitxers del seu compte i a algunes altres zones particulars (per exemple, els temporals en */tmp*), i a utilitzar alguns dispositius per als quals se l'hagi habilitat.

- Comptes especials dels serveis. Com ara *lp*, *news*, *wheel*, *www-data*... Són comptes que no són utilitzats per persones, sinó per serveis interns del sistema, que els usa amb aquests noms d'usuari. Alguns dels serveis també s'utilitzen amb el nom de *root*.

Normalment, un usuari es crea mitjançant l'especificació d'un nom (o identificador d'usuari), una contrasenya paraula de pas (*password*) i un directori personal associat (el compte).

La informació dels usuaris del sistema està inclosa en els fitxers següents:

```
/etc/passwd
/etc/shadow
/etc/group
/etc/gshadow
```

Exemple d'unes línies del /etc./passwd:

```
juan:x:1000:1000:Juan Garcia,,,:/home/juan/bin/bash
root:x:0:0:root:/root:/bin/bash
```

en què s'indica (si apareixen :: seguits, que el camp és buit):

- juan: identificador d'usuari en el sistema.
- x: paraula de pas de l'usuari codificada; si hi ha una *x* és que es troba en el fitxer */etc/shadow*.
- 1000: Codi de l'usuari que utilitza el sistema com a codi d'identitat de l'usuari.
- 1000: Codi del grup principal a què pertany; la informació del grup és a */etc/group*.
- Juan García: Se sol posar-se el nom complet de l'usuari.
- /home/juan: Directori personal associat al seu compte.
- /bin/bash: Intèrpret d'ordres interactiu que utilitzarà l'usuari en interactuar amb el sistema, en mode text, o per intèrpret d'ordres gràfic. En aquest cas, és l'intèrpret d'ordres Bash del GNU, que és l'utilitzat per defecte. El fitxer */etc/passwd* solia contenir les contrasenyes dels usuaris encriptades, però el problema era que qualsevol usuari podia veure el fitxer, i en el seu moment es van dissenyar *cracks* que intentaven trobar la contrasenya en forma bruta, mitjançant la contrasenya encriptada com a punt de partida (paraula codificada amb el sistema *crypt*).

Per a evitar això, avui en dia ja no es col·loquen les contrasenyes en aquest fitxer, sinó sols una *x* que indica que es troben en un altre fitxer, que és només de lectura per a l'usuari *root*, */etc/shadow*, el contingut del qual podria ser semblant al següent:

```
juan:a1gNcs82ICst8CjVJS7ZFCVnu0N2pBcn/:12208:0:99999:7:::
```

en què es troba l'identificador de l'usuari juntament amb la contrasenya encriptada. A més, apareixen com a camps separats per dos punts (:):

- Dies des de l'1 de gener de 1970, en el qual la contrasenya es va canviar per última vegada.
- Dies que falten perquè es canviï (0 no se l'ha de canviar).
- Dies després en què s'ha de canviar (és a dir, termini de canvi).
- Dies en què s'avisarà l'usuari abans que li expiri.
- Dies, una vegada expirat, en què es deshabilitarà el compte.

- Dies des de l'1 gener de 1970 en què el compte està deshabilitat.

- I un camp reservat.

A més, les claus d'encriptació poden ser més difícils, ja que ara es pot utilitzar un sistema denominat *md5* (sol aparèixer com a opció a l'hora d'instal·lar el sistema) per a protegir les contrasenyes dels usuaris. Veurem més detalls sobre això en la unitat dedicada a la seguretat.

En /etc/group hi ha la informació dels grups d'usuaris:

```
jose:x:1000:
```

en què tenim:

```
nom-grup:contrasenya-grup:identificador-del-grup:llista-
usuaris
```

La llista d'usuaris del grup pot ser-hi o no. Com que la informació ja està en */etc/passwd*, no se sol posar en */etc/group*. Si s'hi posa, sol aparèixer una mena de llista d'usuaris separada per comes. Els grups també poden tenir una contrasenya associada (encara que no sol ser tan comuna). Com en el cas dels d'usuari, també hi ha un fitxer de tipus *shadow*: /etc/gshadow.

Altres fitxers interessants són els del directori */etc/skel*, en què es troben els fitxers que s'inclouen en cada compte d'usuari en crear-lo. Recordem que, com vam veure amb els intèrprets d'ordres interactius, podíem tenir uns *scripts* de configuració que s'executaven en entrar o sortir del compte. En el directori *skel* es desen els "esquelets" que es copien en cada usuari en crear-lo. L'administrador sol ser el responsable de crear uns fitxers adequats per als usuaris, posant els *paths* necessaris d'execució, inicialització de variables de sistema, variables que es necessitin per al programari, etc.

A continuació, veurem una sèrie d'ordres útils per a aquesta administració d'usuaris (n'esmentem la funcionalitat i, en el taller, farem algunes proves):

- *useradd*: Afegir un usuari al sistema.

- *userdel*: Esborrar un usuari del sistema.

- *usermod*: Modificar un usuari del sistema.

- *groupadd*, *groupdel*, *groupmod*: El mateix per a grups.

- *newusers*, *chpasswd*: Poden ser d'utilitat en grans instal·lacions amb molts usuaris, ja que permeten crear diversos comptes des de la informació intro-

duïda en un fitxer (o*newusers*) o canviar les contrasenyes a un gran nombre d'usuaris (*chpasswd).*

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

- *chfn*: Canviar la informació de l'usuari, la present en el comentari del fitxer */etc/passwd*.

- *passwd*: Canvia la contrasenya d'un usuari. Es pot executar com a usuari i, llavors, demana la contrasenya antiga i la nova. En el cas de fer-ho, el *root* ha d'especificar l'usuari a qui canviarà la contrasenya (si no, canviaria la seva) i no necessita la contrasenya antiga. És potser l'ordre més utilitzada pel *root*, per als usuaris quan se'ls oblida la contrasenya antiga.

- *su:* Una mena de canvi d'identitat. La utilitzen tant usuaris com el *root* perquè canviï l'usuari actual. En el cas de l'administrador, és bastant utilitzat per a verificar que el compte de l'usuari funciona correctament. Hi ha diferents variants: *su* (sense paràmetres, serveix per a passar a usuari arrel o *root* després d'identificar-se, la qual cosa permet passar a *root* per fer alguna tasca quan estem en un compte d'usuari,); la sentència *su iduser* (canvia l'usuari a *iduser*, però deixant l'entorn com està, és a dir, en el mateix directori...); el mandat *su - iduser* (fa una substitució total, com si el segon usuari hagués entrat en el sistema fent un *login*).

Respecte a l'administració d'usuaris i grups, el que hem comentat aquí fa referència a l'administració local d'una sola màquina. En sistemes amb moltes màquines que comparteixen els usuaris se sol utilitzar un altre sistema de gestió de la informació dels usuaris. Aquests sistemes, denominats genèricament *sistemes d'informació de xarxa*, com NIS, NIS+ o LDAP, utilitzen bases de dades per emmagatzemar la informació dels usuaris i grups, de manera que s'utilitzen màquines servidores, en què s'emmagatzema la base de dades, i altres màquines clients, en què es consulta aquesta informació. Això permet tenir una sola còpia de les dades dels usuaris (o diverses de sincronitzades), i que aquests puguin entrar en qualsevol màquina disponible del conjunt administrat amb aquests sistemes. A més, aquests sistemes incorporen conceptes addicionals de jerarquies, o dominis/zones de màquines i recursos, que permeten representar adequadament els recursos i el seu ús en organitzacions amb diferents estructures d'organització del seu propi personal i les seves seccions internes.

Podem comprovar si estem en un entorn de tipus NIS si en les línies *passwd* i *group* del fitxer de configuració */etc/nsswitch.conf* apareix *compat,* si treballem amb els fitxers locals o amb *nis* o *nisplus* segons el sistema que fem servir. En general, per a l'usuari simple no comporta cap modificació, ja que la gestió de les màquines li és transparent, i més si es combina amb fitxers compartits per NFS, que permet disposar del seu compte sense que importi amb quina màquina treballi. La majoria de les ordres anteriors es poden continuar utilitzant

sense problema amb NIS o NIS+. Són equivalents a excepció del canvi de contrasenya, que en lloc de *passwd,* se sol fer amb *yppasswd* (NIS) o *nispasswd* (NIS+); encara que és habitual que l'administrador els canviï el nom (per un enllaç) a *passwd*, amb la qual cosa els usuaris no notaran la diferència.

Veurem aquest i altres modes de configuració en les unitats d'administració de xarxa.

6. Servidors d'impressió

El sistema d'impressió del GNU/Linux [Gt] [Smi02] és heretat de la variant BSD del UNIX. Aquest sistema es denominava LPD (*line printer daemon*). Aquest és un sistema d'impressió molt potent, ja que integra capacitats per gestionar tant impressores locals com de xarxa, i ofereix, dins mateix, tant el client com el servidor d'impressió.

L'LPD és un sistema bastant antic, ja que es remunta als orígens de la branca BSD del UNIX (mitjan els vuitanta). Per tant, a l'LPD, li sol faltar suport per als dispositius moderns, ja que en origen el sistema no es va pensar per al tipus d'impressores actuals. El sistema LPD no es va pensar com un sistema basat en controladors de dispositiu, ja que normalment només es produïen impressores sèrie o paral·leles d'escriptura de caràcters text.

Per a la situació actual, el sistema LPD es combina amb un altre programari comú, com el sistema Ghostscript, que ofereix sortida de tipus *postscript* per a un rang molt ampli d'impressores per a les quals té controladors. A més, se sol combinar amb algun programari de filtratge, que segons el tipus de document a imprimir, selecciona filtres adequats. Així, normalment, se segueix (bàsicament)el procés següent:

Nota

Els sistemes UNIX disposen, potser, dels sistemes d'impressió més potents i complexos, que aporten una gran flexibilitat als entorns d'impressió.

1) El treball és iniciat per una ordre del sistema LPD.

2) El sistema de filtre identifica quin tipus de treball (o fitxer) s'utilitza i converteix el treball en un fitxer *postscript* de sortida, que és el que s'envia a la impressora. En el GNU/Linux i UNIX, la majoria d'aplicacions suposen que la sortida serà cap a una impressora *postscript*, i moltes generen sortida *postscript* directament. Per aquesta raó, es necessita el pas següent.

3) Ghostscript s'encarrega d'interpretar el fitxer *postscript* rebut i, segons el controlador de la impressora a què s'ha enviat el treball, realitza la conversió al format propi de la impressora. Si és de tipus, *postscript* la impressió és directa, si no, caldrà realitzar la traducció. El treball s'envia a la cua d'impressió.

Nota

Ghostscript:
http://www.cs.wisc.edu/ghost/

A més del sistema d'impressió LPD (amb origen en els BSD UNIX), també hi ha el denominat sistema System V (d'origen en l'altra branca UNIX de System V). Normalment, per compatibilitat, la majoria de UNIX integren actualment ambdós sistemes, de manera que un és el principal i l'altre se simula sobre el principal. En el cas del GNU/Linux, passa una cosa semblant, ja que segons la instal·lació que fem podem tenir només les ordres LPD de sistema d'impressió, però també serà habitual disposar de les ordres System V. Una manera senzilla d'identificar tots dos sistemes (BSD o Sys-

tem V) és amb l'ordre principal d'impressió (el que envia els treballs al sistema) que, en BSD, és *lpr*, i a System V és *lp*.

