Administració de processos. Administració remota. Administració de serveis d'impressió

Joan Muñoz Pastor i Raúl Velaz May
Adaptació de continguts: Joan Muñoz

Índex

Introducció

L'administració del sistema operatiu comprèn un conjunt de tasques i responsabilitats de gran importància pel bon funcionament i productivitat de tot l'engranatge informàtic d'una empresa. Entre aquestes responsabilitats destaquen la necessitat de conèixer el funcionament dels processos, l'administració remota del sistema o la gestió dels serveis d'impressió.

Així doncs en l'apartat "Administració de processos del sistema" ens adonarem que els sistemes operatius actuals són d'una complexitat remarcable i que poden treballar amb múltiples usuaris i tasques alhora gràcies a la gestió dels processos. El processos són tasques independents i dinàmiques que van canviant constantment, així que cal conèixer el seu funcionament, la manera com neixen, canvien d'estat i moren. Hem de saber com comunicar-nos amb ells, enviar i rebre senyals o canviar la seva prioritat d'execució en funció de les necessitats del sistema i del mateix usuari. Aquest coneixement ens farà entendre les seqüències d'arrencada i aturada, així com els nivells d'execució i els serveis que hi estan associats, que ens garanteixen el funcionament del sistema.

En l'apartat "Serveis d'accés i administració remota" veurem com la generalització de les comunicacions entre ordinadors, tant en xarxa local com mitjançant Internet, ha introduït canvis dràstics en l'administració, permetent que pugui fer-se de forma remota. Tant si es basa en la línia d'ordres, en una aplicació gràfica o si usa del navegador com a interfície gràfica, l'administració remota facilita a l'administrador la tasca de configuració i gestió del sistema informàtic, tenint sempre en compte la capacitat i ample de banda de la comunicació i la necessària atenció a la seguretat i a la privacitat.

En l'apartat "Administració de servidors d'impressió" abordarem l'administració dels serveis d'impressió, una de les tasques tradicionals que sempre han tingut espai en l'administració de sistemes, que consisteix en la conversió del resultat del procés informàtic a un suport de paper definitiu i directament interpretable pel usuari. En aquest sentit, l'evolució constant de les necessitats dels usuaris i de la tecnologia ens ha portat a sistemes que garanteixen la gestió de múltiples treballs en un entorn multisusuari, la comunicació i transmissió en xarxa amb els protocols adequats, l'administració de cues d'impressió i el suport a una tecnologia d'impressió gràfica dels perifèrics cada vegada més sofisticada.

Per treballar els continguts d'aquesta unitat, és convenient llegir amb atenció la part teòrica, dur a terme les activitats pràctiques proposades i fer els exercicis d'autoavaluació del material web.

Resultats d'aprenentatge

1. Administració de processos del sistema descrivint i aplicant criteris de seguretat i eficiència.
    - Descriu els conceptes de procés del sistema, tipus, estats i cicle de vida.
    - Utilitza interrupcions i excepcions per descriure els esdeveniments interns del processador.
    - Diferencia entre procés, fil i treball.
    - Realitza tasques de creació, manipulació i acabament de processos.
    - Utilitza el sistema de fitxers com a mitjà lògic per al registre i identificació dels processos del sistema.
    - Utilitza eines gràfiques i ordres per al control i seguiment dels processos del sistema.
    - Comprova la seqüència d'arrencada del sistema, els processos implicats i la relació entre ells.
    - Pren mesures de seguretat davant l'aparició de processos no identificats.
    - Documenta els processos habituals del sistema, la seva funció i relació entre ells.
2. Administració remota del sistema operatiu en xarxa valorant la seva importància i aplicant criteris de seguretat.
    - Descriu mètodes d'accés i administració remota de sistemes.
    - Diferencia entre els serveis orientats a sessió i els no orientats a sessió.
    - Utilitza eines d'administració remota subministrades pel propi sistema operatiu.
    - Instal·la serveis d'accés i administració remota.
    - Utilitza ordres i eines gràfiques per gestionar els serveis d'accés i administració remota.
    - Crea comptes d'usuari per a l'accés remot.
    - Realitza proves d'accés i administració remota entre sistemes heterogenis.
    - Utilitza mecanismes d'encriptació de la informació transferida.
    - Documenta els processos i serveis del sistema administrats de forma remota.
3. Administració servidors d'impressió descrivint les seves funcions i integrant-los en una xarxa.
    - Descriu la funcionalitat dels sistemes i servidors d'impressió.

- Identifica els ports i els protocols utilitzats.
- Utilitza les eines per a la gestió d'impressores integrades en el sistema operatiu.
- Instal·la i configura un servidor d'impressió en entorn web.
- Crea i classifica impressores lògiques.
- Crea grups d'impressió.
- Gestiona impressores i cues de treballs mitjançant ordres i eines gràfiques.
- Comparteix impressores en xarxa entre sistemes operatius diferents.
- Documenta la configuració del servidor d'impressió i de les impressores creades.

1. Administració de processos del sistema

Tot programari executable, inclòs el mateix sistema operatiu, s'acaba constituint en una sèrie d'unitats anomenades *processos* que s'organitzen, ordenen i consumeixen recursos, per acabar sent executats pel processador. L'administració d'aquests processos i dels seus recursos associats constitueix una de les tasques bàsiques i primordials del nucli del sistema operatiu que, entre d'altres, comprendrà funcions com ara:

- La creació i destrucció de processos
- El despatx (*dispatcher*), és a dir, l'assignació de processos a execució en el processador
- Els canvis d'estat que anomenem transicions
- La suspensió i represa de processos
- La sincronització de processos
- La comunicació entre processos
- La manipulació de blocs de control de procés (PCB)

  Moltes d'aquestes activitats estan només en mans del sistema operatiu i seran transparents per a l'usuari, però cal conèixer bé els mecanismes de l'administració de processos, ja que l'usuari pot intervenir en aspectes crucials com ara la configuració de l'arrencada del sistema, l'assignació de prioritats o la finalitzant, suspensió i represa d'aquests processos.

1.1 Conceptes bàsics sobre processos

Un procés és la instància d'un programa en execució que necessita per realitzar la seva tasca recursos com ara:

- Temps de processador
- Memòria
- Arxius
- Dispositius d'entrada/sortida

Aquests recursos es poden assignar en el moment de la creació del procés o bé durant la seva execució.

Des de aquesta perspectiva, un procés es pot considerar unes línies de codi carregades a memòria i un context associat constituït per l'activitat del processador en un moment determinat, és a dir:

- El valor del comptador del programa
- El contingut dels registres del processador

i per una sèrie de dades emmagatzemades a la memòria:

- Variables globals i memòria dinàmica
- Dades temporals a la pila (*stack*) com adreces de retorn i variables locals

1.1.1 Conceptes d'aplicació, procés i servei

Es considera generalment una **aplicació** informàtica com un programa dissenyat per interaccionar amb l'usuari, per tant s'executa en primer pla (*foreground*) amb una determinada interfície d'usuari. Són exemples típics d'aplicacions els programes d'ofimàtica, els navegadors, els reproductors multimèdia, etc.

En canvi, un **servei** és un terme que correspon al concepte i funcionalitat del **dimoni** (daemon) en la terminologia de Linux. Anomenem *servei* a un programa normalment associat al sistema operatiu, encara que també es pot engegar de forma manual, i que treballa en segon pla (*background*) sense interfície d'usuari, donant suport a altres programes. Els serveis proporcionen prestacions com serveis de pàgines web, registre d'esdeveniments, serveis d'impressió, criptografia...

El concepte de **procés,** en canvi, té més a veure amb el funcionament intern del sistema operatiu i consisteix en una sèrie d'instruccions allotjades a la memòria que, mitjançant cues i un mecanisme de planificació, accedeixen a la CPU per executar-se. Així, doncs, quan s'engega qualsevol aplicació o servei genera un o més processos en el sistema.

1.1.2 Tipus de processos i classificació

Podem fer algunes classificacions dels processos en funció del seu propietari, el seu comportament en concurrència, la forma d'execució i ubicació a la memòria o els recursos del sistema que comparteixin.

Segons del propietari del procés tenim:

- **Processosde sistema**: Associats al funcionament del nucli del sistema o dels diferents serveis en funcionament (dimonis). Molts d'aquests processos estan associats a usuaris virtuals del sistema (lp, www, mail, etc.).

- **Processos de superusuari**: Associats al compte de l'administrador arrel (*root*).
- **Processos d'usuari**: Associats a l'execució d'aplicacions d'un usuari determinat.

També podem classificar els processos que s'executen concurrentment com a:

- **Independents**: No afecten ni són afectats per altres processos.
- **Cooperants**: Poden compartir dades i per tant afectar i ser afectats per altres processos.

Segons la forma d'execució i ubicació a la memòria poden ser:

- **Residents**: Els processos residents són sempre a la memòria mentre dura l'execució.
- **Intercanviables**: Poden ser portats de la memòria principal al disc dur mentre estan bloquejats. Així, la memòria alliberada pot ser utilitzada per altres processos que la necessitin.

Finalment segons els recursos que comparteixen trobem processos:

- **Pesants**: Els processos pesants són independents i tenen tots els seus propis recursos.
- **Lleugers**: També anomenats fils d'execució o *threads*, comparteixen entre si espai de memòria i recursos d'entrada i sortida.

1.1.3 Planificació de processos

Tal com hem avançat, una de les obligacions d'un sistema operatiu com a gestor de processos és la planificació de processos: l'execució de múltiples processos optimitzant l'ús del processador.

Així, doncs, les polítiques de planificació de processos sorgeixen com a necessitat d'acomplir dos dels principals objectius funcionals dels sistema operatiu: la multiprogramació i el temps compartit:

- **Multiprogramació**: Té com a objectiu maximitzar l'aprofitament de la CPU tenint sempre en execució algun procés en tot moment.
- **Temps compartit**: Atès que una CPU només pot realitzar un procés alhora es tracta d'establir un sistema de commutació de la CPU entre els processos amb tal freqüència que l'usuari pugui interactuar-hi i tingui la impressió que s'estan executant en paral·lel.

La planificació de processos es desenvolupa a tres nivells: llarg, mig i curt termini.

Planificació a llarg termini

El planificador a llarg termini és l'encarregat de crear els processos determinant quins treballs s'admeten en el sistema per al seu processament i carregant-los en la memòria disponible. Els sistemes operatius de temps compartit gairebé no tenen algorisme de planificació a llarg termini i es limiten a crear i posar en estat "preparat" qualsevol procés nou. En canvi, té més sentit la planificació a llarg termini en sistemes que admeten processament en lots, ja que és convenient compensar tasques que demanin més temps de processador amb tasques que demanin més operacions d'entrada i sortida, per equilibrar així els recursos del sistema.

Planificació a mig termini

La planificació a mig termini regula el grau de multiprogramació fixat en un principi pel planificador a llarg termini. Si el sistema no té prou recursos, en especial de memòria, per atendre tots els processos en marxa, es poden passar alguns d'ells a l'estat de suspesos i alliberar així recursos i memòria interna.

Grau de multiprogramació

Correspon al nombre de processos actius en un moment determinat, que estan carregats a la memòria principal del sistema.

Per fer això és fa servir la tècnica de l'intercanvi (*swapping*, en anglès), que s'encarrega de suspendre processos enviant-los a memòria secundària, habitualment el disc dur (*swap out*), i reactivant-les posteriorment, tornant-los a carregar de la memòria secundària a la memòria interna (*swap in*).

Planificació a curt termini

El planificador a curt termini o *dispatcher* té la tasca essencial de decidir quin procés passa a execució d'entre els que estan a la cua dels processos preparats (*ready*). Per fer aquesta tasca, els planificadors a curt termini implementen una sèrie d'algorismes que tenen com a objectiu optimitzar l'eficiència, la productivitat i el temps de resposta de la CPU.

Aquests algorismes de planificació es poden classificar en algorismes apropiatius i no apropiatius, i també hem de considerar altres tècniques combinades, com ara les cues multinivell.

Algorismes no apropiatius

El sistema operatiu no expulsa mai el procés de la CPU fins que aquest acaba l'execució o en surt voluntàriament, per exemple, quan el procés passa a bloquejat a l'inici d'una operació d'entrada/sortida. Exemples d'aquests tipus d'algorisme poden ser:

Cues FIFO i LIFO

En una cua FIFO (*first input first output*) els elements de la cua surten en el mateix ordre que han entrat, mentre que en una cua LIFO (*last input first output*), també anomenada pila o stack, és el darrer element arribat el primer que surt de la cua.

- **FCFS** (*first come first served*). És una cua FIFO que dóna preferència segons l'ordre d'arribada a la cua.
- **SJF** (*shortest job first*). Algorisme no apropiatiu on la prioritat d'accés a l'execució depèn del temps de ràfega de CPU necessari. Així, es

beneficien aquells processos més curts d'executar i es maximitza el nombre de processos executats per unitat de temps.

Algorismes apropiatius

El sistema operatiu pot decidir expulsar un procés de la CPU i executar-ne un altre. Alguns exemples d'aquest tipus d'algorisme:

- **Round Robin**: Algorisme de roda en el qual el sistema operatiu assigna un temps determinat a cada procés. Aquest temps, anomenat **quàntum**, pot ser constant o calcular-se dinàmicament. El procés abandona la CPU quan esgota el seu quàntum. L'elecció del valor del quàntum és molt important, ja que un quàntum molt petit produeix massa canvis de context.

- **SRT** (*shortest remaining time*): Algorisme depenent del temps de ràfega com l'SJF, però en aquest cas és apropiatiu. És a dir, si arriba un nou procés a la cua de preparats i el seu temps d'execució és menor que el que li resta al que s'està executant en aquell moment, es pot apropiar de la CPU i expulsar-lo.

Cues multinivell

Consisteixen en diferents cues de processos en estat de preparats. Cada cua té la seva prioritat. Per accedir a l'execució en el processador s'escull el procés de la cua més prioritària que no estigui buida. Dins de cada cua es pot aplicar una política i un algorisme de planificació diferent.

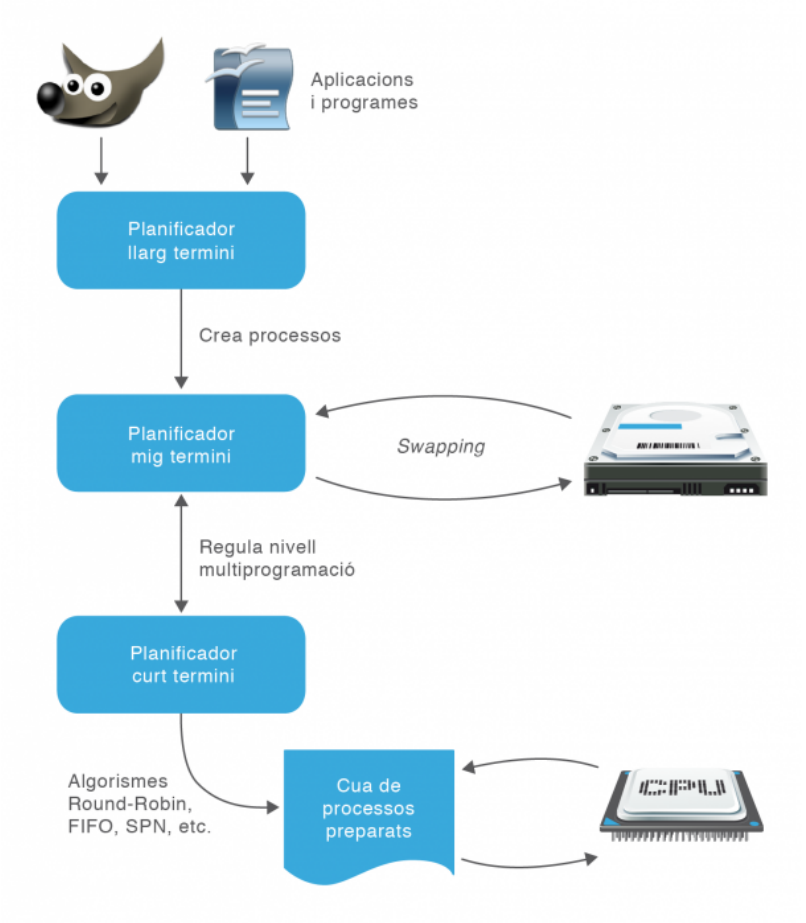
Cues multinivell realimentades

És el mateix cas de les cues multinivell, però els processos poden avançar i retrocedir per les diferents cues de prioritats. Per exemple, un procés de prioritat alta que exhaureixi el seu quàntum podria passar a una cua de preparats de menys prioritat.

El sistema operatiu Unix/Linux fa servir **cues multinivell realimentades** utilitzen l'algorisme Round Robin a cada cua. Cada segon es recalculen les prioritats dels processos en funció d'unes prioritats dinàmiques i d'unes de fixes, definides d'una banda pel sistema, segons el tipus de procés, i de l'altra pel mateix usuari mitjançant la configuració de la prioritat *nice*.

A la figura 1.1 veiem esquemàticament la intervenció dels diferents nivells de planificació en la creació de processos, la regulació del nivell de multiprogramació mitjançant l'intercanvi amb el disc dur i finalment l'assignació òptima dels processos preparats per ser executats en el processador.

FIGURA 1.1. Esquema del sistema de planificació de processos



1.1.4 Estats i transicions dels processos

Una característica fonamental d'un procés és l'estat en el qual es troba, que ve definit per la seva incorporació dins de les diferents cues que organitzen la gestió del processador. Una primera aproximació ens dóna fins a cinc estats diferents possibles per a un procés, encara que la incorporació del sistema d'intercanvi (*swapping*) ens aportarà dos possibles estats addicionals.

Estats d'un procés

Procés zombi

Podem considerar-ho com una variant de l'estat finalitzat. És un procés que ha finalitzat i ja no fa sevir la CPU, però que ha de mantenir en memòria certa informació d'utilitat per al seu procés pare. A Linux, els processos finalitzats passen directament a estat zombi fins que el procés pare se n'assabenta i els esborra definitivament.

Els cinc estats bàsics d'un procés són els següents:

- **Nou** (*new*): El procés s'està creant.
- **Preparat** (*ready*): El procés està a la cua, preparat per a l'assignació d'un processador.
- **En execució** (*running*): Les instruccions del procés estan sent executades pel processador.
- **Bloquejat** (*waiting*): El procés està parat en espera d'un esdeveniment

que el desbloquegi (finalització d'una entrada/sortida, recepció d'un senyal, etc.).

- **Finalitzat** (*terminated*): El procés està finalitzat i s'alliberen els recursos que feia servir.

Només un procés pot estar en estat d'execució en un processador i en un moment determinats, però molts processos poden estar preparats o en espera bloquejats.

Transicions d'un procés

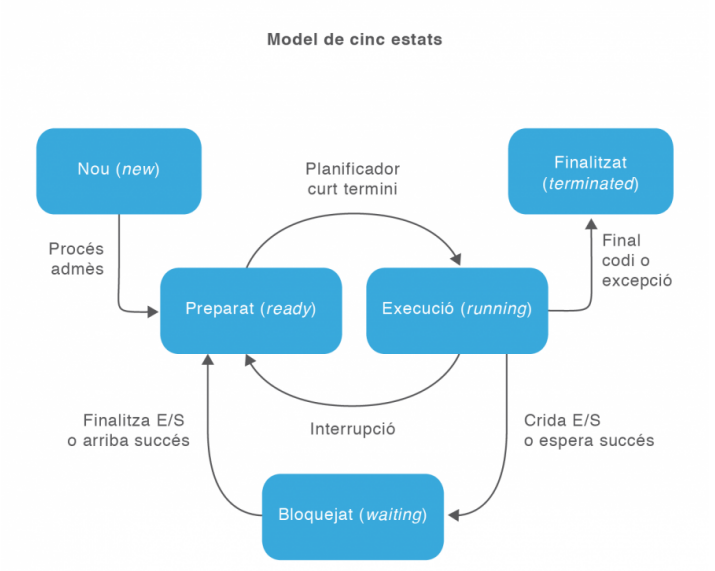
Un procés és un element essencialment dinàmic que va canviant entre els diferents estats mitjançant transicions. Partint d'un model de cinc estats com els que hem definit, les possibles transicions entre estats són les següents:

- **Nou** → **preparat**: el sistema està preparat per admetre un nou procés perquè disposa de recursos suficients.
- **Preparat** → **execució**: el planificador a curt termini, tot seguint diferents algorismes de planificació (Round Robin, SFJ, SPN, FCFS, etc.) decideix quin dels processos en cua de preparats passa a execució.
- **Execució** → **preparat**: si l'algorisme de planificació és apropiatiu, el procés en execució pot tornar a la cua de preparats si esgota el seu temps prefixat d'execució (quàntum) o si rep una interrupció per l'arribada d'un procés amb més prioritat.
- **Execució** → **bloquejat**: si un procés en execució necessita una operació d'entrada/sortida o algun altre esdeveniment per continuar, passa a l'estat de bloquejat i allibera la CPU.
- **Bloquejat** → **preparat**: si en un procés bloquejat arriba el succés o finalitza l'operació d'entrada/sortida que esperava, llavors retorna a la cua de processos preparats.
- **Execució** → **finalitzat**: si un procés finalitza l'execució de les seves línies de codi o bé rep una excepció per algun tipus d'error greu, passa a procés finalitzat i allibera els recursos emprats.

Model de cinc estats

Aquest model descrit de cinc estats i les seves transicions descrit anteriorment es pot veure gràficament en el diagrama d'estats i transicions de la figura 1.2.

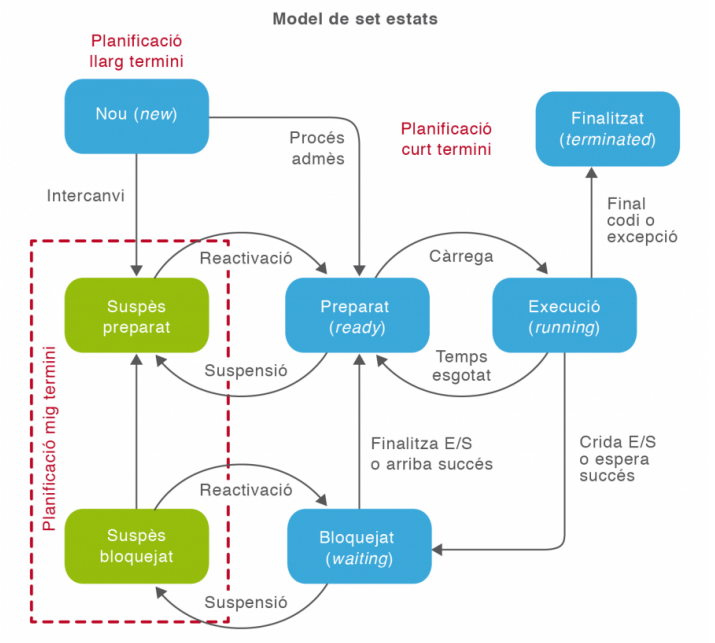
FIGURA 1.2. Model de cinc estats dels processos amb les seves transicions



Model de set estats

Si al model de cinc estats li afegim la planificació a mig termini que gestiona l'intercanvi de processos a la memòria secundària, podem considerar dos nous estats:

FIGURA 1.3. Model de set estats dels processos amb la inclusió dels estats de suspensió



- **Suspès preparat**: Aquells processos que ja estan preparats per ser executats però que, per falta de recursos del sistema, el planificador a mig termini ha decidit passar temporalment a la memòria secundària (disc dur).
- **Suspès bloquejat**: Processos que estan en espera d'un esdeveniment o de la

finalització d'una operació d'entrada/sortida i que són passats a la memòria secundària per alliberar recursos.

Així, doncs, ens queda un diagrama d'estats i transicions com el de la figura 1.3.

1.2 Control i estructura dels processos

Controlar i fer operatiu tot aquest sistema de gestió dels processos requereix la definició d'una sèrie d'estructures de dades que continguin tota la informació associada a cada procés, així com les llistes i taules per controlar les diferents cues implicades en la planificació. Dues d'aquestes estructures bàsiques d'informació són el **bloc de control de procés** (PCB) i el **bloc de control de sistema**(SCB).

D'altra banda, cal comprendre el mecanisme que permet al processador canviar el procés que està executant per un altre. Aquest mecanisme s'anomena **canvi de context**. La problemàtica de la penalització en temps de CPU que suposa el canvi de context ens portarà a l'aparició de tècniques més òptimes, com els sistemes multifil basats en fils d'execució o *threads*.

1.2.1 Bloc de control de procés

Cada procés es representa al sistema operatiu com una estructura de dades anomenada **bloc de control de processos**(en anglès *process control block* o PCB) que descriu el procés i conté una sèrie de camps d'informació:

- **Identificador** del procés.
- **Estat**: estat actual del procés (preparat, bloquejat, en execució, etc.)
- **Comptador del programa**: indica l'adreça de la següent instrucció d'aquest procés que s'executarà.
- **Registres de la CPU**: acumuladors, punters de pila (*stack*) i registres generals.
- **Informació de planificació**: com la prioritat del procés i el valor del quàntum.
- **Informació de la gestió de la memòria**: definició de la finestra de memòria amb el registre base i límit, taula de pàgines o segments.
- **Informació de comptabilitat**: temps de CPU, temps consumit, etc.
- **Informació de l'estat de les entrades/sortides**: dispositius d'entrada/sortida assignats al procés, llista d'arxius oberts, etc.

- **Punter a la memòria**: apunta al node següent de la llista enllaçada de PCB.

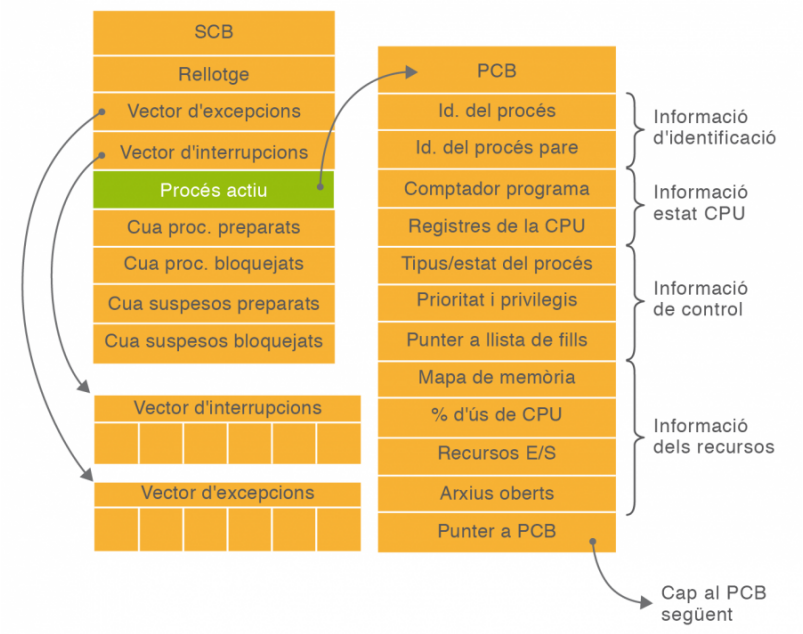
Per gestionar els processos el sistema operatiu ha de llegir, tractar i manipular aquestes estructures de dades.

1.2.2 Bloc de control del sistema

Els sistemes operatius disposen també d'una estructura anomenada **bloc de control del sistema** (de l'anglès *system control block* o SCB) per controlar i reunir informació de totes les estructures de dades dels processos, les interrupcions i les excepcions. El bloc de control de sistema conté habitualment la informació següent:

- Un punter cap al descriptor (PCB) del procés que està fent ús del processador (procés actiu)
- Un punter a una cua de descriptors dels processos preparats (*ready*)
- Un punter a una cua de descriptors dels processos bloquejats (*waiting*)
- Un punter a una cua de descriptors dels processos suspesos preparats
- Un punter a una cua de descriptors dels processos suspesos bloquejats
- Punters als vectors d'interrupcions i d'excepcions que són referències a les rutines necessàries per atendre aquests esdeveniments

FIGURA 1.4. Estructures de control de processos: bloc de control de procés i de sistema



Una **interrupció**, sigui de programari (crides al sistema) o de maquinari (rellotge, dispositius d'entrada/sortida, reinicialització, etc.), és la presència d'un esdeveniment que obliga el sistema operatiu a prendre el control del processador per analitzar i tractar la situació mitjançant rutines específiques.

Una **excepció** és un tipus d'interrupció generada al sistema per un error (divisió per zero, desbordament de memòria...).

Vector d'interrupcions/excepcions

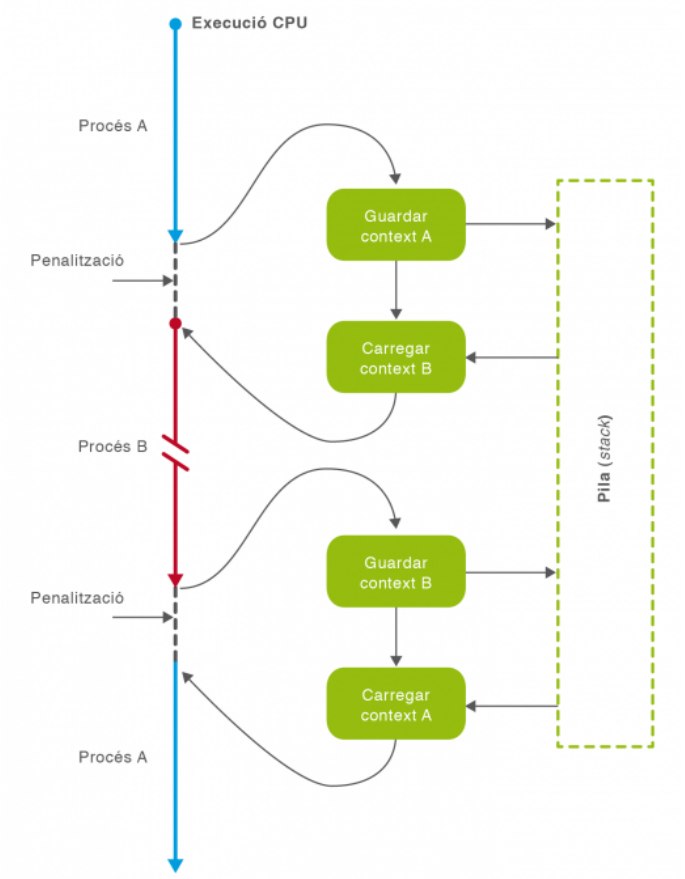
És una taula de les adreces de memòria on estan situades les rutines a executar pel tractament de la interrupció/excepció generada.

A la figura 1.4 es mostra un esquema d'aquestes estructures de dades.

1.2.3 Canvi de context

Quan un procés en execució ha d'abandonar la CPU per algun esdeveniment (interrupció, exhauriment del quàntum, operació d'entrada/sortida, etc.) i s'ha de començar o reprendre un altre procés és necessari fer un **canvi de context**. Per fer això cal guardar el context del procés que surt, és a dir, el seu estat, registres, comptador de programa, etc., que està representat pel seu PCB (bloc de control de procés) i recuperar el context del procés que entra en execució. Vegeu la figura 1.5.

Figura 1.5. Accions a efectuar en un canvi de context



Els canvis de context permeten la commutació de la CPU entre diferents processos, però també suposen una penalització pel temps necessari per efectuar-los. Per intentar reduir aquesta penalització (*overhead*) es fan servir noves estructures com els fils d'execució (*threads*).

1.2.4 Fils d'execució

La definició de procés inclou dos conceptes separats i potencialment independents: un de relatiu a la propietat dels recursos, i un altre que fa referència a la assignació de CPU per a l'execució del codi.

- **Recursos del procés**: A un procés se l'assigna sempre un espai de memòria i a vegades també altres recursos com dispositius d'entrada/sortida i arxius.
- **Assignació de CPU**: Un procés és un flux d'execució, també anomenat traça, quan aquest és assignat a un processador per a l'execució del codi.

  A la majoria de sistemes operatius aquests dos requeriments són l'essència del procés, però poden ser tractats de manera independent. Aquesta distinció ha portat, en els sistemes operatius actuals, a desenvolupar el concepte de **fil d'execució**, també anomenat procés lleuger o *thread*, en anglès. Tenint en compte aquesta divisió de característiques, podem definir fil d'execució com la unitat d'assignació de CPU.

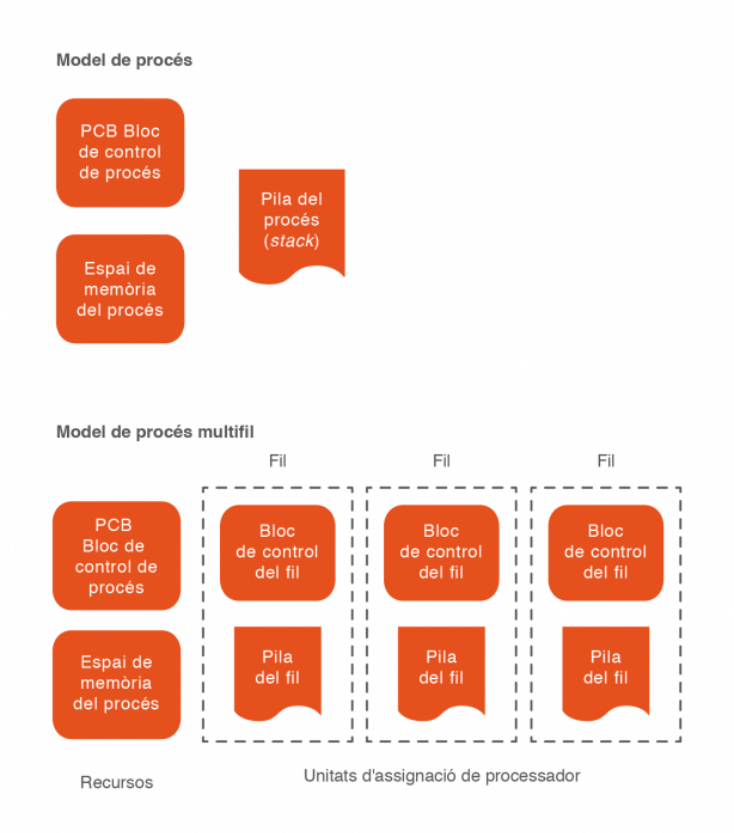
Així, doncs, en un procés hi pot haver un o més fils d'execució que disposen dels seus propis:

- Context, és a dir, el comptador de programa, estat del fil (preparat, bloquejat, etc.) i els registres de CPU, tot definit en una estructura de dades anomenada *bloc de control del fil*.
- Pila d'execució per les variables locals.

