Administració de xarxa

Remo Suppi Boldrito

P07/M2003/02285

UOC
www.uoc.edu

Índex

Introducció

El sistema operatiu Unix (GNU/Linux) es pren com a exemple d'una arquitectura de comunicacions estàndard. Des del mític UUCP (servei de còpia entre sistemes operatius Unix) fins a les xarxes actuals, l'Unix sempre ha mostrat la seva versatilitat en aspectes relacionats amb la comunicació i l'intercanvi d'informació. Amb la introducció de xarxes d'ordinadors (àrea local LAN, àrea àmplia WAN o, les més actuals, àrea metropolitana MAN) que ofereixen enllaços multipunt a diferents velocitats (56 kbits/s fins a 1 Gbit/s), han anat sorgint nous serveis basats en protocols més ràpids, portables entre diferents ordinadors i millor adaptats, com el TCP/IP (*transport control program / Internet protocol*). [Com01, Mal96, Cis00, Gar98, KD00]

1. Introducció a TCP/IP (TCP/IP *suite*)

El protocol TCP/IP sintetitza un exemple d'estandardització i una voluntat de comunicació global.

El protocol TCP/IP és en realitat un conjunt de protocols bàsics que s'han anat agregant al principal per a satisfer les diferents necessitats en la comunicació ordinador–ordinador. Entre aquests protocols, tenim TCP, UDP, IP, ICMP, ARP [Mal96].

Nota

Utilització típica de TCP/IP *remote login:*

```
telnet localhost
Debian GNU/Linux 4.0
login:
```

En l'actualitat, la utilització més freqüent de TCP/IP per a l'usuari és la connexió remota a altres ordinadors (Telnet, SSH Secure Shell), la utilització de fitxers remots (*network file system*, NFS) o la seva transferència (*file transfer protocol,* FTP; *hipertext markup protocol,* HTTP).

1.1. Serveis sobre TCP/IP

Els serveis TCP/IP tradicionals més importants són els següents [Gar98]:

- **Transferència de fitxers.** El *file transfer protocol (*FTP) permet a un usuari d'un ordinador obtenir i enviar fitxers d'un ordinador a un altre. Per a això, l'usuari haurà de tenir un compte en l'ordinador remot i identificar-se amb el seu nom (*login*) i una paraula clau (*password*), o bé l'ordinador remot haurà de tenir un repositori d'informació (programari, documentació...). En aquest cas, l'usuari es connectarà com a anònim (*anonymous*) per transferir (llegir) aquests fitxers al seu ordinador. Això no és el mateix que els sistemes de fitxers de xarxa més recents, NFS i Network File System (o protocols *netbios* sobre TCP/IP, "invent" totalment insegur sobre el Windows i que és millor reemplaçar per una versió més antiga, però més segura, del mateix concepte anomenat *netbeui*), que permeten virtualitzar el sistema de fitxers d'una màquina perquè s'hi pugui accedir de manera interactiva sobre un altre ordinador.

- **Connexió** (*login*) remota: El protocol de terminal de xarxa (Telnet) permet, a un usuari, connectar-se a un ordinador remotament. L'ordinador local s'utilitza com a terminal de l'ordinador remot i s'hi executa tot. Aquest ordinador resta invisible durant la sessió. Actualment, aquest servei s'ha reemplaçat pel SHH (*secure shell*) per raons de seguretat. En una connexió remota mitjançant Telnet, els missatges circulen tal qual (text net), és a dir, si algú veu els missatges en la xarxa, equivaldrà a veure la pantalla de l'usu-

ari. SSH codifica la informació (que significa un cost afegit a la comunicació) i fa que els paquets en la xarxa siguin il·legibles a un node estrany.

- ***e-mail***. Aquest servei permet enviar missatges als usuaris d'altres ordinadors. Aquest mode de comunicació s'ha transformat en un element vital en la vida dels usuaris i permet que els *e-mails* (correus electrònics) siguin enviats a un servidor central, i que després puguin recuperats per mitjà de programes específics (clients) o ser llegits per mitjà d'una connexió web.

L'avenç de la tecnologia i el baix cost dels ordinadors han permès que determinats serveis s'hi hagin especialitzat i s'ofereixen configurats en determinats ordinadors que treballen en un model client–servidor. Un servidor és un sistema que ofereix un servei específic per a la resta de la xarxa. Un client és un altre ordinador que utilitza aquest servei. Generalment, tots aquests serveis s'ofereixen dins de TCP/IP:

- **Sistemes de fitxers en xarxa** (*network file systems*). Permeten a un sistema accedir als **fitxers** en un sistema remot d'una manera més integrada que FTP. Els dispositius d'emmagatzematge (o una part) s'exporten al sistema al qual es vol accedir, i aquest els pot veure com si fossin dispositius locals. Aquest protocol permet a qui exporta posar les regles i determinar les formes d'accés, la qual cosa (ben configurat) fa que el lloc en què es troba la informació físicament sigui independent del lloc en què es veu.

- **Impressió remota**. Permet accedir a impressores connectades a altres ordinadors.

- **Execució remota**. Permet que un usuari executi un programa sobre un altre ordinador. Hi ha diferents maneres de realitzar aquesta execució: o bé mitjançant una ordre (*rsh*, *ssh*, *rexec*) o mitjançant sistemes amb RPC (*remote procedure call*), que permeten que un programa en un ordinador local executi una funció d'un programa sobre un altre ordinador. Els mecanismes RPC han estat objecte d'estudi i hi ha diverses implementacions, però les més comunes són Xerox's Courier i Sun's RPC (aquesta última adoptada per la majoria dels Unix).

- **Servidors de nom** (*name servers*). En grans instal·lacions hi ha un conjunt de dades que s'han de centralitzar per a millorar-ne la utilització, per exemple, els noms d'usuari, paraules clau, adreces de xarxa, etc. Tot això facilita que un usuari disposi d'un compte per a totes les màquines d'una organització. Per exemple, Sun's Yellow Pages (NIS en les versions actuals de Sun) està dissenyat per a manejar tot aquest tipus de dades i es troba disponible per a la majoria d'Unix. El DNS (*domain name system*) és un altre servei de noms però que manté una relació entre el nom de la màquina i la seva identificació lògica (adreça IP).

- **Servidors de terminal** (*terminal servers*). Connecten terminals a un servidor que executa Telnet per connectar-se a l'ordinador central. Aquest tipus d'instal·lacions permet, bàsicament, reduir costos i millorar les connexions a l'ordinador central (en determinats casos).

- **Servidors de terminals gràfiques** (*network-oriented window systems*). Permeten que un ordinador visualitzi informació gràfica sobre un *display* que està connectat a un altre ordinador. El més comú d'aquests sistemes és l'X Windows.

1.2. Què és TCP/IP?

El TCP/IP són, en realitat, dos protocols de comunicació entre ordinadors independents l'un de l'altre.

D'una banda, el TCP (*transmission control protocol*), defineix les regles de comunicació perquè un ordinador (*host*) pugui "parlar" amb un altre (si es pren com a referència el model de comunicacions OSI/ISO es descriu la capa 4; vegeu la taula següent).

El TCP està orientat a la connexió, és a dir, com un telèfon, i la comunicació es tracta com un corrent de dades (*stream*).

D'altra banda, l'IP (*Internet protocol*) defineix el protocol que permet identificar les xarxes i establir els camins entre els diferents ordinadors.

És a dir, encamina les dades entre dos ordinadors per les xarxes. Correspon a la capa 3 del model OSI/ISO i és un protocol sense connexió (vegeu la taula següent). [Com01, Rid00, Dra99]

Una alternativa al TCP la conforma el protocol UDP (*user datagram protocol*), el qual tracta les dades com un missatge (datagrama) i envia paquets. És un protocol sense connexió (l'ordinador de destinació no ha d'estar escoltant, necessàriament, quan un altre ordinador hi estableix comunicació) i té l'avantatge d'exercir menys sobrecàrrega en la xarxa que les connexions de TCP, però la comunicació no és fiable (és possible que els paquets no arribin o arribin duplicats).

Hi ha un altre protocol alternatiu anomenat ICMP (*Internet control message protocol*). L'ICMP s'utilitza per a missatges d'error o control. Per exemple, si algú intenta connectar-se a un equip (*host*), l'ordinador local pot rebre un missatge ICMP que digui "*host unreachable*". L'ICMP també es pot utilitzar per a extreure informació sobre una xarxa. L'ICMP és similar a l'UDP,

ja que maneja missatges (datagrames), però és més simple que l'UDP perquè no té identificació de ports (els ports són bústies en què es dipositen els paquets de dades i des d'on les aplicacions servidores llegeixen els paquets esmentats) en l'encapçalament del missatge.

El model de comunicacions de l'OSI/ISO (*open systems interconnection reference model* / *international standards organization*) és un model teòric adoptat per moltes xarxes. Hi ha set capes de comunicació en què cada una té una interfície per a comunicar-se amb l'anterior i la posterior:

Nivell	Nom	Utilització
7	Aplicació	SMTP, *simple mail transfer* protocol, el servei pròpiament dit.
6	Presentació	Telnet, FTP implementa el protocol del servei.
5	Sessió	Generalment no s'utilitza.
4	Transport	TCP, UDP, transformació d'acord amb el protocol de comunicació.
3	Xarxa	IP permet encaminar el paquet (*routing*).
2	Link	Controladors (*drivers*) de la transformació d'acord amb el protocol físic.
1	Físic	Ethernet, ADSL... envien del paquet físicament.

En resum, el TCP/IP és una família de protocols (que inclouen IP, TCP i UDP), que proveeixen d'un conjunt de funcions a baix nivell utilitzades per la majoria de les aplicacions. [KD00, Dra99].

Alguns dels protocols que utilitzen els serveis esmentats han estat dissenyats per Berkeley, Sun o altres organitzacions. Oficialment, ells no formen part d'*Internet protocol suite* (IPS). Tanmateix, s'implementen utilitzant TCP/IP i, per tant, es consideren part formal de l'IPS. Una descripció dels protocols disponibles en Internet pot es consultar en l'RFC 1011 (vegeu referències sobre RFC [IET]), que llista tots els protocols disponibles. Actualment, hi ha una nova versió del protocol IPv6, també anomenat IPng (IP *next generation*), que reemplaça l'IPv4. Aquest protocol millora notablement l'anterior en temes com major nombre de nodes, control de trànsit, seguretat o aspectes de *routing*.

1.3. Dispositius físics (maquinari) de xarxa

Des del punt de vista físic (capa 1 del model OSI), el maquinari més utilitzat per a LAN es coneix com a Ethernet (o FastEthernet o GigaEthernet). Els seus avantatges són el baix cost, velocitats acceptables (10, 100, o 1.000 megabits per segon) i instal·lació fàcil.

Hi ha tres modes de connexió en funció del tipus de cable d'interconnexió: gruixut (*thick*), fi (*thin*) i parell trenat (*twisted par*).

Els dos primers són obsolets (utilitzen cable coaxial), mentre que l'últim es realitza amb cables (parells) trenats i connectors similars als telefònics (es coneixen com a RJ45). La connexió amb parell trenat es coneix com a 10 baseT o 100 baseT (segons la velocitat) i utilitza repetidors anomenats *hubs* com a punts d'interconnexió. La tecnologia Ethernet utilitza elements intermedis de comunicació (*hubs*, *switchs* i *routers*) per a configurar múltiples segments de xarxa i dividir el trànsit per millorar les prestacions de transferència d'informació. Normalment, en les grans institucions, aquestes LAN Ethernet estan interconnectades per fibra òptica utilitzant tecnologia FDDI (*fiber distributed data interface*), que és molt més cara i d'instal·lació molt més complexa, però permet obtenir velocitats de transmissió equivalents a Ethernet i no té la limitació de la distància d'aquesta (FDDI admet distàncies de fins a 200 km). El seu cost es justifica per a enllaços entre edificis o entre segments de xarxa molt congestionats. [Rid00, KD00]

A més, hi ha un altre tipus de maquinari menys comú, però no menys interessant, l'ATM (*asynchronous transfer mode*). Aquest maquinari permet muntar una LAN amb una qualitat de servei elevada, i és una bona opció quan s'han de muntar xarxes d'alta velocitat i baixa latència com, per exemple, les que involucren distribució de vídeo en temps real.

Hi ha un altre maquinari suportat pel GNU/Linux per a la interconnexió d'ordinadors, entre els quals podem esmentar els següents: Frame Relay o X.25 (utilitzat en ordinadors que accedeixen a WAN o l'interconnecten, i en servidors amb grans necessitats de transferències de dades), Packet Radio (interconnexió per ràdio utilitzant protocols com AX.25, NetRom o Rose) o dispositius *dialing up*, que utilitzen línies sèries, lentes però molt barates, mitjançant mòdems analògics o digitals (XDSI, DSL, ADSL, etc.). Aquestes últimes són les que s'acostumen a utilitzar en pimes o a casa, i requereixen un altre protocol per a la transmissió de paquets, com SLIP o PPP. Per virtualitzar la diversitat de maquinari sobre una xarxa, el TCP/IP defineix una interfície abstracta mitjançant la qual es concentraran tots els paquets que s'enviaran per un dispositiu físic (la qual cosa també significa una xarxa o un segment d'aquesta xarxa). És per això que, per cada dispositiu de comunicació en la màquina, estendrem una interfície corresponent en el *kernel* del sistema operatiu.

Exemple

En el GNU/Linux, Ethernet s'anomena *ethx* (en què, en totes, *x* indica un número d'ordre començant per 0), la interfície a línies sèries (mòdems) s'anomena *pppx* (per a PPP) o *slx* (per a SLIP), per a FDDI són *fddix*. Aquests noms són utilitzats per les ordres per a configurar-los i assignar-los el número d'identificació que, posteriorment, permetrà comunicar-se amb altres dispositius en la xarxa.

