Administració de sistemes operatius de xarxes

Víctor Carceler Hontoria

Índex

Introducció 5

Objectius 6

1. Servidor d'arxius 7
1.1. Límits als recursos d'emmagatzematge. Quotes de disc 7
1.1.1. Àmbit d'aplicació dels límits als recursos d'emmagatzematge 9
1.1.2. Implementació de les quotes de disc 10
1.1.3. Posada en marxa de les quotes de disc 10
1.1.4. Cas pràctic: quotes de disc amb la Linkat 2.0 11
1.2. Sistema de fitxers NFS 15
1.2.1. Configuració del servidor NFS 16
1.2.2. Exemples de configuració en el servidor NFS 18
1.2.3. Accés a sistemes de fitxers NFS des de l'estació client 19
1.3. Samba i el sistema de fitxers en xarxa SMB/CIFS 22
1.3.1. Conceptes relacionats amb l'SMB/CIFS 23
1.3.2. Descripció del Samba 24
1.3.3. El Samba funcionant com a client 24
1.3.4. El Samba funcionant com a servidor 25

2. Seguretat 28
2.1. Seguretat interna, controlant els processos i usuaris 29
2.1.1. Llistes de control d'accés (ACL) 29
2.1.2. Registre dels programes executats pels usuaris: *process accounting* 30
2.1.3. AppArmor 31
2.1.4. Security-Enhanced Linux, SELinux 34
2.1.5. Sistemes de detecció d'intrusions 34
2.2. Seguretat en la xarxa: tallafocs, DMZ i política per defecte 35
2.2.1. Tipus de tallafocs 36
2.2.2. Introducció a netfilter/iptables 37
2.2.3. Fer persistents les polítiques de filtratge de paquets 40
2.2.4. Treball amb taules, cadenes i regles 40
2.2.5. Detalls de filtratge en les regles 41

3. Administració remota 44
3.1. Administració remota basada en la línia d'ordres 45
3.1.1. Introducció a SSH 47

3.1.2. Alguns conceptes criptogràfics 47
3.1.3. Funcions bàsiques de l'SSH 49
3.1.4. SSH: mecanismes d'autenticació 52
3.1.5. Configuració pel client SSH 54
3.1.6. Configuració del servidor SSH 55
3.2. Administració remota amb interfície gràfica 57
3.2.1. La transparència de xarxa del sistema X Window 57
3.2.2. *Virtual network computing* (VNC) 61
3.2.3. *Remote desktop protocol* (RDP) 65
3.3. Gestió remota mitjançant una aplicació local 65
3.3.1. Introducció a la Webmin..... 66

Introducció

Al llarg del temps, l'arquitectura client-servidor ha provat les seves virtuts en diferents àmbits d'aplicació. A tot arreu on hi ha un grup d'usuaris treballant es necessiten serveis centralitzats d'emmagatzematge, impressió, execució i autenticació entre d'altres. Aquests serveis s'executen en servidors que funcionen amb un sistema operatiu de xarxes.

L'administració dels servidors de xarxa constitueix el centre d'atenció del personal d'administració i és la base essencial per desenvolupar qualsevol activitat productiva a la xarxa.

Al llarg d'aquest crèdit es descriuran les tasques principals a realitzar en l'administració de xarxes d'àrea local, identificant els serveis de xarxa més comuns i les pràctiques més adequades per al manteniment dels servidors.

En el nucli d'activitat "Servidor d'arxius" es desenvoluparan les funcions d'aquest servei explorant diferents eines.

En el nucli d'activitat "Seguretat" s'estudiaran els riscos dels servidors de xarxa i les pràctiques adequades per reduir-los. Tot i que la seguretat és un procés i no un producte, l'eina central d'estudi serà el tallafoc.

En el nucli d'activitat "Administració remota" s'explicaran diferents solucions per administrar màquines per mitjà d'un terminal de treball; això és necessari perquè els servidors solen estar ubicats en armaris o sales amb accés restringit, quan no estan instal·lats a diferents ciutats o països.

Per assimilar els continguts d'aquesta unitat didàctica, és convenient anar fent les activitats i els exercicis d'autoavaluació.

Objectius

En acabar la unitat didàctica, heu de ser capaços del següent:

1. Determinar les accions i els paràmetres requerits per a la instal·lació i l'operativa del maquinari i programari en entorns monousuaris i multiusuaris, segons les instruccions dels manuals.

2. Especificar els elements i els components del sistema que s'han d'afegir o modificar per adaptar la configuració del sistema als requeriments establerts, segons la documentació tècnica disponible.

3. Establir els procediments i els mitjans que garanteixin la seguretat del sistema i de la informació, en la configuració del maquinari i instal·lació del programari base.

4. Establir el procés d'instal·lació i generació del programari base, segons els requeriments proposats i la composició actual del sistema.

5. Realitzar, sobre un sistema operatiu monousuari o multiusuari, les funcions d'usuari referents a connexió i desconnexió, gestió de l'espai d'emmagatzematge, gestió de processos, i utilització de perifèrics i comunicació amb altres usuaris, amb precisió i destresa.

6. Definir i assignar valors a variables d'usuari i del sistema, d'acord amb l'entorn de treball i amb els requeriments de l'explotació del sistema.

7. Definir l'espai d'emmagatzematge, i l'esquema de seguretat i confidencialitat de la informació, adequats als usuaris i a la informació que cal guardar.

8. Instal·lar el programari per a la prevenció d'errades que afectin la integritat de les dades i la lògica dels processos, segons les instruccions del manual.

1. Servidor d'arxius

Els servidors d'arxius proporcionen espai d'emmagatzematge que els ordinadors client poden tractar com si fos espai local seu. D'aquesta manera es pot proveir d'emmagatzematge centralitzat un grup d'ordinadors amb els avantatges que això implica: estalvi de maquinari, gran flexibilitat, facilitat en l'administració (còpies de seguretat, compartició d'informació, etc.).

Atès que un servidor d'arxius presta serveis a diferents clients, s'imposa la necessitat d'arbitrar l'assignació dels recursos d'emmagatzematge entre els usuaris, per tal d'evitar abusos en què un usuari acapara tots els recursos. Les quotes de disc imposen límits als recursos assignats a un client i garanteixen que el servidor es podrà compartir d'una manera adequada.

La comunicació entre el servidor i els clients es realitzarà mitjançant la xarxa amb independència del sistema operatiu de les màquines i, fins i tot, del sistema de fitxers utilitzat. Per aquesta raó és necessari un protocol de comunicació que sigui independent dels detalls propis del sistema operatiu dels equips i del sistema de fitxers físic utilitzat a cadascuna de les màquines. Aquesta és la tasca del sistema de fitxers de xarxa. Com sempre trobarem diferents opcions per escollir, cadascuna amb les seves particularitats, avantatges i desavantatges.

1.1. Límits als recursos d'emmagatzematge. Quotes de disc

Quan diferents usuaris es disputen uns recursos d'emmagatzematge comuns cal establir límits per tal d'evitar que un usuari s'apropiï de tots els recursos. En els sistemes de fitxers, aquests límits s'imposen mitjançant les quotes de disc.

Cal establir límits a qualsevol recurs finit. En el cas dels sistemes de fitxers cal establir límits per al nombre de fitxers i per a la capacitat.

Els dispositius de l'Unix

En l'Unix hi ha dispositius de blocs, dispositius de caràcters i dispositius orientats a sòcols o *sockets*.

Els dispositius de blocs realitzen les operacions d'entrada/sortida amb blocs d'informació d'una mida fixa. Poden permetre un accés aleatori a les dades i un *buffer* de memòria intermèdia per gestionar aquestes operacions.

Els sistemes de fitxers permeten utilitzar un dispositiu de blocs, com un disc dur o un *pen drive* USB, com un mitjà d'emmagatzematge on l'usuari pot desar els fitxers ordenats en diferents directoris. En el moment de crear el sistema de fitxers es defineix l'estructura que permetrà desar les dades de l'usuari. Es defineixen alguns blocs amb funció especial (superbloc, node d'identificació o *inode*) i d'altres que guardaran dades per a

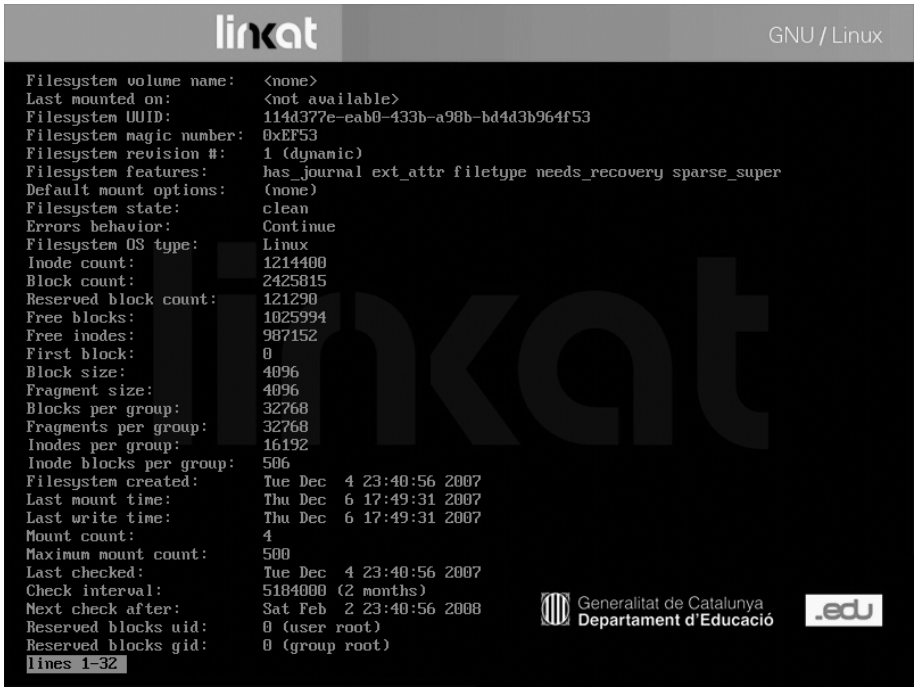
l'usuari. És per aquesta raó que la capacitat total del dispositiu no estarà disponible per desar dades de l'usuari.

En els sistemes de fitxers de l'Unix, l'estructura on es guarden l'usuari, el grup, els permisos i les diferents dades d'un fitxer (no el seu nom) s'anomena *node d'identificació* (*inode*). Per a cada fitxer cal, com a mínim, un node d'identificació per referenciar-lo. Si el fitxer és molt gran poden fer falta més nodes d'identificació. Per aquesta raó, si en un sistema de fitxers no queden nodes d'identificació disponibles no es podrà crear cap fitxer nou, malgrat que hi hagi un munt de blocs de dades lliures.

Si emprem els sistemes de fitxers EXT2/EXT3, la utilitat **`dumpe2fs`** ens permet obtenir informació sobre el sistema de fitxers, i ens indica, entre altres dades, el nombre de nodes d'identificació total i els que manquen lliures.

En la figura 1 es mostren els detalls proporcionats per `dumpe2fs` sobre un sistema de fitxers EXT3. En l'exemple es pot comprovar que el sistema de fitxers té 1.214.400 nodes d'identificació (*inode count*) i 2.425.815 blocs per dades (*block count*), dels quals resten lliures 987.152 nodes d'identificació (*free inodes*) i 1.025.994 blocs (*free blocks*). Els nodes d'identificació lliures ens determinen el nombre de fitxers que podrem crear, i els blocs de dades, la màxima capacitat d'informació que podrem emmagatzemar. El sistema de fitxers d'exemple té una capacitat de 9,2 GB.

Figura 1. Exemple de sortida de l'ordre `dumpe2fs`



Instal·lació de la Linkat 2

La instal·lació de la Linkat 2 escull per defecte el sistema d'arxius EXT3. Durant la instal·lació, si es decideix personalitzar les particions del disc es pot utilitzar l'XFS. Una vegada instal·lat el sistema operatiu, és possible instal·lar el paquet **`jfsutils`** per tal d'aconseguir suport per al JFS.

Sistemes d'arxius en el Linux

L'EXT3 és probablement el sistema d'arxius emprat per defecte en més distribucions del GNU/Linux. Malgrat que és un sistema d'arxius molt estable, no és el més ràpid ni el més eficient dels dis-

ponibles per al GNU/Linux. El JFS i XFS són dos sistemes de fitxers que proporcionen un rendiment millor. Mentre el nombre de nodes d'identificació en l'EXT3 es decideix en el moment de la seva creació, el JFS i XFS són capaços de crear nodes d'identificació de manera dinàmica durant el seu funcionament sempre que disposem de blocs lliures.

L'altre recurs obvi dels sistemes de fitxers és la capacitat d'emmagatzematge per a les dades de l'usuari. La utilitat **`df`** ens informarà de la mida i ús de cada sistema de fitxers muntat. Cal recordar que la informació dels fitxers es desa en blocs de dades d'una mida fixa. El bloc és la unitat d'assignació i la seva mida es defineix durant la creació del propi sistema de fitxers. Per això, malgrat que un fitxer només contingui 1 byte ocuparà un bloc sencer. Podem utilitzar l'ordre **`du`**, que ens permetrà esbrinar l'espai que ocupen els fitxers en el disc.

Els mateixos fitxers poden ocupar una quantitat d'espai diferent en el disc si els dispositius utilitzen una mida de bloc diferent.

Agrupació de cues

Alguns sistemes de fitxers, com el ReiserFS, implementen un sistema, anomenat *tail packing*, que permet agrupar les cues de diferents fitxers en un únic bloc.

Aquesta funció és especialment útil en treballar amb un nombre elevat de fitxers petits. En aquest escenari és molt evident l'aprofitament millor de l'espai i en alguns casos un increment de velocitat.

1.1.1. Àmbit d'aplicació dels límits als recursos d'emmagatzematge

Les quotes de disc es poden activar/desactivar de manera independent per a cada sistema d'arxius. És a dir, si l'arrel del sistema (/) és en /**dev**/**sda1**, i els directoris personals (/**home**) són a /**dev**/**sdb1**, es podran posar en marxa les quotes de disc per a cada dispositiu per separat. I en cada dispositiu es podran definir uns límits diferents per al mateix usuari/grup.

Dins d'un sistema d'arxius es poden fixar límits per a un usuari o per a un grup d'usuaris, i aquests límits poden ser tous o durs. Els límits tous es poden depassar, pràcticament només serveixen com a nivell d'advertència que indica que ens apropem al límit dur. Els límits durs no es poden excedir mai.

Dispositius

/dev/sda1 i /dev/sdb1 són dues particions de disc. De fet, /dev/sda és el primer disc SCSI, i /dev/sda1 la primera partició d'aquest disc. /dev/sdb serà el segon disc SCSI. Normalment, el GNU/Linux tracta els discos SATA com si fossin SCSI.

Cada cop que es depassa un límit tou, comença el compte enrere del període de gràcia. Si el període de gràcia esmentat transcorre sense que s'hagin alliberat recursos (i, per tant, ja no es depassi el límit tou), el sistema impedirà reservar nous recursos encara que no hàgim arribat al límit dur. La idea és que l'usuari, o el grup d'usuaris, normalment només necessita certs recursos d'emmagatzematge que estan per sota del límit tou. Si de manera excepcional necessités més recursos, que els definits pel límit tou, no hi hauria cap problema. Es poden consumir sempre que no arribem al límit dur. Però aquesta situació no es pot perllongar de manera indefinida. Una vegada resolta la situació excepcional, s'han d'alliberar recursos per tal de tornar a la situació de normalitat.

Els límits tous es poden excedir durant el període de gràcia. Els límits durs no es poden excedir mai.

1.1.2. Implementació de les quotes de disc

Un sistema que treballa amb quotes de disc necessita guardar informació sobre els recursos consumits per cada usuari/grup. Aquesta informació normalment s'enregistra en uns fitxers de control que són en l'arrel del sistema d'arxius.

Però hi ha altres sistemes de fitxers que integren la informació de control de quotes en les metadades dels fitxers, per exemple, XFS. En aquest cas no existeixen els fitxers aquota.user i aquota.group. Tota la informació de control està integrada en diferents atributs dels fitxers. La implementació de les quotes de disc en l'XFS està integrada dins del sistema de fitxers, d'aquesta manera la informació de control de quotes sempre està actualitzada.

Els fitxers de control de quotes són **aquota.user** i **aquota.group**. Si s'utilitza XFS no existeixen, perquè s'utilitzen els atributs dels fitxers per emmagatzemar la informació de control de les quotes.

En altres sistemes de fitxers, si es desactiven les quotes i es fa servir el sistema de fitxers, la informació de control enregistrada en els fitxers aquota.user i aquota.group es desactualitzarà i caldrà fer servir l'eina **`quotacheck`** per tornar-la a actualitzar.

A més a més, per tal que es puguin activar les quotes serà necessari muntar el sistema d'arxiu amb les opcions de suport per a les quotes d'usuari i/o de grup. Aquestes opcions són **`usrquota`** i **`grpquota`**. L'ordre **`mount`** ens pot informar en tot moment de les opcions de cada sistema d'arxiu muntat.

1.1.3. Posada en marxa de les quotes de disc

Per tal d'activar les quotes de disc en un sistema que ja funciona, serà necessari el següent:

1) Instal·lar les eines de control de quotes. En la Linkat 2 és el paquet **quota**.
2) Muntar el sistema d'arxius amb les opcions adequades **`usrquota`** i/o **`grpquota`**.
3) Utilitzar l'eina **`quotacheck`** per tal d'examinar el sistema d'arxius i recopilar la informació de control de quotes, és a dir, per crear els fitxers aquota.user i aquota.group si no es tracta d'un sistema de fitxers XFS. En un sistema de fitxers XFS no utilitzarem **`quotacheck`**.
4) Es poden definir límits per als usuaris i/o grups mitjançant **`edquota`**.
5) Es poden activar les quotes mitjançant l'ordre **`quotaon`**.

Una vegada s'han imposat límits als recursos d'emmagatzematge, cada usuari pot consultar el seu consum de recursos i el seu límit en executar l'ordre **quota**. L'administrador podrà fer servir les ordres **repquota**, **quotastats**, **quotaon**, **quotaoff**, **quotacheck** i **edquota**.

1.1.4. Cas pràctic: quotes de disc amb la Linkat 2.0

Per tal de fer més entenedor el procés d'administració de quotes, en veurem un exemple amb la Linkat 2. Per no treballar directament amb els sistemes de fitxers en ús, i per tal de no haver de reparticionar el nostre disc dur, utilitzarem els dispositius de bucle com si fossin discos durs reals.

Dispositius de bucle

Els dispositius de bucle, /dev/loop[X], ens permetran tractar un fitxer com si fos un dispositiu de blocs, és a dir, com un disc dur o partició. Per exemple, ens permeten muntar una imatge .iso com si tinguéssim el CD o DVD real.

1) Preparem els nostres dispositius d'emmagatzematge virtuals

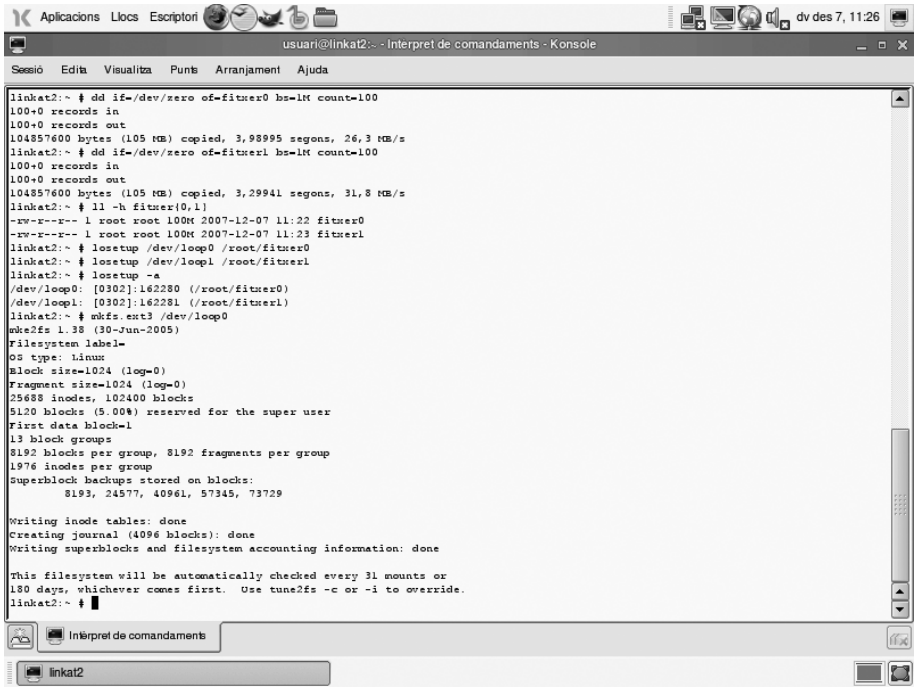
Com a usuari administrador crearem dos fitxers de 100 MiB cadascun. Associarem un dispositiu de bucle a cada fitxer per obtenir un disc dur virtual i crearem un sistema de fitxers en el seu interior, EXT3 per a **/dev/loop0** i XFS per a **/dev/loop1**.