En canvi, tots els fils d'execució d'un mateix procés comparteixen certs recursos com ara l'espai de memòria, les variables globals i els arxius i dispositius d'entrada i sortida.

Els beneficis clau de la implementació de sistemes multifil rau en la millora del rendiment. És triga menys temps a crear i finalitzar fils d'execució, així com a fer canvis de context entre fils d'un mateix procés. La comunicació entre fils també és més ràpida, ja que comparteixen memòria i recursos als que poden accedir sense invocar el nucli del sistema operatiu. Finalment, en sotmetre un mateix procés a diferents fluxos d'execució es manté una única còpia del codi en memòria i, en sistemes multiprocessador, fins i tot és possible l'execució simultània de diferents fils del mateix procés en diferents processadors. Vegeu la figura 1.6.

FIGURA 1.6. Model tradicional de procés i model de procés multifil



En un sistema multifil cada fil d'execució és independent i fins i tot por executar-se simultàniament en processadors diferents, però compartint els mateixos recursos de memòria i dispositius d'entrada/sortida.

1.3 Prioritats dels processos

Un dels factors fonamentals que intervenen en la planificació a curt termini dels processos és la prioritat, que determina la cua on un procés haurà d'esperar el seu torn. A cada procés se li assigna una prioritat en funció de si és més crític o més urgent i dels recursos que necessita, la qual cosa determinarà finalment la freqüència d'accés al processador.

1.3.1 Tipus de prioritats

Podem establir una classificació de les prioritats en funció de la seva possibilitat de variació al llarg de l'execució del procés. Així, tindrem:

- **Prioritat estàtica:** Aquelles que no poden ser modificades mentre s'executa el procés.

- **Prioritat dinàmica:** Quan un procés pot modificar la seva prioritat en funció de determinats esdeveniments.

  Cada procés té assignada una prioritat que pot ser estàtica, dinàmica o una combinació de les dues. El problema de la prioritat estàtica és la inanició, és a dir, que un procés mai s'executi per no tenir mai la prioritat suficient. Per solucionar això es fa servir la prioritat dinàmica per envelliment, en la qual els processos van augmentant la seva prioritat en funció del temps que fa que estan en espera.

D'altra banda, la prioritat d'un procés està composta per dos factors:

- **Prioritat assignada pel sistema operatiu** en funció del tipus de procés, algorismes de planificació, polítiques per preveure la inanició, etc.
- **Prioritat assignada pel propi usuari**. Naturalment només afecta als processos d'usuari i permet tenir en compte els seus interessos i necessitats. Aquesta llibertat comporta un cert risc, com, per exemple, deixar bloquejada la resta del sistema si un programa amb prioritat alta assignada per l'usuari entra en un bucle infinit.

Als sistemes Unix/Linux, la prioritat d'usuari s'anomena prioritat *nice* i es pot assignar amb les ordres *nice* i *renice*.

1.3.2 Ordres per gestionar la prioritat

Més és menys

Nice vol dir 'amable' en anglès. Així s'entén millor per què quan més gran és el valor de nice menys prioritat té el procés i per tan és més "amable" amb la resta de processos que competeixen pels recursos.

Tots els processos tenen una prioritat assignada per l'usuari, anomenada **prioritat*nice***. Els processos amb més prioritat tenen valors negatius de *nice* i fan servir més recursos que els altres. La prioritat màxima assignada a *nice* és –20 i la prioritat mínima és +19. Si la prioritat no està definida, cada procés s'executarà amb una prioritat *nice* de 0.

Ordre nice

L'ordre *nice* permet arrencar un procés assignant-li d'entrada una determinada prioritat i té com a argument el nom de l'ordre que genera el procés sobre el qual volem actuar. Si no especifiquen el nivell de prioritat queda definida per defecte en 10.

```
1 $ nice –n 19 987
```

Aquesta ordre deixa amb prioritat mínima el procés de PID=987.

Ordre renice

Per canviar la prioritat *nice* d'un procés ja engegat es fa servir l'ordre ***renice***. Per exemple:.

```
$ renice –n –5 987 –u root
```

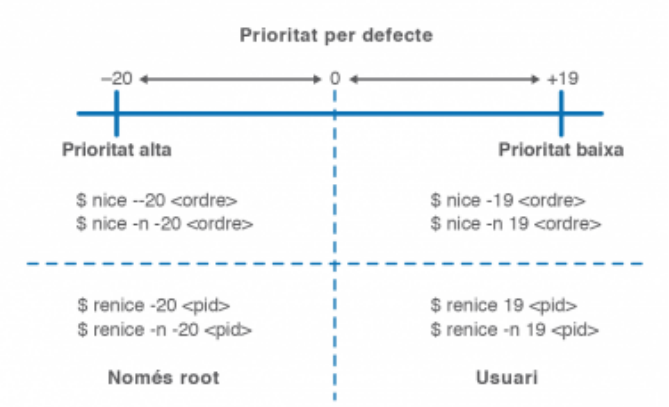
Aquesta ordre donarà prioritat –5 al procés de PID=987 i a tots els processos de l'usuari *root*.

> Qualsevol usuari pot reduir la prioritat del seus processos, però només el superusuari pot augmentar la prioritat d'un procés aplicant un valor de *nice* negatiu.

També es pot canviar la prioritat *nice* amb l'ordre interactiva ***top***.

Podem veure un esquema de la prioritat *nice* i la sintaxi de les ordres *nice* i *renice* a la figura 1.7:

Figura 1.7. Canvis de prioritat amb nice i renice



Herència de la prioritat

Els processos fill hereten la prioritat del pare, però si es canvia la prioritat del procés pare, els processos fills ja nascuts no canvien de prioritat.

1.4 Comunicació amb els processos

Hi ha diferents mecanismes que permeten la comunicació entre els processos, alguns dels quals veurem en aquest apartat. D'una banda, tot procés té associat uns arxius (stdin, stdout, stderr) que reben la comunicació d'entrada i sortida de dades. Aquests fluxos de dades dels processos poden ser intercomunicats mitjançant **canonades**.

D'altra banda, tot procés que finalitza torna informació a una **variable de retorn**. Aquesta variable pot ser llegida per un altre procés i condicionar la seva funcionalitat.

Finalment, un procés pot rebre informació i ordres mentre s'està executant mitjançant l'enviament de determinats senyals.

1.4.1 Canonades

Entre els recursos associats a un procés hi ha la interfície estàndard de comunicació amb l'exterior. Aquesta interfície consta de tres fitxers coneguts com a arxius d'entrada, de sortida i d'errors estàndard. A Unix/Linux, quan un procés obre un arxiu, el nucli del sistema li lliura un número sencer que el procés farà servir per realitzar operacions d'entrada i sortida. Els descriptors associats als arxius estàndard estan indicats a la taula 1.1.

TAULA 1.1. Arxius estàndard d'entrada i sortida

Descriptor	Denominació	Descripció
0	**stdin**	Entrada estàndard, associada per defecte al teclat del terminal.
1	**stdout**	Sortida estàndard, associada per defecte a la pantalla del terminal.
2	**stderr**	Sortida d'errors, associada per defecte a la pantalla del terminal. Mostra els missatges d'error.

1.4.2 Valors de retorn

Tot procés que finalitza torna un número comprès entre 0 o 255 que representa la causa per la qual va finalitzar. A la taula 1.2 podem veure alguns valors de retorn.

La variable d'entorn ***?*** conté el valor de retorn de la darrera ordre executada i es pot visualitzar amb *echo$?*.

TAULA 1.2. Valors de retorn d'un procés

retorn	Descripció
0	Finalització correcta del procés. Representa un valor lògic vertader si es fa servir en una comparació o iteració. Qualsevol altre valor diferent de 0 es considera fals.
1	Indica error. Finalització anormal del procés
129 ... 160	Indica que el procés ha finalitzat com a conseqüència d'un senyal. Restant 128 al valor de retorn s'obté el codi d'aquest senyal.

1.4.3 Senyals

El senyals són interrupcions que rep el procés mentre està en execució per indicar que s'ha produït algun esdeveniment significatiu davant del qual ha de respondre.

El senyals poden ser enviats pel nucli del sistema, per altres processos o pel mateix usuari. Alguns d'aquests senyals poden ser capturats (*trap*), és a dir, poden ser ignorats o rebre un tractament específic pel procés si així està programat. En canvi, els senyals `SIGKILL` i `SIGSTOP` s'han d'atendre obligatòriament i no es poden ignorar.

Hi ha un bon grapat de senyals que podem visualitzar amb l'ordre *kill* (vegeu la figura 1.8).

```
$ kill -l
```

Figura 1.8. Llistat dels senyals amb kill -l

```
root@ioc-Server:~# kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL       5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
```

Els senyals més importants s'han resumit a la taula 1.3.

Taula 1.3. Senyals més emprats amb la seva descripció

Núm.	Senyal	*Trap*	Descripció
1	**SIGHUP**	SÍ	Finalització per tancament terminal o del procés pare
2	**SIGINT**	SÍ	Finalització per teclat (*Ctrl+C*)
9	**SIGKILL**	NO	Finalització forçada
15	**SIGTERM**	SÍ	Finalització ordenada per defecte
18	**SIGCONT**	SÍ	Continua un procés aturat
19	**SIGSTOP**	NO	Atura el procés
20	**SIGSTP**	SÍ	Atura el procés per teclat (*Ctrl+Z*)

Hi ha un grup de senyals de finalització que tenen com a objectiu l'acabament del procés i l'alliberament del recursos que feia servir (senyals 1, 2, 9 i 15). D'altres tenen com a objectiu aturar el procés però sense treure'l de memòria per poder engegar-lo de nou en el mateix punt en què es va deixar (senyals 19 i 20).

Per obtenir informació completa dels senyals:

```
$ man -s7 signal
```

1.4.4 Ordres relacionades amb senyals

A continuació es detallen una sèrie de combinacions de tecles i ordres de consola relacionades amb l'enviament de senyals als processos.

Senyals de teclat

Hi ha una sèrie de combinacions de tecles que envien senyals directament al procés que s'està executant en primer pla:

- ***Ctrl+Z***: Envia el senyal 20 (**SIGTSTP**). Aquest senyal atura el procés, però el deixa a la memòria i es pot reprendre en el punt en el qual va quedar mitjançant un senyal 19 (SIGCONT).
- ***Ctrl+C***: Envia el senyal 2 (**SIGINT**). La majoria d'aplicacions està programades per finalitzar quan reben aquest senyal, però cal tenir en compte que es pot programar una aplicació perquè capturi aquest senyal i l'ignori.
- ***Ctrl+***: Envia el senyal 3 (**SIGQUIT**). També depèn de com estigui configurada l'aplicació, però en principi ha de produir una finalització del procés amb un bolcat a un arxiu del seu estat en el moment de la finalització.

Ordre kill

Podem fer servir l'ordre interna *kill* per enviar senyals a qualsevol procés que haguem creat tant en primer com en segon pla. A més, si som administradors podem enviar senyals a processos d'altres usuaris.

L'ordre *kill* necessita identificar el procés mitjançant el seu PID (identificador de procés), tot i que si es tracta d'un procés en segon pla del nostre terminal, també pot fer servir el seu número de treball (*job*) precedit del símbol %.

> Per defecte, tant l'ordre *kill* com *killall* envien el senyal 15 SIGTERM de finalització ordenada del programa.

Veguem-ne algun exemple:

```
# kill 2343
```

En aquest cas s'envia el senyal de finalització ordenada SIGTERM al procés de PID=2343. En alguns casos, els processos no responen al senyal de finalització per defecte i cal enviar el senyal de finalització forçada, que no és pot capturar i s'ha d'obeir. Qualsevol de les sintaxis següents és vàlida:

```
# kill -9 2343
# kill -SIGKILL 2343
```

Ordre killall

Amb l'ordre *kill* podíem enviar senyals a un procés determinat definit pel seu PID. Si volem enviar senyals a un procés a partir del seu nom hem de fer servir ***killall***.

```
# killall -9 mozilla
```

Aquesta instrucció tancarà tots els navegadors Mozilla oberts per l'usuari que executa l'ordre.

Ordre nohup

Quan tanquem una sessió, tancant la finestra del terminal si estem en l'entorn gràfic o mitjançant l'ordre *exit* o *Ctrl+D* si estem en l'entorn de línia d'ordres, el procés que gestiona el terminal (*getty*, *mingetty*) intenta tancar tots els processos associats a aquell terminal. Per fer aquest tancament envia el senyal SIGHUP a tots els processos en marxa i també als aturats en segon pla, ja que prèviament els ha activat amb un senyal de SIGCONT.

Si volem que algun dels nostres processos es mantingui actiu després de tancar la sessió, l'hem de protegir d'aquest senyal SIGHUP amb l'ordre *nohup*. La sintaxi és simple: cridem l'ordre *nohup* i li donem com a paràmetre l'ordre a protegir:

```
# nohup ./prova.sh
nohup: es descarta l'entrada i s'afegeix l'eixida a «nohup.»out
```

En aquest exemple, l'execució de l'script ./prova.sh està protegida contra el senyal SIGHUP, la seva sortida estàndard ja no és la consola (que podria estar tancada), sinó que la sortida és redreçada i afegida a un arxiu específic anomenat ***nohup.out***.

Ordre disown

L'ordre interna ***disown*** també ens permet protegir processos del senyal SIGHUP. Les diferències amb *nohup* són les següents:

1. L'ordre *disown* s'aplica a treballs en segon pla que ja existeixen.
2. A diferència de *nohup*, no canvia les sortides estàndard i d'error.

Vegem-ne alguns exemples:

```
# disown -h %3
```

Protegeix contra SIGHUP el treball 3 de segon pla.

```
# disown -a
```

Protegeix contra SIGHUP tots els treballs en segon pla.

```
# disown -r
```

Protegeix contra SIGHUP tots els treballs en segon pla que estan en marxa (*running*).

Ordre trap

És una ordre interna que ens permet capturar un senyal i especificar el que volem fer quan ens arribi. La seva sintaxi és *trap* [ordres][senyals]:

```
trap "" SIGTERM
```

Aquesta ordre dins d'un script capturarà el senyal de finalització SIGTERM i l'ignorarà, ja que com a ordres a seguir li hem posat una cadena buida.

1.5 Seqüència d'arrencada del sistema

En el moment que engeguem l'interruptor de l'ordinador, comença una llarga seqüència d'accions i execucions de processos que finalment ens porta fins que a la pantalla ens apareix l'entorn gràfic del sistema operatiu o una consola de text ens convida amb el *prompt* (indicador d'ordres) a introduir una ordre.

Fem ara un repàs resumit de tota aquesta seqüència i, al llarg dels diferents epígrafs, posarem especial atenció en els darrers esdeveniments que culminen en la creació de l'arbre de jerarquia de processos i en la càrrega de l'intèrpret d'ordres o *shell*.

1. Després de reiniciar (*reset*) o d'engegar l'interruptor de l'equip, els components electrònics reben alimentació. El microprocessador comença a executar instruccions des de una posició del mapa de memòria determinada, anomenada v*ector de reset* i comença a executar un programa especial anomenat BIOS (*basic input/output system*) que està allotjat a la memòria fixa del sistema de tipus ROM (*memòria només de lectura*).

GRUB
Acrònim de grand unified bootloader. Carregador d'arrencada múltiple del projecte GNU que es fa servir per engegar un dels sistemes operatius instal·lats en el mateix ordinador.

2. Aquest programa fix que incorpora cada placa base constitueix l'anomenat *firmware* i conté rutines de comprovació de memòria, detecció de dispositius de maquinari, rutines POST (*power-on self test*) per a la verificació del components. També ofereix a l'usuari la interacció opcional amb el sistema de memòria CMOS, que guarda la configuració de diferents característiques del sistema com ara el rellotge de temps real, la memòria secundària, la seqüència d'arrencada, etc.

3. Un cop executades les instruccions de la BIOS, la darrera acció que realitza és la cerca del sistema operatiu a la memòria secundària, habitualment el disc dur, per carregar-lo a la memòria. Per fer això s'adreça al primer sector del disc dur anomenat MBR (*master boot record*).

4. Tradicionalment, l'MBR conté un gestor d'arrencada simple i la taula de particions del disc dur, que indica quina és la partició activa que conté el sistema operatiu a carregar. Tanmateix, gairebé totes les distribucions Linux fan servir programes carregadors d'arrencada múltiple (*bootloader*) més complexos, com ara GRUB.

5. Així, doncs, en un sistema Debian, s'inicia a l'MBR la **fase 1**de**GRUB** anomenada *primer carregador de l'arrencada* (*initial program loader* o ILP).

6. Com que gairebé no hi ha espai a l'MBR per a un programa complex, cal l'execució d'una fase anomenada **fase 1.5**, que conté codi addicional i està situada als primers sectors després de l'MBR, sempre a la primera pista del disc dur per motius de compatibilitat.

7. La resta del codi de GRUB s'executa en la **fase 2** i normalment s'instal·la al sector d'arrencada de la partició on hi ha instal·lat el sistema Linux. Aquest codi interpreta la configuració de l'arxiu /boot/grub/grub.cfg, dóna suport a diferents sistemes d'arxius i permet el pas de paràmetres al nucli del sistema, a més d'interaccionar amb l'usuari mostrant el menú de selecció a pantalla.

8. Una vegada escollida la partició d'arrencada a partir del menú de GRUB es comença la càrrega del nucli del sistema operatiu pròpiament dit. En el cas de Linux, una vegada carregat el nucli del sistema operatiu comença la creació de processos que s'executen en un ordre determinat i culmina en la creació de tota una estructura jeràrquica de processos i serveis.

1.5.1 Jerarquia de processos (PID, PPID)

A Unix/Linux tots els processos s'identifiquen amb un número sencer de 16 bits que s'assigna seqüencialment a cada nou procés que es crea. Aquest número és únic per a cada procés i s'anomena *identificador de procés* o PID (de l'anglès *proces identifier*). A més, tot procés, a excepció del procés arrel *init* amb PID=1, ha estat creat per un procés pare. Així, doncs, un procés pare pot tenir molts processos fills, però qualsevol procés només té un procés pare, identificat pel seu PPID (*parent procés identifier*). Això crea una estructura jeràrquica de processos en forma d'arbre amb el procés *init* com a arrel.

Procés pare i procés fill poden o no compartir recursos i espai de memòria, i estar o no sincronitzats, és a dir, es poden executar concurrentment o bé el procés pare pot restar en espera fins a la finalització del procés fill.

1.5.2 Creació i finalització de processos. Crides al sistema

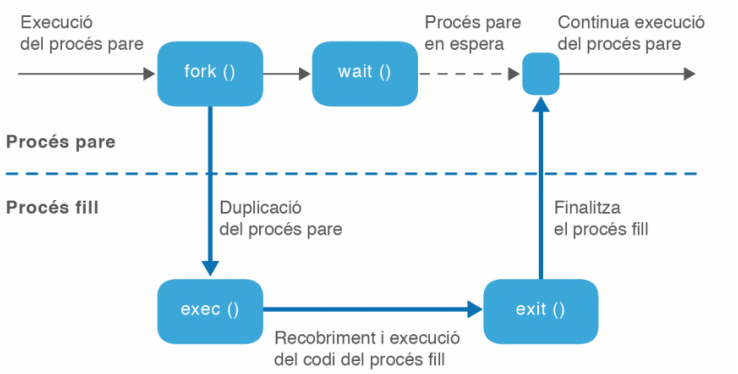
A Unix/Linux els processos en creen mitjançant crides al sistema que fan primer una duplicació del procés pare i després un recobriment del codi carregant i

executant les noves instruccions. Tot això es fa mitjançant les crides a les funcions de sistema següents:

- **Funció*fork()***: aquesta funció crea un procés fill que és una còpia pràcticament exacta del procés pare (clonació) i hereta el context del pare. Tots dos processos continuen executant-se després del punt en el qual s'ha fet la crida *fork()*, amb l'única diferència del seu PID.
- **Funció*exec()***: després de la clonació, el procés pare o bé el mateix fill han de cridar la funció *exec*, que carrega al segment de dades (recobriment) el nou codi a executar.
- **Funció*exit()***: s'invoca després de la darrera instrucció del codi per efectuar la finalització del procés i alliberar els recursos que feia servir.
- **Funció*wait()***: si el procés pare vol esperar que el procés fill finalitzi, haurà de cridar la funció *wait()*, que aturarà l'execució i el passarà a la cua de bloquejats (*waiting*) fins que el procés fill finalizi.

A la figura 1.9 es resumeixen les crides al sistema anteriors.

FIGURA 1.9. Esquema de les crides al sistema implicades en la creació d'un procés fill



A més de la finalització voluntària del procés mitjançant la crida a la funció exit(), un procés també pot finalitzar involuntàriament per:

- **Excepció**: error fatal motivat per diverses causes, instrucció privilegiada, excepció de coma flotant, violació de segment...
- **Funció *kill()***: un procés por ser finalitzat per un altre mitjançant la crida al sistema *kill()* i l'enviament del senyal de finalització corresponent.

1.5.3 Treballs en segon pla d'execució

Quan un procés pare crea un procés fill hi ha dues possibilitats en termes d'execució:

1. El procés pare espera la finalització del procés fill.
2. Pare i fill s'executen concurrentment.

D'aquesta manera en l'entorn del procés de *shell*, quan fem una crida a una ordre de sistema aquesta pot generar un procés o processos fills del *shell* que es poden executar en:

- **Primer pla (*foreground*)**: quan el *shell* queda a l'espera de la finalització de l'ordre executada i per tant el terminal no accepta noves ordres fins a la finalització del procés fill.
- **Segon pla (*background*)**: si els procés o processos fills s'executen concurrentment al procés *shell* pare, que continua acceptant noves ordres.

Directiva &

Per poder enviar l'execució d'una ordre a segon pla només cal afegir el símbol *&* al final de la línia d'ordres. Per exemple:

```
# find / -name "*.txt" &
#
```

En aquest exemple, mentre el sistema cerca en tot l'arbre de directoris els arxius amb extensió *txt*, tornem de seguida a tenir l'indicador de sistema (*prompt*), que ens acceptarà noves ordres.

El problema és que els missatges de l'ordre *find* en segon pla en apareixen a la consola i dificulten el treball, però sempre podem redreçar la sortida principal i la d'errors cap a un arxiu:

```
# find / -name "*.txt" >Llistat 2>Errors &
[1] 1544
#
```

Se'ns mostra el número de treball [1] i el PID associat al procés, 1544. Després apareix de nou l'indicador que ens convida a introduir noves ordres, com per exemple:

```
# yes
```

L'ordre *yes* genera contínuament el caràcter "y" i bloqueja el terminal. Podem fer servir la combinació de tecles *Ctrl+Z*, que enviarà aquest procés a segon pla i el deixarà aturat.

```
# ^Z
[2]+ Aturat yes
```

Ara tornem a fer la mateixa ordre, però amb més cura, enviant l'ordre directament a segon pla i la seva sortida al dispositiu *null* perquè no ens faci nosa:

```
# yes >/dev/null &
[3] 1711
```

Ja tenim un tercer treball en segon pla amb PID=1711.

Ordre jobs

L'ordre *jobs* ens permet visualitzar els treballs que tenim en segon pla i veure l'estat en el qual es troben. Si hem efectuat les anteriors ordres veurem alguna cosa semblant a:

```
# jobs
[2]+ Aturat yes
[3]– S'est? executant yes > /dev/null &
```

Com podeu veure, la primera ordre *find* ja no apareix, doncs ha finalitzat. Tenim el segon treball *yes* que hem aturat a segon pla amb la combinació *Ctrl+Z* i un tercer treball en marxa executant-se en segon pla, però amb la seva sortida redreçada al dispositiu *null*.

Ordres fg i bg

Aquestes ordres permeten reprendre una tasca aturada a segon pla en el punt on es va aturar. Aquesta tasca es pot reprendre en primer pla amb *fg* o en segon pla amb *bg*. Per indicar quina de les tasques aturades en segon pla volem reprendre cal indicar el nombre de treball precedit del símbol %.

```
# fg %2
```

Amb aquesta ordre reprendrem a primer pla el treball [2] que estava aturat. Començarà a sortir lletres "y" a la pantalla. Podem tornar a aturar el procés i enviar-lo a segon pla amb la combinació *Ctrl+Z* o parar el procés definitivament amb *Ctrl+C* i podrem tornar a cridar *jobs*:

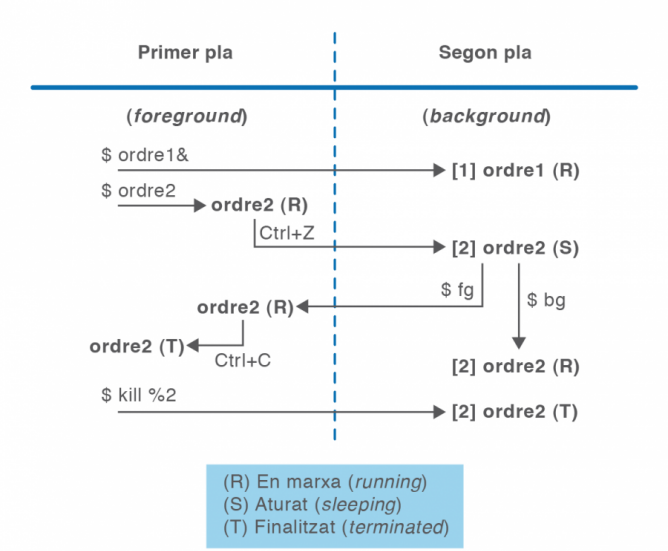
```
^C
# jobs
[3]+ S'est? executant yes > /dev/null
```

Ara només ens queda un treball en execució en segon pla. Per finalitzar definitivament un procés que s'està executant al *background* podem fer servir l'ordre *kill* tot indicant el número de PID o bé el número de treball precedit del símbol %.

```
kill %3
```

A la figura 1.11 podeu veure un resum d'aquestes ordres i de la seva funcionalitat.

FIGURA 1.10. Ordres principals per gestionar el processos a primer i segon pla



1.5.4 Dimonis

Un dimoni (de l'anglès *daemon*) és un procés no interactiu en segon pla que generalment tenim carregat a la memòria en espera d'algun senyal provinent d'un dispositiu o del mateix nucli del sistema per a despertar-se i realitzar les accions i funcions necessàries i oferir un determinat servei.

Daemon és l'acrònim de *disk and execution monitor*.

Els dimonis no disposen d'interfície amb l'usuari, per tant, no utilitzen les entrades i sortides estàndard per comunicar errors o enregistrar el seu funcionament, sinó arxius de registre (*log*), situats habitualment al directori /var/log.

Encara que un dimoni sigui un procés com qualsevol altre, que s'executa en segon pla (*background*), la manera de gestionar-lo i invocar-lo és diferent a la resta d'ordres i programes del sistema. Generalment, els dimonis tenen un guió de *shell* (*shell script*) situat al directori /etc/init.d/ que permet iniciar-los, parar-los o veure el seu estat d'execució segons la sintaxi:

```
1 # /etc/init.d/NomDimoni Accio
```

Els paràmetres d'acció bàsics que ha d'acceptar el guió del dimoni són:

- ***start***: per iniciar el dimoni. Si aquest ja s'executa es mostra un missatge d'error.
- ***stop***: per parar el dimoni. Si no s'executa es mostra un missatge d'error.
- ***restart***: reinicia el dimoni i serveix perquè es tornin a llegir els seus arxius de configuració.
- ***reload***: encara que no tots els dimonis ho permeten, aquesta acció permet recarregar els arxius de configuració sense haver de parar el dimoni.

Per exemple:

```
# /etc/init.d/cupsd start
# /etc/init.d/networking restart
```

La primera línia posa en marxa el servei d'administració d'impressores CUPS i la segona reinicia la xarxa llegint els seus arxius de configuració.

Algunes distribucions, entre elles Debian, disposen de l'ordre *service* que permet fer el mateix sense especificar la ruta completa:

```
# service cupsd stop
```

1.5.5 Nivells d'execució

Els dimonis que estan en execució en un moment determinat ens marquen els serveis que un sistema operatiu ofereix com a servidor i que rep com a client. Per organitzar adequadament la quantitat i interacció dels serveis que necessitem en un entorn o circumstància determinats disposem del nivells d'execució (*runlevels*, en anglès). Així, doncs, un nivell d'execució no és més que l'agrupació d'una sèrie de dimonis en execució que té com a finalitat crear un entorn de serveis ajustat a unes necessitats concretes.

Generalment, els sistemes Unix/Linux ens proporcionen diferents nivells d'execució, amb la funcionalitat indicada a la taula 1.4.

TAULA 1.4. Nivells d'execució (runlevels) i la seva funcionalitat

Nivel	Funcionalitat
0	El nivell d'execució 0 està configurat per aturar el sistema.
1	Nivell per a un usuari únic (*single user*). Es posen en marxa els dimonis mínims per fer tasques de manteniment i només permet l'entrada a l'arrel (*root*).
2 a 5	Nivells destinats a ser configurats segons les necessitats de cada instal·lació encara que habitualment tots són multiusuari. En algunes distribucions el nivell 2 està configurat per defecte com a multiusuari sense servei de xarxa, el nivell 3 com a multiusuari amb servei de xarxa, i el nivell 5 per engegar el sistema amb l'entorn gràfic.
6	Aquest nivell està preparat per reiniciar el sistema.

El nivell d'execució de la vostra màquina es pot consultar des de consola amb l'ordre *runlevel*, que requereix privilegis de superadministrador (root)

Cada nivell d'execució té un directori associat a `/etc/rcX.d` (on X és el número del nivell). En aquests directoris hi trobem **enllaços simbòlics** als guions de *shell* que controlen als dimonis i que ja hem vist que estan situats al directori `/etc/init.d`.

A més dels subdirectoris dels diferents nivells, n'hi ha un altre, `/etc/rcS.d`, que conté les referències als serveis bàsics que s'executen prèviament a qualsevol altre nivell.

Vegem per exemple el contingut del directori associat al nivell d'execució 0:

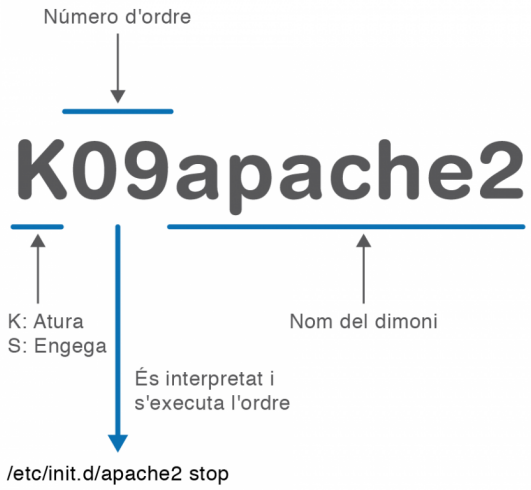
```
# ls /etc/rc0.d

K09apache2 K70vboxadd S15wpa-ifupdown S40umountfs
K10monit K70vboxadd-x11 S20sendsigs S60umountroot
K20netdiag K74bluetooth S30urandom S90halt
K20ntop S35networking
```

La primera lletra del nom d'aquestos enllaços simbòlics porta informació sobre l'acció que han de fer:

- **K**: si l'enllaç comença per *K* (*kill*) indiquem que volem aturar el dimoni.
- **S**: si l'enllaç comença per *S* (*start*) indiquem que volem activar el dimoni.

Després d'aquesta lletra es posa un número de dues xifres, entre 00 i 99, que indica l'ordre en la seqüència d'inici i aturada de serveis. Aquest ordre és important, ja que alguns dimonis tenen dependències, és a dir, necessiten que altres estiguin en execució abans de ser iniciats. Finalment trobem el nom del dimoni en qüestió que ens interessa parar o activar. Vegeu la figura 1.11:

FIGURA 1.11. Sintaxi dels enllaços simbòlics als scripts de control dels serveis



L'ordre per canviar de nivell d'execució és ***init*** i li passem com a paràmetre el nivell d'execució que volem engegar. Així, per reinicialitzar el sistema:

```
# init 6
```

El procés de canvi de nivell és el següent: el sistema inspecciona el directori corresponent al nivell que volem habilitar i començarà primer a detenir els dimonis corresponents a les entrades que comencen per K passant el paràmetre *stop* i després iniciarà els corresponents a les entrades que comencen per S mitjançant el paràmetre *start*.

La modificació de la configuració d'un nivell d'execució es pot fer manualment:

1. Incloure el script de tractament del servei en el directori `/etc/init.d`.
2. Crear els enllaços simbòlics en cada directori de nivell d'execució (`/etc/rcX.d`) donant en el mateix nom la informació de ordre seqüencial d'execució i començant per K o S si el procés s'ha d'aturar o engegar respectivament.

També es pot fer de manera més còmoda amb l'ordre *update-rc.d*.

Ordre update-rc.d

A Debian disposem de l'ordre *update-rc.d* per crear automàticament els enllaços simbòlics necessaris per a cada nivell d'execució. Vegem-ne alguns exemples:

```
1 # update-rc.d cups defaults
```

Inscriu el servei d'impressió CUPS amb els paràmetres per defecte, és a dir, que s'engegui en els nivells del 2 al 5 i que s'aturi en els nivells 0, 1 i 6. L'ordre d'aturada/engegada serà el 20 per defecte. L'script de servei ha d'estar a `/etc/init.d/cups`.

```
1 # update-rc.d cups start 10 3 . stop 05 0 1 6
```

En aquest cas només s'engegarà en l'ordre de posició 10 en el nivell 3 i s'aturarà en els nivells 0, 1 i 6 en la posició 05.

```
1 # update-rc.d -f cups remove
```

El paràmetre *remove* suprimeix els vincles dels diferents directoris, però l'script de servei associat (`/etc/init.d/cups`) ja no ha d'existir. En cas contrari s'ha de fer servir l'opció -f per forçar la supressió dels vincles.

chkconfig

Altres distribucions com Fedora/Red hat i openSUSE fan servir l'ordre chkconfig per configurar els serveis en els diferents nivells d'execució.