Aquest era el panorama inicial dels sistemes d'impressió del GNU/Linux, però en els últims anys han sorgit més sistemes, que permeten una major flexibilitat i una major disposició de controladors per a les impressores. Els dos sistemes principals són CUPS i, en menor grau, LPRng. De fet, últimament, CUPS és l'estàndard *de facto* per al GNU/Linux, encara que els altres sistemes han de ser suportats per compatibilitat amb sistemes UNIX existents.

Nota

LPRng: http://www.lprng.org
CUPS: http://www.cups.org

Tots dos (tant CUPS com LPRng) són una mena de sistemes de nivell més alt, però que per a l'usuari no es diferencien gaire dels BSD i System V estàndard: per exemple, s'utilitzen les mateixes ordres client (o compatibles en opcions) per a imprimir. Per a l'administrador sí que suposarà diferències, ja que els sistemes de configuració són diferents. En certa manera, podem considerar LPRng i CUPS com a noves arquitectures de sistemes d'impressió, que per a l'usuari són compatibles amb les ordres antigues.

En les distribucions GNU/Linux actuals ens podem trobar amb els diferents sistemes d'impressió. Si la distribució és antiga, pot ser que només porti incorporat el sistema BSD LPD. En les actuals, tant Debian com Fedora/Red Hat utilitzen CUPS. En algunes versions de Red Hat hi havia una eina, *Print switch*, que permetia canviar el sistema, commutar de sistema d'impressió, encara que últimament només hi ha disponible CUPS. A Debian es poden instal·lar ambdós sistemes, però són exclusius, i només un pot gestionar la impressió.

En el cas de Fedora Core, el sistema d'impressió per defecte és CUPS (LPRng desapareix a Fedora Core 4), i l'eina *Print switch* ja no existeix perquè no és necessària, s'utilitza *system-config-printer* per a la configuració de dispositius. Debian utilitza BSD LPD per defecte, però és comú instal·lar CUPS (i és previsible que sigui l'opció per defecte en noves revisions), i també pot utilitzar LPRng. A més, recordem que també teníem la possibilitat (vista en la unitat de migració) d'interaccionar amb sistemes Windows mitjançant protocols Samba, que permetien compartir les impressores i l'accés a aquestes.

Respecte a cada un dels sistemes [Gt]:

- BSD LPD. És un dels estàndards del UNIX, i algunes aplicacions assumeixen que tindran les ordres i el sistema d'impressió disponibles, per la qual cosa, tant LPRng com CUPS emulen el funcionament i les ordres de BDS LPD. El sistema LPD és utilitzable, però no gaire configurable, sobretot en el control d'accés. Per això les distribucions s'han mogut als altres sistemes més moderns.

- LPRng. Es va dissenyar, bàsicament, per a reemplaçar el BSD, per tant, la major part de la configuració és semblant i únicament difereix en alguns fitxers de configuració.

- CUPS. Es tracta d'una desviació major del BSD original, i la configuració és pròpia. Es proporciona informació a les aplicacions sobre les impressores disponibles (també en LPRng). A CUPS tant el client com el servidor han de disposar de programari CUPS.

Tots dos sistemes tenen emulació de les ordres d'impressió de System V.

Per a la impressió en GNU/Linux, cal tenir en compte diversos aspectes:

- Sistema d'impressió que s'utilitza. BSD, LPRng o CUPS.

- Dispositiu d'impressió (impressora). Pot tenir connexió local a una màquina o estar col·locada en xarxa. Les impressores actuals poden estar col·locades per connexions locals a una màquina mitjançant interfícies sèrie, paral·lel, USB, etc. O es poden posar simplement en xarxa, com una màquina més, o amb protocols especials propietaris. Les connectades a xarxa, poden actuar normalment elles mateixes de servidor d'impressió (per exemple, moltes làser HP són servidors BSD LPD), o bé es poden penjar d'una màquina que actuï de servidor d'impressió per a elles.

- Protocols de comunicació utilitzats amb la impressora, o el sistema d'impressió. Pot ser TCP/IP directe (per exemple, una HP amb LPD) o bé d'altres de nivell més alt sobre TCP/IP, com IPP (CUPS), JetDirect (algunes impressores HP), etc. Aquest paràmetre és important, ja que l'hem de conèixer per a instal·lar la impressora en un sistema.

- Sistema de filtres utilitzats. Cada sistema d'impressió en suporta un o diversos.

- Controladors de les impressores. En el GNU/Linux n'hi ha de bastants tipus diferents. Podem esmentar, per exemple, controladors de CUPS, propis o dels fabricants (per exemple, HP i Epson els proporcionen); Gimp, programa de retoc d'imatges que també té controladors optimitzats per a la impressió d'imatges; Foomatic, sistema de gestió de controladors que funciona amb la majoria de sistemes (CUPS, LPD, LPRng, i d'altres); els controladors de Ghostscript; etc. Gairebé totes les impressores tenen un o més controladors d'aquests conjunts.

Nota

Es pot trobar informació de les impressores més adequades i dels controladors a:

http://www.linuxprinting.org/foomatic.html

Respecte a la part clienta del sistema, les ordres bàsiques són iguals per als diferents sistemes. Les ordres del sistema BSD (cada sistema suporta emulació d'aquestes ordres) són les següents:

- *lpr*. Envia un treball a la cua de la impressora per defecte (o a la que se selecciona), el dimoni d'impressió (*lpd*) s'encarrega d'enviar-lo a la cua corresponent i assigna un número de treball, que s'utilitzarà amb les altres ordres. Normalment, la impressora per defecte està indicada per una variable de sistema PRINTER, o s'utilitza la primera que hi hagi definida, o en alguns sistemes s'utilitza la cua *lp* (com a nom per defecte).

Exemple

Exemple d'lpr:

```
lpr -Pepson dades.txt
```

Aquesta instrucció enviaria el fitxer dades.txt a la cua d'impressió associada a una impressora que hem definit com a *epson*.

- *lpq*: ens permet examinar els treballs que hi ha en la cua.

Exemple

Exemple:

```
# lpq -P epson
Rank Owner   Job Files    Total    Size
1st          juan         15       dades.txt     74578 bytes
2nd          marta        16       fpppp.F       12394 bytes
```

Aquesta ordre ens mostra els treballs en cua, amb el seu ordre i les seves mides. Els fitxers poden aparèixer amb noms diferents, ja que depèn de si els hem enviat amb *lpr*, o amb una altra aplicació que els pot canviar de nom en enviar-los, o de si han hagut de passar per algun filtre en convertir-los.

- *lprm*. Elimina treballs de la cua. Podem especificar un nombre de treball o un usuari per a cancel·lar els treballs.

Exemple

```
lprm -Pepson 15
```

Eliminar el treball amb *id* 15 de la cua.

Respecte a la part administrativa (a BSD), l'ordre principal seria lp. Aquesta ordre permet activar, desactivar cues, moure treballs en l'ordre de les cues i activar o desactivar les impressores (es poden rebre treballs en les cues però no s'envien a les impressores).

També podem esmentar que, per al cas de System V, les ordres d'impressió solen estar disponibles, normalment simulades sobre els de BSD. En el cas client, les ordres són *lp*, *lpstat* i *cancel*, i per a temes d'administració són *lpadmin*, *accept*, *reject*, *lpmove*, *enable*, *disable*, *lpshut*.

En les seccions següents, veurem com s'ha de configurar un servidor d'impressió per als tres sistemes principals. Aquests servidors serveixen tant per a la impressió local com per a atendre les impressions de clients de xarxa (si estan habilitats).

6.1. BSD LPD

En el cas del servidor BSD LPD, hi ha dos fitxers principals per examinar: d'una banda, la definició de les impressores en */etc/printcap*, i de l'altra, els permisos d'accés per xarxa en */etc/hosts.lpd*.

Respecte al tema dels permisos, per defecte BSD LPD només deixa accés local a la impressora i, per tant, cal habilitar-lo expressament en */etc/hosts.lpd*.

Exemple

El fitxer podria ser el següent:

```
#fitxer hosts.lpd
second
first.the.com
192.168.1.7
+@groupnis
-three.the.com
```

que indicaria que està permesa la impressió en una sèrie de màquines, llistades pel seu nom DNS o per l'adreça IP. Es poden afegir grups de màquines que pertanyin a un servidor NIS (com en l'exemple *groupnis*) o bé permetre accés a determinades màquines indicant-ho amb un guió (-).

Respecte a la configuració del servidor en */etc/printcap*, es defineixen entrades, en què cada una representa una cua del sistema d'impressió a la qual poden anar a parar els treballs. La cua pot estar tant associada a un dispositiu local, com a un servidor remot, tant si aquest és una impressora com un altre servidor.

En cada entrada hi pot haver les opcions següents:

- *lp* =. Ens indica a quin dispositiu està connectada la impressora, per exemple, *lp = /dev/lp0* indicaria el primer port paral·lel. Si la impressora és de tipus LPD, per exemple, una impressora de xarxa que accepta el protocol LPD (com una HP), podem deixar el camp buit i omplir els següents.
- *rm* =. Ens indica l'adreça amb el nom o IP de la màquina remota que disposa de la cua d'impressió. Si es tracta d'una impressora de xarxa, serà l'adreça d'aquesta.
- *rp* =. Ens indica el nom de la cua remota. A la màquina s'indica abans amb *rm*.

Vegem un exemple:

```
# Entrada d'una impressora local
lp|epson|Epson C62:\
 :lp=/dev/lp1:sd=/var/spool/lpd/epson:\
 :sh:pw#80:pl#72:px#1440:mx#0:\
 :if = /etc/magicfilter/StylusColor@720dpi-filter:\filtre
 :af = /var/log/lp-acct:lf = /var/log/lp-errs:
# Entrada d'impressora remota
hpremota|hpr|hp remota del departament|:\
 :lp = :\
 :rm = servidor:rp = colahp:\
 :lf = /var/adm/lpd_rem_errs:\fitxer de log.
 :sd = /var/spool/lpd/hpremota:spool local associat
```

6.2. LPRng

En el cas del sistema LPRng –ja que es va fer per a mantenir la compatibilitat amb BSD i, entre d'altres, millorar qüestions d'accés–, el sistema és compatible en la configuració de cues, i es du a terme amb el mateix format de fitxer de */etc/printcap*, amb alguns afegits propis.

On la configuració resulta diferent és en l'accés. En aquest cas, es realitza per mitjà d'un fitxer */etc/lpd.perms* en general per a tot el sistema, i també hi poden haver configuracions individuals de cada cua, amb el fitxer *lpd.perms*, col·locat en el directori corresponent a la cua, normalment */var/spool/lpd/nom-cua*.