En la figura 2 es poden seguir els passos necessaris per crear els fitxers de 100 MiB (omplerts de bytes amb valor 0), connectar un dispositiu de bucle amb els fitxers i crear un sistema d'arxius EXT3 a **/dev/loop0**. Restaria crear un sistema d'arxius XFS a **/dev/loop1** amb l'ordre **mkfs.xfs /dev/loop1**. Hem executat totes les instruccions com a administrador, i el directori de treball és /root.

Discos virtuals

Per construir un sistema d'arxius hem fet servir l'ordre **dd** per tal de generar un fitxer amb 100 MiB. Després, mitjançant l'ordre **losetup**, hem associat un dispositiu de bucle amb el fitxer per poder-lo fer servir com un disc dur. Finalment, amb l'eina **mkfs.ext3** hem escrit un sistema d'arxius en el nostre disc dur virtual.

Figura 2. Preparació dels dispositius virtuals



Tot seguit crearem dos directoris que serviran de punt de muntatge, /mnt/disc0 i /mnt/disc1.

```
reached:~ # cd /mnt
reached:/mnt # ls
reached:/mnt # mkdir disc0 disc1
```

I muntarem els nostres sistemes de fitxers amb les opcions de suport de quotes:

```
reached:/mnt # mount -o usrquota,grpquota /dev/loop0 /mnt/disc0
reached:/mnt # mount -o usrquota,grpquota /dev/loop1 /mnt/disc1
```

Podem fer servir mount per comprovar que els sistemes de fitxers estan muntats amb les opcions adequades, i df per comprovar-ne la capacitat.

```
reached:/mnt # mount
/dev/hda2 on / type ext3 (rw,acl,user_xattr)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
debugfs on /sys/kernel/debug type debugfs (rw)
udev on /dev type tmpfs (rw)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
securityfs on /sys/kernel/security type securityfs (rw)
/dev/loop0 on /mnt/disc0 type ext3 (rw,usrquota,grpquota)
/dev/loop1 on /mnt/disc1 type xfs (rw,usrquota,grpquota)
reached:/mnt # df -h
S. fitxers                          Mida      En ús
                                  Lliure      %Ús  Muntat en
/dev/hda2                           9,2G      5,4G  3,3G 62%
                                       /
udev                                253M      64K   253M 1%
                                    /dev
/dev/loop0                           97M      4,1M  88M  5%
                                /mnt/disc0
/dev/loop1                           92M      232K  92M  1%
                                /mnt/disc1
reached:/mnt #
```

2) Examinem els sistemes d'arxius per obtenir la informació de control

L'eina **quotacheck** ens permetrà examinar un sistema de fitxers per tal de generar la informació de control de quotes. L'ordre permet actualitzar els fitxers aquota.user i aquota.group o crear-los si no existeixen.

És molt important executar **quotacheck** només en sistemes de fitxers que no tenen les quotes activades i en què no s'estan fent canvis.

Utilitzarem l'ordre **quotacheck -ugc <punt_de_muntatge>** per generar els fitxers de control de quota per a usuaris i grups de **/mnt/disc0**.

Objectes perduts

El directori **lost+found** és un directori propi del sistema de fitxers EXT3. La seva funció és guardar les dades recuperades en netejar el sistema d'arxius, per exemple, després d'una aturada sobtada.

El sistema de fitxers XFS registra la informació de control de quotes com a atributs dels propis fitxers, per això no necessita l'execució de quotacheck.

Objectes perduts

El directori **lost+found** és un directori propi del sistema de fitxers EXT3. La seva funció és guardar les dades recuperades en netejar el sistema d'arxius, per exemple, després d'una aturada sobtada.

```
reached:/mnt # quotacheck -ugc /mnt/disc0
reached:/mnt # ll /mnt/disc0
total 24
-rw------- 1 root root   6144   2007-12-07 19:01 aquota.group
-rw------- 1 root root   6144   2007-12-07 19:01 aquota.user
drwx------ 2 root root   12288  2007-12-07 11:26 lost+found
reached:/mnt # ll /mnt/disc1
total 0
reached:/mnt #
```

3) Establim límits a cada sistema de fitxers

L'ordre **edquota** ens permetrà fixar els límits per a un usuari o grup. Quan s'executa se'ns obre l'editor seleccionat per la variable d'entorn EDITOR, per defecte, **vi**. En l'editor veurem el consum de recursos i els límits actuals, i si escau podrem canviar els límits. En desar les dades s'imposaran els nous límits.

Editors amb GUI

Si us agraden els editors amb entorn gràfic podeu provar a fer **export EDITOR=gedit** o **export EDITOR=kwrite** abans d'executar edquota.

Podem imposar límits per a l'usuari *usuari* al sistema de fitxers /mnt/disc0 amb l'ordre **edquota -u -f /mnt/disc0 usuari**.

```
Disk quotas for user usuari (uid 1000):
     Filesystem   blocks   soft   hard   inodes   soft   hard
     /dev/loop0        0      0      0        0      0      0
```

En l'editor es mostra el consum de recursos i els límits establerts per a l'usuari en cada sistema de fitxers. En aquest cas, solament és per a /dev/loop0.

La descripció de les columnes és la següent:

Mode d'inserció a vi

L'editor **vi** té dos modes de funcionament: ordres i inserció. En carregar l'editor caldrà prémer *i* per passar al mode inserció i començar a escriure. Per desar i sortir hem de prémer seqüencialment ESC, :, w, q, INTRO.

- *Filesystem*: sistema de fitxers
- *Blocks*: blocs de dades consumits
  - *Soft*: límit tou per als blocs
  - *Hard*: límit dur per als blocs
- *Inodes*: nombre de nodes d'identificació consumits
  - *Soft*: límit tou per als nodes d'identificació
  - *Hard*: límit dur per als nodes d'identificació

Fixem-nos en el següent:

- Només té sentit fer canvis en els límits. La resta de dades són informatives.
- La capacitat d'emmagatzematge es mesura en blocs, per tant, cal saber la mida del bloc del sistema de fitxers.

- Podem interpretar les dades referides a nodes d'identificació com a nombre de fitxers.

- Els límits que tenen valor 0 no són actius.

4) Activar i desactivar les quotes de disc

En cada sistema de fitxers, amb **quotaon** i **quotaoff**, es poden activar i desactivar les quotes de manera independent. Si s'utilitza XFS no cal activar de manera explícita les quotes. Pel sol fet de muntar una partició amb les opcions de suport de quotes se n'activa l'ús. Però és possible desactivar l'aplicació dels límits i utilitzar les quotes de disc únicament com a sistema comptable dels recursos.

```
reached:/mnt # quotaon -ugv /mnt/disc0
/dev/loop0 [/mnt/disc0]: group quotas turned on
/dev/loop0 [/mnt/disc0]: user quotas turned on
reached:/mnt #
```

Amb les quotes actives, l'usuari rebrà un missatge d'error en intentar depassar el límit dur. En l'exemple s'ha imposat un límit dur de cinc nodes d'identificació per a l'usuari.

```
usuari@reached:/mnt/disc0> touch a b c d e f
loop0: warning, user file quota exceeded.
loop0: write failed, user file limit reached.
touch: no se puede efectuar `touch' sobre «f»: Se ha excedido la cuota de disco
usuari@reached:/mnt/disc0>
```

Els fitxers a, b, c, d i e es creen sense problemes, però crear f suposaria excedir els límits.

5) Consulta dels límits

Un usuari pot consultar en tot moment el consum de recursos i els límits que té fixats mitjançant l'ordre **quota**.

```
usuari@reached:/mnt/disc1> quota
Disk quotas for user usuari (uid 1000):
   Filesystem     blocks   quota   limit   grace      files   quota      limit     grace
   /dev/loop0          0       0       0                 5*       3          5
   /dev/loop1          0       0       0                  2       3          5
usuari@reached:/mnt/disc1>
```

En l'exemple, l'usuari té limitats els nodes d'identificació a tres i cinc (límit tou i dur) en ambdós sistemes de fitxers. En el dispositiu /dev/loop0 ha consumit el màxim de nodes d'identificació. Per això apareix un * en la llista. En el dispositiu /dev/loop1 no ha depassat el límit tou.

L'usuari administrador pot obtenir informes sobre el consum de recursos de tots els usuaris amb l'ordre **repquota**.

```
reached:~ # repquota -ua
*** Report for user quotas on device /dev/loop0
Block grace time: 7days; Inode grace time: 7days
               Block limits                File limits
User        used   soft   hard  grace    used   soft   hard  grace
----------------------------------------------------------------------
root--      4127      0      0              3      0      0
usuari-+       0      0      0              5      3      5  6days


*** Report for user quotas on device /dev/loop1
Block grace time: 7days; Inode grace time: 7days
               Block limits                File limits
User        used   soft   hard  grace    used   soft   hard  grace
----------------------------------------------------------------------
root--         0      0      0              3      0      0
usuari-+       0      0      0              2      3      5


reached:~ #
```

6) Límits per a usuaris i grups

És possible combinar els límits d'usuaris i grups. Per exemple, una organització educativa pot utilitzar un disc dur de 250 GB per als directoris personals dels usuaris. Si tingués deu usuaris i fixés 20 GB de límit tou per usuari i 30 GB de límit dur, ens trobaríem, si tots els usuaris decideixen consumir al màxim la seva capacitat d'emmagatzematge, amb un sistema de fitxers ple. De fet, s'ompliria abans de satisfer la demanda de tots els usuaris (30 GB × 10 usuaris > 250 GB).

L'administrador pot evitar aquesta situació fixant un límit per al grup 'usuaris', suposant que tots deu usuaris tenen com a grup inicial 'usuaris', de diguem 240 GB. D'aquesta manera s'imposa un límit global per al conjunt d'usuaris, i es garanteix que mai s'omplirà al màxim el sistema de fitxers, però no es garanteix un repartiment equitatiu dins del grup. Si els usuaris consumeixen l'espai en disc de manera seqüencial, els vuit primers obtindran els seus 30 GB (un total de 8 × 30 GB = 240 GB). La resta no podrà fer ús de l'espai d'emmagatzematge.

El sistema de quotes de disc convencional no permet fer reserves d'espai per a usuaris o grups.

1.2. Sistema de fitxers NFS

NFS és l'acrònim de *network file system*, en català, sistema de fitxers de xarxa.

El sistema de fitxers NFS va ser desenvolupat per Sun Microsystems l'any 1984. Permet accedir a dispositius d'emmagatzematge remot, mitjançant

una xarxa, com si es tractés de dispositius locals. Els diferents Unix, i totes les distribucions del GNU/Linux, el suporten tot i que no és exclusiu de l'Unix. L'NFS està pensat per funcionar en una arquitectura client-servidor amb independència dels sistemes operatius i, fins i tot, del protocol de transport.

Hi ha diferents versions del protocol NFS:

1) NFS v1. Només es va utilitzar com a versió experimental en la mateixa Sun, i mai no es va publicar.

2) NFS v2. És la primera versió que es va fer pública. Està suportada per la majoria dels equips. No pot treballar amb arxius de més de 4 GB. El protocol de transport tradicional és l'UDP. Les escriptures són síncrones.

3) NFS v3. Inclou suport per a sistemes d'arxiu de 64 bits, la qual cosa permet treballar amb fitxers de més de 4 GB (permet arribar fins els 16 EiB). Pot fer servir com a protocol de transport UDP o TCP. Suporta escriptures asíncrones per millorar el rendiment.

4) NFS v4. És la primera versió publicada per l'IETF. Està influenciat per altres sistemes d'arxius en xarxa com CIFS i AFS, incorporant millores de seguretat i rendiment.

IETF

L'**IETF**, Internet Engineering Task Force, és l'organització mundial que estandarditza els protocols i serveis que s'utilitzen a Internet.

1.2.1. Configuració del servidor NFS

L'NFS està basat en una arquitectura client-servidor en què els servidors proporcionen sistemes de fitxers als clients. En configurar el servidor es decideix què es comparteix i amb quines màquines. Els ordinadors client podran muntar el sistema d'arxius remot i accedir-hi com si fos local.

És important adonar-se que en l'NFS es comparteixen sistemes de fitxers entre màquines, no entre usuaris. Quan un servidor exporta un directori ho fa per a la màquina client, no per a un determinat usuari de la màquina client. Dins de la màquina client es pot controlar l'accés al directori mitjançant l'usuari, grup i permisos tradicionals del sistema operatiu.

> La instal·lació per defecte de la Linkat 2.0 ja inclou el servei NFS. En altres distribucions caldrà buscar el paquet corresponent i instal·lar-lo.

El fitxer de configuració del servidor NFS és **/etc/exports**, i la seva sintaxi és prou senzilla. En cada línia es configura un directori diferent per exportar i la llista de màquines client que hi podran accedir amb les seves opcions

Comentaris amb #

En el fitxer **/etc/exports**, els comentaris comencen amb el caràcter **#**.

Els fitxers de configuració també tenen pàgina de manual. Feu '**man exports**' per consultar totes les opcions de sintaxi.

entre parèntesis. És molt important que les opcions de cada client vagin enganxades, sense espai, amb la definició de la màquina.

Les màquines client es poden definir amb flexibilitat. Entre les opcions acceptades trobem les següents:

1) Una màquina: es pot especificar mitjançant la seva IP o bé amb qualsevol nom DNS que el *resolver* sigui capaç de resoldre.

2) Xarxes IP: es poden especificar xarxes amb la màscara separada per punts (192.168.0.0/255.255.255.0) o bé de manera compacta (192.168.0.0/24).

3) Comodins en el nom DNS: es poden fer servir els caràcters * o ? com a comodins. Per exemple, *.dominiexemple.net es refereix a tots els *hosts* de dominiexemple.net i de qualsevol dels seus subdominis.

Entre les opcions bàsiques per exportar un directori trobem les següents:

- **`secure`**. El servidor només acceptarà peticions de clients que s'originin en un port per sota del 1024. Està activa per defecte. Si es vol desactivar cal especificar **`insecure`**.

- **`rw`**. Exporta en mode de lectura/escriptura. Per defecte s'exporta en mode de només de lectura. El mode de només de lectura es pot especificar amb **`ro`**.

- **`async`**. Fa treballar el servidor de manera asíncrona. És a dir, indicarà al client que l'escriptura s'ha completat malgrat que el suport físic encara estigui treballant. Aquesta opció accelera l'operació però augmenta el risc d'acabar amb dades malmeses en cas que s'interrompi la connexió de xarxa, o s'aturi el servidor, de manera sobtada. Per prevenir aquest problema, per defecte es fan escriptures síncrones, que es poden indicar amb l'opció **`sync`**.

Però algunes de les opcions més importants tenen a veure amb el fet que la màquina client no ha de tenir els mateixos usuaris que la màquina servidora, o potser totes dues tenen alguns usuaris en comú però aquests usuaris són persones diferents.

L'Unix distingeix els usuaris mitjançant l'identificador d'usuari UID. L'usuari administrador és el que té un 0 per UID, independentment de l'inici de sessió o *login* que tingui, malgrat que per conveni l'usuari administrador de l'Unix se sol anomenar *root*. Però una qüestió clau és que, si tenim un servidor i un client, l'usuari *root* del servidor probablement no sigui la mateixa persona que l'usuari *root* del client.

Tant el servidor com el client utilitzen l'UID i el GID per saber a quin usuari i grup pertanyen els fitxers. En fer una llista llarga, **`ls -l`**, el sistema consulta els fitxers **/etc/passwd** i **/etc/group** per mostrar el *login* i el nom del grup en lloc de l'UID i el GID anotats en el sistema d'arxius. Per això, si al servidor l'usuari amb UID=1000 té per *login* 'usuariA', i al client l'usuari amb UID=1000 té per *login* 'usuariB'. En el servidor, els fitxers creats des del client es veuran com a fitxers d'"usuariA' mentre que, al client, els mateixos fitxers es veuran com a fitxers d'"usuariB'.

GID

En l'Unix, s'utilitza l'identificador d'usuari (UID) per identificar els usuaris, i l'identificador de grup (GID) per identificar els grups.

Un grup és una col·lecció d'usuaris.

Atès que l'UID i el GID dels fitxers condicionen amb els permisos qui pot fer què, és una qüestió força important decidir si l'usuari primari o usuari *root* del client tractarà el servidor com l'usuari primari o bé com un altre usuari. Es pot fer el mateix per als altres usuaris, encara que per al cas de l'administrador sigui gairebé una necessitat.

Per defecte, el servidor tractarà tots els accessos de l'usuari amb UID 0 de la màquina client com si els fes l'usuari amb UID 65534 (*nobody*). Aquesta funció rep el nom de *root squash* (aixafar l'usuari primari). És possible mapar l'usuari primari del client a un usuari diferent al servidor, també és possible indicar que qualsevol usuari del client ha de ser tractat en el servidor com l'usuari *nobody*.

Les opcions per mapar usuaris són les següents:

- **`root_squash`**. Mapa les peticions amb UID/GID 0 a l'usuari i grup anònim, per defecte *nobody*/*nogroup*.

- **`no_root_squash`**. Desactiva el *root squash*.

- **`all_squash`**. Mapa les peticions de qualsevol UID/GID a l'usuari i grup anònims. L'opció contrària és **`no_all_squash`** que es troba activa per defecte.

- **`anonuid`** i **`anongid`**. Permeten indicar l'UID i el GID de l'usuari sense privilegis.

1.2.2. Exemples de configuració en el servidor NFS

Suposem que en el punt de muntatge **/mnt/disc0** hem muntat un sistema d'arxius on tindrem diferents directoris exportats mitjançant NFS. Hi haurà un directori **Dades** que exportarem en mode de només de lectura i un directori **Temporal** que serà de lectura/escriptura.

Per reduir la complexitat de l'exemple suposarem que el client NFS s'executa en el mateix ordinador que el servidor NFS. Per això, al fitxer

de configuració **/etc/exports** donarem permís per accedir al servidor en la màquina amb adreça IP 127.0.0.1. Aquesta configuració pot ser entenedora i ajuda a provar el cas a la pràctica, però no és gaire real. En una situació real, el client o els clients seran màquines diferents del servidor, i en el fitxer de configuració s'hauran d'especificar les adreces, o la xarxa, dels clients.

Després de canviar el fitxer de configuració /etc/exports, caldrà reiniciar el servidor NFS o indicar-li que torni a llegir el fitxer de configuració. Amb l'ordre **`/etc/init.d/nfsserver restart`** o **`/etc/init.d/nfsserver reload`**.

Vegem els exemples següents:

1) Directori Dades exportat com de només de lectura i directori Temporal com de lectura/escriptura

```
/mnt/disc0/dades      127.0.0.1(ro,sync)
/mnt/disc0/temporal   127.0.0.1(rw,sync)
```

En aquest exemple, tots dos directoris s'exporten per la màquina 127.0.0.1, Dades en mode de lectura i Temporal com de lectura/escriptura. Atès que l'opció **ro** està activa per defecte no caldria escriure-la. L'opció **`sync`** també està activa per defecte, però si no s'especifica **`sync`** o **`async`**, el servidor NFS escriu una advertència cada vegada que s'inicia per notificar el comportament per defecte a l'administrador.

El servidor emprarà l'UID i el GID indicats pel client, amb l'excepció de l'UID/GID 0 que en el servidor es maparan a l'usuari i grup *nobody*/*nogroup*.