Directori /etc/default

Molt sovint cal configurar algunes variables per controlar el comportament dels scripts dels serveis que es troben al directori `/etc/init.d`.

Si aquestes variables es defineixen dins de l'script del servei s'haurien de reconfigurar cada vegada que actualitzem el paquet. Per facilitar la tasca d'administració i garantir que les variables de configuració estan sempre disponibles es poden guardar en arxius específics i independents situats al directori **/etc/default**.

Aquests arxius només han de contenir la declaració de variables i, en tot cas, comentaris explicatius. A continuació posem un exemple del contingut d'un d'aquest arxius:

```
1 $ cat /etc/default/cups
2 # Cups configure options
3 # LOAD_LP_MODULE: enable/disable to load "lp" parallel printer driver module
4 LOAD_LP_MODULE=yes
```

1.5.6 Sistema d'arrencada

La jerarquia de l'arbre de processos, el seu mecanisme de creació i finalització, el concepte de dimoni i servei i l'agrupació de serveis en diferents nivells d'execució configurables, són part fonamental de la seqüència final d'arrencada del sistema operatiu Linux i la creació del seu entorn de processos i serveis.

El nucli del sistema operatiu finalment està carregat ja a la memòria i activa dos processos previs a la generació de tot l'arbre jeràrquic de processos:

- **El planificador (*scheduler*)**: procés amb PID=0 que gestiona l'assignació de processos a la CPU per a la seva execució.
- **El procés d'inici*init*** amb PID=1. El pare de tots els processos. Tot l'arbre de processos de Linux és fill d'aquest procés.

Per generar l'arbre de processos tenim dos enfocaments: un de seqüencial, en el qual els diferents serveis s'engeguen seguint un ordre prefixat i configurat prèviament en els nivells d'execució, i un enfocament basat en esdeveniments, en el qual els serveis s'inicien depenent de l'ocurrència de determinats successos.

Sistema seqüencial: procés init

Tradicionalment, els sistemes Linux han fet servir un sistema d'arrencada heretat d'Unix System V i que està basat en el procés *init*. Aquest procés l'inicia el nucli del sistema i és l'encarregat de posar en marxa tots els serveis necessaris definits en el nivell d'execució configurat per defecte.

Per realitzar la seva tasca, el procés *init* fa servir la informació continguda en l'arxiu de configuració **/etc/inittab**, que conté, habitualment:

- El nivell d'execució per defecte en arrencar.
- Els scripts que s'han d'executar previs al nivell d'execució per defecte (continguts a /etc/rcS.d).
- La configuració dels diferents nivells d'execució disponibles al sistema.
- L'acció que s'ha de realitzar en prémer *Ctrl+Alt+Del* o amb altres combinacions de tecles.
- La definició de consoles obertes en cada nivell d'execució.

Vegem un exemple de l'arxiu /etc/inittab amb alguns comentaris:

```
$ cat /etc/inittab

# Es defineix el nivell d'execució per defecte.

```

```
id:2:initdefault:

# Aquest és el primer script que s'executa previ a qualsevol # nivell d'execuci
    ó

si::sysinit:/etc/init.d/rcS

# Engega el script /etc/init.d/rc i li passa com paràmetre el nivell d'execució
    . Aquest script rc és el que recorre el directori /etc/rcN.d corresponent
    i executa en l'ordre adequat els inicis i finalitzacions dels serveis
    indicats.

l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6

# Ordre a executar en cas de CTRL-ALT-DEL

ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now

# Indicacions del terminals tty engegats en cada nivell d'execució. Es veu que
    en els nivells 2 i 3 tenim les sis consoles activades. L'acció respawn vol
     dir que si el procés finalitza, torna a arrencar automàticament.

1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
3:23:respawn:/sbin/getty 38400 tty3
4:23:respawn:/sbin/getty 38400 tty4
5:23:respawn:/sbin/getty 38400 tty5
6:23:respawn:/sbin/getty 38400 tty6
```

Un cop s'han acabat d'executar tots aquests scripts, s'executa finalment l'script d'execució local ***/etc/rc.local***.

Sistema basat en esdeveniments: procés upstart

El sistema d'arrencada de System V basat en *init* és un procés síncron que executa seqüencialment una sèrie de tasques definides amb antelació i que no activa un procés fins a haver finalitzat l'anterior. A més, aquest sistema només executa aquestes tasques quan *init* canvia d'estat, cosa que només passa quan la màquina s'encén, s'apaga o s'està reiniciant. Aquest fet fa que *init* no gestioni adequadament altres serveis que és necessari executar no quan es canvia de nivell, sinó en funció de determinats esdeveniments.

Per això, algunes distribucions de Linux estan començant a substituir el tradicional *init* de System V per un sistema basat en esdeveniments anomenat ***upstart***, però intentant sempre mantenir la compatibilitat enrere fent transparent el canvi a l'usuari. De fet, l'arxiu binari que gestiona *upstart* es diu igualment /sbin/init, encara que la seva manera de funcionar és molt diferent.

El model basat en esdeveniments d'*upstart* permet respondre de manera específica a determinats successos i no a seqüències predeterminades. Aquest asincronia fa que es puguin posar en marxa en paral·lel diferents tasques amb l'objectiu de minimitzar el temps d'arrencada.

En el sistema *upstart* l'arxiu de configuració /etc/inittab desapareix i es fan servir uns arxius de definició de treballs (*jobs*, en la terminologia *upstart*) situats al directori /etc/init.

```
ls /etc/init
acpid.conf mountall.conf rcS.conf anacron.conf mountall-net.conf rc-sysinit.
    conf apport.conf mountall-shell.conf rsyslog.conf atd.conf mounted-dev.
    conf tty1.conf
avahi-daemon.conf mounted-tmp.conf tty2.conf
...
```

Aquests arxius d'extensió *conf* defineixen les condicions i els esdeveniments que s'han de produir per engegar o parar els serveis. Vegem un exemple resumit d'un d'aquest arxius:

```
$ cat /etc/init/mysql.conf
# MySQL Service

description "MySQL Server"
author "Mario Limonciello <superm1@ubuntu.com>"

start on (net-device-up
and local-filesystems
and runlevel [2345])
stop on runlevel [016]
respawn

env HOME=/etc/mysql
umask 007
...
exec /usr/sbin/mysqld
end script
```

Veiem que l'arxiu mysql.conf defineix que el dimoni mysqld es posarà en marxa (exec /usr/sbin/mysqld) quan es detecti que hi ha xarxa (esdeveniment net-device-up), el sistema d'arxius ha estat muntat (esdeveniment local-filesystems) i el nivell d'execució sigui multiusuari (entre el 2 i el 5). Si es detecta un esdeveniment de nivell d'execució 0, 1 o 6 el dimoni s'aturarà.

També es configura amb un d'aquests arxius (control-alt-delete.conf) l'acció associada amb l'esdeveniment de teclat *Ctrl+Alt+Supr*:

```
$ cat /etc/init/control-alt-delete.conf

# control-alt-delete - emergency keypress handling
#
# This task is run whenever the Control-Alt-Delete key combination is pressed,
    and performs a safe reboot of the machine.

description"emergency keypress handling"
author"Scott James Remnant <scott@netsplit.com>"

start on control-alt-delete

task
exec shutdown -r now "Control-Alt-Delete pressed"
```

Adopció d'upstart

Les primeres distribucions en fer servir *upstart* com a sistema d'arrencada han estat Ubuntu i Fedora.

Upstart té com un dels seus objectius clau ser compatible amb el sistema Unix System V. Per això:

- Conserva el directori **/etc/init.d** per als scripts de control dels serveis que

encara no s'hagin migrat al sistema d'esdeveniments d'*upstart*.

- Implementa un treball (*job*) específic **/etc/init/rc-sysinit.conf** que simula els canvis de nivell d'execució, gestiona els scripts de /etc/init.d i permet definir el nivell d'execució per defecte.

- Manté l'ús del directori **/etc/default** per als arxius de configuració de variables que permetran ara el control del comportament tant dels scripts tradicionals de /etc/init.d com els arxius de definició de treballs d'*upstart* de /etc/init.

- Per engegar un servei manualment també podrem fer servir la instrucció ***service***.

Seqüència de processos getty, login i shell

Sigui quin sigui el procediment de posada en marxa dels serveis del sistema s'arriba finalment, dins de l'arbre de processos, a activar els terminals o consoles *tty* mitjançant el **procés*getty*** o bé *mingetty*. Aquests terminals es defineixen a l'arxiu **/etc/inittab** (als arxius /etc/init/ttyN.conf, en el cas d'*upstart*) i fan servir els dispositius /dev/ttyN, on N és el número de la consola. En aquest moment es visualitza el contingut de l'arxiu **/etc/issue** i la creació de processos s'atura en espera que l'usuari iniciï una sessió.

PAM

Acrònim de pluggable authentication modules. Consisteix en un mecanisme d'autenticació flexible que permet abstraure les aplicacions del procés d'identificació.

Per iniciar una sessió, l'usuari ha d'identificar-se amb un nom i una contrasenya mitjançant el **procés de login**. Aquestes credencials són verificades a /etc/passwd i /etc/shadow o bé fent servir mòduls PAM.

Una vegada identificat l'usuari, es presenta el contingut de l'arxiu **/etc/motd** (de l'anglès *message of the day*), i es llegeixen diferents arxius de configuració de perfil tan generals com particulars de l'usuari concret, entre ells:

- `/etc/profile`
- `/etc/bash.bashrc`
- `~/.bashrc`
- `~/.profile`

Finalment es carrega l'intèrpret d'ordres o ***shell*** que s'hagi configurat a l'arxiu /etc/passwd per a aquell usuari concret. Hi ha tota una sèrie d'alternatives de *shells* que es poden veure llistades a /etc/shells. Un dels intèrprets d'ordres més emprats és el Bash, evolució de l'original intèrpret *sh*.

La càrrega del *shell*Bash dóna per finalitzat el sistema d'arrencada de Linux i la posada en marxa de l'entorn de serveis, i el sistema operatiu queda en espera de la introducció d'ordres per part de l'usuari.

1.5.7 Aturada del sistema

Ja hem vist que *init* gestiona les aturades del sistema amb els nivells 0 i 6:

- **Nivell d'execució 0**: Para el sistema aturant els diferents processos configurats.
- **Nivell d'execució 6**: Atura el sistema i el reinicia.

Tanmateix, disposem d'una ordre específica, *shutdown*, que ens proporciona funcionalitats addicionals.

Senyals d'aturada

En la seqüència de parada del sistema s'envia primer un senyal SIGTERM a tots els processos actius i, passats uns segons, un senyal d'aturada forçada SIGKILL.

Ordre shutdown

En definitiva, *shutdown* crida a *init* 0 o *init* 6, però accepta paràmetres com ara el temps de termini per a l'apagada o reinicialització del sistema i la possibilitat d'enviar missatges d'advertiment. La seva sintaxi és:

```
shutdown <paràmetres><termini><missatge>
```

Vegeu les opcions de l'ordre en la taula 1.5.

Taula 1.5. Opcions de l'ordre shutdown

opció	Significat
-k	No atura el sistema sinó que envia un missatge d'advertiment a tots els usuaris.
-r	Reinicialització del sistema (*reboot*).
-h	Aturada del sistema (*halt*).
-f	Impedeix l'execució de la utilitat d'anàlisi i correcció del sistema d'arxius (fsck) en l'arrencada.
-F	Força l'execució d'*fsck* en l'arrencada.
-c	Cancel·la l'execució de *shutdown*.

Es pot especificar el termini de tancament del sistema de diferents maneres:

- Una hora i un minut concret amb ***hh:mm***
- Passat un nombre de minuts concret amb ***+m***
- De manera immediata amb ***now*** (àlies de +0)

  En l'exemple es programa una represa per d'aquí a 10 minuts amb un missatge d'avís:

```
# shutdown -r +10 "Represa del sistema en 10 minuts"

Broadcast message from root@ioc-Server (pts/1) (Sun Mar 1 17:58:54 2012):
Reinicialització del sistema per manteniment en 10 minuts
```

```
The system is going DOWN for reboot in 10 minutes!
```

Aquesta represa es pot cancel·lar des d'una altra consola d'arrel amb l'opció *-c*.

```
# shutdown –c "represa cancel·lada"

Broadcast message from root@ioc–Server (pts/3) (Sun Mar 4 18:03:34 2012):
reinicialització cancel·lada
```

Altres ordres de tancament del sistema que encara es conserven per garantir la compatibilitat cap enrere:

- ***halt*** és equivalent a # *shutdown -h now.*
- ***reboot*** és equivalent a # *shutdown -r now.*

1.6 Monitorització de processos a Unix/Linux

Per administrar i gestionar processos en Linux es poden fer servir directament ordres de consola, tot i que també és pot treballar des de l'entorn gràfic amb la utilitat de monitorització del sistema.

1.6.1 El directori virtual /proc

Directori virtual

Definim el directori /proc com a virtual perquè no està realment al disc dur, sinó que el nucli del sistema operatiu el crea dins de la memòria RAM.

En arrencar, el nucli del sistema (*kernel*) posa en marxa un sistema d'arxius virtual /proc que emmagatzema informació que recull del sistema. El directori /proc està implementat a la memòria i no es guarda al disc dur. Les dades que conté són tant de naturalesa estàtica com dinàmica, és a dir, que varien al llarg de l'execució.

Una de les característiques interessants del directori /proc és que hi podem trobar les imatges dels processos en execució amb tota la informació gestionada pel nucli del sistema. Cada procés es pot trobar en un directori etiquetat amb el seu identificador de procés /proc/PID_proces. Aquesta informació és útil per als programes de depuració o per a les mateixes ordres del sistema com ara *ps* o *top*, que la fan servir per veure l'estat en el qual es troben els processos.

D'altra banda, a /proc hi podem trobar també arxius amb informació sobre l'estat global del sistema com ara:

- `/proc/cpuinfo`: informació sobre el processador. Per exemple, el tipus, fabricació, model i rendiment.
- `/proc/devices`: llista de controladors de dispositiu configurats en el nucli que està funcionant actualment.

- `/proc/diskstats`: conté informació estadística de les operacions d'entrada/sortida per cada dispositiu de disc.
- `/proc/dma`: mostra quins canals de DMA (accés directe a memòria) s'estan utilitzant.
- `/proc/filesystems`: llista dels sistemes d'arxius configurats dins del nucli.
- `/proc/ide`: directori d'informació del bus IDE, característiques dels discos.
- `/proc/interrupts`: mostra quines línies d'interrupció s'estan utilitzant i, per a cadascuna, un comptador de quantes n'hi ha hagut.
- `/proc/ioports`: mostra quins ports d'entrada/sortida s'estan utilitzant.
- `/proc/loadavg`: la mitjana de la càrrega del sistema expressada en tres indicadors que representen la mitjana de processos en marxa en el darrer minut, cinc minuts i quinze minuts.
- `/proc/meminfo`: informació sobre la utilització de la memòria, tant la física com la d'intercanvi.
- `/proc/net`: directori amb arxius d'informació d'estat sobre protocols de xarxa. Els fa servir l'ordre *netstat*.
- `/proc/partitions`: conté el noms de les particions del sistema i la seva mida en blocs.
- `/proc/pci`: dispositius PCI del sistema.
- `/proc/stat`: diverses estadístiques sobre el nucli del sistema, com el temps que la CPU ha dedicat a diferents tasques, el nombre d'interrupcions ateses o el temps transcorregut des de l'arrencada del sistema.
- `/proc/version`: la versió de nucli del sistema operatiu.

Tot i que els fitxers que es poden consultar a /proc acostumen a ser arxius de text de lectura fàcil amb l'ordre cat o amb qualsevol processador de textos, de vegades el format en dificulta la interpretació. És per això que hi ha altres ordres del mateix sistema que agafen aquesta informació i la presenten a l'usuari d'una manera més elaborada perquè l'entengui millor. Per exemple, l'ordre *free* llegeix el fitxer /proc/meminfo, i l'ordre *uptime* accedeix i presenta la informació de l'arxiu /proc/loadavg.

1.6.2 Línia d'ordres

El sistema operatiu Linux disposa de diferents ordres en l'entorn de consola de text per a la gestió i monitoratge de processos. Les podríem classificar en diferents apartats funcionals:

- **Monitorització de processos**: ordres *ps*, *pstree*, *pidof*, *time* i *top*.
- **Prioritat dels processos**: ordres *nice* i *renice*.
- **Ordres relacionades amb senyals**: ordres *kill*, *killall*, *nohup*, *disown*, *trap*.
- **Execució diferida**: ordres *at*, *crontab*.
- **Tasques en segon pla**: directiva *&* i ordres *jobs*, *fg* i *bg*.
- **Altres ordres**: *wait*, *sleep*.

A continuació detallarem les ordres de monitorització de processos.

Ordre ps

A la majoria de sistemes operatius de la família Unix, l'ordre *ps* (process status) permet visualitzar els processos en execució. Aquesta ordre està basada en la informació continguda en el directori virtual /proc i admet un gran nombre d'opcions que fins i tot varien en funció dels diferents formats de sintaxi:

- **Estil Unix**: les opcions són lletres majúscules o minúscules precedides d'un guionet. Per exemple: *ps -A*.
- **Estil BSD**: les opcions no porten guionet. Per exemple: *ps r*.
- **Estil GNU**: les opcions fan servir noms llargs i porten doble guionet. Per exemple: *ps – user*.

Les opcions completes de *ps* es troben a les pàgines del manual.

```
1 # man ps
```

Però per veure un resum de les opcions més comuns podem fer servir l'opció *help* en estil GNU:

```
1 # ps -–help
```

El resultat es mostra a la figura 1.12:

FIGURA 1.12. Resum de les principals opcions de l'ordre ps

```
root@ioc-Server:~# ps --help
********* simple selection *********  ********* selection by list *********
-A all processes                      -C by command name
-N negate selection                   -G by real group ID (supports names)
-a all w/ tty except session leaders  -U by real user ID (supports names)
-d all except session leaders         -g by session OR by effective group name
-e all processes                      -p by process ID
T  all processes on this terminal     -s processes in the sessions given
a  all w/ tty, including other users  -t by tty
g  OBSOLETE -- DO NOT USE             -u by effective user ID (supports names)
r  only running processes             U  processes for specified users
x  processes w/o controlling ttys     t  by tty
*********** output format **********  ********** long options **********
-o,o user-defined  -f full            --Group --User --pid --cols --ppid
-j,j job control   s  signal          --group --user --sid --rows --info
-O,O preloaded -o  v  virtual memory  --cumulative --format --deselect
-l,l long          u  user-oriented   --sort --tty --forest --version
-F   extra full    X  registers       --heading --no-heading --context
                     ********* misc options *********
-V,V  show version      L  list format codes  f  ASCII art forest
-m,m,-L,-T,H  threads   S  children in sum    -y change -l format
-M,Z  security data     c  true command name  -c scheduling class
-w,w  wide output       n  numeric WCHAN,UID  -H process hierarchy
```

Sigui quin sigui l'estil d'ordre utilitzat, depenent de les opcions invocades apareixeran diferents columnes amb informació sobre els processos. Les dades principals que apareixen en aquests camps estan resumides en la taula 1.6:

TAULA 1.6. Descripció dels camps més habituals de l'ordre ps

Camp	Significat
PID	Identificador del procés
PPID	Identificador del pare del procés
UID	Identificador de l'usuari propietari del procés
USER	Nom de l'usuari propietari del procés
TTY	Terminal associada al procés. Sense terminal associat apareix un *?*
TIME	Temps de CPU acumulat pel procés
CMD	El nom de l'ordre que va iniciar el procés
SIZE	Mida virtual de la imatge del procés
NI	Prioritat *nice* assignada per l'usuari
%CPU	Percentatge de CPU usat pel procés
%MEM	Percentatge de memòria usat pel procés en relació a la memòria física
START	Hora d'inici del procés
VSZ	Memòria virtual en quilobytes
RSS	Memòria resident en quilobytes que fa servir el procés
STAT	Estat del procés com per exemple: * R (*running*): en execució * S (*sleeping*): procés bloquejat en espera d'un succés * T (*stopped*): procés detingut però que es pot reiniciar * Z (*zombie*): procés finalitzat però que roman a la memòria

Si no hi posem cap argument, només es mostraran aquells processos iniciats per l'usuari actual i que fan referència al terminal que aquest està utilitzant.

```
# ps
PID TTY TIME CMD
1532 pts/0 00:00:00 bash
1842 pts/0 00:00:00 ps
```

La primera columna mostra l'identificador del procés, la terminal associada al procés que en aquest cas es tracta del primer pseudoterminal (pts/0), atès que hem fet servir un terminal de l'entorn gràfic, el temps acumulat d'ús de CPU i finalment l'ordre que ha generat el procés.

Afegint algunes opcions (*aux*) en format BSD podem obtenir més informació (vegeu la figura 1.13).

```
# ps aux
```

Figura 1.13. Llistat de processos i informació obtinguda amb l'ordre ps aux

```
root@ioc-Server:~# ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   2032   704 ?        Ss   Mar04   0:03 init [2]
root         2  0.0  0.0      0     0 ?        S    Mar04   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Mar04   0:00 [migration/0]
root         4  0.0  0.0      0     0 ?        S    Mar04   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S    Mar04   0:00 [watchdog/0]
root         6  0.0  0.0      0     0 ?        S    Mar04   0:03 [events/0]
 ...
root      2235  0.0  0.1   4784  1992 pts/3    Ss   Mar04   0:00 bash
root      4068  0.0  0.0   3176   500 pts/3    T    Mar04   0:00 yes
jmunoz    4336  0.0  2.7 103160 28600 ?        S    00:18   0:06 gedit
root      5152  0.0  0.1   4552  1760 pts/2    Ss+  01:59   0:00 bash
root      5447  0.0  0.1   3872  1040 pts/3    R+   02:45   0:00 ps aux
```

Per veure més informació, com per exemple els valors de prioritat (vegeu la figura 1.14):

```
# ps -l
```

Figura 1.14. Visualització dels valors de prioritat amb ps

```
root@ioc-Server:~/Documents# ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0  2235  2174  0  80   0 -  1196 -      pts/3    00:00:00 bash
0 T     0  4068  2235  0  80   0 -   794 -      pts/3    00:00:00 yes
4 R     0  5419  2235  0  80   0 -   915 -      pts/3    00:00:00 ps
```

Ordre pstree

L'ordre *pstree* mostra una llista de processos en forma d'arbre que segueix la jerarquia de processos Unix i que permet identificar fàcilment quin és el procés pare d'un altre.

```
$ pstree
init———NetworkManager
      |—3*[VBoxClient———{VBoxClient}]
      |—VBoxService———6*[{VBoxService}]
      |—acpid
      |—apache2———5*[apache2]
      |—atd
      |—avahi—daemon———avahi—daemon
      |—cron
      |—cupsd
      |—6*[getty]
      |—gnome—keyring—d———2*[{gnome—keyring—}]
      |—gnome—screensav
      |—gnome—settings—
      |—gnome—terminal———bash———pstree
      |                 |—bash—man—pager
      |                 |—gnome—pty—helpe
      |                 |—{gnome—terminal}
```

Analitzem la informació mostrada en aquest exemple. Podem veure que *init* és el pare de tots els processos, que hi han alguns dimonis coneguts en marxa (Apache, cron, CUPS, etc.), que tenim sis terminals en espera de login (`6*[getty]`). Al terminal gràfic (*gnome-terminal*) trobem que hi ha un parell de processos Bash en funcionament, i en un d'ells, la mateixa ordre *pstree* que genera l'arbre.

Ordre pidof

L'ordre *pidof* permet trobar els identificadors dels processos associats a una determinada ordre:

```
# pidof init
1
# pidof getty
1110 785 782 780 773 770
```

En l'exemple veiem que el procés *init* té PID=1 i també podem veure tots els PID de les sis consoles obertes.

Ordre time

Aquesta ordre permet executar qualsevol altra aplicació i enregistrar els recursos que ha emprat. Per defecte només presenta el temps real (estimat amb el rellotge del sistema) i el temps de CPU emprat tant en mode usuari com en mode sistema. Tanmateix, aquesta mateixa ordre executada amb privilegis de superadministrador i fent servir diferents modificadors pot subministrar més paràmetres d'informació.

```
# time gedit
real 0m13.289s
user 0m1.380s
sys 0m2.016s
```

Ordre sleep

Suspèn l'execució durant els segons especificats com a paràmetre.

```
# sleep 20;echo "Han passat 10 segons"
Han passat 10 segons
```

Ordre uptime

L'ordre *uptime* dóna l'hora del sistema, el temps que porta encès, la quantitat d'usuaris connectats i la càrrega mitjana del sistema durant l'últim minut, els últims cinc minuts i els últims quinze minuts.

```
# uptime
11:51:25 up 58 min,3 users, load average: 1.38, 1.35, 0.96
```

Ordre vmstat

L'ordre *vmstat* dóna informació de l'estat de la memòria física, de la memòria virtual, de l'intercanvi entre memòria interna i disc (*swapping*), de les transferències de disc, les interrupcions i ús del processador. Admet l'ús de modificadors i permet monitoratge continuat (vegeu la figura 1.15).

Figura 1.15. Resultat de l'ordre vmstat

```
root@ioc-Server:/home/jmunoz# vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa
 0  0      0 139992  14592 234084    0    0   371   225  100  329  4  5 83  9
```

La informació que mostra aquesta ordre és resumeix a la taula 1.7.

Taula 1.7. Camps d'informació de l'ordre vmstat

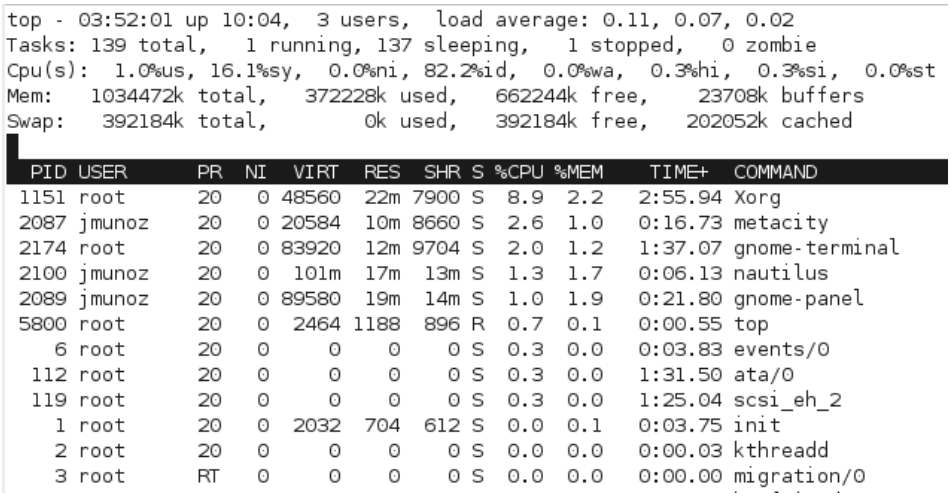
Secció	Descripció
Procs	Processos en espera de ser executats (r) i en estat d'espera (b)
Memory	Memòria virtual (swpd), memòria lliure (free), memòria intermèdia (buff) i memòria cau (cache)
IO	Blocs enviats i rebuts des de dispositius d'entrada/sortida
System	Nombre d'interrupccions per segon (in) i nombre de canvis de context (cs)
CPU	Percentatges de distribució de temps entre el mode usuari (us), mode sistema (sy) i temps ociós (id)

Ordre top

L'ordre *top* presenta la informació dels processos de manera dinàmica i interactiva, en temps real, amb una actualització per defecte cada tres segons, i ordenats pel percentatge d'ús de CPU (vegeu la figura 1.16).

```
# top
```

FIGURA 1.16. Visualització de la ordre de gestió de processos interactiva top



Com es veu al llistat, la primera línia d'informació és la mateixa que dóna l'ordre *uptime*. A continuació mostra informació dels processos en execució amb una sèrie de camps d'informació, semblants als obtinguts amb l'ordre *ps* però amb petites variacions, com podem veure a la taula 1.8.

TAULA 1.8. Descripció dels camps més habituals de l'ordre top

Camp	Significat
PID	Identificador del procés
USER	Nom de l'usuari propietari del procés
PR	Prioritat del procés
NI	Prioritat *nice* assignada per l'usuari
VIRT	Memòria virtual en quilobytes
RES	Memòria resident en quilobytes que fa servir el procés
SHR	Memòria compartida
S	Estat del procés
%CPU	Percentatge de CPU usat pel procés
%MEM	Percentatge de memòria usat en relació a la memòria física
TIME	Hora d'inici del procés
COMMAND	El nom de l'ordre que va iniciar el procés

L'ordre *top* no només mostra la informació actualitzant les dades dinàmicament, també pot interactuar amb l'usuari, que pot actualitzar la informació, ordenar els processos per qualsevol camp d'informació o, el que és més important, finalitzar el procés enviant un senyal de *kill* o fins i tot canviar-li la prioritat.

La llista de les principals ordres de *top* es mostra a continuació, a la taula 1.9:

TAULA 1.9. Descripció de les ordres interactives de l'ordre top

Ordre	Descripció
h o ?	Mostra l'ajuda.
f	Permet marcar com a visibles/ocults els diferents camps d'informació.
o	Configura l'ordre en el qual es presenten els camps d'informació.
F	Defineix el camp d'ordenació del llistat de processos.
k	Finalització d'un procés amb el senyal de *kill*.
r	Canvia la prioritat d'usuari d'un procés (*renice*).
q	Sortir de l'entorn *top*.

Per a un llistat exhaustiu de les opcions i ordres de *top*, cal fer servir l'opció *h* o *?* i s'obté la informació de la figura 1.17.

FIGURA 1.17. Llistat de les principals opcions de top

```
Help for Interactive Commands - procps version 3.2.8
Window 1:Def: Cumulative mode Off.  System: Delay 3.0 secs; Secure mode Off.

  Z,B       Global: 'Z' change color mappings; 'B' disable/enable bold
  l,t,m     Toggle Summaries: 'l' load avg; 't' task/cpu stats; 'm' mem info
  1,I       Toggle SMP view: '1' single/separate states; 'I' Irix/Solaris mode

  f,o     . Fields/Columns: 'f' add or remove; 'o' change display order
  F or O  . Select sort field
  <,>     . Move sort field: '<' next col left; '>' next col right
  R,H     . Toggle: 'R' normal/reverse sort; 'H' show threads
  c,i,S   . Toggle: 'c' cmd name/line; 'i' idle tasks; 'S' cumulative time
  x,y     . Toggle highlights: 'x' sort field; 'y' running tasks
  z,b     . Toggle: 'z' color/mono; 'b' bold/reverse (only if 'x' or 'y')
  u       . Show specific user only
  n or #  . Set maximum tasks displayed

  k,r       Manipulate tasks: 'k' kill; 'r' renice
  d or s    Set update interval
  W         Write configuration file
  q         Quit
          ( commands shown with '.' require a visible task display window )
Press 'h' or '?' for help with Windows,
any other key to continue
```

Eines sysstat

El paquet sysstat és una col·lecció d'eines de monitorització de rendiment per a sistemes Linux. Proporciona dades instantànies de rendiment, que es poden emmagatzemar en arxius històrics. En entorns de servidor, aquestes dades proporcionen informació valuosa per detectar carències i colls d'ampolla del sistema.

L'ordre d'instal·lació és la següent:

```
1 # apt-get install sysstat
```

Algunes de les eines que inclou aquest paquet són:

- ***mpstat***: recull informació del rendiment de cadascun dels processadors del sistema. Permet fer un monitoratge continuat i admet modificadors.
- ***iostat***: és una eina més completa que l'anterior, ja que a més de presentar estadístiques de la CPU, dóna informació dels dispositius d'entrada i sortida, les particions i els sistemes d'arxius.
- ***pidstat***: informació estadística dels processos de Linux.
- ***SAR*** (system activity report): recull, enregistra i presenta informació sobre l'estat dels components principals del sistema (CPU, memòria, discs, interrupcions, interfícies de xarxa, consoles, *kernel*, etc.) i de la seva càrrega en temps real i de manera continuada. Emmagatzema les dades en els fitxers /var/log/saXX on XX és el dia del mes del monitoratge. Aquest fitxers permeten l'anàlisi a posteriori de la informació. *SAR* admet modificadors molt interessants com:
  - `-P` : dóna informació per a cada processador.
  - `-r` : dóna informació del rendiment de la memòria.
  - `-B` : dóna informació relativa a la paginació de la memòria.
  - `-W` : dóna informació relativa al swapping.
  - `-d` : dóna informació del rendiment de disc.
  - `-n` : dóna informació relativa a la xarxa.

Totes les ordres accepten com a paràmetre el nombre de mostres que volem capturar i l'interval entre mostres expressat en segons.

