Administració de xarxa

Remo Suppi Boldrito

PID_00215353

Universitat Oberta de Catalunya

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, el 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) amb enllaços multipunt a diferents velocitats (56 kbps, fins a 1/10 Gbps), han anat sorgint nous serveis basats en protocols més ràpids, portables entre diferents ordinadors i més ben adaptats, com el TCP/IP (*transport control program / Internet protocol*) [Com01, Mal96, Cis00, Gar98, KD00].

TCP/IP

V. Cerf i altres (1976) en el seu document original *Specification of Internet Transmission Control Program* utilitza per primera vegada les sigles TCP.

Molts canvis s'han produït en els últims anys amb l'increment de dispositius connectats a Internet (que ha tingut un creixement exponencial i s'han esgotat les adreces de dispositius IPv4) derivats de la proliferació de telèfons intel·ligents (*smartphones*), tauletes (però també televisors, consoles de videojocs, cotxes, etc.) i les necessitats d'amplada de banda que han provocat evolucions que encara trigaran alguns anys a estabilitzar-se.

Entre elles la més directa és el canvi de protocol IP que serà en la seva versió 6 (IPv6) i que obligarà a canviar tots els sistemes als operadors i usuaris per a admetre aquest nou protocol de connexió (i que no és compatible amb l'anterior si bé poden coexistir de manera segmentada i mitjançant ponts d'interconnexió).

L'altre concepte significatiu en els entorns de les comunicacions és el d'**Internet de les Coses** *[Internet of Things]* (que s'atribueix a Auto-ID Labs http://www.autoidlabs.org/ amb base a l'Institut Tecnològic de Massachusetts –MIT). La idea és molt simple (si bé la seva aplicació pot resultar complexa): si tots els articles (roba, aliments, llibres, cotxes, neveres, etc.) tinguessin un dispositiu d'identificació i fos possible el seu seguiment remot la vida quotidiana dels éssers humans del planeta es transformaria. Això implicaria que cada article podria ser identificat i gestionat remotament amb notables avenços en determinats camps però, d'altra banda, amb moltes incerteses. Per exemple, ja no caldria passar tots els productes per un lector de codis de barres per a, després, tornar-los a posar al carro durant la compra en un supermercat, però a l'instant es podria tenir informació sobre els hàbits de consum d'una persona identificada (per exemple, per la targeta de crèdit durant el pagament), acabaríem amb els articles desapareguts però en podríem tenir el seguiment en línia o saber què fa una persona que porta unes sabates amb connectivitat a la Xarxa.

La base de IofT està en IPv6 per a poder identificar aquests objectes (cosa que no es pot fer amb IPv4), per a poder codificar entre 100 milions i 100.000 milions d'objectes i seguir-ne el moviment. D'acord amb les dades de les grans consultores, el 2020 hi haurà entre 25.000 milions i 30.000 milions (!) de dispositius connectats a Internet i serà necessari desenvolupar noves tecnologies (o fer evolucionar les actuals), a més Wi-Fi, Bluetooth, RDFI, NFC, HSDPA com per exemple ZigBee, UWB, etc. (les anomenades PA *–personal area network–*, i per a això hi ha un grup de treball IEEE 802.15 especialitzat en *wireless personal area networks*, WPAN) perquè consumeixin menys energia i la comunicació sigui efectiva. [HeMa, LeCe, daCos, Com, Mal, Cis, Gar, KD].

1. Introducció a TCP/IP (paquet TCP/IP)

El protocol TCP/IP sintetitza un exemple d'estandardització i una voluntat de comunicació a escala 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, com són TCP, UDP, IP, ICMP, ARP.[Mal96]

La utilització més freqüent de TCP/IP per a l'usuari actualment són la connexió remota a altres ordinadors (*telnet*, SSH[1]), la utilització de fitxers remots (NFS[2]) o la transferència (FTP[3], HTTP[4]).

(1)De l'anglès *secure shell*.

(2)De l'anglès *network file system*.

(3)De l'anglès *file transfer protocol*.

(4)De l'anglès *hipertext markup protocol*.

1.1. Serveis sobre TCP/IP

Els serveis TCP/IP tradicionals (i que encara existeixen o han evolucionat) més importants són [Gar98]:

a) Transferència d'arxius: l'FTP permet a un usuari d'un ordinador obtenir o enviar arxius d'un ordinador a un altre ordinador. Per a això, l'usuari haurà de tenir un compte, en l'ordinador remot, identificar-se per mitjà del seu nom (*login*) i una paraula clau (*contrasenya*) o en ordinadors en què hi ha un repositori d'informació (programari, documentació...), l'usuari es connectarà com a anònim (*anonymous*) per a transferir (llegir) aquests arxius al seu ordinador. Això no és el mateix que els sistemes d'arxius de xarxa més recents, NFS (o protocols Netbios sobre TCP/IP), que permeten virtualitzar el sistema d'arxius d'una màquina perquè pugui ser accedit de manera interactiva sobre un altre ordinador. Avui dia és un protocol poc utilitzat, ja que protocols com WebDAV sobre http han permès que la transferència d'arxius es faci de manera més simple i sense aplicacions específiques més enllà d'un navegador i un servidor d'http.

b) 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 tot és executat sobre aquest alhora que l'ordinador local roman invisible des del punt de vista de la sessió. Aquest servei actualment s'ha reemplaçat per l'SHH per raons de seguretat. En una connexió remota mitjançant *telnet*, els missatges circulen com a text pla; és a dir, si algú "observa" els missatges a la xarxa, equivaldrà a mirar la pantalla de l'usuari. SSH codifica la informació (que significa un cost afegit a la comunicació), que fa que els paquets a la xarxa siguin illegibles per a un observador

extern. Amb la millora de les velocitats de xarxa i dels processadors dels dispositius no té pràcticament impacte sobre la comunicació i si és extremadament necessari quan s'envia informació confidencial o es vol mantenir la privacitat.

TCP/IP

Utilització típica de TCP/IP remote login (actualment, per qüestions de seguretat, telnet ha estat substituït per ssh):

ssh remote_host

```
Linux 3.2.0-4-amd64 x86_64
Last login: Fri May 9 17:03:26 2014 from master.hpc.local
$
```

c) Correu electrònic: 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 permeten que els correus electrònics siguin enviats a un servidor central perquè després puguin ser 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 sobre 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. Tots aquests serveis generalment són oferts dins de TCP/IP:

- **Sistemes d'arxius en xarxa** (NFS): permet a un sistema accedir als arxius sobre un sistema remot d'una manera més integrada que FTP. Els dispositius d'emmagatzematge (o part d'aquests) són exportats cap al sistema que vol accedir i els poden "veure" com si fossin dispositius locals. Aquest protocol permet a qui exporta posar les regles i les formes d'accés, cosa que (ben configurada) fa independent el lloc on es troba la informació físicament del lloc on es "veu" la informació.

- **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 fer aquesta execució: o bé per mitjà d'una instrucció (*rsh, ssh, rexec*) o per mitjà de sistemes amb RPC[5], que permet a un programa en un ordinador local executar 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).

(5)De l'anglès *remote procedure call*.

- **Servidors de nom** (*name servers*): en grans instal·lacions hi ha un conjunt de dades que necessiten ser centralitzades per a millorar-ne la utilització,

(6)De l'anglès *domain name system*.

com per exemple, nom d'usuaris, 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 està disponible per a la majoria de Unix. El DNS[6] és un altre servei de noms, però que guarda una relació entre el nom de la màquina i la identificació lògica d'aquesta màquina (adreça IP).

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

- **Servidors de terminals gràfiques** (*network-oriented window systems*): permeten que un ordinador pugui visualitzar informació gràfica sobre una pantalla que està connectada a un altre ordinador. El més comú d'aquests sistemes és X-Window i funciona mitjançant un *display manger* (dm).