Aquests fitxers *lpd.perms* tenen una capacitat superior de configurar l'accés, i permeten les ordres bàsiques següents:

```
DEFAULT ACCEPT
DEFAULT REJECT
ACCEPT [ key = value[,value]* ]*
REJECT [ key = value[,value]* ]*
```

En què les dues primeres ens permeten establir el valor per defecte d'acceptar-ho tot o rebutjar-ho tot, i les dues següents, d'acceptar o rebutjar una configuració concreta especificada en la línia. Es poden acceptar (o rebutjar) peticions d'un *host*, usuari o ports IP específics. A més, es pot configurar quin tipus de servei es proporcionarà a l'element: X (es pot connectar), P (impressió de treballs), Q (examinar cua amb lpq), M (esborrar treballs de la cua, *lprm*), C (control d'impressores, ordre *lpc*), entre altres, així en el fitxer:

```
ACCEPT SERVICE = M HOST = first USER = jose
ACCEPT SERVICE = M SERVER REMOTEUSER = root
REJECT SERVICE = M
```

Es permet esborrar treballs de la cua a l'usuari (*jose*) de la màquina (*first*) i a l'usuari *root* del servidor on hi hagi el servei d'impressió (*localhost*); a més, es rebutgen qualssevol altres peticions d'esborrar de la cua treballs que no siguin les peticions ja establertes.

Amb aquesta configuració s'ha d'anar amb molt de compte perquè, en algunes distribucions, els serveis LPRng són oberts per defecte. És pot limitar la connexió per exemple amb el següent:

```
ACCEPT SERVICE = X SERVER
REJECT SERVICE = X NOT REMOTEIP = 100.200.0.0/255
```

El servei de connexió només és accessible a la màquina local del servidor, i es rebutja si no pertany a la nostra subxarxa (en aquest cas, suposem que és 100.200.0.x).

Per a l'administració de línia d'ordres, s'usen les mateixes eines que el BSD, estàndard. En l'administració gràfica del sistema, es pot destacar l'eina *lprngtool* (no disponible en totes les versions del sistema LPRng).

Figura 1. *lprngtool*, configuració d'una impressora

Hi ha diversos paquets de programari relacionats amb LPRng, per exemple, en una Debian trobem els següents:

```
lprng - lpr/lpd printer spooling system
lprng-doc - lpr/lpd printer spooling system (documentation)
lprngtool - GUI frontend to LPRng based /etc/printcap
printop - Graphical interface to the LPRng print system.
```

6.3. CUPS

CUPS és una nova arquitectura per al sistema d'impressió bastant diferent, té una capa de compatibilitat cap a BSD LPD, que li permet interaccionar amb servidors d'aquest tipus. També suporta un nou protocol d'impressió anomenat IPP (basat en *http*), però només està disponible quan client i servidor són de tipus CUPS. A més, utilitza un tipus de *drivers* denominats PPD que identifiquen les capacitats de la impressora. CUPS ja porta alguns d'aquests controladors, i alguns fabricants també els ofereixen (cas d'HP i Epson).

CUPS té un sistema d'administració completament diferent, basat en diferents fitxers: */etc/cups/cupsd.conf* centralitza la configuració del sistema d'impressió, */etc/cups/printers.conf* controla la definició d'impressores, i */etc/cups/classes.conf* els grups d'aquestes.

En */etc/cups/cupsd.conf*, configurem el sistema segons una sèrie de seccions del fitxer i les directrius de les diferents accions. El fitxer és bastant gran i destacarem algunes directrius importants:

- Allow. Ens permet especificar quines màquines podran accedir al servidor, que poden ser grups o màquines individuals, o segments IP de xarxa.
- AuthClass. Permet indicar si es demanarà que s'autentifiquin els usuaris clients o no.
- BrowseXXX. Hi ha una sèrie de directrius relacionades amb la possibilitat d'examinar la xarxa per a trobar impressores servides. Aquesta possibilitat està activada per defecte (*browsing* en *on*), per tant, normalment trobarem disponibles totes les impressores disponibles en la xarxa. La podem desactivar per veure només les impressores que hàgim definit. Una altra opció important és BrowseAllow, que diu a qui donem la possibilitat de preguntar per les nostres impressores; per defecte està habilitada, per la qual cosa qualsevol pot veure la nostra impressora des de la nostra xarxa.

Figura 2. Interfície per a l'administració del sistema CUPS

Destaquem que, en principi, CUPS està pensat perquè tant els clients com el servidor funcionin amb el mateix sistema. Si els clients utilitzen LPD o LPRng, cal instal·lar un dimoni de compatibilitat anomenat *cups-lpd* (normalment, en paquets com *cupsys-bsd*). En aquest cas, CUPS accepta treballs que provinguin d'un sistema LPD o LPRng, però no controla els accessos (*cupsd.conf* només serveix per al propi sistema CUPS), per tant, caldrà implementar alguna estratègia de control d'accés, de tipus tallafoc, per exemple (vegeu la unitat de seguretat).

Per a l'administració des de línia d'ordres, CUPS és una mica peculiar, ja que accepta tant ordres LPD com System V en els clients, i l'administració sol utilitzar l'ordre *lpadmin* de SystemV. Quant a eines gràfiques, disposem de *gnome-cups-manager*, *gtklp* o la interfície per a web que porta el mateix sistema CUPS, accessible en *http://localhost:631*.

Respecte als paquets de programari relacionats amb CUPS, en una Debian trobem (entre d'altres):

```
cupsys - Common UNIX Printing System(tm) - server
cupsys-bsd - Common UNIX Printing System(tm) - BSD commands
cupsys-client - Common UNIX Printing System(tm) - client programs (SysV)
cupsys-driver-gimpprint - Gimp-Print printer drivers for CUPS
cupsys-pt - Tool for viewing/managing print jobs under CUPS
cupsomatic-ppd - linuxprinting.org printer support - transition package
foomatic-db - linuxprinting.org printer support - database
foomatic-db-engine - linuxprinting.org printer support - programs
foomatic-db-gimp-print - linuxprinting - db Gimp-Print printer drivers
foomatic-db-hpijs - linuxprinting - db HPIJS printers
foomatic-filters - linuxprinting.org printer support - filters
foomatic-filters-ppds - linuxprinting - prebuilt PPD files
foomatic-gui - GNOME interface for Foomatic printer filter system
gimpprint-doc - Users' Guide for GIMP-Print and CUPS
gimpprint-locales - Locale data files for gimp-print
gnome-cups-manager - CUPS printer admin tool for GNOME
gtklp - Frontend for cups written in gtk
```

7. Discos i gestió sistemes de fitxers

Respecte a les unitats d'emmagatzematge, com hem vist, tenen una sèrie de dispositius associats, depenent del tipus d'interfície:

- IDE: dispositius

  /dev/hda disc màster, primer connector IDE

  /dev/hdb disc *slave* del primer connector

  /dev/hdc màster segon connector

  /dev/hdd slave segon connector

- SCSI: dispositius */dev/sda /dev/sdb*... seguint la numeració que tinguin els perifèrics en el Bus SCSI.

- Disquets: dispositius */dev/fdx*, amb *x* número de disquetera (començant en 0). Hi ha diferents dispositius depenent de la capacitat del disquet, per exemple, el disquet d'1.44 MB a la disquetera A seria */dev/fd0H1440*.

Respecte a les particions presents, el número que segueix el dispositiu representa l'índex de la partició dins del disc, i es tracta com un dispositiu independent: */dev/hda1* primera partició del primer disc IDE, o */dev/sdc2*, segona partició del tercer dispositiu SCSI. En el cas dels discos IDE, aquests permeten quatre particions anomenades *primàries*, i un major nombre en lògiques. Així, si en */dev/hdan n* és inferior o igual a 4, es tractarà d'una partició primària, si no, es tractarà d'una partició lògica amb *n* superior o igual a 5.

Amb els discos i els sistemes de fitxers (*file systems*) associats, els processos bàsics que podem realitzar els englobem en el següent:

- Creació o modificació de particions. Mitjançant ordres com *fdisk* o semblants (*cfdisk*, *sfdisk)*.

- Formatatge de disquets. Per als disquets, es poden utilitzar diferents eines: *fdformat* (formatació de nivell baix), *superformat* (formatació a diferents capacitats en format *msdos*), *mformat* (formatació específica creant sistemes de fitxers *msdos* estàndard).

- Creació de sistemes de fitxers Linux, en particions, mitjançant l'ordre *mkfs*. Hi ha versions específiques per a crear sistemes de fitxers diversos *mkfs.ext2*,

mkfs.ext3, i també sistemes de fitxers diferents de Linux: *mkfs.ntfs*, *mkfs.vfat*, *mkfs.msdos*, *mkfs.minix* o d'altres. Per a CD-ROM hi ha versions com *mkisofs* per a crear els iso9660 (amb extensions *joliet* o *rock ridge*), que puguin ser una imatge del que després s'acabarà gravant sobre un CD/DVD. Després, mitjançant ordres com *cdrecord*, es podran crear o gravar els CD/DVD. Un altre cas particular és l'ordre *mkswap*, que permet crear àrees de *swap* en particions, que després es poden activar o desactivar amb *swapon* i *swapoff*.

- Muntatge dels sistemes de fitxers. Mitjançant les ordres *mount, umount*.

- Verificació d'estat. La principal eina de verificació de sistemes de fitxers Linux és l'ordre *fsck*. Aquesta ordre comprova les diferents àrees del sistema de fitxers per verificar-ne la consistència, comprovar possibles errors i, si és possible, corregir-los. El propi sistema activa automàticament l'ordre en l'arrencada quan detecta situacions en què s'ha produït una parada incorrecta (una apagada elèctrica o accidental de la màquina), o bé quan ha detectat aquestes situacions un cert nombre de vegades en què el sistema s'ha engegat; aquesta comprovació acostuma a durar una estona, normalment alguns minuts (depenent de la mida de dades). També hi ha versions particulars per a altres sistemes de fitxers: *fsck.ext2*, *fsck.ext3*, *fsck.vfat*, *fsck.msdos*, etc. Normalment, el procés de l'*fsck* es realitza amb el dispositiu en mode de *només lectura* amb particions muntades. Es recomana desmuntar les particions per realitzar el procés si es detecten errors i cal aplicar correccions. En determinats casos, per exemple, si el sistema per comprovar és l'arrel / i es detecta algun error crític, se'ns demanarà que canviem de mode d'execució del sistema (*runlevel*) pel mode només *root*, i llavors fem la verificació. En general, si cal fer la verificació, es recomana el mode *superusuari* (podem commutar en mode *runlevel* amb les ordres *init* o *telinit*).

- Processos de còpia de seguretat. Poden ser del disc, blocs de disc, particions, sistemes de fitxers, fitxers... Hi ha diverses eines útils per a les còpies de seguretat: *tar* ens permet copiar fitxers en un fitxer o en unitats de cinta; *cpio*, de manera semblant, pot realitzar còpies de seguretat de fitxers en un fitxer; tant *cpio* com *tar* mantenen informació de permisos i propietaris dels fitxers; *dd* permet còpies, tant de fitxers com de dispositius, particions o discos a fitxer; és una mica complex i cal tenir informació de baix nivell, tipus, mida, bloc o sector, i també es pot enviar a cintes.

- Utilitats diverses. Algunes ordres individuals, algunes utilitzades pels processos anteriors per a fer tractaments diversos, com *badblocks* per a trobar blocs defectuosos en el dispositiu; *dumpe2fs* per a obtenir informació sobre sistemes de fitxers Linux; i *tune2fs*, que permet fer processos de *tunning* de sistemes de fitxers Linux de tipus *ext2* o *ext3* i ajustar diferents paràmetres de comportament.