Aquest conjunt d'utilitats ens donen eines en l'àmbit de la monitorització de l'ús del processador i del seguiment de l'activitat dels processos. Vegem-ne un parell d'exemples:

Veure l'ús dels diferents processadors del sistema:

```
# mpstat -P ALL

Linux 2.6.32-28-generic (desktop) 20/02/12 _i686_ (1 CPU)

12:56:13 CPU usr %nice %sys %iowait %irq %soft %steal %guest %
    idle
12:56:13 all 2,51 1,19 14,45 0,42 0,27 0,10 0,00 0,00 81,05
```

Veure el percentatge d'ús dels processos més actius:

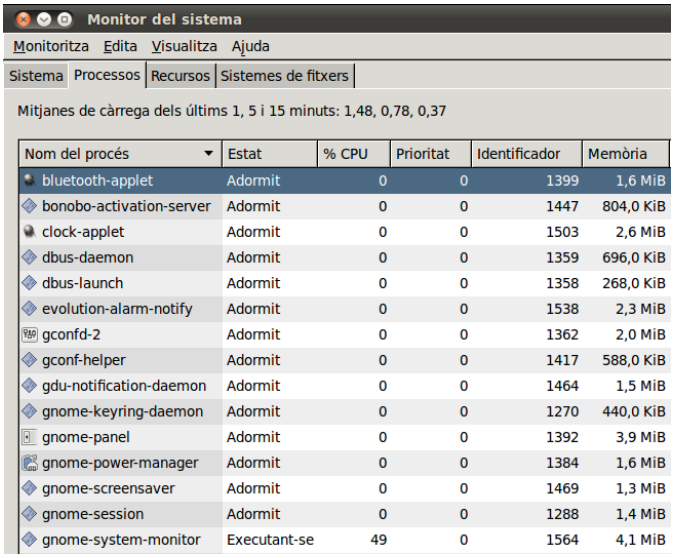
```
# pidstat 1 2

Linux 2.6.32-28-generic (desktop) 26/04/11 _i686_ (1 CPU)

13:00:09 PID %usr %system %guest %CPU CPU Command
13:00:10 984 0,00 1,72 0,00 1,72 0 Xorg
13:00:10 1357 0,00 0,86 0,00 0,86 0 wnck-applet
13:00:10 1868 0,00 3,45 0,00 3,45 0 firefox-bin
13:00:10 2508 2,59 5,17 0,00 7,76 0 pidstat

13:00:10 PID %usr %system %guest %CPU CPU Command
13:00:11 984 1,01 5,05 0,00 6,06 0 Xorg
13:00:11 1868 0,00 3,03 0,00 3,03 0 firefox-bin
```

```
13:00:11 2323 0,00 1,01 0,00 1,01 0 gnome-terminal
13:00:11 2508 0,00 9,09 0,00 9,09 0 pidstat

Average: PID %usr %system %guest %CPU CPU Command
Average: 984 0,47 3,26 0,00 3,72 - Xorg
Average: 1357 0,00 0,47 0,00 0,47 - wnck-applet
Average: 1868 0,00 3,26 0,00 3,26 - firefox-bin
Average: 2323 0,00 0,47 0,00 0,47 - gnome-terminal
Average: 2508 1,40 6,98 0,00 8,37 - pidstat
```

1.6.3 Entorn gràfic (monitor del sistema)

La mateixa eina gràfica **del monitor del sistema**, a més de la visualització en temps real dels recursos, disposa d'una pestanya on podem controlar els processos que s'estan executant i el consum de recursos del sistema que fan (vegeu la figura 1.18).

FIGURA 1.18. Finestra de gestió de processos al monitor de sistema



Per a cada procés actiu en el sistema mostra la informació següent:

- Nom del procés.
- L'estat en què es troba el procés (adormit, parat, en execució).
- El seu identificador (PID).
- El percentatge d'ocupació de processador que està utilitzant.
- La prioritat d'execució.
- La quantitat de memòria que fa servir.

Seleccionant un d'aquests processos podem efectuar diferents accions:

- Posar el procés en espera i rellançar-lo.

- Aturar-lo de manera ordenada o immediata.
- Canviar la seva prioritat d'execució.
- Visualitzar el seu mapa de memòria.
- Visualitzar els arxius oberts amb els quals treballa.

El monitor del sistema es pot configurar canviant l'interval de recollida de dades, els paràmetres de monitorització i el tipus de gràfic de presentació.

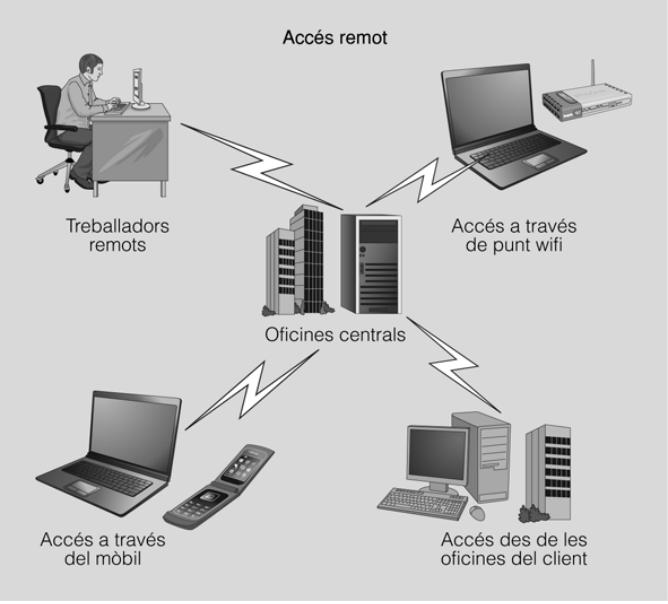
2. Serveis d'accés i administració remota

La generalització de les comunicacions entre ordinadors, tant en xarxa local com mitjançant Internet, ha introduït canvis dràstics en la forma d'accedir als equips i en la seva administració, permetent que pugui fer-se de forma remota. Tant si aquest accés es basa en la línia d'ordres, en una aplicació gràfica o en l'ús del navegador com a interfície gràfica, l'accés i l'administració remota faciliten a l'administrador la tasca de configuració i gestió del sistema informàtic, tenint sempre en compte la capacitat i ample de banda de la comunicació, i la necessària atenció a la seguretat i a la privacitat.

2.1 Introducció a l'accés remot a equips

La capacitat d'accedir remotament a arxius i informació en ordinadors a través d'Internet és interessant tant des del punt de vista de l'administració de sistemes com de l'execució i explotació de qualsevol tipus d'aplicació remota. Pot ser una eina útil per recuperar un arxiu oblidat a l'ordinador o per permetre a un administrador de sistemes modificar la configuració d'un servidor. També és utilitzat per les companyies per accedir a la informació comercial i administrativa del seus sistemes, ja sigui des de les oficines del client o des del domicili dels seus treballadors.

Figura 2.1. Accés remot des de diferents entorns



En un mateix sistema poden coexistir diferents tipus d'accés remot.

Aquest ús tan diversificat fa que hi hagi moltes tecnologies disponibles per

permetre aquest tipus d'accés: des del sistema de fitxers compartit incorporat en la majoria de sistemes operatius fins a eines més específiques desenvolupades per empreses.

L'ús tan divers fa que també hi hagi una gran diversitat de maneres i mitjans d'accedir remotament a informació, tal com podeu veure a la figura 2.1. Per fer front a aquesta diversitat hi ha nombrosos protocols i eines que donen accés remot des d'un portàtil amb connexió Wi-Fi, un telèfon mòbil UMTS o una connexió fixa d'Internet.

Wi-Fi

És l'acrònim de wireless fidelity (fidelitat sense fils) i estableix un conjunt d'estàndards de compatibilitat per a comunicacions en xarxes locals sense fils.

UMTS

És l'acrònim d'*universal mobile telecomunication system*. L'UMTS forma part dels sistemes de comunicacions mòbils de tercera generació (3G), que permeten una gran qualitat de veu, funcions multimèdia i un major ample de banda per Internet.

2.1.1 Usos més freqüents de l'accés remot a ordinadors

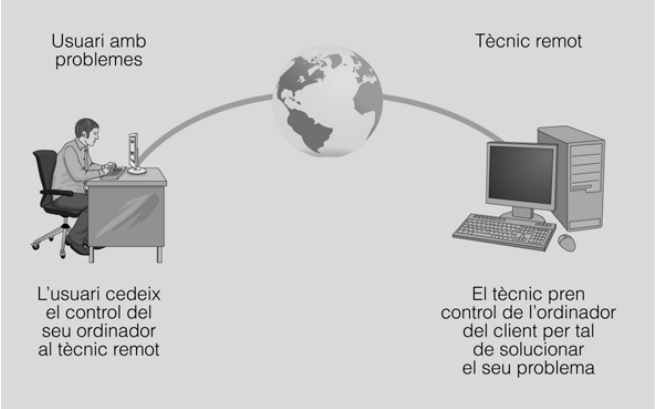
El programari que permet l'administració remota és cada vegada més comú i s'utilitza sovint quan és difícil o molt difícil estar físicament a prop d'un sistema per usar-lo o per tal d'accedir al material d'Internet que no està disponible en la mateixa ubicació.

Els servidors i altres equips de xarxa, per diverses raons, a vegades es distribueixen en distàncies considerables. Fins i tot quan estan relativament a prop, dins del mateix edifici o la mateixa planta, poden estar instal·lats en espais amb accés dificultós o restringit. Per aquestes raons l'administració remota, és a dir l'administració d'un equip des d'un altre equip, és una necessitat quotidiana.

A continuació mostrem una llista dels usos més freqüents de l'accés remot a ordinadors:

- **Administració de sistemes**: gestionar, administrar i configurar equips i servidors de forma remota en xarxa local o mitjançant Internet.
- **Suport i assistència tècnica de forma remota** (*helpdesk*): solucionar problemes tècnics a domicili sense necessitat de desplaçaments físics (vegeu la figura 2.2).
- **Treball a distància**: accés des del domicili als recursos de la xarxa de l'oficina (arxius, escriptori, correus, impressores, etc.), cosa que possibilita el teletreball.
- **Reunions i presentacions en línia**: compartir escriptori amb assistents situats en diferents llocs de treball. Inclou prestacions addicionals com videoconferència i xats de text i de veu.
- **Aplicacions d'ensenyament**: el professor pot monitoritzar les activitats escolars, compartir el seu escriptori i donar ajuda de forma remota.
- **Supervisió d'activitats**: tasques de treballadors, supervisió parental, etc.

FIGURA 2.2. Suport i assistència tècnica remota a usuaris



2.1.2 Tipus d'accés remot

Les tasques d'accés i administració remots es poden dur a terme amb mitjans diferents, però sempre estan condicionats per la xarxa que connecta l'equip que cal administrar amb l'estació on hi ha l'administrador.

La xarxa imposa condicions de:

1. **Capacitat**: si la connexió no té una amplada de banda suficient no serà pràctic treballar amb una interfície gràfica remota. Com més augmenti el retard a la xarxa, més frustrant serà el treball interactiu.
2. **Seguretat**: és obvi que la comunicació entre l'administrador i l'equip remot no ha de ser interceptada per altres usuaris de la xarxa.

Els diferents mitjans d'administració remota es poden agrupar en tres categories bàsiques:

1. Sessió de treball a la consola
2. Sessió de treball amb interfície gràfica
3. Client o eina d'administració local

Cada categoria té els seus punts forts i febles. Una **sessió de treball a la consola** mitjançant **SSH** o l'obsolet i insegur **Telnet**no exigeix grans capacitats a la xarxa. És el mètode més lleuger i fins i tot es pot utilitzar amb comoditat per administrar màquines molt distants o amb xarxes de capacitat escassa. A més, és relativament senzill automatitzar tasques mitjançant guions de programació per repetir les mateixes operacions en un conjunt de màquines.

SSH (*secure shell*) és el nom d'un protocol i del programa que l'implementa, i serveix per accedir a màquines remotes a través de la xarxa. Permet gestionar completament l'ordinador mitjançant l'intèrpret d'ordres.

TELNET

És un protocol que emula un terminal remot per connectar-se a una màquina multiusuari. El seu principal problema és la seguretat.

LAN

Una LAN (de l'anglès local area network o xarxa d'àrea local) és un tipus de xarxa informàtica caracteritzada pel seu caràcter "local" o de distància curta, com ara una casa, una oficina, un hotel, etc., és a dir, la seva extensió està limitada a uns 200 metres i podria arribar a un quilòmetre si es fessin servir repetidors.

L'administració remota amb **interfície gràfica** permet obrir a l'estació local finestres d'aplicacions que s'executen en el servidor remot. Fins i tot permet veure l'escriptori complet de l'estació remota. Tot i que pot ser un mètode de treball còmode, requereix capacitats de la xarxa que normalment només es troben disponibles dins d'una LAN.

Emprar una **eina d'administració local** feta a mida, com un assistent per configurar una impressora remota, o genèrica, com un navegador web, intenta conjugar punts forts de les dues tècniques anteriors. En aquest cas, l'administrador dialoga amb una aplicació local o una pàgina web i no necessita recordar ordres. A més, com que la representació gràfica es produeix localment, no s'exigeix gran capacitat a la xarxa. El problema de les eines específiques és que es tracta d'un programari que cal instal·lar a cada estació que l'administrador vulgui accedir. Sovint aquesta eina només està disponible per a un sistema operatiu concret o fins i tot per a una de les seves versions.

Aquest problema s'obvia amb les interfícies web, ja que un navegador web és un programari comú present en tots els equips. A la taula 2.1 podeu veure un resum dels principals avantatges i inconvenients dels tres tipus d'administració remota.

TAULA 2.1. Els tres principals mitjans d'administració remota

Mètode d'administració	Avantatges	Inconvenients
Treball a la línia d'ordres	Requisits mínims per a la xarxa. És flexible i es pot automatitzar.	Cal conèixer la sintaxi i recordar les ordres.
Interfície gràfica	Visual i flexible. No cal recordar ordres.	Imposa requisits de capacitat a la xarxa.
Client local	Visual i flexible. No cal recordar ordres. No consumeix gaires recursos de la xarxa.	Si no es tracta d'una interfície web (navegador), cal instal·lar programari.

2.2 Administració remota basada en la línia d'ordres

Les connexions remotes mitjançant la línia d'ordres són l'opció que menys capacitats exigeix a la xarxa i per tant es poden fer servir fins i tot en els casos en el qual el canal de comunicació no té una gran amplada de banda disponible o quan el retard és important. Exigeixen conèixer la sintaxi pròpia de l'intèrpret d'ordres emprat i de les seves eines, però són un mecanisme molt flexible que permet fer moltes automatitzacions.

2.2.1 Telnet

Les eines principals per iniciar una sessió de treball remota amb una línia d'ordres són **Telnet** i **SSH**. Totes dues permeten treballar com si s'estigués davant del teclat i de la pantalla de l'estació remota i estan basades en una arquitectura

client-servidor, però les diferències entre elles són importants. **Telnet** encara s'utilitza per raons històriques: és possible que trobem commutadors o servidors d'impressió als quals podem accedir remotament per fer tasques d'administració. Però el fet que la informació es transmeti com a text clar suposa un problema greu de seguretat. Avui dia el seu ús està completament desaconsellat i s'exigeix prendre mesures addicionals.

Protocol telnet

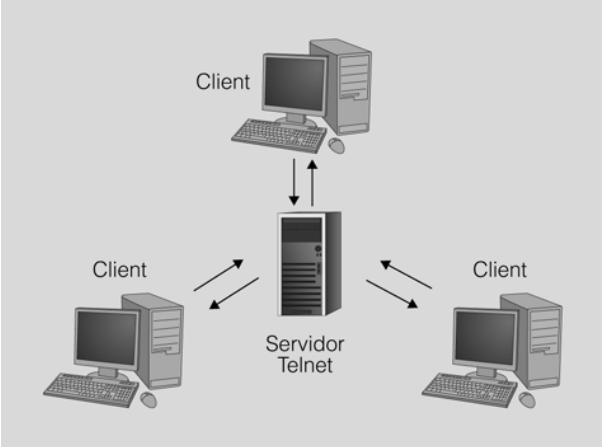
Normalment, el servidor telnet acostuma a esperar les connexions dels clients al port TCP 23. És un protocol antic, desenvolupat el 1969, la seguretat del qual pot veure's compromesa amb qualsevol programari captador de paquets (packet sniffer).

La falta de mesures de seguretat modernes impedeix emprar Telnet de manera generalitzada, però encara és una eina molt eficaç per connectar clients amb servidors que treballen amb text clar (com els servidors HTTP i SMTP) per diagnosticar problemes (figura 2.3).

HTTP

El protocol de transferència d'hipertext o HTTP (hypertext transfer protocol) estableix el protocol per a l'intercanvi de documents d'hipertext i multimèdia al web.

Figura 2.3. Arquitectura client-servidor



El servei Telnet utilitza una arquitectura client-servidor. Els clients es connectaran remotament al servidor Telnet.

Instal·lació de Telnet

En la major part de distribucions GNU/Linux, el client Telnet està instal·lat al sistema de manera predeterminada, però en cas que no ho sigui es pot instal·lar amb la instrucció següent des de l'intèrpret d'ordres:

```
1 root@ioc-Client:~# apt-get install telnet
```

Normalment, utilitzareu l'eina Telnet per connectar-vos a altres serveis de xarxa (com HTTP o SMTP), però també podeu instal·lar un servidor Telnet a la vostra màquina i utilitzar-lo per fer connexions remotes. Recordeu, però, que el protocol que utilitza Telnet no és segur i que per tant la vostra connexió estarà en perill de ser interceptada per tercers. Si, per exemple, escriviu alguna contrasenya en una connexió Telnet, podrà ser interceptada amb facilitat.

SMTP

SMTP és l'acrònim de simple mail transfer protocol, és a dir, protocol simple de transferència de correu, i és un protocol de xarxa basat en text utilitzat per a l'intercanvi de missatges de correu electrònic.

La instal·lació del servidor Telnet es pot fer executant la instrucció següent des de l'intèrpret d'ordres:

S'acostuma a anomenar els dimonis amb el nom del servei acabat amb la lletra d. Així, *telnetd* és el dimoni que gestiona el servei Telnet.

```
1 root@ioc-Server:~# apt-get install telnetd
```

Accés remot mitjançant Telnet

Un cop instal·lat el servidor Telnet, ja hi podem accedir des de qualsevol màquina que tingui el client instal·lat. La sintaxi bàsica de Telnet és la següent:

```
telnet <hostname><port>
```

- `<hostname>` representa l'adreça IP o el nom del servidor al qual ens volem connectar.
- `<port>` és el número de port del servei al qual ens volem connectar. De manera predeterminada és el port 23, que és el port utilitzat per Telnet.

A la taula 2.2 podeu veure un llistat d'alguns dels ports TCP més utilitzats habitualment.

TCP

El protocol de control de les transmissions (transmission control protocol) és un protocol orientat a la connexió dintre del nivell de transport del model OSI que permet el lliurament de paquets, en el cas de TCP anomenats segments, de manera fiable.

TAULA 2.2. Ports TCP

Número de port	Servei
21	FTP (*file transfer protocol*): sistema per transferir fitxers.
22	SSH: protocol i programa que l'implementa. Serveix per accedir a màquines remotes a través de la xarxa de manera segura.
23	Servidor Telnet.
25	SMTP: protocol de xarxa basat en text utilitzat per a l'intercanvi de missatges de correu electrònic.
53	DNS (*domain name server*): servei utilitzat per "traduir" les adreces IP a un nom de domini.
80	HTTP: protocol per a l'intercanvi de documents d'hipertext i multimèdia en el web. És el port utilitzat de manera predeterminada pels servidors de llocs web.

Per exemple, si us voleu connectar a un equip que té una IP 192.168.65.10:

```
jmunoz@ioc-Client:~$ **telnet 192.168.56.10**
Trying 192.168.56.10...
Connected to 192.168.56.10.
Escape character is '^]'.
Debian GNU/Linux 6.0

ioc-Server login: **jmunoz**
Password:

Last login: Wed Oct 26 11:28:51 CEST 2011 from ioc-Client.local on pts/1
Linux ioc-Server 2.6.32-5-686 #1 SMP Mon Oct 3 04:15:24 UTC 2011 i686

The programs included with the Debian GNU/Linux system are free software; the
    exact distribution terms for each program are described in the individual
    files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
    applicable law.
You have new mail.

jmunoz@ioc-Server:~$
```

Com podeu veure, després de rebre un seguit de dades posem el cursor a la línia d'ordres i podem fer qualsevol tasca com si fóssim a l'equip remot.

Una manera diferent de connectar-nos és obrint una sessió de Telnet directament. Apareixerà l'indicador (*prompt*) *telnet>* i llavors es pot utilitzar la instrucció *open* per obrir la connexió.

```
jmunoz@ioc-Client:~$ **telnet**
telnet> **open 192.168.56.10**
```

Podeu trobar les instruccions més habituals de Telnet a la taula 2.3.

Taula 2.3. Algunes instruccions de Telnet

Instrucció	Servei
c - close	Tancar la connexió.
d - display	Mostrar els paràmetres configurats en la connexió.
o - open hostname [port]	Obrir una connexió en un ordinador remot. El paràmetre *port* és opcional.
q - quit	Sortir de Telnet.
?/h - help	Mostrar l'ajuda de Telnet. Es mostren totes les instruccions disponibles.

2.2.2 Introducció a SSH

SSH és un protocol de xarxa que permet l'intercanvi d'informació de manera segura. Utilitza xifrat i criptografia de clau pública per tal de fer l'autenticació de l'estació remota.

Una de les seves funcions més emprades és iniciar una sessió remota per tal d'executar ordres en substitució de Telnet. Però les seves capacitats són més àmplies, ja que també permet:

- Transmetre fitxers de manera segura.
- Fer còpies de seguretat de manera eficient i segura en combinació amb l'ordre *rsync*.
- Fer túnels per assegurar qualsevol servei que no es transmeti encriptat (HTTP, SMTP, VNC, etc.) o per travessar tallafocs que estiguin bloquejant el protocol.
- Reenviament automàtic de sessions X11 des d'un *host* remot (disposa d'aquesta funció openSSH però no altres implementacions d'SSH).
- Navegar pel web a través d'una connexió amb un servidor intermediari (*proxy)* xifrada amb programes clients que siguin compatibles amb el protocol SOCKS.

- Seguiment automatitzat i administració remota de servidors a través d'un o més dels mecanismes exposats anteriorment.
- Fent servir SSHFS (SSH File System), un sistema d'arxius basat en SSH que pot crear de manera segura un directori en un servidor remot i actuar com a sistema de fitxers en xarxa.

SSG file system

SSHFS és un sistema de fitxers en xarxa que fa servir SSH per comunicar els diferents equips.

Les seves possibilitats es poden combinar de moltes maneres diferents. Ateses les seves característiques criptogràfiques, SSH és una eina fonamental per a l'administrador de xarxa. De les capacitats i funcions esmentades, les més utilitzades són les d'inici de sessió remota, la transferència d'arxius i la tunelització d'altres serveis mitjançant redreçament estàtic de ports o bé establint un servei SOCKS amb el redreçament dinàmic de ports.

2.2.3 Instal·lació d'SSH

SSH utilitza una arquitectura client-servidor, en què el client es connecta a una màquina remota, el servidor. En la major part de distribucions GNU/Linux el client ja hi és però, si es vol accedir a una màquina de manera remota, en aquesta màquina hi haurà d'haver el servidor SSH instal·lat.

La implementació més popular d'SSH és la desenvolupada per la fundació OpenSSH, el servidor openSSH-server. Per procedir a instal·lar-la, executem l'ordre següent des de la consola de la màquina a la qual ens connectarem (servidor).

```
1 root@ioc-Server:~# apt-get install openSSH-server
```

Si no hi ha hagut cap error durant la instal·lació podreu veure un missatge en què es creen les claus necessàries per assegurar les comunicacions segures mitjançant l'encriptació.

```
1 S'està configurant openssh-server (1:5.5p1-6+squeeze1)...
2 Creating SSH2 RSA key; this may take some time ...
3 Creating SSH2 DSA key; this may take some time ...
4 Restarting OpenBSD Secure Shell server: sshd.
5 root@ioc-Server:~#
```

L'última part de la configuració bàsica es fa de manera automàtica quan intentem connectar un client per primera vegada. És la generació automàtica de la clau compartida que utilitzaran client i servidor per assegurar que les comunicacions són segures. Un cop s'ha comunicat la clau compartida entre totes dues estacions, el missatge s'encripta de forma convencional.

Arxius de configuració del client SSH

El client d'OpenSSH es pot configurar de manera prou flexible com perquè l'administrador pugui definir una configuració general per a tot el sistema, perquè

cada usuari pugui modificar els paràmetres adients per a les seves connexions o perquè pugui especificar opcions determinades per a cada connexió individual.

Configuració del client SSH

La configuració del client d'OpenSSH és força flexible. Es pot consultar la documentació oficial al manual SSH_config(5).

El client d'OpenSSH farà servir:

- Les opcions indicades a la línia d'ordres
- Els valors especificats en el fitxer de configuració de l'usuari: **$HOME/.ssh/ssh_config**
- Els valors especificats en la configuració per a tot el sistema: **/etc/ssh/ssh_config**

$HOME

Recordeu que HOME és una variable d'entorn de Linux que conté el camí absolut del directori personal de l'usuari actiu.

Per a cada paràmetre, el client farà servir el primer valor trobat. És a dir, si s'especifica un paràmetre a la línia d'ordres no se'n consultarà el valor en els fitxers de configuració. Dins dels fitxers de configuració és possible definir seccions per a diferents equips (mitjançant la paraula reservada ***host***). Les línies buides i les que comencen amb un # (comentari) seran ignorades. A la taula 2.4 podeu trobar una llista amb les opcions més bàsiques de configuració d'un client.

Taula 2.4. Algunes de les opcions més bàsiques de la configuració d'un client

Opció	Funció
Host <patró>	Permet especificar opcions que només s'aplicaran a les connexions amb l'amfitrió indicat. L'amfitrió s'indica mitjançant patrons amb els caràcters * i ?. Especifica el port de destinació per a la connexió, que per defecte és el 22.
CheckHostIP <yes|no>	El seu valor predeterminat és *yes*. Si l'opció està activada, es comprovarà l'adreça de l'estació remota mitjançant el fitxer *known_hosts* per tal d'advertir un possible enverinament de DNS.
Cipher i Chipers	Permeten especificar respectivament l'algorisme d'encriptació per les a connexions SSH1 i la precedència d'algorismes que cal emprar en les connexions SSH2.
Compression <yes|no>	Si la connexió és molt lenta, la compressió pot millorar els resultats. Si la xarxa té prou amplada de banda normalment no es recomana.
Port <port>	Especifica el port de destinació per a la connexió, que de manera predeterminada és el 22.
RekeyLimit <limit>	Especifica el volum màxim d'informació que es pot transmetre abans d'haver de renegociar la clau de sessió. Es poden fer servir els sufixos K, M o G.
User <usuari>	Especifica l'usuari per establir la connexió a l'estació remota.
SendEnv <variables>	Permet enviar el valor de les variables d'entorn especificades a l'estació remota.

A continuació podeu veure un exemple d'un arxiu `$HOME/.ssh/ssh_config`, corresponent a la configuració d'un client SSH.

```
1 Host 192.168.56.10
2 Ciphers aes128-cbc
3 Compression yes
4 User usuariSSH
```

```
5 Port 30
```

En aquest cas, el fitxer indica que, quan establim una connexió amb el servidor amb la IP 192.56.10, s'ha d'utilitzar un algoritme d'encriptació del tipus aes128-cbc i s'han de comprimir les dades que s'envien. A més, la connexió es farà utilitzant l'usuari *usuariSSH* i s'establirà la connexió amb el port 30 del servidor. Observeu que, perquè la connexió funcioni, el servidor SSH, al seu torn, haurà d'estar configurat per treballar al port 30.

Arxius de configuració del servidor SSH

Configuració del servidor SSH

Es pot trobar informació detallada sobre l'ús del servidor OpenSSH a les pàgines del manual del mateix servidor SSHd(8) i a la dedicada al seu fitxer de configuració SSHd_config(5).

La funció del servidor SSH és esperar les connexions dels clients (normalment al port TCP 22), dur a terme la seva autenticació i, si tot ha anat bé, obrir una sessió de treball, executar una ordre o bé redreçar ports.

Cada vegada que es rep un intent de connexió des d'un client, es fan en primer lloc totes les comprovacions i inicialitzacions criptogràfiques per garantir la seguretat. Després es tracta d'autenticar l'usuari i finalment se segueixen els passos d'un procés d'inici de sessió habitual.

Fitxer de configuració sshd_config

Malgrat que en el funcionament del servidor d'OpenSSH hi intervenen diversos fitxers, l'arxiu principal de configuració és **/etc/ssh/sshd_config**. Aquest fitxer conté diferents paraules clau amb el seu valor. Les línies que comencen amb **#** (comentaris) o les que estan en blanc són ignorades.

A la taula 2.5 podeu trobar els principals paràmetres de configuració d'un servidor SSH.

TAULA 2.5. Paràmetres de configuració del servidor SSH

Opció	Funció
AllowGroups	Si s'especifica seguida d'una llista de grups (separats per espais), només els usuaris que tenen algun dels grups indicats com a grup principal o suplementari podran iniciar sessió. És possible emprar els patrons *?* i * en la definició dels grups. Només es poden indicar els grups mitjançant el seu nom, no en format numèric (GID).
AllowUsers	Té la mateixa funció que *AllowGroups*, però per als usuaris. En aquest cas, a més, és possible indicar des de quins amfitrions d'origen s'acceptarà la connexió. Per exemple: *AllowUsers usuari1@192* usuari2*.
Banner <fitxer>	Envia el contingut del fitxer indicat al client abans de dur a terme l'autenticació.
Compression <yes|delayed|no>	Especifica si es farà servir la compressió. El valor *delayed*, l'opció predeterminada, indica que només es farà servir la compressió un cop s'hagi autenticat l'usuari.
DenyGroups	Permet especificar una llista de grups, separats per espais, als quals no es permetrà iniciar sessió. Es poden especificar els grups mitjançant el seu nom, emprant els patrons *?* i * de manera opcional.

TAULA 2.5 (continuació)

Opció	Funció
DenyUsers	Igual que *DenyGroups*, però per als usuaris. En aquest cas és possible indicar un equip (o subxarxa) per a cada usuari.
Port ListenAddress	Permeten especificar el port on el servidor escoltarà les connexions dels clients i les adreces on obrirà aquest port. De manera predeterminada s'utilitza el port 22 de qualsevol adreça local. És possible especificar múltiples vegades aquestes opcions, però convé que *Port* sempre aparegui abans que *ListenAddress*.
LoginGraceTime	Període de temps màxim per dur a terme l'autenticació. El valor 0 expressa que no hi ha límit.
MaxAuthTries	Nombre màxim d'intents d'autenticació que es poden fer.
MaxStartups	Nombre màxim de connexions simultànies que encara no han completat la seva autenticació.
PasswordsAuthenticati on <yes|no>	Indica si s'accepta l'autenticació mitjançant contrasenya (*password).*
PermitEmptyPasswords <no|yes>	Si s'utilitza l'autenticació mitjançant contrasenya, especifica si el servidor permet la connexió a comptes que tenen una contrasenya buida.
PermitRootLogin <yes|without-password|forced-commands-only|no>	Especifica si s'accepta la connexió de superusuari mitjançant SSH. El valor *without-password* indica que el superusuari no podrà fer servir l'autenticació basada en contrasenya i el valor *forced-commands-only*, que només es permetrà l'autenticació de clau pública per executar certes ordres de manera remota (normalment per fer còpies de seguretat).
Protocol	Especifica quins protocols es podran fer servir en les connexions dels clients (1, 2 o tots dos). És important recordar que el protocol SSH2 és força més segur que l'SSH1.
PubkeyAuthentication <yes|no>	Especifica si s'acceptarà l'autenticació de clau pública.
X11Forwarding <no|yes>	Especifica si s'acceptarà el reenviament X11 per tal que les aplicacions gràfiques executades en el servidor obrin la seva finestra en el servidor X del client.

Altres arxius de configuració

Hi ha altres arxius de configuració que permeten de forma genèrica el filtratge, control d'accés i mecanismes de protecció de diferents serveis (POP, Sendmail, Telnet, SSH, etc.) actuant de fet com un tallafocs bàsic.

Així, si volem habilitar o restringir l'accés a determinats equips i serveis podem editar els arxius de configuració **/etc/hosts.deny** i **/etc/hosts.allow** indicant en la directiva dintre de l'arxiu el servei que volem controlar, en aquest cas, el dimoni SSH. D'aquesta manera el sistema, davant d'una petició d'accés al servei, fa la cerca següent, que conclou en el moment de la primera coincidència:

1. Comprova l'arxiu /etc/hosts.allow. Si hi troba coincidència valida l'accés.
2. Comprova l'arxiu /etc/hosts/deny. Si hi troba coincidència no valida l'accés.

3. En cas de no trobar coincidència en cap dels arxius valida l'accés.

Exemples de directives d'aquests arxius:

1. sshd: ALL (permet/denega l'accés ssh a tothom)
2. sshd: 192.168.56.10 (permet/denega l'accés SSH de la IP 192.168.56.10)

Recordeu que perquè qualsevol canvi tingui efecte s'ha de reiniciar el servei:

```
/etc/init.d/ssh restart
```

Per a informació completa sobre els arxius de configuració `/etc/hosts.allow` i `/etc/hosts.deny` consulteu la informació del sistema amb l'ordre: *man hosts_access*.

2.2.4 Connexió a una estació remota amb SSH

La funció més comuna per a SSH és establir una sessió de treball remota fent ús de tècniques criptogràfiques per transmetre la informació. L'ús del client SSH és força senzill.

```
ssh user@host
```

- *user*: és l'usuari que es connectarà a la màquina remota.
- *host*: representa la IP o el nom de domini del servidor SSH al qual ens volem connectar.

És tot el que hem d'escriure per iniciar una sessió remota com a usuari (*user*) en l'equip amfitrió (*host*). En executar l'ordre ens demanarà la contrasenya de l'equip remot i, si l'escrivim de manera correcta, podrem accedir a la sessió de treball remota per escriure ordres.

```
jmunoz@ioc-Client:~$ ssh jmunoz@192.168.56.10
jmunoz@192.168.56.10's password:

Linux ioc-Server 2.6.32-5-686 #1 SMP Mon Oct 3 04:15:24 UTC 2011 i686
The programs included with the Debian GNU/Linux system are free software; the
    exact distribution terms for each program are described in the individual
    files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
    applicable law.

No mail.

Last login: Wed Oct 26 12:20:51 2011 from ioc-client.local

jmunoz@ioc-Server:~$
```

Per finalitzar la connexió escriurem *exit*.

```
jmunoz@ioc–Server:~$ exit
logout
Connection to 192.168.56.10 closed.
jmunoz@ioc–Client:~$
```

Si no s'especifica el nom d'usuari a l'hora d'invocar l'ordre SSH, intentarà fer la connexió amb l'usuari amb el qual estem connectats al terminal de GNU/Linux.

```
usuari1@ioc–Client$ ssh 192.168.56.10
usuari1@192.168.56.10's password:
```

Fixeu-vos que en aquest cas intenta connectar-se com a *usuari1*, ja que no s'ha especificat amb quin usuari s'ha de connectar.

