Administració de servidors

Remo Suppi Boldrito

P07/M2003/02286

UOC
www.uoc.edu

Índex

Introducció

La interconnexió entre màquines i les comunicacions d'alta velocitat han permès que els recursos que s'utilitzin no siguin al mateix lloc geogràfic de l'usuari. UNIX (i sens dubte GNU/Linux) és probablement el màxim exponent d'aquesta filosofia, ja que des del seu inici ha fomentat el compartir recursos i la independència de "dispositius". Aquesta filosofia s'ha plasmat en una cosa comú avui en dia, que són els serveis. Un servei és un recurs (que pot ser universal o no) i que permet en condicions determinades obtenir informació, compartir dades o simplement processar la informació remotament. El nostre objectiu és analitzar els serveis que permeten el funcionament d'una xarxa. Generalment, dins d'aquesta xarxa hi haurà una màquina (o diverses, segons les configuracions) que possibilitarà l'intercanvi d'informació entre les altres. Aquestes màquines es denominen servidors i contenen un conjunt de programes que permeten que la informació estigui centralitzada i sigui fàcilment accessible. Aquests serveis propicien la reducció de costos i amplien la disponibilitat de la informació, però s'ha de tenir en compte que un servei centralitzat presenta inconvenients, ja que pot quedar fora de línia i deixar sense servei tots els usuaris.

Una arquitectura de servidors ha de tenir els serveis replicats (*mirrors*) per a resoldre aquestes situacions.

Els serveis es poden classificar en dos tipus: de vinculació ordinador-ordinador o de relació home-ordinador. En el primer cas, es tracta de serveis requerits per altres ordinadors, mentre que en el segon, són serveis requerits pels usuaris (encara que hi ha serveis que poden actuar en ambdues categories). Dins del primer tipus es troben serveis de noms, com el *domain name system* (DNS), el servei d'informació d'usuaris (NIS/YP), el directori d'informació LDAP o els serveis d'emmagatzematge intermediari (*proxies*). Dins de la segona categoria, s'inclouen serveis de connexió interactiva i execució remota (SSH, telnet), transferència de fitxers (FTP), intercanvi d'informació en l'àmbit d'usuari, com el correu electrònic (MTA, IMAP, POP), *news*, World Wide Web, *Wiki* i arxius (NFS). Per a mostrar les possibilitats de GNU/Linux Debian-FC6, es descriuran cadascun d'aquests serveis amb una configuració mínima i operativa, però sense descuidar aspectes de seguretat i estabilitat.

1. *Domain name system* (DNS)

La funcionalitat del servei de DNS (com es va explicar en la unitat dedicada a l'administració de xarxa) és convertir noms de màquines (llegibles i fàcils de recordar pels usuaris) en adreces IP o viceversa.

Exemple

A la consulta de quin és l'IP de pirulo.remix.com, el servidor respondrà 192.168.0.1 (aquesta acció es coneix com a *mapping*); de la mateixa manera, quan se li proporcioni l'adreça IP, respondrà amb el nom de la màquina (conegut com a *reverse mapping*).

El *domain name system* (DNS) és una arquitectura arborescent per evitar duplicació de la informació i facilitar la recerca. Per això, un únic DNS no té sentit sinó com a part de l'arbre.

L'aplicació que presta aquest servei s'anomena *named*, s'inclou en la majoria de distribucions de GNU/Linux (/usr/sbin/named) i forma part d'un paquet anomenat *bind* (actualment versió 9.x) coordinat per ISC (*Internet Software Consortium*). DNS és simplement una base de dades, per la qual cosa és necessari que les persones que la modifiquin en coneguin l' estructura, ja que, altrament, el servei quedarà afectat. Com a precaució, s'ha de tenir especial compte a l'hora de guardar les còpies dels arxius per a evitar qualsevol interrupció en el servei. El paquet sobre Debian es troba com a bind i bind.doc. [LN01, Deb03c, IET03]. Les configuracions són similars, i en FC, per exemple, heu d'instal·lar bind, bind-utils and caching-nameserver que seran gestionades pel yum.

1.1. Servidor de noms cau

En primer lloc es configurarà un servidor de DNS per a resoldre consultes que actuï com a cau per a les consultes de noms (*resolver, caching only server*). És a dir, la primera vegada consultarà el servidor adequat perquè es parteix d'una base de dades sense informació, però les vegades següents respondrà el servidor de noms cau, amb la disminució del temps de resposta corresponent. Per a configurar el servidor de noms cau, es necessita l'arxiu /etc/bind/named.conf (a Debian), que té el format següent (s'han respectat els comentaris originals dins de l'arxiu, indicats per //):

```
options {
directory "/var/cache/bind";
        // query-source address * port 53;
        // forwarders {
        //0.0.0.0;
        //
        };
        auth-nxdomain no; # conform to RFC1035
        };
```

```
// prime the server with knowledge of the root servers}
zone "." {
        type hint;
        file "/etc/bind/db.root"; };
        // be authoritative for the localhost forward and reverse zones, and for
        // broadcast zones as per RFC 1912
        }
zone "localhost" {
        type master;
        file "/etc/bind/db.local";
        };
zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
        };
zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.0";
        };
zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.255";
        };
// add entries for other zones below here
}
```

La sentència *directory* indica on es troben els arxius de configuració restants (/var/cache/bind en el nostre cas). L'arxiu /etc/bind/db.root contindrà una cosa similar al següent (es mostren només les primeres línies que no són comentaris indicats per ';' i s'ha d'anar amb compte amb els punts (.) a l'inici d'algunes línies –es pot obtenir directament d'Internet actualitzat–):

```
...
; formerly NS.INTERNIC.NET
;
. 3600000 IN NS A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET. 3600000 A 198.41.0.4
;
; formerly NS1.ISI.EDU
;
. 3600000 NS B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET. 3600000 A 128.9.0.107
;
; formerly C.PSI.NET
;
. 3600000 NS C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET. 3600000 A 192.33.4.12
;
...
```

Aquest arxiu descriu els *root name servers* al món. Aquests servidors canvien, per la qual cosa s'ha d'actualitzar periòdicament l'arxiu. Les seccions següents són les *zone*; les *zones localhost* i 127. in-addr.arpa, que es vinculen als fitxers en l'arxiu /etc/bind/db.local i /etc/bind/db.127, es refereixen a la resolució directa i inversa per a la interfície local. Les *zones* següents són per a les zones de difusió (segons RFC 1912) i al final s'hi haurien d'agregar les pròpies. Per exemple, l'arxiu db.local podria ser (';' significa 'comentari'):

```
; BIND reverse data file for local loopback interface
$TTL 604800
@       IN      SOA     ns.remix.bogus.      root.remix.bogus. (
        1                       ; Serial
        604800 ;                 Refresh
```

```
        86400           ; Retry
        2419200         ; Expire
        604800);         Negative Cache TTL
@       IN      NS      ns.remix.bogus.
1.0.0   IN      PTR     localhost.
```

Explicarem la seva utilització més endavant. El següent és posar com a *name server* en el /etc/resolv.conf:

```
search subdomini.su-domini.domini su-domini.domini
# per exemple search remix.bogus bogus
nameserver 127.0.0.1
```

En què s'hauran de reemplaçar els subdomini.su-domini.domini pels valors adequats. La línia *search* indica quins dominis es buscaran per a qualsevol *host* que es vulgui connectar (és possible substituir-hi *search* per *domain*, encara que tenen comportaments diferents) i el *nameserver* l'especifica l'adreça del seu *nameserver* (en aquest cas la seva pròpia màquina, que és on s'executa el *named*). El *search* té el comportament següent: si un client busca la màquina pirulo, primer es buscarà pirulo.subdomini.su-domini.domini, després pirulo.su-domini.domini i finalment, pirulo. Això implica temps de recerca; ara bé, si es té la seguretat que pirulo és a subdomini.su-domini.domini, no és necessari posar-hi els restants.

El pas següent és posar en marxa el *named* i mirar els resultats de l'execució. Per a posar en marxa el *daemon*, es pot fer directament amb l'*script* d'inicialització `/etc/init.d/bind9 start` (en cas que el *named* ja s'estigui executant, feu `/etc/init.d/bind9 reload`) o, si no, `/usr/sbin/named`. En mirar el *log* del sistema en /var/log/daemon.log veurem alguna cosa com:

```
Sep 1 20:42:28 remolix named[165]: starting BIND 9.2.1 \\
Sep 1 20:42:28 remolix named[165]: using 1 CPU \\
Sep 1 20:42:28 remolix named[167]: loading configuration from '/etc/bind/named.conf'
```

Aquí s'indica l'arrencada del servidor i els missatges d'errors (si n'hi ha), els quals s'hauran de corregir i de tornar a començar. Ara es pot verificar la configuració amb ordres com `nslookup` (original, fàcil però obsolet segons alguns autors), `host` o `dig` (recomanat). La sortida de `dig -x 127.0.0.1` serà alguna cosa semblant a:

```
# dig -x 127.0.0.1
;; <<>> DiG 9.2.1 <<>> -x 127.0.0.1
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31245
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0
;; QUESTION SECTION: ;1.0.0.127.in-addr.arpa. IN PTR
;;ANSWER SECTION: 1.0.0.127.in-addr.arpa. 604800 IN PTR localhost.
;; AUTHORITY SECTION: 127.in-addr.arpa. 604800 IN NS ns.remix.bogus.
;; Query time: 1 msec
;; SERVER: 127.0.0.1 #53(127.0.0.1)
;; WHEN: Mon Sep 1 22:23:35 2003
;; MSG SIZE rcvd: 91
```

En què es pot veure que la consulta ha trigat 1 mil·lisegon. Si es disposa de connexió a Internet, es podria buscar alguna màquina dins del vostre domini i veure el comportament del vostre servidor. En BIND9 hi ha el lwresd (*lightweight resolver daemon*), que és el *daemon* que proveeix serveis de noms a clients que utilitzen la biblioteca de BIND9 *lightweight resolver*. És essencialment un servidor cau (com el que s'ha configurat) el que fa les consultes utilitzant el BIND9 *lightweight resolver protocol* en lloc del protocol DNS. Aquest servidor escolta per la interfície 127.0.0.1 (per la qual cosa, només atén processos de la màquina local) en UDP i el port 921. Les consultes dels clients són descodificades i són resoltes utilitzant el protocol DNS. Quan s'obtenen les respostes, el lwresd les codifica en el format *lightweight* i les retorna al client que les ha sol·licitat.

Finalment, com ja s'ha esmentat, el nucli utilitza diverses fonts d'informació, que per a la xarxa s'obtenen des de `/etc/nsswitch.conf`. Aquest arxiu indica des d'on obtenir la font d'informació i per al cas dels noms de màquines i IP hi ha una secció com:

hosts: files dns

Aquesta línia (si no hi és, s'ha d'agregar) indica que qui necessiti un nom d'una màquina o una IP consulti primer en `/etc/hosts` i després en DNS d'acord amb els dominis indicats en `/etc/resolv.conf`.

1.2. *Forwarders*

En xarxes amb una càrrega considerable, és possible equilibrar el trànsit utilitzant la secció de *forwarders*. Si el vostre proveïdor de xarxa (ISP) en té un o més *nameservers* estables, és recomanable utilitzar-los per a descongestionar les consultes sobre el seu servidor. Per això, s'ha de treure el comentari (//) de cada línia de la secció *forwarders* de l'arxiu /etc/bind/named.conf i reemplaçar el 0.0.0.0 amb les IP dels *nameservers* del seu ISP. Aquesta configuració és recomanable quan la connexió és lenta, per exemple, per mòdem.

1.3. Configuració d'un domini propi

DNS té una estructura en arbre i l'origen es coneix com a '.' (vegeu /etc/bind/db.root). Sota el '.' hi ha els TLD (*top level domains*) com org, com, edu, net, etc. Quan es busca en un servidor, si aquest no coneix la resposta, es buscarà recursivament en l'arbre fins a trobar-la. Cada '.' en una adreça (per exemple, pirulo.remix.com) indica una branca de l'arbre de DNS diferent i un àmbit de consulta (o de responsabilitat) diferent que s'anirà recorrent de manera recursiva d'esquerra a dreta.