En el GNU/Linux pot significar haver d'incloure els mòduls adequats per al dispositiu (*network interface card*, NIC) adequat (en el *kernel* o com a mòduls). Això significa compilar el *kernel* després d'haver escollit, per exemple, amb *makemenuconfig*, el NIC adequat, i indicar-lo com a intern o com a mòdul (en aquest últim cas, també s'haurà de compilar el mòdul adequat).

Nota

Com es poden veure les interfícies de xarxa disponibles?

```
ifconfig -a
```

Aquesta ordre mostra totes les interfícies/paràmetres per defecte de cada una.

Els dispositius de xarxa es poden mirar en el directori */dev* que és on hi ha un **fitxer** (especial, tant si és de bloc com de caràcters, segons la seva transferència) que representa cada dispositiu de maquinari. [KD00, Dra99]

2. Conceptes en TCP/IP

Com s'ha vist, la comunicació significa una sèrie de conceptes que ampliarem a continuació [Mal96, Com01]:

- **Internet/intranet**. El terme *intranet* es refereix a l'aplicació de tecnologies d'Internet (xarxa de xarxes) dins d'una organització, bàsicament, per a distribuir i tenir disponible informació dins de la companyia. Per exemple, els serveis oferts pel GNU/Linux, com serveis Internet i intranet, inclouen correu electrònic, WWW, news, etc.

- **Node**. El node (*host*) és una màquina que es connecta a la xarxa (en un sentit ampli, un node pot ser un ordinador, una impressora, una torre o *rack* de CD, etc.), és a dir, un element actiu i diferenciable en la xarxa que reclama o deixa algun servei i comparteix informació.

- **Adreça de xarxa Ethernet** (*Ethernet address* o *MAC address*). Un nombre de 48 bits (per exemple, 00:88:40:73:AB:FF –en octal–, i 0000 0000 1000 1000 0100 0000 0111 0011 1010 1011 1111 1111 –en binari–) que es troba en el dispositiu físic (maquinari) del controlador (NIC) de xarxa Ethernet i és gravat pel seu fabricant (aquest nombre ha de ser únic al món, per la qual cosa cada fabricant de NIC té un rang preassignat).

Nota

Nom de la màquina:
`more/etc/hostname`

- ***Host name***. Cada node ha de tenir, a més, un únic nom en la xarxa. El node pot ser només un nom o bé utilitzar un esquema de noms jeràrquic basat en dominis (*hierarchical domain naming scheme*). Els noms dels nodes han de ser únics, la qual cosa és fàcil en xarxes petites, més difícil en xarxes extenses, i impossible en Internet si no es fa algun control. Els noms han de tenir un màxim de 32 caràcters entre *a-z A-Z 0-9.-*, i no han d'incloure espais o # si comencen per un caràcter alfabètic.

Nota

Adreça IP de la màquina:
`more/etc/hosts`

- **Adreça d'Internet** (*IP address*). Consta de quatre nombres en el rang 0-255 separats per punts (per exemple 192.168.0.1) i s'utilitza universalment per a identificar els ordinadors en una xarxa o Internet. La translació de noms en adreces IP la realitza un servidor DNS (*domain name system*) que transforma els noms de node (llegibles per humans) en adreces IP (aquest servei el realitza una aplicació denominada *named*).

Nota

Ports preassignats en l'Unix:
`more/etc/services`
Aquesta ordre mostra els ports predefinits per ordre i segons suportin TCP o UDP.

- **Port** (*port*). És un identificador numèric de la bústia en un node que permet que un missatge (TCP, UDP) pugui ser llegit per una aplicació concreta dins d'aquest node (per exemple, dues màquines que es comuniquin per Telnet, ho faran pel port 23, però les dues mateixes màquines poden tenir una comunicació *ftp* pel port 21). Es poden tenir diferents aplicacions comunicant-se entre dos nodes mitjançant diferents ports simultàniament.

Nota

Visualització de la configuració del *routing*:
`netstat -r`

- **Node *router*** (*gateway*). És un node que realitza encaminaments (transferència de dades o *routing*). Segons les seves característiques, un encaminador (*router*) podrà transferir informació entre dues xarxes de protocols similars o diferents i, a més, pot ser selectiu.

Nota

Domini i qui és el nostre servidor de DNS:

```
more/etc/default domain
more/etc/resolv.conf
```

- ***Domain name system*** (DNS). Permet assegurar un únic nom i facilita l'administració de les bases de dades que realitzen la translació entre nom i adreça d'Internet, i s'estructuren en forma d'arbre. Per a això, s'especifiquen dominis separats per punts, el més alt (de dreta a esquerra) dels quals descriu una categoria, institució o país (COM, comercial; EDU, educació; GOV, governamental; MIL, govern militar; ORG, sense ànim de lucre; XX dues lletres per país; en casos especials són tres lletres, com CAT, llengua i cultura catalana...). El segon nivell representa l'organització, el tercer i restants representen departaments, seccions o divisions dins d'una organització (per exemple, *www.uoc.edu* o *nteum@pirulo.remix.es*). Els dos primers noms (de dreta a esquerra, *uoc.edu* en el primer cas, *remix.es* en el segon) han de ser assignats (aprovats) pel SRI-NIC (òrgan mundial gestor d'Internet), i els restants poden ser configurats/assignats per la institució.

Nota

Taules d'arp:

```
arp a NomNode
```

- **DHCP, *bootp***. DHCP i *bootp* són protocols que permeten, a un node client, obtenir informació de la xarxa (com l'adreça IP del node). Moltes organitzacions amb gran quantitat de màquines utilitzen aquest mecanisme per a facilitar l'administració a grans xarxes o on hi ha una gran quantitat d'usuaris mòbils.

- **ARP, RARP.** En algunes xarxes (com, per exemple, IEEE 802 LAN, que és l'estàndard per a Ethernet), les adreces IP es descobreixen automàticament mitjançant dos protocols membres d'*Internet protocol suite: address resolution protocol* (ARP) i *reverse address resolution protocol* (RARP). L'ARP utilitza missatges (*broadcast messages*) per a determinar l'adreça Ethernet (especificació MAC de la capa 3 del model OSI) corresponent a una adreça de xarxa particular (IP). El RARP utilitza missatges de tipus *broadcast* (missatge que arriba a tots els nodes) per a determinar l'adreça de xarxa associada amb una adreça de maquinari en particular. El RARP és especialment important en màquines sense disc, en les quals l'adreça de xarxa, generalment, no es coneix en el moment de l'inici (*boot*).

- **Biblioteca de** sockets. En l'Unix, tota la implementació del TCP/IP forma part del *kernel* del sistema operatiu (dins del mateix o com un mòdul que es carrega en el moment de l'inici, com el cas del GNU/Linux amb els controladors de dispositius). La manera d'utilitzar-les per un programador és per mitjà de l'API (*application programming interface*) que implementa aquest operatiu. Per al TCP/IP, l'API més comuna és la Berkeley Socket Library (el Windows utilitza una llibreria equivalent que s'anomena Winsocks). Aquesta biblioteca permet crear un punt de comunicació (*socket*), associar aquest a una adreça d'un node remot/port (*bind*), i oferir el servei de comunicació (per mitjà de *connect*, *listen*, *accept*, *send*, *sendto*, *recv*, *recvfrom*, per exemple). La biblioteca proveeix, a més

de la manera més general de comunicació (família AF INET), de comunicacions més optimitzades per a casos en què els processos es comuniquen en la mateixa màquina (família AF Unix). En el GNU/Linux, la biblioteca de *socket* és part de la biblioteca estàndard de C i Libc, (Libc6, en les versions actuals), i suporta AF_INET, AF_Unix, AF_IPX (per a protocols de xarxes Novell), AF_X25 (per al protocol X.25), AF_ATMPVC-AF_ATMSVC (per al protocol ATM) i AF_AX25, _FNETROM, AF_ROSE (per al *amateur radio protocol*).

3. Com s'assigna una adreça d'Internet?

Aquesta adreça té dos camps: l'esquerre representa la identificació de la xarxa i el dret la identificació del node. Si considerem el que hem esmentat anteriorment (quatre nombres entre 0 i 255, és a dir, 32 bits o 4 bytes), cada byte representa o bé la xarxa o bé el node. La part de xarxa és assignada pel NIC (Network Information Center) o alguna de les seves delegacions territorials, i la part del node és assignada per la institució o el proveïdor).

Hi ha algunes restriccions: **0** (per exemple, 0.0.0.0) en el camp de xarxa està reservat per al *routing* per defecte, i **127** (per exemple, 127.0.0.1) està reservat per a l'autorreferència (*local loopback* o *local host*), **0** en la part de node es refereix a aquesta xarxa (per exemple, 192.168.0.0), i **255** està reservat per a paquets de tramesa a totes les màquines (*broadcast*), per exemple, 198.162.255.255. En les diferents assignacions, es poden tenir diferents tipus de xarxes o adreces:

- **Classe A** (*red.host.host.host*). 1.0.0.1 a 126.254.254.254 (126 xarxes, 16 milions de nodes) defineixen les grans xarxes. El patró binari és **0** +7 bits xarxa + 24 bits de nodes.

- **Classe B** (*red.red.host.host*). 128.1.0.1 a 191.255.254.254 (16 K xarxes, 65 K nodes). Generalment, s'utilitza el primer byte de node per a identificar subxarxes dins d'una institució). El patró binari és **10** +14 bits de xarxa +16 bits de nodes.

- **Classe C** (*red.red.red.host*). 192.1.1.1 a 223.255.255.254 (2 milions de xarxes, 254 de nodes). El patró binari és **110** + 21 bits xarxa + 8 bits de nodes.

- **Classe D i E** (*red.red.red.host*). 224.1.1.1 a 255.255.255.254 reservat per a *multicast* (des d'un node a un conjunt de nodes que formen part d'un grup) i propòsits experimentals.

Alguns rangs d'adreces s'han reservat perquè no corresponguin a xarxes públiques, sinó a xarxes privades (màquines que es connecten entre elles sense tenir connexió amb l'exterior) i els missatges no s'encaminaran per Internet. Això és el que, comunament, es coneix com a intranet). Aquestes són, per a la **classe A**, de 10.0.0.0 a 10.255.255.255; **classe B**, de 172.16.0.0 fins a 172.31.0.0; i **classe C**, de 192.168.0.0 a 192.168.255.0.

L'adreça de *broadcast* és especial, ja que cada node en una xarxa escolta tots els missatges, a més de la seva pròpia adreça. Aquesta adreça permet que es puguin enviar datagrames (generalment, informació de *routing* i missatges d'avís) a una xarxa, i que tots els nodes del mateix segment de xarxa els puguin llegir. Per exem-

ple, quan l'ARP busca l'adreça Ethernet corresponent a un IP, utilitza un missatge de *broadcast*, el qual és enviat a totes les màquines de la xarxa simultàniament. Cada node en la xarxa llegeix aquest missatge i compara l'IP que es busca amb la pròpia i retorna un missatge al node que va fer la pregunta si hi ha coincidència.

Dos conceptes complementaris al que descrit anteriorment és el de les *subxarxes* i el *routing* entre elles. Fer **subxarxes** significa subdividir la part del node en petites xarxes dins de la mateixa xarxa per a, per exemple, millorar el trànsit. Una subxarxa pren la responsabilitat d'enviar el trànsit a certs rangs d'adreces IP, i estén el mateix concepte de xarxes A, B, C, però només aplicant aquesta redirecció en la part node de l'IP. El nombre de bits que són interpretats com a identificador de la subxarxa és donat per una màscara de xarxa (*netmask*) que és un nombre de 32 bits (igual que l'IP). Per a obtenir l'identificador de la subxarxa, s'haurà de fer una operació lògica I (*AND*) entre la màscara i l'IP, la qual cosa donarà l'IP de la subxarxa. Per exemple, si tenim una institució que té una xarxa de classe B amb el número 172.17.0.0, i el seu *netmask* és, per tant, 255.255.0.0. Internament, aquesta xarxa està formada per petites xarxes (una planta de l'edifici, per exemple). Així, el rang d'adreces és reassignat en 20 subnets (plantes per a nosaltres) 172.17.1.0 fins a 172.17.20.0. El punt que connecta totes aquestes plantes (*backbone*) té la seva pròpia adreça, per exemple, 172.17.1.0.

Aquestes subxarxes comparteixen el mateix IP de xarxa, mentre que la tercera s'utilitza per a identificar cada una de les seves subxarxes (per això, s'utilitzarà una màscara de xarxa 255.255.255.0).

El segon concepte, ***routing***, representa la manera en què els missatges són enviats per les subxarxes. Per exemple, imaginem tres departaments amb subxarxes Ethernet:

1) Compres (subxarxa 172.17.2.0).
2) Clients (subxarxa 172.17.4.0).
3) Recursos humans o RH (subxarxa 172.17.6.0).
4) *Backbone* amb FFDI (subxarxa 172.17.1.0).

Per a encaminar els missatges entre els ordinadors de les tres xarxes, es necessitaran tres *gateways* que tindran, cada un, dues interfícies de xarxa per a canviar entre Ethernet i FFDI. Aquestes seran les següents:

1) CompresGW IP:172.17.2.1 i 172.17.1.1.
2) ClientsGW IP:172.17.4.1 i 172.17.1.2.
3) RHGW IP:172.17.6.1 i 172.17.1.3, és a dir, un IP cap al costat de la subnet i una altra cap al *backbone*.

Quan s'envien missatges entre màquines de compres, no és necessari sortir al *gateway*, ja que el protocol TCP/IP trobarà la màquina directament. El problema és quan la màquina *Compres0* vol enviar un missatge a RH3. El missatge ha de circular pels dos *gateways* respectius. Quan Compres0 "veu" que RH3 és en

una altra xarxa, envia el paquet pel *gateway* CompresGW, que al seu torn l'enviarà a RHGW i que, al seu torn, l'enviarà a RH3. L'avantatge de les subxarxes és clar, ja que el trànsit entre totes les màquines de compres, per exemple, no afectarà les màquines de clients o RH (si bé significa un plantejament més complex i car a l'hora de dissenyar i construir la xarxa).