No obstant això, en l'última dècada han proliferat els serveis que s'ofereixen en TCP/IP per a donar resposta a les necessitats tant dels usuaris individuals com dels serveis de grans instal·lacions. Entre els més importants per a sistemes *nix podem enumerar:

Protocol/servei	Descripció
autofs, amd	Muntat automàtic de discos per la xarxa.
arpwatch	Base de dades sobre les adreces física dels controladors Ethernet (adreces MAC) que es veuen en una xarxa i les adreces lògiques associades a ella (IP).
audit	Permet guardar informació remota amb finalitats d'auditoria.
bluetooth	Comunicacions a través del protocol del mateix nom.
bootparamd/tftp	Permet a màquines sense disc obtenir els paràmetres de xarxa i del SO.
cups	Servei d'impression per xarxa.
cvs/subversion	Sistema de versions concurrents.
inetd/xinetd:	Superdaemon de xarxa que centralitza un conjunt de serveis i aplica filtres.
imap/pop	*Servei d'Internet Message Access Protocol* per a l'accés al sistema de correu remot.
dhcp	*Dynamic Host Configuration* proveeix d'informació sobre paràmetres de la xarxa a clients en una subxarxa.

Protocol/servei	Descripció
firewall	*Packet Filtering firewall* utilitzat per gestionar i filtrar tots els paquets IP en *el kernel* de Linux (per exemple iptables/shorewall)
heartbeat	*High Availability Services* per incrementar la redundància en serveis crítics.
http	Servei que implementa el protocol http (i uns altres) per a la gestió de serveis web.
ipmi	Gestió remota de màquines a través d'OpenIPMI.
ipsec, kerberos, ssl/ssh	Protocols/serveis per permetre permet comunicacions encriptades i autenticació.
iscsi	Gestió i accés a discos via el protocol iSCSI sobre una xarxa.
ldap	*Lightweight Directory Access Protocol*, proveeix serveis d'accés a informació en xarxes de gran grandària.
named	*Domain Name System* (per exemple bind).
netdump:	Enviament d'informació sobre la xarxa per a situacions de diagnòstic quan un *kernel* deixa de funcionar.
netfs/nfs/smb	Muntat de disc en xarxa (*Network File System* (NFS), Windows (SMB))
ntalk	Servei de xat.
ntpd	Serveis de sincronització de rellotges.
portmap	*DARPA2RPC- number-mapper* utilitzat per *Remote Procedure Call (RPC)* en protocols com NFS.
proxy	Serveis *de Proxy Caching Server* para http i ftp (per exemple squid).
rsync	*Remote Synchronization* per a serveis de resguard d'arxius (*backups*).
snmpd	Implementa *Simple Network management Protocol* per obtenir informació/gestionar dispositius connectats a un xarxa.
sqld	Servei de bases de dades per xarxa (per exemple mysqld/postgresqld)
vncserver	Servei per a *Virtual Network Computing* i utilització de *Remote Desktop Sharing*.
ypbind/yperserv	Serveis que implementen NIS en sistemes GNU/Linux.
zeroconfDNS	Publica i obté informació de la xarxa quan no existeix un servei de DNS (per exemple, mdnsresponder).

1.2. Què és TCP/IP?

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

D'una banda, TCP[7] defineix les regles de comunicació perquè un ordinador (*amfitrió*) 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). TCP és orientat a connexió, és a dir, equivalent a un telèfon, i la comunicació es tracta com un flux de dades (*stream*).

(7)De l'anglès *Transmission Control Protocol*.

D'altra banda, IP[8], 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 mitjà de les xarxes. Correspon a la capa 3 del model OSI/ISO i és un protocol sense connexió (vegeu la taula següent). [Com01, Rid00, Dra99]

(8)De l'anglès *Internet Protocol*.

Una alternativa al TCP la conforma el protocol UDP[9], el qual tracta les dades com un missatge (*datagrama*) i envia paquets. És un protocol sense connexió[10] i té l'avantatge que exerceix menys sobrecàrrega en la xarxa que les connexions de TCP, però la comunicació no és fiable (els paquets poden no arribar o arribar duplicats).

(9)De l'anglès *User Datagram Protocol*.

(10)L'ordinador de destinació no ha d'estar escoltant necessàriament quan un ordinador estableix comunicació amb ell.

Hi ha un altre protocol alternatiu anomenat ICMP[11]. ICMP s'utilitza per a missatges d'error o control. Per exemple, si algú intenta connectar-se a un equip (*amfitrió*), l'ordinador local pot rebre un missatge ICMP que indiqui *host unreachable*. ICMP també pot ser utilitzat per a extreure informació sobre una xarxa. ICMP és similar a UDP, ja que maneja missatges (datagrames), però és més simple que UDP, ja que no té identificació de ports[12] en l'encapçalament del missatge.

(11)De l'anglès *Internet control message protocol*.

(12)Són bústies on es dipositen els paquets de dades i des d'on les aplicacions servidores llegeixen els paquets esmentats.

El model de comunicacions OSI[13]/ISO[14] és un model teòric adoptat per moltes xarxes. Hi ha set capes de comunicació, i cada una té una interfície per a comunicar-se amb l'anterior i la posterior:

(13)De l'anglès *open systems interconnection reference model*.

(14)De l'anglès *International Standards Organization*.

Nivell	Nom	Utilització
7	Aplicació	SMTP[15], 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	Enllaç	Controladors (*drivers*), transformació d'acord amb el protocol físic

(15)De l'anglès *simple mail transfer protocol*.

Nivell	Nom	Utilització
1	Físic	Ethernet, ADSL... envia del paquet físicament

Encara que el model OSI és el de referència, moltes vegades es prefereix utilitzar el model TCP/IP, que té quatre capes d'abstracció (segons es defineix en l'RFC 1122) i que sovint es compara amb el model OSI de set capes per a establir les equivalències. El model TCP/IP i els protocols relacionats són mantinguts per la *Internet Engineering Task Force* (IETF) i la subdivisió en capes que realitzen és:

- **Capa 4 o capa d'aplicació.** Aplicació, assimilable a les capes 5 (sessió), 6 (presentació) i 7 (aplicació) del model OSI. La capa d'aplicació havia d'incloure els detalls de les capes de sessió i presentació OSI. Es va crear una capa d'aplicació que maneja aspectes de representació, codificació i control de diàleg. D'acord amb el disseny de TCP/IP és la que maneja els protocols d'alt nivell i aspectes de presentació/codificació i diàleg.

- **Capa 3 o capa de transport.** Transport, assimilable a la capa 4 (transport) del model OSI. Està vinculada a protocols i a paràmetres de qualitat de servei en relació amb la confiabilitat i control d'errors. És per això que defineix dos protocols *transmission control protocol* (TCP) orientat a comunicació i lliure d'errors, i *user datagram protocol* (UDP), que és sense connexió i per tant pot haver-hi paquets duplicats o fora d'ordre.

- **Capa 2 o capa d'Internet.** Internet, assimilable a la capa 3 (xarxa) del model OSI. El seu objectiu és enviar paquets origen des de qualsevol xarxa i que aquests arribin a la destinació independentment de la ruta escollida i de les xarxes que han de recórrer per a arribar. El protocol específic d'aquesta capa es denomina *protocol Internet* (IP), el qual ha de decidir la millor ruta i adequació dels paquets per a obtenir l'objectiu desitjat.

- **Capa 1 o capa d'accés al medi.** Accés al medi, assimilable a la capa 2 (enllaç de dades o *link*) i a la capa 1 (física) del model OSI. S'ocupa de tots els aspectes que requereix un paquet IP per a realitzar realment un enllaç físic i inclou els detalls de tecnologia (per exemple LAN i WAN) i els de les capes físiques i d'enllaç de dades del model OSI.

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

Alguns dels protocols que utilitzen els serveis esmentats han estat dissenyats per Berkeley, Sun o altres organitzacions. Ells no formen part oficialment d'*Internet protocol suite* IPS). No obstant això, són implementats utilitzant TCP/

IP i, per tant, considerats com a part formal d'IPS. Una descripció dels protocols disponibles a Internet pot consultar-se en l'RFC 1011 (vegeu referències sobre RFC [IET]), que fa una llista de tots els protocols disponibles.