2) Directori Dades exportat com de lectura/escriptura per una màquina de confiança i mapatge d'usuaris per a directori Temporal

```
/mnt/disc0/dades      127.0.0.1(ro,sync) 192.168.1.1(rw,sync)
/mnt/disc0/temporal   127.0.0.1(rw,sync,all_squash,anonuid=90)
```

El directori Dades ara s'exporta com de lectura/escriptura per la màquina amb adreça IP 192.168.1.1. A més a més, en els accessos a Temporal es maparà l'UID de la màquina client al UID 90 del servidor.

1.2.3. Accés a sistemes de fitxers NFS des de l'estació client

L'administrador de l'estació client pot muntar els sistemes de fitxers exportats per diferents servidors. Un cop estiguin muntats en l'estació client

semblaran sistemes de fitxers locals. Per tal de poder muntar un sistema de fitxers NFS necessitem conèixer l'adreça del servidor i la ruta del directori exportat.

Si preparem els directoris **/mnt/dades** i **/mnt/temporal** com a punt de muntatge, podrem muntar els sistemes de fitxers remots mitjançant les ordres següents.

```
reached:~ # mount 127.0.0.1:/mnt/disc0/dades /mnt/dades
reached:~ # mount 127.0.0.1:/mnt/disc0/temporal /mnt/temporal
```

On podrem comprovar que els sistemes de fitxers remots estan disponibles localment mitjançant l'ordre **`mount`** i **`df`**.

```
reached:~ # mount
/dev/hda2 on / type ext3 (rw,acl,user_xattr)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
debugfs on /sys/kernel/debug type debugfs (rw)
udev on /dev type tmpfs (rw)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
securityfs on /sys/kernel/security type securityfs (rw)
/dev/loop0 on /mnt/disc0 type ext3 (rw,usrquota,grpquota)
/dev/loop1 on /mnt/disc1 type xfs (rw,usrquota,grpquota)
nfsd on /proc/fs/nfsd type nfsd (rw)
127.0.0.1:/mnt/disc0/dades on /mnt/dades type nfs (rw,addr=127.0.0.1)
127.0.0.1:/mnt/disc0/temporal on /mnt/temporal type nfs (rw,addr=127.0.0.1)
reached:~ # df -h
S. fitxers      Mida    En ús   Lliure     %Ús     Muntat en
/dev/hda2       9,2G     5,5G     3,3G     63%     /
udev            253M      64K     253M      1%     /dev
/dev/loop0       97M     4,1M      88M      5%     /mnt/disc0
/dev/loop1       92M     240K       92M      1%     /mnt/disc1
127.0.0.1:/mnt/disc0/dades
                 97M     4,1M      88M      5%     /mnt/dades
127.0.0.1:/mnt/disc0/temporal
                 97M     4,1M      88M      5%     /mnt/temporal
reached:~ #
```

Fixeu-vos que, per defecte, mount intenta muntar tots els sistemes de fitxers en mode de lectura/escriptura. És per això que ens indica que **/mnt/dades** està muntat en mode **`rw`**. Però el servidor no permetrà fer escriptures. De la mateixa manera, si l'administrador de la màquina client volgués impedir les escriptures a Temporal, podria especificar l'opció **`ro`** en muntar el sistema de fitxers.

Opcions per muntar sistemes de fitxers NFS

A l'hora de muntar un sistema de fitxers es poden especificar opcions generals acceptades per tots els sistemes de fitxers (**`ro`**, **`rw`**, **`noatime`**, **`nodiratime`**...) i d'altres d'específiques del sistema de fitxers, en aquest cas, pròpies de l'NFS.

Opcions de muntatge per NFS

L'ordre mount accepta diferents opcions de muntatge, algunes són generals i d'altres depenen del sistema de fitxers. **`man mount`** i **`man 5 nfs`** ens mostraran la documentació d'aquestes opcions.

Entre les opcions específiques de l'NFS trobem les següents:

- **rsize** i **wsize**. Permeten especificar el nombre de bytes que es llegiran i s'escriuran a cada operació de xarxa. El seu valor per defecte és 1024, però s'obté una millora en el rendiment si s'utilitza 8192.

- **acregmin**, **acregmax**, **acdirmin**, **acdirmax**, **actimeo**. Permeten fixar els temps mínims i màxims pels qual els clients poden guardar en la seva memòria cau els atributs dels fitxers i directoris, per tal d'estalviar consultes mitjançant la xarxa. Es pot desactivar completament la *cache* d'atributs als clients mitjançant l'opció **noac**.

- **retry**. Permet indicar el temps, en minuts, per reintentar muntar un sistema d'arxius NFS. El valor per defecte és prou gran, 10.000 minuts.

- **bg**, **fg**. Si en intentar muntar el sistema de fitxers es produeix un *time out*, s'intentarà muntar el sistema de fitxers en segon pla (**bg**, *background*) o en primer pla (**fg**, *foreground*). L'opció per defecte és **fg**.

- **hard**, **soft**. Controlen si el client NFS notifica al programa que està accedint a un sistema de fitxers NFS els errors en la comunicació, o simplement bloqueja la crida i continua reintentant. Amb l'opció per defecte, **hard**, el programa no s'assabenta dels errors de comunicació entre el client i el servidor. Simplement la funció d'entrada/sortida no retorna fins que no es completa l'operació. Amb **soft**, el client NFS comunica al procés que intenta accedir al sistema de fitxers un error després d'un nombre de retransmissions. Configurable mitjançant l'opció **retrans**. No tots els processos estan preparats per rebre aquesta notificació d'error. Per això no s'acostuma a utilitzar l'opció **soft**.

- **tcp**, **udp**. Determinen el protocol de transport. Les primeres versions de l'NFS feien servir el protocol UDP; les noves versions (NFS v3) utilitzen per defecte el protocol TCP, però poden fer servir l'UDP si el servidor no suporta TCP. És molt important no utilitzar NFS amb el protocol UDP en enllaços d'1 Gbps, ja que les dades transportades es podrien malmetre.

Advertència sobre l'ús de l'NFS sobre UDP en enllaços ràpids

Quan s'utilitza UDP com a protocol de transport per a NFS a xarxes ràpides, com Gigabit Ethernet, es pot produir el malmetement silenciós, no detectat, de dades. La raó té a veure amb la reconstrucció de fragments que duu a terme el protocol IP.

MTU

La unitat màxima de transferència, *maximun transfer unit* (MTU), és el datagrama més gran que pot travessar una xarxa, o qualsevol de les seves capes, sense fragmentació.

Atès que una xarxa Ethernet utilitza una MTU de 1.500 bytes, i els servidors NFS per tal de millorar el rendiment intenten fer lectures i escriptu-

res grans, típicament de 8.192 bytes, és clar que a nivell de xarxa cal fraccionar les operacions NFS i tornar-les a reconstruir. El protocol IP és capaç de fraccionar els paquets UDP. Per identificar tots els fragments que corresponen a un mateix paquet UDP s'utilitza un identificador de 16 bits. Aquest identificador permet al receptor reconstruir el paquet original a partir dels fragments. Si en un temps determinat, per defecte trenta segons, no s'obtenen tots els fragments necessaris per reconstruir un paquet, el nucli abandona els fragments rebuts.

El problema és que un servidor NFS funcionant de manera intensa amb un enllaç ràpid pot transmetre més de 65.536 paquets en trenta segons. De manera que el receptor pot acumular fragments que corresponen a diferents paquets però tenen el mateix identificador de paquet. Quan això passa és possible introduir errors en la reconstrucció dels paquets.

Normalment, els errors de reconstrucció es detecten en les capes superiors. El mateix protocol UDP calcula una suma de verificació, *checksum*, de 16 bits que normalment detecta l'error i descarta el paquet. Però atès que el *checksum* solament és de 16 bits hi ha una possibilitat entre 65.536 que un paquet mal reconstruït tingui una suma de comprovació aparentment vàlida. En aquest cas s'haurà produït el malmetement silenciós de les dades.

Aquesta és la raó per la qual es recomana emprar TCP com a transport per NFS sempre que estigui disponible.

Si cal utilitzar NFS sobre UDP en una xarxa ràpida es pot considerar el següent:

Trames Jumbo

Són trames Ethernet que excedeixen l'MTU típica de 1.500 bytes.

Normalment, els equips Gigabit Ethernet accepten trames Ethernet grans, fins a 9.000 bytes.

- Configurar l'ús de trames Jumbo. Molts dispositius de xarxa Gigabit Ethernet suporten trames de 9.000 bytes. D'aquesta manera les operacions NFS de fins 8K no es fragmenten.

- Reduir la temporització o *timeout* per a la reconstrucció de paquets.

1.3. Samba i el sistema de fitxers en xarxa SMB/CIFS

Les diferents versions del Windows utilitzen de manera nativa el protocol SMB/CIFS per accedir a recursos compartits mitjançant una xarxa com impressores o dispositius d'emmagatzematge remot.

El protocol SMB, *server message block*, va ser desenvolupat per IBM. El protocol emprat en el Windows té moltes modificacions introduïdes per Microsoft. L'any 1996 Microsoft va renombrar la seva versió com a CIFS, *common Internet file system*.

Atesa la necessitat d'interaccionar mitjançant SMB/CIFS amb ordinadors Windows, es va desenvolupar el projecte Samba. El projecte Samba, inicialment desenvolupat per Andrew Tridgell, va néixer amb la intenció de fer enginyeria inversa a la versió del protocol emprada per Microsoft. Per obtenir una eina lliure que pogués implementar el protocol en diferents sistemes operatius per tal de garantir la interoperabilitat en la xarxa. Actualment, Samba és l'opció estàndard per als sistemes operatius Unix per compartir recursos amb màquines Windows. Un ordinador amb Samba pot actuar com a client o com a servidor. El Samba és un programari lliure GPL.

rsync

Andrew Tridgell és, amb Paul Mackerras, coautor de l'eina rsync.

Mitjançant rsync es poden sincronitzar directoris o fitxers de diferents equips. Només es transfereixen les diferències, així s'aconsegueix un bon rendiment, la qual cosa fa de rsync una opció excel·lent per actualitzar còpies de seguretat.

1.3.1. Conceptes relacionats amb l'SMB/CIFS

GPL

La GNU General Public License (GPL) és la llicència de programari més emprada pel projecte **GNU**.

El projecte GNU iniciat el 1983 per Richard Stallman té per objectiu el desenvolupament d'una quantitat suficient de programari lliure que permeti viure sense necessitat de cap programa que no sigui lliure.

Actualment, és possible emprar sistemes operatius, com el GNU/Linux, amb milers d'aplicacions que fan servir la llicència GPL.

L'SMB/CIFS és un protocol de nivell d'aplicació que permet accedir a sistemes de fitxers i impressores mitjançant una xarxa. Tradicionalment, l'SMB/CIFS utilitza els serveis de xarxa proporcionats per NetBIOS/NetBEUI. Els dos noms d'aquest protocol corresponen a la versió bàsica i la versió estesa, i va ser dissenyat l'any 1983 per petites xarxes d'àrea local.

Algunes de les seves característiques són les següents:

- **Assigna un nom a cada estació de xarxa.** Aquests noms, que no són jeràrquics com el DNS, es resolen mitjançant el servei de noms NetBIOS *naming server*, NBNS, que Microsoft anomena WINS. Els servidors de noms permeten a les estacions de la xarxa registrar el seu propi nom de manera dinàmica.

- **Els missatges NetBIOS/NetBEUI es poden encapsular dins de diferents protocols.** Actualment, el protocol de xarxa subjacent és TCP/IP, però fa uns anys era IPX/SPX. El protocol NBT és el que s'utilitza per transportar NetBIOS/NetBEUI a sobre de TCP/IP.

Els RFC 1001 i RFC 1002 especifiquen com s'encapsula NetBIOS/NetBEUI sobre el protocol TCP/IP.

A partir del Windows 2000, el protocol SMB/CIFS es pot utilitzar directament sobre TCP/IP sense necessitat d'emprar NetBIOS/NetBEUI, però els clients antics no podran treballar en xarxa. En l'actualitat, almenys en teoria, el servei NBNS (WINS) no és necessari si hi ha disponible un servidor DNS, malgrat que algunes aplicacions, com Microsoft Exchange Server 3, el necessitaran per desplegar tota la seva funcionalitat. El servei Samba també pot treballar com un servei NBNS (WINS).

Podeu accedir a més informació sobre els RFC en la secció "Adreces d'interès" del web del crèdit.

El Windows Vista ha introduït una nova versió del protocol SMB/CIFS anomenada SMB 2.0. Aquesta nova versió ha estat desenvolupada per Microsoft i aporta mecanismes per millorar el rendiment en xarxa, augmentar les capacitats del protocol i reduir les variacions. Des que IBM va

desenvolupar el protocol SMB fins ara, s'han fet moltes modificacions que obligaven a utilitzar un programari complex que pogués tractar els diferents clients. Ara SMB 2.0 simplifica tots aquests casos. La versió 4 de Samba aporta suport experimental per a SMB 2.0.

1.3.2. Descripció del Samba

El Samba és un ampli conjunt de programari GPL que permet als ordinadors GNU/Linux (i d'altres Unix) interaccionar amb equips Windows. Un equip amb Samba pot fer el següent:

- Actuar com a client d'un equip Windows per tal d'accedir a recursos en xarxa com dispositius d'emmagatzematge i/o impressores.

- Actuar com a servidor per a equips Windows o altres clients Samba, proporcionant accés a recursos d'emmagatzematge, impressores, servei d'autenticació d'usuaris i resolució de noms NBNS/WINS.

Aquesta funcionalitat té associada la complexitat pròpia dels diferents subsistemes relacionats. El Samba és fàcil d'emprar, però per utilitzar el Samba amb eficàcia sovint és necessari conèixer amb detall el sistema de permisos dels fitxers i la configuració d'impressores tant en l'Unix com en el Windows, i saber resoldre noms amb NBNS/WINS i DNS, i autenticar els usuaris.

1.3.3. El Samba funcionant com a client

L'eina **`smbclient`** permet dialogar amb un servidor SMB/CIFS (que pot ser una màquina amb Samba o una estació Windows) de manera interactiva, amb un mecanisme molt semblant al que empraríem mitjançant un client FTP. Entre les opcions disponibles trobem les següents:

- Identificació remota mitjançant usuari/contrasenya.
- Consulta dels recursos compartits.
- Connexió a un recurs compartit.
- Transferència de fitxers i manipulació dels sistemes de fitxers locals i remots. Fins i tot és possible fer un arxiu tar amb el contingut d'un recurs remot.
- Impressió de fitxers en l'estació remota.
- Consulta de la cua d'impressió remota.
- Enviament de missatges WinPopup.

Per tal d'accedir als recursos compartits amb la màxima comoditat, es poden muntar com si fossin locals. L'ordre **`smbmount`** permet muntar un sis-

tema de fitxers SMB/CIFS remot en el punt de muntatge indicat. En l'operació de muntatge cal especificar com a paràmetres l'usuari, la contrasenya, el servidor i el recurs compartit que es vol muntar.

També és possible emprar directament (això és el més comú) l'ordre **`mount`** del sistema amb el paràmetre **`-t smbfs`** i les opcions adequades. Per desmuntar el recurs compartit es pot emprar **`smbumount`** o bé directament **`umount`**.

Si l'administrador vol que es munti de manera automàtica cert recurs compartit cada vegada que el sistema s'inicialitzi, pot escriure una línia adequada en el fitxer **/etc/fstab**. Però aquesta no és una bona idea si el recurs compartit està protegit mitjançant una contrasenya que ha de romandre secreta. En aquest cas, es pot emprar el fitxer **/etc/samba/smbfstab**, que bàsicament té la mateixa funcionalitat, però com que només el pot llegir l'administrador permet que els usuaris no llegeixin les contrasenyes.

A més a més de `smbclient` i `smbmount` hi ha altres eines útils:

- **`smbcacls`**. Gestiona les llistes de control d'accés (ACL) als fitxers o directoris.
- **`smbcquotas`**. Gestiona les quotes als recursos NTFS5.
- **`smbget`**. Permet baixar fitxers dels recursos compartits com ho fa el Wget del web.
- **`smbspool`**. Envia un fitxer a una impressora SMB.
- **`smbtree`**. És un petit navegador de l'entorn de xarxa per a la consola.

1.3.4. El Samba funcionant com a servidor

Directori actiu

El directori actiu és una tecnologia de Microsoft que proporciona diferents serveis de xarxa: servei de directori (similar a l'LDAP), servei d'autenticació i servei d'informació a la xarxa.

El directori actiu és una base de dades distribuïda que emmagatzema diferents objectes, com usuaris o impressores. Els ordinadors que tenen part d'aquesta base de dades són servidors que executen el servei de directori actiu.

El servei Samba permet que una estació degudament configurada ofereixi recursos compartits en xarxa mitjançant SMB/CIFS a màquines Windows, bàsicament sistemes de fitxers i impressores, però també els ofereix el servei de resolució de noms i autenticació d'usuaris, definits en el seu propi fitxer de configuració, i fa de controlador de domini (PDC) o, fins i tot, de servidor de directori actiu.

Configuració del Samba

Atès el caràcter introductori d'aquest material, ens fixarem en les possibilitats més bàsiques dels fitxers de configuració.

La configuració del Samba es pot emmagatzemar en diferents llocs. Per exemple, un servidor amb un nombre reduït d'usuaris pot emprar un fitxer per registrar els usuaris mentre que un servidor d'una organització

més gran pot emprar un SGBD com MySQL o, fins i tot, un servei de directori com l'OpenLDAP.

En la Linkat 2, els fitxers de configuració del Samba es troben a **/etc/samba**. El fitxer de configuració principal és **/etc/samba/smb.conf**, que té la pròpia pàgina de manual. El fitxer **smb.conf** es troba dividit en seccions dedicades a recursos compartits (directoris o impressores) particulars i especials com [global], [homes] i [printers]:

SGBD

Un SGBD és un sistema gestor de bases de dades, és a dir, un programa que té per objectiu facilitar la gestió d'un conjunt de dades a una base de dades.

Alguns SGBD lliures molt emprats són MySQL i PostgreSQL.

- La secció [global] estableix opcions generals per a tot el servidor, que es converteixen en valors per defecte per a la resta de seccions. En aquesta secció, típicament s'estableix el nom del grup de treball i el servei d'impressió a emprar (actualment, CUPS en la majoria de distribucions GNU/Linux o MacOS X).

- La secció [homes] estableix opcions per als directoris personals dels usuaris. No cal afegir una configuració particular per exportar en xarxa el directori personal de cada usuari, tot i que també es pot fer de manera particular.

- La secció [printers] permet establir opcions per a les impressores exportades. D'aquesta manera no caldrà afegir una configuració particular per exportar cada impressora, tot i que es pot fer. La secció [printers] permet exportar totes les impressores definides en el fitxer **/etc/printcap**.

Es poden definir tots els recursos compartits que siguin necessaris de manera particular. Per a cada recurs cal definir una secció amb el seu nom [*recurs_compartit*] i totes les propietats. La sintaxi és prou flexible per definir amb força detall el tipus d'accés (lectura i/o escriptura), qui el podrà realitzar (qualsevol, un usuari autenticat, un membre d'un grup), des d'on es podrà realitzar (adreça i/o xarxa del client), i altres paràmetres com el mode de creació per als nous fitxers i/o directoris, o el seu usuari/grup.

El fitxer de configuració **/etc/samba/lmhosts** proporciona un mitjà per associar noms de xarxa NetBIOS amb adreces IP, tal com ho fa el fitxer equivalent al Windows.

El fitxer **/etc/samba/smbpasswd** és la font per a l'autenticació d'usuaris del Samba, que no cal que siguin usuaris del mateix sistema en què s'executa el Samba.

El fitxer **/etc/samba/smbusers** permet mapar usuaris clients amb usuaris locals.

Altres eines d'utilitat per al servidor Samba són les següents:

- **`smbstatus`**. Informa sobre les connexions actuals del servidor.

- **`testparm`**. Repassa el fitxer de configuració per tal de trobar possibles errors.
- **`smbcontrol`**. Envia missatges als dimonis *smbd*, *nmbd* o *winbindd*.
- **`smbpasswd`**. Permet establir la contrasenya per als usuaris Samba.
- **SWAT**. Eina d'administració web del mateix servidor Samba.