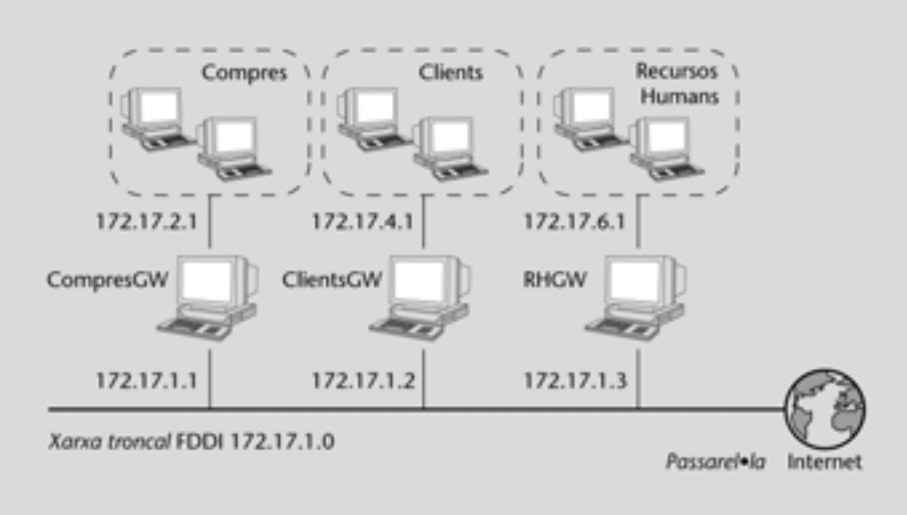

Figura 1. Configuració de segments i *gateways* en una Intranet

El IP utilitza una taula per a fer el *routing* dels paquets entre les diferents xarxes i en la qual hi ha un *routing* per defecte associat a la xarxa 0.0.0.0. Totes les adreces que coincideixen amb aquesta –ja que cap dels 32 bits no són necessaris– s'envien pel *gateway* per defecte (*default gateway*) cap a la xarxa indicada. Sobre compresGW, per exemple, la taula podria ser la següent:

Adreça	Màscara	Gateway	Interfície
172.17.1.0	255.255.255.0	-	fddi0
172.17.4.0	255.255.255.0	172.17.1.2	fddi0
172.17.6.0	255.255.255.0	172.17.1.3	fddi0
0.0.0.0	0.0.0.0	172.17.2.1	fddi0
172.17.2.0	255.255.255.0	-	eth0

El guionet (–) significa que la màquina està directament connectada i no necessita *routing*. El procediment a per identificar si es realitza el *routing* o no, es du a terme amb una operació molt simple amb dos AND lògics (subxarxa AND *mask* i *origen* AND *mask*) i una comparació entre els dos resultats. Si són iguals no hi ha *routing*, sinó que s'ha d'enviar la màquina definida com a *gateway* en cada màquina perquè aquesta realitzi el *routing* del missatge.

Per exemple, un missatge de la 172.17.2.4 cap a la 172.17.2.6 significarà el següent:

172.17.2.4 AND 255.255.255.0 = 172.17.2.0
172.17.2.6 AND 255.255.255.0 = 172.17.2.0

Com que els resultats són iguals, no hi haurà *routing*. En canvi, si fem el mateix amb 172.17.2.4 cap a 172.17.6.6, podem veure que hi haurà un *routing* per mitjà del 172.17.2.1 amb un canvi d'interfície (*eth0* a *ffdi0*) a la 172.17.1.1, i d'aquesta cap a la 172.17.1.2 amb un altre canvi d'interfície (*fddi0* a *eth0*) i, després, cap a la 172.17.6.6. Per defecte, el *routing* s'utilitzarà quan cap regla no satisfaci la coincidència. En cas que dues regles coincideixin, s'utilitzarà la que ho faci de manera més precisa, és a dir, la que tingui menys zeros. Per a construir les taules de *routing*, es pot utilitzar l'ordre *route* durant l'arrencada de la màquina, però si és necessari utilitzar regles més complexes (o *routing* automàtic), es pot utilitzar el *routing information protocol* (RIP) o, entre sistemes autònoms, l'*external gateway protocol* (EGP) o també el *border gateway protocol* (BGP). Aquests protocols s'implementen en l'ordre *gated*.

Per a instal·lar una màquina sobre una xarxa, és necessari, per tant, disposar de la informació següent obtinguda del proveïdor de xarxa o del seu administrador: adreça IP del node, adreça de la xarxa IP, adreça de *broadcast*, adreça de màscara de xarxa, adreça de *router* i adreça del DNS.

Si es construeix una xarxa que mai no tindrà connexió a Internet, es poden escollir les adreces que es prefereixin, però és recomanable mantenir un ordre adequat en funció de la mida de xarxa que es vulgui tenir i per a evitar problemes d'administració dins de la xarxa esmentada. A continuació, es veurà com es defineix la xarxa i el node per a una xarxa privada (cal anar amb compte, ja que, si es té la màquina connectada a la xarxa, es podria perjudicar un altre usuari que tingués assignada aquesta adreça): adreça de node 192.168.110.23, màscara de xarxa 255.255.255.0, part de xarxa 192.168.110., part de node .23, adreça de xarxa 192.168.110.0, adreça de *broadcast* 192.168.110.255.

4. Com s'ha de configurar la xarxa?

4.1. Configuració de la interfície (NIC, *network interface controller*)

Una vegada carregat el *kernel* del GNU/Linux, aquest executa l'ordre *init* que, al seu torn, llegeix el fitxer de configuració /etc./inittab i comença el procés d'inicialització. En general, l'*inittab* té seqüències com ara si::sysinit:/etc/init.d/boot, que representa el nom del fitxer d'ordres (*script*) que controla les seqüències d'inicialització. Generalment, aquest *script* crida d'altres *scripts*, entre els quals es troba la inicialització de la xarxa.

Exemple

A Debian s'executa *etc/init.d/network* per a la configuració de la interfície de xarxa i en funció del nivell d'arrencada. Per exemple, en el 2 s'executaran tots els fitxers S* del directori */etc/rc2.d* (que són enllaços al directori */etc/initd*), i en el nivell d'apagat, tots els K* del mateix directori. D'aquesta manera, l'*script* només hi és una vegada (*/etc/init.d*) i, d'acord amb els serveis volguts en aquest estat, es crea un enllaç en el directori corresponent a la configuració del node-estat.

Els dispositius de xarxa es creen automàticament quan s'inicialitza el maquinari corresponent. Per exemple, el controlador d'Ethernet crea les interfícies eth[0..n] seqüencialment quan es localitza el maquinari corresponent.

Nota

Consulteu
`man ifconfig`
per a les diferents opcions de l'ordre.

A partir d'aquest moment, es pot configurar la interfície de xarxa, la qual cosa implica dos passos: assignar l'adreça de xarxa al dispositiu i inicialitzar els paràmetres de la xarxa al sistema. L'ordre utilitzada per a això és l'*ifconfig* (*interface configure*). Vegem-ne un exemple:

```
ifconfig eth0 192.168.110.23 netmask 255.255.255.0 up
```

Això indica configurar el dispositiu *eth0* amb adreça IP 192.168.110.23 i màscara de xarxa 255.255.255.0. *up* indica que la interfície passarà a l'estat actiu (per a desactivar-la s'hauria d'executar *ifconfig eth0 down*). L'ordre assumeix que si alguns valors no s'indiquen, es prenen per defecte. En aquest cas, el *kernel* configurarà aquesta màquina com a Tipus-C, i configurarà la xarxa amb 192.168.110.23 i l'adreça de *broadcast* amb 192.168.110.255. Per exemple:

```
ifconfig eth0 192.168.110.23 netmask 255.255.255.0 up
```

Hi ha ordres com l'*ifup* i *ifdown* que permeten configurar i desconfigurar la xarxa de manera més simple utilitzant el fitxer */etc/network/interfaces* per a obtenir tots els paràmetres necessaris (consulteu *man interfaces* per a la seva sintaxi).

A Debian, a fi de facilitar la configuració de la xarxa, hi ha una altra manera de configurar la xarxa (considerada d'alt nivell) que utilitza les ordres esmen-

tades anteriorment *ifup* i *ifdown*, i el fitxer */etc/network/interfaces*. Si es decideix utilitzar aquestes ordres, **no** s'hauria de configurar la xarxa a baix nivell, ja que aquestes ordres són suficients per a configurar i desconfigurar la xarxa.

Per a modificar els paràmetres de xarxa de la interfície *eth0* (consulteu *man interfaces* en la secció 5 del manual d'Unix inclòs amb el sistema operatiu per a més informació del format), es pot fer el següent:

```
ifdown eth0                 per a tots els serveis de xarxa sobre eth0
vi /etc/network/interfaces  editi i modifiqui els que necessiti
ifup eth0                   posa en marxa els serveis de xarxa sobre eth0
```

Suposem que es vol configurar sobre Debian una interfície *eth0* que té una adreça IP fixa `192.168.0.123`., i amb `192.168.0.1` com a porta d'enllaç (*gateway*). S'ha d'editar */etc/network/interfaces* de manera que inclogui una secció com la següent:

```
iface eth0 inet static
        address 192.168.0.123
        netmask 255.255.255.0
        gateway 192.168.0.1
```

Si té instal·lat el paquet *resolvconf* pot afegir línies per a especificar la informació relativa al DNS. Per exemple:

```
iface eth0 inet static
        address 192.168.0.123
        netmask 255.255.255.0
        gateway 192.168.0.1
        dns-search remix.org
        dns-nameservers 195.238.2.21 195.238.2.22
```

Després d'activar-se la interfície, els arguments de les opcions *dns-search* i *dns-nameservers* queden disponibles per a *resolvconf* per a la seva inclusió en *resolv.conf*. L'argument *remix.org* de l'opció *dns-search* correspon a l'argument de l'opció *search* en *resolv.conf* (es veurà més endavant) i els arguments 195.238.2.21 i 195.238.2.22 de l'opció *dns-nameservers* correspon als arguments de les opcions *nameserver* en *resolv.conf* (consulteu *man resolv.conf*). També es pot configurar la xarxa a baix nivell mitjançant l'ordre *ip* (que és equivalent a *ifconfig* i *route*). Si bé aquesta ordre és molt més versàtil i potent (permet establir túnels, *routings* alternatius, etc.), és més complexa i es recomana utilitzar els procediments anteriors per a configuracions bàsiques de la xarxa.

4.1.1. Configuració de xarxa en (estil) Fedora

Red Hat i Fedora utilitzen estructures de fitxers diferents per a la configuració de la xarxa: */etc/sysconfig/network*. Per exemple, per a la configuració estàtica de la xarxa:

```
NETWORKING=yes
HOSTNAME=my-hostname          Nom del host definit pel cmd hostname
FORWARD_IPV4=true             True per a NAT firewall gateways i routers
                              False per a qualsevol altre cas
GATEWAY="XXX.XXX.XXX.YYY"     Porta de sortida a Internet
```

Per a configuració per DHCP s'ha de treure la línia de GATEWAY, ja que l'assignarà el servidor. I en cas d'incorporar NIS, s'ha d'agregar una línia amb el servidor de domini: NISDOMAIN=NISProject1

Per a configurar la interfície *eth0* en el fitxer */etc/sysconfig/network-scripts/ifcfg-eth0*:

DEVICE=eth0
BOOTPROTO=static
BROADCAST=XXX.XXX.XXX.255
IPADDR=*XXX.XXX.XXX.XXX*
NETMASK=255.255.255.0
NETWORK=*XXX.XXX.XXX.0*
ONBOOT=yes Activarà la xarxa en el *boot*.

També, a partir d'FC3, es poden agregar:
TYPE=Ethernet
HWADDR=*XX:XX:XX:XX:XX:XX*
GATEWAY=*XXX.XXX.XXX.XXX*
IPV6INIT=no
USERCTL=no
PEERDNS=yes

O, si no, per a configuració per DHCP:

DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp

Per deshabilitar DCHP, s'ha de canviar BOOTPROTO=dhcp per BOOTPROTO=none. Qualsevol canvi en aquests fitxers haurà de reiniciar els serveis amb *service network restart* (o, sinó, */etc/init.d/network restart*).

Per a canviar el nom del *host*, s'han de seguir aquests tres passos:

1) Executar l'ordre `hostname nom-nou`.

2) Canviar la configuració de la xarxa per */etc/sysconfig/network* editant `HOSTNAME=nom-nou`.

3) Restaurar els serveis (o fer un *reboot*):

- `service network restart` (o: */etc/init.d/network restart*).
- Reiniciar el *desktop* passant a mode consola `init 3` i canviant a mode GUI `init 5`.

Cal verificar si el nom tampoc no està donat d'alta en el */etc/hosts*. El *hostname* es pot canviar en temps d'execució amb `sysctl -w kernel.hostname=` "`nom-nou`".

4.1.2. Configuració d'un xarxa Wi-Fi (sense fil)

Per a la configuració d'interfícies Wi-Fi s'utilitza, bàsicament, el paquet *wireless-tools* (a més d'*ifconfig* o *ip*). Aquest paquet utilitza l'ordre *iwconfig* per a configurar una interfície sense fil, però també es pot fer amb */etc/network/interfaces*.