A continuació destaquem dos temes relacionats amb la concepció de l'espai d'emmagatzematge, que són utilitzats en diversos ambients per a la creació

base de l'espai d'emmagatzematge: l'ús de RAID de programari i la creació de volums dinàmics.

7.1. RAID per programari

Les configuracions de discos mitjançant esquemes RAID és un dels esquemes d'emmagatzematge d'alta disponibilitat més utilitzats actualment quan disposem de diversos discos per a implementar els nostres sistemes de fitxers.

L'enfocament principal de les diferents tècniques es basa en la tolerància a diferents tipus d'errors, tant físics com de sistema, per a evitar les pèrdues de dades o els errors de coherència en el sistema. Aquesta tolerància es proporciona des d'un nivell de dispositiu, el conjunt de discos. A més a més, l'enfocament de les diferents tècniques també es basa en alguns esquemes que estan dissenyats per a augmentar les prestacions del sistema de discos, ampliant l'amplada de banda d'aquests disponible per al sistema i les aplicacions.

Avui en dia, podem trobar RAID en maquinari, principalment en servidors empresarials (encara que comencen a tenir certa presència en equips d'escriptori), en què hi ha disponibles diferents solucions de maquinari que compleixen aquests requisits, en particular, per a aplicacions intensives en disc, com *streaming* d'àudio o vídeo, o grans bases de dades.

En general, aquest maquinari es troba en forma de targetes (o integrat a la màquina) de tipus controladores RAID de discos, que implementen la gestió d'un o més nivells (de l'especificació RAID) sobre un conjunt de discos administrat per aquesta controladora.

En la RAID, es distingeixen una sèrie de nivells (o configuracions possibles) que es poden proporcionar (cada fabricant de maquinari, o el programari concret, pot suportar un o més d'aquests nivells). Cada nivell de RAID s'aplica sobre un conjunt de discos, de vegades anomenat *array* RAID (o matriu de discos RAID), que solen ser discos de la mateixa mida (o iguals a mides de grups). Per exemple, per a realitzar un cas d'*array*, es podrien utilitzar 4 discos de 100 GB o, en un altre cas, per exemple, 2 grups (a 100 GB) de 2 discos, un de 30 GB i un altre de 70 GB. En alguns casos de controladors maquinari, no es permet que els discos (o els grups) siguin de diferents mides, i en d'altres es poden utilitzar, però l'*array* queda definit per la mida del disc (o grup) més petit.

Descrivim conceptes bàsics d'alguns nivells en la llista següent (tingueu en compte que, en alguns casos, la terminologia no és plenament acceptada, i pot dependre de cada fabricant):

- RAID 0. Les dades es distribueixen equitativament entre un o més discos sense informació de paritat o redundància, i no s'ofereix tolerància a l'error. Només es reparteixen dades. Si el disc falla físicament, la informació es perd i l'hem de recuperar a partir de còpies de seguretat. El que sí augmenta

és el rendiment, depenent de la implementació de RAID 0, ja que les operacions de lectura i escriptura es dividiran entre els diferents discos.

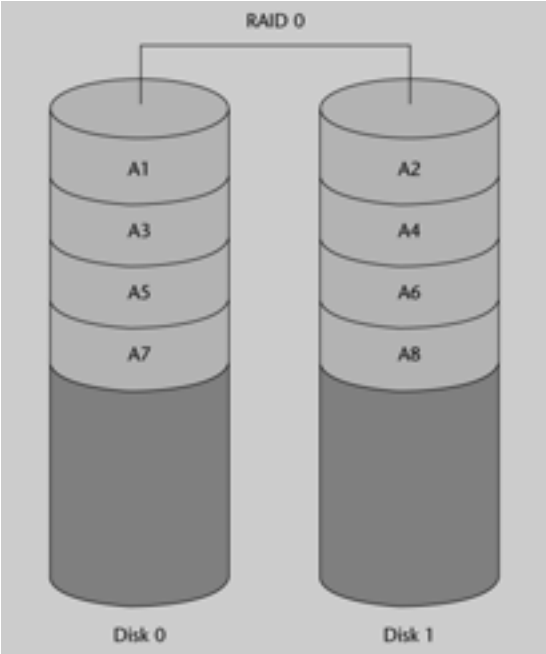

Figura 3

- RAID 1. Es crea una còpia exacta (*mirror*) en un conjunt de dos o més discos (denominat *array* RAID). En aquest cas, resulta útil per al rendiment de lectura (que es pot incrementar linealment amb el nombre de discos) i, en especial, per a disposar de tolerància a l'error d'un dels discos, ja que (per exemple, amb dos discos) es disposa de la mateixa informació. RAID 1 sol ser adequat per a alta disponibilitat, com entorns de 24 x 7, en què hem de disposar críticament dels recursos. Aquesta configuració també ens permet (si el maquinari el suporta) l'intercanvi en calent dels discos. Si detectem l'error en un d'ells, el podem substituir, sense apagar el sistema, per un disc nou.

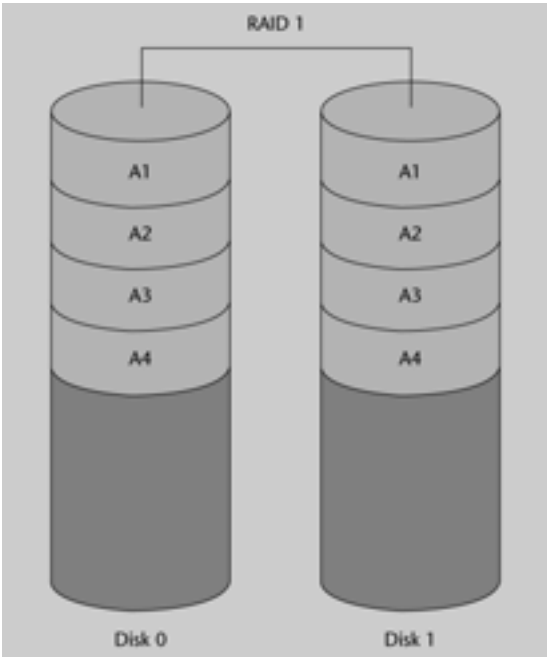

Figura 4

- RAID 2. En els anteriors es divideixen les dades en blocs a repartir; aquí, es divideixen en bits i s'utilitzen codis de redundància, per a la correcció de dades. Pràcticament no s'utilitza, malgrat les altes prestacions que assoliria, ja que idealment necessita un nombre molt alt de discos, un per bit de dades, i diversos per al càlcul de la redundància (per exemple, en un sistema de 32 bits, arribaria a utilitzar 39 discos).

- RAID 3. Utilitza divisió en bytes amb un disc dedicat a la paritat dels blocs. Tampoc no s'utilitza gaire, ja que segons la mida de les dades i posicions no permet accessos simultanis. RAID 4 és semblant –encara que es divideix a nivell de blocs en lloc de bytes–, la qual cosa permet que sí es puguin servir peticions simultànies quan se sol·licita un únic bloc.

- RAID 5. S'usa divisió en l'àmbit de blocs, i es distribueix la paritat entre els discos. S'utilitza molt a causa de l'esquema senzill de paritat i perquè aquest càlcul s'implementa de manera senzilla per maquinari, amb bones prestacions.

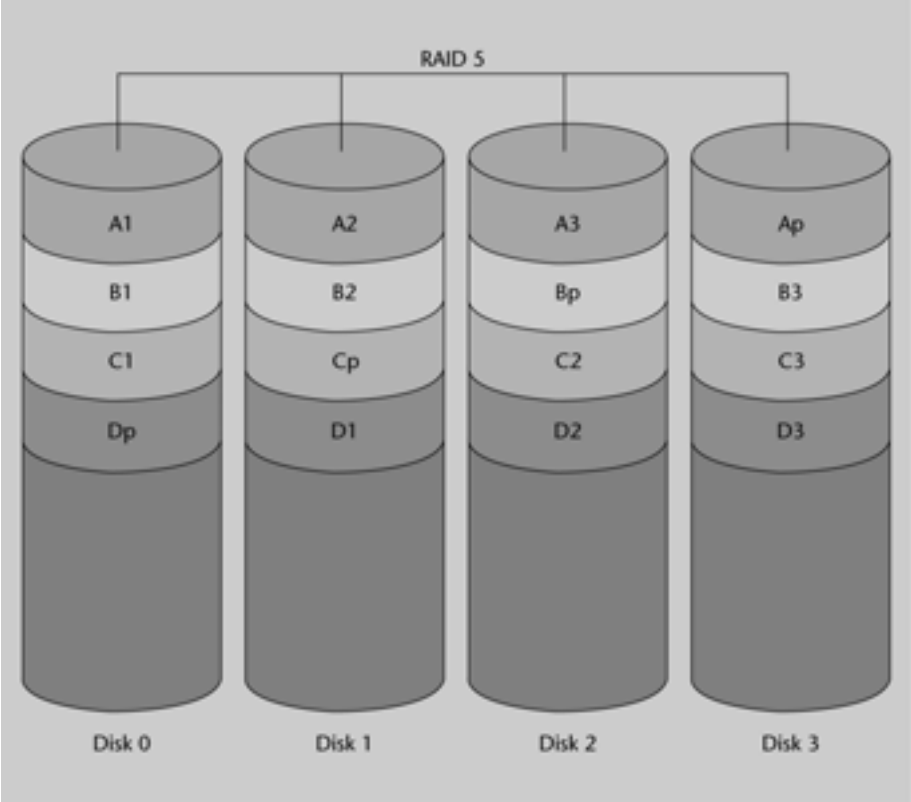

Figura 5

- RAID 0 +1 (o 01). És un *mirror* de divisions, es tracta d'un nivell de RAID imbricat, en què s'implementen, per exemple, dos grups de RAID 0, que s'utilitzen en RAID 1 per a crear *mirror* entre elles. Un avantatge és que, en cas d'error, es pot reconstruir el nivell de RAID 0 utilitzat gràcies a l'altra còpia, però si es volen afegir discos cal afegir-los a tots els grups de RAID 0 de la mateixa manera.

- RAID 10 (1 +0). És la divisió de *mirrors*, grups de RAID 1 sota RAID 0. Així, en cada grup de RAID 1 pot arribar a fallar un disc sense que es perdin dades. És clar que això obliga a reemplaçar-los, ja que, si no, el disc que queda en el grup es converteix en possible punt d'error de tot el sistema. És una configuració que s'acostuma a utilitzar per a base de dades d'altes prestacions (per la tolerància a errors i la velocitat en no estar basada en càlculs de paritat).