2. Seguretat

Bruce Schneier afirma que la seguretat és un procés, no un producte. De la mateixa manera que una porta blindada, al bell mig del no res, no és suficient per garantir l'accés a un indret protegit, la instal·lació d'un tallafoc o qualsevol altra eina en la nostra xarxa no la pot assegurar de manera definitiva. És necessari contemplar la seguretat amb una mirada àmplia i mantenir una actitud constant de revisió de les amenaces i prevenció dels atacs.

Bruce Schneier

Bruce Schneier és un reconegut criptògraf i expert en seguretat que ha desenvolupat, i codesenvolupat, diversos algorismes de xifratge molt coneguts. Entre ells, un que s'anomena *solitari* i que es pot fer anar mitjançant una baralla de cartes.

Sovint, la màxima seguretat d'una xarxa implica desconnectar-la. Aquesta obvietat ens pot servir per il·lustrar dues idees:

- No tot ha d'estar connectat en xarxa.
- La seguretat sempre imposa limitacions.

Un pas previ al plantejament de mesures de seguretat en un sistema és decidir quins sistemes i serveis han d'estar interconnectats. Després, en el procés continu per garantir la seguretat s'imposaran limitacions per impedir usos inadequats dels recursos que a vegades afectaran els usos legítims, i sovint faran més pesat el treball normal amb el sistema. Cal trobar un equilibri entre les limitacions i la funcionalitat, tenint en compte que la seguretat que entorpeix massa el treball és probablement la pitjor, ja que tothom la deshabilita.

En el procés d'assegurar un sistema es valorarà com a mínim el següent:

- La seguretat física
- Les amenaces internes
- Les amenaces externes
- Els possibles accidents, avaries i desastres

Un sistema que no impedeix l'accés físic és un sistema vulnerable. Totes les mesures preses per limitar l'accés a nivell d'aplicació no poden impedir el robatori dels dispositius d'emmagatzematge, la interrupció del subministrament elèctric o la intercepció de la nostra xarxa. Tecnologies com el xifratge poden pal·liar els efectes adversos d'algunes d'aquestes situacions. Si ens roben un disc dur és infinitament millor si les dades estiguin xifrades que si no ho estan, però la seguretat física continua sent una qüestió a tenir en compte.

Les amenaces internes són una font de perill constant. Qualsevol usuari corrent del sistema parteix d'una posició inicial que li permet provocar un

dany greu. D'altra banda, el procés de seguretat ha de considerar les amenaces externes, que si es presten serveis a Internet són pràcticament innombrables.

Altres esdeveniments extraordinaris com els accidents, inundacions, focs i avaries també impliquen riscos de seguretat. En aquest cas, la solució comuna pot ser un servei correcte de còpies de seguretat i sistemes redundants, però garantir l'emmagatzematge segur de les dades és tot un problema per si mateix.

2.1. Seguretat interna, controlant els processos i usuaris

Els sistemes operatius inclouen els propis sistemes per garantir la compartició d'informació mitjançant un sistema de permisos als fitxers. És ben conegut el sistema de permisos de l'Unix per als fitxers, els dispositius i els segments de memòria dels processos. Aquesta és una característica bàsica d'un sistema operatiu multiusuari, però es pot estendre per millorar la seguretat.

2.1.1. Llistes de control d'accés (ACL)

El model bàsic de permisos de l'Unix es pot estendre utilitzant llistes de control d'accés. Aquest és el model de seguretat nadiu del Windows NT i els seus derivats.

Una llista de control d'accés és una llista de permisos enganxada en un objecte. La llista especifica qui, usuari o procés, pot executar determinada operació sobre l'objecte. Cada vegada que algú intenta realitzar una operació sobre l'objecte cal consultar la llista de control d'accés per saber si es permet o es denega l'operació.

Una qüestió clau és definir qui edita una ACL i quins canvis pot fer. Si l'usuari propietari, el seu creador, pot editar sense restriccions la llista de control d'accés, per exemple per concedir accés total a altres usuaris, es diu que s'està emprant un *discretionary access control*. Si el sistema permet definir unes restriccions que s'imposen a tot el sistema que anul·len l'establert per les ACL, aleshores s'està emprant un *non-discretionary access control*, també anomenat *mandatory access control*.

En la Linkat 2 es troba suport per a les ACL. Les eines **`getfacl`** i **`setfacl`** permeten respectivament consultar i canviar l'ACL d'un fitxer o directori. La implementació de les ACL en el GNU/Linux està basada en els atributs estesos. Un atribut estès està format per l'identificador de l'atribut i el seu valor, i és semblant a una variable d'entorn però aplicat al sistema de fitxers. Cada

fitxer pot tenir els propis atributs estesos. El suport per a atributs estesos depèn del sistema de fitxers, però la majoria dels sistemes de fitxers actuals els suporten. De fet, els sistemes de fitxers ext2, ext3, ReiserFS, XFS i JFS implementen atributs estesos i, per tant, poden suportar ACL.

Exemple d'aplicació de les ACL

En definir un fitxer nou, només compta amb els permisos estàndard. Però amb **`getfacl`** podem comprovar com els permisos es poden representar mitjançant una ACL equivalent.

```
reached:~ # echo Hola >prova.txt
reached:~ # ll prova.txt
-rw-r--r-- 1 root root 5 2008-03-03 23:16 prova.txt
reached:~ #
reached:~ # getfacl prova.txt
# file: prova.txt
# owner: root
# group: root
user::rw-
group::r--
other::r--
```

La llista de **`getfacl`** inclou una capçalera de tres línies, que es pot ometre amb l'opció **`--omit-header`**. La capçalera mostra el nom del fitxer, el propietari i el grup. Després es mostra la llista d'entrades en l'ACL. En aquest cas, tres entrades que tenen el mateix significat que els permisos de l'Unix.

Emprant **`setfacl`** podem afegir una nova entrada en l'ACL del fitxer **prova.txt** de manera que l'usuari **usuari** tingui un accés complet.

```
reached:~ # setfacl -m user:usuari:rwx prova.txt
reached:~ # ll prova.txt
-rw-rwxr--+ 1 root root 5 2008-03-03 23:16 prova.txt
reached:~ # getfacl prova.txt
# file: prova.txt
# owner: root
# group: root
user::rw-
user:usuari:rwx
group::r--
mask::rwx
other::r--
```

En aquesta última llista podem veure el següent:

- L'ordre **`ls -l`** mostra un caràcter + després dels caràcters que representen els permisos estàndard per tal d'assenyalar que el fitxer conté una ACL pròpia.
- La llista de **`getfacl`** mostra dues noves entrades: una especifica el mode d'accés particular per a l'usuari *usuari*. L'entrada 'mask' s'ha creat de manera automàtica a partir de la unió dels permisos del grup propietari i tots els usuaris i grups especificats pel seu nom.

2.1.2. Registre dels programes executats pels usuaris: *process accounting*

Els fitxers de registre són una eina fonamental per a l'administració del sistema. El paquet **`acct`** és una eina bàsica de seguretat, i la seva funció és registrar l'execució de tots els programes d'usuari, de manera que l'ad-

ministrador pugui esbrinar en tot moment qui va executar què i en quin moment.

Un cop instal·lades aquestes eines, l'administrador podrà obtenir informes sobre l'execució dels diferents processos mitjançant les ordres **ac** i **sa**. També podrà controlar la comptabilitat de processos mitjançant l'ordre **accton**. Finalment, l'ordre **last** li mostrarà una llista amb les connexions/desconnexions dels usuaris en el sistema i els seus reinicis.

2.1.3. AppArmor

El Linux inclou un sistema de control d'accés discrecional (*discretionary access control*, DAC) en què un usuari propietari d'un fitxer pot permetre l'accés total a d'altres usuaris. En un entorn on la seguretat és una prioritat, sovint és necessari imposar límits a tot el sistema per evitar fugues d'informació, de manera que un usuari no pugui compartir determinats fitxers amb altres usuaris encara que sigui el propietari. Aquest últim sistema de control d'accés s'anomena *obligatori* (*mandatory access control*, MAC). A la pràctica, el control d'accés no només implica els fitxers, sinó també les connexions de xarxa, segments de memòria i, en definitiva, qualsevol objecte gestionat pel sistema operatiu.

Un sistema de control d'accés obligatori permet especificar restriccions individuals per processos, per evitar que una aplicació funcionant indegudament, tal vegada perquè ha rebut una entrada malintencionada d'un atacant, no pugui fer allò que se suposa que no ha de fer.

Programari fiable i programari segur

Segons Ivan Arce, cap del departament tecnològic de Core Security Technologies, "El programari fiable fa el que se suposa que ha de fer. El programari segur fa el que se suposa que ha de fer... i res més".

Per tal d'implementar un control d'accés obligatori al GNU/Linux, es va desenvolupar una nova interfície per al nucli anomenada Linux Security Modules (LSM). Aquesta interfície permet implementar sistemes de control d'accés obligatori en el nucli. Dos exemples són AppArmor i SELinux.

AppArmor és l'opció instal·lada per defecte en la Linkat2 (i d'altres distribucions com openSUSE i Ubuntu). No necessita cap suport especial del sistema de fitxers, de manera que pot funcionar, fins i tot, sobre els sistemes de fitxers en xarxa NFS. El seu funcionament es basa en la creació de perfils amb la definició de les operacions permeses per a cada aplicació sobre els diferents objectes amb els quals treballa.

AppArmor pot funcionar en dos modes diferents: *learning*/*complain* i *enforcing*. El primer mode de funcionament permet totes les operacions, fins i tot les prohibides pel perfil de seguretat. Les operacions que violen el perfil establert es registren de manera que la creació i modificació de nous perfils sigui fàcil. Si AppArmor funciona en el mode *enforcing* no-

més es permeten les operacions establertes en el perfil de l'aplicació, la resta es deneguen.

Per carregar les polítiques de seguretat i per informar l'administrador del sistema, AppArmor empra el sistema de fitxers virtual del Linux **securityfs**, que normalment es troba muntat a **/sys/kernel/security**. Si AppArmor està instal·lat degudament, el fitxer **/sys/kernel/security/apparmor/profile** posarà en una llista els diferents perfils carregats pel nucli i el mode d'operació de cadascun.

Per exemple, en una instal·lació nova de la Linkat2 trobem el següent:

```
reached:/sys/kernel/security/apparmor # cat profiles
/usr/sbin/traceroute (enforce)
/usr/sbin/ntpd (enforce)
/usr/sbin/nscd (enforce)
/usr/sbin/named (enforce)
/usr/sbin/mdnsd (enforce)
/usr/sbin/identd (enforce)
/sbin/syslogd (enforce)
/sbin/syslog-ng (enforce)
/sbin/klogd (enforce)
/bin/ping (enforce)
reached:/sys/kernel/security/apparmor #
```

Els perfils es troben a **/etc/apparmor.d**, i utilitzen una sintaxi senzilla ben definida en la pàgina de manual d'**apparmor.d**. Per exemple, el perfil dedicat al programa Ping és el següent:

```
#include <tunables/global>

/bin/ping {
#include <abstractions/base>
#include <abstractions/consoles>
#include <abstractions/nameservice>

capability net_raw,
capability setuid,

/bin/ping mixr,
/etc/modules.conf r,
}
```

Bàsicament s'enumeren els fitxers als quals pot accedir qualsevol procés ping i el seu mode d'accés. També cal enumerar les habilitats (*capabilities*) dels processos ping. Les sentències **#include** permeten compartir fragments comuns dels perfils en emmagatzemar-los en diferents fitxers que es poden reutilitzar.

L'administrador en l'Unix

Normalment, el nom de l'administrador, o superusuari, en l'Unix és *root*.

Però l'usuari administrador pot tenir qualsevol nom. La raó per la qual és l'administrador és perquè el seu UID (identificador d'usuari) és 0.

Linux i les habilitats (capabilities)

Tradicionalment, els sistemes operatius Unix tenen un sistema de permisos que distingeix entre els processos privilegiats (els de l'usuari amb UID

0) i els no privilegiats (la resta). El sistema de permisos examina les credencials (UID efectiu, GID efectiu i llista de grups suplementaris) de cada procés per determinar si pot realitzar l'accés o no. Totes aquestes comprovacions s'obvien si es tracta d'un procés privilegiat, ja que un procés privilegiat ho pot fer tot.

A partir de la versió del nucli Linux 2.2, totes les habilitats tradicionals dels processos privilegiats es van començar a dividir en diferents unitats anomenades *habilitats* (*capabilities*). Aquestes habilitats es poden activar o desactivar de manera individual per a cada fil d'execució. De tal manera que un procés que, per exemple, necessita fer ús dels sòcols (*sockets*) RAW ho pot fer si té concedida l'habilitat **CAP_NET_RAW** sense haver de ser un procés privilegiat. Aquest model de seguretat té una granularitat molt més fina que el tot o res basat en el model de processos privilegiats i no privilegiats.

Es pot consultar la llista d'habilitats definides pel nucli en la pàgina de manual **capabilities(7)**. Aquestes són les habilitats que AppArmor pot concedir de manera individual a cada procés.

Threads

Els sistemes operatius que permeten realitzar de manera concurrent diferents tasques normalment empren un procés per a cada tasca.

El planificador de tasques del sistema operatiu assigna els processos a les CPU disponibles. Alguns sistemes operatius com el GNU/Linux permeten tenir més d'un fil d'execució, o *thread*, en cada procés. En aquest cas s'aconsegueix una millora del rendiment, ja que és menys costós commutar l'execució de diferents fils en el mateix procés que passar d'executar un procés a un altre.

Pàgines de manual

En el GNU/Linux es poden consultar de manera interactiva les pàgines de manual. Aquestes pàgines estan ordenades en diferents capítols: les ordres d'usuari, les ordres d'administració, els fitxers de configuració, etc.

L'ordre **man** ens permet consultar aquestes pàgines. Per exemple, per consultar la pàgina dedicada a les habilitats del Linux faríem `man 7 capabilities`.

Eines per treballar amb AppArmor

Els programes **complain**/**enforce** canvien el mode d'un perfil. El mode *complain* d'un perfil s'especifica mitjançant la presència de la cadena **flags=(complain)** després de la identificació del programa, la seva absència és pròpia del mode *enforcing*.

Per exemple:

```
#include <tunables/global>

/bin/ping flags=(complain) {
#include <abstractions/base>
#include <abstractions/consoles>
#include <abstractions/nameservice>

capability net_raw,
capability setuid,

/bin/ping mixr,
/etc/modules.conf r,
}
```

L'eina **genprof** està pensada per a la creació automatitzada de perfils. En executar-la indicant una aplicació (per a la qual es generarà el perfil), es crea un perfil nou en mode *complain*. L'usuari tindrà l'oportunitat de provar les diferents funcionalitats de l'aplicació de manera que, en el fitxer de registre d'AppArmor, quedaran reflectits tots els accessos que violin alguna política de seguretat. L'eina **genprof** permet repassar el fitxer de registre per deduir els permisos que cal concedir al perfil que s'està generant perquè l'aplicació pugui funcionar. Una vegada que es pensa que el perfil és suficientment complet es passa al mode *enforce*.

L'eina **unconfined** busca ports oberts per processos que no tenen un perfil d'AppArmor associat. En una instal·lació nova del Linkat2 **unconfined** fa una llista amb *portmap*, *cupsd*, *zmd-bin*, *postfix*, *dhclient* i *sshd*.

2.1.4. Security-Enhanced Linux, SELinux

SELinux és una implementació del control d'accés obligatori (*mandatory access control*, MAC) per al nucli Linux construït per la National Security Agency (NSA). És una solució alternativa a AppArmor, amb una gran funcionalitat i flexibilitat, però que moltes vegades ha estat criticada per la dificultat excessiva per implantar-la i mantenir-la.

National Security Agency

La National Security Agency (NSA) és una organització del Govern dels EUA encarregada d'obtenir i analitzar la informació transmesa per qualsevol mitjà, a més d'assegurar les comunicacions pròpies envers a organitzacions equivalents d'altres països.

Algunes de les diferències clau entre SELinux i AppArmor són les següents:

- AppArmor identifica els fitxers mitjançant la seva ruta, mentre que SELinux utilitza els nodes d'identificació.

- SELinux es basa en etiquetes aplicades als objectes, i en el cas del sistema de fitxers és necessari que hi hagi suport per als atributs estesos. AppArmor no necessita etiquetar els fitxers.

2.1.5. Sistemes de detecció d'intrusions

Si un atacant compromet el sistema informàtic, és molt important que l'administrador ho detecti immediatament. Per aquesta raó es fa servir programari que revisa de manera periòdica els fitxers del propi sistema operatiu i adverteix l'administrador de canvis en aquests.

Normalment, les eines de detecció d'intrusions fan servir tècniques criptogràfiques per generar sumes de verificació per als fitxers que han de controlar. Després, recalculen de manera periòdica la suma de verificació per als fitxers i la comparen amb la suma original. Per aquesta raó és molt important que l'atacant no pugui regenerar o alterar el fitxer en què s'han

emmagatzemat les sumes de verificació originals. Per posar les coses realment difícils a l'atacant, un bon administrador pot fer servir un dispositiu d'emmagatzematge de només de lectura per registrar aquesta informació. De fet, un administrador de sistemes podria emprar el mateix dispositiu per emmagatzemar el nucli del sistema operatiu i totes les eines que mai no voldria veure compromeses malgrat que un atacant aconseguís accedir al sistema.

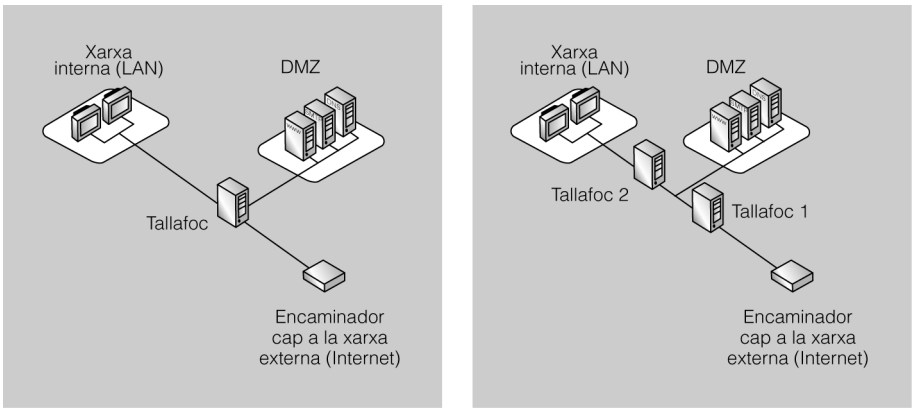
Tripwire és una de les eines més emprades per a la detecció d'intrusions.

2.2. Seguretat en la xarxa: tallafocs, DMZ i política per defecte

La funció fonamental d'un tallafoc (*firewall*) és regular el trànsit de xarxa, normalment entre xarxes amb diferents nivells de confiança. Com a exemple, pot servir qualsevol organització amb una LAN connectada a Internet. La LAN és una xarxa de confiança amb usuaris coneguts, però Internet és una xarxa sense confiança. Cal regular els accessos des d'Internet cap a la xarxa local, i potser també des de la LAN cap a Internet.

Tot i que no és l'únic lloc on es pot utilitzar un tallafoc, en la defensa perimetral de la xarxa és una necessitat. Si una organització vol mantenir alguns servidors accessibles des d'Internet, seria convenient formar una tercera xarxa amb aquest servidor per tal de no haver d'acceptar connexions des d'Internet cap a la LAN. Aquesta tercera xarxa on estan connectats els servidors accessibles des d'Internet s'anomena *zona desmilitaritzada* o DMZ (figura 3).

Figura 3. LAN, DMZ i xarxa externa amb un o dos tallafocs



Font: adaptació de la Viquipèdia

S'anomena *zona desmilitaritzada*, o DMZ, la xarxa en què es connecten els servidors que han de rebre connexions des de l'exterior. Una DMZ evita haver d'exposar tots els ordinadors de la xarxa interna a les connexions externes.

Els tallafocs tenen definida una política per defecte que pot consistir a denegar o permetre les connexions. Si per defecte es deneguen totes les connexions, caldrà aplicar regles al tallafoc per permetre els usos acceptats per la xarxa. Aquesta política és la més segura però exigeix un bon coneixement de totes les aplicacions que ha de suportar la xarxa. La política d'acceptar per defecte totes les connexions i només prohibir de manera explícita alguns usos de la xarxa és més fàcil de posar en marxa, però també és molt més insegura.