Un altre aspecte important, a més del domini, és l'in-addr.arpa (*inverse mapping*), el qual també està imbricat com els dominis i serveix per a obtenir noms quan es consulta per l'adreça IP. En aquest cas, les adreces s'escriuen al revés, en concordança amb el domini. Si pirulo.remix.com és la 192.168.0.1, s'escriurà com a 1.0.168.192, en concordança amb pirulo.remix.com.

A continuació, configurarem el domini propi remix.bogus a l'arxiu /etc/bind/db.127 [LN01]:

```
; BIND reverse data file for local loopback interface
$TTL 604800
@ IN SOA ns.remix.bogus. root.remix.bogus. (
        1                       ; Serial
        604800  ;               Refresh
        86400                   ; Retry
        2419200                 ; Expire
        604800 )                ; Negative Cache TTL
@       IN      NS              ns.remix.bogus.
1.0.0   IN      PTR             localhost.
```

S'ha de tenir en compte el '.' al final dels noms de domini. L'origen de la jerarquia d'una *zone* és especificat per la identificació de la zona, en el nostre cas 127.in-addr.arpa. Aquest arxiu (db.127) conté 3 registres: SOA, NS, PTR. El SOA (*start of authority*) ha de ser en tots els arxius de zona a l'inici, després de TTL, i el símbol @ significa l'origen del domini; NS, el servidor de noms per al domini, i PTR (*domain name pointer*), que és el host 1 en la subxarxa (127.0.0.1) i es denomina *local host*. Aquest és l'arxiu sèrie 1 i el seu responsable és root@remix.bogus (últim camp de la línia SOA). Ara es podria reiniciar el *named* de la forma abans indicada i amb el `dig -x 127.0.0.1`, veure el seu funcionament (que seria idèntic al mostrat anteriorment).

A continuació, caldrà afegir una nova zona en el named.conf:

```
zone "remix.bogus" {
   type master;
   notify no;
   file "/etc/bind/remix.bogus";
   };
```

S'ha de recordar que en el named.conf els dominis van sense el '.' final. A l'arxiu remix.bogus es posaran els *hosts* dels quals serem responsables:

```
; Zone file for remix.bogus
$TTL  604800
@       IN      SOA     ns.remix.bogus. root.remix.bogus. (
        199802151       ; serial, todays date + todays serial
        604800          ; Refresh
        86400           ; Retry
        2419200         ; Expire
        604800 )        ; Negative Cache TTL
@       NS      ns      ; Inet Address of name server
        MX      10      mail.remix.bogus. ; Primary Mail Exchanger
localhost       A       127.0.0.1
ns              A       192.168.1.2
mail            A       192.168.1.4
                TXT "Mail Server"
ftp             A       192.168.1.5
                MX      10 mail
www             CNAME   ftp
```

Hi surt un nou registre MX que és el Mail eXchanger. És el lloc al qual s'enviaran els correus electrònics que arribin, algú@remix.bogus, i serà a mail.remix.bogus (el número indica la prioritat si tenim més d'un MX). Cal recordar el '.' necessari sempre en els arxius de *zone* al final del domini (si no s'hi posen, el sistema agrega el domini SOA al final, la qual cosa transformaria, per exemple, mail.remix.bogus en mail.remix.bogus.remix.bogus, que és incorrecte). CNAME (*canonical name*) és la forma de dona a una màquina un o diversos àlies. A partir d'ara ens trobaríem en condicions (després de /etc/init.d/bind9 reload) de provar, per exemple, `dig www.remix.bogus`.

L'últim pas és configurar la zona inversa, és a dir, perquè pugui convertir adreces IP en noms, per exemple, agregant-hi una nova zona:

```
zone "192.168.1.in-addr.arpa" {
        type master;
        notify no;
        file "/etc/bind/192.168.1";
        };
```

I l'arxiu /etc/bind/192.168.1 similar a l'anterior:

```
$TTL 604800
@       IN      SOA     ns.remix.bogus. root.remix.bogus. (
        199802151       ; serial, todays date + todays serial
        604800          ; Refresh
        86400           ; Retry
        2419200         ; Expire
        604800 ) ; Negative Cache TTL
@       NS      ns.remix.bogus.
2       PTR     ns.remix.bogus
4       PTR     mail.remix.bogus
5       PTR     ftp.remix.bogus
```

Aquest es podria provar novament amb `dig -x 192.168.1.4`. S'ha de tenir en compte que aquests exemples són sobre IP privades, és a dir, no IP d'Internet. Una altra qüestió important és no oblidar el *notify no*, ja que si no els nostres experiments amb DNS es propagaran als servidors de l'arbre de DNS (fins i tot modificant els DNS del nostre proveïdor o institució). Només s'ha de modificar quan estem segurs que funciona i volem propagar els canvis. Per a veure un exemple real, consulteu DNS-HOWTO en http://tldp.org/HOWTO/DNS-HOWTO-7.html.

Una vegada creat un *master server,* s'ha de crear un *slave server* per seguretat, que és idèntic al *master*, excepte que la zona en lloc de *type master* haurà de tenir *slave* i la IP del *master*. Per exemple:

```
zone "remix.bogus" {
        type slave;
        notify no;
        masters {192.168.1.2; }
        };
```

2. NIS (YP)

A fi de facilitar l'administració i donar comoditat a l'usuari, en xarxes de diferents mides que executen GNU/Linux (o Sun o algun altre sistema operatiu amb suport per a aquest servei) s'executen serveis de Network Information Service, NIS (o *yellow pages,* YP, en la definició original de Sun). GNU/Linux pot donar suport com a client/servidor de NIS i poden actuar com a client (versió "beta") de NIS+, que és una versió més segura i optimitzada de NIS. La informació que es pot distribuir en NIS és: usuaris (*login names*), paraules d'accés (*passwords*) (/etc/passwd), directoris d'usuari (*home directories*), informació de grups (*group information*) (/etc/group), la qual cosa presenta l'avantatge que, des de qualsevol màquina client o des del mateix servidor, l'usuari es podrà connectar amb el mateix compte i *password* i al mateix directori (encara que el directori s'haurà de muntar anteriorment sobre totes les màquines clients per NFS o utilitzant el servei d'*automount*). [Miq01, Kuk03]

L'arquitectura NIS és del tipus client-servidor, és a dir, hi ha un servidor que disposa de totes les bases de dades i uns clients que consulten aquestes dades de manera transparent per a l'usuari. Per això, s'ha de considerar la configuració de servidors "de reforç" (anomenats secundaris) perquè els usuaris no quedin bloquejats davant la caiguda del servidor principal. Per això, l'arquitectura realment es denomina de múltiples servidors (*master+mirrors*+clients).

2.1. Com iniciar un client local de NIS en Debian?

Un client local significa annexar l'ordinador a un domini NIS ja existent:

- Primer s'ha de verificar que es tenen instal·lats els paquets *netbase* (xarxa bàsica TCP/IP), *portmap* (servidor que converteix números RPC en ports DARPA i és necessari per a programes que executen RPC, inclosos NFS i NIS), i *nis* (específic). Es recomana utilitzar l'ordre kpackage o directament amb apt-get (es pot verificar si està instal·lat amb `apt-cache pkgnames`) en forma de text. El procediment d'instal·lació del paquet NIS sol·licita un domini (NIS *domainname*). Aquest és un nom que descriu el conjunt de màquines que utilitzaran el NIS (no és un nom de *host*). Cal tenir en compte que NISPirulo és diferent de Nispirulo com a nom de domini. Per a configurar aquest domini, es pot utilitzar l'ordre `nisdomainname`, domini que s'emmagatzema a /proc/sys/kernel/domainname.

- En primer lloc s'ha d'iniciar el servei portmap amb:

  – `/etc/init.d/portmap start`
  – Es pot comprovar si aquests serveis són actius amb `rpcinfo -p`.

- Si el servidor NIS no és local, s'haurà d'utilitzar l'ordre ypbind. L'ordre ypbind s'utilitza per a trobar un servidor per al domini especificat, ja sigui mitjançant *broadcast* (no aconsellat per insegur) o buscant el servidor indicat a l'arxiu de configuració /etc/yp.conf (recomanable). L'arxiu /etc/yp.conf té la sintaxi següent:

  – *domain nisdomain server hostname*: indica que s'ha d'utilitzar el *hostname* per al domini *nisdomain*. Es podria tenir més d'una entrada d'aquest tipus per a un únic domini.

  – *domain nisdomain broadcast*: indica que s'ha d'utilitzar *broadcast* sobre la xarxa local per a descobrir un servidor de domini *nisdomain*.

  – *ypserver hostname*: indica que cal utilitzar *hostname* com a servidor. És recomanable utilitzar aquesta línia (*ypserver*) en la qual s'haurà d'introduir l'adreça IP del servidor NIS. Si s'hi indica el nom ens hem d'assegurar que es pot trobar per DNS la IP o la mateixa figura en l'arxiu /etc/hosts ja que, d'una altra manera, el client es bloquejarà.

- Inicïi el servei executant:

  – `/etc/init.d/nis stop`

  i després:

  – `/etc/init.d/nis start`

- A partir d'aquest moment, el client NIS estarà funcionant (es pot confirmar amb `rpcinfo -u localhost ypbind`, que mostra les dues versions del protocol actiu) o es pot utilitzar l'ordre `ypcat mapname` (per exemple, `ypcat passwd`, que mostra els usuaris NIS definits en el servidor) en què la relació *mapnames* a taules de la base de dades NIS estan definits en /var/yp/nicknames.

2.2. Quins recursos s'han d'especificar per a utilitzar en NIS?

Considerarem que tenim instal·lada una de les últimes distribucions de Debian (per exemple, 3.0 Woody o 3.1 *Sarge*), que suporta la Libc6 (igualment per a FC4 o superior), i es vol que els usuaris en una màquina client puguin

accedir a la informació del servidor. En aquest cas, s'ha d'orientar la consulta del *login* a les bases de dades adequades de la manera següent:

1) Verificar el fitxer /etc/nsswitch.conf i assegurar-se que les entrades passwd, group, shadow i netgroup són similars a:

passwd: compat
group: compat
shadow: compat
...
netgroup: nis
Vegeu *man nsswitch.conf* per a la sintaxi d'aquest arxiu.

2) Agregar la línia següent a les màquines clients NIS en el fitxer /etc/passwd al final de l'arxiu (indicarà que si l'usuari no és local, s'ho preguntarà al servidor de NIS):

+:::::: (un '+' i sis ':')

3) S'ha de tenir en compte que en el /etc/passwd es pot utilitzar el + i el **?** davant cada nom d'usuari en el /etc/passwd per a incloure/excloure el *login* d'aquests usuaris (*override*). Si s'està utilitzant *passwords* amb *shadow* (més segur, ja que no permet que un usuari normal pugui veure el *password* encriptat d'altres usuaris) haurà d'incloure la línia següent al final de l'arxiu /etc/shadow:

+::::::::: (un '+' i vuit ':')

4) Ha d'afegir també la línia següent al final de /etc/group:

+::: (un '+' i tres ':')

5) Les cerques de *hosts* (*hosts lookups*) es fan mitjançant DNS (i no per NIS), per la qual cosa, per a aplicacions Libc6 en el fitxer /etc/nsswitch.conf caldrà canviar l'entrada hosts per la línia següent: `hosts: files dns`. O, si es prefereix fer per NIS, `hosts: files nis`. Per a aplicacions Libc5, s'haurà de modificar el fitxer /etc/host.conf posant-hi order hosts, dns o order hosts, nis segons es vulgui.