Com s'ha esmentat anteriorment, hi ha una nova versió del protocol IPv6 que reemplaça l'IPv4. Aquest protocol millora notablement l'anterior en temes com un nombre major de nodes, control de trànsit, seguretat o millores en 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 és conegut com a Ethernet (o FastEthernet o GigaEthernet). Els seus avantatges són el baix cost, velocitats acceptables (10, 100, o 1.000 megabits per segon, tot iq ue actualment ja hi ha dispositius comercials que assoleixen els 10Gbits/seg) i facilitat en la instal·lació.

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

Les dues primeres són obsoletes (utilitzen cable coaxial), mentre que l'última es fa per mitjà de cables (parells) trenats i connectors similars als telefònics (es coneixen com a RJ45). La connexió de parell trenat és coneguda com a 10baseT, 100baseT o 1000baseT (segons la velocitat) i utilitza repetidors anomenats concentradors (o *hubs*) com a punts d'interconnexió. La tecnologia Ethernet utilitza elements intermedis de comunicació (*concentradors, commutadors, encaminadors*) per a configurar múltiples segments de xarxa i dividir el trànsit per a millorar les prestacions de transferència d'informació. Normalment, en les grans institucions aquestes LAN Ethernet estan interconnectades per mitjà de fibra òptica amb tecnologia FDDI[16], que és molt més cara i complexa d'instal·lar, però es poden obtenir velocitats de transmissió equivalents a Ethernet i no tenen la limitació de la distància (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. [Rid, KD]

(16)De l'anglès *Fiber Distributed Data Interface*.

En el cas de les connexions domèstiques o per a petites empreses, la tecnologia de xarxes que ha tingut gran difusió és mitjançant connexions per mitjà d'un cable telefònic de coure (DSL, VDSL o ASDL[17]) o últimament per fibra òptica (FTTH, *fiber to the home*) on en qualsevol cas és necessari un dispositiu (encaminador) que converteixi aquests protocols en Ethernet (o en protocols sense fil Wi-Fi). La velocitat és un dels paràmetres de selecció entre un tipus o un altre de tecnologia i el cost comença a ser assumible per a connexions particulars sobretot per la incorporació de serveis de banda ampla, com TV digital o *video on demand*, als ja habituals. Sobre ASDL trobem opcions des de 3 Mbits a 50 Mbits depenent de la ubicació de la central i el client, i sobre fibra

(17)Respectivament, *Digital Subscriber Line, Very high bit-rate Digital Subscriber Line, Asymmetric Digital Subscriber Line*.

podem trobar velocitats entre 100 Mbits i 200 Mbits. S'ha de tenir en compte que per a aquest tipus de connexions els valors que es donen són generalment de baixada, mentre que de pujada sol ser aproximadament 1/10 del valor de baixada.

Hi ha a més un altre tipus de maquinari menys comú, però no menys interessant, com és ATM[18]. Aquest maquinari permet muntar LAN (xarxes d'àrea local) 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.

[18]De l'anglès *asynchronous transfer mode*.

Hi ha altre maquinari suportat per GNU/Linux per a la interconnexió d'ordinadors, entre els quals podem esmentar:

- **Frame Relay** o **X.25**, utilitzat en ordinadors que accedeixen o interconnecten WAN i per a servidors amb grans necessitats de transferències de dades.
- **Packet Radio**, interconnexió via ràdio amb protocols com AX.25, NetRom o Rose, o dispositius *dialing up*, que utilitzen línies en sèrie, lentes però molt barates, per mitjà d'un mòdem analògic o digital (XDSI, DSL, ADSL, etc.).

Per a virtualitzar la diversitat de maquinari sobre una xarxa, TCP/IP defineix una interfície abstracta mitjançant la qual es concentraran tots els paquets que seran enviats per un dispositiu físic (la qual cosa també significa una xarxa o un segment d'aquesta xarxa). Per això, per cada dispositiu de comunicació en la màquina estendrem una interfície corresponent en el nucli del sistema operatiu.

Ethernet

Ethernet en GNU/Linux s'anomena amb *ethx* (la *x* indica un número d'ordre començant per 0), en algunes distribucions, *emx* o *ibx*, essent aquesta última denominació per a targetes d'Infinibad, la interfície a línies en sèrie (mòdem) s'anomena amb *pppx* (per a PPP) o *slx* (per a SLIP), i per a FDDI és *fddix*. Aquests noms són utilitzats per les ordres per a configurar els seus paràmetres i assignar-los el número d'identificació que posteriorment permetrà comunicar-nos amb altres dispositius a la Xarxa.

En GNU/Linux pot implicar haver d'incloure els mòduls adequats per al dispositiu (NIC[19]) adequat (en el nucli o com a mòduls), i això significa compilar el nucli després d'haver escollit, per exemple, amb `make menuconfig`, el NIC adequat, indicant-lo com a intern o com a mòdul (en aquest últim cas s'haurà de compilar el mòdul adequat també).

[19]De l'anglès *network interface card*.

Els dispositius de xarxa es poden mirar en el directori */dev*, que és on hi ha un arxiu (especial, ja sigui de bloc o de caràcters, segons la seva transferència), que representa cada dispositiu maquinari [KD, Dra]. També es poden observar a */proc/net* els dispositius configurats i tota la informació (dinàmica) del seu estat i configuració. Per exemple, */proc/net/dev* dóna la informació instantània dels paquets enviats i rebuts i el seu estatus per cada interfície (s'han omès les columnes amb 0):

ifconfig -a

Per a veure les interfícies de xarxa disponibles cal executar l'ordre *ifconfig -a*. Aquesta ordre mostra totes les interfícies/paràmetres per defecte de cada una.

Interface	Receive			Transmit		
	bytes	packets	errs	bytes	packets	errs
el	530998295	1272798	0	530998295	1272798	0
eth1	201192	5984	0	2827048	66035	0
eth0	18723180	24837	0	1123703	16419	0

2. Conceptes en TCP/IP

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

- **Internet/intranets:** el terme *intranet* es refereix a l'aplicació de tecnologies d'Internet (xarxa de xarxes) dins d'una organització, bàsicament per distribuir i tenir disponible informació dins de la companyia. Per exemple, els serveis oferts per GNU/Linux com a serveis Internet i intranet inclouen correu electrònic, WWW, grups de notícies, etc. i generalment són muntats sobre adreces IP privades (es veurà més endavant), per la qual cosa aquestes màquines no són reconegudes des d'Internet i la seva sortida cap a Internet és mitjançant un encaminador (en la majoria dels casos passa el mateix amb màquines que tenim en un entorn domèstic). Si és necessari que algun d'aquests serveis tingui accés des de fora de la institució, és necessari posar serveis de Proxy o encaminadors que puguin redireccionar els paquets cap al servidor intern.

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

- **Adreça de xarxa Ethernet** (*Ethernet address* o *MAC[20] address*): un número de 48 bits (per exemple 00:88:40:73:AB:FF –en octal–, o 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 fabricant (aquest número ha de ser únic al món, per la qual cosa cada fabricant de NIC té un rang preassignat). S'utilitza en la capa 2 del model OSI i és possible tenir-ne 2^{48}; és a dir, 281.474.976.710.656.

- **Nom de l'amfitrió:** cada node ha de tenir a més un únic nom a la xarxa. Poden ser només noms 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 resulta fàcil en petites xarxes, i més dificultós en xarxes extenses, i impossible a Internet si no es fa algun control. Els noms han de ser d'un màxim de 32 caràcters, han d'usar a-zA-Z0-9.-, no han de contenir espais o # i han de començar per un caràcter alfabètic.

[20] MAC: *media access control*.