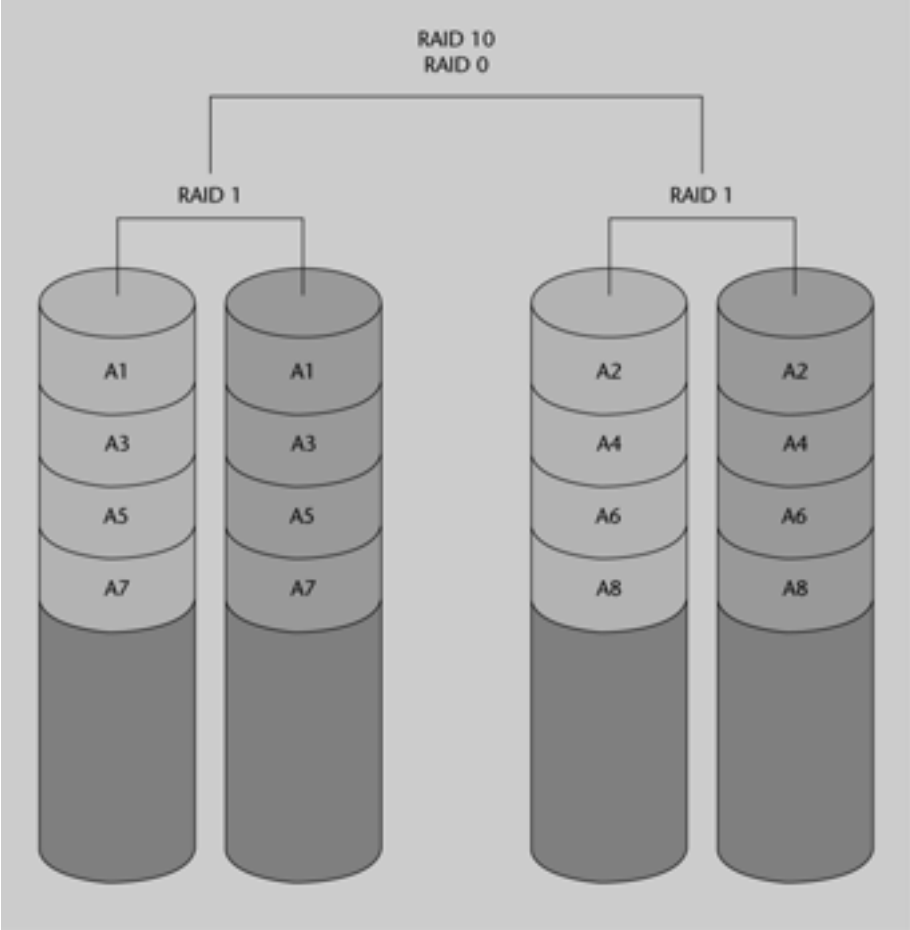

Figura 6

Algunes consideracions a tenir en compte sobre la RAID en general:

- RAID millora l'*uptime* del sistema, ja que alguns dels nivells permeten que hi hagi discos que fallen i el sistema segueixi sent consistent i depengui del maquinari; fins i tot, es pot canviar el maquinari problemàtic en calent sense necessitat de parar el sistema, qüestió especialment important en sistemes crítics.

- RAID pot millorar el rendiment de les aplicacions, en especial, en els sistemes amb implementacions de *mirror* és possible que la divisió de dades permeti que les operacions lineals de lectura s'incrementin significativament, a causa de la possibilitat que els discos ofereixin simultàniament parts d'aquesta lectura, la qual cosa augmentaria la taxa de transferència de dades.

- RAID no protegeix les dades. Evidentment, no hi ha protecció contra la destrucció per altres mitjans (virus, mals funcionaments generals, o desastres naturals). Ens hem de basar en esquemes de còpies de seguretat.

- No se simplifica la recuperació de dades. Si un disc pertany a un *array* RAID, s'ha d'intentar recuperar en aquest ambient. Es necessita programari específic o els controladors de maquinari per a accedir a les dades.

- Al contrari, no sol millorar aplicacions típiques d'usuari, encara que siguin d'escriptori, perquè aquestes aplicacions tenen components alts d'accés aleatori a dades i a conjunts de dades petites, per la qual cosa pot ser que no es beneficiïn de lectures lineals o de transferències de dades sostingudes. En aquests ambients, és possible que amb prou feines es noti una millora de prestacions.

- No es facilita el trasllat d'informació. Sense RAID és bastant fàcil traslladar dades, simplement movent el disc d'un sistema a un altre. En el cas de RAID, és gairebé impossible (tret que disposem del mateix maquinari) moure un *array* de discos a un altre sistema.

En el cas del GNU/Linux, es dóna suport al maquinari RAID mitjançant diversos mòduls de *kernel*, associats a diferents conjunts de fabricants o circuits base, *chipsets*, d'aquestes controladores RAID. Així, es permet al sistema abstreure's dels mecanismes de maquinari i fer-los transparents al sistema i a l'usuari final. En tot cas, aquests mòduls de *kernel* ens permeten l'accés als detalls d'aquestes controladores i a la seva configuració de paràmetres de nivell molt baix que, en alguns casos (especialment, en servidors que suporten càrrega elevada d'E/S), poden ser interessants per a processos de *tunning* del sistema de discos que utilitzi el servidor, amb la finalitat de maximitzar les prestacions del sistema.

L'altra possibilitat que analitzarem aquí és la realització d'aquests processos mitjançant components de programari, en concret, el component de programari RAID del GNU/Linux.

El GNU/Linux disposa en *kernel* de l'anomenat Multiple Device (*md*), que podem considerar com el suport mitjançant *driver* del *kernel* per a RAID. Amb aquest *driver*; podem implementar nivells de RAID, generalment, 0,1,4,5 i imbricats (per exemple, RAID 10) sobre diferents dispositius de bloc com discos IDE o SCSI. També disposa del nivell *linear* com a nivell en què es produeix una combinació lineal dels discos disponibles (en què tant és que siguin de mides diferents), de manera que s'escriu consecutivament en els discos.

Per fer servir la RAID de programari en el Linux, hem de disposar del suport RAID en el *kernel* i, si escau, dels mòduls *md* actius, a més d'alguns *drivers* específics segons el cas (vegeu *drivers* disponibles associats a RAID, per exemple,

a Debian amb *modconf*). El mètode preferit per a la implementació d'*arrays* de discos RAID, mitjançant el programari RAID ofert pel Linux, és mitjançant (o bé durant la instal·lació) o bé mitjançant la utilitat *mdadm*. Aquesta utilitat ens permet crear els *arrays* i gestionar-los.

Vegem alguns exemples (imaginem uns discos SCSI */dev/sda*, */dev/sdb*... en els quals disposem de diverses particions disponibles per a implementar RAID):

Creació d'un *array linear:*

```
# mdadm -create -verbose /dev/md0 -level=linear -raid-devices=2 /dev/sda1 /dev/sdb1
```

En què es crea un *array* linear a partir de les particions primeres de */dev/sda* i */dev/sdb*, la qual cosa crea el nou dispositiu */dev/md0*, que ja pot ser usat com a nou disc (suposant que existeixi el punt de muntatge */media/discoRAID*):

```
# mkfs.ext2fs /dev/md0
# mount /dev/md0 /media/discoRAID
```

Per a un RAID 0 o RAID 1 podem canviar simplement el nivell (*-level*) a *raid0* o *raid1*. Amb *mdadm -detail /dev/md0* podrem comprovar els paràmetres del nou *array* creat.

També podem consultar l'entrada *mdstat* en */proc* per determinar els *arrays* actius i els seus paràmetres. En especial, en els casos amb *mirror* (per exemple en els nivells 1, 5...), podrem veure la reconstrucció inicial de les còpies en la seva creació. En */proc/mdstat* indicarà el nivell de reconstrucció (i el temps aproximat de finalització).

mdadm disposa de moltes opcions que ens permeten examinar i gestionar els diferents *arrays* RAID per programari creats (podem veure una descripció i exemples en *man mdadm*).

Una altra consideració important són les optimitzacions a què es poden sotmetre els *arrays* RAID per a millorar-ne el rendiment, tant per monitoritzar-ne el comportament per optimitzar paràmetres del sistema de fitxers, com per realitzar un ús més efectiu dels nivells RAID i les seves característiques.

Nota

L'optimització dels *arrays* RAID pot ser una font important de sintonització del sistema. Es recomana examinar algunes qüestions en Software-RAID-Howto, o en la pròpia pàgina *man* de *mdadm*.

7.2. Volums Lògics (LVM)

D'altra banda, sorgeix la necessitat d'abstreure's del sistema físic de discos, la seva configuració i nombre de dispositius, perquè el sistema (operatiu) s'encarregui d'aquest treball i no ens hàgim de preocupar d'aquests paràmetres directament. En aquest sentit, es pot veure el sistema de volums lògics com una capa de virtualització de l'emmagatzematge que permet una visió més simple que faciliti la utilització fluida i senzilla.

En el *kernel* Linux es disposa d'LVM (*logical volume manager*), que es basa en idees desenvolupades de gestors de volums d'emmagatzematge utilitzats en HP-UX (una versió UNIX propietària d'HP). Actualment, hi ha dues versions, i la LVM2 és la més utilitzada per una sèrie de prestacions afegides.

L'arquitectura d'una LVM consisteix típicament en els components (principals):

- **Volums físics** (PV). Són els discos durs o les seves particions, o qualsevol altre element que aparegui com un disc dur per al sistema (per exemple, un RAID de programari o maquinari).

- **Volums lògics** (LV). És l'equivalent a la partició del disc físic. Aquesta LV és visible en el sistema com un dispositiu de blocs (absolutament equivalent a una partició física), i pot contenir un sistema de fitxers (per exemple, el */home* dels usuaris). Normalment, els volums tenen més sentit per als administradors, ja que es poden utilitzar noms per a identificar-los (així, podem utilitzar un dispositiu lògic, anomenat *estoc*, o màrqueting, en lloc d'*hda6* o *sdc3*).

- **Grups de volums** (VG). És l'element de la capa superior. La unitat administrativa que engloba els nostres recursos, que poden ser volums lògics (LV) o físics (PV). En aquesta unitat, es desen les dades dels PV disponibles, i com es formen les LV a partir dels PV. Evidentment, per a poder utilitzar un grup VG, hem de disposar de suports físics PV, que s'organitzin en diferents unitats lògiques LV.

Per exemple, en la figura següent, veiem un grup de volums, en què disposem de 7 PV, en forma de particions de discos, que s'han agrupat per formar dos volums lògics (que s'han acabat utilitzant per a formar els sistemes de fitxers de */usr* i */home*):

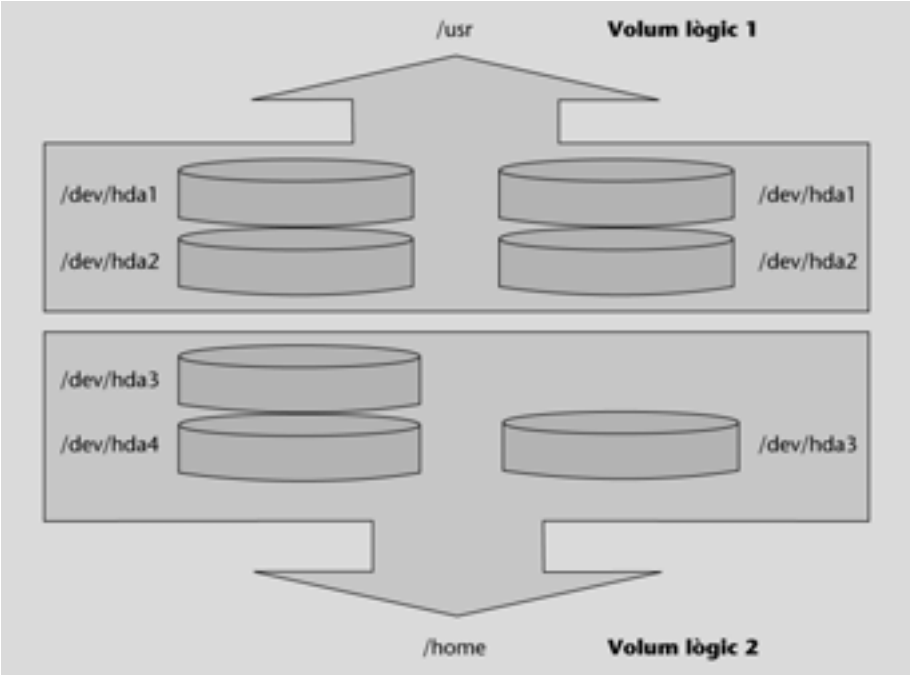

Figura 7. Esquema d'un exempe d'LVM

Amb l'ús dels volums lògics, permetem un tractament més flexible de l'espai en el sistema d'emmagatzematge (que podria tenir un gran nombre de discos i particions diferents), segons les necessitats que ens apareguin, i gestionar l'espai, tant per identificadors més adequats com per operacions que ens permetin adequar les necessitats a l'espai disponible en cada moment.