El client d'**OpenSSH** disposa de diferents opcions que es detallen en el seu manual. Algunes de les més freqüents són les descrites a la taula 2.6.

Taula 2.6. Opcions més freqüents del client OpenSSH

Opció	Funció
-1	Força l'ús de la versió 1 del protocol SSH. Només es recomana emprar SSH1 per connectar-se a servidors antics que no són compatibles amb SSH2.
-2	Força l'ús de la versió 2 del protocol: SSH2.
-4	Força l'ús de l'adreçament IPv4.
-6	Força l'ús de l'adreçament Ipv6.
-C	Activa la compressió *gzip* en la connexió. Es recomana activar la compressió si s'està emprant SSH amb un enllaç lent, com un mòdem. Si l'enllaç és de banda ampla es recomana treballar sense compressió.
-p port	Port al qual es connectarà en l'equip remot. De manera predeterminada el servidor SSH s'executa al port TCP 22, però si es tracta d'un servidor accessible des d'Internet és recomanable escollir un altre port per evitar els intents de connexió.
-q	No imprimeix els missatges d'advertència, només els errors. Amb una altra *-q* no imprimeix ni els errors.
-X	Activa la retransmissió X11 per tal que els programes gràfics llançats en l'estació remota obrin la seva interfície gràfica en el servidor X local.
-x	Desactiva la retransmissió X11.

Algunes vegades només es desitja executar una ordre a l'estació remota, no obrir un ***shell*** (intèrpret d'ordres) per treballar. En aquest cas, és possible indicar l'ordre que cal executar en la mateixa crida de SSH.

tail

És una instrucció de GNU/Linux que permet veure les 10 últimes línies. El paràmetre *n* permet especificar el nombre de línies que es vol mostrar. En aquest cas, se'n mostren tres.

Per exemple, per tal de veure el final del fitxer de registre /var/log/messages al servidor 192.168.56.10 farem:

```
jmunoz@ioc–Client:~$ **ssh jmunoz@192.168.56.10 tail /var/log/messages**
jmunoz@192.168.56.10's password:
```

2.2.5 Transferència d'arxius entre equips

Entre les utilitats incloses en la distribució d'**OpenSSH** trobem les ordres ***scp*** i ***sftp***. Aquestes ordres permeten transferir fitxers amb totes les garanties de seguretat d'SSH.

Còpies segures (scp)

La sintaxi bàsica de la instrucció *secure copy* (*scp)* és:

```
scp [opcions] [[user@]host1:]fitxer1 [[user@]host2:]fitxer2
```

- host1: representa la màquina d'origen.
- host2: representa l'estació de destinació.

L'ordre ***scp*** es pot veure com una versió estesa de l'ordre ***cp***, que permet copiar fitxers fins i tot entre màquines diferents. De fet, és el substitut d'SSH per l'ordre ***rcp***, que és antiga i insegura. En fer una còpia mitjançant ***scp***, es pot escollir qualsevol combinació de fitxers locals o remots tant per l'origen com per la destinació.

remote copy

rcp és una instrucció GNU/Linux que permet copiar fitxers entre l'equip local i un de remot.

En l'exemple que podeu veure a continuació copiem el fitxer local manual.pdf en una estació remota amb IP 192.168.56.10.

```
jmunoz@ioc–Client:~$ **scp manual.pdf 192.168.56.10:**
jmunoz@192.168.56.10's password:
manual.pdf 100% 179KB 179.5KB/s 00:00
```

Si la contrasenya és correcta es procedirà a la transferència de l'arxiu i s'indicarà de manera interactiva el percentatge que ja s'ha enviat, la grandària en quilobytes, l'índex de transferència i el temps transcorregut des del començament de la transmissió.

No només es poden copiar fitxers locals en una estació remota, sinó que es poden copiar fitxers de l'estació remota en la local i fins i tot entre dues estacions remotes. A la taula 2.7 podeu trobar un exemple de cada tipus.

TAULA 2.7. Exemples de diferents usos de la instrucció scp

Exemple	Ordre
Copiar fitxer local a equip remot	scp fitxer usuari@192.168.56.10:/home/usuari
Copiar fitxer remot a equip local	scp usuari@192.168.56.10:/home/usuari/fitxer ./
Copiar un fitxer d'un equip remot a un altre equip remot	scp usuari@192.168.56.12:/home/usuari/fitxer usuari@192.168.56.11:/home/usuari

La pàgina del manual d'***scp*** ens mostrarà els seus paràmetres, la major part dels quals són els mateixos que té l'ordre ***ssh***. No obstant això, té algunes opcions

pròpies que són particularment útils i que podeu veure a la taula 2.8.

Taula 2.8. Opcions de la instrucció scp

Opció	Funció
-l limit	Establir un límit a l'amplada de banda en kbps.
-p	Preservar el temps de modificació, accés i els permisos dels fitxers copiats.
-r	Fer una còpia recursiva pels directoris.

Transferències segures de fitxers (sfpt)

L'ordre ***sftp*** és un substitut per al client d'FTP, que és el tradicional però que és insegur. En emprar *sftp* s'estableix una connexió al sistema remot i després, de manera interactiva, es podran indicar ordres per explorar el sistema d'arxius remot i fer modificacions.

El protocol de transferència de fitxers o FTP (de l'anglès *file transfer protocol*) és un programari estandarditzat per enviar fitxers entre ordinadors amb qualsevol sistema operatiu. Forma part de la capa d'aplicació del model TCP/IP.

La sintaxi bàsica de la instrucció *sftp* és:

```
1 sftp [opcions][[user@]host1:]fitxer1 [[user@]host2:]fitxer2
```

- host1: representa la màquina d'origen.
- host2: representa l'estació de destinació.

Un cop establerta la connexió segura, podem executar les mateixes ordres que si ens haguéssim connectat mitjançant FTP. A la taula 2.9 podeu trobar les ordres que *sftp* pot interpretar.

Taula 2.9. Ordres que poden ser interpretades per sftp

Ordre	Funció
bye, exit, quit	Finalitzar la sessió.
cd camí	Canviar el directori remot.
chgrp, chown, chmod	Canviar el grup, el propietari o els permisos en el sistema remot .
get fitxer	Transmetre el fitxer remot indicat al directori de treball local.
put fitxer	Transmetre el fitxer local indicat al directori de treball remot.
ls, mkdir, pwd, rename, rm, rmdir, ln	Elaborar llistats, crear directoris, consultar la ruta, canviar de nom, eliminar fitxers i directoris i crear enllaços simbòlics al sistema de fitxers remot.
lcd, lls, lmkdir, lpwd	Versions de les ordres *cd*, *ls*, *mkdir* i *pwd* per treballar al sistema de fitxers local.

Vegem un exemple de com fer transferències segures entre l'ordinador local i un equip amb la IP 192.168.100.10:

```
jmunoz@ioc-Client:~$ **sftp jmunoz@192.168.56.10**
jmunoz@192.168.56.10's password:
Connected to 192.168.56.10.
sftp>
```

En aquest punt ja s'ha establert la connexió segura amb l'equip remot i ja es poden començar a utilitzar les ordres pròpies d'*sftp*. En l'exemple següent es transferirà el fitxer fitxer.pdf de l'equip remot al local.

```
sftp> **cd Documents/**
sftp> **get manual.pdf **
Fetching /home/jmunoz/Documents/manual.pdf to manual.pdf
/home/jmunoz/Documents/manual.pdf 100% 179KB 179.5KB/s 00:00
sftp>
```

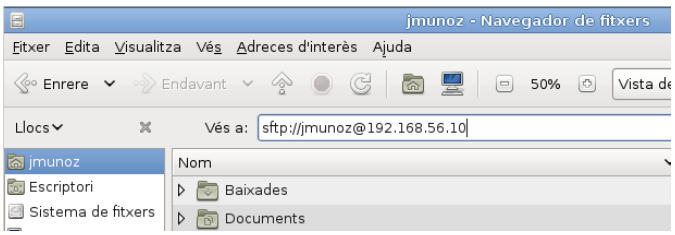
sftp a l'escriptori GNU/Linux

Els escriptoris **GNOME** i **KDE** també proporcionen un accés senzill a servidors SSH per tal de treballar en xarxa. A GNOME, el navegador d'arxius **Nautilus** pot mostrar directoris remots com si fossin locals fent servir el GNOME VFS (sistema de fitxers virtual de GNOME).

Com que la distribució Debian utilitza GNOME per defecte, per tal d'utilitzar gràficament la instrucció *sftp* al visualitzador d'arxius Nautilus, només és necessari introduir a la barra de lloc la mateixa ordre que s'utilitza per obrir una connexió SSH mitjançant ***Vés > Ubicació*** (o bé *Ctrl+L*). Vegeu la figura 2.4.

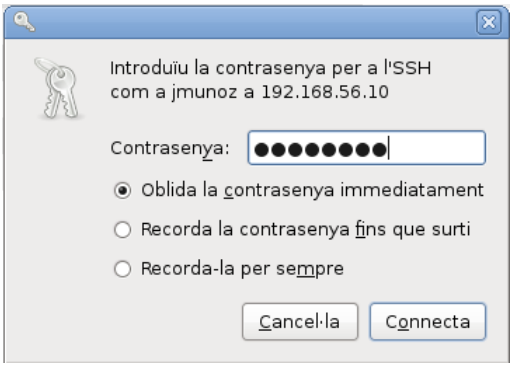
```
sftp://nomUsuari@ipServidor
```

FIGURA 2.4. Introduir a la barra d'adreces l'ordre de connexió ssh



Després d'uns segons per establir la comunicació, el navegador d'arxius Nautilus detecta que es vol fer una connexió segura i demana la contrasenya de l'usuari de l'equip remot amb una finestra similar a la de la figura 2.5.

FIGURA 2.5. Nautilus demana la contrasenya per connectar-se a l'equip remot



Si la contrasenya és correcta, s'estableix la connexió remota i Nautilus passa al mode de transferència segura d'arxius i la unitat remota queda muntada. A la figura 2.6 es pot observar que es visualitzen els arxius de l'equip remot.

FIGURA 2.6. Nautilus mostra la unitat remota per fer la transferència segura d'arxius

Utilitzant aquesta finestra, sempre que es tinguin els permisos necessaris, es poden copiar arxius, crear carpetes o fer qualsevol altra operació típica d'un navegador de fitxers.

2.2.6 Redreçament de ports TCP i túnels amb SSH

La major part dels protocols que utilitzem en les nostres comunicacions estan basats en dissenys de fa gairebé 30 anys, quan la seguretat en xarxes telemàtiques no era un problema. Telnet, FTP, POP3, protocols d'ús quotidià, descuiden la seguretat i confidencialitat de les dades que envien. No serveix de res protegir els servidors, implantar una bona política de contrasenyes i actualitzar les versions dels nostres dimonis si després, quan un usuari de POP3, per exemple, vol veure el seu correu electrònic des de la nostra xarxa, envia el seu usuari i contrasenya en text pla per la xarxa.

Fent servir SSH es poden establir túnels encriptats pels quals es pot transmetre qualsevol protocol que faci servir TCP. Així és possible, per exemple, emprar un túnel SSH per:

- descarregar el correu mitjançant POP3 i enviar-lo fent servir SMTP.

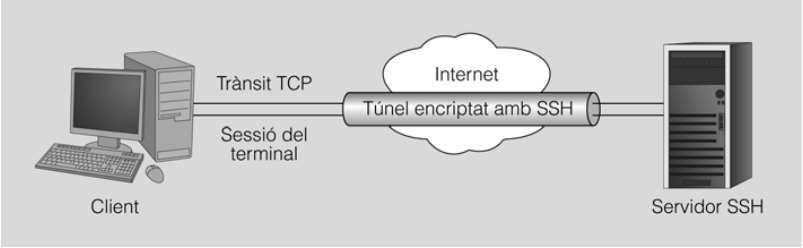
- accedir a un servidor web mitjançant HTTP.

- accedir a un sistema d'arxius remot mitjançant NFS.

Els protocols de l'exemple (POP3, SMTP, HTTP i NFS) no utilitzen tècniques criptogràfiques. El seu ús en una xarxa que no és de confiança no ens permet garantir la confidencialitat. Però com que tots aquests serveis utilitzen TCP com a protocol de transport, poden ser tunelitzats per SSH.

La idea en què es basa aquest procediment, i que es representa a la figura 2.7, és la de fer un túnel pel qual viatjaran les dades de manera segura (*tunneling*). A cada un dels extrems del túnel hi ha les aplicacions estàndard (un dimoni POP3 estàndard, el nostre client de correu preferit, FTP, un navegador web, etc.) i la comunicació s'assegura fent ús de tota la potència criptogràfica d'SSH. SSH recull les dades que el client vol enviar i les reenvia pel túnel o canal segur. A l'altre costat del túnel es recullen les dades i es tornen a enviar al servidor corresponent.

FIGURA 2.7. Túnel fet amb SSH per fer segures les comunicacions que utilitzen el protocol TCP



Redreçament estàtic de ports: túnel SSH per accedir al servei SMTP

El redreçament estàtic de ports ens permet crear un canal de comunicació segur i transparent a l'usuari entre un port de la màquina local i un altra port de la màquina remota, que pot ser un dels ports estàndards que fan servir qualsevol dels serveis de xarxa.

Per exemple, per establir un túnel SSH entre el port local 10025 i el port 25 (corresponent al servei SMTP) de l'estació 192.168.10.100 establint la connexió amb l'usuari *usuari* faríem:

```
1 ssh usuari@192.168.10.100 –L 10025:192.168.10.100:25
```

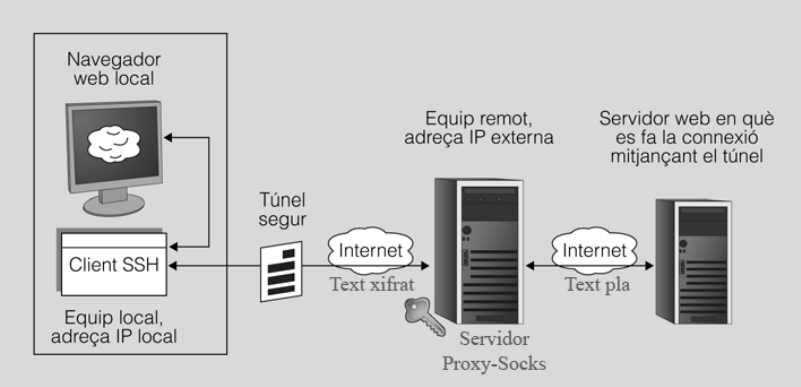
Un cop establerta la sessió SSH, i mentre es mantingui, existirà un túnel encriptat entre el port TCP local 10025 i el 25 de l'estació 192.168.10.100.

En altres paraules, per fer una connexió segura amb el servidor SMTP de l'estació remota, hauríem de connectar amb el port local 10025. SSH s'encarregarà de fer el transport de manera segura entre totes dues estacions.

Redreçament dinàmic de ports: Servidor SOCKS

Una altra de les possibles aplicacions del túnel SSH és accedir a un servidor web des d'una IP diferent de la de la nostra màquina. Com es pot veure a la figura 2.8, el navegador utilitza el túnel creat per SSH per connectar-se al servidor i navegar per pàgines web utilitzant la IP externa del servidor.

Figura 2.8. Navegar per Internet utilitzant la IP del servidor

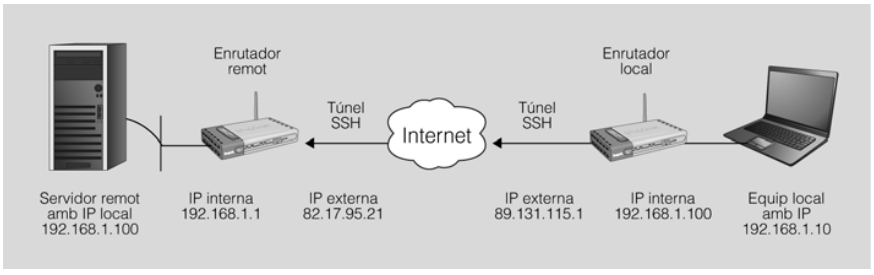


SSH permet doncs crear un túnel des de l'equip local fins a un servidor remot. Un cop connectats a aquest servidor remot utilitzem el túnel per fer la navegació per Internet. A la figura 2.9 podeu veure les IP de cada un dels equips de l'exemple. Fixeu-vos que tant la màquina local com el servidor remot tenen la seva pròpia IP local i una IP externa (NAT). Per connectar-nos al servidor remot haurem d'utilitzar l'adreça IP externa de la xarxa on està connectat, i només funcionarà si l'encaminador al qual ens connectem té el port 22 redreçat al servidor amb el qual farem el túnel. És a dir, quan l'encaminador remot rebi una connexió pel port 22 ha de redreçar aquest trànsit al port 22 del nostre servidor.

NAT (network address translation)

La traducció d'adreces de xarxa és una tècnica que amaga un espai d'adreces, que generalment consisteix en un conjunt d'adreces de xarxa privada, darrere d'una única adreça IP, sovint en l'espai d'adreces IP públiques. Aquest mecanisme s'implementa en un encaminador que utilitza unes taules per assignar les adreces "ocultes" a una sola adreça IP, de manera que els paquets a la sortida semblen originar-se en l'encaminador.

Figura 2.9. Configuració d'equips per fer un túnel SSH



Abans de fer el túnel SSH, si es fa una connexió a Internet des de l'equip local amb un navegador qualsevol, es farà des de l'adreça IP 89.131.115.1.

A continuació veiem com fer el túnel amb l'equip remot:

```
1 ssh –D 1080 –p 22 –Nf usuari@82.17.95.21
```

L'opció *-D* habilita el **redreçament dinàmic de ports locals** que a la pràctica permet fer servir SSH com un servidor SOCKS.

Un **servidor SOCKS** dóna a la intranet un servei similar al que proporciona un servidor web intermediari (*proxy*), però no està limitat al protocol HTTP/HTTPS, sinó que permet redreçar qualsevol tràfic TCP/IP. Fins i tot la versió 5 de SOCKS permet el tràfic UDP.

SOCKS

Protocol d'Internet que facilita l'encaminament de paquets de xarxa entre les aplicacions client-servidor a través d'un servidor intermediari.

El servidor SOCKS funciona mitjançant l'assignació d'un sòcol per escoltar el port local. En el nostre cas hem triat el port 1080, ja que és l'estàndard per a aquest servei. Cada vegada que s'estableix una connexió a aquest port, la connexió es transmet pel canal segur. A continuació el protocol d'aplicació és l'encarregat de determinar on es fa la connexió a la màquina remota.

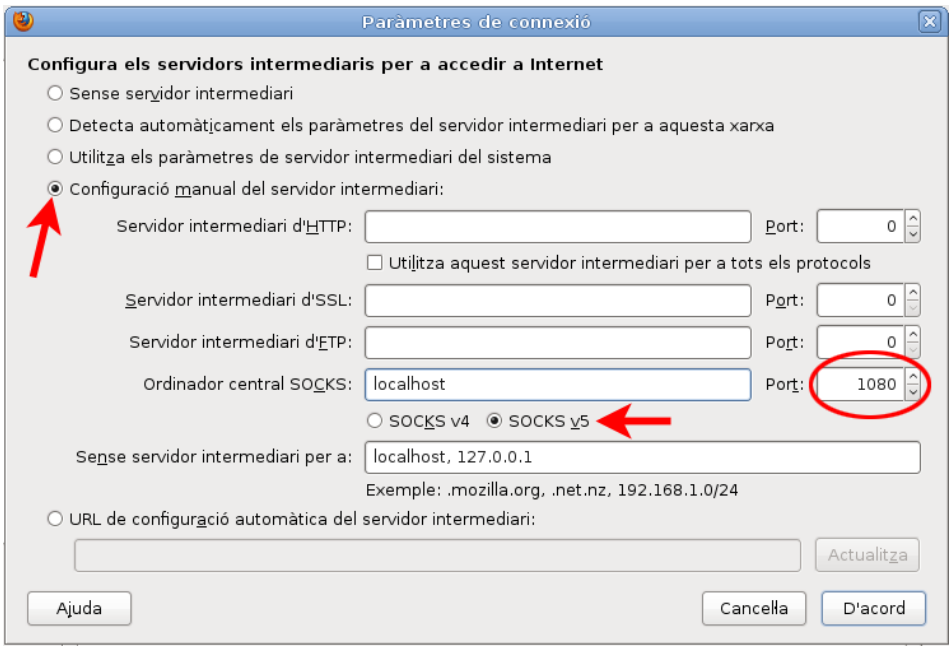
A la taula 2.10 veiem amb detall les opcions emprades.

Taula 2.10. Opcions de redreçament dinàmic de l'ordre ssh

Opció	Descripció
-D 1080	Habilita el redreçament dinàmic de ports fent servir, en aquest cas, el port 1080 estàndard per a servidors SOCKS.
-p 22	Permet indicar el port pel qual es farà el túnel SSH, habitualment el número 22.
-N	Especifica que no es vol executar cap ordre remota i no s'obrirà cap terminal interactiu. És d'utilitat quan es fa servir SSH per al redreçament de ports.
-f	Com que no ens cal interactivitat, aquesta opció fa que SSH quedi en segon pla i es dissocii de la *shell* actual, cosa que converteix el procés en un dimoni (cal l'opció -N).

Un cop establert el túnel, cal configurar a nivell d'aplicació el programa que el farà servir. En aquest cas l'aplicació que hem de configurar és el navegador local perquè utilitzi el túnel SSH per accedir a la xarxa d'Internet.

Figura 2.10. Configuració del navegador Firefox per utilitzar el servidor SOCKS



Si utilitzeu el navegador Firefox, heu d'anar a *Edita* > *Preferències* > *Avançat*, seleccionar la pestanya *Xarxa* i prémer el botó *Paràmetres*. S'ha de configurar, tal com mostra la figura 2.10: fer servir la configuració manual del servidor intermediari (*proxy*), que l'ordinador central SOCKS sigui *localhost* i que utilitzi el port 1080.

Ara el navegador ja utilitza el túnel SSH i quan es fa una connexió a Internet des de l'equip local es fa des de l'adreça pública de l'encaminador remot (en aquest cas, amb IP 82.17.95.21).

A Linux hi ha diverses aplicacions que permeten a les aplicacions no compatibles amb SOCKS fer les seves peticions TCP/IP a través d'un servidor SOCKS. Les més conegudes són tsocks i proxychains.

Hi ha aplicacions que no admeten específicament l'ús del servidor SOCKS i que no disposen de la possibilitat de configurar aquesta opció. En aquest cas es pot carregar un programa addicional que detecti peticions a la pila TCP/IP per modificar-les i redreçar-les a través del servidor SOCKS sense que la aplicació hagi de tenir implementada aquesta possibilitat.

2.3 Administració remota amb interfície gràfica

Si la xarxa de comunicacions té una amplada de banda suficient i un retard raonable, es poden fer servir eines que permeten treballar amb aplicacions que s'executen en un equip remot, però que mostren la seva interfície gràfica en un terminal local. En tot cas, la seguretat ha de ser un punt fonamental a considerar, atès que la funcionalitat administrativa de l'aplicació farà que transportem per la xarxa dades importants.

És possible que tot l'escriptori visualitzat es correspongui amb l'escriptori d'un equip remot o combinar en el mateix escriptori aplicacions locals i remotes. En aquest últim cas, fins i tot les aplicacions remotes es poden estar executant en diferents equips.

També cal diferenciar els protocols de comunicació remota, que són convencions que defineixen la comunicació entre la màquina client i la que ofereix el servei de les aplicacions client-servidor que fan servir aquests protocols per donar un servei concret d'accés gràfic remot.

Així, doncs, trobem que hi ha diferents aplicacions que implementen aquest protocols i permeten dur a terme l'administració remota d'equips mitjançant una interfície gràfica. Concretament algunes de les opcions més conegudes són:

- El sistema servidor X Window (transparència de xarxa)
- Computació virtual en xarxa VNC (*virtual network computing*)
- Webmin, eina gràfica per a l'administració remota de Linux

2.3.1 Protocols d'accés remot a interfícies gràfiques

Hi ha dues propostes tecnològiques principals quant a protocols d'accés remot a interfícies gràfiques. D'una banda, es poden transmetre directament les diferents directives que donen instruccions als motors gràfics dels servidors de finestres respectius. De l'altra, es pot transmetre informació relativa a cada píxel que permet fer una representació del mapa de memòria gràfica del gestor de finestres.

Alguns dels protocols poden incorporar les seves pròpies tècniques de seguretat i xifrat o bé incorporar una capa addicional de seguretat, per exemple fent servir túnels SSH.

Protocol X11

Protocol desenvolupat inicialment a mitjan anys vuitanta a l'Institut Tecnològic de Massachusetts (MIT) com a part del sistema de finestres X Window amb l'objectiu de proporcionar una interfície gràfica als sistemes Unix.

> La versió del protocol que es fa servir des de 1987 i fins a l'actualitat és la v11. Per aquesta raó s'acostuma a anomenar X11 tant al protocol com als servidors i clients del sistema X Window.

Aquest protocol és independent del sistema operatiu i permet la interacció gràfica remota entre un client i un servidor fent transparent la xarxa per a l'usuari, que veu el sistema com si fos un terminal gràfic virtual.

Tecnologia NX

FreeNX

És una implementació en codi lliure amb llicència GPL de la tecnologia client-servidor NX.

Més que un protocol de comunicació pròpiament dit, NX és una tecnologia que permet gestionar i millorar les connexions X Window fent servir compressió del tràfic de dades del mateix protocol X11 i afegint mecanismes de memòria cau per accelerar la comunicació i reduir el requeriments d'amplada de banda i latència.

Remote framebuffer (RFB)

El protocol de xarxa RFB (remote framebuffer protocol) també és lliure i fa servir la tècnica d'enviar informació dels píxels que conformen la memòria d'imatge gràfica. Malgrat que les funcions inicials del protocol RFB es limitaven a transmetre els esdeveniments desencadenats per l'usuari en el teclat i el ratolí locals a l'estació remota, i a enviar en direcció contrària el contingut gràfic de la pantalla, les darreres versions han estat ampliades per oferir més funcionalitats, com per exemple la transferència de fitxers.

La memòria d'imatge

La memòria d'imatge (*framebuffer*) és l'àrea de la memòria que emmagatzema les dades que defineixen la imatge gràfica que apareix a la pantalla. Normalment, les pantalles actuals estan basades en píxels, és a dir, mostren mapes de bits en comptes d'imatges vectorials.

La memòria d'imatge conté la informació necessària per definir l'estat de cada píxel de la pantalla. La quantitat d'informació necessària dependrà de la resolució i la profunditat de color (la quantitat de bits necessaris per codificar el color, i potser la transparència, de cada píxel).

Els programes més coneguts que fan servir el protocol RFB són tota la família d'aplicacions **VNC** (*virtual network computing*).

El protocol RFB fa servir per defecte el port 5900 del servidor per establir la comunicació. Alternativament, en cas de connexió a través d'un navegador es fa servir per defecte el port 5800.

Cal assenyalar que RFB no és un protocol segur. En fer servir VNC en una xarxa que no sigui de confiança, caldrà combinar el seu ús amb SSH o bé una altra tecnologia per crear una xarxa privada virtual (VPN).

VPN

Una xarxa privada virtual (VPN) és una xarxa informàtica que s'aplica en una capa lògica addicional a la d'una xarxa existent. Té el propòsit de crear un àmbit privat de les comunicacions per ordinador o fer una extensió segura d'una xarxa privada en una xarxa insegura, com Internet. Una aplicació comuna és protegir les comunicacions a través de la xarxa pública d'Internet.

Remote desktop protocol (RDP)

Protocol propietari desenvolupat per Microsoft, que el fa servir en el seu servidor de serveis d'escriptori (*terminal services*). Incorpora els seus propis sistemes de compressió i xifrat i fa servir el port TCP3389 per defecte per escoltar les peticions al servidor.

Microsoft té lògicament el seu propi client (*terminal services client*) però hi ha també clients per a una gran varietat de sistemes operatius (Windows Mobile, Linux, Unix, Mac OS X, Android, etc.).

Altres protocols

Hi han altres protocols menys coneguts, com ara ICA (*Independent Computing Architecture*), producte propietari de la companyia Citrix Systems per als seus servidors d'aplicacions, o bé AIP (A*daptive Internet Protocol*), protocol propietari que fa servir Sun.

2.3.2 El servidor X Window

El sistema X Window implementa les funcions necessàries per controlar finestres i dispositius d'entrada com el ratolí o el teclat. Aquest sistema s'utilitza en la major part de versions d'Unix per implementar la interfície gràfica. Val a dir que X Window no defineix com ha de ser aquesta interfície, només permet implementar-la. Escriptoris tan coneguts a GNU/Linux com GNOME, KDE o Xfce utilitzen el mateix sistema X Window malgrat que són escriptoris diferents amb les seves particularitats.

Entre les característiques pròpies del sistema X Window trobem que és independent del sistema operatiu emprat, només es tracta d'una capa d'aplicació que, des del principi, va ser pensada per treballar en xarxa.

X.Org

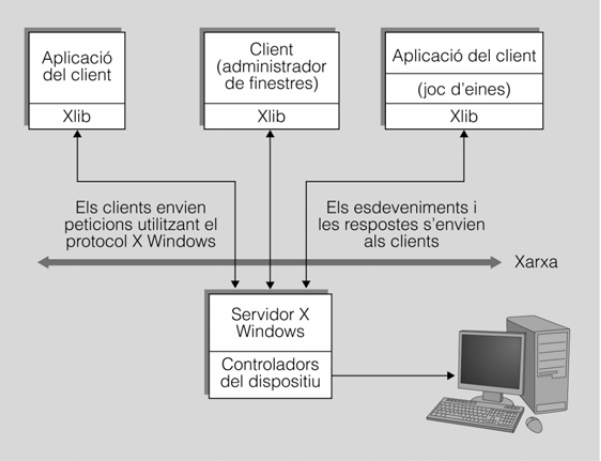
Logotip de X.Org Foundation

Existeix la fundació que gestiona el desenvolupament del sistema X Window com la implementació de referència d'aquest sistema. És programari lliure i des del seu inici com a ramificació del projecte XFree86 ha guanyat popularitat. Actualment, és la implementació del sistema X Window emprada en gairebé totes les distribucions GNU/Linux i en altres sistemes operatius.

El servei de finestres X Window fa servir el protocol X11 i permet separar en una xarxa l'estació que representa la interfície gràfica de l'estació on s'executa realment l'aplicació, de forma nativa i transparent per a l'usuari. Aquesta característica del sistema de finestres X Window s'anomena també **transparència de xarxa**.

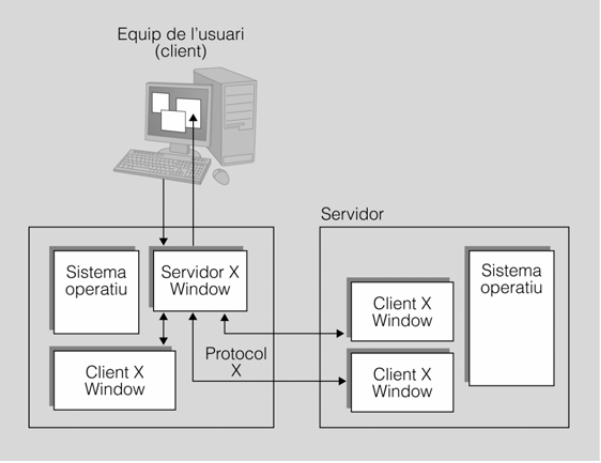
La figura 2.11 representa el funcionament d'X Window i il·lustra el comportament de la transparència de xarxa, tot i que, en molts casos, client i servidor gràfic poden estar al mateix ordinador.

FIGURA 2.11. Transparència del sistema X Window



En la nomenclatura del sistema X Window, el programari que permet dibuixar finestres rep el nom de servidor X i els programes que obren les seves finestres en el servidor X s'anomenen **clients X**. Així, el servidor X s'executarà en l'estació local de l'usuari per tal que aquest pugui interaccionar amb la interfície gràfica, l'estació remota que executa una aplicació serà el client X. Aquest ús dels termes servidor i client centrat en el punt de vista del programari sovint despista els usuaris novells. A la figura 2.12 es pot observar una arquitectura client-servidor; en aquest cas, és el client d'aquesta estructura el que proporciona el servei X Window (el client de l'aplicació fa de servidor de finestres X Window).

FIGURA 2.12. Estructura client-servidor des del punt de vista de l'aplicació



La comunicació entre el servidor X i el client X es fa sense cap tipus d'encriptació; per això només es podrà fer servir en xarxes de confiança. En la resta de casos serà necessari emprar SSH per tunelitzar el trànsit del sistema X Window, o bé crear una xarxa privada virtual (VPN).

Configuració d'un servidor X Window

Bàsicament, un servidor X11 representa un terminal gràfic amb els seus propis teclat i ratolí. Per tal que els clients puguin emprar aquest terminal és necessari fer dues accions:

1. Configurar el servidor X11 perquè permeti la connexió dels clients.
2. Indicar als programes clients on és el servidor X11.

Configurar un servidor X11 de manera poc curosa és un risc de seguretat molt greu. Qui es connecta al servidor X11 rep missatges amb notificacions, per exemple, de les tecles premudes i dels moviments del ratolí. Així, doncs, cal establir qui es pot connectar a un servidor X11.