Exemple: Configurar una Wi-Fi a Debian Sarge(Etch) (similar en FC6)

Suposem que volem configurar una targeta de xarxa sense fil Intel Pro/Wireless 2200BG (molt comú en una gran quantitat de portàtils com ara el Dell, HP...). Normalment, el programari que controla les targetes es divideix en dues parts: el mòdul programari que es carregarà en el *kernel* per mitjà de l'ordre *modprobe*, i el *firmware*, que és el codi que es carregarà en la targeta i que ens dóna el fabricant (consulteu la pàgina d'Intel per a aquest model). Com que parlem de mòduls, és interessant utilitzar el paquet de Debian *module-assistant* que ens permet crear i instal·lar fàcilment un mòdul (una altra opció seria instal·lar les fonts i crear el mòdul corresponent). El programari (el trobem en la pàgina del fabricant i el denomina *ipw2200*), el compilarem i instal·larem amb l'ordre *m-a* del paquet *module-assistant*.

```
aptget install module-assistant (instal·lem el paquet)
m-a -t update
m-a -t -f get ipw2200
m-a -t -build ipw2200
m-a -t install ipw2200
```

Des de l'adreça indicada pel fabricant (en la seva documentació) es baixa la versió del *firmware* compatible amb la versió del *driver*. En el nostre cas, per al *driver* versió 1.8, la versió del *firmware* és la 2.0.4, obtinguda des de la pàgina següent:

http://ipw2200.sourceforge.net/firmware.php

I a continuació es descomprimeix i instal·la el microprogramari:

```
tar xzvf ipw2200fw2.4.tgz C /tmp/fwr/
cp /tmp/fwr/*.fw /usr/lib/hotplug/firmware/
```

Amb això, es copiaran tres paquets (*ipw2200-bss.fw*, *ipw2200-ibss.fw* i *ipw2200-sniffer.fw*). A continuació, es carrega el mòdul amb `modprobe ipw2200`, es reinicia el sistema (*reboot*) i, després, des de la consola, podem fer `dmesg | grep ipw`. Aquesta ordre ens mostrarà algunes línies similars a les que es mostren tot seguit i que indicaran que el mòdul està carregat (es pot verificar amb lsmod):

```
ipw2200: Intel(R) PRO/Wireless 2200/2915 Network Driver, git1.0.8
ipw2200: Detected Intel PRO/Wireless 2200BG Network Connection
...
```

Després, es baixa el paquet *wirelesstools*, que conté *iwconfig*, i amb *aptget install wirelesstools*, entre altres, executem *iwconfig*. El resultat serà semblant a aquest:

```
eth1 IEEE 802.11b ESSID:"Nom-de-la-Wifi"
Mode:Managed Frequency:2.437 GHz
Access Point:00:0E:38:84:C8:72
Bit Rate=11 Mb/s TxPower=20 dBm
Security mode:open
...
```

A continuació, cal configurar el fitxer de xarxes, per exemple, *gedit /etc/network/interfaces*, i afegir la interfície *wifi eth1*, per exemple:

```
iface eth1 inet dhcp
      pre-up iwconfig eth1 essid "Nom de la Wifi"
      pre-up iwconfig eth1 key open XXXXXXXXXX
```

Les línies *pre-up* executen l'ordre *iwconfig* abans d'activar la interfície. Aquesta configuració es fa si es vol utilitzar un servei en mode DHCP (assignació automàtica d'IP, que es veurà més endavant). S'ha d'utilitzar la paraula *static* en comptes de *dhcp* i, a més, posar, per exemple, les línies següents (com en una targeta de cable):

```
   address 192.168.1.132
   netmask 255.255.255.0
   network 192.168.0.0
   broadcast 192.168.0.255
   gateway 192.168.1.1
```

Un mètode alternatiu per a configurar la interfície és el següent:

```
iface eth1 inet dhcp
      wireless-essid "Nom de la Wifi"
      wireless-key 123456789e
```

A continuació, es pot posar en marxa la xarxa amb *ifup eth1*, que ens donarà informació sobre la connexió i ens indicarà el seu estat i la qualitat de recepció. Per a buscar (*scan*) les xarxes Wi-Fi disponibles (punts d'accés), podem utilitzar `iwlist scan`, que ens mostrarà informació de les xarxes disponibles, i si ens volem connectar a una de diferent, es pot utilitzar l'ordre *iwconfig* per a canviar de xarxa o punt d'accés (Access Point).

4.2. Configuració del Name Resolver

El següent pas consisteix a configurar el *name resolver*, que converteix noms com *pirulo.remix.come* en 192.168.110.23. El fitxer */etc/resolv.conf* és l'utilitzat per a aquesta finalitat. El seu format és molt simple (una línia de text per sentència). Hi ha tres paraules clau per a aquesta finalitat: *domain* (domini local), *search* (llista de dominis alternatius) i *name server* (l'adreça IP del *domain name server*).

Exemple de /etc/resolv.conf

```
domain remix.com
search remix.com piru.com
name server 192.168.110.1
name server 192.168.110.65
```

Aquesta llista de servidors de nom, sovint, depenen de l'entorn de xarxa, que pot canviar depenent d'on sigui o es connecti la màquina. Els programes de connexió a línies telefòniques `(pppd)` o obtenció d'adreces IP automàticament `(dhclient)` són capaços de modificar `resolv.conf` per a inserir o eliminar servidors; però aquestes característiques no sempre funcionen adequadament i, de vegades, poden entrar en conflicte i generar configuracions errònies. El paquet **`resolvconf`** (encara en *unstable*) soluciona adequadament el problema i permet una configuració simple dels servidors de nom en forma dinàmica. El paquet `resolvconf` està dissenyat per a funcionar sense que sigui necessària cap configuració manual, no obstant això, el paquet és bastant nou i pot requerir alguna intervenció per a aconseguir que funcioni adequadament. Per a més informació:

http://packages.debian.org/unstable/net/resolvconf

Un fitxer important és el */etc/host.conf*, que permet configurar el comportament del *name resolver*. La seva importància rau en el fet que indica on es resol primer l'adreça o el nom d'un node. Aquesta consulta es pot fer al servidor DNS o a taules locals dins de la màquina actual *(/etc/hosts*).

Exemple de /etc/host.conf

```
order hosts,bind
multi on
```

Aquesta configuració indica que s'ha de verificar el */etc/hosts* abans de sol·licitar una petició al DNS, i també indica (2a. línia) que s'han de retornar totes les adreces vàlides que es trobin en */etc/hosts*. Per aquest motiu, les adreces locals es col·loquen en el fitxer */etc/hosts*, el qual també serveix per a accedir a nodes sense haver de consultar el DNS. La consulta és molt més ràpida, però té el desavantatge que si el node canvia, l'adreça serà incorrecta. En un sistema correctament configurat, només hauran d'aparèixer el node local i una entrada per a la interfície *loopback*.

Exemple de /etc/hosts

```
127.0.0.1     localhost          loopback
192.168.1.2   pirulo.remix.com   pirulo
```

Per al nom d'una màquina es poden utilitzar àlies, la qual cosa significa que aquesta màquina es pot anomenar de diferents maneres per a la mateixa adreça IP. Pel que fa a la interfície *loopback,* és un tipus especial d'interfície que permet realitzar al node connexions amb si mateix (per exemple, per verificar que el subsistema de xarxa funciona sense accedir a la xarxa). Per defecte, l'adreça IP 127.0.0.1 ha estat assignada específicament al *loopback* (una ordre *telnet 127.0.0.1* es connectarà amb la mateixa màquina). La seva configuració és molt fàcil (l'efectuen generalment els *scripts* d'inicialització de xarxa).

Exemple del *loopback*

```
ifconfig lo 127.0.0.1
route add host 127.0.0.1 lo
```

En la versió 2 de la biblioteca GNU, hi ha un reemplaçament important respecte a la funcionalitat del fitxer *host.conf*. Aquesta millora inclou la centralització d'informació de diferents serveis per a la resolució de noms, la qual cosa presenta grans avantatges per a l'administrador de xarxa. Tota la informació de consulta de noms i serveis s'ha centralitzat en el fitxer */etc/nsswitch.conf*, el qual permet a l'administrador configurar l'ordre i les bases de dades de manera molt simple. En aquest fitxer cada servei apareix un cop per línia amb un conjunt d'opcions, en què, per exemple, la resolució de noms de node n'és una. En aquest s'indica que, segons l'ordre de consulta de les bases de dades per a obtenir l'IP del node o el seu nom, primer serà el servei de DNS (que utilitzarà el fitxer */etc/resolv.conf* per a determinar l'IP del node DNS) i, si no el pot obtenir, utilitzarà el de les bases de dades local *(/etc/hosts/*). Altres opcions per a això podrien ser *nis* i *nisplus*, que són altres serveis d'informació que descriu-

rem en unitats posteriors. El comportament de cada consulta també es pot controlar per mitjà d'accions (entre claudàtors), per exemple:

```
hosts: xfn nisplus dns [NOTFOUND = return] files
```

Això indica que quan es realitzi la consulta al DNS, si no hi ha un registre per a aquesta consulta, s'ha de retornar al programa que la va fer amb un zero. Es pot utilitzar el signe d'admiració per a negar l'acció, per exemple:

```
hosts dns [!UNAVAIL = return] files
```

Nota

Exemple de *nsswitch.conf*:

```
...
hosts: dns files
...
networks: files
```

4.3. Configuració del *routing*

Un altre aspecte que cal configurar és el *routing*. Si bé hi ha el tòpic sobre la seva dificultat, generalment, es necessiten uns requisits de *routing* molt simples. En un node amb moltes connexions, el *routing* consisteix a decidir on cal enviar i què es rep. Un node simple (una sola connexió de xarxa) també necessita *routing*, ja que tots els nodes disposen d'un *loopback* i una connexió de xarxa (per exemple, Ethernet, PPP, SLIP...). Com ja s'ha explicat, hi ha una taula anomenada *routing table* que conté files amb diversos camps, però hi ha tres camps summament importants: adreça de destinació, interfície per on sortirà el missatge, i adreça IP, que efectuarà el pas següent en la xarxa (*gateway*).

Nota

Consulta de taules de *routing*:

```
route -n
```

o també

```
netstat -r
```

L'ordre *route* permet modificar aquesta taula per a realitzar les tasques de *routing* adequades. Quan arriba un missatge, es mira la seva adreça de destinació, es compara amb les entrades en la taula i s'envia per la interfície l'adreça de la qual coincideix més amb la destinació del paquet. Si un *gateway* s'especifica, s'envia a la interfície adequada.

Considerem, per exemple, que el nostre node està en una xarxa de classe C amb l'adreça 192.168.110.0 i té una adreça 192.168.110.23; i l'encaminador (*router*) amb connexió a Internet és el 192.168.110.3. La configuració serà la següent:

- Primer la interfície

  ```
  ifconfig eth0 192.168.110.23 netmask 255.255.255.0 up
  ```

- Més endavant, s'indica que tots els datagrames per a node amb adreces 192.168.0. * s'han d'enviar al dispositiu de xarxa

  ```
  route add -net 192.1 ethernetmask 255.255.255.0 eth0
  ```

El *-net* indica que és una ruta de xarxa però també es pot utilitzar *-host* 192.168.110.3. Aquesta configuració permetrà connectar-se a tots els nodes

dins del segment de xarxa (192.1), però, què passarà si es vol connectar amb un altre node fora d'aquest segment? Seria molt difícil tenir totes les entrades adequades per a totes les màquines a què es vol connectar. Per a simplificar aquesta tasca, hi ha el *default route*, que s'utilitza quan l'adreça de destinació no coincideix en la taula amb cap de les entrades. Una possibilitat de configuració seria la següent:

```
route add default gw 192.168.110.3 eth0
```

(el *gw* és l'IP o nom d'un *gateway* o node *router*).

Una altra manera de fer-ho és la següent:

```
ifconfig eth0 inet down deshabilitem la interfície
ifconfig lo Link encap:Local Loopback
 ... (no mostrarà cap entrada per a eth0)
route
... (no mostrarà cap entrada en la taula de rutes)
```

Després s'habilita la interfície amb una nou IP i una nova ruta:

```
ifconfig eth0 inet up 192.168.0.111 \
      netmask 255.255.0.0 broadcast 192.168.255.255
 route add -net 10.0.0.0 netmask 255.0.0.0 \
      gw 192.168.0.1 dev eth0
```

La barra (\) indica que l'ordre continua en la línia següent. El resultat:

ifconfig

```
eth0 Link encap:Ethernet HWaddr 08:00:46:7A:02:B0
        inet addr:192.168.0.111 Bcast: 192.168.255.255 Mask:255.255.0.0
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        ...
lo Link encap:Local Loopback
        inet addr:127.0.0.1 Mask:255.0.0.0
        ...
```

route

```
Kernel IP routing table
  Destination     Gateway         Genmask Flags Metric Ref Use Iface
  192.168.0.0     *                       255.255.0.0 U         0 0 0 eth0
  10.0.0.0        192.168.0.1             255.0.0.0 UG 0 0 0 eth0
```

Per a més informació, vegeu les ordres ifconfig(8) i route(8).

4.4. Configuració de l'inetd

El pas següent en la configuració de xarxa és la configuració dels servidors i serveis que permetran a un altre usuari accedir a la màquina local o als seus serveis. Els programes servidors utilitzaran els ports per a escoltar les peticions

dels clients, els quals es dirigiran a aquest servei com a IP:port. Els servidors poden funcionar de dues maneres diferents: *standalone* (en el qual el servei escolta en el port assignat i sempre es troba actiu) o per mitjà de l'*inetd*.

L'*inetd* és un servidor que controla i gestiona les connexions de xarxa dels serveis especificats en el fitxer */etc/inetd.conf*, el qual, davant d'una petició de servei, posa en marxa el servidor adequat i li transfereix la comunicació.