Els sistemes de gestió de volums ens permeten el següent:

- Redimensionar grups i volums lògics, aprofitant nous PV o extraient-ne alguns dels disponibles inicialment.
- Tenir instantànies del sistema de fitxers (lectura en LVM1, i lectura o escriptura en LVM2). Això permet crear un nou dispositiu que sigui una instantània en el temps de la situació d'una LV. A més, permet crear la instantània, muntar-la, provar diverses operacions, o configuració nova de programari, o altres elements, i si no funciona com esperàvem permet tornar el volum original al seu estat abans de les proves.
- RAID 0 de volums lògics.

En LVM no s'implementen configuracions de RAID de tipus 1 o 5, si són necessàries (és a dir redundància i tolerància a error), llavors, o bé s'utilitza programari de RAID o bé controladora de maquinari RAID que l'implementi, i col·loquem LVM com a capa superior.

Vegem un breu exemple de creació típica (en molts casos, l'instal·lador de la distribució realitza un procés semblant, si permetem un LVM com a sistema inicial d'emmagatzematge). Bàsicament, es fa el següent: 1) la creació dels volums físics (PV), 2) la creació del grup lògic (VG), 3) la creació del volum lògic i 4) la utilització per a creació d'un sistema de fitxers i el seu muntatge posterior:

1) Per exemple, disposem de tres particions de diferents discos, de manera que es creen tres PV i s'inicialitza el contingut:

```
# dd if=/dev/zero of=/dev/hda1 bs=1k count=1
# dd if=/dev/zero of=/dev/hda2 bs=1k count=1
# dd if=/dev/zero of=/dev/hdb1 bs=1k count=1
# pvcreate /dev/hda1
Physical volume "/dev/sda1" successfully created
# pvcreate /dev/hda2
Physical volume "/dev/hda2" successfully created
# pvcreate /dev/hdb1
Physical volume "/dev/hdb1" successfully created
```

2) Ho col·loquem en un VG creat dels diferents PV:

```
# vgcreate grup_discos /dev/hda1 /dev/hda2 /dev/hdb1
Volume group "grup_discos" successfully created
```

3) Creem l'LV (en aquest cas mesura 1 GB) a partir dels elements que tenim en el grup VG (*-n* indica el nom del volum):

```
# lvcreate -L1G -n volum_logic grup_discos
lvcreate -- doing automatic backup of "grup_discos"
lvcreate -- logical volume "/dev/grup_discos/ volum_logic"
successfully created
```

I, finalment, creem un sistema de fitxers (en aquest cas, un *reiser*):

```
# mkfs.reiserfs /dev/grup_discos/volum_logic
```

Que, per exemple, podríem col·locar d'espai de còpia de seguretat

```
# mkdir /mnt/backup
# mount -t reiserfs /dev/grup_discos/volum_logic /mnt/backup
```

Finalment, disposem del dispositiu com un volum lògic que implementa un sistema de fitxers del nostra màquina.

8. Programari: actualització

Per a l'administració de la instal·lació o actualització del programari en el nostre sistema, dependrem en primera instància del tipus de paquets de programari que utilitzi el nostre sistema:

- RPM. Paquets que utilitza la distribució Fedora/Red Hat (i derivades). Se solen manejar mitjançant l'ordre *rpm*. Contenen informació de dependències del programari amb d'altres, a alt nivell mitjançant *Yum* (o*up2date* en algunes distribucions derivades de Red Hat).

- DEB. Paquets de Debian que se solen manejar amb un conjunt d'eines que treballen a diferents nivells amb paquets individuals o grups. Entre aquestes, podem esmentar *dselect*, *tasksel*, *dpkg*, i *apt-get*.

- TAR, o bé els *tgz* (també *tar.gz*). Són purament paquets de fitxers units i comprimits mitjançant ordres estàndard com *tar* i *gzip* (aquestes s'utilitzen per a la descompressió). Aquests paquets no contenen informació de dependències i, normalment, es poden instal·lar en llocs diferents, si no és que porten informació de ruta (*path*) absoluta.

Per a manejar aquests paquets, hi ha diverses eines gràfiques. Per exemple, RPM té Kpackage; DEB té Synaptic i Gnome-apt; Tgz té Kpackage. Aquest paquets també es poden manejar des del propi gestor de fitxers gràfics (en Gnome o KDE). També hi acostumen a haver utilitats de conversions de paquets. Per exemple, a Debian tenim l'ordre *alien,* que permet convertir paquets RPM a DEB. Encara que cal prendre precaucions, perquè el paquet, per tenir una distribució destinació diferent, no modifiqui algun comportament o fitxer de sistema no esperat.

Depenent de l'ús dels tipus de paquets o eines, l'actualització o instal·lació del programari del nostre sistema es produirà de maneres diferents:

1) Des dels propis CD d'instal·lació del sistema. Normalment, totes les distribucions busquen el programari en seus CD. Però cal tenir en compte que aquest programari no ha de ser antic i, per aquesta raó, no ha d'incloure alguns pedaços com actualitzacions o noves versions amb més prestacions. Per aquest motiu, si s'instal·la a partir de CD, és bastant comú verificar després que no hi hagi cap versió més recent.

2) Mitjançant serveis d'actualització o recerca de programari, que poden ser gratuïts, com el cas de l'eina *apt-get* de Debian, o *yum* a Fedora, o serveis de subscripció (de pagament o amb facilitats bàsiques) com el Red Hat Network de les versions Red Hat comercials.

3) Mitjançant reposadors de programari que ofereixen paquets de programari preconstruïts per a una distribució determinada.

4) Mitjançant el propi creador o distribuïdor del programari, que ofereix una sèrie de paquets d'instal·lació del seu programari. És possible que no trobem el tipus de paquets necessari per a la nostra distribució.

5) Mitjançant programari sense empaquetament o amb un empaquetament només de compressió sense cap tipus de dependències.

6) Mitjançant només codi font, en forma de paquet o de fitxer comprimit.

9. Treballs no interactius

En les tasques d'administració, sol ser necessària l'execució a intervals de certes tasques, per programar les tasques per realitzar-les en horaris de menor ús de la màquina, o bé per la pròpia naturalesa periòdica de les tasques, que es vulguin desenvolupar.

Per realitzar aquest tipus de treballs "fora d'hores", com serveis periòdics o programats, hi ha diversos sistemes que ens permeten construir un tipus d'agenda de tasques (planificació d'execució de tasques):

- *nohup* pot ser el cas més simple utilitzat pels usuaris. Els permet l'execució d'una tasca no interactiva una vegada hagin sortit del seu compte. Normalment, en sortir del compte, l'usuari perd els seus processos, però *nohu* permet deixar-los en execució, encara que l'usuari es desconnecti.

- *at* ens permet llançar una acció per a més tard, programant un determinat instant en què s'iniciarà, especificant-ne l'hora (hh:mm) i data, o bé especificant si es farà avui (*today*) o demà (*tomorrow*). Exemples:

  ```
  at 10pm tasca
  ```

  la tasca es realitza a les deu de la nit.

  ```
  at 2am tomorrow tasca
  ```

  la tasca es realitza a les dues de la matinada.

- *cron* permet establir una llista de treballs per a realitzar amb la seva programació. Aquesta configuració es desa en */etc./crontab*; concretament, en cada entrada d'aquest fitxer tenim els minuts i l'hora en què s'efectuarà la tasca, quin dia del mes, quin mes, quin dia de la setmana, i juntament amb què (pot ser una tasca, o bé un directori en què hi hauran les tasques a executar). Per exemple, de manera estàndard el contingut és semblant a aquest:

```
25 6 * * * root test -e /usr/sbin/anacron || run-parts --report /etc/cron.daily
47 6 * * 7 root test -e /usr/sbin/anacron || run-parts --report /etc/cron.weekly
52 6 1 * * root test -e /usr/sbin/anacron || run-parts --report /etc/cron.monthl
```

en què es programa que una sèrie de tasques es faran cada dia (* indica *qualsevol*), setmanalment (el 7è. dia de la setmana) o mensualment (l'1 de cada mes). Normalment, els treballs serien executats per l'ordre *crontab*, però el sistema *cron* suposa que la màquina sempre està engegada; si no és així, és millor uti-

litzar *anacron*, que verifica si l'acció no es va realitzar quan ho hauria hagut de fer, i l'executa. En cada línia del fitxer anterior es verifica que hi hagi l'ordre *anacron* i s'executen els *scripts* associats a cada acció, que en aquest cas estan desats en uns directoris assignats per a això, els *cron*.

També hi poden haver uns fitxers *cron.allow* i *cron.deny* per a limitar qui pot col·locar (o no) treballs en *cron*. Mitjançant l'ordre *crontab*, un usuari pot definir treballs en el mateix format que hem vist abans, els quals es desaran habitualment en */var/spool/ cron/crontabs*. En alguns casos, també hi ha un directori */etc/cron.d* en què es poden col·locar treballs i que és tractat com si fos una extensió del fitxer */etc/crontab*.

10. Taller: pràctiques combinades dels diferents apartats

Començarem per examinar l'estat general del nostre sistema. Farem els diferents passos en un sistema Debian. Es tracta d'un sistema Debian *unstable* (la versió inestable, però més actualitzada); tanmateix, els procediments són majoritàriament traslladables a altres distribucions com Fedora / Red Hat (esmentarem alguns dels canvis més importants). El maquinari consisteix en un Pentium 4 a 2.66 GHz amb 768 MB i diversos discos, DVD i gravador de CD, a més d'altres perifèrics, però ja obtindrem aquesta informació pas a pas.

Vegem, primer, com s'ha engegat el nostre sistema l'última vegada:

```
# uptime
17:38:22 up 2:46, 5 users, load average: 0.05, 0.03, 0.04
```

Aquesta ordre ens indica quant temps fa que el sistema està "llevat" des que es va arrencar l'última vegada, 2 hores 46 minuts. En el nostre cas, tenim cinc usuaris, que no cal que siguin cinc usuaris diferents, sinó que normalment les sessions d'usuari seran obertes (per exemple, mitjançant un terminal). L'ordre *who* permet llistar aquests usuaris. El *load average* és la càrrega mitjana del sistema en els últims 1, 5 i 15 minuts.