> Amb aquesta configuració es pot fer una connexió local (sobre el client NIS) a un usuari que no estigui definit en el fitxer /etc/passwd, és a dir, un usuari definit en una altra màquina (ypserver).

Per exemple, es podria fer `ssh -l user localhost`, en què *user* és un usuari definit en ypserver.

2.3. Com s'ha d'executar un *master* NIS *server*?

Considerem que sobre la màquina es té instal·lat el paquet nis i el portmap (aquest últim en funcionament) i que les bases de dades del NIS estan creades (vegeu l'apartat següent):

- Cal assegurar-se que en el /etc/hosts es troben totes les màquines que formaran part del domini en el format FQDN (*fully qualified domain name*), que és on s'indiquen la *IP*, el nom i domini i el nom sense domini de cada màquina (per exemple, 192.168.0.1 pirulo.remix.com pirulo). Això és necessari només en el *server,* ja que el NIS no utilitza DNS.

- A més, és a l'arxiu /etc/defaultdomain amb el nom del domini escollit. No utilitzeu el vostre domini DNS per a no incórrer en un risc de seguretat, excepte que configureu adequadament els arxius /etc/ypserv.securenets (que indica amb un parell netmask/network des de quin lloc s'hi poden connectar els clients) i /etc/ypserv.conf (que fa un control més detallat perquè indica quins *hosts* poden accedir a quins mapes, per exemple: passwd.byname o shadow.byname).

- S'ha de verificar que hi ha *NISSERVER* = *master* en /etc/default/nis.

- Per motius de seguretat, es pot agregar el número de xarxa local a l'arxiu /etc/ypserv.securenets.

- Cal iniciar el servidor executant l'ordre `/etc/init.d/nis stop` i després `/etc/init.d/nis start`. Aquesta sentència iniciarà el *server* (ypserv) i el *password daemon* (yppasswdd), els quals es poden consultar si és actiu amb `ypwich -d domain`.

2.4. Com s'ha de configurar un *server*?

La configuració del *server* es fa amb l'ordre `/usr/lliura/yp/ypinit -m` tanmateix, és necessari verificar que hi és l'arxiu `/etc/networks`, que és imprescindible per a aquest *script*.

Si aquest arxiu no hi és, creeu-ne un de buit amb `touch /etc/networks`. També es pot fer que sobre el servidor s'executi el client ypbind; així, tots els usuaris entren per NIS, com es va indicar anteriorment, modificant el fitxer /etc/passwd en el qual totes les entrades normals abans de la línia +:::::: són ignorades pel NIS (només hi poden accedir localment), mentre que les posteriors hi poden accedir pel NIS des de qualsevol client. [Miq01]

Considereu que a partir d'aquest moment les ordres per a canviar el passwd o informació dels usuaris com passwd, chfn, adduser no són vàlids. Al seu lloc, s'hauran d'utilitzar ordres com yppasswd, ypchsh i ypchfn. Si es canvien els usuaris o es modifiquen els arxius esmentats, caldrà reconstruir les taules de NIS executant l'ordre `make` en el directori /var/yp per a actualitzar les taules.

Tenir en compte que Libc5 no suporta shadow passwd, per la qual cosa no s'ha d'utilitzar *shadow* amb NIS si es tenen aplicacions amb Libc5. No hi haurà cap problema si es té Libc6 que accepta NIS amb suport *shadow*.

La configuració d'un *slave server* és similar a la del *master* llevat pel fet que *NISSERVER* = *slave* en /etc/default/nis. Sobre el *master* s'ha d'indicar que distribueixi les taules automàticament als *slaves* posant-hi *NOPUSH* = "*false*" a l'arxiu /var/yp/Makefile.

Ara s'ha d'indicar al *master* qui és el seu esclau executant:

```
/usr/lib/yp/ypinit -m
```

i entrant els noms dels *slave servers*. Això reconstruirà els mapes, però no enviarà els arxius als *slaves*. Per això, sobre el *slave*, cal executar:

```
/etc/init.d/nis stop
/etc/init.d/nis start
```

i, finalment:

```
/usr/lib/yp/ypinit -s nom_master_server.
```

Així l'*slave* carregarà les taules des del *master*.

També es podria posar en el directori /etc/cron.d l'arxiu nis amb un contingut similar a (recordeu fer un `chmod 755 /etc/cron.d/nis`):

```
20 * * * * root /usr/lib/yp/ypxfr_1perhour >/dev/null 2 > & 1
40 6 * * * root /usr/lib/yp/ypxfr_1perday >/dev/null 2 > & 1
55 6,18 * * * root /usr/lib/yp/ypxfr_2perday >/dev/null 2 > & 1
```

Amb això, ens assegurem que tots els canvis del *master* es transfereixen al servidor NIS *slave*.

Recomanació: Després d'usar `adduser` per a agregar un nou usuari sobre el servidor executeu `make -C /var/yp` per a actualitzar les taules NIS (i sempre que es canviï alguna característica de l'usuari, per exemple la paraula clau amb l'ordre passwd, que només canviarà el *password* local i no el de NIS). Per a provar que el sistema està funcionant i l'usuari donat d'alta és en el NIS, podeu fer `ypmatch userid passwd` en què userid és l'usuari donat d'alta amb adduser prèviament i després d'haver fet el make. Per a verificar el funcionament del sistema NIS, es pot utilitzar l'*script* de http://tldp.org/HOWTO/NIS-HOWTO/verification.html, que permet una verificació més detallada sobre el NIS.

3. Serveis de connexió remota: telnet i ssh

3.1. Telnet i telnetd

Telnet és una ordre (client) utilitzada per a comunicar-se interactivament amb un altre *host* que executa el *daemon* telnetd. L'ordre telnet es pot executar com a telnet host o interactivament com a telnet, el qual posarà el prompt "telnet>" i, després per exemple: *open host*. Una vegada establerta la comunicació, s'haurà d'introduir l'usuari i el *password* amb el qual es vulgui connectar el sistema remot. Es disposa de diverses ordres (de manera interactiva) com open, logout, mode (definiu les característiques de visualització), close, encrypt, quit, set, unset, o es poden executar ordres externes amb '!'. Es pot utilitzar l'arxiu /etc/telnetrc per a definicions per defecte, o .telnetrc per a definicions d'un usuari particular (haurà de ser en el directori home d'usuari).

El *daemon* telnetd és el servidor de protocol telnet per a la connexió interactiva. telned el posa en marxa generalment el *daemon* inetd i es recomana incloure-hi un wrapper tcpd (que utilitza les regles d'accés en host.allow i el host.deny) en la crida al telnetd dins de l'arxiu /etc/inetd.conf. Per exemple, pot incloure una línia com:

telnet stream tcp nowait telnetd.telenetd /usr/sbin/tcpd /usr/bin/in.telnetd

Per a incrementar la seguretat del sistema, vegeu la unitat dedicada a la seguretat. En algunes distribucions (Debian 3.0 o superiors), la funcionalitat d'inetd es pot reemplaçar per xinetd, que requereix que es configuri l'arxiu /etc/xinetd.conf (vegeu la unitat dedicada a la d'administració de xarxa). Si igualment es vol posar en marxa inetd a tall de proves es pot utilitzar la sentència `/etc/init.d/inetd.real start`. Si l'arxiu /etc/uissue.net és present, el telnetd en mostrarà el contingut a l'inici de la sessió. També es pot utilitzar /etc/security/access.conf per a habilitar/deshabilitar *logins* d'usuari, *host* o grups d'usuaris, segons es connectin.

S'ha de recordar que, si bé el parell telnet-telnetd poden funcionar de manera *encrypt* en les últimes versions (transferència de dades encriptades, encara que han d'estar compilades amb l'opció corresponent), és una ordre que ha quedat en l'oblit per la seva falta de seguretat encara que es pot fer servir en xarxes segures o situacions controlades.

Si no està instal·lat es pot utilitzar (Debian) `apt-get install telnetd` i després verificar que s'ha donat d'alta o bé en /etc/inetd.conf o en /etc/xinetd.conf (o en el directori que estiguin definits els arxius per exemple /etc/xinetd.d

segons s'indiqui a l'arxiu anterior amb la sentència include /etc/xinetd.d). O bé en el xinetd.conf o bé en l'arxiu /etc/xinetd.d/telentd haurà d'incloure una secció com (qualsevol modificació en xinetd.conf haurà de rearrencar el servei amb `service xinetd restart`):

```
service telnet
{
disable = no
flags = REUSI
socket_type = stream
wait = no
user = root
server = /usr/sbin/in.telnetd
log_on_failure += USERID
}
```

Es recomana, en lloc d'utilitzar telnetd, o bé utilitzar SSL telnet(d), el qual reemplaça el telnet(d) utilitzant encriptació i autenticació per SSL, o bé utilitzar SSH (secció següent). El SSLTelnet(d) pot funcionar amb el telnet(d normal en ambdues direccions, ja que a l'inici de la comunicació verifica si de l'altre costat (peer) suporta SSL i si no continua amb el protocol telnet normal. Els avantatges respecte al telnet(d) són que els seus *passwords* i dades no circulen per la xarxa en forma de text pla i ningú que utilitzi per exemple tcpdump no podrà veure el contingut de la comunicació. També amb SSLtelnet es pot utilitzar per a connectar-se per exemple a un servidor web segur (per exemple https://servidor.web.org) simplement fent: `telnet servidor.web.org 443`

3.2. SSH, *Secure shell*

Un canvi aconsellable avui en dia és utilitzar ssh en lloc de telnet, rlogin o rsh. Aquestes ordres són insegures (excepte SSLTelnet) per diverses raons: la més important és que tot el que es transmet per la xarxa, inclosos *usernames* i *passwords*, és en text pla (encara que hi ha versions de telnet-telnetd encriptats, han de coincidir en el fet que ambdós ho siguin), qualsevol que tingui accés a aquesta xarxa o a algun dels seus segments pot obtenir tota aquesta informació i després suplantar la identitat de l'usuari. La segona és que aquests ports (telnet, rsh...) són al primer lloc al qual un *cracker* s'intentarà connectar. El protocol ssh (en la seva versió OpenSSH) proveeix una connexió encriptada i comprimida molt més segura que, per exemple, telnet (és recomanable utilitzar la versió 2 del protocol). Totes les distribucions actuals incorporen el client ssh i el servidor sshd per defecte.

3.2.1. ssh

Per executar l'ordre, feu:

```
ssh -l login name host o ssh user@hostname
```

Per mitjà de SSH es poden encapsular altres connexions com X11 o qualsevol altra TCP/IP. Si s'omet el paràmetre -l, l'usuari es connectarà amb el mateix usuari local i en ambdós casos el servidor sol·licitarà el passwd per a validar la identitat de l'usuari. SSH suporta diferents maneres d'autenticació (vegeu man ssh) basades en algoritme RSA i clau pública.

Si s'utilitza l'ordre `ssh-keygen -t rsa|dsa`, es poden crear les claus d'identificació d'usuari. L'ordre crea en el directori .shh de l'usuari el fitxer (per exemple, per a l'algoritme d'encriptació RSA) id_rsa i id_rsa.pub les claus privada i pública respectivament. L'usuari podria copiar la pública (id_rsa.pub) a la màquina remota en el directori .ssh de l'usuari remot, a l'arxiu authorized_keys. Aquest arxiu pot contenir tantes claus públiques com llocs des d'on es vulgui connectar aquesta màquina de manera remota. La sintaxi és d'una clau per línia i el seu funcionament és equivalent a l'arxiu .rhosts (encara que les línies tindran una mida considerable). Després d'haver introduït les claus públiques de l'usuari-màquina en aquest arxiu, aquest usuari i des d'aquesta màquina es podrà connectar sense *password*.