Nota

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

- **Adreça d'Internet** (*IP address*): està compost per un conjunt de nombres i dependrà de la versió del protocol IP i s'utilitza universalment per a identificar els ordinadors sobre una xarxa o Internet. Per a la versió 4 (IPV4) està format per quatre nombres en el rang 0-255 (32 bits) separats per punts (per exemple, 192.168.0.1), la qual cosa possibilita 4.294.967.296 (2^{32}) adreces de *host* diferents, cosa que s'ha mostrat insuficient sobretot perquè cada individu disposa de més d'un ordinador, tauleta, telèfon, PDA, etc. Per a la versió 6 (IPv6) l'adreça és de 128 bits i s'agrupen en quatre dígits hexadecimals formant vuit grups. Per exemple, fe80:0db8:85a3:08d3:1319:7b2i:0470:0534 és una adreça IPv6 vàlida. Es pot comprimir un grup de quatre dígits si aquest és nul (0000). Per exemple, fe80:0db8:0000:08d3:1319:7b2i:0470:0534=fe80:0db8::08d3:1319:7b2i:0470:0534. Seguint aquesta regla, si dos grups consecutius són nuls es poden agrupar, per exemple fe80:0db8:0000:0000:0000:0000:0470:0534=fe80:0db8::0470:0534. S'ha d'anar amb compte, ja que per exemple fe80:0000:0000:0000:1319:0000:0000:0534 no es pot resumir com a fe80::1319::0534 perquè no se sap la quantitat de grups nuls de cada costat. També els zeros inicials es poden ometre quedant fe80:0db8::0470:0534 com fe80:db8::470:534. Per tant, en IPv6 s'admeten 340.282.366.920.938.463.463.374.607.431.768.211.456 adreces (2^{128} o 340 sextilions d'adreces), la qual cosa significa aproximadament $6{,}7 \times 10^{17}$ (670 mil bilions) adreces per cada mil·límetre quadrat de la superfície de la Terra que es demostra amb un nombre prou gran per a no tenir les limitacions d'IPv4.
  La translació de noms en adreces IP la fa un servidor DNS (*domain name system*) que transforma els noms de node (llegibles per humans) en adreces IP (`named` és un dels serveis que en GNU/Linux realitza aquesta conversió)

Nota

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

- **Port** (*port*): 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 aquestes mateixes màquines poden tenir una comunicació FTP pel port 21). Es poden tenir diferents aplicacions comunicant-se entre dos nodes per mitjà de diferents ports simultàniament.

Nota

Ports preassignats en Unix: `more /etc/services`. Aquesta ordre mostra els ports predefinits per ordre, i segons si suporten TCP o UDP.

- **Node encaminador** (passarel·la o *gateway*): és un node que fa encaminaments (transferència de dades *routing*). Un encaminador o *router*, segons les seves característiques, podrà transferir informació entre dues xarxes de protocols similars o diferents, i pot ser, a més, selectiu.

Nota

Visualització de la configuració de l'encaminament: **netstat -r**.

- ***Domain name system*** (DNS): permet assegurar un únic nom i facilitar l'administració de les bases de dades que fan 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, militar (govern); org, sense finalitat de lucre; dues lletres per a un país, o en casos especials tres lletres, com cat, llengua i cultura catalana...). El segon nivell representa l'organització, el tercer i els restants els departaments, seccions o divisions dins d'una organització (per exemple, www.uoc.edu o pirulo@nteum.remix.cat). Els dos primers noms (de dreta a esquerra, uoc.edu en el primer cas, remix.cat en el segon, han de ser assignats (aprovats) per l'Internet Network Information Center (NIC, òrgan mundial gestor d'Internet) i els restants poden ser configurats o assignats per la institució.

  Una regla que regeix aquests noms és la FQDN (*fully qualified domain name*), que inclou el nom d'un ordinador i el nom de domini associat a aquest equip. Per exemple, si tenim el *host* que té per nom *nteum* i el nom de domini *remix.cat.*, el FQDN serà nteum.remix.cat. En els sistemes de nom de domini de zones, i més especialment en els FQDN, els noms de domini s'han d'especificar amb un punt al final del nom.

  En algunes xarxes en què no es disposa de DNS o no s'hi té accés, es pot utilitzar l'arxiu */etc/hosts* (el qual ha d'estar tots els dispositius de la xarxa) i ha de complir el mateix objectiu, o també es pot instal·lar el servei mDNS (per exemple Avahi), que implementa el que es denomina *zero-configuration networking* (*zeroconf*) per a la configuració per *multicast* de DNS/DNS-SD. Aquest servei permet als programes publicar i descobrir serveis i *hosts* sobre una xarxa local. Per exemple, un usuari pot connectar l'ordinador a la xarxa local i automàticament el servei mDNS descobrirà impressores, arxius, *hosts*, usuaris o serveis (Bonjour és un servei equivalent en sistemes Mac/W).

Nota

El nostre domini i servidor de DNS: **more /etc/default domain; more /etc/resolv.conf**.

- **DHCP, bootp:** DHCP i *bootp* són protocols que permeten a un node client obtenir informació de la xarxa (com l'adreça IP del node, la màscara, la passarel·la, DNS, etc.). 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 són descobertes automàticament per mitjà de dos protocols membres d'IPS: ARP[21] i RARP[22]. ARP utilitza missatges de difusió (*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). RARP utilitza missatges de difusió (missatge que arriba a tots els nodes) per a determinar l'adreça de xarxa associada amb una adreça de maquinari en particular. RARP és especialment important

(21)De l'anglès *address resolution protocol*.

(22)De l'anglès *reverse address resolution protocol*.

en màquines sense disc, en les quals l'adreça de xarxa generalment no es coneix en el moment de l'inici (*boot*).

- **Biblioteca de sòcols (sockets):** a Unix tota la implementació de TCP/IP forma part del nucli del sistema operatiu (o bé a dins o com un mòdul que es carrega en el moment de l'inici, com el cas de GNU/Linux amb els controladors de dispositius).

La manera d'utilitzar-les per part d'un programador és per mitjà de l'API[23] que implementa aquest sistema operatiu. Per a TCP/IP, l'API més comuna és la Berkeley Socket Library (Windows utilitza una biblioteca equivalent que es diu Winsocks). Aquesta biblioteca permet crear un punt de comunicació (sòcol), associar-lo a una adreça d'un node remot/port (vincle) 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 forma més general de comunicació (família AF INET), comunicacions més optimitzades per a casos en els quals els processos es comuniquen a la màquina mateixa (família AF Unix). En GNU/Linux, la biblioteca de sòcols és part de la biblioteca estàndard de C, Libc (Libc6 en les versions actuals), i suporta AF_INET, AF_UNIX en els seus protocols TCP/UDP i d'altres com ara AF_IPX (per a xarxes Novell), AF_X25 (per a X.25), AF_ATMPVC-AF_ATMSVC (per a ATM), AF_ROSE (per a l'Amateur Radio Protocol), etc.

(23)De l'anglès *application programming interface.*

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

Al principi, aquesta adreça era assignada per l'Internet Network Information Center, que va ser l'organisme governamental d'Internet responsable dels noms de domini i les adreces IP (fins al 18/9/1998). Posteriorment, aquest rol va ser assumit per la **Internet Corporation for Assigned Names and Numbers** (ICANN), accessible des de http://internic.net/NIC.

Per a aquest apartat haurem de separar les definicions en relació amb el protocol IPv4 i el protocol IPv6.

3.1. IPv4

L'adreça IP en IPv4 té dos camps: l'esquerre representa la identificació de la xarxa i el dret la identificació del node. Considerant el que hem esmentat anteriorment (4 nombres entre 0-255, o sigui 32 bits o 4 bytes), cada byte representa o bé la xarxa o bé el node. La part de xarxa és assignada per l'ICANN, 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 és reservat per a l'encaminament per defecte i **127** (per exemple, 127.0.0.1) és reservat per a l'autoreferència (*local loopback* o *local host*), **0** en la part de node es refereix a aquesta xarxa (per exemple, 192.168.0.0) i 255 és reservat per a paquets de tramesa a totes les màquines (difusió) (per exemple, 198.162.255.255).