Vegem el *log* de l'arrencada del sistema (ordre *dmesg*), les línies que es generaven en la càrrega del sistema (s'han suprimit diferents línies per claredat):

```
Linux version 2.6.20-1-686 (Debian 2.6.20-2) (waldi@debian.org)
(gcc version 4.1.2 20061115 (prerelease) (Debian 4.1.1-21)) #1 SMP
Sun Apr 15 21:03:57 UTC 2007
BIOS-provided physical RAM map:
 BIOS-e820: 0000000000000000 - 000000000009f800 (usable)
 BIOS-e820: 000000000009f800 - 00000000000a0000 (reserved)
 BIOS-e820: 00000000000ce000 - 00000000000d0000 (reserved)
 BIOS-e820: 00000000000dc000 - 0000000000100000 (reserved)
 BIOS-e820: 0000000000100000 - 000000002f6e0000 (usable)
 BIOS-e820: 000000002f6e0000 - 000000002f6f0000 (ACPI data)
 BIOS-e820: 000000002f6f0000 - 000000002f700000 (ACPI NVS)
 BIOS-e820: 000000002f700000 - 000000002f780000 (usable)
 BIOS-e820: 000000002f780000 - 0000000030000000 (reserved)
 BIOS-e820: 00000000ff800000 - 00000000ffc00000 (reserved)
 BIOS-e820: 00000000fffffc00 - 0000000100000000 (reserved)
0MB HIGHMEM available.
759MB LOWMEM available.
```

Aquestes primeres línies ja ens indiquen diverses dades interessants: la versió del *kernel* Linux és la 2.6.20-1-686, una versió 2.6 revisió 20 a revisió 1 de Debian, i per a màquines 686 (arquitectura Intel 32 bits). També indica que engeguem un sistema Debian, amb aquest *kernel* que va ser compilat amb un compilador GNU *gcc* versió 4.1.2 i la data. A continuació, hi ha un mapa de zones de memòria

utilitzades (reservades) per la BIOS i, després, el total de memòria detectada en la màquina: 759 MB, a les quals caldria sumar el primer 1 MB, total de 760 MB.

```
Kernel command line: BOOT_IMAGE=LinuxNEW ro root=302 lang=es acpi=force
Initializing CPU#0
Console: colour dummy device 80x25
Memory: 766132k/777728k available (1641k kernel code, 10968k reserved, 619k data,
208k init, 0k highmem)
Calibrating delay using timer specific routine.. 5320.63 BogoMIPS (lpj=10641275)
```

Aquí ens refereix com ha estat l'arrencada de la màquina, quina línia d'ordres s'ha passat al *kernel* (es poden passar diferents opcions, per exemple, des del *lilo* o *grub*). I estem arrencant en mode consola de 80 x 25 caràcters (això es pot canviar). Els BogoMIPS són una mesura interna del *kernel* de la velocitat de la CPU. Hi ha arquitectures en què és difícil detectar amb quants MHz funciona la CPU, i per això s'utilitza aquesta mesura de velocitat. Després, ens dóna més dades de la memòria principal, i perquè s'utilitza en aquest moment de l'arrencada.

```
CPU: Trace cache: 12K uops, L1 D cache: 8K
CPU: L2 cache: 512K
CPU: Hyper-Threading is disabled
Intel machine check architecture supported.
Intel machine check reporting enabled on CPU#0.
CPU0: Intel P4/Xeon Extended MCE MSRs (12) available
CPU0: Intel(R) Pentium(R) 4 CPU 2.66GHz stepping 09
```

A més, ens proporciona dades diverses de la CPU, la mida de les *caché* de primer nivell, *caché* interna CPU, L1 dividida en una TraceCache del Pentium4 (o *instruction cache*), la *caché* de dades i la *caché* unificada de segon nivell (L2), tipus de CPU, i velocitat d'aquesta i del bus del sistema.

```
PCI: PCI BIOS revision 2.10 entry at 0xfd994, last bus=3
Setting up standard PCI resources
...
NET: Registered protocol
IP route cache hash table entries: 32768 (order: 5, 131072 bytes)
TCP: Hash tables configured (established 131072 bind 65536)
checking if image is initramfs... it is
Freeing initrd memory: 1270k freed
fb0: VESA VGA frame buffer device
Serial: 8250/16550 driver $Revision: 1.90 $ 4 ports, IRQ sharing enabled
serial8250: ttyS0 at I/O 0x3f8 (irq = 4) is a 16550A
00:09: ttyS0 at I/O 0x3f8 (irq = 4) is a 16550A
RAMDISK driver initialized: 16 RAM disks of 8192K size 1024 blocksize
PNP: PS/2 Controller [PNP0303:KBC0,PNP0f13:MSE0] at 0x60,0x64 irq 1,12
i8042.c: Detected active multiplexing controller, rev 1.1.
serio: i8042 KBD port at 0x60,0x64 irq 1
serio: i8042 AUX0 port at 0x60,0x64 irq 12
serio: i8042 AUX1 port at 0x60,0x64 irq 12
serio: i8042 AUX2 port at 0x60,0x64 irq 12
serio: i8042 AUX3 port at 0x60,0x64 irq 12
mice: PS/2 mouse device common for all mice
```

Continuen les inicialitzacions del *kernel* i dispositius, esmenta la inicialització de protocols de xarxa. Els terminals, i els ports sèrie *ttyS0* (seria el *com1*) i *ttyS01* (el *com2*). Dóna informació dels discos RAM que usem, i detecta dispositius PS2, teclat i ratolí.

```
ICH4: IDE controller at PCI slot 0000:00:1f.1

 ide0: BM-DMA at 0x1860-0x1867, BIOS settings: hda:DMA, hdb:pio
 ide1: BM-DMA at 0x1868-0x186f, BIOS settings: hdc:DMA, hdd:pio
Probing IDE interface ide0...
hda: FUJITSU MHT2030AT, ATA DISK drive
ide0 at 0x1f0-0x1f7,0x3f6 on irq 14
Probing IDE interface ide1...
hdc: SAMSUNG CDRW/DVD SN-324F, ATAPI CD/DVD-ROM drive
ide1 at 0x170-0x177,0x376 on irq 15
SCSI subsystem initialized
libata version 2.00 loaded.
hda: max request size: 128KiB
hda: 58605120 sectors (30005 MB) w/2048KiB Cache, CHS=58140/16/63<6>hda:
hw_config=600b, UDMA(100)
hda: cache flushes supported
 hda: hda1 hda2 hda3
kjournald starting. Commit interval 5 seconds
EXT3-fs: mounted filesystem with ordered data mode.
hdc: ATAPI 24X DVD-ROM CD-R/RW drive, 2048kB Cache, UDMA(33)
Uniform CD-ROM driver Revision: 3.20
Addinf 618492 swap on /dev/hda3.
```

Detecta de dispositius IDE, detecta el xip IDE en el bus PCI, i informa que està controlant dos dispositius: *hda*, i *hdc*, que són, respectivament, un disc dur (Fujitsu), un segon disc dur, un DVD Samsung, i una enregistradora de CD (ja que en aquest cas es tracta d'una unitat *combo*). Indica particions actives. Més endavant, detecta el sistema principal de fitxers del Linux, un *ext3* amb *journal*, que activa i afegeix l'espai de *swap* disponible en una partició.

```
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
input: PC Speaker as /class/input/input1
USB Universal Host Controller Interface driver v3.0
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 2 ports detected
uhci_hcd 0000:00:1d.1: UHCI Host Controller
uhci_hcd 0000:00:1d.1: new USB bus registered, assigned bus number 2
uhci_hcd 0000:00:1d.1: irq 11, io base 0x00001820
usb usb2: configuration #1 chosen from 1 choice
hub 2-0:1.0: USB hub found
hub 2-0:1.0: 2 ports detected
hub 4-0:1.0: USB hub found
hub 4-0:1.0: 6 ports detected
```

Es detecten més dispositius. En aquest cas, l'USB (i els mòduls que corresponen) ha detectat dos dispositius *hub* (amb un total de 8 USB *ports*).

```
parport: PnPBIOS parport detected.
parport0: PC-style at 0x378 (0x778), irq 7, dma 1
[PCSPP,TRISTATE,COMPAT,EPP,ECP,DMA]
input: ImPS/2 Logitech Wheel Mouse as /class/input/input2
ieee1394: Initialized config rom entry 'ip1394'
eepro100.c:v1.09j-t 9/29/99 Donald Becker
Synaptics Touchpad, model: 1, fw: 5.9, id: 0x2e6eb1, caps: 0x944713/0xc0000
input: SynPS/2 Synaptics TouchPad as /class/input/input3

agpgart: Detected an Intel 845G Chipset
agpgart: Detected 8060K stolen Memory
agpgart: AGP aperture is 128M
eth0: OEM i82557/i82558 10/100 Ethernet, 00:00:F0:84:D3:A9, IRQ 11.
 Board assembly 000000-000, Physical connectors present: RJ45
e100: Intel(R) PRO/100 Network Driver, 3.5.17-k2-NAPI
usbcore: registered new interface driver usbkbd
```

```
Initializing USB Mass Storage driver...
usbcore: registered new interface driver usb-storage
USB Mass Storage support registered.

lp0: using parport0 (interrupt-driven).
ppdev: user-space parallel port driver
```

I, finalment, es detecta la resta de dispositius: port paral·lel, model de ratolí, port *firewire* (*ieee1394*) targeta de xarxa (Intel), un plafó tàctil, la targeta de vídeo AGP (i845). Més dades de la targeta de xarxa, una Intel pro 100, registre d'USB com *mass storage* (indica un dispositiu d'emmagatzematge per USB, com un disc extern), i detecció del port paral·lel.

Tota aquesta informació que hem vist mitjançant l'ordre *dmesg*, també, la podem trobar abocada en el *log* principal del sistema */var/log/messages*. En aquest *log*, entre d'altres, trobarem els missatges del *kernel* i dels dimonis, i errors de xarxa o dispositius, els quals comuniquen els seus missatges a un dimoni especial anomenat *syslogd*, que és l'encarregat d'escriure els missatges en aquest fitxer. Si hem engegat la màquina recentment, veurem que les últimes línies contenen exactament la mateixa informació que l'ordre *dmesg*. Per exemple, si ens quedem amb la part final del fitxer (sol ser molt gran):

```
# tail 200 /var/log/messages
```

veiem les línies d'abans, i també algunes informacions més, com per exemple:

```
shutdown[13325]: shutting down for system reboot
kernel: usb 4-1: USB disconnect, address 3
kernel: nfsd: last server has exited
kernel: nfsd: unexporting all filesystems
kernel: Kernel logging (proc) stopped.
kernel: Kernel log daemon terminating.

exiting on signal 15
syslogd 1.4.1#20: restart.

kernel: klogd 1.4.1#20, log source = /proc/kmsg started.
Linux version 2.6.20-1-686 (Debian 2.6.20-2) (waldi@debian.org) (gcc version 4.1.2
20061115 (prerelease) (Debian 4.1.1-21)) #1 SMP Sun Apr 15 21:03:57 UTC 2007
kernel: BIOS-provided physical RAM map:
```

La primera part correspon a la parada anterior del sistema, i ens diu que el *kernel* ha deixat de col·locar informació en */proc*, s'està parant el sistema... Al principi de l'arrencada nova, s'activa el dimoni Syslogd, que genera el *log*, i comença la càrrega del sistema, que ens diu que el *kernel* començarà a escriure informació en el seu sistema, */proc*; veiem les primeres línies de *dmesg* de menció de la versió que s'està carregant de *kernel* i, després, trobarem el que hem vist amb *dmesg*.