De manera normal (si no s'han creat les claus), es preguntarà a l'usuari un *password*, però com que la comunicació serà encriptada sempre, mai no serà accessible a altres usuaris que puguin escoltar sobre la xarxa. Per a més informació, consulteu `man ssh`. Per a executar remotament una ordre, simplement heu de fer:

```
ssh -l login name host_ordre_remota
```

Per exemple:

```
ssh -l user localhost ls -al
```

3.2.2. sshd

L'sshd és el servidor (*daemon*) per a l'ssh (es pot instal·lar si no ho estan amb `apt-get install ssh` la qual cosa instal·la el servidor i el client). Junts reemplacen el rlogin, telnet, rsh i proveeixen una comunicació segura i encriptada en dos *hosts* insegurs a la xarxa.

Aquest s'arrenca generalment per mitjà dels arxius d'inicialització (/etc/init.d o /etc/rc) i espera connexions dels clients. L'sshd de la majoria de les distribucions actuals suporta les versions 1 i 2 del protocol SSH. Quan s'instal·la el paquet, crea una clau RSA específica del *host*, i quan el *daemon* s'inicia, en crea una altra, l'RSA per a la sessió, que no s'emmagatzema al disc i la canvia cada hora. Quan un client inicia la comunicació, el client genera un nombre aleatori de 256 bits que és encriptat amb les dues claus del servidor i enviat. Aquest número s'utilitzarà durant la comunicació com a clau de sessió per a encriptar la comunicació que es farà per mitjà d'un algoritme d'encriptació estàndard.

L'usuari en pot seleccionar qualsevol dels disponibles oferts pel servidor. Hi ha algunes diferències (més segur) quan s'utilitza la versió 2 del protocol. A partir d'aquest moment, s'inicien alguns dels mètodes d'autenticació d'usuari descrits en el client o se li sol·licita el *password*, però sempre amb la comunicació encriptada. Per a més informació, consulteu `man sshd`.

3.2.3. Túnel sobre SSH

Moltes vegades tenim un accés a un servidor sshd, però per qüestions de seguretat no a altres serveis que no són encriptats (per exemple un servei de consulta de correu POP3 o un servidor de finestres X11) o simplement es vol connectar un servei al qual només es té accés des de l'entorn de l'empresa. Per això és possible establir un túnel encriptat entre la màquina client (per exemple amb Windows, i un client ssh anomenat *putty* de programari lliure) i el servidor amb sshd. En aquest cas, en vincular el túnel amb el servei, el servei entén la petició com si vingués de la mateixa màquina. Per exemple, si volem establir una connexió per a POP3 sobre el port 110 de la màquina remota (i que també té un servidor sshd) fem:

```
ssh -C -L 1100:localhost:110 usuari-id@host
```

Aquesta ordre demana el *password* per a l'usuari-id sobre *host*, i una vegada connectat s'haurà creat el túnel. Cada paquet que s'enviï a la màquina local sobre el port 1100 serà enviat a la màquina remota *localhost* sobre el port 110, que és on escolta el servei POP3 (l'opció -C comprimeix el trànsit pel túnel).

Fer túnels sobre altres ports és molt fàcil. Per exemple, suposem que només tenim accés a un *remote proxy server* des d'una màquina remota (*remote login*) –no des de la màquina local–, es pot fer un túnel per a connectar el navegador mitjançant el túnel a la màquina local. Considerem que tenim *login* sobre una màquina *`gateway`*, la qual pot accedir a la màquina anomenada *`proxy`*, que executa l'Squid proxy server sobre el port 3128. Executem:

```
ssh -C -L 8080:proxy:3128 user@gateway
```

Després de connectar-nos tindrem un túnel escoltant sobre el port local 8080, que reconduirà el trànsit des de *gateway* cap a *proxy* al 3128. Per a navegar de manera segura, només s'haurà de fer `http://localhost:8080/`.

4. Serveis de transferència de fitxers: FTP

L'FTP (*file transfer protocol*) és un protocol client/servidor (sota TCP) que permet la transferència d'arxius des de i cap a un sistema remot. Un servidor FTP és un ordinador que executa el *daemon* ftpd.

Alguns llocs que permeten la connexió anònima sota l'usuari *anonymous* són generalment dipòsits de programari. En un lloc privat, caldrà un usuari i un *password* per a accedir-hi. També és possible accedir a un servidor ftp mitjançant un navegador, i generalment avui en dia els dipòsits de programari són substituïts per servidors de web (p. ex. Apache) o altres tecnologies com Bittorrent (que utilitza xarxes punt a punt –P2P–). No obstant això, es continua utilitzant en alguns casos i Debian, per exemple, accés amb usuari o passwd o la possibilitat de pujar arxius al servidor (si bé amb serveis web també és possible fer-ho). El protocol (i servidors/clients que l'implementen) d'ftp per definició no són encriptats (les dades, usuaris i passwds es transmeten en text clar per la xarxa) amb el risc que això suposa. Però hi ha una sèrie de servidors/clients que suporten SSL i, per tant, encriptació.

4.1. Client ftp (convencional)

Un client ftp permet accedir a servidors FTP i hi ha una gran quantitat de clients disponibles. La utilització de l'ftp és summament simple, des de la línia d'ordre, cal executar:

ftp *nom-servidor*

O també ftp, i després en forma interactiva:

open *nom-servidor*

El servidor sol·licita un *username* i un *password* (si accepta usuaris anònims, s'hi introduirà *anonymous* com a usuari i la nostra adreça d'e-mail com a *password*) i a partir del prompt de l'ordre (després d'alguns missatges), podrem començar a transferir fitxers.

El protocol permet transferència en forma ASCII o binaris. És important decidir el tipus de fitxer que cal transferir perquè una transferència d'un binari en forma ASCII inutilitzarà el fitxer. Per a canviar d'una forma a l'altra, s'ha d'executar l'ordre *ascii* o *binary*. Ordres útils del client ftp són la *ls* (navegació en el directori remot), *get* nom_del_fitxer (per a descarregar fitxers) o *mget* (que admet *),

put nom_del_fitxer (per a enviar fitxers al servidor) o *mput* (que admet *); en aquests dos últims s'ha de tenir permís d'escriptura sobre el directori del servidor. Es poden executar ordres locals si abans de l'ordre s'hi insereix un '!'. Per exemple !cd /tmp significa que els arxius que baixin a la màquina local es descarregaran en /tmp. Per a poder veure l'estat i el funcionament de la transferència, el client pot imprimir marques, o *ticks*, que s'activa amb les ordres *hash* i *tick*. Hi ha altres ordres que es poden consultar en el full del manual (man ftp) o en fer *help* dins del client.

Disposem de nombroses alternatives per als clients, per exemple en forma de text: ncftp, lukemftp, lftp, cftp, yafc Yafc, o en forma gràfica: gFTP, WXftp, LLNL XFTP, guiftp. [Bor00]

4.2. Servidors FTP

El servidor tradicional de UNIX s'executa mitjançant el port 21 i el posa en marxa el *daemon* inetd (o xinetd segons com es tingui instal·lat). En inetd.conf és convenient incloure el wrapper tcpd amb les regles d'accés en host.allow i el host.deny en la crida al ftpd per l'inetd per a incrementar la seguretat del sistema (consulteu el capítol dedicat a la seguretat). Quan rep una connexió, verifica l'usuari i el *password* i el deixa entrar si l'autenticació és correcta. Un FTP *anonymous* treballa de manera diferent, ja que l'usuari només pot accedir a un directori definit a l'arxiu de configuració i a l'arbre subjacent, però no cap a dalt, per motius de seguretat. Aquest directori generalment conté directoris pub/, bin/, etc/, i lib/ perquè el *daemon* de ftp pugui executar ordres externes per a peticions de ls. El *daemon* ftpd suporta els arxius següents per a la seva configuració:

- /etc/ftpusers: llista d'usuaris que no són acceptats sobre el sistema, un usuari per línia.
- /etc/ftpchroot: llista d'usuaris a qui se'ls canviarà el directori base chroot quan es connectin. Necessari quan volem configurar un servidor anònim.
- /etc/ftpwelcome: anunci de benvinguda.
- /etc/motd: notícies després del *login*.
- /etc/nologin: missatge que es mostra després de negar la connexió.
- /var/log/ftpd: *log* de les transferències.

Si en algun moment volem inhibir la connexió a l'ftp, es pot fer incloent-hi l'arxiu /etc/nologin. L'ftpd mostra el seu contingut i acaba. Si hi ha un arxiu .message en un directori, l'ftpd ho mostrarà quan s'hi accedeixi.

La connexió d'un usuari passa per cinc nivells diferents:

1) Tenir una contrasenya vàlida.
2) No sortir en la llista de /etc/ftpusers.
3) Tenir un intèrpret de dades estàndard vàlid.
4) Si surt en /etc/ftpchroot, es canviarà al directori home (inclòs si és *anonymous* o ftp).
5) Si l'usuari és *anonymous* o ftp, haurà de tenir una entrada en el /etc/passwd amb user ftp, però s'hi podrà connectar especificant qualsevol passwd (per convenció s'utilitza l'adreça d'e-mail).

És important tenir en compte que els usuaris que només estiguin habilitats per a utilitzar el servei ftp no disposen d'un intèrpret de dades a l'entrada corresponent d'aquest usuari en /etc/passwd per a impedir que aquest usuari tingui connexió, per exemple, per ssh o telnet. Per això, quan es creï l'usuari, caldrà indicar, per exemple:

```
useradd -d/home/nteum -s /bin/false nteum
```

I després:

```
passwd nteum
```

La qual cosa indica que l'usuari nteum no té intèrpret de dades per a una connexió interactiva (si l'usuari ja existeix, es pot editar el fitxer /etc/passwd i canviar-hi l'últim camp per /bin/false). Després s'ha d'agregar com a última línia /bin/false en /ect/shells. En [Mou01] es descriu pas a pas com crear tant un servidor ftp segur amb usuaris registrats com un servidor ftp *anonymous* per a usuaris no registrats. Dos dels servidors no estàndards més comuns són el WUFTPD (http://www.wuftpd.org) i el ProFTPD (http://www.proftpd.org). [Bor00, Mou01]

Per a instal·lar el Proftpd sobre Debian cal executar: `apt-get install proftpd`. Un cop descarregat, debconf us preguntarà si el voleu executar per inetd o de forma manual (és recomanable elegir l'última). Si es vol parar el servei (per a canviar la configuració per exemple) `/etc/init.d/proftpd stop`, i per a modificar el fitxer, `/etc/proftpd.conf`.

Consulteu http://www.debian-administration.org/articles/228 per a configurar-lo de forma encriptada (TSL) o per a tenir-hi accés anonymous.

Un servidor (Debian) que és molt interessant és el PureFtpd (pure-ftpd) que és molt segur, permet usuaris virtuals, quotes, SSL/TSL i un conjunt de característiques interessants. La seva instal·lació/configuració es pot consultar a http://www.debian-administration.org/articles/383.

5. Serveis d'intercanvi d'informació en l'àmbit d'usuari

5.1. El *mail transport agent* (MTA)

Un MTA (*mail trasnport agent*) s'encarrega d'enviar/rebre els correus des d'un servidor de correu electrònic cap a/des d'Internet, que implementa el protocol SMTP (*simple mail transfer protocol*). Debian utilitza per defecte exim, ja que és més fàcil de configurar que altres paquets MTA, com smail o sendmail (aquest últim és un dels precursors). exim presenta característiques avançades com rebutjar connexions de llocs d'SPAM coneguts, té defenses contra correus brossa o bombardejos i és extremadament eficient en el processament de grans quantitats de correus. La seva execució es fa mitjançant inetd en una línia a l'arxiu de configuració /etc/inetd.conf amb paràmetres per a configuracions normals (o xinetd).

exim utilitza un arxiu de configuració en /etc/exim/exim.conf, que es pot modificar manualment, però és recomanable fer-ho amb un *shel lscript* anomenat `eximconfig`, per a poder configurar exim de forma interactiva. Els valors de la configuració depenen de la situació de la màquina; tanmateix, la seva connexió és summament fàcil, ja que el mateix *script* suggereix valors per defecte. No obstant això, en /usr/doc/exim es poden trobar exemples de configuració típiques.