Dos fitxers importants han de ser configurats: `/etc/services` i `/etc/inetd.conf`. En el primer s'associen els serveis, els ports i el protocol, i en el segon, els programes servidors que respondran davant d'una petició a un port determinat. El format de */etc/services* és `name port/protocol aliases`, en què el primer camp és el nom del servei; el segon, el port en què s'atén aquest servei i el protocol que utilitza; i, el següent, un àlies del nom. Per defecte, hi ha una sèrie de serveis que ja estan preconfigurats. A continuació, es mostra un exemple de */etc/services* (# indica que el que hi ha a continuació és un comentari):

```
tcpmux      1/tcp # TCP port service multiplexer
echo        7/tcp
echo        7/udp
discard     9/tcp           sink null
discard     9/udp sink      null
systat      11/tcp          users
...
ftp         21/tcp
ssh         22/tcp # SSH Remote Login Protocol
ssh         22/udp # SSH Remote Login Protocol
telnet      23/tcp
# 24 - private
smtp        25/tcp          mail
...
```

El fitxer */etc/inetd.conf* és la configuració per al servei mestre de xarxa (*inetdserver daemon*). Cada línia conté set camps separats per espais:

```
service socket_type proto flags user server_path server_args
```

en què *service* és el servei descrit en la primera columna de */etc/services*; *socket_type* és el tipus de *socket* (valors possibles, *stream*, *dgram*, *raw*, *rdm* o *seqpacket*); *proto* és el protocol vàlid per a aquesta entrada (ha de coincidir amb el de */etc/services*); *flags* indica l'acció que s'ha de realitzar quan hi ha una nova connexió sobre un servei que està atenent una altra connexió (*wait* li diu a *inetd* que no posi en marxa un nou servidor, o *nowait* significa que *inetd* ha de posar en marxa un nou servidor);

user és l'usuari amb el qual s'identificarà qui ha posat en marxa el servei; *server_path* és el directori en què es troba el servidor; i *server_args* són arguments possibles que es passaran al servidor. Un exemple d'algunes línies de */etc/inetd.conf* és (recordem que # significa *comentari*, pel qual, si un servei té # abans del nom, significa que no està disponible):

```
...
telnet stream tcp nowait root /usr/sbin/tcpd /usr/sbin/in.telnetd
ftp stream tcp nowait root /usr/sbin/tcpd /usr/sbin/in.ftpd
# fsp dgram udp wait root /usr/sbin/tcpd /usr/sbin/in.fspd
shell stream tcp nowait root /usr/sbin/tcpd /usr/sbin/in.rshd
login stream tcp nowait root /usr/sbin/tcpd /usr/sbin/in.rlogind
# exec stream tcp nowait root /usr/sbin/tcpd /usr/sbin/in.rexecd...
...
```

A partir de Debian Woody 3.0 r1, la funcionalitat d'*inetd* s'ha reemplaçat per *xinetd* (recomanable), el qual necessita el fitxer de configuració */etc/xinetd.conf* (vegeu el final de la unitat). Si es vol posar en marxa el servei d'*inetd*, s'ha d'executar (i crear els *links* adequats en els directoris */etc/rcX.d*) */etc/init.d/inetd.real start* (vegeu el final del capítol exemple de configuracions).

A més de la configuració d'*inetd* o *xinetd*, la configuració típica dels serveis de xarxa en un entorn d'escriptori o servidor bàsic podria incloure (molts d'aquests serveis es veuran al capítol de servidors) els següents:

- **ssh**. Connexió interactiva segura com a reemplaç de telnet, i inclou dos fitxers de configuració /etc/ssh/ssh_config (per al client) i /etc/ssh/sshd_config (per al servidor).

- **exim**. Agent de transport de correu (MTA), que inclou els fitxers de configuració /etc/exim/exim.conf, /etc/mailname, /etc/aliesses, /etc/email-addresses.

- **fetchmail**. Dimoni utilitzat per a descarregar el correu d'un compte POP3 `/etc/fetchmailrc`.

- **procmail**. Programa per a filtrar i distribuir el correu local ~/.procmailrc.

- **tcpd**. Serveis de filtres de màquines i dominis habilitats i deshabilitats per a connectar-se al servidor (*wrappers*): */etc/hosts.allow*, */etc/hosts.deny*.

- **DHCP**. Servei per a la gestió (servidor) o obtenció d'IP (client), */etc/dhcp3/dhclient.conf* (client), /etc/default/dhcp3-server (servidor), */etc/dhcp3/dhcpd.conf* (servidor).

- **CVS**. Sistema de control de versions concurrents, */etc/cvs-cron.conf* i */etc/cvs-pserver.conf*.

- **NFS**. Sistema de fitxers de xarxa /etc/exports.

- **Samba**. Sistema de fitxers de xarxa i compartició d'impressores en xarxes Windows /etc/samba/smb.conf.

- **lpr.** Dimoni per al sistema d'impressió /etc/printcap (per al sistema lpr, no per a CUPS).

- **Apatxe** i **Apache2.** Servidor de web */etc/apache/** i */etc/apache2/**.

- **squid.** Servidor *proxy-caché /etc/squid/**.

4.5. Configuració addicional: *protocols* i *networks*

Hi ha altres fitxers de configuració que, en la majoria dels casos, no s'utilitzen, però que poden ser interessants. El */etc/protocols* és un fitxer que relaciona identificadors de protocols amb noms de protocols, així, els programadors poden especificar els protocols pels seus noms en els programes.

Exemple de */etc/protocols*

```
ip          0       IP              # internet protocol, pseudo protocol number
#hopopt     0       HOPOPT          # IPv6 Hop-by-Hop Option [RFC1883]
icmp        1       ICMP            # internet control message protocol
```

El fitxer */etc/networks* té una funció similar a */etc/hosts*, però indica noms de xarxa en relació amb la seva adreça IP (en aquest cas, l'ordre *route* mostrarà el nom de la xarxa i no, la seva adreça).

Exemple de */etc/networks*

```
loopnet 127.0.0.0
localnet 192.168.0.0
amprnet 44.0.0.0
...
```

4.6. Aspectes de seguretat

És important tenir en compte els aspectes de seguretat en les connexions a xarxa, ja que una font d'atacs importants es produeix per la xarxa. En la unitat corresponent a seguretat, ja es parlarà més d'aquest tema. Tanmateix, hi ha unes quantes recomanacions bàsiques que s'han de tenir en compte per a minimitzar els riscos immediatament abans i després de configurar la xarxa del nostre ordinador:

- No activar serveis en */etc/inetd.conf* que no s'utilitzaran, i inserir un # abans del nom per evitar fonts de risc.

- Modificar el fitxer */etc/ftpusers* per denegar, a certs usuaris, la connexió per *ftp* amb la seva màquina.

- Modificar el fitxer /etc/securetty per indicar (un nom per línia) des de quines terminals (per exemple, tty1 tty2 tty3 tty4) es permet la connexió del superusuari (*root*). El *root* no es podrà connectar des dels altres terminals.

- Utilitzar el programa tcpd. Aquest servidor és un *wrapper* que permet acceptar o negar un servei des d'un node determinat, i es col·loca en */etc/inetd.conf* com a intermediari d'un servei. El *tcpd* verifica unes regles d'accés a dos fitxers: */etc/hosts.allo*w i */etc/host.deny*.

Si s'accepta la connexió, posa en marxa el servei adequat passat com a argument, per exemple, la línia del servei d'ftp que hem mostrat en *inetd.conf*:

```
ftp stream tcp nowait root /usr/sbin/tcpd/usr/sbin/in.ftpd
```

Primer, *tcpd* cerca */etc/hosts.allow* i, després, */etc/hosts.deny*. El fitxer *hosts.deny* conté les regles que indiquen els nodes que no tenen accés a un servei dins d'aquesta màquina. Una configuració restrictiva és ALL: ALL, ja que només es permetrà l'accés als serveis des dels nodes declarats en */etc/hosts.allow*.

El fitxer */etc/hosts.equiv* permet l'accés a aquesta màquina sense haver d'introduir una clau d'accés (*password*). Es recomana no utilitzar aquest mecanisme i aconsellar als usuaris no utilitzar l'equivalent des del compte d'usuari per mitjà del fitxer *.rhosts*.

A Debian és important configurar */etc/security/access.conf*, el fitxer que indica les regles sobre qui i des d'on es pot connectar (*login*) a aquesta màquina. Aquest fitxer té una línia per ordre amb tres camps separats per dos punts: del tipus permís: usuaris: origen. El primer serà un +o- (accés o denegat); el segon, un nom d'usuari o usuaris, grup o *user@host*; i el tercer, un nom d'un dispositiu, node, domini, adreces de node o de xarxes, o ALL.

Exemple de access.conf

Aquesta ordre no permet *root logins* sobre *tty1*:

```
ALL EXCEPT root:tty1...
```

Permet accedir a *u1*, *u2*, *g1* i tots els de domini *remix.com*:

```
+:u1 u2 g1 .remix.com:ALL
```

4.7. Opcions de l'IP

Hi ha una sèrie d'opcions sobre el trànsit IP que és convenient esmentar. La seva configuració es realitza inicialitzant el fitxer corresponent en el directori */proc/sys/net/ipv4/*. El nom del fitxer és el mateix que el de l'ordre i, per a activar-los, s'ha de posar un 1 dins del fitxer, i 0 per a desactivar-lo.

Exemple

Per exemple, si es vol activar *ip_forward*, s'hauria d'executar el següent:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Els més utilitzats són *ip_forward*, utilitzat per al *routing* entre interfícies o amb IP Masquerading; *ip_default_ttl*, que és el temps de vida per a un paquet IP (64 mil·lisegons per defecte); *ip_bootp_agent*, variable lògica (booleà) que accepta paquets (o no) amb adreça d'origen del tipus *0.b.c.d* i destinació d'aquest node, *broadcast* o *multicast*.

4.7.1. Ordres per a la solució de problemes amb la xarxa

Si hi ha problemes en la configuració de la xarxa, es pot començar verificant la sortida de les ordres següents per obtenir una primera idea:

```
ifconfig
cat /proc/pci
cat /proc/interrupts
dmesg | more
```

Per a verificar la connexió a la xarxa, es poden utilitzar les ordres següents (ha de tenir instal·lat *netkit-ping*, *traceroute*, *dnsutils*, *iptables* i *net-tools*):

```
ping uoc.edu         # verificar la connexió a Internet
traceroute uoc.edu   # rastrejar paquets IP
ifconfig             # verificar la configuració del host
route -n             # verificar la configuració de la ruta www.uoc.edu
dig [@dns.uoc.edu]   # verificar registres de www.uoc.edu
                     # sobre el servidor dns.uoc.edu
iptables -L -n |less # verificar filtrat de paquets (kernel >=2.4)
netstat -a           # mostra tots els ports oberts
netstat -l --inet    # mostra els ports en escolta
netstat -ln --tcp    # mostrar ports tcp en escolta (numèric)
```

5. Configuració del DHCP

DHCP són les sigles de *dynamic host configuration protocol*. La seva configuració és molt simple i serveix perquè, en lloc de configurar cada node d'una xarxa individualment, es pugui fer de manera centralitzada i la seva administració sigui més fàcil. La configuració d'un client és molt fàcil, ja que només s'ha d'instal·lar un dels paquets següents: *dhcp3-client* (versió 3, Internet Software Consortium), *dhcpcd* (Yoichi Hariguchi i Sergei Viznyuk), *pump* (Red Hat), i agregar la paraula *dhcp* en l'entrada corresponent a la interfície que es vol que funcioni sota el client dhcp (per exemple, */etc/network/interfaces* ha de tenir *iface eth0 inet dhcp*...).

La configuració del servidor requereix una mica més d'atenció, però no presenta complicacions. Primer, perquè el servidor pugui servir a tots els clients DHCP (inclòs el Windows), s'han de resoldre algunes qüestions prèvies relacionades amb les adreces de *broadcast*. Per a això, primer, el servidor ha de poder enviar missatges a l'adreça 255.255.255.255, la qual cosa no és vàlida en el GNU/Linux. Per provar-ho, executa el següent:

```
route add -host 255.255.255.255 dev eth0
```

Si apareix el missatge *255.255.255.255: Unknown host*, s'ha d'afegir l'entrada següent en */etc/hosts*: *255.255.255.255 dhcp* i intentar el següent:

```
route add -host dhcp dev eth0
```

La configuració de *dhcpd* es pot realitzar amb la interfície gràfica de *linuxconf*, o bé editar */etc/dhcpd.conf*. Vegem un exemple d'aquest fitxer:

```
# Exemple de /etc/dhcpd.conf:
default-lease-time 1200;
max-lease-time 9200;
option domain-name “remix.com”;
deny unknown-clients;
deny bootp;
option broadcast-address 192.168.11.255;
option routers 192.168.11.254;
option domain-name-servers 192.168.11.1, 192.168.168.11.2;
subnet 192.168.11.0 netmask 255.255.255.0
{ not authoritative;
        range 192.168.11.1 192.168.11.254
        host marte {
                hardware ethernet 00:00:95:C7:06:4C;
                fixed address 192.168.11.146;
                option host-name “marte”;
                }
        host saturno {
                hardware ethernet 00:00:95:C7:06:44;
                fixed address 192.168.11.147;
                option host-name “saturno”;
        }
}
```

Això permetrà al servidor assignar el rang d'adreces 192.168.11.1 al 192.168.11.254, tal com es descriu cada node. Si no hi ha el segment *host* { ... } corresponent, s'assignen aleatòriament. Els IP s'assignen per un temps mínim de 1.200 segons i màxim de 9.200 (si aquests paràmetres no existeixen, s'assignen indefinidament).

Abans d'executar el servidor, s'ha de verificar si hi ha el fitxer */var/state/dhcp/dhcpd.leases* (en cas contrari, caldrà crear-lo amb *touch/var/state/dhcp/dhcpd.leases)*. Per a executar el servidor utilitzarem */usr/sbin/dhcpd* (o bé el posarem en els scripts d'inicialització). Amb */usr/sbin/dhcpd -d -f*, es podrà veure l'activitat del servidor sobre la consola del sistema. [Mou01, Rid00, KD00, Dra99].

És important no oblidar la sentència *not authoritative*, atès que altrament aquest servidor pot deixar sense funcionament altres servidors de *dhcp* que serveixin IP d'altres segments.