2.2.1. Tipus de tallafocs

Els tallafocs es poden classificar en funció de la capacitat per analitzar el trànsit. Els més senzills només poden prendre decisions basades en els camps de la capçalera IP, els més sofisticats són capaços d'entendre el protocol de la capa aplicació (com HTTP, FTP o SMTP) i prendre la decisió de denegar o acceptar la connexió en funció d'un gran ventall de situacions. Bàsicament, podem classificar els tipus de tallafocs de la manera següent:

1) Filtratge de paquets. El filtratge de paquets és la tècnica emprada pels tallafocs més simples i bàsics. Només poden acceptar o rebutjar paquets de manera independent a partir de les dades de la capçalera, normalment IP, i si es tracta de TCP o UDP, els ports d'origen/destinació.

Són fàcils de configurar, però malauradament no poden ser gaire flexibles. Si sempre s'utilitzen els ports estandarditzats (80 per a l'HTTP o 22 per a l'SSH) es pot tractar de permetre determinats protocols i excloure'n d'altres.

2) Filtratge de paquets amb estat. Aquests tallafocs representen internament les connexions que els travessen, de manera que poden prendre decisions no solament basant-se en la capçalera dels paquets sinó també en l'estat de la connexió a la qual pertanyen.

Exemple de filtratge de paquets amb estat

Un tallafoc amb estat (*stateful firewall*) pot acceptar paquets que pertanyen a connexions iniciades però no permetre que se n'iniciïn de noves.

3) Filtratge de paquets en la capa d'aplicació o tallafocs *proxy*. Són els tallafocs amb una major compressió del trànsit que regulen, i prenen decisions en funció de paràmetres propis de la capa d'aplicació. No es limiten a reenviar els paquets sense, o amb mínimes, variacions. Sempre actuen com a intermediaris en la comunicació. És a dir, per a les connexions permeses actuen com a intermediaris generant ells mateixos els paquets que envien als destinataris. Això ofereix una gran protecció contra atacs amb paquets malformats.

L'inconvenient principal del filtratge de paquets en la capa d'aplicació és que aquests tallafocs necessiten més potència, problema que s'aguditza si

s'espera que siguin capaços de filtrar grans volums de trànsit. Una altra limitació és que mentre la resta de tallafocs pot tractar de la mateixa manera qualsevol comunicació, els tallafocs *proxy* necessiten lògica específica per a cada protocol d'aplicació. Així, només se solen emprar aquest tipus de tallafocs en els protocols d'aplicació més comuns: HTTP, SMTP, DNS i FTP. En canvi, conèixer el trànsit que regulen els permet realitzar funcions avançades.

Exemple de funcions avançades d'un tallafoc proxy

Un *proxy* HTTP pot, per exemple, deixar passar només els paquets que pertanyen a una sessió vàlida a l'aplicació a la qual accedeix l'usuari, o impedir l'accés a determinats URL o continguts. Un *proxy* SMTP pot inspeccionar tots els correus electrònics en busca d'adjunts potencialment perillosos. Els usuaris també poden fer servir la criptografia, mitjançant la qual l'administrador veuria la informació xifrada encara que no la pogués interpretar, i l'esteganografia per tal d'ocultar el missatge real dins d'un altre missatge (pot ser un adjunt de correu electrònic) aparentment inofensiu.

Traducció d'adreces de xarxa (NAT)

La traducció d'adreces de xarxa (NAT) és un procediment mitjançant el qual es modifiquen les adreces de xarxa dels datagrames que travessen un encaminador o *router* per tal de mapar un espai d'adreces en un altre.

2.2.2. Introducció a netfilter/iptables

Netfilter és l'entorn de treball (*framework*) del nucli del Linux per interceptar i processar paquets de xarxa. La seva funcionalitat es pot aprofitar per construir tallafocs, però també serveix per fer NAT, fer el seguiment de les connexions, definir polítiques per a les cues de paquets en l'espai d'usuari, per filtrar a nivell d'adreça MAC i per modificar paquets de manera dinàmica.

Netfilter també proporciona eines d'usuari, de les quals **`iptables`** és la més coneguda. Tot i que iptables només és una eina d'usuari per administrar les regles del tallafoc i NAT sovint s'utilitza com a sinònim de tot l'entorn de treball.

Netfilter es pot estendre de moltes maneres, és possible escriure nous mòduls per a **`iptables`** i és possible escriure aplicacions que facin qualsevol processament dels paquets a l'espai d'usuari; de fet, terceres parts han desenvolupat diferents components per a netfilter. Però, pel que fa a la seva funció com a tallafoc, en la distribució bàsica, és un tallafoc amb control d'estat. Si pensem incloure un tallafoc *proxy* en la nostra xarxa haurem d'emprar aplicacions com Apache per al protocol HTTP, Squid per als protocols HTTP/FTP, i qualsevol servidor SMTP com Postfix o Qmail per al correu electrònic.

Taules i cadenes en iptables

L'eina **`iptables`** permet a l'administrador del sistema implementar polítiques per al filtratge de paquets. Aquestes polítiques es defineixen mitjançant un conjunt de regles que estan ordenades en manera de cadenes. Les cadenes pertanyen a taules, i les taules estan associades amb diferents tipus de processament de paquets.

Qualsevol paquet fa un recorregut, per la infraestructura de netfilter, que depèn de les seves circumstàncies: potser és un paquet que ha arribat per una interfície de xarxa en direcció a una aplicació local, o potser és per a una altra màquina i s'ha de reenviar, o és un paquet generat localment, etc.

El recorregut del paquet pel nucli implicarà un recorregut per diferents cadenes, com a mínim una. En cada cadena es valoraran en ordre les regles que la formen, i cada regla podrà coincidir amb el paquet o no. Cada regla està formada per un criteri de selecció de paquets i una destinació per als que coincideixin amb el criteri de selecció. Bàsicament, el destí serà rebutjar el paquet o bé deixar que faci el seu camí, també és possible fer una crida a una altra cadena.

Algunes cadenes ja estan predefinides. L'administrador en pot gestionar les regles però no pot esborrar aquestes cadenes, tot i que pot definir (i aquestes sí, esborrar) totes les cadenes auxiliars que necessiti. Les cadenes predefinides tenen una política per defecte, de tal manera que un paquet que arriba al final de la cadena és acceptat o rebutjat. Les cadenes definides per l'administrador no tenen política per defecte i, si un paquet arriba al final, aleshores es retorna a la cadena que va fer la crida a aquesta cadena, ja que les regles poden especificar crides a d'altres cadenes.

Les taules tenen funcions específiques. Les tres taules bàsiques són *FILTER* (filtratge), *NAT* (traducció d'adreces de xarxa) i *MANGLE* (peça). L'usuari no pot alterar de manera directa les taules, tot i que carregar nous mòduls pot afegir noves taules.

El nom de les taules i les cadenes predefinides és autodescriptiu. La seva classificació és la següent:

- **Taula *FILTER***. Tots els paquets passen per alguna de les cadenes d'aquesta taula. Així es pot filtrar, acceptar o rebutjar qualsevol paquet del nucli.

  - Cadena *INPUT*. Per a tots els paquets que arriben per alguna interfície de xarxa en direcció a aquest *host*.
  - Cadena *OUTPUT*. Per a tots els paquets generats per aquesta estació.
  - Cadena *FORWARD*. Per a tots els paquets que arriben per alguna interfície de xarxa en direcció a un altre *host*.

- **Taula *NAT***. Serveix per aplicar els diferents tipus de traducció d'adreces de xarxa disponibles, la qual cosa altera les adreces d'origen, destinació i/o els ports.

  - Cadena *PREROUTING*. Els paquets que arriben al nucli recorren aquesta cadena abans de prendre la decisió de l'encaminament. Permet fer DNAT, és a dir, alterar l'adreça de destinació.

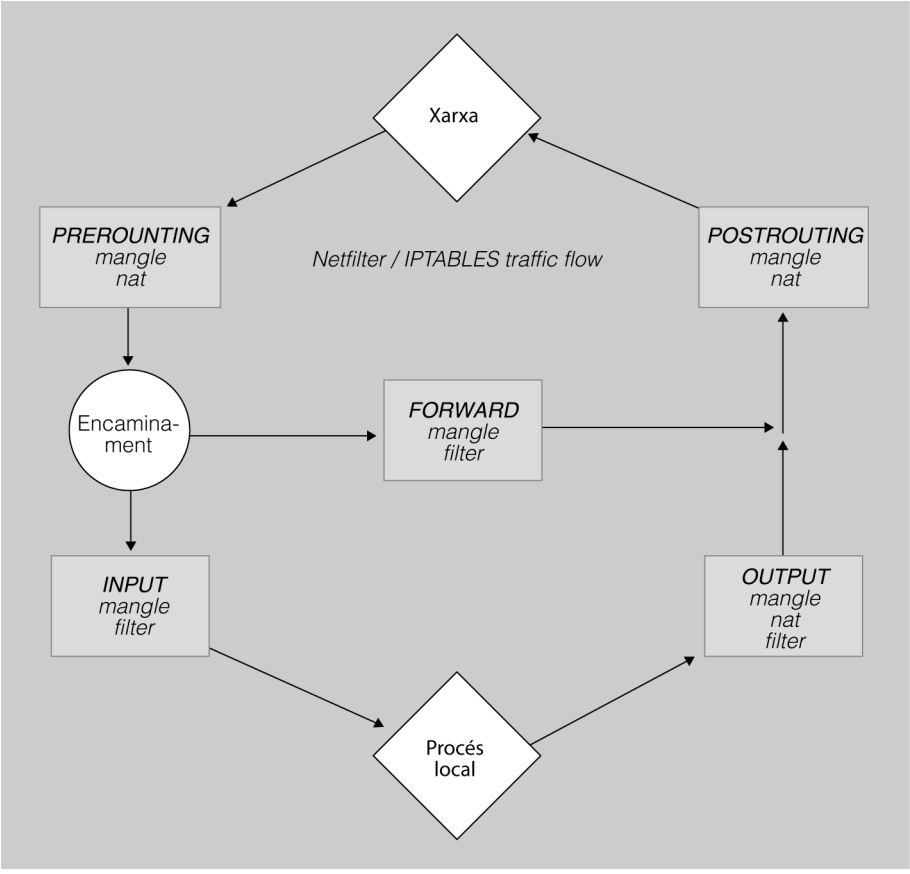
  - Cadena *POSTROUTING*. Els paquets que surten recorren la cadena un cop presa la decisió de l'encaminament. Permet fer SNAT, és a dir, alterar l'adreça de l'origen.
  - Cadena *OUTPUT*. Permet fer un DNAT limitat en als paquets generats localment.

- **Taula *MANGLE***. Aquesta taula està pensada per fer modificacions avançades en els paquets, com alterar-ne els bits de qualitat de servei. Tots els paquets passen per aquesta taula, la qual conté totes les cadenes predefinides.

  - Cadena *PREROUTING*. Tots els paquets que arriben a l'estació, abans de prendre la decisió de l'encaminament per saber si es tracta d'un paquet per a aquest *host* (cadena *INPUT*) o de reenviament (cadena *FORWARD*).
  - Cadena *INPUT*. Tots els paquets destinats a aquest sistema.
  - Cadena *FORWARD*. Tots els paquets que arriben a aquest sistema amb destinació a un altre.
  - Cadena *OUTPUT*. Tots els paquets creats en aquest sistema.
  - Cadena *POSTROUTING*. Tots els paquets que surten d'aquest sistema.

En la figura 4 es pot veure un diagrama del recorregut dels paquets per la infraestructura definida per netfilter.

Figura 4. Diagrama del recorregut dels paquets en *netfilter/iptables*



2.2.3. Fer persistents les polítiques de filtratge de paquets

L'ordre **iptables** permet administrar regles i cadenes per definir una política de filtratge de paquets amb seguiment de la connexió al nucli del Linux. Per tal d'aconseguir persistència per a aquesta política al llarg dels diferents cicles d'aturada/engegada del sistema, és necessari enregistrar la política de filtratge en un fitxer de manera que es torni a carregar en la propera arrancada.

Per a aquesta tasca, la distribució de *netfilter/iptables* inclou les eines **iptables-save** i **iptables-restore**. La primera produeix un abocament de totes les regles de filtratge del nucli, que es pot reencaminar a un fitxer. La segona pot carregar aquest fitxer en el nucli.

Normalment, **iptables-restore** es crida en els *scripts* d'engegada del sistema.

2.2.4. Treball amb taules, cadenes i regles

L'ordre **iptables** permet gestionar les cadenes de regles a netfilter. La pàgina de manual, **iptables(8)**, detalla les diferents opcions.

> Cal recordar que les cadenes formen part de les taules: *FILTER*, *NAT* i *MANGLE*. I que cada taula té una funció específica. La taula adequada per al filtratge de paquets és *FILTER*, i aquesta és justament la taula per defecte per a l'ordre **iptables**. Si es vol treballar amb una altra taula cal especificar-ne el nom mitjançant el paràmetre opcional **–t**.

Les opcions sobre les cadenes són les següents:

- Només per a les cadenes predefinides: canviar-ne la política per defecte per descartar (*DROP*) o acceptar (*ACCEPT*) els paquets que arriben al final. (`iptables -P INPUT DROP`).
- Fer una llista amb les cadenes i les seves regles (`iptables -L`).
- Definir noves cadenes d'usuari (`iptables -N novacadena`), esborrar cadenes (`iptables -X novacadena`) o renombrar-ne (`iptables -E nom_antic nom_nou`).
- Buidar totes les regles d'una cadena, o de totes les cadenes de la taula si no s'especifica la cadena (`iptables -F`).

- Reiniciar els comptadors de paquets i bytes per a totes les regles d'una cadena, o de totes les cadenes (`iptables -Z`).

En cada cadena, es poden manipular les regles:

- Afegir una nova regla al final de la cadena (–A) o bé inserir la regla en una posició determinada (–I).

- Canviar la posició d'una regla dins de la cadena (–R).

- Esborrar una regla (–D) i indicar-ne la posició o bé la descripció.

En la taula 1 teniu altres opcions generals de l'ordre *iptables*.

Taula 1. Algunes opcions de l'ordre `iptables`

Opció	Funció
`-v, --verbose`	Activa el mode prolix on es mostra més informació en les llistes.
`-n, --numeric`	Les adreces IP i els ports es mostraran en format numèric. Per defecte s'intenta mostrar el nom DNS i el nom del servei associat (en el fitxer /etc/services) al port.
`-x, --exact`	En les llistes, es mostraran les xifres dels comptadors de paquets i bytes sense utilitzar els multiplicadors K, M o G (que tenen per valor 10^3, 10^6 i 10^9 respectivament).
`--line-numbers`	Inclou la seva numeració en la llista de regles.

2.2.5. Detalls de filtratge en les regles

En cada cadena, les regles especifiquen diferents detalls que s'examinaran en els paquets per tal de veure si se'ls aplica la regla en qüestió o no. Com **`iptables`** és extensible, es poden trobar mòduls per fer multitud de comprovacions, però les més bàsiques són les següents:

1) Adreça d'origen i destinació

Les adreces es poden especificar en diferents formats:

- Amb el nom DNS, per exemple, `localhost` o `ioc.xtec.cat`
- Una adreça IP, per exemple, 213.176.177.11
- Un bloc d'adreces IP representat per una IP i la seva màscara en notació decimal separada per punts, `192.168.0.0/255.255.255.0`, o bé en notació CIDR, `192.168.0.0/24`

CIDR

En l'encaminament interdominis sense classes (CIDR), per fer un ús eficient de les adreces, s'utilitza una màscara variable.

En l'ordre **`iptables`**, per tal d'especificar l'adreça origen es farà servir **`-s`**, **`-source`** o bé **`-src`**. I per tal d'especificar la destinació es farà servir **`-d`**, **`-destinantion`** o bé **`-dst`**.

Exemple de bloqueig de paquets

Per bloquejar tots els paquets provinents de `ioc.xtec.cat` farem el següent:

```
iptables -A INPUT -s ioc.xtec.cat -j DROP
iptables -A FORWARD -s ioc.xec.cat -j DROP
```

2) Protocol

Mitjançant l'indicador **`-p`**, es pot especificar el protocol: TCP, UDP o ICMP.

3) Interfície de xarxa

Els indicadors **`-i`** o **`-in-interface`** permeten definir la interfície d'entrada, mentre que **`-o`** o **`-out-interface`** permeten especificar la interfície de sortida.

En definir regles que filtren per la interfície cal recordar que en la cadena INPUT només es podrà examinar la interfície d'entrada, i en la cadena OUTPUT només es podrà examinar la interfície de sortida. Només en la cadena FORWARD es podrà filtrar per la interfície d'entrada i sortida.

En definir la interfície es pot emprar el caràcter + com un comodí per a totes les interfícies que tenen un nom que comença de la mateixa manera.

Exemple de bloqueig de paquets que arriben per interfície Ethernet

Per bloquejar els paquets que arriben per qualsevol interfície Ethernet farem el següent:

```
iptables -A INPUT -i eth+ -j DROP
iptables -A FORWARD -i eth+ -j DROP
```

4) Fragments

En una xarxa, la MTU especifica la mida màxima d'un datagrama. Si cal transmetre un paquet que excedeix l'MTU, aleshores es fragmenta abans de la transmissió i es reassemblen els fragments un cop rebuts. Així, mitjançant la fragmentació un paquet pot ser l'origen d'un paquet més diversos fragments. En aquesta situació només el primer fragment es pot considerar un paquet amb tota la capçalera IP (i si fos el cas TCP, UDP o ICMP), i la resta de fragments tindrà una capçalera reduïda.

L'RFC 791 descriu els procediments de fragmentació, transmissió i reassemblatge en el protocol IP.

Qualsevol regla que especifiqui detalls que no apareixen en els fragments mai no actuarà sobre aquests. Per aquesta raó **`iptables`** proporciona un indicador per als fragments: **`-f`**.

Exemple de fragments cap a un host

Per descartar els fragments cap a `ioc.xtec.cat` farem el següent:

```
iptables -A OUTPUT -f -d ioc.xtec.cat -j DROP
iptables -A FORWARD -f -d ioc.xtec.cat -j DROP
```

5) La negació: !

Els indicadors que permeten especificar adreces, interfícies, fragments i protocols permeten emprar el caràcter `!` per fer una negació.

Exemple sobre la negació

Per descartar en la cadena INPUT qualsevol paquet que no vingui de l'adreça `ioc.xtec.cat` farem el següent:

```
iptables -A INPUT -s ! ioc.xtec.cat -j DROP
```

6) Eines per automatitzar l'administració de regles per a iptables

Especificar de manera manual les diferents regles d'*iptables* per aconseguir una política de seguretat adequada és un treball feixuc per al qual és necessari un bon coneixement de molts detalls de baix nivell. Per aquesta raó hi ha diferents eines que generen les regles d'una manera automatitzada a partir d'uns fitxers de configuració d'alt nivell.

Shorewall i Firestarter

Shorewall és probablement l'eina més emprada per generar de manera automàtica les regles de filtratge a partir d'uns fitxers de configuració. És molt flexible però demana a l'usuari coneixements tècnics sobre xarxes.

Firestarter és una de les solucions més senzilles per configurar el tallafoc de Linux. És una eina completament gràfica que pot fer servir qualsevol usuari.

En la Linkat2 està disponible **SuSEfirewall2**, que a partir dels valors definits a `/etc/sysconfig/SuSEfirewall2` generarà un conjunt de regles per a *iptables*. És important observar que una vegada s'han generat les regles per a *iptables* finalitza el treball de **SuSEfirewall2**, o d'altres eines similars com **Shorewall**, i comença el treball de filtratge propi del nucli del Linux.

En el fitxer de configuració de **SuSEfirewall2** es distingeixen tres zones diferents:

- **Zona externa.** La xarxa sense confiança, normalment Internet, però pot ser qualsevol xarxa.
- **Zona interna.** La xarxa que s'intenta protegir amb el tallafoc.
- **Zona desmilitaritzada (DMZ).** La xarxa on són els servidors que han de ser accessibles des de la zona externa, i també la interna. Però tot i que els servidors de la DMZ poden contestar les peticions de la zona interna, no podran iniciar connexions cap a la zona interna.

El filtratge només afectarà els paquets originats en màquines remotes, no els generats de manera local. Atès que la política per defecte consistirà a denegar qualsevol connexió que no s'hagi aprovat de manera explícita, per a totes les interfícies de xarxa amb trànsit d'entrada caldrà especificar la zona a la qual pertanyeran. Es definiran els protocols i serveis acceptats, per a cada zona.