Es pot provar si la configuració és vàlida amb `exim -bV` i, si hi ha errors a l'arxiu de configuració, el programa els mostrarà per pantalla o, si tot és correcte, només hi posarà la versió i la data. Per a provar si pot reconèixer una bústia (*mailbox*) local, utilitzeu:

```
exim -v -bt usuari_local
```

En què es mostraran les capes de transport utilitzades i l'adreça local de l'usuari. També es pot fer el test següent amb un usuari remot reemplaçant usuari local per una adreça remota per a veure el seu comportament. Després intenteu enviar un correu localment i remotament, passant directament els missatges a exim (sense utilitzar un agent per exemple, mailx), teclejant per exemple (tot junt):

```
exim postmaster@elseuDomini
From: user@domini
To: postmaster@ elseuDomini
Subject: Test Exim
Missatge de prova
^D
```

A continuació, es poden analitzar els arxius de traça mainlog i paniclog en /var/log/exim/ per a veure el seu comportament i quins són els missatges

d'error generats. Òbviament, també us podeu connectar al sistema com l'usuari *postmaster* (o a qui s'hagi enviat el correu) i llegir els correus per a veure si tot és correcte. L'altra forma consisteix a executar-lo de forma *debug* utilitzant com a paràmetre -dNro, en què Nro és el nivell de *debug* (19). El paràmetre normal amb el qual s'ha de posar en marxa és `exim -bs`, ja sigui per inetd o per xinetd. També és possible executar-lo com a *daemon* per mitjà de /etc/init.d/exim start en sistemes que necessitin prestacions elevades al tractament dels correus. Consulteu la documentació (inclosa a Debian el paquet exim-doc-html) per a configurar filtres, verificació de *hosts*, de *sender*, etc. És interessant també instal·lar el paquet eximon, que és un monitor de l'exim i permet que l'administrador vegi la cua de correus, *logs* i que faci diferents accions amb els missatges en cua per ser distribuïts (*freezing*, *bouncing*, *thawing*...).

L'última versió d'Exim és Exim4 (es pot instal·lar amb `apt-get install exim4-daemon-heavy` [i també instal·lar exim4-config que servirà per a configurar exim4]; cal tenir en compte que hi ha diferents paquets amb diferents possibilitats però exim4-daemon-heavy és el més complet). És recomanable llegir `/usr/share/doc/exim/README.Debian.gz` and `update-exim4.conf(8)`. Per a més informació es pot consultar el HowTo http://www.exim.org/docs.html. Unes petites diferències que cal tenir en compte en la configuració és que en lloc de tenir una única configuració `exim.conf` (que és el que tindrà si s'instal·la exim des dels font) el paquet exim4-config (és convenient instal·lar-lo per a configurar exim4) utilitza petits arxius de configuració en lloc d'un únic i que seran en `/etc/exim4/conf.d/*` i seran concatenats tots en un únic arxiu (`/var/lib/exim4/config.autogenerated` per defecte) per update-exim4.conf.

5.2. *Internet message access protocol* (IMAP)

Aquest servei permet accedir als correus allotjats en un servidor per mitjà d'un client de correu com per exemple Thunderbird o el client de correu de Seamonkey (ambdós en mozilla.org). Aquest servei suportat pel *daemon imapd* (els actuals suporten el protocol IMAP4rev1) permet un arxiu de correu electrònic (*mail file*) que es troba en una màquina remota. El servei imapd es presta per mitjà dels ports 143 (*imap2*) o 993 (*imaps*) quan suporta encriptació per SSL. Si s'utilitza inetd, aquest servidor es posa en marxa mitjançant una línia en /etc/inetd.conf com:

```
imap2 stream tcp nowait root /usr/sbin/tcpd /usr/sbin/imapd
imap3 stream tcp nowait root /usr/sbin/tcpd /usr/sbin/imapd
```

En aquest exemple es crida el wrapper tcpd que funciona amb hosts.allow i hosts.deny per a incrementar la seguretat. Les aplicacions més populars són uw-imapd (Universitat de Washington i instal·lat per defecte a Debian) o la seva versió segura uw-imapd-ssl, cyrus-imap o courier-imap. Per a provar que el servidor imap funciona, es podria utilitzar un client, per exemple, `seamonkey -mail`

i crear un compte per a un usuari local i configurar-lo adequadament perquè es connecti sobre la màquina local, cosa que verifica el funcionament d'imap.

Sobre Debian, la versió d'imap ha estat compilada per a suportar MD5 com a mètode d'autenticació dels usuaris remots, per a encriptar els *passwords* de connexió i evitar suplantació d'identitat per *sniffing* a la xarxa (el client utilitzat per a connectar-se al servidor imap també ha de suportar el mètode d'autenticació per MD5). El mètode és molt simple i segur, però el servidor ha de conèixer els *passwords* en text pla dels usuaris de correu, per la qual cosa es recomana utilitzar la versió d'imapd sobre SSL i que funciona sobre el port 993. El protocol imaps que igual que l'ssh es basa a encriptar la comunicació mitjançant un certificat del *host* (el client utilitzat per a connectar-se al servidor també ha de suportar aquest mètode de connexió per exemple, thunderbird o seamonkey -mail). Per a configurar el servidor imaps, instal·leu-hi el paquet uw-imap-dssl de Debian que és el servidor imap amb suport SSL.

La instal·lació genera un certificat autofirmat vàlid per un any i l'emmagatzema en /etc/ssl/certs/imapd.pem. Aquest certificat es pot reemplaçar per un de firmat per una companyia certificadora o se'n pot generar un de propi amb OpenSSL. És convenient deixar només l'entrada imaps en l'arxiu /etc/inetd.conf i treure les entrades imap2 i imap3 si únicament es vol que l'accés a imap sigui per SSL.

Un altre protocol de característiques similars que ha tingut molta popularitat en el passat, però que avui s'ha vist superat per IMAP, és POP (*post office protocol*) versió 2 i 3. La seva instal·lació i posada en marxa és anàloga a la d'IMAP. Hi ha multitud de servidors POP, però els més comuns són courier-pop, cyrus-pop3d, ipopd (Universitat de Washington), qpopper, solid-pop3d.

5.2.1. Aspectes complementaris

Suposem que com a usuaris tenim quatre comptes de correus en servidors diferents i volem que tots els correus electrònics que arriben a aquests comptes es recullin en un únic compte, que hi puguem accedir externament i que hi hagi un filtre de correu brossa (*antispam*) també.

Primer s'ha d'instal·lar Exim + Imap i comprovar que funcionen. S'ha de tenir en compte que si s'hi instal·la courier-imap (que segons alguns autors és millor que uw-imapd) aquest funciona sobre un format de correu anomenat Maildir, que s'hauria de configurar Exim perquè també funcioni sobre maildir amb la configuració següent en /etc/exim/exim.conf (o en la corresponent si es té exim4), i canviar-hi l'opció mail_dir format = true (els correus es guardaran en el compte de l'usuari local en un directori anomenat Maildir). Després s'ha de reiniciar el servidor exim amb `/etc/init.d/exim restart`, repetir la prova de funcionament

enviant-nos un correu i llegir-lo amb un client que suporti maildir (per exemple `mutt -mailx` el suporta– vegeu http://www.mutt.org).

Per a recollir els correus de diferents comptes s'utilitzarà fetchmail, (que s'instal·la amb `apt-get install fetchmail`). S'ha de crear a continuació el fitxer .fetchmailrc en el nostre $HOME (també es pot utilitzar l'eina fetchmailconf) que haurà de tenir alguna cosa semblant a:

```
set postmaster "pirulo"
set bouncemail
set no spambounce
set flush

poll pop.domain.com proto pop3
user 'user1' there with contrasenya 'secret' is pirulo here

poll mail.domain2.com
   user 'user5' there with contrasenya 'secret2' is 'pirulo' here
   user 'user7' there with contrasenya 'secret3' is 'pirulo' here
```

L'acció `set` indica a fetchmail que aquesta línia conté una opció global (tramesa d'errors, esborrar els correus dels servidors...). A continuació, s'especifiquen dos servidors de correu: un perquè comprovi si hi ha correu amb el protocol POP3 i un altre perquè provi d'usar diversos protocols per a trobar-ne un que funcioni. Es comprova el correu de dos usuaris amb la segona opció de servidor, però tot el correu que es trobi s'envia a l'*spool* de correu del pirulo. Això permet comprovar diverses bústies de diversos servidors com si es tractés d'una única bústia MUA. La informació específica de cada usuari comença amb l'acció `user`. El fetchmail es pot posar en el cron (per exemple en `/var/spool/cron/crontabs/pirulo agregant 1 **** /usr/bin/fetchmail -s`), perquè s'executi automàticament o executar-los de forma *daemon* (posar set daemon 60 en .fetchmailrc i executar-lo una vegada per exemple en Autostart de Gnome/KDE o en el .bashrc –s'executarà cada 60 segons).

Per a treure el correu brossa s'utilitzarà SpamAssassin (`apt-get install spamassassin`) i es pot configurar Kmail o Evolution (vegeu bibliografia per a consultar com configurar-lo) perquè l'executin. En aquesta configuració s'utilitzarà Procmail que és una eina molt potent (permet repartir el correu, filtrar-lo, reexpedir-lo automàticament...). Una vegada instal·lat (`apt-get install procmail`), s'ha de crear un fitxer anomenat .procmailrc en el home de cada usuari que cridarà el Spamassassin:

- Poseu *yes* per a missatges de funcionament o depuració

  VERBOSE=no

- Considerem que els correus són en "~/.Maildir"), canviar si és un altre PATH=/usr/bin:/bin:/usr/local/bin:.

  MAILDIR=$HOME/Maildir

DEFAULT=$MAILDIR/
– Directori per a emmagatzemar els fitxers
PMDIR=$HOME/.procmail
– Comentar si no volem *log* de Procmail
LOGFILE=$PMDIR/log
– filtre d'Smap
INCLUDERC=$PMDIR/spam.rc

L'arxiu ~/.procmail/spam.rc conté:

- Si el spamassassin no hi és en el PATH, cal agregar a la variable PATH el directori:

  0fw: spamassassin.lock
  | spamassassin -a

- Les tres línies següents mouran el correu Spam a un directori anomenat "spam-folder". Si es vol desar a la safata d'entrada, per a després filtrar-lo amb el client, cal comentar les tres línies.

  :0:
  * ^X-Spam-Estatus: Yes
  spam-folder

L'arxiu ~/.spamassassin/user_prefs conté alguna configuració útils per a spamassassin (consulteu la bibliografia):

- user preferences file. Vegeu man Mail::SpamAssassin::Conf

- Llindar per a reconèixer un Spam: Default 5; però amb 4 funciona una mica millor:
  required_hits 4

- Llocs dels quals considerarem que mai no arribarà Spam:
  whitelist_from root@debian.org
  whitelist_from *@uoc.edu

- Llocs dels quals sempre arriba SPAM (separat per comes):
  blacklist_from viagra@dominio.com

- adreces a Whitelist i blacklist són patrons globals com:
  "amic@lloc.com", "*@isp.net", o "*.domain.com".

- Inseriu la paraula "[SPAM]" en el subject (facilita fer filtres).

- Si no es vol comentar la línia:
  subject_tag [SPAM]

Això generarà un tag X-Spam-Status: *Yes* a la capçalera del missatge si creu que el missatge és Spam. Després s'hauran de filtrar i posar-los en una altra carpeta o esborrar-los directament. Es pot utilitzar el procmail per a filtrar correus de dominis, usuaris etc. Per a més informació consulteu la pàgina d'Internet http://www.debian-administration.org/articles/242. Finalment es pot instal·lar un client de correu i configurar-hi els filtres perquè seleccioni tots els correus amb X-Spam-Status: *Yes* i els esborri o els enviï a un directori en el qual després verificarem els falsos positius (correus identificats com a brossa però que no ho són). Un aspecte complementari en aquesta instal·lació és si es vol tenir un servidor de correu per mitjà de webmail (és a dir, poder consultar els correus del servidor amb un navegador sense haver d'instal·lar un client ni configurar-lo –de la mateixa manera que consultar un compte de gmail o hotmail) és possible instal·lar Squirrelmail (`apt-get install squirrelmail`) per a prestar aquest servei. Per a Debian consulteu: http://www.debian-administration.org/articles/200.