Per fer aquesta funció es poden fer servir dos mètodes: ***xhost*** i ***xauth***. El primer mètode permet especificar permisos per a diferents equips en indicar-ne l'adreça IP o el seu DNS. Si es concedeix accés a un equip determinat, qualsevol aplicació d'aquell amfitrió (sigui quin sigui el seu usuari) podrà accedir al servidor X11.

El segon mètode està basat en una clau anomenada *galeta màgica* (*magic cookie*), que és un conjunt arbitrari de dades que es crea en activar el servei. Cada client que vulgui connectar-se al servidor ha de provar que té coneixement d'aquesta galeta.

Tanmateix, potser l'alternativa més senzilla i eficaç per solucionar la qüestió de la seguretat sigui la utilització d'un túnel fent servir un protocol de xifrat com ara SSH (secure shell).

Per tal de configurar un servidor X11 per permetre connexions dels clients, es poden emprar les ordres *xhost* i *xauth*. La primera suposa riscos de seguretat i la segona és confusa d'emprar. Per això, la millor opció és utilitzar SSH per redreçar el trànsit X11.

D'altra banda s'ha d'indicar als programes clients on és el servidor X Window que han d'emprar per obrir les seves finestres. Per fer això disposem de la variable d'entorn DISPLAY. Aquesta variable d'entorn utilitza una cadena amb la sintaxi següent:

```
host:display.screen
```

A la taula 2.11 podeu veure els principals paràmetres de configuració d'un servidor gràfic X11.

TAULA 2.11. Paràmetres de configuració d'un servidor X11

Paràmetre	Significat
Host	Adreça IP de l'estació on s'executa el servidor X11. Si no es defineix se suposa que es tracta de l'equip local.
:display	Nombre de displays dins del servidor X11. Igual que un equip físic pot tenir diferents terminals virtuals (/dev/ttyX), un servidor X11 pot tenir diferents displays. Normalment el primer és :0
.screen	Nombre de pantalles dins del display. Normalment no s'especifica, en aquest cas es fa servir la pantalla .0

Exemples

```
#export DISPLAY=:0
```

S'està indicant que s'utilitzi el primer terminal gràfic del servidor X de l'equip local.

```
#export DISPLAY=192.168.0.50:2
```

S'està indicant que s'utilitzi el segon terminal gràfic del servidor X de l'equip remot amb la IP 192.168.0.50.

```
#export DISPLAY=192.168.0.50:2.1
```

S'està indicant que s'utilitzin el segon terminal gràfic i la pantalla 1 del servidor X de l'equip remot amb la IP 192.168.0.50.

Iniciació d'un client X mitjançant SSH

Servidors X

Si estem fent servir GNU/Linux, la nostra distribució empaquetarà la versió estable d'un servidor X, probablement X.Org. Per a altres sistemes operatius disposem de programari lliure, com Cygwin/X o Xming, que permeten veure aplicacions gràfiques Unix a l'escriptori de Microsoft Windows.

Si el nostre equip està executant un servidor X, podrem establir una sessió SSH en un equip remot per executar allà qualsevol aplicació i que la seva interfície gràfica es representi al nostre servidor X.

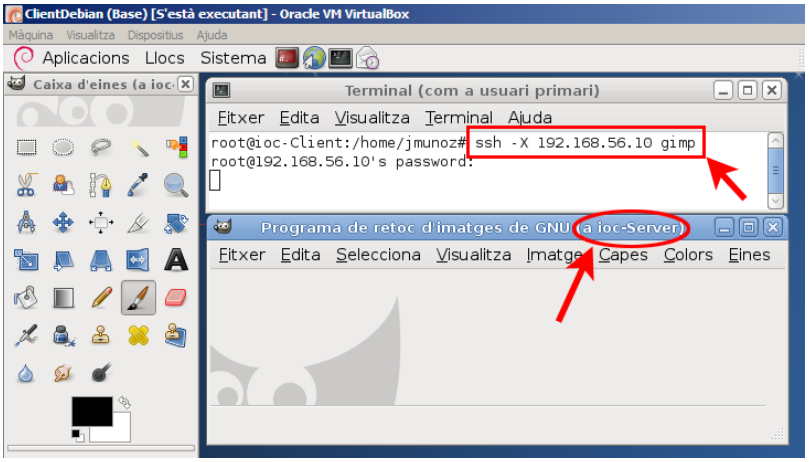
Aquesta és probablement la manera més senzilla i segura de protegir la comunicació entre servidor i client X Window. En aquest cas SSH s'encarrega de:

- Tots els procediments criptogràfics necessaris per establir un túnel entre els dos equips i dur a terme l'autenticació de l'usuari remot.
- Establir el valor adient per a la variable d'entorn DISPLAY de l'equip remot per tal que les aplicacions gràfiques utilitzin el servidor X11 local (és a dir, el que s'executa de manera local en l'estació on hem executat SSH per connectar-nos a l'equip remot).
- Transportar tot el trànsit X11 protegit dins del túnel SSH.
- Permetre l'ús del servidor X11 local per a les aplicacions de l'estació remota. Només per a l'usuari propietari de la sessió SSH, no per a la resta de processos dels altres usuaris.

A la figura **??** es mostra una captura de pantalla en què s'ha emprat SSH per iniciar una sessió de treball en una estació remota en la qual s'ha executat el programa GIMP. En aquest cas és important observar que en fer la connexió SSH s'ha indicat el paràmetre *-X* per tal d'activar el redreçament de trànsit X11.

```
1 #ssh –X jmunoz@192.168.56.10 gimp
```

FIGURA 2.13. Execució de GIMP en una estació remota



L'aplicació GIMP s'està executant en l'ordinador servidor remot (en aquest cas ioc-Server). Aquest envia la informació gràfica mitjançant un canal segur SSH a l'ordinador client (que actua com a servidor gràfic X Window) i mostra les finestres resultants a l'usuari.

Cal indicar que el servidor SSH haurà de tenir activat el redreçament del protocol X (normalment està activat per defecte), és a dir que haurà de tenir el paràmetre següent en una línia de l'arxiu de configuració `/etc/ssh/ssh_config`:

```
1 // Activació del redreçament X en /etc/ssh/sshd_config
2 X11Forwarding yes
```

La combinació del sistema client-servidor X Window en xarxa amb les característiques de seguretat d'SSH suposa una combinació de flexibilitat, seguretat i facilitat d'ús difícilment igualable per altres tècniques. Tot i així, cal recordar que és important que la xarxa tingui una bona amplada de banda i una latència petita, ja que la comunicació client-servidor genera gran quantitat de peticions-respostes (anomenades en anglès *roundtrip*) que poden alentir el sistema. Tal com hem comentat, per intentar millorar aquests inconvenients hi ha tecnologies i programes per gestionar connexions X Window amb eficiència, com ara FreeNX.

XDMCP per accedir a escriptoris remots

Normalment, en arrencar un ordinador amb GNU/Linux apareix una pantalla gràfica demanant nom d'usuari i contrasenya per iniciar sessió. Aquest programa, que en cas d'autenticar l'usuari carrega tot l'escriptori, és un administrador de pantalla (display manager). El protocol que controla aquest programa respon a l'acrònim XDMCP (*X display manager control protocol* o protocol de control d'un

administrador de pantalla) i funciona sobre un servidor X Window. Té diferents implementacions, de les quals les més usuals són: **gdm** per a l'escriptori GNOME, **kdm** per a l'escriptori KDE, i **xdm**, el més bàsic per llençar les X.

En la major part dels casos l'administrador de pantalles gestiona un servidor X11 que s'està executant de manera local. Però gràcies al protocol **XDMCP** també pot gestionar servidors X11 remots. En aquest cas, un equip pot mostrar la benvinguda (demanant usuari i contrasenya) en un equip remot i, després de dur a terme una autenticació positiva, mostrar a l'usuari un escriptori remot, no només la finestra d'una aplicació.

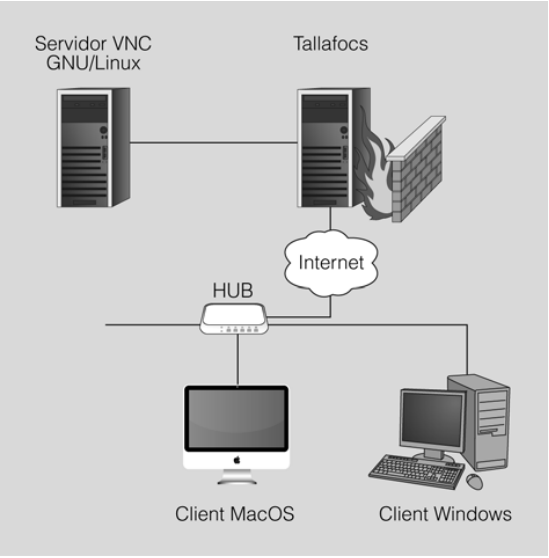
La configuració de tots els elements necessaris és força extensa per tractar-la aquí, però és convenient saber que la transparència de xarxa del sistema X Window permet mostrar tot un escriptori remot, no només les finestres aïllades de les aplicacions remotes.

2.3.3 Virtual network computing (VNC)

VNC és un sistema per veure, i si s'escau també controlar, un escriptori remot. Originalment va ser desenvolupat per Olivetti i hi ha implementacions gratuïtes de les eines que permeten treballar amb VNC en sistemes operatius diferents.

Aquesta aplicació permet accedir des d'un ordinador anomenat client a un altre ordinador anomenat servidor connectats mitjançant una xarxa. Un cop s'ha establert la comunicació, l'equip client pot utilitzar l'equip servidor sense cap limitació, llevat de les imposades per l'amplada de banda de la xarxa. El resultat és una visualització en pantalla de l'equip remot, de l'escriptori i de tots els seus programes.

FIGURA 2.14. Execució remota d'un sistema GNU/Linux en dues màquines client amb sistemes operatius diferents



Per aconseguir això, cal instal·lar dos programes: un servidor VNC a la màquina

a la qual vulguem accedir i un visualitzador VNC a la màquina client. És possible utilitzar un visualitzador en un sistema operatiu determinat i un servidor en un de diferent. D'aquesta manera, és possible executar un sistema Windows complet des d'un Macintosh o GNU/Linux, o fer qualsevol combinació que puguem imaginar. Com es pot veure a la figura 2.14, es pot executar un sistema GNU/Linux des d'un equip amb sistema operatiu Mac OS o Windows.

Funcionament de les aplicacions VNC

VNC fa servir el protocol RFB (*remote framebuffer*) que està basat en una memòria d'imatge i així pot treballar amb qualsevol sistema de finestres, ja sigui X Window, Windows o un altre. En aquest sistema sempre s'anomena client l'estació en què es troba l'usuari (equip local) i es denomina servidor l'equip que defineix el contingut de la pantalla (equip remot).

Compte! La nomenclatura per definir el client i el servidor és inversa a la que utilitza el sistema X Window.

Tot el sistema està pensat per treballar amb clients lleugers (*thin clients*) de manera que el client es pugui implementar sense gaires recursos. De fet, tot el protocol i el funcionament del sistema intenten ser molt senzills. El pilar fonamental són les actualitzacions gràfiques que el servidor envia cap al client, en forma de rectangles, per modificar les dades de la memòria d'imatge. Aquests missatges poden emprar diferents directives, però n'hi ha unes de bàsiques que tots els clients i servidors VNC han de conèixer.

Una característica fonamental del sistema VNC és que el client no conserva cap estat. Així és possible tancar un client i tornar-lo a obrir, fins i tot en un altre equip, com si es tractés d'un monitor virtual.

En els sistemes GNU/Linux és possible exportar mitjançant VNC l'escriptori actual o fins i tot un terminal de text, o bé crear un escriptori virtual nou. Els sistemes Microsoft Windows no permeten la creació d'escriptoris virtuals nous, només permeten exportar en xarxa l'únic escriptori disponible. En el moment d'exportar un escriptori es poden definir dues contrasenyes diferents, una per a aquells usuaris que podran controlar de manera remota aquest escriptori i una altra per a aquells usuaris que el podran veure però no interactuar amb el teclat i el ratolí.

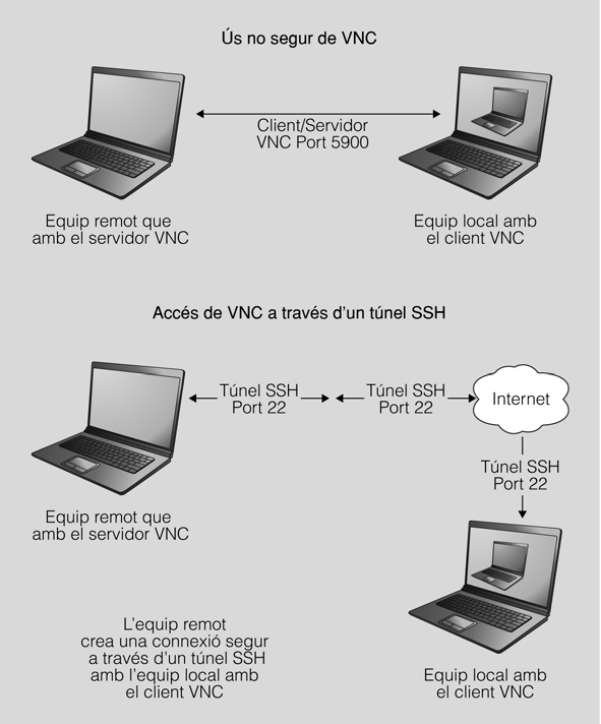
El procés per fer servir VNC és el següent:

1. A l'estació remota cal executar un servidor VNC per exportar l'escriptori de treball o un de nou.

2. A l'estació local es farà servir un client VNC per interactuar amb el servidor remot.

A la figura 2.15 es pot veure un esquema d'una connexió VNC a través d'un túnel SSH.

FIGURA 2.15. Ús no segur de VNC i ús a través d'un túnel SSH



La major part de servidors VNC inclouen un petit servidor web que ofereix la descàrrega d'una miniaplicació Java que es pot fer servir com a client VNC. D'aquesta manera, en l'ordinador local només cal un navegador compatible amb Java per poder accedir a l'escriptori remot.

Implementacions VNC

Si el desenvolupament original de VNC té algun hereu oficial, és RealVNC. Un producte desenvolupat per la companyia del mateix nom, que està formada per alguns dels desenvolupadors originals. Les eines de RealVNC s'ofereixen amb llicències diferents: l'edició Free, amb llicència GPL, i la resta, amb llicències privatives.

GPL

La llicència pública general (GPL, de l'anglès *general public license*) és un tipus de llicència per a programari que permet la còpia, distribució (comercial o no) i modificació del codi, sempre que qualsevol modificació es continuï distribuint amb la mateixa llicència GPL. La llicència GPL no permet la distribució de programes executables sense el codi font corresponent.

Una altra implementació força estesa és TightVNC, programari lliure amb llicència GPL que, tot i que és plenament compatible amb les eines originals, introdueix algunes extensions al protocol RFB. Aquestes extensions només es podran emprar en fer servir TightVNC, tant en el client com en el servidor. Entre les extensions introduïdes per TightVNC hi ha codificacions noves que redueixen la càrrega de la xarxa i la possibilitat de transmetre fitxers.

A la taula 2.12 podeu trobar un llistat dels productes més populars que implementen VNC.

TAULA 2.12. Programaris que implementen VNC

Programari	Característiques
VNC	És el programari original. En l'actualitat només s'utilitza com a referència i per a les proves de compatibilitat.

TAULA 2.12 (continuació)

Programari	Característiques
TightVNC	Una versió que s'ofereix gratuïtament amb llicència GPL. S'han millorat significativament els algorismes de compressió de VNC i permet la transferència de fitxers.
RealVNC	És una versió desenvolupada per alguns dels desenvolupadors d'AT&T. L'equip de RealVNC també està oferint-ne una versió empresarial i ha elaborat un producte molt interessant que pretén ser accessible per a qualsevol client VNC.
UltraVNC	UltraVNC ofereix una funció de transferència d'arxius, una gestió millorada de la compressió de vídeo i una funció de xat. Recentment s'ha afegit l'opció de controlar una sola finestra del programa en lloc de controlar tot l'escriptori.

Altres eines relacionades amb VNC i que cal destacar són **Vino**, una eina de GNOME per compartir l'escriptori i que acostuma a incloure Ubuntu. Tenim també diversos clients VNC com **Vinagre** i el potent client VNC que inclou la distribució Debian anomenat **Remmina**, que no només és compatible amb el protocol RFB de VNC sinó també amb la tecnologia NX o el protocol RDP de Microsoft Windows.

2.3.4 Programari d'accés remot TightVNC

El programari d'accés remot **TightVNC** és una de les implementacions de VNC més avançades. Es distribueix amb llicència GPL, la qual cosa permet un accés lliure al seu codi font, a la seva distribució i a la seva instal·lació.

Instal·lació del servidor TightVNC

El paquet del servidor TightVNC s'anomena *tightvncserver* i per instal·lar-lo només cal escriure el següent a la línia d'ordres:

```
# apt-get install tightvncserver
```

Un cop instal·lat ja podem arrencar el servidor VNC, *vncserver*, juntament amb una sèrie d'opcions.

```
# vncserver :1 -geometry 1024x768 -depth 16 -pixelformat rgb565
```

Opcions:

- `:1`: indica la pantalla en la qual s'establirà la sessió per fer les connexions remotes.
- `-geometry`: estableix les dimensions de la finestra amb què es farà la connexió.

- `-depth`: estableix la profunditat dels colors en bits per píxel. Ha de ser un valor entre 8 i 32.

Contrasenya

Atenció! Hi ha un límit de 8 caràcters per a la contrasenya de TightVNC.

- `-pixelformat`: estableix el format de la representació dels píxels.

A continuació, el programari ens demanarà dues contrasenyes, la segona de les quals és opcional. La primera és la necessària per permetre l'accés total a l'equip servidor des de l'equip remot. La segona és per permetre només un accés de visualització de l'escriptori.

```
#You will require a password to access your desktops.
Password:
Warning: password truncated to the length of 8.
Verify:
Would you like to enter a view-only password (y/n)? n
```

Per aturar el servidor TightVNC es pot fer servir l'ordre *vncserver* i s'indica la pantalla en què s'ha invocat.

```
#vncserver -kill :1
Killing Xtightvnc process ID 4223
```

Instal·lació del client TightVNC

El paquet que permet instal·lar el client TightVNC es diu *xtightvncviewer*.

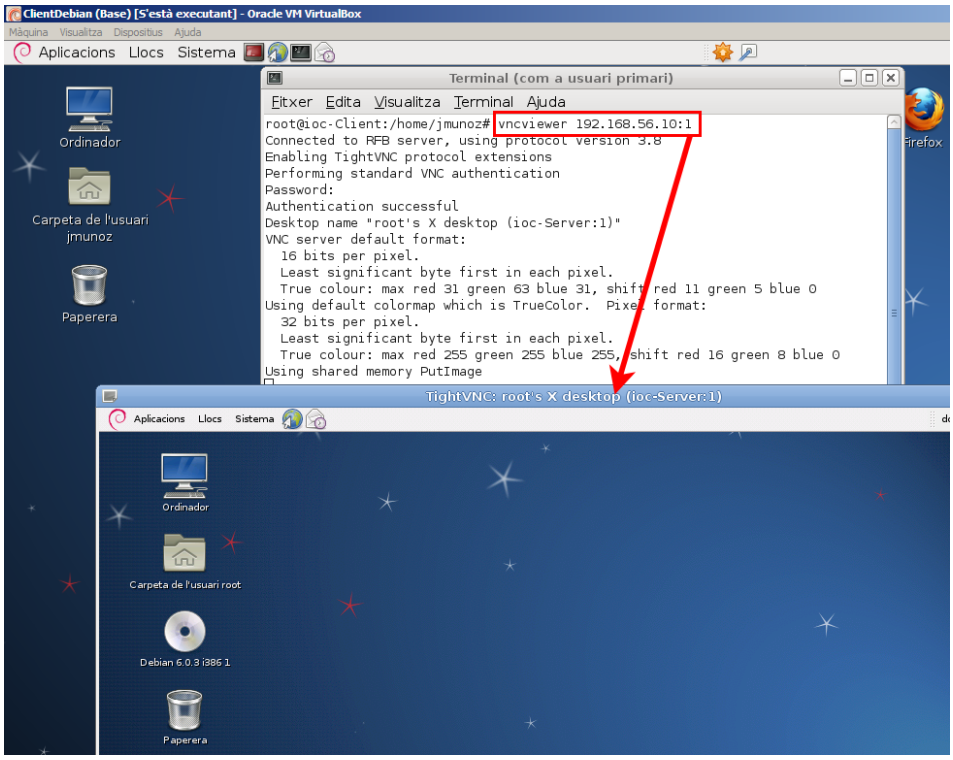
```
# apt-get install xtightvncviewer
```

A continuació, un cop s'està executant el servidor de TightVNC, es pot fer la connexió des del client remot. S'ha d'indicar l'adreça IP del servidor i la pantalla on està funcionant el servidor.

```
# vncviewer 192.168.56.10:1
```

A la figura 2.16 es pot veure com accedim des del terminal remot (ioc-Client) al servidor d'IP 192.168.56.10 i podem veure com s'obre l'escriptori d'aquest servidor (ioc-Server). Fixeu-vos que hem introduït la contrasenya que permet accedir a la sessió. Si hi haguéssim accedit amb la contrasenya de només visualització, no hauríem pogut navegar pels menús del sistema.

FIGURA 2.16. Accés remot mitjançant TightVNC



Accés remot per web i TightVNC

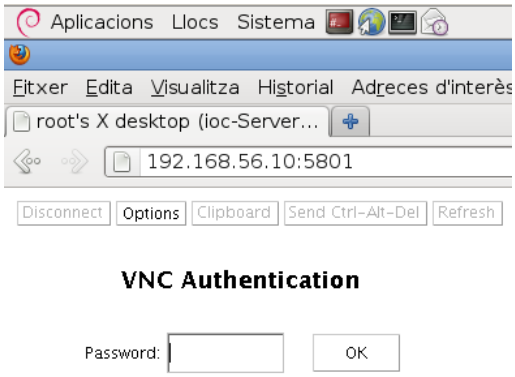
L'accés remot es pot fer també mitjançant un navegador web que sigui compatible amb les miniaplicacions (*applets*) de Java, com Firefox. Per poder accedir d'aquesta manera, el primer que s'ha de fer és instal·lar al servidor el component que permetrà aquest tipus d'accés.

```
1 # apt-get install tightvnc-java
```

A partir d'aquest moment ja es podrà accedir al servidor a través del servidor web i del port 5801. Cal tenir clar que no serà suficient tenir en compte només el component *tightvncjava* instal·lat; serà necessari, a més, tenir un servidor web també instal·lat. Per sort, la major part de distribucions GNU/Linux (inclosa Debian) porten instal·lat de manera predeterminada el servidor web Apache.

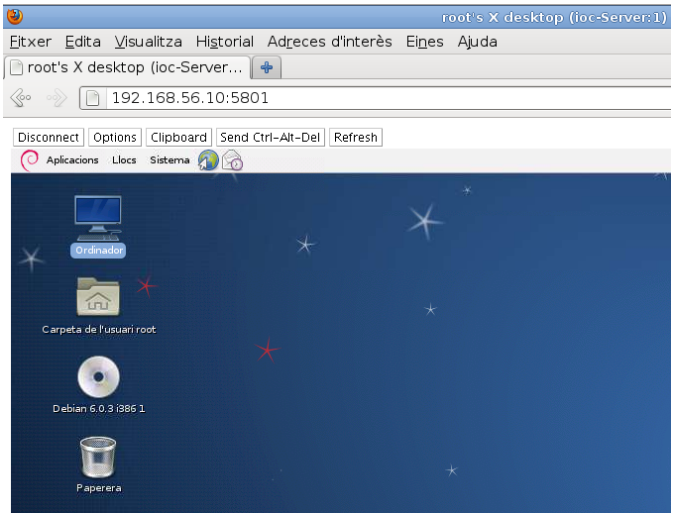
A la figura 2.17 podeu veure que, un cop accedim al servidor mitjançant l'adreça `http://192.168.56.10:5801`, ens demana una contrasenya d'autenticació VNC.

FIGURA 2.17. Accés remot mitjançamt un navegador web



Si introduïu bé la contrasenya, i segons si és la que permet accés total o de només visualització, s'accedirà, tal com mostra la figura 2.18 a una sessió remota.

FIGURA 2.18. Inici de sessió remota mitjançant un navegador web



Recordeu que el navegador ha de tenir configurat el component necessari per a l'execució de les miniaplicacions de Java (*applets*).

Nous clients d'escriptori remot multiprotocol

La tendència actual és que els clients d'accés a escriptoris remots funcionin amb diferents protocols i que per tant es pugui accedir a diferents plataformes i sistemes operatius. És el cas del projecte Remmina, desenvolupat en GTK+ (i per tant compatible amb els escriptoris GNOME i XFCE), que permet fins i tot mantenir diferents connexions de manera simultània.

Remmina és un client d'escriptori remot que està incorporat per defecte en les darreres versions de Linux Debian i que és compatible amb els protocols RDP (Microsoft Windows), RFB (plataforma VNC), XDMCP (protocol de gestió de finestres X), NX i SSH, entre d'altres.

2.3.5 Gestió remota mitjançant una aplicació gràfica local

Una aproximació diferent a l'administració d'equips remots consisteix a executar una aplicació local amb interfície gràfica que permeti gestionar l'estació remota. El gran avantatge d'aquesta aproximació rau en el fet que, atès que l'usuari interactua amb una aplicació local, el retard en la xarxa no suposa cap problema a l'hora de treballar. A més, com que es tracta d'una aplicació local, la interfície gràfica serà coneguda per l'usuari i no es donarà la situació, desconcertant per a usuaris no tècnics, d'haver de commutar entre la interfície de l'equip local i la de l'equip remot.

Aquesta solució s'utilitza amb molta freqüència per administrar petits equips de xarxa domèstics, als quals el fabricant de l'equip proporciona un programari de gestió remota. El problema d'aquesta solució és que evidentment el programari de gestió s'haurà d'executar de manera nativa en el sistema operatiu de l'estació de l'usuari, que en principi és diferent per a usuaris diferents.

L'ús més adient d'aquest enfocament és emprar un navegador web com a eina d'administració remota. És raonable suposar que, sigui quin sigui el sistema operatiu de l'usuari, aquest disposarà d'un navegador web. En aquest cas el fabricant del dispositiu implementarà una interfície web d'administració que permetrà la gestió del dispositiu per a tots els seus usuaris. Avui dia la major part de dispositius de xarxa, com ara els commutadors, les impressores o els punts d'accés, incorporen una interfície web per a la seva administració.

Fins i tot és possible administrar servidors complets a través del web, fent servir eines com **Webmin**.

Introducció a Webmin

Webmin és una aplicació que permet administrar un servidor Unix/Linux de manera remota mitjançant qualsevol navegador web modern.

Es tracta de programari lliure amb llicència GPL, escrit en Perl i que es pot executar en sistemes operatius com GNU/Linux, OpenSolaris i FreeBSD.

Un cop instal·lat i en funcionament, Webmin proporciona una interfície web (normalment accessible al port 10000) per administrar un gran ventall d'opcions en l'equip i els seus serveis. Es tracta d'una aplicació modular que permet la gestió de diferents àrees en funció dels mòduls instal·lats. En la distribució estàndard s'hi inclouen mòduls que permeten gestionar des del sistema operatiu (usuaris, quotes de disc, processos) fins als serveis de xarxa (Apache, Bind, Samba, etc.). I tot, amb una interfície web molt intuïtiva per a l'usuari.

Perl

És un llenguatge de programació dinàmic, dissenyat per Larry Wall i disponible en moltes

plataformes diferents. Es tracta d'un llenguatge de propòsit general amb el qual es poden implementar tot tipus d'aplicacions. Però la construcció d'aplicacions web dinàmiques i l'administració de sistemes són dues especialitats que no es poden discutir. En molts sentits Python és una alternativa més moderna a Perl, malgrat que no treu cap vigència a Perl. Tots dos són eines excel·lents.

Instal·lació de Webmin

A la pàgina de Webmin (www.webmin.com) es pot descarregar l'aplicació en diferents formats inclòs Debian (*.deb*). Atès que es tracta d'una aplicació escrita en Perl, no hi ha cap dependència d'una arquitectura específica. Només necessitarem tenir instal·lat un intèrpret de Perl amb les seves dependències. Potser la manera més senzilla d'instal·lar-lo és mitjançant l'ordre *aptitude*:

```
1 # aptitude install webmin
```

La instal·lació es fa automàticament al directori /usr/share/webmin i, un cop instal·lat, la interfície web estarà disponible al port TCP 10000. L'usuari administrador de Webmin serà el superusuari (*root*) amb la seva contrasenya.

Webmin pot funcionar mitjançant HTTP i HTTPS; després de la instal·lació inicial, només estarà disponible l'accés mitjançant HTTPS.

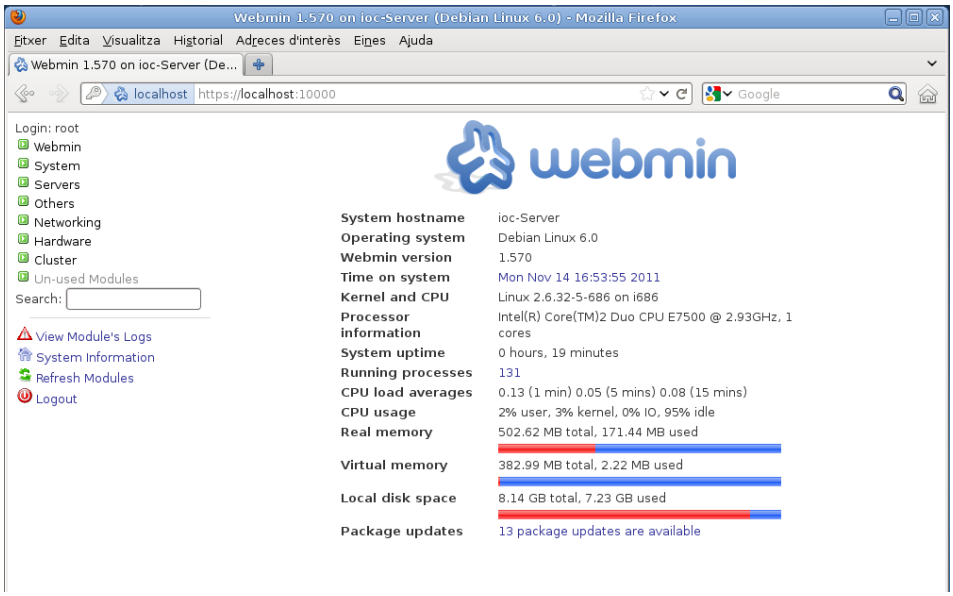
Podem treballar en mode local fent servir Webmin per administrar el propi equip. Per fer això només caldrà obrir un navegador web i dirigir-lo a l'adreça `https://localhost:10000` (o bé `https://127.0.0.1:10000`). Si volem accedir de forma remota des de un altre ordenador connectat en xarxa cal substituir *localhost* per l'adreça IP de l'equip que volem administrar, sempre, lògicament, que aquest tingui iniciat el servei Webmin.

A la figura 2.19 podrem observar la pàgina d'inici de sessió per a Webmin que s'està executant de manera local, després d'haver-nos identificat com a superusuari.

HTTP i HTTPS

La diferència entre aquests dos protocols és que el segon consisteix a fer servir HTTP combinat amb SSL (*secure socket layer*), per tal d'encriptar tota la informació transmesa.

FIGURA 2.19. Pàgina d'entrada a Webmin



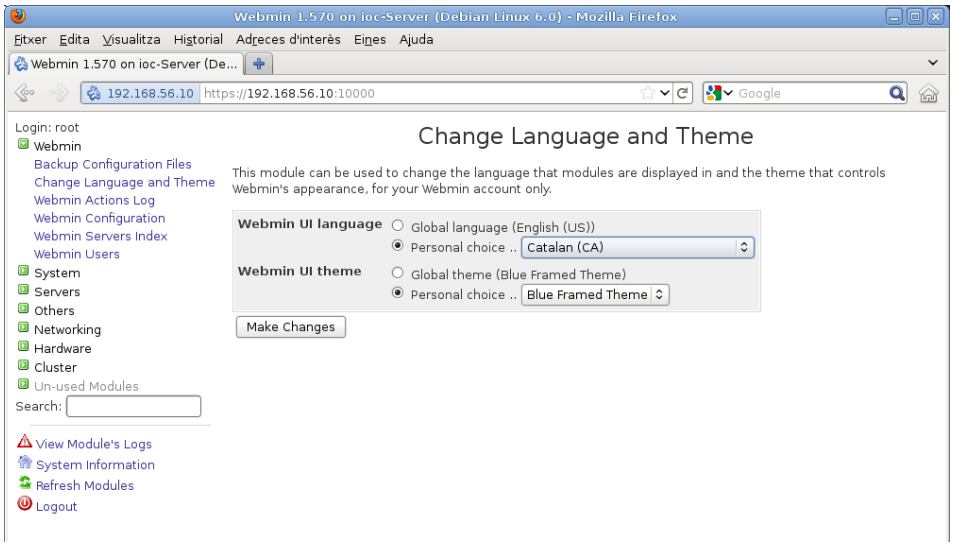
Un cop instal·lat Webmin, el fitxer **/etc/init.d/webmin** ens permetrà engegar i aturar l'aplicació com qualsevol altre servei (directives *start*, *stop*, *restart*).

Funcions de Webmin

Webmin és una aplicació modular que aporta una gran funcionalitat. Les diferents opcions s'exposen mitjançant una interfície clara i intuïtiva disponible en diferents idiomes.

Segurament, una de les primeres accions que fareu dins de Webmin serà configurar l'idioma de la interfície. Aquesta interfície està disponible en molts llenguatges diferents, que es poden seleccionar de manera individual per a cada usuari de l'aplicació o de manera general per a tota l'aplicació. Juntament amb l'idioma es pot seleccionar l'aspecte visual de la interfície. Aquests canvis es poden fer a *Webmin* > *Change Language and Theme*per a l'usuari actual, com es mostra a la figura 2.20.