En aquest punt, una altra ordre útil per a saber com s'ha produït la càrrega és *ismod*, que ens permetrà saber quins mòduls s'han carregat en el *kernel* (versió resumida):

```
# lsmod
Module Size Used by
nfs 219468      0
nfsd 202192     17
```

```
exportfs 5632      1 nfsd
lockd 58216        3 nfs,nfsd
nfs_acl 3616       2 nfs,nfsd
sunrpc 148380      13 nfs,nfsd,lockd,nfs_acl
ppdev 8740         0
lp 11044                   0
button 7856        0
ac 5220                    0
battery 9924       0
md_mod 71860       1
dm_snapshot 16580          0
dm_mirror 20340            0
dm_mod 52812       2 dm_snapshot,dm_mirror
i810fb 30268       0
vgastate 8512      1 i810fb
eeprom 7184        0
thermal 13928      0
processor 30536    1 thermal
fan 4772                   0
udf 75876                  0
ntfs 205364        0
usb_storage 75552          0
hid 22784                  0
usbkbd 6752        0
eth1394 18468      0
e100 32648         0
eepro100 30096     0
ohci1394 32656     0
ieee1394 89208     2 eth1394,ohci1394
snd_intel8x0 31420         1
snd_ac97_codec 89412       1 snd_intel8x0
ac97_bus 2432      1 snd_ac97_codec
parport_pc 32772           1
snd 48196          6 snd_intel8x0,snd_ac97_codec,snd_pcm,snd_timer
ehci_hcd 29132      0
ide_cd 36672       0
cdrom 32960        1 ide_cd
soundcore 7616     1 snd
psmouse 35208      0
uhci_hcd 22160     0
parport 33672      3 ppdev,lp,parport_pc
intelfb 34596              0
serio_raw 6724     0
pcspkr 3264        0
pci_hotplug 29312          1 shpchp
usbcore 122312     6 dvb_usb,usb_storage,usbkbd,ehci_hcd,uhci_hcd
intel_agp 22748    1
agpgart 30504      5 i810fb,drm,intelfb,intel_agp
ext3 121032        1
jbd 55368                  1 ext3
ide_disk 15744     3
ata_generic 7876           0
ata_piix 15044     0
libata 100052      2 ata_generic,ata_piix
scsi_mod 133100            2 usb_storage,libata
generic 4932       0 [permanent]
piix 9540                  0 [permanent]
ide_core 114728            5 usb_storage,ide_cd,ide_disk,generic,piix
```

Veiem que, bàsicament, tenim els controladors per al maquinari que hem detectat, i altres relacionats o necessaris per dependències.

Ja tenim, doncs, una idea de com s'han carregat el *kernel* i els seus mòduls. En aquest procés, pot ser que ja hàgim vist algun error, si hi ha maquinari mal configurat o mòduls del *kernel* mal compilats (no eren per a la versió del *kernel* adequada), inexistents, etc.

El pas següent serà l'observació dels processos en el sistema amb l'ordre *ps* (*process status*), per exemple (només s'han tret els processos de sistema, no els dels usuaris):

```
#ps -ef
UID PID PPID C STIME TTY TIME CMD
```

Dóna informació dels processos, de l'UID usuari que ha llançat el procés (o amb quin identificador s'ha llançat), i PID, codi del procés assignat pel sistema –són consecutius a mesura que es llancen els processos, el primer sempre és el 0, que correspon al procés d'*init*. PPID és l'*id* del procés pare de l'actual. STIME és el temps en què va ser engegat el procés, TTY és el terminal assignat al procés (si en té algun), i CMD és la línia d'ordre amb què va ser llançat.

```
root 1 0 0 14:52 ? 00:00:00 init [2]
root 3 1 0 14:52 ? 00:00:00 [ksoftirqd/0]
root 143 6 0 14:52 ? 00:00:00 [bdflush]
root 145 6 0 14:52 ? 00:00:00 [kswapd0]
root 357 6 0 14:52 ? 00:00:01 [kjournald]
root 477 1 0 14:52 ? 00:00:00 udevd --daemon
root 719 6 0 14:52 ? 00:00:00 [khubd]
```

Diversos dimonis de sistema, com *kswapd daemon*, que controlen l'intercanvi de pàgines amb memòria virtual. Maneig de *buffers* del sistema (*bdflush*). Maneig de *journal* de sistema de fitxers (*kjournald*), maneig d'USB (*khubd*). O el dimoni d'*udev* que controla la connexió en calent de dispositius. Els dimonis no sempre s'acostumen a identificar per una *d* final, i si porten una *k* inicial, normalment són fils (*threads*) interns del *kernel*.

```
root    1567 1 0 14:52 ? 00:00:00 dhclient -e -pf ...
root    1653 1 0 14:52 ? 00:00:00 /sbin/portmap
root    1829 1 0 14:52 ? 00:00:00 /sbin/syslogd
root    1839 1 0 14:52 ? 00:00:00 /sbin/klogd -x
root    1983 1 0 14:52 ? 00:00:09 /usr/sbin/cupsd
root    2178 1 0 14:53 ? 00:00:00 /usr/sbin/inetd
```

Tenim *dhclient*. Això indica que aquesta màquina és clienta d'un servidor DHCP per a obtenir el seu IP. Syslogd, un dimoni que envia missatges al *log*. El dimoni de *cups*, com hem vist, relacionat amb el sistema d'impressió. I *inetd*, que, com veurem en la part de xarxes, és una mena de "superservidor" o intermediari d'altres dimonis relacionats amb serveis de xarxa.

```
root    2154 1 0 14:53 ?          00:00:00 /usr/sbin/rpc.mountd
root    2241 1 0 14:53 ?          00:00:00 /usr/sbin/sshd
root    2257 1 0 14:53 ?          00:00:00 /usr/bin/xfs -daemon
root          2573 1 0 14:53 ?          00:00:00 /usr/sbin/atd
root    2580 1 0 14:53 ?          00:00:00 /usr/sbin/cron
root    2675 1 0 14:53 ?          00:00:00 /usr/sbin/apache
www-data      2684 2675 0 14:53 ?       00:00:00 /usr/sbin/apache
www-data      2685 2675 0 14:53 ? 00:00:00 /usr/sbin/apache
```

També hi ha el *sshd*, servidor d'accés remot segur (una versió millorada que permet serveis compatibles amb Telnet i *ftp*). *xfs* és el servidor de fonts (tipus de lletra) de l'X Window. Les ordres *atd* i *cron* serveixen per a manejar tasques

programades en un moment determinat. Apache és el servidor web que pot tenir diversos fils actius (*threads*) per a atendre diferents peticions.

```
root 2499 2493 0 14:53 ?          00:00:00 /usr/sbin/gdm
root 2502 2499 4 14:53 tty7       00:09:18 /usr/bin/X :0 -dpi 96 ...
root 2848 1 0 14:53 tty2          00:00:00 /sbin/getty 38400 tty2
root 2849 1 0 14:53 tty3          00:00:00 /sbin/getty 38400 tty3
root 3941 2847 0 14:57 tty1       00:00:00 -bash
root 16453 12970 0 18:10 pts/2    00:00:00 ps -ef
```

Gdm és la connexió gràfica del sistema d'escriptori Gnome (l'entrada que ens demana el *login* i contrasenya), i els processos *getty* són els que gestionen els terminals virtuals de text, els que podem veure amb les tecles Alt+Fx (o Ctrl+Alt+Fx si estem en mode gràfic). X és el procés de servidor gràfic de l'X Window System, imprescindible perquè s'executi qualsevol entorn d'escriptori per sobre d'ell. Un intèrpret d'odres obert (*bash*) i, finalment el procés que hem generat en demanar aquest *ps* des de la línia d'ordres.

L'ordre *ps* té moltes opcions de línia d'ordres per a ajustar la informació que volem de cada procés, tant si és el temps que fa que s'executa com el percentatge de CPU usat, memòria utilitzada, etc. (vegeu *man* de *ps)*. Una altra ordre molt interessant és *top,* que fa el mateix que *ps* però de manera dinàmica, és a dir, s'actualitza amb cert interval, ens permet classificar els processos per ús de CPU, de memòria, i també ens dóna informació de l'estat de la memòria global.

Altres ordres útils per a ús de recursos són *free* i *vmstat*, que ens donen informació sobre la memòria utilitzada i el sistema de memòria virtual:

Nota

Vegeu *man* de les ordres per interpretar les sortides.

```
# free
          total used free shared buffers cached
Mem:   767736 745232 22504 0 89564 457612
-/+ buffers/cache: 198056 569680
Swap: 618492 1732 616760

# vmstat
procs -----------memory---------- ---swap-- -----io-- --system-- ----cpu----
 r b swpd free buff cache si so bi bo in cs us sy id wa
 1 0 1732 22444 89584 457640 0 0 68 137 291 418 7 1 85 7
```

En l'ordre *free*, també es pot veure la mida del *swap* present, aproximadament de 600 MB, que de moment no s'utilitzen intensament perquè hi ha prou espai de memòria física: encara queden prop de 22 MB lliures (la qual cosa indica un ús intens de la memòria física, i que el *swap* s'usarà pròximament). L'espai de memòria i el *swap* (a partir dels *kernels* 2.4) són additius per a compondre el total de memòria del sistema i, en aquest cas, fan un total d'1,4 GB de memòria disponible. Això pot semblar molt, però dependrà de les aplicacions que s'executin.

Activitats

1. L'espai de *swap* permet complementar la memòria física per disposar de més memòria virtual. Depenent de les mides de memòria física i swap, es pot arribar a esgotar la memòria? Ho podem solucionar d'una altra manera, que no sigui afegint més memòria física?

2. Suposem que tenim un sistema amb dues particions Linux, una / i l'altra de *swap*. Com solucionaríem el cas en què els comptes dels usuaris esgotessin l'espai de disc? I en el cas de tenir una partició */home* aïllada que s'estigués esgotant, com ho solucionaríem?

3. Instal·leu el sistema d'impressió CUPS, definiu la nostra impressora perquè funcioni amb CUPS i proveu l'administració per interfície web. Tal com està el sistema, seria recomanable modificar d'alguna manera la configuració que porta per defecte CUPS? Per què?

4. Examineu la configuració per defecte que porti el sistema GNU/Linux de treballs no interactius per cron. Quins treballs i quan s'estan realitzant? Alguna idea per a nous treballs que calgui afegir?

5. Reproduïu l'anàlisi del taller (més els altres apartats de la unitat) sobre la màquina que es tingui, es veuen en el sistema alguns errors o situacions anòmals? En aquest cas, com els corregim?

Altres fonts de referència i informació

[Wm02] [Fri02] [Smi02] Llibres d'administració del GNU/Linux i UNIX, en què es comenten àmpliament aspectes d'administració local i gestió de sistemes d'impressió.

[Gt] Es pot trobar informació actualitzada dels sistemes d'impressió i la seva configuració, i detalls d'algunes impressores. Per a detalls concrets de models d'impressora i controladors, ens podem adreçar a *http://www.linuxprinting.org/*.

[Hin][Koe] Trobem informació sobre els diferents sistemes de fitxers disponibles i els esquemes de creació de particions per a la instal·lació del sistema.