Hi ha una altra possibilitat com es comenta en l'article electrònic http://www.debian-administration.org/articles/364 si s'hi instal·la MailDrop en lloc de Procmail, Postfix en lloc d'Exim, o Clamav/Amavisd com a antivirus (Amavisd permet vincular postfix amb spamassassin i clamav).

5.3. *News*

Les *news* o grups de discussió són suportats pel protocol NNTP. Instal·lar un servidor de *news* és necessari quan es volen llegir *news* fora de línia, quan es vol tenir un repetidor dels servidors centrals o es vol un propi servidor *master* de *news*. Els servidors més comuns són INN o CNEWS, però són paquets complexos i destinats a grans servidors. Leafnode és un paquet USENET que implementa servidor TNP, especialment indicat per a llocs amb grups reduïts d'usuaris, però en els quals es vol accedir a gran quantitat de grups de notícies. Aquest servidor s'instal·la en la configuració bàsica de Debian i es pot reconfigurar amb `dpkg-reconfigure leafnode`, tots paràmetres com els servidors centrals, el tipus de connexió, etc. Aquest *daemon* es posa en marxa des d'inetd de manera similar a l'imap (o amb xinetd). Leafnode suporta filtres mitjançant expressions regulars indicades (del tipus ^Newsgroups:. * [,] alt.flame$) en /etc/news/leafnode/filters, en què per a cada missatge es compara la capçalera amb l'expressió regular i, si hi ha coincidència, el missatge és rebutjat.

La configuració d'aquest servidor és simple i tots els arxius han de ser propietat d'un usuari *news* i amb permís d'escriptura (cal verificar que aquest propietari és a /etc/passwd). Tots els arxius de control, *news* i configuració es troben en /var/spool/news excepte la configuració del propi servidor que hi ha en el fitxer /etc/news/leafnode/config. En la configuració hi ha alguns paràmetres obligatoris que han de ser configurats (per exemple, perquè el servidor es pugui connectar amb els servidors mestres). Són *server* (servidor de *news* des d'on s'obtenen i envien les *news*) i *expire* (nombre de dies que un *thread*

o sessió ha estat llegida i s'esborrarà). Tenim, així mateix, un conjunt de paràmetres opcionals d'àmbit general o específics del servidor que es podrien configurar. Per a més informació, consulteu la documentació (*man leafnode* o /usr/doc/leafnode/README.Debian).

Per a verificar el funcionament del servidor, es pot fer:

```
telnet localhost nntp
```

i si tot funciona correctament, sortirà la identificació del servidor i es quedarà esperant una ordre, com a prova, es pot introduir *help* (per a avortar, feu Ctrl+ i després Quit).

5.4. World Wide Web (httpd)

Apache és un dels servidors més populars i d'altes prestacions d'HTTP (*hypertext transfer protocol*). Apache té un disseny modular i suporta extensions dinàmiques de mòduls durant la seva execució. És altament configurable en el nombre de servidors i de mòduls disponibles i suporta diversos mecanismes d'autenticació, control d'accés, *metafiles*, *proxy caching*, servidors virtuals, etc. Amb mòduls (inclosos a Debian) és possible tenir PHP3, Perl, Java Servlets, SSL i altres extensions (podeu consultar la documentació en http://www.apache.org).

Apache està dissenyat per a executar-se com un procés *daemon standalone*. En aquesta forma crea un conjunt de processos fill que s'encarregaran de les peticions d'entrada. També es pot executar com a Internet *daemon* per mitjà d'inetd, per la qual cosa es posarà en marxa cada vegada que es rebi una petició. La configuració del servidor pot ser extremadament complexa segons les necessitats (consulteu la documentació), tanmateix, aquí veurem una configuració mínima acceptable. Els arxius de configuració es troben en /etc/apache i són httpd.conf (arxiu principal de configuració), srm.conf, access.conf (aquests dos últims són mantinguts per compatibilitat i la seva funcionalitat és en l'anterior), mime.conf (formats MIME) i magic (número d'identificació d'arxius). Els arxius *log* són a /var/log/apache i són error.log (registra els errors en les peticions del servidor), access.log (registre de qui i a què ha accedit) i apache.pid (identificador del procés).

Apache es posa en marxa des de l'*script* d'inici /etc/init.d/apache i els /etc/rcX.d, però es pot controlar manualment mitjançant l'ordre `apachectl`. També es pot utilitzar l'ordre apacheconfig per a configurar el servidor. Els directoris per defecte (a Debian) són:

- /var/www: directori de documents HTML.
- /usr/lib/cgibin: directori d'executables (*cgi*) pel servidor.
- http://server.domini~/user: pàgines personals dels usuaris.
- /home/user/public.html: directori de pàgines personals.

L'arxiu per defecte que es llegeix de cada directori és index.html. Una vegada instal·lats els paquets *apache* i *apache-common*, Debian configura bàsicament el servidor i el posa en marxa. Es pot comprovar que funciona obrint un *browser* (per exemple, el Konqueror, i posant a la barra d'URL http://localhost, la qual cosa carrega la pàgina /var/www/index.html).

5.4.1. Configuració manual (mínima) d'httpd.conf

Veurem alguns dels paràmetres més importants en la configuració d'Apache (l'exemple es pren de la versió 1.X d'Apache i hi ha alguns petits canvis si s'utilitza la versió 2).

ServerType standalone	Recomanat, més eficient
ServerRoot /etc/apache	On són els arxius de configuració
Port 80	On el servidor escolta les peticions
User www-data	*User* i *group* amb els quals s'executa el servidor (important per seguretat) han de ser usuaris vàlids (poden ser *locked*)
Group www-data ServerAdmin webmaster@pirulo.remix.com	Adreça d'usuari que atendrà els errors
ServerName pirulo.remix.com	Nom del servidor enviat als usuaris –ha de ser un nom vàlid en /etc/host o DNS–
DocumentRoot /var/www	Directori on hi haurà els documents
Alias /icons/ /usr/share/apache/icons/	On es troben les icones
ScriptAlias /cgibin/ /usr/lib/cgibin/	On es troben els *script* CGI

5.4.2. Apache 2.2 + SSL + PHP + MySQL

Un aspecte important per a servidors web dinàmics és aprofitar els avantatges d'Apache en forma segura (SSL), PHP (és un llenguatge de programació usat generalment per a la creació de contingut per a llocs web) i MySQL+PHPAdmin (base de dades de què parlarem en pròxims capítols i interfície gràfica per a la seva gestió) tot això funcionant conjuntament. Partirem de la base d'instal·lar-lo sobre un Debian Sarge, però no mitjançamt paquets deb, sinó des del programari baixat dels llocs respectius, així es pot repetir l'experiència sobre altres distribucions. Òbviament aquests paquets després no es poden controlats per apt o un altre gestor de paquets. S'ha d'anar amb compte amb les versions que poden canviar i amb no superposar la instal·lació a paquets ja instal·lats.

a) Descàrrega dels fitxers necessaris (per exemple dins del directori /root -> cd /root):

1) Apache: des de http://httpd.apache.org/download.cgi: httpd-2.2.4.tar.bz2

2) PHP: des de http://www.php.net/downloads.php PHP 5.2.1 (tar.bz2)

3) MySQL des de http://mysql.org/get/Downloads/MySQL-4.1/mysql-standard-4.1.21-pc-linux-gnu-i686.tar.gz/from/pick

4) PHPAdmin des de http://prdownloads.sourceforge.net/phpmyadmin/phpMyAdmin-2.9.1-all-languages.tar.bz2?download

b) Utilitats: bzip2 libssl-dev openssl gcc g++ cpp make (verifiqueu que no es tenen instal·lades o si no, feu apt-get install bzip2 libssl-dev openssl gcc g++ cpp make).

c) Apache:

```
cd /root
tar jxvf httpd-2.2.4.tar.bz2
cd httpd-2.2.4
```

Amb prefix, indiquem que s'instal·larà per exemple `/usr/local/apache2`

```
. /configuri --prefix=/usr/local/apache2 –with ssl=/usr/include/openssl\
--enable-ssl
make
make install
```

Modifiquem el fitxer de configuració /usr/local/apache2/conf/httpd.conf i canviem l'usuari i grup de treball per www-data

```
User www-data
Group www-data
```

Canviem el propietari i grup del directori de dades a www-data:

```
chown -R www-data:www-data /usr/local/apache2/htdocs
```

Modifiquem l'usuari www-data per a canviar el seu directori *home* en /etc/passwd:

```
www-data:x:33:33:www-data:/usr/local/apache2/htdocs:/bin/sh
```

Servidor apache instal·lat. Per a iniciar-lo (per a parar-lo canviar *start* per *stop*):

```
/usr/local/apache2/bin/apachectl start
```

Es pot col·locar un *script* per a arrencar el servidor apache en *boot*:

```
ln -s /usr/local/apache2/bin/apachectl /etc/rcS.d/S99apache
chmod 755 /etc/rcS.d/S99apache
```

d) SSL:

En /usr/local/apache2/conf/httpd.conf traiem el comentari de la línia:

```
Include conf/extra/httpd-ssl.conf
```

Es generen els fitxers amb les claus per al servidor segur, en /root fem (cal adequar les versions a les que s'hagin descarregat) –la primera ordre openssl és una línia sencera i acaba amb 1024:

```
openssl genrsa -rand ../httpd-2.2.4.tar.bz2:../php-5.2.1.tar.bz2:../
phpMyAdmin-2.9.1-all-languages.tar.bz2 -out server.key 1024
openssl rsa -in server.key -out server.pem
openssl req -new -key server.key -out server.csr
openssl x509 -req -days 720 -in server.csr -signkey server.key -out
server.crt
```

Es copien els fitxers...

```
cp server.crt /usr/local/apache2/conf/
cp server.key /usr/local/apache2/conf/
```

Reiniciem el servidor...

```
/usr/local/apache2/bin/apachectl restart
```

Es pot consultar com agregar el mòdul SSL a un servidor que no el tingui instal·lat en http://www.debian-administration.org/articles/349.

e) MySQL (per a més informació vegeu el mòdul 8):

Creem un grup i un usuari per a MySQL si no existeix:

```
groupadd mysql
useradd -g mysql mysql
```

En el directori en el qual s'instal·larà MySQL (/usr/local/) fem:

```
cd /usr/local/
gunzip < /root/mysql-standard-4.1.21-pc-linux-gnu-i686.tar.gz | tar xvf -
ln -s mysql-standard-4.1.21-pc-linux-gnu-i686 mysql
cd mysql
```

Creo una base de dades i canvio els permisos:

```
scripts/mysql_install_db --user=mysql
chown -R root.
chown -R mysql data
chgrp -R mysql.
```

Es pot col·locar un *script* per a iniciar el servidor mysql:

```
ln -s /usr/local/mysql/support-files/mysql.server /etc/rcS.d/S99mysql.server
chmod 755 /etc/rcS.d/S99mysql.server
```

Iniciem el servidor:

```
/etc/rcS.d/S99mysql.server start
```

Es pot entrar en la BD i canviar el *password* del *root* per seguretat (consulteu http://dev.mysql.com/doc/refman/5.0/en/index.html per a la sintaxi):

```
/usr/local/mysql/bin/mysql
```

Dins fem:

USE mysql

Col·loquem el *password* pirulo a l'usuari *root*

UPDATE user SET Password=PASSWORD('pirulo') WHERE User='root';
FLUSH privileges;