Figura 2.20. Opcions d'idioma i aspecte per a la interfície de Webmin



Les opcions de la interfície estan agrupades dins de pestanyes generals, que podeu trobar a la taula 2.13.

Taula 2.13. Opcions de Webmin

Pestanya	Funcions
Webmin	Configuració general de l'aplicació: idioma, aspecte, usuaris i fitxers de registre de Webmin. Permet gestionar configuracions exportant un fitxer i tornant-lo a carregar.
Sistema	Administració del sistema informàtic en el qual s'executa Webmin. Des del control d'usuaris, les quotes de disc, les tasques periòdiques i la gestió de paquets fins a les còpies de seguretat.
Servidors	Mòduls per administrar els diferents serveis que es poden executar en l'equip. Alguns del més corrents són: Samba, Postix, OpenSSH, Squid, etc.

TAULA 2.13 (continuació)

Pestanya	Funcions
Altres	Permet gestionar fitxers, comprovar el funcionament dels serveis, instal·lar/desinstal·lar programari i fins i tot establir connexions Telnet/SSH des del mateix navegador al servidor.
Xarxa	Configurar tots els paràmetres de xarxa de l'equip i els seus adaptadors. A més, permet configurar, entre d'altres, el tallafocs o la monitorització dels adaptadors de xarxa.
Maquinari	Permet gestionar tot el maquinari: impressores, particions, RAID, gestor d'engegada GRUB, etc.
Cluster	Permet gestionar de manera unificada un grup d'equips que executen Webmin.

2.4 Tendències actuals de l'accés i administració remota d'equips

El món globalitzat i intercomunicat mitjançant Internet ha fet possible pensar en els programes d'aplicació com a serveis disponibles no en el propi ordinador sinó en el "núvol", entès com a metàfora de la xarxa global.

Quan es parla de computació en el núvol es refereix a l'accés a recursos i serveis des de qualsevol lloc, fer servir programari específic d'aplicació de forma remota mitjançant la interfície del mateix navegador d'Internet i emmagatzemant les dades en servidors externs.

Tot això fa que l'accés i l'administració remota prenguin una nova dimensió on els recursos, programes, potència de càlcul, sistemes d'emmagatzematge, etc. poden estar distribuïts i intercomunicats en qualsevol lloc de la xarxa.

El projecte eyeOS va néixer de la mà d'un grup de joves programadors d'Olesa de Montserrat (Baix Llobregat) l'agost de 2005 i ha obtingut un gran èxit i reconeixement internacional.

Dos exemples que il·lustren aquesta tendència:

- **Chrome Remote Desktop**: Es tracta d'una extensió de Chrome, el navegador de Google que permet accedir a l'escriptori des de qualsevol lloc. L'usuari pot connectar-se i gestionar diferents equips, sigui quina sigui la seva plataforma i el seu sistema operatiu, mentre disposin de navegador Chrome. Per tant, no cal la instal·lació de cap programa addicional.
- **eyeOS** és un escriptori virtual, multiplataforma i de codi lliure que inclou tota la estructura d'un sistema operatiu i algunes aplicacions ofimàtiques. El continguts, arxius i programes que conformen aquest sistema operatiu es troben en servidors privats o públics a la xarxa i es pot accedir als seus recursos des de qualsevol navegador web modern.

3. Administració de servidors d'impressió

L'administració dels serveis d'impressió ha estat sempre una tasca necessària en els processos associats als sistemes d'informació. El resultat dels càlculs i la manipulació de les dades per obtenir informació útil havien de traslladar-se, en algun moment, a un suport definitiu i directament interpretable pels usuaris: el suport de paper. En aquest sentit, l'evolució constant de les necessitats dels usuaris i de la tecnologia ens ha portat a sistemes que han de garantir la gestió de múltiples treballs d'impressió en un entorn multiusuari, la comunicació i transmissió en xarxa amb els protocols adequats, l'administració de cues d'impressió i el suport a una tecnologia d'impressió gràfica dels perifèrics cada vegada més sofisticada.

3.1 Sistemes d'impressió

En l'actualitat els sistemes d'impressió poden ser complexos de gestionar i administrar: treballs d'alta qualitat gràfica, impressores compartides, perifèrics distribuïts a la xarxa, impressores de molts tipus i tecnologies, diferents opcions de ports de connexió, cues d'impressió, prioritats, seguretat, sistemes operatius heterogenis, etc. Tota aquesta gestió fa necessària la implementació de sistemes d'impressió d'una certa complexitat que permetin una adequada administració d'aquest servei bàsic.

3.1.1 Una mica d'història

A l'inici de l'era dels ordinadors personals, la tasca d'impressió era molt simple. Les impressores eren gairebé totes matricials, es comunicaven amb l'ordinador majoritàriament pel port paral·lel i només acceptaven caràcters de text. Així, imprimir un arxiu de text en un sistema Unix es podia fer senzillament redreçant l'arxiu al dispositiu que representava el port d'impressió, generalment lp0:

```
1 $ cat mitext.txt > /dev/lp0
```

Quan els ordenadors IBM PC van aparèixer el 1981, van fer servir el mateix model d'Unix. Les aplicacions incloïen informació sobre els codis de control de les impressores més populars i conegudes i l'usuari indicava a cada aplicació quins codis d'escapada havia de fer servir per a una impressora particular.

Codi d'escapada d'impressora

Els codis d'escapada són determinades seqüències de caràcters que s'insereixen en els textos a imprimir i són interpretats per la impressora com a ordres per activar o desactivar alguna funció, per exemple la lletra negreta o itàlica, fer un salt de línia o de pàgina, etc.

Amb l'aparició dels sistemes operatius gràfics, primer Apple Mac OS i després Microsoft Windows, es va canviar la filosofia d'impressió, ja que es va abstraure la interfície d'impressió de les diferents aplicacions. Així, els programes d'aplicació

només han d'indicar al sistema d'impressió del sistema operatiu on i què imprimir, i és aquest sistema d'impressió el que trasllada la petició correctament formatada a la sortida de la impressora seleccionada.

Aquests sistemes d'impressió havien d'acceptar gràfics, fonts tipogràfiques i imatges, i van aparèixer tecnologies d'impressió com les làser o les d'injecció, juntament amb llenguatges de descripció de pàgines com PostScript, per donar resposta a aquestes noves necessitats gràfiques. També la interconnectivitat i les xarxes van aportar-hi complexitat i la necessitat de protocols per comunicar-se amb impressores i servidors d'impressió remots gestionant prioritats, usuaris i cues de treballs.

En l'actualitat, els sistemes operatius moderns, i entre ells les famílies Unix/Linux, han de resoldre i incorporar les eines d'administració per gestionar les necessitats d'impressió en aquests entorns de treball complexos.

3.1.2 Dades, interfícies i protocols

El procés d'impressió consisteix bàsicament a transmetre una sèrie de **dades** textuals, gràfiques, imatges, etc. des de l'ordinador que les ha processat fins a un determinat perifèric d'impressió. Per establir una comunicació amb qualsevol perifèric cal un canal de comunicació física, es a dir una **interfície** electrònica que permeti transmetre els senyals que porten la informació. Aquests canals de comunicació poden ser ports i bussos de comunicació local o bé una infraestructura de xarxa.

En el cas d'intercomunicar ordinadors i perifèrics mitjançant una xarxa compartida també cal tenir present la necessitat d'implementar uns **protocols** de comunicació adequats per assolir les funcionalitats necessàries en qualsevol sistema d'impressió.

Dades

En els primers sistemes d'impressió, els perifèrics emprats eren impressores de caràcters (*line printer*), que rebien informació alfanumèrica caràcter a caràcter. En aquest entorn, les dades acostumaven a estar codificades en ASCII (*American standard code for information interchange)* de 7 bits, que permetia imprimir els números, els caràcters anglesos i alguns signes de puntuació, a més d'un grapat de caràcters de control que gairebé totes les impressores reconeixien, com ara el salt de línia o el salt de pàgina.

Amb la implementació del codi ASCII de 8 bits es van poder imprimir nous caràcters particulars d'altres idiomes (vocals accentuades, lletra *ñ*, etc.), així con determinats caràcters semigràfics molt senzills.

Per fer el pas següent i poder imprimir dades més complexes com gràfics, imatges, diferents tipografies de lletra, etc. es van crear uns **llenguatges de descripció de**

pàgina. Aquest llenguatges permeten definir el format de la pàgina, la col·locació del text, els mapes de bits i els objectes vectorials per obtenir una qualitat d'impressió òptima.

En aquest cas, l'ordinador envia a la impressora un arxiu amb la descripció de les pàgines a imprimir, aquesta interpreta les instruccions i construeix una representació de la pàgina en forma de mapa de bits que la tecnologia d'impressió (làser, injecció, etc.) s'encarrega de passar al paper.

Els llenguatges de descripció de pàgina més emprats són:

- **PostScript:** Estàndard de facto desenvolupat per Adobe. La majoria d'impressores modernes entenen PostScript i, en tot cas, es poden fer servir filtres i traductors per a la majoria d'impressores.
- **PCL**(*printer command language*)**:**És un llenguatge de descripció de pàgines desenvolupat per Helwett-Packard com a protocol de comunicació per a les seves impressores làser i d'injecció (vegeu la figura 3.1). És una mica més senzill i ràpid, i necessita menys memòria, però no és tan potent com Postcript que, en teoria, pot arribar a generar impressions de més qualitat que PCL.

FIGURA 3.1. Impressora làser Hewlett-Packard amb llenguatge PCL

Interfícies de comunicació

Per transferir dades a qualsevol perifèric en general i a les impressores en particular ens cal una interfície física de comunicació que permeti la transmissió de bits d'informació per mitjans electrònics, òptics o de radiofreqüència.

Aquestes interfícies poden transmetre grups de bits simultàniament, paraula a paraula en el cas de les interfícies paral·leles, o bé bit a bit en el cas de les interfícies de sèrie.

Aquesta connexió física s'anomena genèricament **port**, encara que, en rigor, els ports són interfícies que connecten un sistema informàtic directament amb un determinat perifèric, mentre que els anomenats **bussos** són interfícies de comunicació que poden ser compartides per més d'un perifèric alhora.

Els tipus més habituals d'interfícies de comunicació per a la connexió d'impressores en l'àmbit local són:

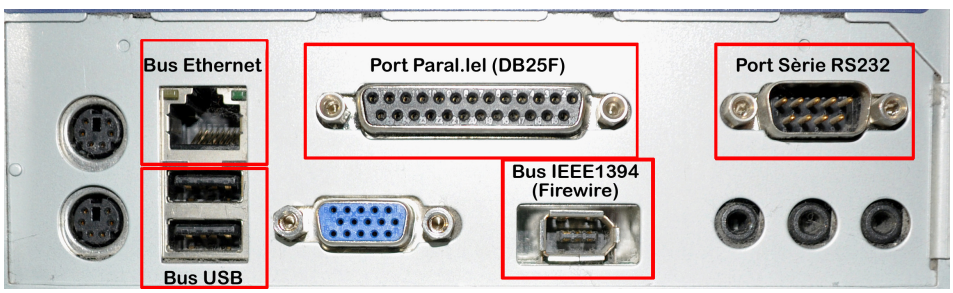
- **Port paral·lel**: també anomenat ***port Centronics***. Implementat en el disseny del primer IBM PC, és un port de comunicació paral·lel bidireccional de 8 bits de dades més un conjunt de línies de control. Aquest port va ser la interfície d'impressió local per excel·lència fins a l'aparició del port USB.
- **Port de sèrie RS232**: interfície de sèrie habitualment implementada amb connectors DB9M (nou pins mascle), dissenyada per a distàncies de fins a 15 metres i velocitats baixes, d'uns 20 Kbps (kilobits per segon). Encara avui dia es fa servir en alguns tipus d'impressores especials de tiquets i de codis de barres.
- **Bus USB** (*universal serial bus*): aquesta interfície de tipus sèrie i topologia de bus permet la connexió simultània de diferents perifèrics que comparteixen el canal de comunicació. Aquest tipus d'interfície permet la connexió de perifèrics en calent (*hot swap*) i el seu reconeixement i configuració automàtica (*plug and play*). En la seva versió 2.0 arriba a una velocitat de transmissió de 480 Mbps mentre que en la versió 3.0 pot aconseguir una velocitat de fins a 3.200 Mbps. En l'actualitat és la interfície més emprada en la connexió de perifèrics de tot tipus (ratolins, teclats, discos externs, etc.), incloses les impressores.
- **Bus FireWire:** aquesta interfície definida en la norma IEEE1394 és coneguda com a **FireWire** en l'entorn d'Apple i com a **i.link** en l'entorn de Sony. Té característiques semblants a l'USB en tant que és una interfície de sèrie de topologia bus, i les seves diferents versions permeten velocitats de transmissió que van dels 400 Mbps als 3.200 Mbps. La diferència principal és que el bus USB requereix sempre un ordinador *host*, mentre que FireWire pot funcionar de punt a punt (*peer-to-peer*), habilitant el diàleg de dos perifèrics entre si, sense la intervenció d'un ordinador. Algunes impressores disposen d'aquesta interfície, encara que el seu ús més habitual el trobem en perifèrics multimèdia, com ara càmeres de vídeo digital.

Plug and play

El sistema *plug and play* (endollar i funcionar) descriu aquells dispositius o interfícies que permeten reconèixer automàticament el maquinari d'un sistema i configurar-ho automàticament, i resolen els possibles conflictes en l'ús dels recursos. Alguns sistemes *plug and play* que es fan servir actualment són les interfícies USB, FireWire o les ranures d'expansió PCI i PCI Express.

Podem veure els connectors d'aquestes interfícies a la figura 3.2.

FIGURA 3.2. Principals interfícies de connexió per a impressores



Fora de l'àmbit de la connexió local i directa, la generalització de les infraestructures en xarxa **Ethernet** ha tingut també com a conseqüència la possibilitat de connectar les impressores directament a la xarxa com un node més. En aquest cas, a més de la connectivitat física Ethernet, usualment amb l'ús de connectors RJ45, s'ha de proveir el perifèric de la corresponent adreça IP i establir protocols de comunicació adients (IPP, SMB, HP Jetdirect, AppleTalk, etc.).

Protocols

Els protocols de comunicació descriuen el format dels missatges a intercanviar entre equips interconnectats per poder establir una comunicació i diàleg efectiu. Aquesta descripció inclou:

- **Sintaxi**: especifica com són i com es construeixen els missatges.
- **Semàntica**: descriu què significa cada ordre i la seva resposta.
- **Sincronització**: defineix la seqüència correcta de peticions i respostes.

Els protocols poden implementar-se per maquinari o per programa, poden incloure autenticació, detecció i correcció d'errors, i són necessaris en els diferents nivells OSI de la xarxa, des del nivell físic fins al nivell d'aplicació.

Els principals protocols de comunicació que es poden fer servir en l'àmbit dels sistemes d'impressió són el següents:

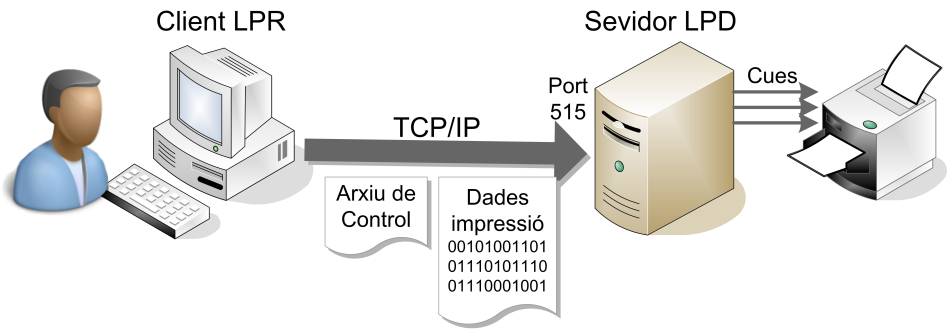
- Protocol LPD/LPR
- Protocol IPP
- AppSocket
- AppleTalk
- SMB/CIF

Protocol LPD/LPR

El Protocol LPD/LPR (*line printer daemon/line printer remote protocol*) és el protocol originalment implementat a la plataforma BDS Unix. Fa servir TCP/IP per establir connexions entre impressores i ordinadors en una xarxa (vegeu la figura 3.3). Aquest protocol treballa normalment escoltant peticions al port TCP 515 i consta de dos components:

- ***Line printer remote*(LPR)** és el terme per al procés client d'enviament de treballs d'impressió a una impressora o cua d'impressió. L'estació de treball que envia la tasca d'impressió és el client LPR.
- ***Line printer daemon*** es refereix al procés de recepció de treballs d'impressió des d'un client LPR. Les impressores o servidors d'impressió que reben els treballs d'impressió s'anomenen *servidors LPD*.

FIGURA 3.3. Impressió mitjançant el protocol LPD/LPR



Quan un usuari envia un document per imprimir, l'ordinador (LPR client) genera unt treball en un determinat format, per exemple, PostScript. Les dades d'impressió es composen d'un fitxer de dades amb el contingut real que es vol imprimir, i un fitxer de control que inclou la descripció de l'arxiu de dades, nom del treball, propietari, nombre de còpies a imprimir, cua de destí, etc. A continuació, el treball d'impressió s'envia a l'adreça IP del servidor LPD, que el rep habitualment a través del port TCP/IP 515. Es poden configurar i definir diverses cues al servidor d'impressió LPD, de manera que el fitxer de control del treball d'impressió ha d'incloure informació sobre quina cua ha de ser assignada.

Protocol IPP

El protocol d'impressió per Internet (IPP o *Internet printing protocol*) defineix extensions del conegut protocol HTTP (*hypertext transport protocol*) per donar suport als serveis d'impressió remots, configuració d'impressores i gestió de cues. A diferència d'altres protocols, IPP permet control d'accés, autenticació i xifrat per donar solucions d'impressió més completes i segures. És el protocol usat de forma nativa per CUPS i acostuma a fer servir el port 631.

HTTP

El protocol de transport d'hipertext (hypertext transport protocol) es fa servir per transferir pàgines web a través d'Internet.

AppSocket

Aquest protocol està basat en el protocol **Jetdirect** de Hewlett-Packard. Acostuma a treballar al port 9100 i es considera molt fiable, senzill i ràpid.

AppleTalk

És un conjunt de protocols de comunicació d'Apple que permet, entre altres coses, la comunicació amb impressores i servidors d'impressió

SMB/CIF

El servidor de blocs de missatges (*server message block*) i la seva evolució, el sistema d'arxius comú d'internet (*common Internet file system*), són protocols de xarxa de la capa d'aplicació del model OSI que permeten gestionar i compartir arxius i impressores entre nodes d'una xarxa. SMB fa servir el port TCP 445 i va ser ideat per IBM, però modificat i perfeccionat per Microsoft, que el fa servir en els seus sistemes operatius Windows. També hi ha implementacions del protocol en codi lliure per a sistemes Linux.

Samba

És la més popular implementació en codi lliure del protocol SMB/CIFS (entre d'altres) i permet la compartició d'arxius i impressores en xarxes heterogènies.

3.1.3 Descripció d'un sistema d'impressió

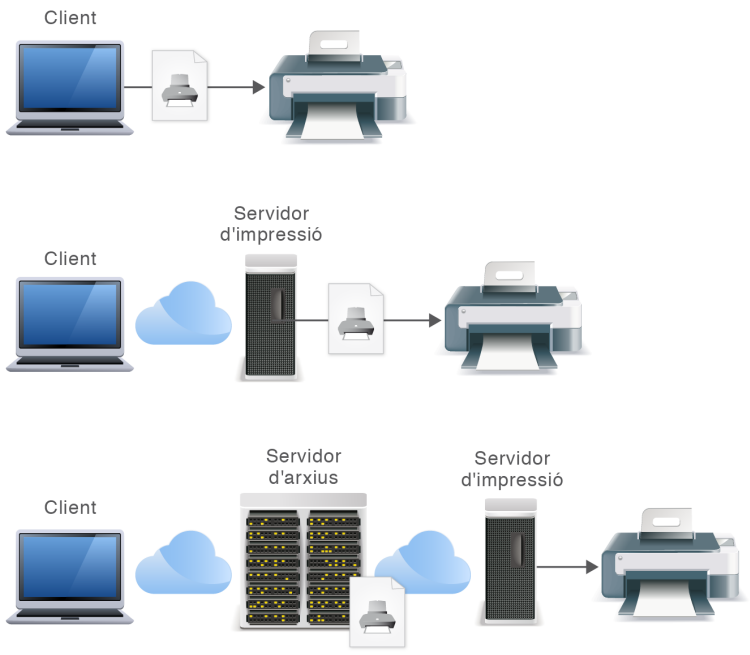
Imprimir pot semblar una tasca senzilla, però comporta una sèrie d'inconvenients i necessitats que els sistemes d'impressió actuals resolen mitjançant la implementació de diferents mòduls:

- El planificador de cues d'impressió (*spooler*)
- Els filtres
- Els controladors d'interfície o *backends*

El primer problema sorgeix perquè la majoria de les impressores no tenen memòria suficient per carregar un document complet. Aquest és un dels motius que fan necessari disposar de **cues d'impressió**i d'un sistema que les gestioni (*spooler*). El gestor de la cua monitoritza la impressora i envia el treball següent en el moment en què queda lliure. La ubicació de les cues d'impressió està determinada per la complexitat del sistema d'impressió. Les cues poden estar situades, tal com es mostra a la figura 3.4, a les localitzacions següents:

- L'ordinador client
- El servidor d'impressió
- Un servidor intermediari d'arxius

Figura 3.4. Sistemes d'impressió i ubicació de les cues d'impressió (spooler) en l'ordinador client, en el servidor d'impressió i en el servidor d'arxius

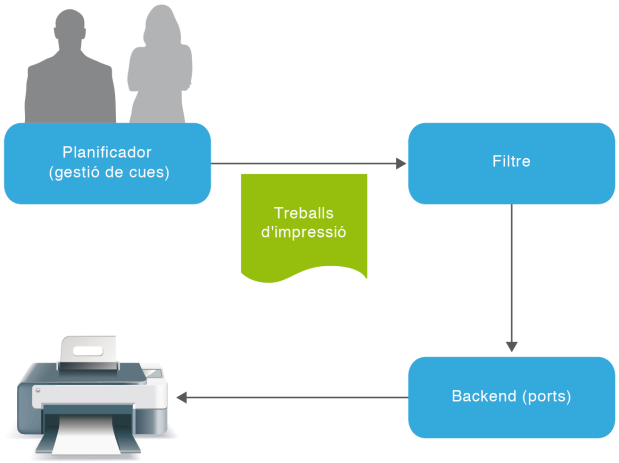


El següent problema a resoldre és el gran nombre de models d'impressora disponibles al mercat que fan servir seqüències de control diferents, és a dir que

parlen llenguatges propis que requereixen una traducció. El **filtres** són les eines, entre el planificador i les impressores, que fan aquesta traducció entre les dades d'entrada i el llenguatge que entén la impressora.

El darrer pas de la cadena és l'enviament de dades al perifèric d'impressió mitjançant algun dels canals i interfícies físiques que s'han descrit. Cadascuna d'aquestes interfícies requereix un controlador de dispositiu o ***backend*** que gestiona i controla el port de comunicació físic (USB, paral·lel, RS232...) o bé la connexió de xarxa (IPP, AppSocket, etc.). El concepte modular del controlador *backend* permet afegir nous tipus d'interfícies segons avanci la tecnologia sense afectar la resta de l'arquitectura.

FIGURA 3.5. Estructura d'un sistema d'impressió



A la figura 3.5 es mostren esquemàticament el mòduls que descriuen un sistema d'impressió modern.

3.1.4 Sistemes d'impressió Unix/Linux

La tasca d'impressió a Unix s'ha realitzat històricament fent servir un d'aquest dos sistemes de gestió d'impressió: el sistema d'impressió d'AT&T (Unix System V) i el dimoni d'impressores en línia (LPD) de Berkeley. Aquests sistemes van estar dissenyats als anys 70 amb l'únic propòsit d'imprimir text en impressores de caràcters. Per aquest motiu han aparegut noves opcions amb més prestacions, com ara LPRng o el sistema CUPS, encara que s'ha intentat sempre conservar la compatibilitat i sintaxi de les ordres de consola originals.

A continuació s'enumeren breument els principals sistemes d'impressió a Unix/-Linux:

/etc/pintcap

Aquest és l'arxiu de configuració del sistema d'impressió *lpd* que conté el registre i definició de totes les impressores del sistema, amb indicació de l'àlies, nom del dispositiu, directori de la cua de treballs, filtres, etc.

- **LPD, sistema d'impressió Berkeley**: aquest és el tradicional gestor d'impressió d'Unix de la plataforma Unix BSD. És controlat pel dimoni d'impressores de línia (*line printer daemon*) situat a /usr/sbin/lpd i fa servir

el protocol LPD/LPR. Els clients es comuniquen amb el dimoni mitjançant el dispositiu /dev/printer i fan servir l'arxiu de configuració /etc/printcap per determinar el directori de la cua de treballs d'impressió.

- **LPRng**: LPR és una implementació en codi lliure que millora el sistema d'impressió BDS però mantenint la compatibilitat, ja que també fa servir el dimoni LPD. Inclou noves funcionalitats en el control de la cua de treballs i les funcions de servidors d'impressió en xarxa.

- **CUPS:** el sistema d'impressió comú d'Unix és un potent i complet gestor d'impressió basat en el protocol IPP (*Internet printing protocol*) i integrant PostScript com el llenguatge de definició de pàgines estàndard. La majoria de distribucions Linux ja incorpora CUPS com a sistema d'impressió per defecte.

3.2 Sistema d'impressió comú d'Unix (CUPS)

CUPS (*common Unix printing system*) és un sistema d'administració i gestió d'impressió portable i extensible per als sistemes operatius de la família Unix/Linux. Va ser desenvolupat per Easy Software Products i alliberat com a programari lliure amb llicència GNU/GPL.

CUPS fa servir el protocol IPP, funciona amb LPD/LPR i proporciona accés a clients Linux, Windows i Mac OS.

3.2.1 Descripció del sistema CUPS

De manera semblant a altres sistemes d'impressió, CUPS està dissenyat al voltant d'un procés de planificació d'impressió (*scheduler*) que distribueix els treballs d'impressió, processa les ordres administratives i proporciona informació i monitorització de l'estat de les impressores i els treballs. Aquest procés correspon al dimoni anomenat **cupsd**, que s'acostuma a carregar en l'arrencada del sistema treballant en segon pla i al que farem referència com a *planificador*.

El planificador o *scheduler* gestiona les peticions que arriben tot seguint el protocol IPP. Aquest treballs poden provenir de l'ordinador local o bé de clients en xarxa de diverses plataformes (Linux, Windows, Mac OS) i generen dos arxius, un amb les dades pròpiament dites, i un arxiu de control amb informació sobre el treball d'impressió. Aquests arxius es guarden al directori `/var/spool/cups`.

Els treballs d'impressió poden incorporar text, gràfics i imatges vectorials o de mapa de bits. Aquests tipus de dades són reconegudes fent servir les entrades de l'arxiu `/etc/cups/mime.types` (o bé `/var/share/cups/mime/mime.types`).

Per descriure aquests elements, CUPS fa servir de forma estàndard el llenguatge de programació de gràfics **PostScript** desenvolupat per Adobe. Els filtres de conversió en funció del tipus de dades estan determinats a l'arxiu `/etc/cups/mime.convs` (o bé a `/var/share/cups/mime/mime.convs`).

El planificador gestiona les impressores disponibles a la xarxa local i els envia els treballs d'impressió fent servir els filtres i controladors d'interfície (*backends*) apropiats, que estan situats a `/usr/lib/cups/backend`.

```
$ ls /usr/lib/cups/backend

behdnssd hpfaxipp parallel serial socket
cups-pdf hp httplpd scsi snmp usb
```

Les impressores compatibles amb PostScript poden interpretar directament les dades rebudes del planificador sense necessitar un filtre específic. En el cas d'impressores que no siguin PostScript cal fer la traducció al llenguatge objectiu de la impressora mitjançant els filtres subministrats o amb altres aplicacions com ara **Ghostscript**, programari també de codi lliure.

Ghostscript

És un programa intèrpret d'arxius PS (PostScript) i PDF (*portable document format*) que permet representar-los en pantalla i traduir-los de manera que puguin ser impresos en una impressora amb capacitat gràfica.

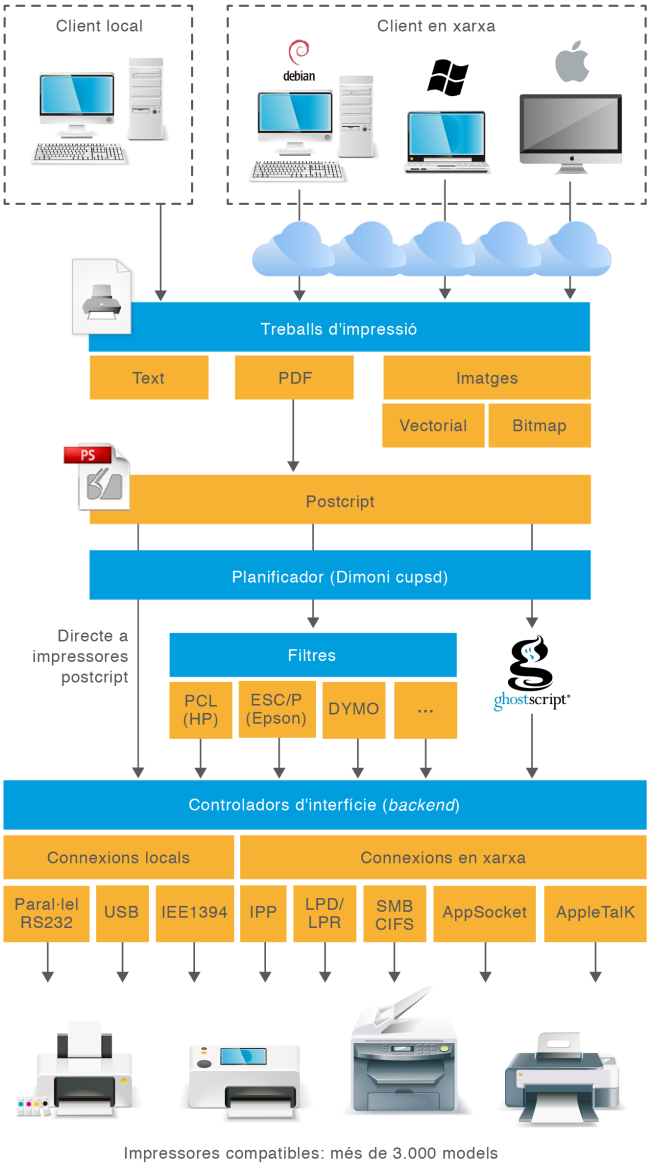
Un cop el treball ha estat transferit a la impressora, el planificador esborra l'arxiu de dades de la cua a `/var/spool/cups`, però deixa l'arxiu de control, on queden numerats correlativament (des del primer, `/var/spool/cups/c00001`).

Altres característiques addicionals proporcionades per CUPS són:

- **Servei de directori:**anomenat *examinador d'impressores*(*printer browsing*). Els clients poden trobar automàticament i fer servir impressores de qualsevol servidor de la xarxa. Si aquesta funció està activada, el servidor distribueix per la xarxa la informació de les impressores disponibles mitjançant missatges de difusió general (*broadcasting*) cada poc temps.
- **Classes**: CUPS també permet definir **classes**, que són grups d'impressores de característiques semblants. Els treballs d'impressió enviats a una classe són redreçats a la primera impressora disponible d'aquella classe. Aquest esquema també permet habilitar seguretat i repartiment de càrregues definint la mateixa impressora en diversos servidors.
- **Suport clients LPD**: fent servir un minidimoni (cups-lpd) que atén totes les tasques d'impressió rebudes per LPD i les redreça al subsistema CUPS tot convertint-les a protocol IPP.
- **Administració web**: aquest servidor també actua com a servidor web per a la documentació, monitorització d'estat i administració del sistema.
- **Impressió des de línia d'ordres**: CUPS entén directament molts tipus diferents d'arxius, incloent text, PostScript, PDF i arxius d'imatge. Això li permet imprimir aquest arxius des de les aplicacions d'usuari o directament des de la línia d'ordres.

La figura 3.6 mostra l'esquema descriptiu del sistema d'impressió CUPS.

FIGURA 3.6. Esquema descriptiu del sistema d'impressió CUPS



3.2.2 Instal·lació i configuració de CUPS

Instal·lació de CUPS

La instal·lació de CUPS no suposa cap complicació. CUPS està incorporat per defecte en la majoria de distribucions però, en qualsevol cas, només cal instal·lar el paquet:

```
1 $ apt-get install cups
```

Per instal·lar el client de CUPS a la plataforma Linux:

```
1 $ apt-get install cups-client
```

En principi CUPS està pensat perquè tant els clients com el servidor funcionin amb el mateix protocol IPP. Si els clients fan servir LPD/LPR o LPRng, cal instal·lar un dimoni de compatibilitat (cups-lpd) disponible amb el paquet:

```
$ apt-get install cups-bds
```

Convé també tenir instal·lats els paquets que contenen els arxius PPD (*PostScript printer description*) que Debian distribueix en dos paquets: openprinting-ppds per a impressores PostScript i foomatic-filters-ppds per a impressores no PostScript.

Tots aquests paquets esmentats solen estar ja instal·lats en la configuració servidor de Debian.

```
$ apt-get install openprinting-ppds foomatic-filters-ppds
```

Un altra utilitat que sol incorporar l'escriptori GNOME és system-config-printer, que permet gestionar el servei CUPS. En cas de necessitar instal·lació:

```
$ apt-get install system-config-printer
```

Finalment, és interessant també instal·lar cups-pdf, una impressora virtual per imprimir documents a un arxiu PDF, ja que resulta molt útil per fer proves d'impressió i conversió de documents a format PDF.