En les diferents assignacions es poden tenir diferents tipus de xarxes o adreces:

- **Classe A** (*xarxa.amfitrió.amfitrió.amfitrió*): 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** (*xarxa.xarxa.amfitrió.amfitrió*): 128.1.0.1 a 191.255.254.254 (16K xarxes, 65K 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** (*xarxa.xarxa.xarxa.amfitrió*): 192.1.1.1 a 223.255.255.254 (2 milions de bits de xarxes, 254 de nodes). El patró binari és **110** + 21 bits xarxa + 8 bits de nodes.

- **Classe D i E** (*xarxa.xarxa.xarxa.amfitrió*): 224.1.1.1 a 255.255.255.254, reservat per a multidestinació (des d'un node a un conjunt de nodes que formen part d'un grup) i propòsits experimentals.

Alguns rangs d'adreces han estat reservats perquè no corresponguin a xarxes públiques, sinó a xarxes privades, i els missatges no seran encaminats per mitjà d'Internet, cosa que es coneix com a *intranets*. Aquestes són:

Xarxes privades

Màquines que es connecten entre elles sense tenir connexió amb l'exterior.

- **Classe A** des de 10.0.0.0 fins a 10.255.255.255
- **Classe B** des de 172.16.0.0 fins a 172.31.0.0
- **Classe C** des de 192.168.0.0 fins a 192.168.255.0

Un concepte associat a una adreça IP és el de *màscara*, que després ens permetrà definir subxarxes i realitzar l'encaminament dels paquets de manera automàtica entre aquestes subxarxes. Per a això és necessari definir una màscara de xarxa que seran els bits significatius de la subxarxa i que ens permetrà definir si dues IP estan dins de la mateixa xarxa o no. Per exemple, en una adreça IPv4 estàtica determinada (per exemple, 192.168.1.30), la màscara de xarxa 255.255.255.0 (és a dir, 11111111111111111111111100000000 en representació binària) indica que els primers 24 bits de l'adreça IP corresponen a l'adreça de xarxa, i els altres 8 són específics de la màquina. En IPv6 i atès que són 128 bits, solament s'expressarà la quantitat d'1 (notació que també s'utilitza en IPv4), per a facilitar-ne la lectura. En l'exemple anterior per a IPv4 seria 24, i generalment es posaria 192.168.1.30/24, i en IPv6 per exemple per a l'adreça fe80:0db8::0470:0534 es podria expressar la màscara com a fe80:0db8::0470:0534/96, en què indica que per a aquesta adreça els 96 primers bits corresponen a la xarxa.

L'adreça de difusió és especial, ja que cada node en una xarxa escolta tots els missatges (a més de la seva adreça pròpia). Aquesta adreça permet que datagrames, generalment informació d'encaminament (o *routing*) i missatges d'avís, puguin ser enviats a una xarxa i tots els nodes del mateix segment de xarxa els puguin llegir. Per exemple, quan ARP busca l'adreça Ethernet corresponent a una IP, utilitza un missatge de difusió (o *broadcast*), el qual és enviat a totes les màquines de la xarxa simultàniament. Cada node a la xarxa llegeix aquest missatge i compara la IP que es busca amb la pròpia i retorna un missatge al node que va fer la pregunta si hi ha coincidència. Per exemple, en una xarxa 192.168.1.0/24, l'adreça de difusió és 192.168.1.255.

3.2. IPv6

Els tipus d'adreces IPv6 poden identificar-se tenint en compte els rangs definits pels primers bits de cada adreça (el valor que s'especifica després de la barra és la màscara equivalent al nombre de bits que no es consideren d'aquesta adreça).

- ::/128 L'adreça amb tot de zeros s'utilitza per a indicar l'absència d'adreça, i no s'assigna cap node.

- ::1/128 Representa l'adreça de *loopback* que pot usar un node per a enviar-se paquets a si mateix (correspon a 127.0.0.1 d'IPv4). No pot assignar-se a cap interfície física.

- ::1.2.3.4/96 L'adreça IPv4 compatible s'usa com un mecanisme de transició en les xarxes duals IPv4/IPv6 (no utilitzat).

- ::ffff:0:0/96 L'adreça IPv4 mapada s'usa com a mecanisme de transició.

- fe80::/10 El prefix de l'enllaç local específica que l'adreça només és vàlida en l'enllaç físic local.

- fec0:: El prefix local (*site-local prefix*) específica que l'adreça només és vàlida dins d'una organització local. L'RFC 3879 el va declarar obsolet i han de ser substituïts per adreces Local IPv6 Unicast.

- ff00::/8 El prefix de *multicast*. S'usa per a les adreces *multicast*.

Cal destacar que no existeixen les adreces de difusió (*broadcast*) en IPv6, i la funcionalitat pot emular-se utilitzant l'adreça *multicast* FF01::1/128.

Si l'adreça és una adreça IPv4 incrustada, els últims 32 bits poden escriure's amb base decimal com ::ffff:192.168.1.1 o ::ffff:c0a8:0101 i no s'han de confondre amb ::192.168.89.9 o ::c0a8:0101

El format ::ffff:1.2.3.4 es denomina *adreça IPv4 mapada*, i el format ::1.2.3.4 *adreça IPv4 compatible*.

Les adreces IPv4 es poden transformar fàcilment al format IPv6. Per exemple, si l'adreça decimal IPv4 és 158.109.64.1 (en hexadecimal, 0x9i6d4001), pot ser convertida a 0000:0000:0000:0000:0000:0000:9i6d:4001 o :: 9i6d:4001. En aquest cas es pot utilitzar la notació mixta IPv4 compatible que seria::158.109.64.1. Aquest tipus d'adreça IPv4 compatible s'està utilitzant molt poc encara que els estàndards no l'han declarat obsoleta.

Quan el que es vol és identificar un rang d'adreces diferenciable per mitjà dels primers bits, s'afegeix aquest nombre de bits després del caràcter de barra "/".

fe80:0db8::0470:0534/96 seria equivalent a fe80:0db8::

fe80:0db8::0674:9966/96 seria equivalent a fe80:0db8:: i també a fe80:0db8::0470:0534/96

Les adreces IPv6 es representen en el DNS mitjançant registres AAAA (també anomenats *registres de quad-A*, ja que tenen quatre vegades la longitud dels registres A per a IPv4) especificats per l'RFC 3363 (hi ha una altra visió anomenada A6, però si bé és més genèrica és més complexa i pot complicar encara més la transició entre ipv4 i ipv6).

Un dels grans problemes de la transició cap a IPv6 és l'esgotament de les adreces IPv4 i els problemes que aquest està ocasionant (mireu, per exemple, per a Europa a http://www.ripe.net/internet-coordination/ipv4-exhaustion/ipv4-available-pool-graph) i és per això que alguns països (com l'índia o la Xina) ja han començat la seva migració. Hi ha una sèrie de mecanismes que permetran la convivència i la migració progressiva tant de les xarxes com dels equips d'usuari que pot classificar-se en tres grups: doble pila, túnels, traducció.

La doble pila fa referència a una solució de nivell IP amb doble pila (RFC 4213), que implementa les piles de tots dos protocols, IPv4 i IPv6, en cada node de la xarxa. Cada node amb doble pila a la xarxa tindrà dues adreces de xarxa, una IPv4 i una altra IPv6, i com a avantatges presenta que és fàcil de desplegar i molt suportada, i com a desavantatges que la topologia de xarxa requereix dues taules d'encaminament. Els túnels permeten connectar-se a xarxes IPv6 "saltant" sobre xarxes IPv4 encapsulant els paquets IPv6 en paquets IPv4 (tenint com a capa IP següent el protocol número 41 i d'això el nom de *proto-41*). Hi ha moltes tecnologies de túnels disponibles i es diferencien en el mètode per a determinar la direcció a la sortida del túnel. La traducció és necessària quan un node que només suporta IPv4 intenta comunicar-se amb un node que només suporta IPv6 i bàsicament s'agrupen en "amb estat" (NAT-PT, RFC 2766; TCP-UDP Relay, RFC 3142; *socks-based gateway*, RFC 3089) o "sense estat" (*bump-in-the-stack; bump-in-the-API*, RFC 276).