Per a entrar en MySQL haurem de fer

```
/usr/local/mysql/bin/mysql -o root -ppirulo
```

f) PHP (reemplaçar amb les versions adequades):

Utilitats necessàries:

```
apt-get install libxml2-dev curl libcurl3-dev libjpeg-mmx-dev zlib1g-dev \ libpng12-dev
```

Amb el servidor Apache aturat fem:

```
cd /root
tar jxvf php-5.2.0.tar.bz2
cd php-5.2.0
```

Amb prefix es pot indicar on es vol instal·lar (tot en una línia):

```
. /configure --prefix=/usr/local/php5 --enable-mbstring --with-apxs2=/
usr/local/apache2/bin/apxs --with-mysql=/usr/local/mysql --with-curl=/
usr/include/curl --with-jpeg-dir=/usr/include --with-zlib-dir=/usr/
include --with-gd --with-xml --enable-ftp --enable-bcmath

make
make install
cp php.ini-dist /usr/local/php5/lib/php.ini
```

Modifiquem Apache (/usr/local/apache2/conf/httpd.conf) en la part indicada:

<IfModule mime_module>
AddType application/x-httpd-php .php .phtml
AddType application/x-httpd-php-source .phps

I també:

DirectoryIndex index.php index.html

Reiniciem el servidor.

g) PHPAdmin

```
cd /usr/local/apache2/
```

Es descomprimeix phpmyadmin en el directori d'apache2 (atenció amb les versions):

```
tar jxvf /root/phpMyAdmin-2.9.1-all-languages.tar.bz2
mv phpMyAdmin-2.9.1-all-languages phpmyadmin
cd phpmyadmin
cp config.sample.inc.php config.inc.php
```

S'ha de modificar el fitxer de configuració (config.inc.php):

$cfg['blowfish_secret'] = 'pirulo';

Trec l'usuari i *password* de l'usuari per defecte dos (') seguides:

```
$cfg['Servers'][$i]['controluser'] = '';
$cfg['Servers'][$i]['controlpass'] = '';
```

Canvio apache (/usr/local/apache2/conf/httpd.conf) afegint en
< IfModule alias_module >:

```
<IfModule alias_module>
        Alias /phpmyadmin "/usr/local/apache2/phpmyadmin/"
<Directory "/usr/local/apache2/phpmyadmin/">
        Order allow,deny
        Allow from all
</Directory>
```

Reiniciem el servidor i es pot cridar amb http://localhost/phpadmin

Es pot tenir més informació en els webs respectius de cada aplicació i en LWP.

6. Servei de *proxy*: Squid

Un servidor *proxy* (PS) s'utilitza per a salvar amplada de banda de la connexió de xarxa, millorar la seguretat i incrementar la velocitat per a obtenir pàgines de la Xarxa (*web-surfing*).

Squid és un dels principals PS, ja que és OpenSource, accepta ICP (característiques que li permeten intercanviar *hints* amb d'altres PS), SSL (per a connexions segures entre *proxies*) i suporta objectes FTP, Gopher, HTTP i HTTPS (segur). El seu funcionament és simple, emmagatzema els objectes més sol·licitats en memòria RAM i els menys sol·licitats en una base de dades al disc. Els servidors Squid, a més, es poden configurar de manera jeràrquica per a formar un arbre de *proxies* depenent de les necessitats. Hi ha dues configuracions possibles:

1) Com a accelerador d'httpd per a aconseguir més prestacions al servei de web.

2) Com a *proxy-caching server* per a permetre que els usuaris d'una corporació utilitzin el PS per a sortir cap a Internet.

En la primera manera, actua com a *proxy* invers, és a dir, accepta una petició del client, serveix l'objecte si el té i si no, el sol·licita i el passa al client quan el té, i l'emmagatzema per a la vegada següent. En la segona opció es pot utilitzar com a control i per a restringir els llocs en els quals es pot connectar a Internet o autoritzar l'accés a determinades hores del dia. Una vegada instal·lat (paquet squid en Debian, també es pot instal·lar squid-cgi, squidguard o squidtaild) es generen tres arxius: /etc/squid.conf (configuració), /etc/init.d/squid (inicialització) i /etc/logrotate.d/squid (de control dels *logs*).

6.1. Squid com a accelerador d'http

En aquesta manera, si el servidor de web és a la mateixa màquina en la qual és el PS, s'haurà de reconfigurar perquè atengui peticions del port 81 (en Apache, canviar Port 80 per Port 81 en httpd.conf). L'arxiu de configuració (/etc/squid.conf) conté una gran quantitat d'entrades, però aquí només veurem les indispensables [Mou01]:

```
http_port 80                                  On escolta httpd
icp_port 0                                    On escolta ICP
hierarchy_stoplist cgi-bin \?
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY
cache_mem 100 MB                              Memòria per a objectes en curs
```

```
redirect_rewrites_host_header off
cache_replacement_policy lru
memory_replacement_policy lru
cache_dir ufs /var/spool/squid 100 16 256        Tipus i lloc on hi ha la base de dades de
                                                 cau de disc
emulate_httpd_log on
acl all src 0.0.0.0/0.0.0.0                      Accés per a tots
http_access allow all                            I a tot
cache_mgr root                                   Correu responsable
cache_effective_user proxy                       UID
cache_effective_group proxy                      GID
httpd_accel_host 192.168.1.1                     Servidor real de web
httpd_accel_port 81                              Port
logfile_rotate 0
log_icp_queries off
buffered_logs on
```

D'aquesta manera, l'opció httpd_accel_host desactiva la possibilitat que s'executi com a *proxy-caching*. Per a més informació consulteu http://www.squid-cache.org/.

6.2. Squid com a *proxy-caching*

D'aquesta manera s'habilita l'squid perquè controli l'accés a Internet, quan s'hi accedirà i a què s'accedirà. En aquest cas, l'arxiu de configuració haurà d'incloure les modificacions/agregats següents en /etc/squid.conf:

```
acl localnet src 192.168.1.0/255.255.255.0
acl localhost src 127.0.0.1/255.255.255.255
acl Safe_ports port 80 443 210 70 21 102565535
acl CONNECT method CONNECT
acl all src 0.0.0.0/0.0.0.0
http_access allow localnet
http_access allow localhost
http_access deny
http_access deny CONNECT
http_access deny all
cache_emulate_httpd_log on
```

La gran diferència amb l'altra opció són les línies acl, en aquest cas es permetrà que els clients de la classe C 192.168.1.0 accedeixin al PS, també el *localhost* IP i altres ports que podran accedir a Internet 80 (http), 443 (https), 210 (whois), 70 (gopher), i 21(ftp), a més, es nega el mètode *connect* per a evitar que des de fora es puguin connectar al PS i després es neguen tots els IP i ports sobre el PS. [Mou01] Més informació a http://www.squid-cache.org/ i per a un *transparent-proxy* a http://tldp.org/HOWTO/TransparentProxy-1.html.

7. OpenLdap (Ldap)

LDAP significa *lightweight directory access protocol* i és un protocol per a accedir a dades basades en un servei X.500. Aquest s'executa sobre TCP/IP i el directori és similar a una base de dades que conté informació basada en atributs. El sistema permet organitzar aquesta informació de manera segura i utilitzar rèpliques per a mantenir la seva disponibilitat, de manera que assegura la coherència i la verificació de les dades accedides-modificades.

El servei es basa en el model client-servidor, en què hi ha un servidor o més d'un que conté les dades; quan un client s'hi connecta i sol·licita informació, el servidor respon amb les dades o un punter a un altre servidor en el qual podrà extreure més informació, però el client només veurà un directori d'informació global. [Mou01, Mal07]

Per a importar i exportar informació entre servidors ldap, o per a descriure una sèrie de canvis que s'han d'aplicar al directori, el format utilitzat es diu LDIF (LDAP *data interchange format*). LDIF emmagatzema la informació en jerarquies orientades a objectes que després seran transformades al format intern de la base de dades. Un arxiu LDIF té un format similar a:

```
dn: o = UOC, c = SP
o: UOC
objectclass: organization
dn: cn = Pirulo Nteum, o = UOC, c = SP
cn: Pirulo Nteum
sn: Nteum
mail: nteum@uoc.edu
objectclass: person
```

Cada entrada és identificada per un nom indicat com a DN (*distinguished name*). El DN consisteix en el nom de l'entrada més una sèrie de noms que el relacionen amb la jerarquia del directori i en què hi ha un *objectclass* que defineix els atributs que poden ser utilitzats en aquesta entrada. LDAP proveeix un conjunt bàsic de classes d'objectes: grups (inclou llistes desordenades d'objectes individuals o grups d'objectes), localitzacions (com països i la seva descripció), organitzacions i persones. Una entrada pot, a més, pertànyer a més d'una classe d'objecte, per exemple, un individu és definit per la classe persona, però també pot ser definit per atributs de les classes inetOrgPerson, groupOfNames, i organization. L'estructura d'objectes del servidor (anomenat *schema*) determina quins són els atributs permesos per a un objecte d'una classe (els quals es defineixen en /etc/ldap/schema com opeldap.schema, corba.schema, nis.schema, inetorgperson. schema, etc.).

Totes les dades són representades com un parell *atribut* = *valor*, en què atribut és descriptiu de la informació que conté, per exemple, l'atribut utilitzat per a emmagatzemar el nom d'una persona és commonName, o cn, és a dir, per a una persona anomenada Pirulo Nteum, serà representat per cn: Pirulo Nteum i portarà associat altres atributs de la classe persona com givenname: Pirulo surname: Nteum mail: pirulo@uoc.edu. En les classes hi ha atributs obligatoris i optatius i cada atribut té una sintaxi associada que indica quin tipus d'informació conté l'atribut, per exemple, bin (*binary*), ces (*case exact string*, s'ha de buscar igual), cis (*case ignore string*, es pot ignorar M-m durant la cerca), tel (*telephone number string*, s'ignoren espais i '-'), dn (*distinguished name*). Un exemple d'un arxiu en format LDIF podria ser:

```
dn:             dc = UOC, dc = com
objectclass:    top
objectclass:    organizationalUnit

dn:             ou = groups, dc = UOC, dc = com
objectclass:    top
objectclass:    organizationalUnit
ou:             groups

dn:             ou = people, dc = UOC, dc = com
objectclass:    top
objectclass:    organizationalUnit
ou:             people

dn:             cn = Pirulo Nteum, ou = people, dc = UOC, dc = com
cn:             Pirulo Nteum
sn:             Nteum
objectclass:    top
objectclass:    person
objectclass:    posixAccount
objectclass:    shadowAccount
                uid:pirulo
                userpassword:{crypt}p1pss2ii(0pgbs*do&@ = )eksd
                uidnumber:104
                gidnumber:100
                gecos:Pirulo Nteum
                loginShell:/bin/bash
                homeDirectory: /home/pirulo
                shadowLastChange:10877
                shadowMin: 0
                shadowMax: 999999
                shadowWarning: 7
                shadowInactive: -1
                shadowExpire: -1
                shadowFlag: 0

dn:             cn = unixgroup, ou = groups, dc = UOC, dc = com
objectclass:    top
objectclass:    posixGroup
cn:             unixgroup
                gidnumber:200
                memberuid:pirulo
                memberuid:un altre-usuari
```

Les línies llargues poden ser continuades a sota començant per un espai o un *tab* (format LDIF). En aquest cas, s'ha definit la base DN per a la institució dc = UOC, dc = com, la qual conté dues subunitats: *people* i *groups*. Després s'ha descrit un usuari que pertany a *people* i a *group*. Una vegada preparat l'arxiu amb les dades, ha de ser importat al servidor perquè estigui disponible per als

clients LDAP. Hi ha eines per a transferir dades de diferents bases de dades a format LDIF. [Mal07]

Sobre Debian, s'ha d'instal·lar el paquet slapd que és el servidor d'OpenLdap. Durant la instal·lació farà una sèrie de preguntes com: *Mètode d'instal·lació del directori:* auto; *extensions al directoro [domain-host, lloc, institució]: host, domain, password de l'Adm*; *replicar canvis locales a altres servidors:* no. Aquesta instal·lació generarà un arxiu de configuració en /etc/ldap/slapd.conf i la base de dades sobre /var/lib/ldap. També hi ha un altre arxiu /etc/ldap/ldap.conf (o hi pot haver el ~/.ldaprc), que és l'arxiu de configuració utilitzat per a inicialitzar valors per defecte quan s'executen clients ldap. En aquest s'indica quina és la base de dades, quin és el servidor ldap, paràmetres de seguretat, mida de la cerca, etc.