3. Administració remota

Per diverses raons, els servidors i d'altres equips de xarxa a vegades són distribuïts al llarg de distàncies considerables. Fins i tot quan són relativament a prop, dins del mateix edifici o la mateixa planta, poden estar instal·lats en espais d'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.

Les tasques d'administració remota es poden dur a terme amb diferents mitjans, però sempre s'estarà condicionat per la xarxa que connecta l'equip a administrar amb l'estació on hi ha l'administrador.

La xarxa imposarà condicions següents:

- **Capacitat**. Si la connexió no té una amplada de banda suficient no serà pràctic treballar amb una interfície gràfica remota. En la mesura en què el retard en la xarxa augmenti, el treball interactiu serà més frustrant.

- **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 classificar en tres categories bàsiques:

- Sessió de treball en la consola
- Sessió de treball amb interfície gràfica
- Client o eina d'administració local

Cadascuna té punts forts i febles. Una sessió de treball en 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 emprar amb comoditat per administrar màquines molt distants o amb xarxes d'escassa capacitat. A més és relativament senzill automatitzar tasques mitjançant *scripts* per repetir les mateixes operacions en un conjunt de màquines.

L'administració remota amb interfície gràfica permet obrir en 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 són 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, atès que la representació gràfica es realitza 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 en cada estació que l'administrador vulgui emprar. Sovint aquesta eina només està disponible per a un sistema operatiu o per a una versió determinada d'aquest. Aquest problema s'obvia amb les interfícies web, ja que un navegador web és un programari comú que es troba en tots els equips.

En la taula 2 teniu un resum dels avantatges i inconvenients de cada un dels mètodes d'administració.

Taula 2. Avantatges i inconvenients dels mètodes d'administració

Mètode d'administració	Avantatges	Inconvenients
Treball en la línia d'ordres	Requisits mínims per a la xarxa. Flexibilitat i automatitzable.	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 a la xarxa.	Si no es tracta d'una interfície web, cal instal·lar programari.

3.1. Administració remota basada en la línia d'ordres

Les connexions remotes mitjançant la línia d'ordres són l'opció que menys capacitats exigeixen a la xarxa i, per tant, es poden fer servir fins i tot en els casos en què el canal de comunicació no té una gran amplada de banda disponible o 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.

Les eines principals per iniciar una sessió de treball remota amb una línia d'ordres són **Telnet** i **SSH**. Ambdues permeten treballar com si s'estigués al teclat i pantalla de l'estació remota, i totes dues estan basades en una arquitectura client-servidor, però les diferències que les separen són profundes. **Telnet** encara s'utilitza per raons històriques. És possible que ens trobem commutadors o servidors d'impressió als quals podem accedir mitjançant Telnet per administrar-los. Però atès que la informació es transmet com a text net, implica un greu problema de seguretat. Avui dia, el seu ús està completament desaconsellat i exigeix prendre mesures addicionals.

Telnet

El servidor Telnet acostuma a esperar les connexions dels clients en el port TCP 23. És un protocol antic, desenvolupat el 1969, que pot ser compromès amb qualsevol *packet sniffer*.

La falta de mesures de seguretat modernes impedeixen emprar Telnet de manera generalitzada, però encara és una eina molt eficaç per connectar

amb servidors que treballen amb text net (com els servidors HTTP, SMTP o Jabber) per diagnosticar problemes.

La Linkat2 en la instal·lació per defecte no inclou el servidor Telnet però sí el client que ens permet, per exemple, establir una connexió amb el servidor de correu local (en la Linkat2 s'executa el servidor SMTP Postfix). Una vegada establerta la connexió, podrem dialogar amb el servidor per sol·licitar l'entrega d'un missatge de usuari@elmeudomini.net a root@localhost.

Exemple de connexió Telnet

En l'exemple següent es troba ressaltat en negreta tot el que hem escrit en el nostre terminal. La resta és sortida generada de manera automàtica.

```
reached:~ # telnet localhost smtp
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 linkat2.qemu ESMTP Postfix
HELO elmeudomini.net
250 linkat2.qemu
MAIL FROM: usuari@elmeudomini.net
250 Ok
RCPT TO: root@localhost
250 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Subject: Aquest és un correu de prova
From: usuari@elmeudomini.net
To: root@localhost

Hola,
Emprant una connexió telnet amb el servidor SMTP (Postfix)
estem escrivint un missatge de prova per a root@localhost.

Adéu.
.
250 Ok: queued as BD700FF3E2
quit
221 Bye
Connection closed by foreign host.
```

En tancar la connexió amb el servidor SMTP, l'intèrpret d'ordres (*shell*) ens indicarà que hem rebut un nou missatge. El podrem llegir mitjançant l'ordre **`mail`**.

```
You have mail in /var/mail/root
reached:~ # mail
mailx version nail 11.25 7/29/05. Type ? for help.
"/var/mail/root": 1 message 1 new
>N 1 usuari@elmeudomini Wed Apr 2 13:35 18/664 Aquest és un
correu de prova
? 1
Message 1:
From usuari@elmeudomini.net Wed Apr 2 13:35:26 2008
X-Original-To: root@localhost
Delivered-To: root@localhost.linkat2.qemu
Subject: Aquest és un correu de prova
From: usuari@elmeudomini.net
To: root@localhost.linkat2.qemu
Date: Wed, 2 Apr 2008 13:33:18 +0200 (CEST)
```

```
Hola,
Emprant una connexió telnet amb el servidor SMTP (Postfix)
estem escrivint un missatge de prova per a root@localhost.

Adéu.

?
```

3.1.1. Introducció a SSH

SSH, *secure shell*, és un protocol de xarxa que permet l'intercanvi d'informació de manera segura. Utilitza xifratge i criptografia de clau pública per realitzar l'autenticació de l'estació remota.

Una de les funcions més emprades és iniciar una sessió remota per executar ordres. Però les seves capacitats són més àmplies, i també permet el següent:

- Transmetre fitxers de manera segura.
- Fer túnels per assegurar qualsevol servei que no es transmeti xifrat (HTTP, SMTP, VNC...) o per travessar tallafocs que estiguin bloquejant el protocol.
- Reenviar automàticament sessions X11.
- Fins i tot, en fer servir SSHFS, actuar com a sistema de fitxers de xarxa.

Reenviament de sessions X11

El reenviament de sessions X11 permet executar programari en un equip remot visualitzant la interfície gràfica en l'equip local.

SSHFS

L'SSH *file system* (SSHFS) és un sistema de fitxers en xarxa que fa servir l'SSH per comunicar els diferents equips.

Les seves possibilitats es poden combinar de moltes maneres diferents. Ateses les seves capacitats criptogràfiques, l'SSH és una eina fonamental per a l'administrador de xarxa.

Breu història de l'SSH

El 1995 Tatu Ylönen va dissenyar la primera versió del protocol SSH, coneguda com a SSH-1, per reemplaçar les ordres insegures telnet i rlogin.

L'any següent es va revisar el protocol, i es va desenvolupar la versió SSH-2 per afegir millores de seguretat i noves funcionalitats.

Tot i que les primeres eines eren lliures, l'empresa que el mateix Tatu va fundar per desenvolupar l'SSH el va fer evolucionar cada vegada més cap al programari privatiu.

El projecte OpenBSD va desenvolupar la implementació OpenSSH com a programari totalment lliure. Aquesta és la implementació més estesa en l'actualitat.

3.1.2. Alguns conceptes criptogràfics

La criptografia ens aporta tècniques per xifrar i desxifrar informació. De tal manera que es puguin transmetre, amb seguretat, missatges per un canal que no és de confiança. L'objectiu evident és assegurar que el missatge transmès romandrà secret. Però en les comunicacions electròniques so-

vint també s'utilitza per autenticar que l'emissor i el destinatari del missatge són qui pensem, i per garantir que el propi missatge no ha estat alterat durant la transmissió.

Hi ha diferents tècniques criptogràfiques, però dues de les més emprades són les següents:

- Criptografia simètrica o de clau compartida
- Criptografia asimètrica o de clau pública

En emprar un algorisme de xifratge de clau compartida s'utilitza la mateixa clau per xifrar i desxifrar. Aquesta característica és el seu inconvenient principal, ja que si es vol canviar la clau es necessitarà un mitjà de comunicació segur per fer-la arribar de l'emissor al receptor. L'avantatge principal de la criptografia simètrica és la velocitat. Alguns algorismes de xifratge simètrics són 3DES, AES, Blowfish i IDEA.

La criptografia asimètrica o de clau pública evita el problema de l'intercanvi de claus. En aquest cas, tant l'emissor com el receptor tindran una parella de claus diferents. En cada parella de claus n'hi ha una de pública i una de privada. La clau privada mai no viatja pel canal de comunicació, i cal garantir-ne la protecció. La clau pública, com el seu nom ho indica, es pot oferir a tothom que la demani, pot viatjar per mitjans insegurs o, fins i tot, es pot publicar.

Cada parella de claus està estretament relacionada:

- Els missatges que es xifren emprant una clau pública només es poden desxifrar mitjançant la clau privada associada.

- Les claus privades permeten signar un missatge perquè el receptor verifiqui la identitat de l'emissor. La comprovació de la signatura es farà mitjançant la clau pública associada.

En una comunicació típica tindrem el següent:

1) L'emissor farà servir la pròpia clau privada per signar el missatge.
2) L'emissor farà servir la clau pública del receptor per xifrar el missatge.
3) Es transmet el missatge xifrat i signat.
4) El receptor desxifrarà el missatge fent servir la pròpia clau privada.
5) El receptor comprovarà la signatura del missatge fent servir la clau pública de l'emissor.

En aquest procediment mai es transmeten les claus privades. L'intercanvi de les claus públiques és senzill, ja que no cal que es transmetin utilitzant un canal segur. Es garanteix que el missatge no pot ser compromès en la

transmissió i tant l'emissor com el receptor comproven l'autenticitat de l'altra entitat.

L'inconvenient principal de la criptografia de clau pública és el consum de recursos. És més costosa en temps de xifratge/desxifratge, i el missatge xifrat ocupa més que el text net.

En la taula 3 teniu un resum d'avantatges i inconvenients de cada tècnica criptogràfica.

Taula 3. Avantatges i inconvenients de les tècniques criptogràfiques

Tècnica criptogràfica	Avantatges	Inconvenients
Criptografia clau compartida	Consumeix pocs recursos. Es xifra/desxifra a gran velocitat.	L'intercanvi de claus necessita un canal segur.
Criptografia clau pública	No necessita un canal segur per intercanviar les claus.	Consumeix recursos. El procés de xifratge/desxifratge és lent.
Criptografia híbrida	Es xifra/desxifra a gran velocitat. No necessita un canal segur per intercanviar les claus.	

La **criptografia híbrida** combina el sistema de clau compartida i el de clau pública per obtenir tots els avantatges sense els inconvenients. En aquest sistema només s'utilitza la criptografia de clau pública per transmetre una clau compartida, normalment generada de manera dinàmica per aquesta comunicació, al receptor. Una vegada s'ha comunicat la clau compartida a les dues estacions, s'utilitza xifratge convencional per xifrar el gros del missatge.

Els sistemes basats en criptografia híbrida són tan forts com ho siguin els sistemes de clau compartida i clau pública que utilitzin.

Criptografia híbrida en la realitat

Moltes eines com l'SSH, GnuPG o PGP combinen la criptografia de clau compartida i pública. Són criptosistemes híbrids.

3.1.3. Funcions bàsiques de l'SSH

La funció més comuna de l'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 usuari@host
```

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

a la sessió de treball remota per escriure ordres. Per finalitzar, escriurem `exit`.

El client d'**OpenSSH** suporta diferents opcions que es detallen en la seva pàgina de manual. Algunes de les més freqüents les podeu veure en la taula 4.

Taula 4. Opcions de l'OpenSSH

Opció	Funció
`-1`	Força l'ús de la versió 1 del protocol. Només es recomana emprar l'SSH-1 per connectar amb servidors antics que no suporten l'SSH-2.
`-2`	Força l'ús de la versió 2 del protocol, SSH-2.
`-4`	Força l'ús de l'adreçament IPv4.
`-6`	Força l'ús de l'adreçament IPv6.
`-C`	Activa la compressió, gzip, a 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. Per defecte, el servidor SSH s'executa en el 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 un altre **`-q`** no imprimeix ni els errors.
`-X`	Activa la retransmissió X11, perquè els programes gràfics llançats a l'estació remota obrin la seva interfície gràfica al servidor X local.
`-x`	Desactiva la retransmissió X11.

Algunes vegades només es vol executar una ordre en l'estació remota, no obrir un intèrpret d'ordres per treballar. En aquest cas, és possible indicar l'ordre a executar a la pròpia crida de l'SSH.

Per exemple, per veure el final del fitxer de registre /var/log/messages a l'estació 192.168.100.10 faríem el següent:

```
ssh usuari@192.168.100.10 cat /var/log/messages
```

Transferir fitxers entre diferents 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 de l'SSH.

L'ordre **`scp`** es pot veure com una versió estesa de l'ordre **`cp`** que permet copiar fitxers, fins i tot, entre diferents màquines. De fet és el substitut de l'SSH per l'antiga, i no segura, ordre **`rcp`**. En fer una còpia mitjançant **`scp`**, es pot escollir qualsevol combinació de fitxer local o remot tant per l'origen com per la destinació. En la taula 5 teniu diversos exemples d'ús de l'ordre **`scp`**.

Taula 5. Exemples d'ús de l'ordre **scp**

Exemple	Ordre
Copiar fitxer local en estació remota	`scp fitxer usuari@192.168.100.10:/home/usuari`
Copiar fitxer remot en estació local	`scp usuari@192.168.100.10:/home/usuari/fitxer`
Copiar un fitxer d'un equip remot en un altre equip remot	`scp usuari@192.168.100.10:/home/usuari/fitxer usuari@192.168.100.11:/home/usuari`

La pàgina de manual d'**`scp`** ens mostrarà els seus paràmetres, la majoria ja coneguts de l'ordre **`ssh`**. Alguns propis d'**`scp`** particularment útils són **`-l limit`**, per establir un límit a l'amplada de banda en kilobits per segon; **`-p`**, per preservar el temps de modificació, accés i els permisos dels fitxers copiats, i **`-r`** per fer una còpia recursiva per als directoris.

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

En la taula 6 teniu un conjunt de les ordres que pot interpretar **`sftp`**.

Taula 6. Funció de les principals ordres d'**sftp**

Ordre	Funció
`bye, exit, quit`	Finalitzar la sessió
`cd camí`	Canviar el directori remot
`chgrp, chown, chmod`	Canviar el grup, propietari o 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`	Llistes, creació de directoris, consulta del *path*, canvi de nom, esborrament de fitxers i directoris, i creació d'enllaços simbòlics al sistema de fitxers remot
`lcd, lls, lmkdir, lpwd`	Versió de les ordres cd, ls, mkdir i pwd per treballar en el sistema de fitxers local

L'SSH en l'escriptori

Els escriptoris **Gnome** i **KDE** també proporcionen un accés senzill a servidors SSH per 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), i a KDE es pot fer servir el IOSlave FISH per a la mateixa funció. Només cal especificar *fish://usuari@host* en la barra d'adreces del navegador d'arxius **Konqueror** o **Dolphin**.

Redirecció de ports TCP, túnels amb SSH

L'SSH és un reemplaçament segur per a les eines insegures: telnet, rcp i ftp. Però, què hi ha de la resta de serveis que utilitzen la xarxa sense fer ús de tècniques criptogràfiques? Fent servir l'SSH es poden establir túnels xifrats pels quals es pot transmetre qualsevol protocol que faci servir TCP. Així és possible, per exemple, emprar un túnel SSH per al següent:

- Baixar el correu mitjançant POP3, i enviar-ho fent servir SMTP.
- Accedir a un servidor web mitjançant HTTP.
- Accedir a un sistema d'arxius remot mitjançant NFS.

SSH, túnels i VPN

L'SSH permet establir túnels pels quals es pot transmetre qualsevol connexió TCP. D'aquesta manera és possible emprar, amb seguretat, protocols insegurs en xarxes que no són de confiança.

Aquesta és la funció de les VPN (*virtual private network*, o xarxa privada virtual). Si es vol fer servir una VPN per transportar més d'un servei, o per transportar trànsit que no sigui TCP, caldrà examinar d'altres solucions com **OpenVPN**.

Els protocols de 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.

Exemple de túnel SSH

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 el següent:

```
ssh usuari@192.168.10.100 -L 10025:192.168.10.100:25
```

Una vegada establerta la sessió SSH, i mentre es mantingui, hi haurà un túnel xifrat entre el port TCP local 10025 i el 25 de l'estació 192.168.10.100. D'aquesta manera, per fer una connexió segura amb el servidor SMTP de l'estació remota, hauríem de connectar amb el port local 10025. L'SSH s'encarregarà de fer el transport de manera segura entre totes dues estacions.

3.1.4. SSH: mecanismes d'autenticació

L'autenticació és l'acció de comprovar la identitat del remitent en una comunicació. El servidor SSH, en rebre una nova sol·licitud de connexió, necessita autenticar l'origen d'aquesta connexió com a pas previ per autoritzar o denegar la connexió.

El servidor SSH pot emprar quatre mecanismes diferents d'autenticació. Per ordre de preferència, aquests mecanismes són els següents:

1) Autenticació basada en el *host*
2) Autenticació de clau pública
3) Autenticació desafiament–resposta
4) Autenticació basada en la contrasenya

Abans de l'autenticació, tant a SSH-1 com a SSH-2, s'establirà una comunicació xifrada entre el client i el servidor. Per establir aquesta connexió xifrada es fan servir tècniques criptogràfiques de clau pública. Una vegada acordada una clau comuna s'utilitzarà un xifratge simètric per a la resta de la sessió. El procediment per negociar la clau de sessió és diferent a SSH-1 i SSH-2, els detalls s'expliquen en la pàgina de manual de **sshd(8)**. Sempre que sigui possible és recomanable emprar només SSH-2 on s'utilitza Diffie-Hellman per acordar la clau de sessió, a més de comptar amb mecanismes criptogràfics per garantir la integritat de les dades a tota la sessió.

Protocol Diffie-Hellman

El protocol Diffie-Hellman permet la negociació segura de claus entre dues entitats que no han tingut comunicació prèvia, mitjançant un canal insegur i de manera anònima (no autenticada).

Autenticació basada en el host

L'autenticació basada en el *host* és la tècnica més segura: impedeix els atacs basats en la suplantació d'adreces IP, de DNS i de la taula de rutes.

L'SSH genera una parella de claus, pública i privada, durant la seva instal·lació en cada equip. Aquestes claus identifiquen l'estació on s'executa l'SSH, no els seus usuaris. Són les anomenades *claus de host*, i llevat que el compte de *root* sigui compromès (és a dir, que algú s'hagués convertit en *root* de manera fraudulenta) no és necessari canviar-les.

Les claus de *host* es troben en el directori **/etc/ssh** i el seu nom és **ssh_host_dsa_key**, **ssh_host_dsa_key.pub**, **ssh_host_rsa_key** i **ssh_host_rsa_key.pub** per a SSH-2, i **ssh_host_key/ssh_host_key.pub** per a la versió SSH-1.

La versió SSH-2 pot emprar els algorismes RSA i DSA, i la versió SSH-1 només pot fer servir RSA.

RSA i DSA

RSA i DSA són dos algorismes que fan servir el xifratge de clau pública.

RSA són les inicials dels investigadors que van desenvolupar l'algorisme: Ron Rivest, Adi Shamir i Leonard Adleman.

DSA són les sigles de *digital signature algorithm*, és a dir, algorisme de signatura digital.

Perquè l'autenticació basada en el *host* permeti a l'usuari iniciar sessió mitjançant SSH, cal el següent:

1) Que el servidor pugui verificar la clau de *host* del client. Aquesta verificació es produeix si el servidor registra en el fitxer **known_hosts** (fitxer que resideix a **/etc/ssh/ssh_known_hosts** per a tot el sistema o bé a **~/.ssh/known_hosts** per a cada usuari) la clau de *host* pública del client.

Fitxer known_hosts

Quan un client SSH estableix una connexió per primera vegada amb un *host*, enregistra en el fitxer **known_hosts** la clau pública de l'estació. La propera vegada que es realitzi una connexió es comprovarà aquesta clau pública.

2) Que la màquina que inicia la connexió estigui en una llista en el fitxer **/etc/ssh/shosts.equiv** (o la seva versió insegura **/etc/hosts.equiv**), i el nom d'usuari en l'estació client i en el servidor sigui el mateix. O bé que en el servidor, en el directori de l'usuari que intenta iniciar sessió hi hagi el fitxer **.shosts** (o la seva versió insegura **.rhosts**) posant en una llista la màquina client i el nom d'usuari en aquella estació.

Autenticació de clau pública

L'autenticació de clau pública és un sistema que es basa, com el nom ho indica, en tècniques criptogràfiques de clau pública on el que es xifra mitjançant la clau pública només es pot desxifrar mitjançant la clau privada associada. I on la clau privada es pot emprar per signar els missatges perquè el receptor pugui autenticar l'emissor.

Passphrase

La pàgina de manual **ssh-keygen(1)** ens recomana emprar una *passphrase* per bloquejar les claus. Aquesta frase és com una contrasenya però més llarga. De fet, la longitud recomanada està entre els deu i trenta caràcters, tot i que pot ser més llarga.

Es pot pensar que introduir una *passphrase* tan llarga cada vegada que es volen fer servir les claus és incòmode. Per això, l'eina **ssh-agent** pot racionalitzar aquesta funció fent ús de l'SSH amb un mètode d'autenticació de clau pública més còmode.

Cal que l'usuari que vol iniciar una sessió remota en el servidor construeixi una parella de claus, normalment utilitzant l'eina **`ssh-keygen`**. Aquestes claus s'emmagatzemaran en el directori **~/.ssh/** (en els fitxers **id_rsa/id_rsa.pub**, **id_dsa/id_dsa.pub** per a SSH-2, o bé **identity/identity.pub** per a SSH-1).

En construir la parella de claus, l'eina **`ssh-keygen`** ens permetrà definir si volem emprar una clau RSA o DSA. També serà possible protegir les claus

mitjançant una contrasenya que serà necessari escriure per desbloquejar la clau i poder-la emprar. De fet, aquesta és una opció molt recomanable, i només es deixaran claus sense protegir si es tracta de les claus de *host* o bé es vol iniciar la sessió de manera automatitzada (mitjançant *scripts*) sense que sigui necessària la intervenció de l'usuari per desbloquejar les claus.

Una vegada s'han generat les claus serà necessari transportar la clau pública a l'estació on es vol iniciar sessió, i afegir-la al fitxer ~/**.ssh/authorized_keys** de l'usuari corresponent.

A partir d'aquest moment, l'usuari podrà iniciar una sessió SSH sense haver d'especificar la seva contrasenya. Però si ha especificat una frase de pas per bloquejar les claus, encara serà necessari introduir-la per poder-les fer servir.

3.1.5. Configuració pel client SSH

Configuració client SSH

La configuració del client d'OpenSSH és força flexible. Es pot consultar la documentació oficial en la màquina de manual **ssh_config(5)**.

El client d'OpenSSH permet establir la seva configuració d'una manera prou flexible perquè l'administrador pugui definir una configuració general per a tot el sistema, cada usuari pugui modificar els paràmetres adequats per a les seves connexions i, de manera concreta, pugui especificar determinades opcions per a cada connexió individual.

El client d'OpenSSH farà servir el següent:

1) Les opcions indicades en la línia d'ordres.
2) Els valors especificats en el fitxer de configuració de l'usuari: ~/**.ssh**/**ssh_config**.
3) Els valors especificats en la configuració per a tot el sistema: /**etc**/**ssh**/**ssh_config**.

Per a cada paràmetre, el client farà servir el primer valor trobat. És a dir, si s'especifica un paràmetre en 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. En la taula teniu algunes de les opcions més bàsiques per al client d'OpenSSH.

Taula 7. Opcions del client d'OpenSSH

Opció	Funció
`Host <patró>`	Permet especificar opcions que només s'aplicaran a les connexions amb el *host* indicat. El *host* s'indica mitjançant patrons amb els caràcters * i ?.
`CheckHostIP <yes|no>`	El seu valor per defecte és **yes**. Si l'opció està activa, 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 de xifratge per a les connexions SSH-1 i la precedència d'algorismes a emprar en les connexions SSH-2.

Opció	Funció
`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, per defecte 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 realitzar la connexió a l'estació remota.
`SendEnv <variables>`	Permet enviar el valor de les variables d'entorn especificades a l'estació remota.

3.1.6. Configuració del servidor SSH

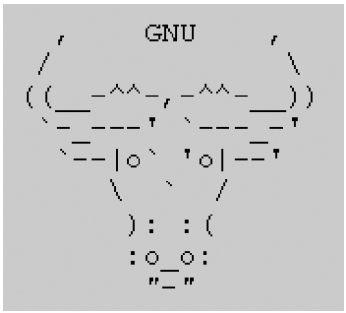
La funció del servidor SSH és esperar les connexions dels clients (normalment al port TCP 22), fer la seva autenticació i, si tot ha anat bé, obrir una sessió de treball, executar una ordre o bé reencaminar ports.

Configuració del servidor SSH

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

Cada vegada que es rep un intent de connexió des d'un client, en primer lloc es realitzen totes les comprovacions i inicialitzacions criptogràfiques per garantir la seguretat. Després es tracta d'autenticar l'usuari i, finalment, se segueixen una sèrie de passos que tenen molt en comú amb qualsevol altra manera d'iniciar la sessió en l'equip:

1) Si es tracta d'una sessió de treball en una **tty**, s'imprimeix la data i l'hora de l'última connexió. Es registra l'instant actual per mostrar-ho en la propera connexió i es mostra el contingut del fitxer **/etc/motd** si existeix.



Es comprova si el fitxer **/etc/nologin** existeix, i en cas afirmatiu només deixarà connectar l'usuari primari o *root*. Tampoc no permetrà que es connectin els usuaris que tenen un compte bloquejat.

2) El procés servidor passarà a execució amb privilegis d'usuari normal.

3) Si l'usuari pot definir el seu entorn, s'emprarà el fitxer **~/.ssh/environment** i s'executarà la inicialització definida a **~/.ssh/rc** o **/etc/ssh/sshrc**.

/etc/motd

Si existeix, el contingut del fitxer /etc/motd es mostra en la consola cada vegada que un usuari inicia una sessió.

La seva funció és informativa, i alguns administradors aprofiten per explicar-hi les regles d'ús del sistema informàtic o indicar la seva adreça de contacte. És molt freqüent emprar ASCII Art per decorar-lo.

4) Finalment, s'executarà un intèrpret d'ordres (*shell*) per a l'usuari o la seva ordre.

Malgrat que per al funcionament del servidor d'OpenSSH intervenen diferents fitxers, el fitxer 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. En la taula 8 teniu algunes de les paraules reservades més bàsiques que podeu trobar en el fitxer **/etc/ssh/sshd_config**.

Taula 8. Paraules reservades al fitxer /etc/ssh/sshd_config

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 *hosts* origen s'acceptarà la connexió. Per exemple: `AllowUsers usuari1@192* usuari2.`
Banner <fitxer>	Envia el contingut del fitxer indicat al client abans de realitzar l'autenticació.
Compression <yes|delayed|no>	Especifica si es farà servir la compressió. El valor `delayed`, l'opció per defecte, indica que només es farà servir la compressió un cop realitzada l'autenticació de l'usuari.
DenyGroups	Permet especificar una llista de grups, separats per espais, als quals no es permetrà iniciar sessió. Cal especificar els grups mitjançant el seu nom emprant els patrons ? i * de manera opcional.
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. Per defecte s'utilitza el port 22 de qualsevol adreça local. És possible especificar múltiples vegades aquestes opcions, però convé que `Port` sempre estigui abans que `ListenAddress`.
LoginGraceTime	Màxim període de temps per realitzar l'autenticació. El valor 0 expressa que no hi ha límit.
MaxAuthTries	Màxim nombre d'intents d'autenticació que es poden realitzar.
MaxStartups	Màxim nombre de connexions simultànies que encara no han completat la seva autenticació.
PasswordsAuthentication <yes|no>	Indica si s'accepta l'autenticació mitjançant contrasenya (*password*).
PermitEmptyPasswords <no|yes>	Si s'utilitza l'autenticació mitjançant *password*, especifica si el servidor permet la connexió a comptes que tenen un *password* buit.
PermitRootLogin <yes|without-password|forced-commands-only|no>	Especifica si s'accepta la connexió de *root* mitjançant SSH. El valor `without-password` indica que *root* no podrà fer servir l'autenticació basada en *password*, 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 *backups*).
Protocol	Especifica quins protocols es podran fer servir en les connexions dels clients (1, 2 o tots dos). És important recordar que el protocol SSH-2 és força més segur que l'SSH-1.
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 finestra al servidor X del client.

3.2. 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 mostren la seva interfície gràfica a un terminal local.

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 a diferents equips.

Hi ha diferents eines i protocols per dur a terme l'administració remota mitjançant una interfície gràfica. En tots els casos, la seguretat ha de ser un punt fonamental a considerar, ja que per la pròpia funció de l'aplicació estarem transportant per la xarxa tota l'entrada/sortida amb l'equip remot.

Algunes de les possibilitats per realitzar l'administració mitjançant una interfície gràfica són les següents:

- La transparència de xarxa del sistema X Window
- VNC
- FreeNX
- RDP

3.2.1. La transparència de xarxa del sistema 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 majoria de versions de l'Unix per implementar la interfície gràfica. Val a dir que l'**X Window** no defineix com ha de ser aquesta interfície, només permet implementar-la. Escriptoris tan coneguts en el GNU/Linux com **Gnome**, **KDE** o **Xfce** utilitzen el sistema **X Window** malgrat que són escriptoris diferents amb les seves particularitats.

Font: Viquipèdia

X.Org

L'X.Org és tant la fundació que gestiona el desenvolupament del sistema X Window com la implementació de referència d'aquest. 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 a gairebé totes les distribucions del GNU/Linux i d'altres sistemes operatius.

Entre les característiques pròpies del sistema X Window trobem que és independent del sistema operatiu emprat, només es tracta d'una capa d'aplicació, i que des de bon començament va ser pensat per treballar en xarxa. La transparència de xarxa permet separar l'estació que representa la interfície gràfica de l'estació on s'executa l'aplicació, tot i que evidentment poden ser la mateixa estació.

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 al servidor X s'anomenen *clients*. Així, el servidor X s'executarà en l'estació local de l'usuari perquè aquest pugui interaccionar amb

la interfície gràfica, i 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 –el servidor proporciona un servei a l'aplicació client– sovint despista els usuaris novells.

La comunicació entre el servidor X i el client X es realitza sense cap tipus de xifratge, 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 VPN.

La versió actualment en ús del protocol és la v11 des de 1987. Per aquesta raó s'acostumen a anomenar **X11** els servidors i clients del sistema X Window.

Bàsicament, un servidor X11 representa un terminal gràfic amb el propi teclat i ratolí. Perquè els clients puguin emprar aquest terminal és necessari realitzar dues accions:

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

Configurar un servidor X11 de manera relaxada 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, és necessari establir qui es pot connectar amb un servidor X11. Sense entrar en gaires detalls, per a aquesta funció es poden fer servir dos mètodes: **Xhost** i **Xauth**. El primer mètode permet especificar el permís per a diferents equips en indicar la seva adreça IP o el seu nom DNS. Si es concedeix l'accés a un equip determinat, qualsevol aplicació d'aquell *host* (independentment del seu usuari) podrà accedir al servidor X11. El segon mètode està basat en una clau anomenada *magic cookie*.

Xhost i Xauth

Per configurar un servidor X11 que permeti connexions dels clients, es poden emprar les ordres **xhost** i **xauth**.

La primera comporta riscos de seguretat, i la segona és pesada d'emprar. Per això normalment s'utilitza l'SSH per reenviar el trànsit X11.

La variable d'entorn **DISPLAY** permet conèixer a una aplicació quin és el servidor X Window que ha d'emprar per obrir les seves finestres. Aquesta variable d'entorn utilitza una cadena amb la sintaxi següent: '**host:display.screen**' (taula 9).

Taula 9. Significat dels paràmetres host, display i screen

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, i en aquest cas es fa servir la pantalla **.0**

Exemples de la variable DISPLAY

```
export DISPLAY=:0
export DISPLAY=192.168.0.50:2
export DISPLAY=192.168.0.50:2.1
```

Iniciació d'un client X mitjançant SSH

Si el nostre equip està executant un servidor X, podrem establir una sessió SSH en un equip remot per executar una aplicació de manera que la seva interfície es representi en el nostre servidor X.

Servidors X

Si fem servir el GNU/Linux, la nostra distribució empaquetarà la versió estable d'un servidor X, probablement X.Org.

Si estem en altres sistemes operatius i volem emprar un servidor X Window, caldrà que n'instal·lem un. Afortunadament hi ha opcions lliures com Cygwin/X, Xming o XDarwin.

Aquesta és probablement la manera més senzilla i segura d'emprar la transparència de xarxa del sistema X Window. En aquest cas, l'SSH s'encarrega del següent:

1) Tots els procediments criptogràfics necessaris per establir un túnel entre tots dos equips i realitzar l'autenticació de l'usuari remot.

2) Establir el valor adequat a la variable d'entorn DISPLAY de l'equip remot per tal de 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 amb l'equip remot).

3) Transportar tot el trànsit X11 protegit dins del túnel SSH.

4) Permetre l'ús del servidor X11 local per les aplicacions de l'estació remota. Només per l'usuari propietari de la sessió SSH, no per la resta de processos dels altres usuaris.

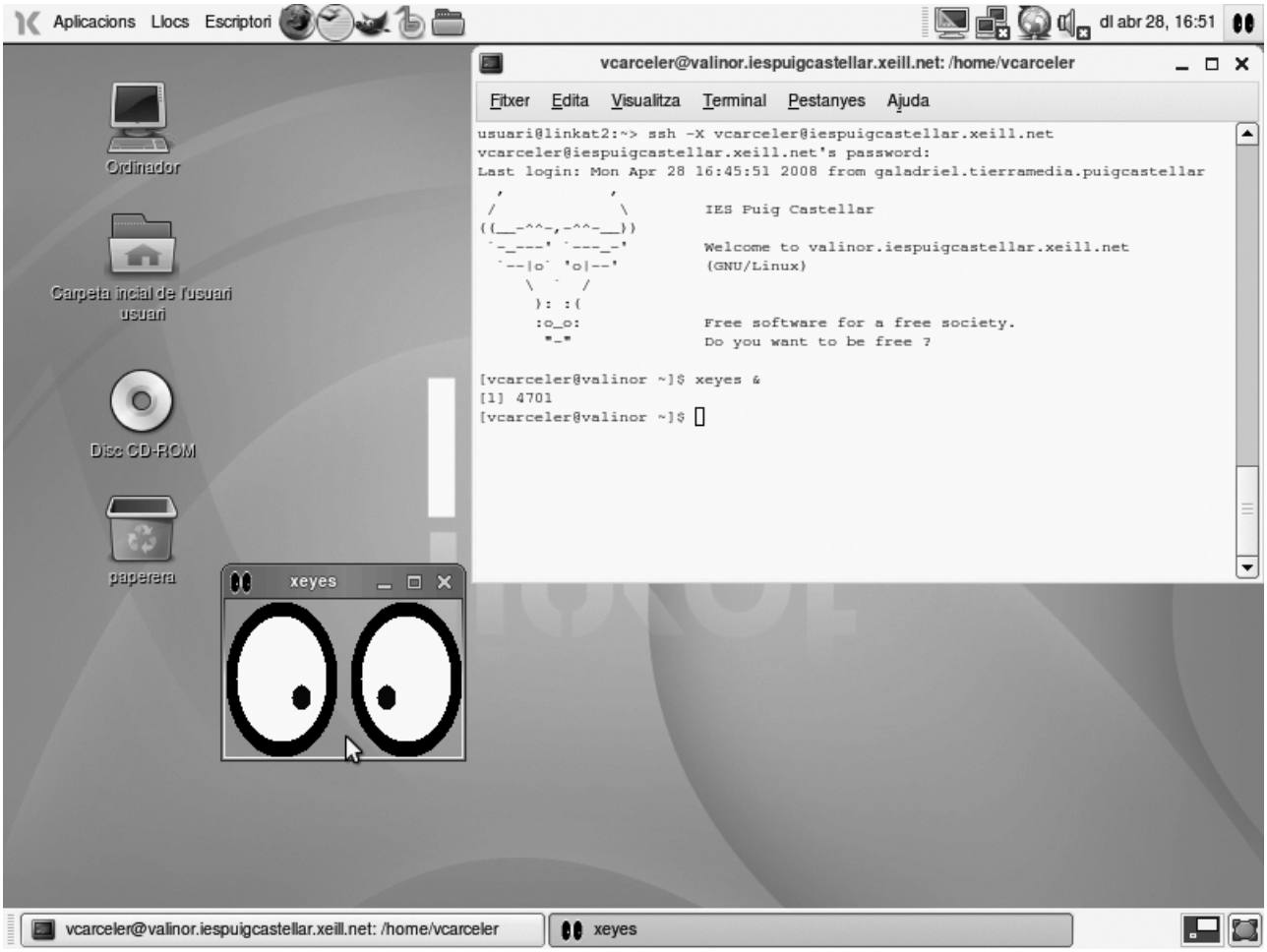
En la figura 5 es mostra una captura de pantalla en la qual s'ha emprat l'SSH per iniciar una sessió de treball en una estació remota en què s'ha executat el programa **Xeyes**. El programa **Xeyes** és molt simple, només obre una finestra en la qual apareixen dos ulls que miren cap a la posició del ratolí. En aquest cas és important remarcar que en fer la connexió SSH s'ha indicat el paràmetre **–X** per tal d'activar el reenviament de trànsit X11.

És possible configurar l'SSH perquè per defecte es realitzi el reenviament de trànsit X11 mitjançant el paràmetre de configuració **ForwardX11**. També és possible denegar de manera explícita mitjançant l'opció **`-x`** el reenviament de trànsit X11.

Els curiosos que examinin el valor de la variable d'entorn DISPLAY dins d'una sessió SSH comprovaran que, aparentment, apunta a un servidor X11 local, per exemple el `:10`, en comptes d'apuntar a l'adreça IP origen de la sessió SSH. La solució a aquest misteri és que, un cop iniciada la sessió SSH a l'equip remot es llança un nou servidor X11, en aquest cas el número 10. Aquest servidor X11, que fa la funció de *proxy*, acceptarà

connexions de les aplicacions client que s'executen en el mateix ordinador per a l'usuari corresponent i reenviarà totes les dades, mitjançant el túnel SSH, fins a l'ordinador origen de la sessió SSH on finalment s'utilitzarà el servidor X11 que veu l'usuari final.

Figura 5. Execució de **Xeyes** en una estació remota



La combinació de la transparència de xarxa del sistema X Window amb les característiques de seguretat de l'SSH implica una combinació de flexibilitat, seguretat i facilitat d'ús difícilment igualable per altres tècniques.

XDMCP per accedir a escriptoris remots

Normalment, en arrencar un ordinador amb el GNU/Linux apareix una pantalla gràfica que demana el nom d'usuari i contrasenya per iniciar la sessió. Aquest programa, que en cas d'autenticar l'usuari carrega tot l'escriptori, és un *display manager*. Hi ha diferents implementacions de les quals les més usuals són **`gdm`** per a l'escriptori Gnome, **`kdm`** per a escriptoris KDE, i **`xdm`**, el més bàsic per llançar les X.

Les X

Sovint s'abreuja tot el sistema X Window o bé alguna de les seves parts, com el servidor X, indicant les X.

En la majoria dels casos, el *display manager* gestiona un servidor X11 que s'executa de manera local. Però gràcies al protocol **XDMCP** també pot gestionar servidors X11 remots. En aquest cas, un equip pot mostrar la

benvinguda (demanant l'usuari i contrasenya) a un equip remot i, després de realitzar una autenticació positiva, mostrar a l'usuari un escriptori remot, i no solament la finestra d'una aplicació.

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

3.2.2. *Virtual network computing* (VNC)

Virtual network computing (VNC) és un sistema per veure, i si escau també controlar, un escriptori remot. Originalment va ser desenvolupat per Olivetti i hi ha implementacions lliures de les eines que permeten treballar amb VNC per a diferents sistemes operatius. El protocol de xarxa que es fa servir, **RFB** (*remote frame buffer protocol*), també és lliure.

Malgrat que les funcions inicials del protocol RFB es limitaven a transmetre els esdeveniments desencadenats per l'usuari en el teclat i ratolí local cap a l'estació remota, i a enviar en direcció contrària el contingut gràfic de la pantalla, les darreres versions s'han estès per suportar més funcionalitats, com per exemple la transferència de fitxers.

> La memòria d'imatge o ***frame buffer*** és l'àrea de memòria que emmagatzema les dades que defineixen la imatge gràfica que apareix en la pantalla. Normalment, totes les pantalles actuals es basen 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 del color (la quantitat de bits necessaris per codificar el color, i potser transparència, de cada píxel).

Com que el VNC es basa en una memòria d'imatge, pot treballar amb qualsevol sistema de finestres com X Window, Windows, o altres. En aquest sistema sempre s'anomena *client* l'estació en què l'usuari està assegut, l'estació local, i *servidor* l'equip que defineix el contingut de la pantalla (equip remot). Aquesta nomenclatura és inversa a l'emprada pel sistema X Window.

Tot el sistema està pensat per treballar amb clients lleugers de manera que el client es pugui implementar sense gaires recursos. De fet, tot el protocol i el funcionament del sistema intenta ser molt senzill. 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 codificacions, 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 nou escriptori virtual. Els sistemes Windows no permeten la creació de nous escriptoris virtuals, només permeten exportar en xarxa l'únic escriptori disponible. En el moment d'exportar un escriptori es poden definir dues contrasenyes diferents, una per als usuaris que podran controlar manera remota aquest escriptori i una altra per als usuaris que només podran veure, però no interaccionar, amb el teclat i el ratolí.

El procés per fer servir VNC és molt simple:

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

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

Cal remarcar que l'RFB no és un protocol de seguretat. En fer servir VNC en una xarxa que no sigui de confiança caldrà combinar el seu ús amb l'SSH o una altra tecnologia per crear una VPN.

La majoria de servidors VNC inclouen un petit servidor web que ofereix la baixada d'una miniaplicació (*applet*) de Java que es pot fer servir com a client VNC. D'aquesta manera, en l'ordinador local només cal un navegador amb suport Java per accedir a l'escriptori remot.

VNC, moltes eines i diferents implementacions

Si el desenvolupament original de VNC té algun hereu oficial, aquest 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 diferents llicències, l'edició Free sota la GPL i la resta amb llicències privatives.

Una altra implementació força estesa és TightVNC, programari lliure 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 es troben noves codificacions que redueixen la càrrega de la xarxa i la possibilitat de transmetre fitxers. En la taula 10 teniu un resum de les eines que trobarem a VNC.

Taula 10. Eines de VNC

Eina	Funció
vncviewer	Client VNC, permet representar l'escriptori remot.
vncserver	Exporta un escriptori. Aquesta eina crida de manera automàtica Xvnc. S'utilitzarà per llançar i per aturar, mitjançant el paràmetre –kill, els servidors VNC.
Xvnc	És un servidor X Window virtual que accepta connexions de les aplicacions X11 clients (que hi obren finestres). No utilitza la targeta de vídeo per representar el seu contingut. Actua com a servidor VNC per transmetre aquestes dades als clients. Normalment, l'usuari no utilitza directament Xvnc, sinó vncserver.
vncpasswd	Permet crear i canviar les contrasenyes d'accés a un servidor VNC. És possible especificar dues contrasenyes diferents: una per controlar l'escriptori (mitjançant el teclat i el ratolí) i una que només permet veure però bloqueja tota interacció.
vncconnect	Només a TightVNC. Permet establir una connexió reversa en què el servidor contacta amb un client que està en mode escolta.
vncconfig	Només a RealVNC. Permet controlar un servidor VNC que està en funcionament, per exemple sol·licitant una connexió 'reversa' amb un client.

Altres eines relacionades amb VNC són **vino**, l'eina pròpia del Gnome per compartir l'escriptori i el client **vinagre**. En l'escriptori KDE, les eines són **krfb** per compartir l'escriptori, i **krdc** per accedir a escriptoris remots mitjançant RFB o RDP.

Exemples d'ús del sistema VNC

Les pantalles VNC utilitzen una nomenclatura similar a les pantalles del sistema X Window, és a dir, un número diferent per a cada pantalla en el mateix equip. L'escriptori real en un equip sempre serà `localhost:0` o directament `:0`. Si ens referim al primer servidor VNC de l'equip 192.168.10.10 ho podrem fer com a `192.168.10.10:1`.

En arrencar un servidor VNC es pot indicar el número de *display* que volem emprar, la seva resolució i profunditat de color. Tots aquests paràmetres són opcionals i es poden ometre si els valors per defecte ens van bé.

Llançament d'un nou servidor VNC:

```
usuari@linkat2:~> vncserver

You will require a password to access your desktops.

Password:
Verify:
Would you like to enter a view-only password (y/n)? y
```

```
Password:
Verify:

New 'X' desktop is linkat2:2

Creating default startup script /home/usuari/.vnc/xstartup
Starting applications specified in /home/usuari/.vnc/xstartup
Log file is /home/usuari/.vnc/linkat2:2.log
```

En aquest moment ja està disponible el nou servidor VNC `linkat2:2`. En el directori ~/**.vnc** trobarem els fitxers de registre i el fitxer **xstartup**, que defineix els programes que s'executaran de manera inicial en aquest servidor.

Per aturar el servidor només caldrà escriure **`vncserver -kill :2.`**

En aquest cas, com que es tracta del servidor :2 escoltarà al port TCP 5902, i en el 5802 hi haurà el miniservidor web. Per a cada servidor VNC s'utilitzarà el port 5900+<*número de servidor*>, i en el port 5800+<*número de servidor*> hi haurà el servidor web que permet baixar la miniaplicació de Java per emprar com a client web.

Si es volgués obrir un client VNC de manera manual per connectar amb el servidor de l'exemple, caldria escriure en la consola **`vncviewer <adreça ip de l'estació linkat2>:2.`**

Com que el client no emmagatzema cap estat, es pot tancar en qualsevol moment sense interrompre les aplicacions que s'executen en el servidor. Tancar el client VNC és funcionalment equivalent a apagar un monitor, no es veu res però totes les aplicacions obertes continuen funcionant. En el cas del VNC, com que el monitor és virtual, després es podrà tornar a obrir en un altre lloc.

Assegurament del VNC mitjançant SSH

El trànsit de VNC és molt sensible, al cap i a la fi es transfereix tota la interacció de l'usuari amb les aplicacions. Si algú monitorés aquest trànsit podria veure tot el que l'usuari veu i tota la seva interacció, pulsació de tecles i moviment de ratolí. Per aquesta raó és fonamental emprar alguna eina per xifrar aquesta informació, i l'SSH és una eina excel·lent per a aquesta funció.

És possible establir un túnel SSH entre un port local i un port remot. Si aquest túnel es fa, per exemple, pel port 5901 remot, només haurem de sol·licitar al client VNC que realitzi la connexió al port local i deixi que l'SSH s'encarregui del transport a la xarxa fins a l'equip remot. En aquest exemple, el túnel SSH permetria accedir al primer escriptori VNC de l'equip remot.

Exemple d'establiment d'un túnel SSH

Per establir un túnel SSH entre el port 20001 local i el port 5901 de l'estació 192.168.10.10, i utilitzar el client VNC per accedir a l'escriptori remot, faríem el següent:

```
ssh -f -N -L 20001:localhost:5901 192.168.10.10
vncviewer localhost:20001
```

Les opcions **`-f`** i **`-N`** en establir la connexió SSH indiquen que la connexió es farà en segon pla i que no s'executarà cap ordre en l'estació remota. D'aquesta manera, l'execució d'SSH no bloquejarà el terminal de treball mentre el túnel estigui establert. Per aturar un túnel establert amb aquestes opcions caldrà buscar el procés SSH corresponent mitjançant l'ordre **`ps`** i emprar l'ordre **`kill`** per matar el procés.

3.2.3. *Remote desktop protocol* (RDP)

En els sistemes operatius Windows, tot i que es pot instal·lar VNC, s'utilitza de manera nativa el protocol RDP per accedir a escriptoris remots. Els ordinadors que accepten connexions als seus escriptoris estan executant el servei *terminal services*.

Entre les seves característiques n'hi ha algunes que estenen el concepte d'escriptori remot més enllà de la redirecció de la pantalla i teclat/ratolí. Algunes d'aquestes funcionalitats són les següents:

- **Redirecció d'àudio.** Permet executar aplicacions multimèdia remotes redirigint l'àudio al sistema local.
- **Redirecció del sistema d'arxius.** Permet als usuaris emprar els seus fitxers locals dins de la sessió remota.
- **Redirecció d'impressores.** Permet als usuaris emprar la seva impressora local dins de la sessió remota com si es tractés d'una impressora de xarxa.
- **Redirecció dels ports en sèrie i paral·lel.** Permet a les aplicacions remotes accedir als ports en sèrie i paral·lel de l'equip local.

En el GNU/Linux és possible fer servir el client RDP **`rdesktop`** per connectar amb un servidor RDP i accedir al seu escriptori. El client **`rdesktop`** és programari lliure sota GPL, i pot gestionar connexions fent servir la majoria de les funcionalitats de RDP v5, incloent-hi les redireccions d'àudio, dels ports en sèrie i paral·lel, del sistema d'arxius i de les impressores.

Servidor RDP en el GNU/Linux

Tot i que el més habitual en el GNU/Linux és emprar només el client RDP, per exportar escriptoris ja es fa servir VNC o X11. També és possible executar un servidor RDP com a **xrdp**, que oferirà l'escriptori als clients Windows o d'altres equips que executin **rdesktop**.

3.3. Gestió remota mitjançant una aplicació local

Una aproximació diferent a l'administració d'equips remots consisteix a executar una aplicació local amb una interfície gràfica que permeti gestionar l'estació remota. El gran avantatge d'aquesta aproximació és que, com que l'usuari interactua amb una aplicació local, el retard en la xarxa

no comporta cap problema a l'hora de treballar. A més a més, en ser una aplicació local, l'usuari coneixerà la interfície gràfica i no es donarà la situació, desconcertant per a usuaris no tècnics, d'haver de commutar la interfície de l'estació local i la remota.

Aquesta solució s'utilitza amb molta freqüència per administrar petits equips de xarxa domèstics, en què 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 SO de l'estació de l'usuari, el qual en principi és diferent per a diferents usuaris.

L'ús més adequat d'aquest enfocament té lloc en emprar un navegador web com a eina d'administració remota. És molt raonable suposar que, independentment del SO de l'usuari, aquest disposarà d'un navegador web. En aquest cas, el fabricant del dispositiu implementarà una interfície web d'administració que permetrà una gestió adequada del dispositiu per a tots els usuaris. Avui dia, la majoria de dispositius de xarxa –com els commutadors, impressores o punts d'accés– incorporen una interfície web per a la seva administració.

Fins i tot és possible administrar pel web servidors complets en fer servir eines com Webmin.

Perl

Perl és un llenguatge de programació dinàmic, dissenyat per Larry Wall i disponible en multitud de 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. Totes dues són eines excel·lents.

3.3.1. Introducció a la Webmin

Webmin és una aplicació que permet administrar un servidor de manera remota mitjançant una interfície web. Es tracta de programari lliure sota la GPL, escrit en Perl i que es pot executar en diferents sistemes operatius com GNU/Linux, OpenSolaris i FreeBSD.

Un cop instal·lat i en funcionament, la 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'inclouen mòduls que permeten gestionar des del SO (usuaris, quotes de disc, processos) fins als serveis de xarxa (Apache, Bind, Samba, etc.), tot amb una interfície web molt intuïtiva per a l'usuari.

RPM

RPM és l'acrònim de RedHat package manager, en català gestor de paquets de RedHat.

Com el nom ho indica, l'empresa RedHat va desenvolupar el propi format de paquet de programari per a la seva distribució del GNU/Linux.

Els paquets RPM són molt comuns i es fan servir en altres distribucions del GNU/Linux. L'eina de baix nivell per gestionar els paquets també es diu rpm. Tot i que normalment s'empren eines de nivell més alt que gestionen les dependències de paquets de manera automàtica.

Instal·lació de la Webmin

En la pàgina de la Webmin, es pot baixar l'aplicació en diferents formats. En tractar-se 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. Com que la Linkat2 empra el sistema de paquets RPM, baixarem la Webmin en format .rpm i la instal·larem utilitzant l'eina de gestió adequada.

Tot això es pot veure amb un exemple en dos passos. El darrer caldrà que l'executi l'administrador:

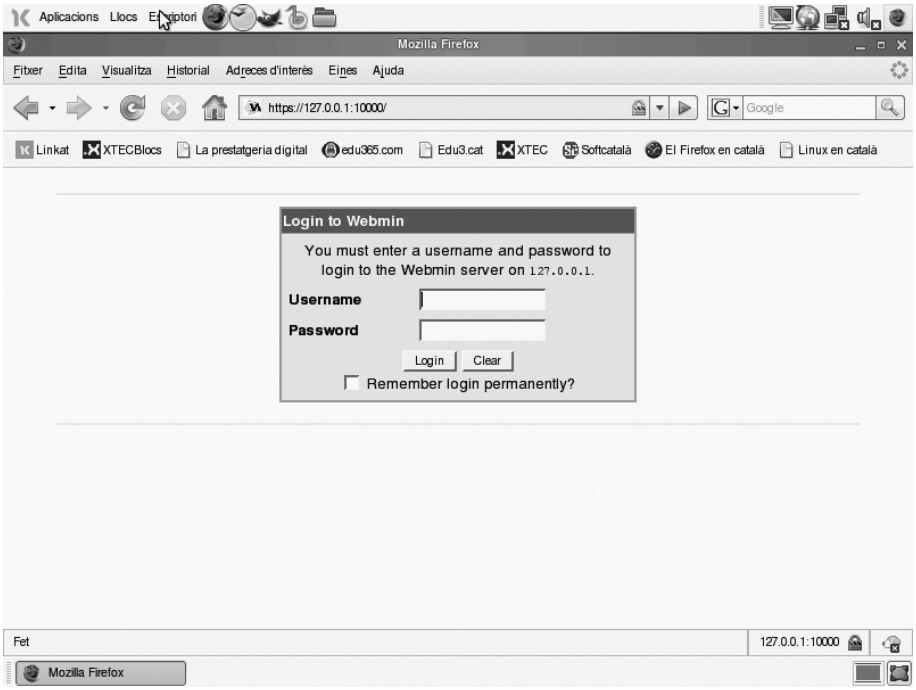
```
linkat2:~ # wget http://prdownloads.sourceforge.net/webadmin/webmin-1.410-1.noarch.rpm
Connecting to kent.dl.sourceforge.net|212.219.56.167|:80... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 14,551,234 (14M) [application/x-redhat-package-manager]

100%[======================================================>] 14,551,234  229.22K/s
                                                             ETA 00:00

13:43:16 (187.51 KB/s) - `webmin-1.410-1.noarch.rpm' saved [14551234/14551234]

linkat2:~ # rpm -Uh webmin-1.410-1.noarch.rpm
########################################### [100%]
Operating system is SuSE Linux
########################################### [100%]
Webmin install complete. You can now login to https://linkat2:10000/
as root with your root password.
linkat2:~ #
```

La instal·lació es fa en el directori /usr/libexec/webmin, la interfície web estarà disponible en el port TCP 10000, i l'usuari administrador de la Webmin serà el mateix usuari administrador del sistema, és a dir, primari amb la seva contrasenya. La Webmin pot funcionar amb HTTP i HTTPS, i després de la instal·lació inicial només està disponible l'accés mitjançant HTTPS. Per això caldrà obrir un navegador web i adreçar-lo a l'adreça **https://127.0.0.1:10000**. En la figura 6 podrem veure la pàgina d'inici de la sessió per a la Webmin que s'està executant de manera local.

HTTP i HTTPS

La diferència entre els protocols HTTP i HTTPS és que el segon consisteix a fer servir l'HTTP combinat amb l'SSL (*secure socket layer*) per xifrar tota la informació transmesa.

Figura 6. Pàgina d'entrada a la Webmin



Un cop instal·lada la Webmin, el fitxer **/etc/init.d/webmin** ens permetrà engegar i aturar l'aplicació com qualsevol altre servei. És important remarcar que la instal·lació per defecte en la Linkat2 no marca el servei per arrencar-lo de manera automàtica durant el procediment d'engegada del sistema.

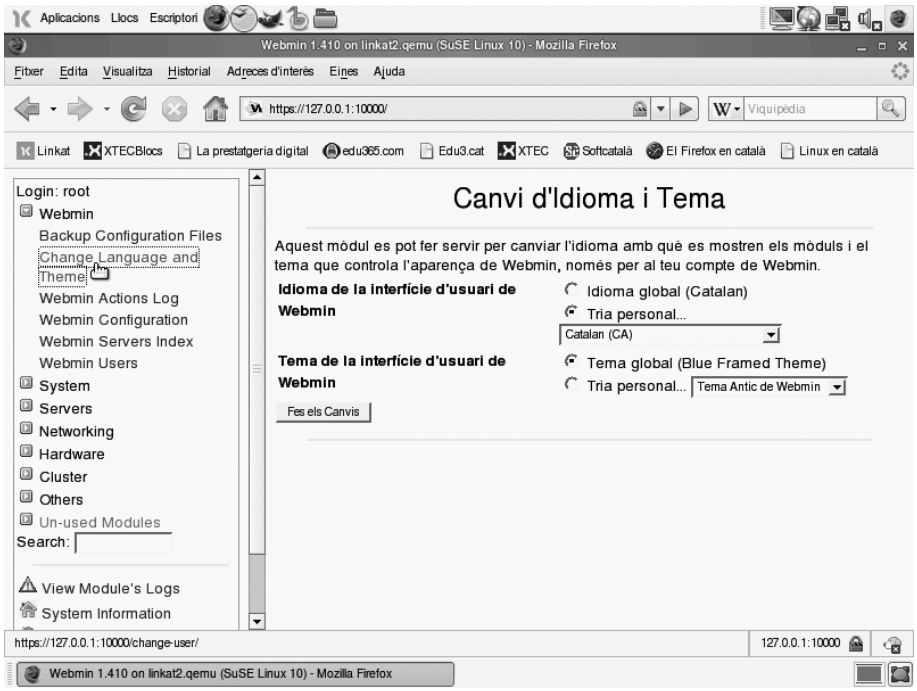
Si reiniciem el servidor Linkat haurem de tornar a engegar de manera manual la Webmin, si la volem engegar de manera automàtica caldrà que fem servir les eines de gestió dels serveis del propi SO.

Funcions de la Webmin

La 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 dins de la Webmin serà configurar l'idioma de la interfície. Aquesta interfície està disponible en molts idiomes 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 mostra la figura 7, o bé a *Webmin\Webmin Configuration* per a tota l'aplicació.

Figura 7. Opcions d'idioma i aspecte per la interfície de la Webmin



La interfície de Webmin és molt clara i està disponible en diferents idiomes.

Les opcions de la interfície de Webmin estan agrupades dins de pestanyes generals, que un cop traduïda la interfície, són les que es mostren en la taula 11.

Taula 11. Pestanyes 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 què s'executa Webmin. Des del control d'usuaris, quotes de disc, tasques periòdiques i gestió de paquets fins a les còpies de seguretat.
Servidors	Mòduls per administrar els diferents serveis que es poden executar en l'equip. Alguns dels més corrents són Samba, Postix, OpenSSH, Squid, etc.
Xarxa	Configurar tots els paràmetres de xarxa de l'equip i els seus adaptadors. A més permet configurar, entre altres, el tallafoc o el monitoratge dels adaptadors de xarxa.
Maquinari	Permet gestionar tot el maquinari: impressores, particions, RAID, gestor d'engegada GRUB, etc.
Clúster	Permet gestionar de manera unificada un grup d'equips que executen Webmin.
Altres	Permet gestionar fitxers, comprovar el funcionament dels serveis, instal·lar/desinstal·lar programari i, fins i tot, realitzar connexions Telnet/SSH des del propi navegador al servidor.