3.3. Subxarxes i encaminament (*routing*)

Dos conceptes complementaris als descrits anteriorment són el de **subxarxes** i **encaminament** entre elles. *Subxarxes* significa subdividir la part del node en petites xarxes dins de la mateixa xarxa, per exemple, per a 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 aquest readreçament en la part de node de la IP. El nombre de bits que són interpretats com a identificador de la subxarxa és donat per una màscara de xarxa (o *netmask*) que és un nombre de 32 bits (igual que la IP).

Qui definirà quins paquets van cap a un costat o un altre serà el *host* que compleixi amb el paper d'encaminador (*router*) i que interconnectarà diversos segments de xarxes/xarxes. Generalment, l'encaminador es coneix com a *porta d'enllaç* (passarel·la) i s'utilitza com el *host* que ajuda a arribar a l'exterior (per exemple, Internet) des de la xarxa local.

Per a obtenir l'identificador de la subxarxa, s'haurà de fer una operació lògica **I** (AND) entre la màscara i la IP, la qual cosa donarà la IP de la subxarxa.

Per exemple, tenim una institució que té una xarxa classe B amb número 172.17.0.0; la seva màscara de xarxa 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 subxarxes: des de 172.17.1.0 fins a 172.17.20.0. El punt que connecta totes aquestes subxarxes (xarxa troncal) té la seva pròpia adreça, com per exemple 172.17.1.0. Aquestes subxarxes comparteixen la mateixa IP de xarxa, mentre que la tercera és utilitzada per a identificar cada una de les subxarxes que hi ha a dins (per això s'utilitzarà una màscara de xarxa 255.255.255.0).

El segon concepte, encaminament, representa la manera com els missatges són enviats per mitjà de les subxarxes. Per exemple, tenim tres departaments amb subxarxes Ethernet:

- Compres (subxarxa 172.17.2.0).
- Clients (subxarxa 172.17.4.0).
- Recursos humans, RH, (subxarxa 172.17.6.0).
- Xarxa troncal amb FFDI (subxarxa 172.17.1.0).

Per a encaminar els missatges entre els ordinadors de les tres xarxes es necessitaran tres portes d'intercanvi (passarel·les), que tindran cada una dues interfícies de xarxa per a canviar entre Ethernet i FFDI. Seran les següents:

- CompresGW IP:172.17.2.1 i 172.17.1.1,
- ClientsGW IP:172.17.4.1 i 172.17.1.2
- RHGW IP:172.17.6.1 i 172.17.1.3, és a dir, una IP cap al costat de la subxarxa i una altra cap a la xarxa troncal.

Quan s'envien missatges entre màquines de compres, no és necessari sortir a la passarel·la, 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 per les dues passarel·les respectives. Quan Compres0 "veu" que RH3 és en una altra xarxa, envia el paquet per mitjà de la passarel·la CompresGW, que al seu torn l'enviarà a RHGW, 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 de recursos humans (si bé significa un plantejament més complex i car a l'hora de dissenyar i construir la xarxa).

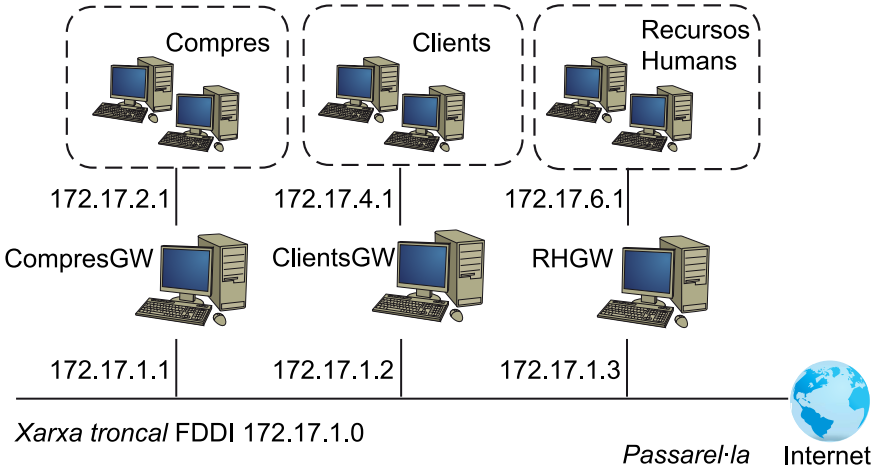

Figura 1. Configuració de segments i passarel·les en una Intranet

IP utilitza una taula per a fer l'encaminament dels paquets entre les diferents xarxes i en la qual hi ha un encaminament 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, són enviades per la passarel·la per defecte (*default gateway*) cap a la xarxa indicada. Sobre CompresGW, per exemple, la taula podria ser:

Adreça	Màscara	Passarel·la	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 "-" significa que la màquina està directament connectada i no necessita encaminament. El procediment per a identificar si es fa l'encaminament o no es du a terme és per mitjà d'una operació molt simple amb dos AND lògics (subxarxa AND màscara i origen AND màscara) i una comparació entre els dos resultats. Si són iguals no hi ha encaminament, sinó que s'ha d'enviar la màquina definida com a passarel·la a cada màquina perquè faci l'encaminament del missatge.

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

```
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 els resultats són iguals, no hi haurà encaminament. En canvi, si fem el mateix amb 172.17.2.4 cap a 172.17.6.6 podem veure que hi haurà un encaminament 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. L'encaminament, per defecte, 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 menys zeros tingui. Per a construir les taules d'encaminament, es pot utilitzar l'ordre `route` durant l'arrencada de la màquina, però si és necessari utilitzar regles més complexes (o encaminament automàtic), es pot utilitzar el RIP[24] o, entre sistemes autònoms, l'EGP[25] o també el BGP[26]. Aquests protocols s'implementen en l'ordre `gated`.

[24] De l'anglès *routing information protocol*.

[25] De l'anglès *external gateway protocol*.

[26] De l'anglès *border gateway protocol*.

Per a instal·lar una màquina sobre una xarxa existent, és necessari, per tant, disposar de la informació següent obtinguda del proveïdor de xarxa o de l'administrador: adreça IP del node, adreça de la xarxa IP, adreça de difusió, adreça de màscara de xarxa, adreça d'encaminador, 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. A continuació, es veurà com es defineix la xarxa i el node per a una xarxa privada (cal ser acurat, ja que si es té la màquina connectada a la Xarxa, es podria perjudicar un altre usuari que tingués assignada aquesta adreça).

4. Com s'ha de configurar la xarxa?

4.1. Configuració de la interfície (NIC)

En aquest apartat veurem amb més detall la configuració de la xarxa i de la xarxa wifi, dos temes que ja es van tractar en el mòdul "nivell usuari".

Una vegada carregat el nucli de GNU/Linux, aquest executa l'ordre `init`, que, al seu torn, llegeix l'arxiu de configuració */etc/inittab* i comença el procés d'inicialització. Generalment, inittab té seqüències com ara *si::sysinit:/etc/init.d/boot*, que representa el nom de l'arxiu d'instruccions (script) que controla les seqüències d'inicialització. Generalment aquest script crida altres scripts, entre els quals hi ha la inicialització de la xarxa.

Exemple

En Debian s'executa `/etc/init.d/networking` 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* està només una vegada (*/etc/init.d*) i d'acord amb els serveis que volem en aquest estat es crea un enllaç en el directori corresponent a la configuració del node-estat. Els paràmetres per defecte per a cada *script* del directori */etc/init.d* es troben en un directori específic. En Debian, per exemple, a */etc/default/* i en Fedora a */etc/sysconfig/*.

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. En cas que les interfícies no estiguin numerades correctament (sol ocórrer quan treballem amb màquines virtuals clonades d'altres màquines virtuals) és necessari esborrar l'arxiu */etc/udev/rules.d/70*net** (o equivalent) que preserva la numeració dels dispositius de xarxa mitjançant el mecanisme d'udev (que veurem més endavant).

Nota

Consulteu **man ifconfig** o **man ip** per a les diferents opcions de les ordres.

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 `ifconfig` (*interface configure*). Un exemple serà:

```
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 la 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 són configurats per defecte. En

aquest cas, el nucli configurarà aquesta màquina com a tipus C i configurarà la xarxa amb 192.168.110.23 i l'adreça de difusió amb 192.168.110.255. Per exemple:

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

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

En GNU/Linux hi ha diverses maneres de configurar la xarxa perquè l'administrador no hagi d'introduir els paràmetres de configuració en cada operació de *boot*. Una de les maneres de fer-ho, en Debian, és mitjançant les ordres esmentades anteriorment (`ifup`, `ifdown`, que s'executen automàticament, i l'arxiu */etc/network/interfaces*. Si es decideix utilitzar aquestes ordres, no cal fer res més.