L'arxiu de configuració del servidor /etc/ldap/slapd.conf (vegeu `man slap.con`) és format per diferents seccions, cadascuna d'elles indicada per una de les directives següents: global, backend specific i database specific, i en aquest ordre. La directiva *global* és de caràcter general i s'aplica a tots els *backends* (bases de dades) i defineixen qüestions generals com els permisos d'accés, atributs, temps d'espera, *schemas*, etc. La directiva *backend specific* defineix els atributs al *backend* específic que defineix (bdb, dnssrv, ldbm...), i el *database specific*, els atributs específics per a aquesta base de dades que defineix. Per a posar en marxa el servidor, s'ha d'executar:

```
/etc/init.d/slapd start (o stop per a aturar-lo)
```

El sistema durant la instal·lació haurà creat els enllaços adequats per a executar-lo després de l'inici.

7.1. Creació i manteniment de la base de dades

Hi ha dos mètodes per a inserir dades en la base de dades de LDAP. El primer és fàcil i adequat per a petites quantitats de dades, és interactiu i s'han d'utilitzar eines com ldapadd (o qualsevol altra com Ldap Browser http://www.iit.edu/~gawojar/ldap/) per a inserir-hi noves entrades. El segon s'ha de treballar fora de línia, és l'adequat per a grans BD i s'utilitza l'ordre slapadd inclosa amb slapd. Per ser més general, descriurem sintèticament el segon mètode, en el qual primer s'ha de verificar que conté els atributs següents en slapd.conf: suffix (*top* del directori, per exemple, suffix "o = UOC, c = SP"); directory /var/lib/ldap (directori en què es creen els índexs i que pugui escriure slapd). S'ha de verificar, a més, que la base de dades conté les definicions dels índexs que es volen:

```
index cn,sn,uid
index objectClass pres,eq
```

Una vegada definit l'slapd.conf, s'ha d'executar l'ordre:

```
slapadd –l entrada –f configuració [-d nivell] [-n sencer | -b sufix]
```

Els arguments són:

- *–l:* arxiu en format LDFI.
- *–f:* arxiu de configuració del servidor, en el qual s'indiquen com crear els índexs.
- *–d:* nivell de depuració.
- *–n:* Núm. de base de dades, si se'n té més d'una.
- *–b:* especifica quina base de dades cal modificar.

Hi ha altres ordres amb slapd com per exemple slapindex, que permet regenerar els índexs, i slapcat, que permet bolcar la BD a un arxiu en format LDIF.

8. Serveis d'arxius (NFS)

El sistema NFS permet que un servidor exporti un sistema d'arxiu perquè puguin ser utilitzats de manera interactiva des d'un client. El servei és format per un servidor nfsd i un client (*mountd*) que permeten compartir un sistema d'arxiu (o una part d'ell) per mitjà de la xarxa.

A Debian cal instal·lar per al client apt-get `install nfs-common portmap` mentre que el server necessita `apt-get install nfs-kernel-server nfs-common portmap`.

El servidor (a Debian) es posa en marxa mitjançant els scripts nfscommon i nfs-kernel-server en /etc/init.d (i els enllaços adequats en /etc/rcX.d).

El servidor utilitza un arxiu (/etc/exports) per a gestionar l'accés i control sobre els sistemes d'arxiu als quals s'accedirà remotament. Sobre el client, el *root* (o un altre usuari mitjançant *sudo*) pot muntar el sistema remot per mitjà de l'ordre:

```
mount Ipserver:directorio-remot directorio_local
```

i a partir d'aquest moment el directori-remot es veurà dins de directori local (aquest hi ha de ser abans d'executar el *mount*). Aquesta tasca en el client es pot automatitzar utilitzant l'arxiu de *mount* automàtic (/etc/fstab) inclosa una línia; per exemple:

pirulo.remix.com:/usr/local /pub nfs rsize=8192,wzise=8192,timeo=14

Aquesta sentència indica que es muntarà el directori /usr/local del host pirulo.remix.com en el directori local /pub. Els paràmetres rsize, wzise són les mides de blocs de lectura i escriptura, timeo és el timeout d'RPC (si no s'hi especifiquen aquests tres valors, es prenen els que inclou per defecte).

L'arxiu /etc/exports serveix d'ACL (llista de control d'accés) dels sistemes d'arxiu que es pot exportar als clients. Cada línia conté un *filesystem* per exportar seguit dels clients que el poden muntar, separats per espais en blanc. A cada client se li pot associar un conjunt d'opcions per a modificar el comportament (consulteu *man exports* per a un llista detallada de les opcions). Un exemple d'això podria ser:

```
# Exemple de /etc/etc/exports
/           /master (rw) trusty(rw,no_root_squash)
/projects   proj*.local.domain(rw)
/usr        *.local.domain(ro) @trusted(rw)
/pub        (ro,insecure,all_squash)
/home       195.12.32.2(rw,no_root_squash) www.first.com(ro)
/user       195.12.32.2/24(ro,insecure)
```

La primera línia exporta el sistema d'arxius sencer (/) a master i trusty en format lectura/escriptura. A més, per a trusty no hi ha *uid squashing* (el *root* del client accedirà com a *root* als arxius *root* del servidor, és a dir, els dos *root* són equivalents malgrat ser de màquines diferents; és indicat per a màquines sense disc). La segona i tercera línies mostren exemples de '*' i de *netgroups* (indicats per @). La quarta línia exporta el directori /pub a qualsevol màquina al món, només de lectura, permet l'accés de clients NFS que no utilitzen un port reservat per a l'NFS (opció *insecure*) i tot s'executa sota l'usuari *nobody* (opció *all squash*). La cinquena línia especifica un client per la seva IP i en la sisena igual però amb màscara de xarxa (/24) i amb opcions entre parèntesis () i que han de ser sense espai de separació. Només hi pot haver espais entre els clients habilitats. És important tenir en compte que d'NFS n'hi ha 3 versions (V2, V3 i recentment V4). Les més comunes són V3 i en algunes instal·lacions V2. Si des d'un client V3 es connecta a un servidor V2, s'ha d'indicar amb un paràmetre aquesta situació.

8.1. Servidor de Wiki

Un (o una) *wiki* (del hawaià, *wiki wiki,* "ràpid") és un lloc web col·laboratiu que pot ser editat per diversos usuaris que poden crear, editar, esborrar o modificar el contingut d'una pàgina web, d'una manera interactiva, fàcil i ràpida; aquestes facilitats fan d'una *wiki* una eina efectiva per a l'escriptura en col·laboració. La tecnologia *wiki* permet que pàgines web allotjades en un servidor públic (les pàgines *wiki*) siguin escrites de manera col·laborativa mitjançant un navegador, utilitzant una notació senzilla per a donar format, crear enllaços, etc., i conservar un historial de canvis que permet recuperar de manera senzilla qualsevol estat anterior de la pàgina. Quan algú edita una pàgina *wiki*, els seus canvis surten immediatament en la web, sense passar per cap tipus de revisió prèvia. *Wiki* també es pot referir a una col·lecció de pàgines hipertext, que poden ser visitades i editades per qualsevol persona (definició de Wikipedia). Debian té el seu *wiki* a http://wiki.debian.org/ i FC a http://fedoraproject.org/wiki/ i ambdues es basen en Moin Moin (http://moinmoin.wikiwikiweb.de/). MoinMoin és una *Python WikiClone* que permet inicialitzar ràpidament la seva pròpia *wiki* i només es necessita un servidor de web i el llenguatge Python instal·lat.

En http://moinmoin.wikiwikiweb.de/MoinMoinPackages/DebianLinux es troben les instruccions detallades per a instal·lar Moin Moin sobre Debian, però bàsicament es redueixen a: 1) instal·leu apache2 i mod_python; 2) configureu Apache per a apuntar en el codi de MoinMoin; 3) instal·leu el paquet moinmoin; 4) configureu moinmoin, i 5) reinicieu Apache. Un exemple de configuració:

```
apt-get install python-moinmoin
mkdir /var/www/mywiki
cp -r /usr/share/moin/data /usr/share/moin/underlay \
/usr/share/moin/server/moin.cgi /var/www/mywiki
chown -R www-data:www-data /var/www/mywiki
chmod -R g+w /var/www/mywiki
```

- Configureu apache2 afegint /etc/apache2/conf.d/wiki (o on tingui el fitxer de configuració):

```
Alias /wiki/ "/usr/share/moin/htdocs/"

<Location /mywiki>
        SetHandler python-program
        PythonPath "['/var/www/mywiki','/etc/moin/']+sys.path"
        PythonHandler MoinMoin.request::RequestModPy.run
        PythonDebug On
</Location>
```

- Reinicieu apache2:

```
/etc/init.d/apache2 reload
```

- Configureu Moinmoin: editeu /etc/moin/farmconfig.py (múltiples *wikis*)

```
wikis = [
 ("mywiki", r"^yoursite.com/mywiki/.*$"),
   ]
```

- També es pot usar (només una *wiki*):

```
wikis = [
 ("mywiki", r".*"),
   ]
```

- També en /etc/moin/farmconfig.py cal treure el comentari data_dir i data_underlay_dir (un per a cada *wiki*) i copieu-hi el fitxer.

```
cp /etc/moin/moinmaster.py /etc/moin/mywiki.py
```

- Llavors, editeu /etc/moin/mywiki.py i canvieu-hi:

```
sitename = u'MyWiki'
data_dir = ' /var/www/mywiki/data'
data_underlay_dir = ' /var/www/mywiki/underlay'
```

La Wiki estarà instal·lada sobre http://yoursite.com/mywiki/

Activitats

1. Configureu un servidor DNS com a cau i amb un domini propi.

2. Configureu un servidor/client NIS amb dues màquines i exporteu els directoris d'usuari del servidor per NFS.

3. Configureu un servidor SSH per a accedir des d'una altra màquina sense passwd.

4. Configureu un servidor Apache+ SSL+ PHP+ MySQL+ PHPAdmin per a visualitzar els fulls personals dels usuaris.

5. Creeu i configureu un sistema de correu electrònic amb Exim, fetchmail, Spam-Assassin i un servidor IMAP per a rebre correus des de l'exterior i poder llegir-los des d'una màquina remota amb el client Mozilla (thunderbird).

6. Instal·leu la Wiki MoinMoin i creeu un conjunt de pàgines per a verificar-ne el funcionament.

Fonts de referència i informació

[Debc, LPD03b, Ibi]

http://tldp.org/HOWTO/DNS-HOWTO-7.html

http://tldp.org/HOWTO/NIS-HOWTO/verification.html

Squid proxy server

Proxy Cache: http://www.squid-cache.org/

Transparent Proxy: http://tldp.org/HOWTO/TransparentProxy-1.html

Proftpd: http://www.debian-administration.org/articles/228

PureFtpd: http://www.debian-administration.org/articles/383

Exim: http://www.exim.org/docs.html

Mutt: http://www.mutt.org

ProcMail: http://www.debian-administration.org/articles/242

LWP:http://www.lawebdelprogramador.com/temas/tema_stablephpapachemysql.php

Moin Moin: (http://moinmoin.wikiwikiweb.de/)

Moin Moin + Debian:

http://moinmoin.wikiwikiweb.de/MoinMoinPackages/DebianLinux

Apache2 + SSL: http://www.debian-administration.org/articles/349