6. IP *aliasing*

Hi ha algunes aplicacions en què és útil configurar diverses adreces IP a un únic dispositiu de xarxa. Els ISP (*Internet service providers*) utilitzen freqüentment aquesta característica per a proveir de característiques personalitzades (per exemple, de World Wide Web i FTP) als seus usuaris. Per a això, el *kernel* ha d'estar compilat amb les opcions de Network Aliasing i IP (*aliasing support*). Un cop s'ha instal·lat el nou *kernel*, la configuració és molt fàcil. Els àlies s'afegeixen a dispositius de xarxa virtuals associats amb el nou dispositiu amb un format com el següent: *dispositiu: nombre virtual*.

Per exemple: *eth0:0, ppp0:8*

Considerem que tenim una xarxa Ethernet que suporta dues subxarxes IP diferents simultàniament, i que la nostra màquina hi vol tenir accés directe. Un exemple de configuració seria el següent:

```
ifconfig eth0 192.168.110.23 netmask 255.255.255.0 up
route add -net 192.168.110.0 netmask 255.255.255.0 eth0
ifconfig eth0:0 192.168.10.23 netmask 255.255.255.0 up
route add -net 192.168.10.0 netmask 255.255.255.0 eth0:0
```

Que significa que tindrem dos IP 192.168.110.23 i 192.168.10.23 per al mateix NIC. Per a esborrar un àlies, s'agregar un guionet (–) al final del nom (per exemple, `ifconfig eth0:0- 0)`. [Mou01, Ran05]

Un cas típic és quan es vol configurar una targeta Ethernet única perquè sigui la interfície de diferents subxarxes IP. Per exemple, suposem que es té una màquina que es troba en una xarxa LAN 192.168.0.x/24, i es vol connectar la màquina a Internet per mitjà d'una adreça IP pública proporcionada amb DHCP usant la seva targeta Ethernet. Una opció és fer com en l'exemple anterior o, també, editar el fitxer */etc/network/interfaces* de manera que inclogui una secció similar a la següent:

```
iface eth0 inet static
address 192.168.0.1
netmask 255.255.255.0
network 192.168.0.0
broadcast 192.168.0.255

iface eth0:0 inet dhcp
```

La interfície *eth0:0* és una interfície virtual i, en activar-se, també ho farà el seu pare *eth0*.

7. IP Masquerade

L'IP Masquerade és un recurs perquè un conjunt de màquines puguin utilitzar una única adreça IP. Això permet que els nodes ocults (és a dir, els que utilitzen un IP privat, per exemple, 198.162.10.1) puguin sortir cap a Internet; però no poden acceptar trucades o serveis de l'exterior directament, sinó mitjançant la màquina que té l'IP real.

Això significa que alguns serveis no funcionen (per exemple *talk*) i d'altres s'han de configurar en mode PASV (passiu) perquè funcionin (per exemple, FTP). Tanmateix, WWW, Telnet o *irc* funcionen adequadament. El *kernel* ha d'estar configurat amb les opcions següents: network firewalls, TCP/IP networking, IP:forwarding/gatewaying, IP: masquerading. Normalment, la configuració més comuna és disposar d'una màquina amb una connexió SLIP o PPP i tenir un altre dispositiu de xarxa (per exemple, una targeta Ethernet) amb una adreça de xarxa reservada. Com henm vist, i d'acord amb la RFC 1918, es poden utilitzar com a IP privats els rangs d'adreces següents (IP/Mask): 10.0.0.0/255.0.0.0, 172.16.0.0/255.240.0.0, 192.168.0.0/255.255.0.0. Els nodes que s'han d'ocultar (*masqueraded*) estaran dins d'aquesta segona xarxa. Cada una d'aquestes màquines hauria de tenir l'adreça de la màquina que realitza el *masquerade* com a *default gateway* o *router*. Sobre l'esmentada màquina podem configurar el següent:

- *Network route* per a Ethernet, considerant que la xarxa té un IP = 192.168.1.0/255.255.255.0:

```
route add -net 192.168.1.0 netmask 255.255.255.0 eth0
```

- Default route per a la resta d'Internet:

```
route add default ppp0
```

- Tots els nodes sobre la xarxa 192.168.1/24 seran *masqueraded*:

```
ipchains -A forward -s 192.168.1.0/24 -j MASQ
```

- Si s'utilitza iptables sobre un *kernel* 2.4 o superior:

```
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

Consulteu les referències en la unitat que tracta de la de seguretat sobre informació d'*ipchains* i *iptables*. [Ran05, KD00]

8. NAT amb el *kernel* 2.2 o superiors

L'IP *network address translation* (NAT) és el reemplaçament que deixa obsoletes les prestacions del GNU/Linux IP Másquerade y que aporta noves prestacions al servei. Dins de les millores introduïdes en la pila de TCP/IP del nucli 2.2 del GNU/Linux, tenim el fet que la NAT forma part del *kernel*. Per a utilitzar-la, és necessari que el *kernel* es compili amb el següent:

CONFIG_IP_ADVANCED_ROUTER, CONFIG_IP_MULTIPLE_TABLES i CONFIG_IP_ROUTE_NAT.

I si es necessita control exhaustiu de les regles NAT (per exemple, per a activar el tallafoc *firewalling*) ha d'estar també la sentència:

CONFIG_IP_FIREWALL y CONFIG_IP_ROUTE_FWMARK

Per a treballar amb aquestes noves característiques, és necessari utilitzar el programa *ip* (es pot obtenir en *ftp://ftp.inr.ac.ru/ip_routing/*). Llavors, per a traslladar adreces de datagrames d'entrada es pot utilitzar el següent:

```
ip route add nat < extaddr>[/<masklen >] via <intaddr>
```

Això farà que l'adreça de destinació d'un paquet d'entrada destinat a *extaddr* (l'adreça visible des de fora d'Internet) es transcrigui a *intaddr* (l'adreça de la seva xarxa interna per mitjà del seu *gateway*/*firewall*). El paquet s'encamina d'acord a la taula local de *route*. Es poden traslladar adreces simples o blocs. Per exemple:

```
ip route add nat 240.0.11.34 via 192.109.0.2
ip route add nat 240.0.11.32/27 via 192.109.0.0
```

El primer fa que l'adreça interna 192.109.0.2 sigui accessible com 240.0.11.34. El segon resitua (*remapping*) el *block* 192.109.0.031 a 240.0.11.3263. En aquest cas, s'han utilitzat com a exemple translacions a adreces de la classe D, E com ara 240.0. *. * a fi de no utilitzar cap adreça pública. L'usuari haurà de reemplaçar aquestes adreces (240.0.11.34 i 240.0.11.3263) per les corresponents adreces públiques a les que vulgui realitzar la translació. [Ran05]

9. Com es configura una connexió *dial-up* i PPP?

Configurar una connexió *dial-up* (connexió amb marcatge directe) sobre PPP en el GNU/Linux és molt simple. El PPP (*point to point protocol*) permet realitzar IP-links entre dos ordinadors amb un mòdem (considerem que ha de ser un mòdem suportat pel GNU/Linux, ja que no tots, especialment els interns o els coneguts com a Winmodems, es poden configurar, perquè molts necessiten programari addicional per a establir la comunicació). [Vas00, Law07, Sec00].

Abans de realitzar la configuració, s'ha de disposar de l'*init string* del mòdem (normalment, no és necessari, però si es necessita i no està disponible, es pot utilitzar ATZ, que funciona en la majoria dels mòdems, o consultar llistes especialitzades d'*init string*).

A més, es necessitaran les dades de l'ISP: identificació de connexió (*login name*), clau (*password*) i número de telèfon. També és aconsellable tenir adreces de DNS, però és opcional en les versions actuals de *pppd*. S'ha de verificar que el mòdem estigui correctament connectat. Amb un mòdem extern, s'ha d'executar *echo > /dev/ttyS0* i mirar si els llums s'encenen. En cas contrari, s'ha d'intentar amb *ttyS1* per si el mòdem està connectat al 2n. port sèrie. Amb un mòdem intern, s'ha de consultar el manual de maquinari suportat per veure si el GNU/Linux el pot reconèixer; en cas afirmatiu, podria ser necessari reconfigurar el *kernel* per a utilitzar-lo. També es pot utilitzar l'ordre *cat /proc/pci* per si es troba en el bus PCI. [PPP00]

Ara, la manera més fàcil de configurar el mòdem és amb el paquet *kppp* (ha d'instal·lar els paquets *kdenetwork-ppp** i *ppp**). Sobre un terminal, executeu */usr/bin/kppp*. Sobre la finestra, completeu les opcions següents:

Accounts ⇒ New Connection
Dial ⇒ Authentication #'PAP/CHAP'
Store Password ⇒ yes
IP ⇒ Dynamic IP Address
Autoconfiguri hostname ⇒ No
Gateway ⇒ DefaultGateway ⇒ Assign the Default Route
DNS ⇒ Configuration Automatic ⇒ Disable existing DNS
Device ⇒ttyS1(com1) o ttyS2 (com2)
Mòdem ⇒ Query Modem per a veure els resultats (si no obteniu resultats, canvieu el dispositiu *ttySx*).

Entrem *login* i *password,* i estarem connectats a Internet (per a verificar la connexió, podríeu executar *ping www.google.com*, per exemple). Aquí, s'ha utilitzat el paquet *kppp*, però també es podria utilitzar *linuxconf* o *gnomeppp* indistintament).

Una manera ràpida de configurar *pppd* a Debian consisteix a usar el programa *pppconfig*, que ve amb el paquet del mateix nom. *pppconfig* configura els fitxers com els anteriors després de formular preguntes a l'usuari per mitjà d'una interfície de menús. Una altra opció diferent per a usar *pppd* consisteix a executar-lo des de *wvdial* que ve amb el paquet *wvdial*. En comptes de fer que *pppd* executi xat per marcar i negociar la connexió, *wvdial* realitza el marcatge, la negociació inicial i, després, inicia *pppd* perquè faci la resta. En la majoria dels casos, donant només el número telefònic, el nom d'usuari i la contrasenya, *wvdial* aconsegueix establir la connexió.

Una vegada configurat el PPP perquè funcioni, per exemple, amb *mi_isp*, s'ha d'editar */etc/network/interfícies* de manera que inclogui una secció com la següent (les ordres *ifup* i *ifdown* utilitzen les ordres *pon* i *poff* per a configurar interfícies PPP):

```
iface ppp0 inet ppp
provider mi_isp
```

amb aquesta secció, *ifup ppp0* fa el següent:

```
pon mi_isp
```

Actualment, no és possible utilitzar *ifup down* per a realitzar una configuració auxiliar de les interfícies PPP. Com que *pon* desapareix abans que *pppd* hagi acabat d'establir la connexió, *ifup* executa els *scripts up* abans que la interfície PPP estigui preparada per a ser utilitzada. Fins que se solucioni aquest error, continua sent necessari realitzar una configuració posterior en */etc/ppp/ip-up* o */etc/ppp/ip-up.d/*.

Molts proveïdors de serveis d'Internet (ISP) de banda ampla utilitzen PPP per a negociar les connexions, fins i tot, quan les màquines dels clients estan connectades amb Ethernet o xarxes ATM. Això s'aconsegueix amb PPP sobre Ethernet (PPPoE), que és una tècnica per a l'encapsulament del corrent PPP dins de les trames Ethernet. Suposem que l'ISP es diu *mi_isp*: primer, cal configurar PPP i PPPoE per a *mi_isp*. La manera més fàcil de fer-ho consisteix a instal·lar el paquet *pppoeconf* i executar *pppoeconf* des de la consola. A continuació, s'edita */etc/network/interfaces* de manera que inclogui un fragment com el següent:

```
iface eth0 inet ppp
provider mi_isp
```

De vegades, sorgeixen problemes amb PPPoE relatius a la unitat de transmissió màxima (*maximum transmit unit* o MTU) en línies DSL (*digital subscriber line*). Es pot consultar el DSL-HOWTO per a més detalls. També s'ha de tenir en compte si el seu mòdem té un encaminador (*router*), ja que llavors el *mòdem/router* maneja per si mateix la connexió PPPoE i apareix del costat de la LAN com una simple porta d'enllaç Ethernet a Internet.

10. Configuració de la xarxa mitjançant *hotplug*

El paquet *hotplug* permet el suport d'arrencada en calent (s'ha de tenir instal·lat el paquet del mateix nom). El maquinari de xarxa es pot connectar en calent tant durant l'arrencada, després d'haver inserit la targeta en la màquina (una targeta PCMCIA, per exemple), com després que una utilitat com *discover* s'hagi executat i s'hagin carregat els mòduls necessaris. Quan el *kernel* detecta nou maquinari, inicialitza el controlador per al maquinari i, després, executa el programa *hotplug* per configurar-lo. Si més tard s'elimina el maquinari, torna a executar *hotplug* amb paràmetres diferents. A Debian, quan es crida el *hotplug*, aquest executa els *scripts* de */etc/hotplug/* i */etc/hotplug.d/*. El maquinari de xarxa recentment connectat és configurat pel */etc/hotplug/net.agent*. Suposem que la targeta de xarxa PCMCIA ha estat connectada, la qual cosa implica que la interfície eth0 està preparada per ser utilitzada. La sentència */etc/hotplug/net.agent* fa el següent:

```
ifup eth0=hotplug
```

Llevat que hagi afegit una interfície lògica anomenada *hotplug* en */etc/network/interfaces*, aquesta ordre no farà res. Perquè aquesta ordre configuri eth0, s'han d'afegir les línies següents al */etc/network/interfaces*:

```
mapping hotplug
script echo
```

Si només es vol que *eth0* s'activi en calent i no, altres interfícies, s'ha d'utilitzar *grep* en comptes de *echo*, com es mostra a continuació:

```
mapping hotplug
script grep
map eth0
```

ifplugd activa o desactiva una interfície segons si el maquinari subjacent està connectat a la xarxa o no. El programa pot detectar un cable connectat a una interfície Ethernet o un punt d'accés associat a una interfície Wi-Fi. Quan *ifplugd* veu que l'estat de l'enllaç ha canviat, executa un *script* que per defecte executa *ifup* o *ifdown* per a la interfície. *ifplugd* funciona en combinació amb *hotplug*. En inserir una targeta (la qual cosa significa que la interfície està preparada per a ser utilitzada), */etc/hotplug.d/net/ifplugd.hotplug* inicia una instància d'*ifplugd* per a l'esmentada interfície. Quan *ifplugd* detecta que la targeta està connectada a una xarxa, executa *ifup* per a aquesta interfície.

Per a associar una targeta Wi-Fi a un punt d'accés, és possible que s'hagi de programar amb una clau de WEP xifrat adequada. Si està utilitzant *ifplugd* per a controlar *ifup* com s'ha explicat abans, llavors, evidentment no podrà configurar la clau de xifrat utilitzant *ifup*, ja que aquesta només és crida quan ja s'ha

associat la targeta. La solució més simple és usar *waproamd*, que configura la clau de WEP xifrat segons els punts d'accés disponibles que es descobreixen mitjançant la recerca de la xarxes WiFi. Per a més informació, consulteu *man waproamd* i la informació del paquet.

11. *Virtual private network* (VPN)

Una VPN (*virtual private network*) és una xarxa que utilitza Internet per a transportar dades, però impedeix que membres externs a ella puguin accedir a les dades.

Això significa tenir, en una xarxa amb VPN, nodes units per un túnel per on viatja el trànsit i on ningú no pot interactuar amb ell. S'utilitza quan es tenen usuaris remots que accedeixen a una xarxa corporativa per mantenir la seguretat i privacitat de les dades. Per a configurar una VPN, es poden utilitzar diversos mètodes SSH (SSL), CIPE, IPSec i PPTP, que es poden consultar en la bibliografia (es recomana consultar VPN PPP-SSH HOWTO, per Scott Bronson, i VPN-HOWTO de Matthew D. Wilson). [Bro01, Wil02]

Per a dur a terme les proves de configuració, en aquest apartat s'utilitzarà l'OpenVPN, que és una solució basada en SSL VPN i es pot utilitzar per a un ampli rang de solucions, per exemple, accés remot, VPN punt a punt, xarxes Wi-Fi segures o xarxes distribuïdes empresarials. OpenVPN implementa OSI *layer* 2 o 3 utilitzant protocols SSL/TLS, i suporta autenticació basada en certificats, targetes (*smart cards*) i altres mètodes de certificació. OpenVPN no és un servidor *proxy* d'aplicacions ni opera per mitjà d'un web *browser*.

Per a analitzar-ho, utilitzarem una opció de la OpenVPN anomenada OpenVPN *for Static key configurations*, que ofereix una manera simple de configurar una VPN ideal per a proves o per a connexions punt a punt. Els seus avantatges són la simplicitat i que no és necessari un certificat X509 PKI (*public key infrastructure*) per mantenir la VPN. Els desavantatges són que només permet un client i un servidor. En no utilitzar clau pública i clau privada, hi poden haver les mateixes claus que en sessions anteriors. Hi ha d'haver una clau en mode text en cada peer, i la clau secreta s'ha d'intercanviar abans per un canal segur.

Exemple simple

En aquest exemple, es configurarà un túnel VPN sobre un servidor amb IP=10.8.0.1 i un client amb IP=10.8.0.2. La comunicació entre el client i el servidor serà encriptada sobre UDP port 1194, que és el port per defecte d'OpenVPN. Després d'instal·lar el paquet (http://openvpn.net/install.html), s'haurà de generar la clau estàtica:

```
openvpn --genkey --secret static.key
```

Després s'ha de copiar el fitxer *static.key* en l'altre *peer* sobre un canal segur (per exemple, utilitzant *ssh* o *scp*). Per exemple, el fitxer de configuració del servidor *openVPN_server*:

```
dev tun
ifconfig 10.8.0.1 10.8.0.2
secret static.key
```

El fitxer de configuració del client, per exemple, *openVPN_client:*

```
remote myremote.mydomain
dev tun
ifconfig 10.8.0.2 10.8.0.1
secret static.key
```

Abans de verificar el funcionament de la VPN, s'ha d'assegurar en el *firewall* que el port 1194 UDP està obert sobre el servidor i que la interfície virtual *tun0* utilitzada per OpenVPN no està bloquejada ni sobre el client ni sobre el servidor. Teniu en copmpte que el 90% dels problemes de connexió trobats per usuaris nous d'OpenVPN estan relacionats amb el *firewall*.

Per a verificar l'OpenVPN entre dues màquines, haureu de canviar els IP pels reals, i el domini pel que tingui, i després executar del costat servidor

```
openvpn [server config file]
```

Que donarà una sortida com a:

```
Sun Feb 6 20:46:38 2005 OpenVPN 2.0_rc12 i686-suse-linux [SSL] [LZO] [EPOLL] built on Feb 5 2005
Sun Feb 6 20:46:38 2005 Diffie-Hellman initialized with 1024 bit key
Sun Feb 6 20:46:38 2005 TLS-Auth MTU parms [ L:1542 D:138 EF:38 EB:0 ET:0 EL:0 ]
Sun Feb 6 20:46:38 2005 TUN/TAP device tun1 opened
Sun Feb 6 20:46:38 2005 /sbin/ifconfig tun1 10.8.0.1 pointopoint 10.8.0.2 mtu 1500
Sun Feb 6 20:46:38 2005 /sbin/route add -net 10.8.0.0 netmask 255.255.255.0 gw 10.8.0.2
Sun Feb 6 20:46:38 2005 Data Channel MTU parms [ L:1542 D:1450 EF:42 EB:23 ET:0 EL:0 AF:3/1 ]
Sun Feb 6 20:46:38 2005 UDPv4 link local (bound): [undef]:1194
Sun Feb 6 20:46:38 2005 UDPv4 link remote: [undef]
Sun Feb 6 20:46:38 2005 MULTI: multi_init called, r=256 v=256
Sun Feb 6 20:46:38 2005 IFCONFIG POOL: base=10.8.0.4 size=62
Sun Feb 6 20:46:38 2005 IFCONFIG POOL LIST
Sun Feb 6 20:46:38 2005 Initialization Sequence Completed
```

I del costat client:

```
openvpn [client config file]
```

Per a verificar que funciona, podríeu fer *ping 10.8.0.2* des del *server*, i *ping 10.8.0.1*, des del client. Per a més informació, consulteu http://openvpn.net/howto.html.

Per a agregar compressió sobre el *link*, s'ha d'afegir la línia següent als dos fitxers de configuració:

```
comp-lzo
```

Per protegir la connexió mitjançant un NAT *router/firewall alive* i seguir el canvis d'IP per mitjà d'un DNS, si un dels *peers* canvia, agregueu els dos fitxers de configuració:

```
keepalive 10 60
ping-timer-rem
persist-tun
persist-key
```

Per executar-se com a dimoni (*daemon*) amb els privilegis d'*user/group nobody*, agregueu els fitxers de configuració:

```
user nobody
group nobody
daemon
```

12. Configuracions avançades i eines

Hi ha un conjunt de paquets complementaris (o que substitueixen els convencionals) i eines que o bé milloren la seguretat de la màquina (recomanats en ambients hostils), o bé ajuden en la configuració de xarxa (i del sistema en general) de manera més senzilla.

Aquests paquets poden ser de gran ajuda a l'administrador de xarxa per a evitar intrusos o usuaris locals que s'excedeixen en les seves atribucions (generalment, el problema no ve de l'usuari local, sinó d'una suplantació d'identitat), o bé poden ajudar l'usuari novell a configurar adequadament els serveis.

En aquest sentit, és necessari preveure el següent:

- **Configuració avançada de TCP/IP.** Per mitjà de l'ordre *sysctl*, és possible modificar els paràmetres del *kernel* durant la seva execució o en l'inici, per ajustar-los a les necessitats del sistema. Els paràmetres susceptibles de modificar són els que es troben en el directori */proc/sys/* i es poden consultar amb *sysctl -a*. La manera més simple de modificar aquests paràmetres és amb el fitxer de configuració */etc/syscntl.conf*. Després de la modificació, s'ha de tornar a engegar la xarxa:

```
/etc/init.d/networking restart
```

En aquest apartat, veurem algunes modificacions per a millorar les prestacions de la xarxa (millores segons condicions) o la seguretat del sistema (consulteu les referències per a més detalls) [Mou01]:

```
net.ipv4.icmp_echo_ignore_all = 1
```

- No respon a paquets ICMP com, per exemple, l'ordre *ping*, que podria significar un atac DoS (*denial-of-service*).

```
net.ipv4.icmp_echo_ignore_broadcasts = 1
```

- Evita congestions de xarxa no responent el *broadcast*.

```
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.lo.accept_source_route = 0
net.ipv4.conf.eth0.accept_source_route = 0
net.ipv4.conf.default.accept_source_route = 0
```

- Inhibeix els paquets d'IP *source routing*, que podrien representar un problema de seguretat (en totes les interfícies).

```
net.ipv4.tcp_syncookies = 1
net.ipv4.conf.all.accept_redirects = 0
```

- Permet rebutjar un atac DoS per paquets SYNC, que consumiria tots els recursos del sistema i obligaria a fer un *reboot* de la màquina.

```
net.ipv4.conf.lo.accept_redirects = 0
net.ipv4.conf.eth0.accept_redirects = 0
net.ipv4.conf.default.accept_redirects = 0
```

- És útil per a evitar atacs amb CMP *redirect acceptance* (aquests paquets s'utilitzen quan el *routing* no té una ruta adequada) en totes les interfícies.

```
net.ipv4.icmp_ignore_bogus_error_responses = 1
```

- Envia alertes sobre tots els missatges erronis en la xarxa.

```
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.lo.rp_filter = 1
net.ipv4.conf.eth0.rp_filter = 1
net.ipv4.conf.default.rp_filter = 1
```

- Habilita la protecció contra l'*IP spoofing* en totes les interfícies.

```
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.lo.log_martians = 1
net.ipv4.conf.eth0.log_martians = 1
net.ipv4.conf.default.log_martians = 1
```

  Generarà *log* sobre tots els *spoofed packets*, *source routed packets* i *redirect packets*.

- Els paràmetres següents permetran que el sistema atengui millor i més ràpidament les connexions TCP.

```
net.ipv4.tcp_fin_timeout = 40      per defecte, 60
net.ipv4.tcp_keepalive_time = 3600 per defecte, 7.200
net.ipv4.tcp_window_scaling = 0
net.ipv4.tcp_sack = 0
net.ipv4.tcp_timestamps = 0        per defecte,tots a 1(habilitats)
```

- *Iptables*. Les últimes versions del GNU/Linux (*kernel* 2.4 o superiors) inclouen nous mecanismes per a construir filtres de paquets anomenats *netfilters* [Mou01]. Aquesta nova funcionalitat és gestionada per una eina denominada *iptables*, que presenta millors característiques que el seu predecessor

(*ipchains*). Com es veurà en la unitat corresponent a seguretat, és summament fàcil construir un *firewall* amb aquesta eina per a detectar els atacs més comuns i fer-los front: *scans*, DoS, IP/MAC *Spoofing*, etc. La seva activació passa per verificar, primer, que el *kernel* és 2.4 o superior, i que està configurat per donar suport d'*ipfilter*, la qual cosa farà que s'hagi de recompilar el *kernel* per a activar l'opció *network packet filtering* [*CONFIG_NETFILTER*], i totes les subopcions específiques. Les regles específiques s'han d'activar durant l'arrencada (per exemple, mitjançant el */etc/init.d* i l'enllaç adequat al directori *rc* adequat) i tenen un format similar (consulteu les referències sobre les capacitats i la sintaxi completa) al següent:

```
iptables -A Type -i Interface -p protocol -s SourceIP --
source-port Port -d DestinationIP --destination-port Port
-j Action
```

- *GnuPG*. Aquesta eina permet encriptar dades per a la seva tramesa posterior (per exemple, correu electrònic) o emmagatzemament, i també per a generar firmes digitals (compleix amb l'estàndard de la RFC2440), i no utilitza algoritmes patentats, la qual cosa significa més llibertat en l'*open source*, però pèrdua de compatibilitat amb altres eines (per exemple, PGP 2.0) que utilitzen algoritmes com l'IDEA i l'RSA. Per a la seva compilació i instal·lació, s'han de seguir les indicacions dels seus autors en http://www.gnupg.org/. En primer lloc, s'han de crear un parell de claus (pública i privada) executant com *root* l'ordre *gpg --gen-key* dues vegades i contestant les seves preguntes. Generalment, aquestes claus s'emmagatzemaran en *~/root*. El següent pas consisteix a exportar (per exemple a una pàgina web) la clau pública perquè altres usuaris la puguin utilitzar per a encriptar els correus/informació que només podrà veure l'usuari que ha generat la clau pública. Per a això, caldrà utilitzar *gpg --export -ao UID,* la qual cosa generarà un fitxer ASCII de la clau pública de l'usuari UID.

  Per a importar una clau pública d'un altre usuari, es pot utilitzar la sentència *gpg --import filename*, i, per a firmar una clau (significa indicar al sistema que s'està d'acord en el fet que la clau firmada és de qui diu ser), es pot utilitzar *gpg --sign-key UID*. Per a verificar una clau, es pot utilitzar *gpg --verify file/data*, i per a encriptar i desencriptar, *gpg -sear UID file g*, *gpg -d file*, respectivament. [Gnu]

- *Logcheck*. Una de les activitats d'un administrador de xarxa és verificar diàriament (més d'una vegada per dia) els fitxers *log* per detectar possibles atacs/intrusions o esdeveniments que en puguin donar indicis. Aquesta eina selecciona (dels fitxers *log*) informació condensada de problemes i riscos potencials i, després, envia aquesta informació al responsable, per exemple, mitjançant un correu. El paquet inclou utilitats per executar-se de manera autònoma i recordar l'última entrada verificada per a les execucions subsegüents. Per a informació sobre la configuració i instal·lació, podeu consultar les referències. [Log]

- *PortSentry* i *Tripwire*. Aquestes eines ajuden en les funcions de seguretat de l'administrador de xarxa. *PortSentry* permet detectar i respondre a accions de cerca de ports (pas previ a un atac o a un *spamming*) en temps real i prendre diverses decisions respecte a l'acció que es du a terme. *Tripwiree* és una eina que ajudarà l'administrador notificant possibles modificacions i canvis en fitxers per evitar possibles danys (majors). Aquesta eina compara les diferències entre els fitxers actuals i una base de dades generada prèviament per a detectar canvis (insercions i esborrament), la qual cosa és molt útil per a detectar possibles modificacions de fitxers vitals com, per exemple, fitxers de configuració. Consulteu les referències sobre la instal·lació i configuració d'aquestes eines. [Tri]

- *Xinetd*. Aquesta eina millora notablement l'eficiència i prestacions d'*inetd* i *tcp -wrappers*. Un dels grans avantatges de *xinetd* és que pot fer front a atacs de DoA (*denial-of-access*) mitjançant mecanismes de control per als serveis basats en la identificació d'adreces del client, en temps d'accés i temps de connexió (*logging*). No s'ha de pensar que *xinetd* és el més adequat per a tots els serveis (per exemple, FTP i SSH és millor que s'executin sols com dimonis), ja que molts d'ells generen una gran sobrecàrrega en el sistema i disposen de mecanismes d'accés segurs que no creen interrupcions en la seguretat del sistema. [Xin]

  La compilació i instal·lació és simple, només cal configurar dos fitxers: */etc/xinetd.conf* (el fitxer de configuració de *xinetd*) i */etc/rc.d/init.d/xinetd* (el fitxer d'inicialització de *xinetd*). El primer fitxer conté dues seccions: *defaults*, que és on es troben els paràmetres que s'aplicaran a tots els serveis, i *service*, que són els serveis que es posaran en marxa per mitjà de *xinetd*.

Un exemple típic de la configuració podria ser el següent:

```
# xinetd.conf
# Les opcions de configuració per defecte que s'apliquen a tots els
# servidors es poden modificar per a cada servei
defaults
{
 instances =10
 log_type = FILE /var/log/service.log
 log_on_success = HOST PID
 log_on_failure = HOST RECORD
}
# El nom del servei s'ha de trobar en /etc/services per a obtenir
# el port correcte
# Si es tracta d'un servidor/port no estàndard, usa "port = X"
service ftp
{
 socket_type = stream
 protocol = tcp
 wait = no
 user = root
 server = /usr/sbin/proftpd
 }
#service telnet
#{
# socket_type = stream
# protocol = tcp
```

```
# wait = no
# user = root
# no_access = 0.0.0.0
# only_from = 127.0.0.1
# banner_fail = /etc/telnet_fail
# server = /usr/sbin/in.telnetd
#}
service ssh
{
 socket_type = stream
 protocol = tcp
 wait = no
 user = root
 port =22
 server = /usr/sbin/sshd
 server_args = -i
}
service http
{
 socket_type = stream
 protocol = tcp
 wait = no
 user = root
 server = /usr/local/apache/bin/httpd
}
#service finger
#{
# socket_type = stream
# protocol = tcp
# wait = no
# user = root
# no_access = 0.0.0.0
# only_from = 127.0.0.1
# banner_fail = /etc/finger_fail
# server = /usr/sbin/in.fingerd
# server_args = -l
#}

# Fi de /etc/xinetd.conf
```

Els serveis comentats (#) no estaran disponibles. En la secció *defaults*, es poden inserir paràmetres com el nombre màxim de peticions simultànies d'un servei, el tipus de registre (*log*) que es vol tenir, des de quins nodes es rebran peticions per defecte, el nombre màxim de peticions per IP que s'atendrà o serveis que s'executaran com a superservidors (*imapd* o *popd*), com per exemple:

```
default {
instances = 20
log_type = SYSLOG
authpriv log_on_success = HOST
log_on_failure = HOST
only_from = 192.168.0.0/16
per_source = 3
enabled = imaps
}
```

La secció *service*, una per cada servei com, per exemple:

```
service imapd {
socket_type = stream
wait = no
user = root
server = /usr/sbin/imapd
only_from = 0.0.0.0/0 #allows every client
```

```
no_access = 192.168.0.1
instances = 30
log_on_success += DURATION USERID
log_on_failure += USERID
nice = 2
redirect = 192.168.1.1 993
#Permet redireccionar el trànsit del port 993
#cap al node 192.168.1.1
bind = 192.168.10.4
#Permet indicar a quina interfície està associat el servei per a
#evitar problemes de suplantació de servei.
}
```

El fitxer */etc/init.d/xinetd* permetrà posar en marxa el servidor (amb l'enllaç adequat, segons el nivell d'execució seleccionat, per exemple, 3, 4 i 5). És convenient canviar els atributs d'ambdós fitxers per a garantir que no són modificats o desactivats amb *chmod 700 /etc/init.d/xinetd*; *chown 0.0 /etc/init.d/xconfig*; *chmod 400 /etc/xinetd.conf*; i *chattr +i /etc/xinetd.conf*.

- *Linuxconf*. És una eina de configuració i administració d'un sistema GNU/Linux, però que ha quedat obsoleta, si bé encara es pot trobar en algunes distribucions. Per a més informació, consulteu http://www.solucorp.qc.ca/linuxconf/.

- *Webmin*. És una altra eina (paquets *webmin-core*, *webmin-dhcp*, *webmin-inetd*, *webmin-sshd*...) que permet, amb una interfície web (és necessari tenir, per exemple, el servidor Apache instal·lat), configurar i afegir aspectes relacionats amb la xarxa. Si bé es continua el seu desenvolupament en moltes distribucions, no s'inclou per defecte. Podeu trobar més informació en http://www.webmin.com/. Per a executar-la una vegada instal·lada des d'un navegador, s'ha de cridar l'URL https://localhost:10000, que sol·licitarà l'acceptació del certificat SSL, l'usuari (inicialment *root*) i la seva clau (*passwd*).

- *System-config-**. A Fedora hi ha una varietat d'eines gràfiques que s'anomenen *system-config-"alguna-cosa"* i en què "alguna-cosa" indica perquè estan dissenyades. En general, si s'està en un entorn gràfic, es pot arribar a cada eina per mitjà d'un menú, tanmateix, cada una implica un menú a recordar. Una eina que centralitza totes les system config és *system-config-control* en una sola entrada de menú i en una única interfície gràfica des de la qual es pot seleccionar d'acord amb una organització d'icones. Per a això, és necessari Applications -> Add/Remove Software, i aquest s'arrenca com a *root* en el gestor gràfic de programari Pirut (s'ha de tenir habilitat el repositori Fedora Extras). En la interfície de Pirut, s'utilitza, per exemple, la recerca de paquets disponibles amb el patró *system-config-**. Feu la vostra selecció de *system-config-control** i feu un clic en Apply. Entre altres opcions, allà es podran configurar gairebé tots els aspectes de xarxa i serveis.

- *Networkmanager*. És una eina que permet manejar fàcilment xarxes sense fil i per cable de manera simple i sense grans complicacions, però no és indicat per a servidors (només per a màquines d'escriptori). La instal·lació és molt fàcil: *apt-get install network-manager-xx*, en què *xx* és gnome o *kde* se-

gons l'escriptori instal·lat. Per a configurar-lo, s'han de comentar totes les entrades en (Debian) */etc/network/interfaces*, excepte la interfície de *loopback interface*, per exemple, deixant només el següent:

```
auto lo
iface lo inet loopback
```

Aquest pas no és obligatori, però accelera el descobriment de les xarxes/interfícies. Sobre Debian, també s'ha d'agregar un pas extra, i és que l'usuari s'ha d'integrar dins del grup *netdev* per una qüestió de permisos. Per a això, caldrà executar (com *root* o, si no, amb l'ordre *sudo* per davant) *adduser usuari_actual netdev* i fer un *reboot* (o, també, reiniciar la xarxa amb */etc/init.d/networking restart* i fer un *logout-login* –sortir i entrar– perquè l'usuari actual quedi inclòs en el grup *netdev*).

- Altres eines (algunes estan recollides en la unitat que tracta sobre la seguretat) són les següents: Nmap (per a explorar i auditar amb finalitats de seguretat una xarxa), Nessus (per a avaluar la seguretat d'una xarxa de manera remota), Wireshark (ex-Ethereal) (analitzador de protocols de xarxa que podem trobar en http://www.wireshark.org/download.html), Snort (sistema de detecció d'intrusos, IDS), Netcat (utilitat simple i potent per a depurar i explorar una xarxa), TCPDump (monitoratge de xarxes i adquisició d'informació) i Hping2 (genera i envia paquets d'ICMP/UDP/TCP per a analitzar el funcionament d'una xarxa).

Activitats

1. Definiu els escenaris de xarxa següents:
a) Màquina aïllada.
b) Petita xarxa local (4 màquines, 1 *gateway*).
c) 2 xarxes locals segmentades (2 conjunts de 2 màquines, un *router* cada una i un *gateway* general).
d) 2 xarxes locals interconnectades (dos conjunts de 2 màquines + *gateway* cada una).
e) 2 màquines connectades per una xarxa privada virtual. Indicqueu els avantatges i desavantatges de cada configuració, per a quin tipus d'infraestructura són adequades i quins paràmetres rellevants es necessiten.

2. Configureu la xarxa de l'opció *a*, *b* i *d* del punt.

Annex. Controlant els serveis vinculats a xarxa en FC6

Un aspecte important de tots els serveis és com es posen en marxa. FC6 inclou una sèrie d'utilitats per a gestionar els serveis –dimonis– (incloent-hi els de xarxa). Com ja s'ha vist en el capítol d'administració local, el *run level* és el mode d'operació que especifica quins dimonis s'executaran. En FC podem trobar *run level* 1 (uniusuari), *run level* 2 (multiusuari), *run level* 3 (multiusuari amb xarxa), *run level* 5 (X11/ més *run level* 3). Típicament, s'executa el nivell 5 o 3 si no es necessiten interfícies gràfiques. Per a determinar quin nivell s'executa, es pot utilitzar */sbin/runlevel*, i per a saber quin nivell és el que s'arrenca per defecte, *cat/etc/inittab | grep :initdefault:*, que ens donarà informació com *id:5:initdefault:* (també es pot editar el */etc/inittab* per a canviar el valor per defecte).

Per a visualitzar els serveis que s'executen, podem utilitzar */sbin/chkconfig –list* i, per a gestionar-los, podem utilitzar *system-config-services* en mode gràfic o *ntsysv* en la línia d'ordres. Per a habilitar serveis individuals, podem utilitzar *chkconfig;* per exemple, l'ordre següent habilita el servei *crond* per als nivells 3 i 5: */sbin/chkconfig --level 35 crond on*. Independentment de com s'hagin posat en marxa els serveis, es pot utilitzar */sbin/service –status-all* o, individualment, */sbin/service crond status* per a saber com està cada servei. També es poden gestionar (*start*, *stop*, *status*, *reload*, *restart*); per exemple, es pot fer servir *service crond stop* per a parar-lo o *service crond restart* per a reiniciar-lo.

És important no deshabilitar els serveis següents (tret que se sàpiga bé el que es fa): *acpid*, *haldaemon*, *messagebus*, *klogd*, *network*, *syslogd*. Els serveis més importants vinculats a la xarxa (si bé no és una llista exhaustiva, es recullen la majoria) són els següents:

- **NetworkManager**, **NetworkManagerDispatcher**. És un dimoni que permet canviar entre xarxes fàcilment (Wi-Fi i Ethernet, bàsicament). Si només té una xarxa no és necessari que s'executi.

- **avahi-daemon**, **avahi-dnsconfd**. És una implementació de *zeroconf* i és útil per a detectar dispositius i serveis sobre xarxes locals sense DNS (és el mateix que *mDNS*).

- *bluetooth*, *hcid*, *hidd*, *sdpd*, *dund*, *pand*. Bluetooth es una xarxa sense fil, és per a dispositius portàtils (no és wifi 802.11). Per exemple, teclats ratolí, telèfons, altaveus, auriculars, etc.

- **capi**, **isdn**. Xarxa basada en maquinari ISDN (XDSI en català).

- **Iptables**: És el servei de *firewall* estàndard del Linux. És imprescindible per seguretat si es té connexió a xarxa (cable, DSL, T1).

- **Ip6tables**: Igual que l'anterior, però per al protocol i xarxes basades en Ipv6.

- **Netplugd**. Netplugd pot monitoritzar la xarxa i executar una ordre quan el seu estat canviï.

- ***netfs***. S'utilitza per a muntar automàticament sistemes de fitxers per la xarxa (NFS, Samba, etc.) durant l'arrencada.

- ***nfs, nfslock***. Són els dimonis estàndard per a compartir sistemes de fitxers per la xarxa en sistemes operatius estil Unix/Linux/BSD.

- ***ntpd***. És un servidor d'hora i data per la xarxa.

- ***portmap***. És un servei complementari per a NFS (*file sharing*) i NIS (*authentication*).

- ***rpcgssd, rpcidmapd, rpcsvcgssd***. S'utilitza per a NFS v4 (nova versió de NFS).

- ***sendmail***. Aquest servei permet gestionar els correus (MTA) o donar suport a serveis com IMAP o POP3.

- ***smb***. Aquest dimoni permet compartir fitxers sobre sistemes Windows.

- ***sshd***. SSH permet, a d'altres usuaris, connectar-se interactivament de manera segura a la màquina local.

- ***yum-updatesd***. És un servei d'actualitzacions per xarxa d'FC.

- ***xinetd***. És un servei alternatiu d'*inetd* que presenta un conjunt de característiques i millores com, per exemple, llançar múltiples serveis pel mateix port (és possible que aquest servei no estigui instal·lat per defecte).