Per a modificar els paràmetres[27] de xarxa de la interfície *eth0*, es pot fer:

[27] Consulteu *man interfaces* a la secció 5 del manual per a més informació del format.

- **ifdown eth0**. Atura tots els serveis de xarxa sobre *eth0*. També es pot fer `/etc/init.d/networking stop` que atura tots els serveis sobre totes les interfícies de xarxa.

- `vi /etc/network/interfaces` (es pot fer amb l'editor que es prefereixi, tot i que és aconsellable fer-ho amb el "`vi`" perquè hi és en tots els sistemes *nix).Permet editar i modificar els paràmetres corresponents.

- **ifup eth0**. Posa en marxa els serveis de xarxa sobre *eth0* (o `/etc/init.d/networking start`).

Suposem que volem 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 (passarel·la). Cal 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 tenim instal·lat el paquet `resolvconf` (en algunes distribucions, com ara Ubuntu, ve instal·lada per defecte) podem afegir línies per a especificar la informació relativa al DNS. Per exemple:

```
auto eth0
iface eth0 inet static
address 192.168.0.123
```

```
netmask 255.255.255.0
gateway 192.168.0.1
dns-search remix.cat
dns-nameservers 195.238.2.21 195.238.2.22
```

En lloc d'`auto eth0`, per exemple, també es pot utilitzar `allow-hotplug eth0`, que indica que la interfície es podrà activar amb ifup `--allow=hotplug eth0`. Les línies que comencin amb `allow-` són utilitzades per a identificar interfícies que podrien ser activades per diferents subsistemes (`allow-auto` i `auto` són sinònims).

Recordem que si la màquina disposa de diverses interfícies de xarxa, es pot repetir la secció prèvia amb el dispositiu corresponent però les tres últimes línies (*gateway, dns-search i dns-nameservers*) solament han d'estar una vegada, ja que són comunes per a totes les interfícies.

Després que s'activi la interfície, els arguments de les opcions *dns-search* i *dns-nameservers* queden disponibles per a la inclusió a */etc/resolv.conf*. L'argument *remix.cat* de l'opció *dns-search* correspon a l'argument de l'opció *search* a *resolv.conf* i els arguments 195.238.2.21 i 195.238.2.22 de l'opció *dns-nameservers* corresponen als arguments de les opcions *nameserver* a *resolv.conf*. Si el paquet *resolvconf* no està instal·lat, es pot modificar manualment l'arxiu */etc/resolv.conf* (i si està instal·lat i no s'utlitza */etc/network/interfaces* per a configurar els DNS, es poden modificar els arxius que hi ha a */etc/resolvcinf.d*).

Una configuració equivalent per DHCP; és a dir, un servidor de DHCP que ens passarà els paràmetres de conficuració de la xarxa, se simplifica de la manera següent:

```
auto eth0
iface eth0 inet dhcp
```

També es pot configurar la xarxa a baix nivell per mitjà de l'ordre `ip` (que és equivalent a `ifconfig` i `route`). Si bé aquesta ordre és molt més versàtil i potent (permet establir túnels, encaminaments alternatius, etc.), és més complexa i es recomana utilitzar els procediments anteriors per a configuracions bàsiques de la xarxa.

Nota: `resolv.conf`

Podeu consultar el manual per a les opcions de DNS *man resolv.conf*.

A continuació veurem una sèrie d'exemples per a la configuració de la xarxa per l'ordre `ip`:

- `ip addr add 192.168.1.1 dev eth0`. Defineix una IP a *eth0*.
- `ip addr show`. Mostra la configuració.

- `ip addr del 192.168.1.1/24 dev eth0`. Elimina IP d'*eth0*.
- `ip route add default via 192.168.0.1`. Agrega un *gateway*.
- `ip link set eth1 up`. Activa interfície.
- `ip link set eth1 down`. Desactiva interfície.
- `ip route show`. Mostra el *routing*.
- `ip route add 10.10.20.0/24 via 192.168.50.100 dev eth0`. Agrega una regla.
- `ip route del 10.10.20.0/24`. Esborra una regla.
- `up ip route add 10.10.20.0/24 via 192.168.1.1 dev eth1`. Per definir una regla estàtica en */etc/network/interfícies*.

Una altra manera de configurar la xarxa (recomanada per a usuaris amb mobilitat i configuracions estàndard) és amb el paquet *network manager* (NM). Aquest paquet consta d'una interfície gràfica (`nm-connection-editor`) per a la configuració dels dispositius de xarxa (i pot coexistir amb les configuracions en */etc/network/interfícies*) o es pot configurar mitjançant els arxius en i tenint en compte que cal desactivar les interfícies que volem que gestioni l'NM a */etc/network/interfícies*. L'NM no gestionarà interfícies definides a */etc/network/interfícies* sempre que a */etc/NetworkManager/NetworkManager.conf* contingui:

```
[main]
plugins=ifupdown,keyfile
[ifupdown]
managed=false
```

S'haurà de canviar `managed=true` si es vol que NM gestioni les interfícies definides a */etc/network/interfaces*. Sempre que es modifiqui l'arxiu */etc/NetworkManager/NetworkManager.conf* per a la seva configuració i després de modificar-lo s'haurà de recarregar aquest amb `nmcli amb reload` (per a aspectes detallats de la configuració consulteu `man NetworkManager.conf` o https://wiki.gnome.org/Projects/NetworkManager/SystemSettings. En algunes situacions l'NM pot generar conflictes amb alguns dispositius de xarxa que hagin estat configurats prèviament amb l'NM i després es desitgi realitzar la configuració des de */etc/network/interfaces* per la qual cosa es recomana desinstal·lar l'NM o eliminar els arxius de configuració de la interfície corresponent del directori */etc/NetworkManager/system-connections/*.

Per a configuracions d'altres dispositius de xarxa com connexions amb PPP (punt a punt) amb mòdem PSTN (*public switched telephone network*), mòdem (genèric) ADSL o compatibles amb PPPOE (*point to point pro-*

tocol over Ethernet) o PPTP amb (*point-to-point tunneling protocol*) consulteu la informació del dispositiu específic i http://qref.sourceforge.net/quick/ch-gateway.es.html o http://debian-handbook.info/browse/es-ES/stable/sect.network-config.html#sect.roaming-network-config.

4.1.1. Configuracions avançades de la xarxa

És necessari en GNU/Linux diferenciar entre una interfície física i una interfície lògica. Una interfície física és el que fins ara hem denominat *interfície* (por exemple, eth0), i una interfície lògica és un conjunt de valors (de vegades anomenats *perfils*) que poden assignar-se als paràmetres variables d'una interfície física. Les definicions iface a */etc/network/interfaces* són, en realitat, definicions d'interfícies lògiques, no d'interfícies físiques; però si no s'indica res, una interfície física es configurarà, per defecte, com a interfície lògica.

No obstant això, si tenim un ordinador portàtil que utilitzem en llocs diferents (por exemple, a casa i a la feina) i necessitem configuracions diferents per a la xarxa de cada lloc, podem fer ús de les definicions d'interfície lògica. Primer s'han de definir dues interfícies lògiques com a casa i feina (en lloc d'eth0 com s'ha fet anteriorment):

```
iface casa inet static
      address 192.168.1.30
      netmask 255.255.255.0
      gateway 192.168.1.1

iface feina inet static
      address 158,.109.65.66
      netmask 255.255.240.0
      gateway 158.109.64.1
```

D'aquesta manera, la interfície física eth0 es pot activar per a la xarxa de casa amb `ifup eth0=casa`, i per reconfigurar-la per a la feina amb `ifdown eth0; ifup eth0=feina`.