PDF (portable document format)
És un format obert d'emmagatzematge de documents desenvolupat per Adobe que pot incorporar text i elements multimèdia (imatges bitmap i vectorials, so, vídeo, enllaços, etc.). És un format independent del dispositiu i especialment dissenyat per a la visualització i impressió de documents.

```
$ apt-get install cups-pdf
```

Configuració i administració de CUPS

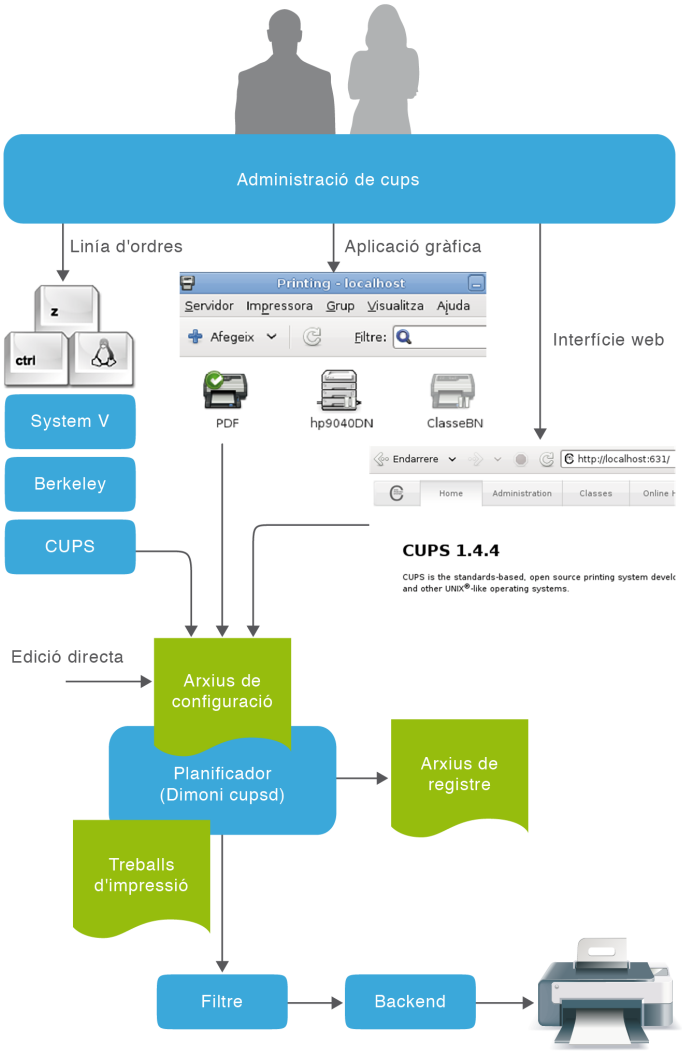
La configuració i administració de CUPS consisteix bàsicament en assignar permisos i accés, definir i configurar impressores, classes i cues d'impressió i gestionar els diferents treballs d'impressió. Aquestes tasques les podem fer de diferents maneres:

- Modificant directament els arxius de configuració
- Mitjançant ordres de consola (SystemV, BDS o pròpies de CUPS)
- Mitjançant una aplicació gràfica nativa (ex. sytem-config-printer)
- Mitjançant la interfície web que proporciona CUPS (`http://localhost:631`)

CUPS també genera una sèrie d'arxius on enregistra informació rellevant per al control i seguiment del seu funcionament (arxius de registre o *log*).

A la figura 3.7 es mostra un esquema amb les alternatives de configuració i administració de CUPS.

FIGURA 3.7. Alternatives per a l'administració i configuració de CUPS



Arxius de configuració del servidor CUPS

CUPS té un sistema d'administració basat en diferents arxius de configuració que s'enumeren amb detall a continuació:

- **Configuració del servidor:/etc/cups/cupsd.conf** és l'arxiu principal que centralitza la configuració del sistema d'impressió. De característiques semblants a l'arxiu de configuració del servidor Apache, defineix les característiques del servidor CUPS.

- **Definició d'impressores: /etc/cups/printers.conf** conté la llista d'impressores locals definides amb informació de la cua d'impressió associada o de com està connectada i amb quina interfície.

- **Arxius PPD**(*PostScript printer description*): cada cua d'impressió té el seu propi arxiu de configuració situat al directori **/etc/cups/ppd.**Aquests arxius contenen les opcions de configuració de la impressora (mida i orientació del paper, resolució, escala...).

Cues d'impressió

Una impressora pot tenir associada més d'una cua d'impressió. Per exemple, en les impressores en color és útil disposar de dues cues, una per als treballs en color i una altra per als treballs en blanc i negre.

- **Classes d'impressores:/etc/cups/classes.conf**conté la llista de classes d'impressores definides localment.
- **Tipus MIME**: **/etc/cups/mime.types**(o bé /usr/share/cups/mime/-mime.types)indica els tipus d'arxius MIME admesos (text/plain, application/postcript...).
- **Regles de conversió**: **/etc/cups/mime.convs**(o bé /usr/share/cups/mime/-mime.convs) defineix quin o quins filtres estan disponibles per convertir arxius d'un format a un altre. Els filtres estàndard admeten text, arxius PDF, PostScript i molts tipus de formats d'imatge.

Tipus MIME

Els tipus MIME (*multipurpose Internet mail extensions*) són unes convencions que descriuen determinats tipus d'arxius (text, àudio, vídeo, etc.) per facilitar el seu intercanvi per Internet.

Arxius de registre

El planificador manté tres arxius bàsics de registre (*log files*), que normalment es guarden al directori /var/log/cups i que són els següents:

- **access_log**: enregistra totes les peticions HTTP i IPP processades pel planificador de cues. Un exemple d'una entrada d'aquest arxiu:

```
192.168.56.11 - - [31/Jan/2012:01:52:58 +0100] "POST /printers/PDF HTTP/1.1"
    200 873 Print-Job successful-ok
```

- **error_log**: conté tots els missatges d'error del planificador per poder fer una anàlisi i seguiment dels problemes. Exemple:

```
E [30/Jan/2012:23:45:45 +0100] Unable to bind socket for address ::1:631 -
    Cannot assign requested address.
```

- **page_log**: enregistra un llistat de totes les pàgines que s'han imprès.

```
DeskJet root 2 [20/May/1999:19:21:05 +0000] 1 1 acme-123 localhost myjob letter
     one-sided
```

Els arxius de registre són gestionats pel mateix CUPS, que en controla la rotació quan la seva mida sobrepassa el màxim configurat, que per defecte és 1 Mb.

logrotate

En el cas de configurar la mida màxima dels arxius de registre a 0, el planificador deixa d'encarregar-se de la seva rotació i es poden fer servir per al manteniment d'aquests arxius altres utilitats de Linux com ara *logrotate*.

Configuració manual del servidor CUPS

El comportament del servidor CUPS es configura mitjançant les directives contingudes a l'arxiu `/etc/cups/cupsd.conf`. Aquest arxiu de configuració té la mateixa sintaxi que l'arxiu principal de configuració del servidor HTTP d'Apache.

Per exemple, la secció **location** especifica les directives de control d'accés i les opcions d'autenticació per al directori o recurs HTTP especificat. Aquests àmbits d'actuació corresponen a cada un dels directoris que s'enumeren a la taula 3.1 i que tenen correlació amb les diferents seccions de la interfície web d'administració.

Taula 3.1. Descripció dels directoris configurables a la secció location

Directori	Descripció
`/`	Directori general de totes les operacions administratives
`/admin`	Directori de totes les operacions administratives (per exemple afegir i esborrar impressores)
`/admin/conf`	Directori d'accés als arxius de configuració de CUPS (cupsd.conf, client.conf, etc.)
`/admin/log`	Directori d'accés als arxius de registre de CUPS (access_log, error_log, page_log)
`/classes`	Directori de les classes d'impressores
`/classes/name`	Directori d'una determinada classe d'impressores
`/jobs`	Directori dels treballs d'impressió
`/printers`	Directori de les impressores
`/printers/name`	Directori d'una determinada impressora
`/printers/name.ppd`	Directori d'un determinat PPD (Printer description file)

Cada secció location defineix l'accés a un directori determinat i als seus subdirectoris. En l'exemple següent la directiva ***allow*** configura l'accés administratiu a tot el directori arrel de CUPS que, en aquest cas, només es permet des de la màquina local (127.0.0.1).

```
<Location />

Order Deny,Allow
Deny From All

Allow From 127.0.0.1

</Location>
```

En aquest mateix exemple la directiva ***order*** configura el comportament de la resta de directives *allow* i *deny* amb les possibilitats següents:

- **Order Allow,Deny**: permet l'accés a totes les IP excepte aquelles que apareixen a la directiva deny.
- **Ordre Deny,Allow**: només permet l'accés a les IP llistades en directives allow.

No només podem determinar des d'on es pot accedir, sinó el nivell d'autenticació necessari mitjançant la directiva ***require***, que té les opcions següents:

- ***group***: per indicar a continuació els noms del grups autoritzats.
- ***user***: tot indicant a continuació els noms dels usuaris autoritzats o bé dels grups que han d'estar precedits del símbol arrova (@). Els grups poden ser grups predeterminats (@SYSTEM, @OWNER...) o bé grups específics (@NomGrup).

- ***valid-user***: permet l'accés a qualsevol usuari correctament identificat.

L'opció per defecte és que no es requereix autenticació.

En aquest segon exemple, la secció **location** configura la impressió i gestió d'impressores. Les directives ***allow*** permeten accedir només des del propi ordinador (*localhost*) o des d'algun ordinador de la xarxa de classe C 192.168.1.(1-254). Les directives ***require*** permeten la validació de l'usuari Joan, de qualsevol usuari dels grups alumnes i professors i del grup predeterminat de sistema.

```
<Location /printers>
Order Deny,Allow
Deny From All

Allow From 127.0.0.1
Allow From 192.168.1.*

Require group alumnes
Require user joan
Require user @professors
Require user @SYSTEM
</Location>
```

Un altra directiva important és ***listen*** que permet definir les adreces i ports que són escoltats a l'espera d'una connexió IPP. En la definició de la IP es pot fer servir el comodí (*). Per defecte s'accepten connexions de la màquina local pel port 631.

```
Listen 127.0.0.1:631
```

Es presenta un resum de les possibles directives a la taula 3.2.

TAULA 3.2. Resum d'algunes directives de l'arxiu /etc/cups/cupsd.conf

Directiva	Descripció
Location	Defineix una secció per a l'assignació de permisos d'accés i nivell d'autorització a un directori determinat.
Allow / ***Deny***	Dins d'una secció **location** especifica les adreces IP que poden accedir a un directori determinat del servidor.
Require	Defineix el conjunt d'usuaris que poden accedir al servei.
Listen	Indica les adreces/ports escoltats pels servidor.
AccesLog ***ErrorLog*** ***PageLog***	Especifica el nom i adreça absoluta de la ubicació dels arxius de registre.
DefaultLanguage	Especifica el llenguatge per defecte que faran servir les connexions del clients (opcions *de*, *en*, es, *fr*, *it*).
Loglevel	Determina el nivell d'enregistrament de l'arxiu error_log.
MaxClients	Defineix el nombre màxim de clients simultanis.
MaxJobs ***MaxJobsPerPrinter*** ***MaxJobsPerUser***	Permet limitar el nombre màxim de treballs d'impressió totals, per impressora o per usuari.
Browsing On ***Browsing Off***	Activa/desactiva l'enviament de paquets UDP broadcast per informar a la xarxa de les impressores disponibles.

Cal recordar que perquè els canvis a l'arxiu `/etc/cups/cupsd.conf` tinguin efecte és necessari reiniciar el servei:

```
/etc/init.d/cups restart
```

Trobareu la referència de totes les directives de configuració del servidor CUPS a la pàgina del manual associada:

```
man cupsd.conf
```

A més de les pàgines de *man,* hi ha una extensa descripció i exemples de totes les directives de /etc/cups/cupsd.conf al manual de referència oficial que trobareu a la secció "Adreces d'interès".

Configuració dels clients CUPS

Com ja s'ha comentat, CUPS pot treballar amb clients de diferents plataformes i protocols, però disposa del seu propi client amb protocol IPP que es configura mitjançant l'arxiu **/etc/cups/client.conf** o bé **~/.cups/client.conf** segons es vulgui configurar per a tots els usuaris o per a un usuari concret. Aquest arxiu només conté dues directives molt simples:

- ***Encryption***: defineix la configuració de xifrat del client (never-IfRequested-Required-Always). Per defecte és IfRequested.
- ***ServerName***: indica el nom o adreça del servidor que rebrà les sol·licituds del client.

Exemple de contingut de l'arxiu client.conf:

```
Encryption IfRequested
ServerName 192.168.56.10:631
```

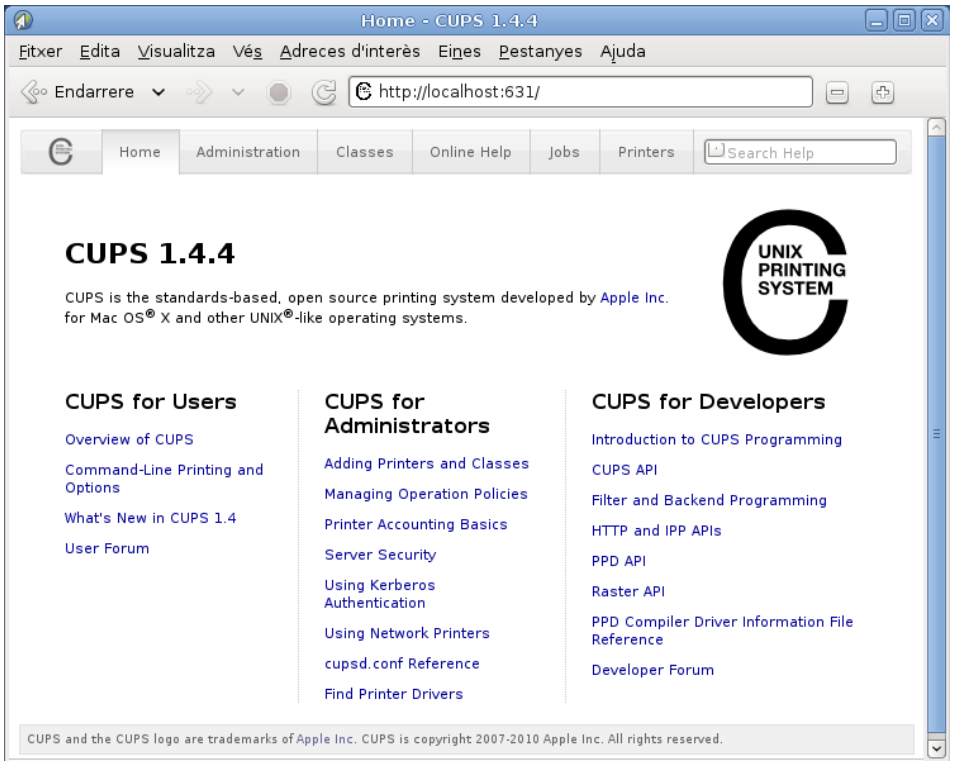
Un altre arxiu de configuració de la part client és **/etc/printcap**, que és genera i actualitza automàticament a partir de l'arxiu /etc/cups/printers.conf. Aquest arxiu és important, ja que el seu contingut és consultat per les aplicacions d'usuari (com per exemple OpenOffice) per generar el llistat d'impressores disponibles en les seves opcions i menús d'impressió.

Administració web de CUPS

El dimoni planificador (*scheduler*) de CUPS és una aplicació de servidor que gestiona peticions HTTP. A més d'atendre les peticions d'impressió rebudes pel protocol IPP, el planificador també actua com un complet servidor web per oferir documentació, monitorització de l'estat de les cues i treballs i administració del sistema d'impressió.

Per defecte s'accedeix a aquest servidor web pel port 631. Així, en la màquina local, només cal obrir qualsevol navegador indicant l'adreça `http://localhost:631` perquè aparegui la pantalla de la figura 3.8.

FIGURA 3.8. Pantalla inicial del servei d'administració web de CUPS



A la primera pàgina hi surten enllaços per arribar a diferents fonts de documentació per a usuaris, administradors i desenvolupadors. La secció que ho resumeix tot és la d'administració (vegeu la figura 3.9), on hi ha enllaços a les altres seccions (impressores, classes i treballs d'impressió) i que permet fer les tasques següents:

Kerberos

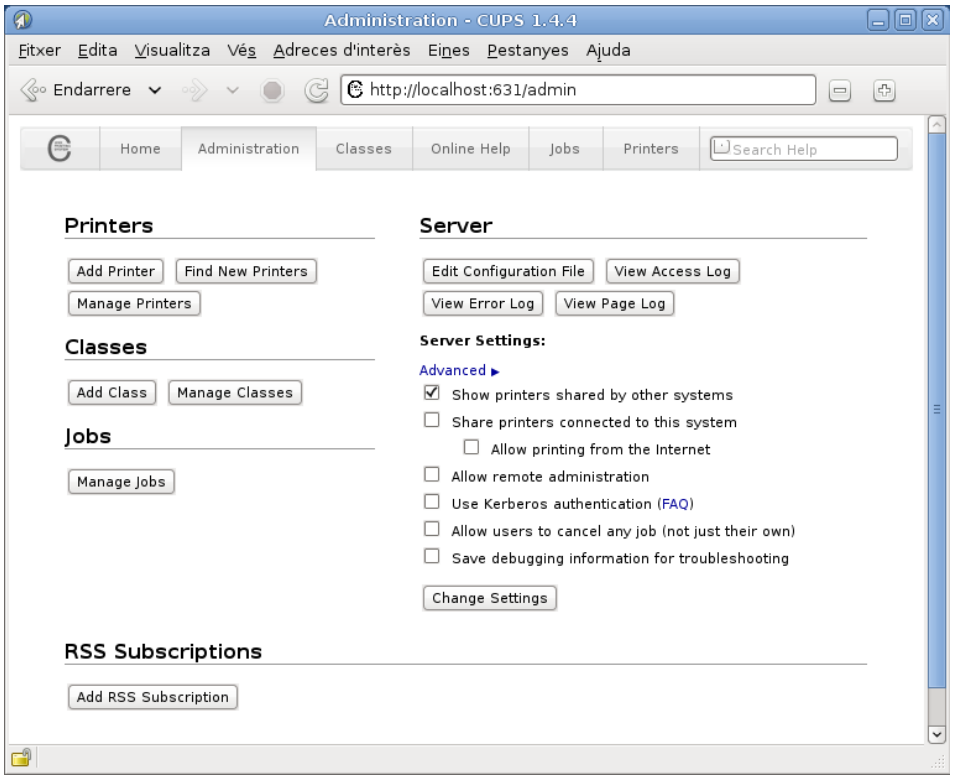
Kerberos és un protocol d'autenticació de xarxes d'ordinadors desenvolupat per l'Institut Tecnològic de Massachusetts (MIT) que permet a dos ordinadors, tant client com servidor, verificar mútuament la seva identitat en una xarxa insegura.

- **Configurar el servidor**: Hi han algunes opcions del servidor que es poden configurar automàticament marcant la casella corresponent.
  - Veure les impressores compartides per altres ordinadors de la xarxa.
  - Compartir les impressores locals.
  - Permetre l'administració des d'ordinadors remots.
  - Fer servir el sistema d'autenticació Kerberos.
  - Permetre als usuaris cancel·lar treballs d'altres usuaris.
  - Configurar en nivell d'enregistrament dels arxius de registre.
- **Edició directa de l'arxiu de configuració**: per ajustar de manera més detallada i completa les opcions de configuració es pot accedir a l'edició directa de les directives contingudes a l'arxiu /etc/cups/cupsd.conf. Un cop fetes les modificacions, el mateix sistema s'encarrega de reiniciar el servei.
- **Visualitzar els arxius de registre**: tant els de accés com els de missatges d'error i la llista de pàgines impreses.
- **Administrar impressores**: gestionar les impressores del sistema, trobar automàticament aquelles que estiguin connectades o bé donar-les d'alta manualment.

- **Administrar classes**: gestionar les classes d'impressores o gestionar les existents.

- **Administrar treball d'impressió**: visualitzar els treballs d'impressió (*jobs*), tant els finalitzats com els que estan en curs que es poden gestionar (cancel·lar, aturar, canviar de cua...).

FIGURA 3.9. Pantalla principal d'administració del servidor

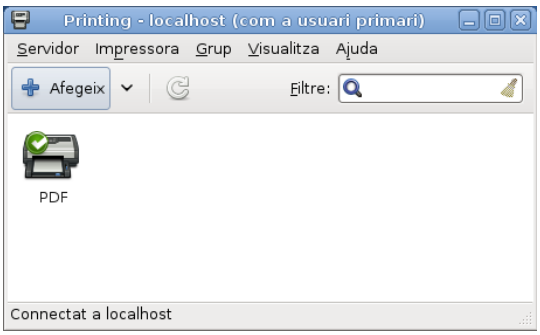


Per poder gestionar CUPS cal ser superusuari o bé usuari amb drets administració d'impressió formant part del grup *lpadmin*.

Administració gràfica del servidor

La majoria de distribucions disposen, a més de l'opció de configuració web, d'alguna eina alternativa per facilitar l'administració gràfica del sistema d'impressió. Una de les més esteses i que està configurada per defecte a la distribució Debian és system-config-printer (vegeu la figura **??**), que es pot iniciar des de consola o bé accedint al menú *Sistema* > *Administració* > *Impressió*.

FIGURA 3.10. Finestra principal del gestor d'impressió system-config-printer



L'eina administrativa system-config-printer ha estat desenvolupada en llenguatge Phyton i permet funcions semblants a l'administrador web, però amb l'avantatge de ser una aplicació nativa. D'altra banda, no està limitada a la configuració de la màquina local, ja que també pot fer servir els protocols IPP i HTTP per comunicar-se amb servidors CUPS remots.

3.2.3 Ordres de consola per a la gestió d'impressores i treballs

CUPS proporciona compatibilitat i emula les ordres de consola tradicionals de les principals plataformes Unix (Unix Berkeley BDS i Unix System V). A la taula **??** es presenten les principals ordres de consola, tot indicant la seva utilitat i de quin sistema provenen originalment.

Unix BDS i Unix System V

Berkeley Software Distribution és una de les plataformes històriques d'Unix, inicialment desenvolupada en aquesta universitat de Califòrnia. Trobem en l'actualitat evolucions i descendents seus en codi lliure com FreeBDS i OpenBDS, així com propietàries com Mac OS X.

Unix System V ésuna altra de les principals plataformes Unix que han evolucionat en implementacions propietàries com Solaris, HP/UX o SCO OpenServer.

TAULA 3.3. Principals ordres de consola reconegudes pel sistema CUPS

Ordre	Sistema	Descripció
lp	SysV	Imprimeix arxius.
lpr	BSD	L'ordre d'usuari per a tasques d'impressió.
lpc	BSD	Control d'impressores i cues (només lectura).
cupsd	CUPS	Dimoni de CUPS.
cupsaddsmb	CUPS	Exporta impressores a Samba per a clients Windows.
lpadmin	SysV	Configura les impressores i classes de CUPS.
lpinfo	CUPS	Mostra els dispositius disponibles.
lpmove	SysV	Mou treballs entre diferents cues d'impressió.
lpq	BSD	Mostra l'estat de la cua d'impressió.
lprm	BSD	Esborra un treball de la cua d'impressió.
cancel	SysV	Cancela treballs d'impressió.
disable	SysV	Atura impressores i classes d'impressora.
enable	SysV	Inicia impressores i classes d'impressora.
lpoptions	CUPS	Mostra o estableix les opcions de la impressora i les opcions per defecte.

TAULA 3.3 (continuació)

Ordre	Sistema	Descripció
lpstat	SysV	Mostra informació de la cua d'impressió.
lppasswd	CUPS	Afegeix, canvia o esborra contrasenyes.
cups-config	CUPS	Obté informació de l'API de CUPS, el compilador i els directoris.

Cal remarcar que CUPS només proporciona compatibilitat amb l'ordre *lpc* d'administració d'Unix BDS en mode de lectura. Així, doncs, per a l'administració per línia d'ordres, és més habitual fer servir l'ordre *lpadmin* de System V.

A continuació es comenten les principals tasques administratives de control d'impressió fent servir ordres de consola.

Generar treballs d'impressió

Ordres relacionades amb la generació de treballs d'impressió:

- Imprimir un arxiu en la impressora per defecte del sistema:

```
$ lp nomArxiu
$ lpr nomArxiu
```

- Imprimir un arxiu en una impressora específica:

```
$ lp -d nomImpressora nomArxiu
$ lpr -P nomImpressora NomArxiu
```

- Impressió de la sortida d'un programa: Les ordres *lp* i *lpr* accepten canonades per imprimir la sortida estàndard de qualsevol altra ordre o programa:

```
$ programa | lp
$ programa | lpr
```

A la taula 3.4 es detallen les principals opcions de les ordres *lp* i *lpr*.

TAULA 3.4. Resum d'opcions de les ordres lp i lpr

Ordre *lp* (System V)	Ordre *lpr* (BSD)	Descripció
-c		Fa una còpia de l'arxiu que està sent imprès.
-m	-m	Envia un correu electrònic en acabar la impressió.
-s		Deixa de mostrar els missatges informatius.

TAULA 3.4 (continuació)

Ordre *lp* (System V)	Ordre *lpr* (BSD)	Descripció
-w		Mostra un missatge a la pantalla en finalitzar la impressió.
-d nomImpres	-P nomImpres	Envia el treball a la impressora "nomImpres".
-n númCòpies	-# númCòpies	Nombre de còpies a imprimir.
-p númPàg		Els números de les pàgines que es volen imprimir en ordre ascendent. Es poden especificar pàgines separades, un rang de números o ambdues opcions.
-q númPrioritat		Assigna una prioritat al treball dins de la cua d'impressió mitjançant un nombre sencer entre 0 (prioritat màxima) i 39 (prioritat mínima).
-t "títol"	-T "títol"	Imprimeix el títol a la pàgina de portada del treball d'impressió.

Llistar impressores disponibles

Ordres relacionades amb el llistat d'impressores disponibles:

- Molts sistemes tenen més d'una impressora a disposició de l'usuari. Per obtenir un llistat de les impressores i classes disponibles:

```
$ lpstat -p
printer ClasseBN disabled since dt 31gen2012 18:57:57 CET
printer hp9040DN is idle. enabled since dt 31gen2012 18:23:36 CET
printer PDF is idle. enabled since dt 31gen2012 01:47:33 CE
```

- Alternativament es pot fer servir l'ordre *lpc*:

```
$ lpc status
```

Assignar impressora per defecte

Per definir quina és la impressora que rebrà els treballs per defecte:

```
$ lpoptions -d nomImpressora
```

Administració de treballs d'impressió

Ordres relacionades amb l'administració de treballs d'impressió:

- Per eliminar un treball d'impressió pendent en la cua.

```
$ cancel id_Treball
$ lprm id_Treball
```

- Cancel·lar tots els treballs d'un usuari determinat:

```
$ cancel -u NomUsuari
$ lprm NomUsuari
```

- Trasllat d'un treball d'impressió. Amb aquesta ordre es mou un treball d'impressió a una altra impressora o classe d'impressores.

```
$ lpmove id_Treball nomImpressora
```

- Podem obtenir l'identificador d'un treball (id_Treball) amb les ordres

```
$ lpq
$ lpstat
```

- Imprimir diverses còpies. Tant l'ordre *lp* com l'*lpr* tenen opcions per a la impressió de més d'una còpia d'un arxiu:

```
$ lp -n num_copies nomArxiu
$ lpr -# num_copies nomArxiu
```

Gestió de classes

Ordres relacionades amb la gestió de les classes d'impressora:

- Afegir una impressora a una classe. En aquest exemple afegim la impressora HP9040 a la classe d'impressores en blanc i negre:

```
$ lpadmin -p hp9040 -c ClasseBN
```

- Retirar una impressora d'una classe. En aquest cas eliminem la impressora virtual PDF de la classe d'impressores en color.

```
$ lpadmin -p PDF -r ClasseColor
```

Si la classe no existeix es crea automàticament en afegir la primera impressora. Si la classe queda buida s'esborra. Es poden comprovar les classes existents a l'arxiu `/etc/cups/classe.conf`.

Habilitar o denegar l'accés a usuaris

Ordres relacionades amb el control d'accés d'usuaris:

- Per determinar el control d'accés dels usuaris a una determinada impressora es fa servir l'ordre *lpadmin* amb l'opció *-u* i a continuació les llistes d'usuaris amb permís (*allow*) o sense permís (*deny*).

```
$ lpadmin -p hp9040DN -u allow:joan,laia
```

- A les llistes es por fer servir l'expressió ***all*** (tots) i ***none*** (ningú). En el següent exemple es permet l'accés a la impressora HP9040DN a tothom menys als usuaris Josep i Anna.

```
$ lpadmin -p hp9040DN -u allow:all deny:josep,anna
```

Imprimir canviant les opcions de la impressora

La configuració per defecte de la impressora pot ser suficient en la majoria d'ocasions. Si cal canviar alguna opció en imprimir un determinat arxiu podem fer servir l'opció *-o* de les ordres *lp* i *lpr*, tal com es mostra en aquest exemple on indiquem orientació de paper, escala i mida del paper:

```
lp -o landscape -o scaling=80 -o media=A4 nomArxiu.jpg
lpr -o landscape -o scaling=70 -o media=A3 nomArxiu.pdf
```

Les opcions d'impressió disponibles varien depenent de la impressora i es poden consultar a l'arxiu PPD corresponent. Les opcions d'impressió estàndard es descriuen breument a la taula 3.5.

TAULA 3.5. Resum de les opcions de configuració estàndard de la impressora

Opció	Descripció
landscape	Indica orientació apaïsada.
media=	Mida del paper. Alguns valors poden ser: *Letter, Legal, A3, A4, A5...* en funció de cada impressora.
sides=	Indica la impressió en les dues cares amb eix al costat curt (*two-sided-short-edge*) o al costat llarg (*two-sided-long-edge*)
page-ranges=	Especifica els números de pàgina (separats per coma) o rangs de pàgines (guionet) a imprimir. Ex. 1-4, 7, 9-12.
page-set=	Imprimeix només les pàgines parells o senars (*odd, even*).
outputorder=	Indica l'ordre d'impressió (*normal, reverse*).
cpi=	Caràcters per polsada (10, 12, 17...).
lpi=	Línies per polsada (6, 8...).
columns=	Formata el text en dues o més columnes.

TAULA 3.5 (continuació)

Opció	Descripció
page-left= **page-right=** **page-top=** **page-bottom=**	Especifica el marge esquerra (*left*), dreta (*right*), superior (*top*) i inferior (*bottom*) mesurat en punts (1 pt = 1/72 polsades).
scaling=	Percentatge entre 1 i 800 de mida en relació a la pàgina.
natural-scaling=	Percentatge entre 1 i 800 en relació a la mida original.
ppi=	Resolució de la imatge (1-1.200) en píxels per polsada.
job-sheets=	Impressió de pàgines de coberta (*none, standard, classified*).
brightness=	Corregeix els valors de brillantor (*brightness*). Per sota de 100 enfosqueix la imatge.
gamma=	Valors de correcció gamma. Per sota de 1.000 enfosqueix la imatge.

Canviar les opcions predeterminades

Totes les opcions anteriors s'apliquen a un treball d'impressió determinat. Si volem que una impressora quedi permanentment configurada amb una opció determinada hem de fer servir l'ordre *lpoptions* amb la sintaxi següent:

```
$ lpoptions –p NomImpressora –o NomOpcio=ValorOpcio
```

Així, en l'exemple següent es deixa com a predeterminda la mida de la pàgina per defecte de la impressora HP9040:

```
$ lpoptions –p hp9040 –o PageSize=Letter
```

Tanmateix, aquest canvi pot tenir efecte per a un conjunt d'usuaris diferent segons qui ha fet l'ordre:

- Si l'ordre ha estat feta per un usuari normal, aquesta configuració es guarda a l'arxiu personal **~/.lpoptions** i només afecta els treballs d'impressió d'aquest usuari.
- Si l'ordre la ha efectuat un usuari administrador, aquesta configuració es guarda a l'arxiu **/etc/cups/lpoptions** i afecta tots els usuaris de la màquina.
- Si la impressora és compartida i volem que aquesta configuració afecti els treballs que enviïn a la cua tots els clients de la xarxa, serà necessari canviar la configuració per defecte de l'arxiu PPD corresponent mitjançant l'ordre ***lpadmin*** i la mateixa sintaxi:

```
$ lpadmin –p hp9040 –o PageSize=Letter
```

Per esborrar un canvi d'opcions predeterminades fet amb *lpoptions* es fa servir la mateixa sintaxi però amb l'opció *-r* (*remove*).

```
$ lpoptions -p hp9040 -r PageSize=Letter
```

Finalment, si volem visualitzar la configuració completa d'una impressora determinada, per exemple:

El símbol * indica el valor actiu en aquest moment, en aquest exemple: mida A4 i resolució de 300 dpi.

```
$ lpoptions p hp9040DN -l

PageSize/Page Size: Custom.WIDTHxHEIGHT 11x14 11x17 13x19 16x20 16x24 2A 4A 8
    x10 8x12 A0 A1 A2 A3 *A4 A5 AnsiA AnsiB AnsiC AnsiD AnsiE ArchA ArchB
    ArchC ArchD ArchE C0 C1 C2 C3 C4 C5 Env10 EnvC5 EnvDL EnvMonarch Executive
     ISOB0 ISOB1 ISOB2 ISOB3 ISOB4 ISOB5 JISB0 JISB1 JISB2 JISB3 JISB4 JISB5
    Ledger Legal Letter RA0 RA1 RA2 RA3 RA4 SRA0 SRA1 SRA2 SRA3 SRA4 SuperA
    SuperB TabloidExtra Tabloid
Resolution/Output Resolution: 150dpi *300dpi 600dpi 1200dpi 2400dpi
```