El mecanisme és molt potent i es pot ampliar mitjançant la configuració en funció d'una sèrie de condicionants utilitzant una secció *mapping*. La sintaxi d'una secció *mapping* és la següent:

```
mapping patró
        script nom_script
        [map script]
```

L'*script* anomenat en la secció *mapping* serà executat amb el nom de la interfície física com a argument i amb el contingut de totes les línies `map` de la secció. Abans de finalitzar, l'*script* mostrarà el resultat de la transformació per la sortida estàndard.

Per exemple, la secció *mapping* següent farà que `ifup` activi la interfície eth0 com a interfície lògica casa:

```
mapping eth0
        script /usr/local/sbin/echo-casa
```

en que */usr/local/sbin/echo-casa* és :

```
#!/bin/sh
echo casa
```

Això pot ser útil si, per exemple, tenim dues targetes de xarxa diferents (una per a casa i una altra per a la feina). El directori */usr/share/doc/ifupdown/examples/* conté un *script* de transformació que es pot usar per a seleccionar una interfície lògica basant-se en l'adreça MAC (*media access controller*). Primer s'ha d'instal·lar l'*script* en un directori apropiat amb:

```
install -m770 /usr/share/doc/ifupdown/examples/get-mac-address.sh /usr/local/sbin/
```

A continuació es pot afegir una secció com la següent al */etc/network/interfaces*:

```
mapping eth0
        script /usr/local/sbin/get-mac-address.sh
        map 02:23:45:3C:45:3C casa
        map 00:A3:03:63:26:93 feina
```

Hi ha altres programes de transformació més sofisticats com `guessnet` o `laptop-net`. Per exemple en el cas de `guessnet`, s'ha d'instal·lar el paquet i declarar una secció a */etc/network/interfaces*:

```
mapping eth0
        script guessnet-ifupdown
        map casa
        map feina
```

Ara en fer `ifup eth0`, `guessnet` verificarà si eth0 ha d'activar-se com a casa o feina utilitzant la informació emmagatzemada en les definicions de les interfícies lògiques.

També és possible configurar els dispositius de xarxa per a l'arrencada en calent mitjançant el paquet `hotplug`. Aquest tipus de configuració és útil quan utilitzem dispositius removibles com, per exemple, un mòdem USB. Podeu trobar més informació a [GRD].

4.1.2. Configuració de la xarxa en IPv6

En relació amb la configuració d'IPv6, els sistemes GNU/Linux incorporen aquesta funcionalitat mitjançant la seva implementació en *el kernel* o mitjançant mòduls (en Debian s'inclouen en *el kernel* i algunes arquitectures específiques mitjançant un mòdul anomenat *ipv6*). Eines bàsiques com `ping` i `traceroute` tenen els seus equivalents IPv6, `ping6` i `traceroute6`, disponibles en els paquets `iputils-ping` i `iputils-tracepath` respectivament. Una xarxa IPv6 es configura de manera similar a IPv4 amb l'arxiu */etc/network/interfaces* (cal verificar prèviament que l'encaminador és compatible amb IPv6 i que retransmeti dades a la xarxa IPv6 global):

```
iface eth0 inet6 static
address fe80:0db8::0470:0534
netmask 64
# Desactivar autoconfiguració
# autoconf 0
# L'encaminador es configura automàticament
# i no té adreça fixa (accept_ra 1). Si la tingués:
# gateway 2001:db8:1234:5::1
```

Generalment, les subxarxes IPv6 tenen una màscara de xarxa de 64 bits, la qual cosa significa que hi ha 264 adreces diferents dins de la subxarxa i permet a un mètode anomenat *autoconfiguració d'adreces sense estat* (SLAAC, *stateless address autoconfiguration*) seleccionar una adreça basada en l'adreça MAC de la interfície de xarxa. De manera predeterminada, si SLAAC està activat en la xarxa, el *kernel* trobarà encaminadors IPv6 automàticament i configurarà les interfícies de xarxa.

Aquest tipus de configuració pot tenir conseqüències en la privacitat, ja que si es canvia de xarxa freqüentment seria fàcil identificar el dispositiu en aquestes xarxes. Les extensions de privacitat d'IPv6 solucionen aquest problema i assignaran adreces addicionals generades aleatòriament a la interfície, les canviaran periòdicament i les utilitzaran per a connexions sortints, mentre que les connexions entrants podran utilitzar les adreces generades per SLAAC. Un exemple d'aquesta configuració és activar en */etc/network/interfaces*:

```
iface eth0 inet6 auto
    # Preferir les adreces assignades
    # aleatòriament per a connexions sortints.
    privext 2
```

Si no es disposa d'una connexió IPv6, el mètode alternatiu és utilitzar un túnel sobre IPv4. D'acord amb [HeMa] un proveïdor (gratuït) d'aquests túnels és http://www.gogo6.net/freenet6/tunnelbroker i per a això és necessari registrar-se amb un compte en el lloc web, instal·lar un paquet (`gogoc`) i configurar el túnel des de l'arxiu */etc/gogoc/gogoc.conf* agregant les línies *userid* i *password* (que es van obtenir després del registre per correu electrònic) i reemplaçar *server* amb *authenticated.freenet6.net*. Consulteu [HeMa] per a la configuració de la seva màquina com *a* encaminador IPv6 per a les màquines de la xarxa local.

4.1.3. Configuració de xarxa en distribucions de tipus 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, tallafocs, passarel·les i encaminadors. False per a qualsevol altre cas
GATEWAY="XXX.XXX.XXX.YYY"
         Adreça IP de la porta de sortida a Internet.
```

Per a la configuració amb DHCP s'ha de treure la línia de *gateway*, ja que serà assignada pel servidor. I en cas d'incorporar NIS cal agregar una línia amb el servidor de domini: *NISDOMAIN=NISProject1*.

Per a configurar la interfície *eth0* en l'arxiu */etc/sysconfig/network-scripts/ifcfg-eth0* (reemplaçar les X amb els valors adequats):

```
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 l'arrencada
```

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 la configuració del DHCP:

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
```

Per a deshabilitar DHCP, cal canviar `BOOTPROTO=dhcp` per `BOOTPROTO=none`. Qualsevol canvi en aquests fitxers implicarà reiniciar els serveis amb `service network restart` (o si no `/etc/init.d/network restart`).

Per a canviar el nom de l'amfitrió s'han de seguir aquests tres passos:

1) L'ordre `hostname nom-nou`.

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

3) Restaurant un dels serveis següents (o reiniciant):

- `service network restart` (o `/etc/init.d/network restart`).
- Reiniciant l'escriptori passant a mode consola amb `init 3` i canviant a mode GUI amb `init 5`.

Verificar si el nom està donat d'alta a */etc/hosts*. El nom de l'ordinador es pot canviar en temps d'execució amb `sysctl -w kernel.hostname="nom-nou"`.

4.1.4. Configuració d'una xarxa Wi-Fi (sense fil)

Recordeu que aquest tema ja es va comentar en el mòdul "Nivell usuari", però en aquest apartat en presentem alguns detalls complementaris o alternatius.

Per a la configuració d'interfícies Wi-Fi s'utilitzen 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 per mitjà de l'arxiu */etc/network/interfaces*.

Exemple: Configurar una Wi-Fi en Debian (similar en FC)

En aquest cas mostrarem els passos per a carregar els *drivers* d'una targeta Intel Pro/Wireless 2200BG com a mètode general, però en els *kernels* actuals aquests *drivers* ja estan inclosos, per la qual cosa no és necessari fer aquests passos previs però serveix d'exemple. Normalment, el programari que controla les targetes es divideix en dues parts: el mòdul programari, que es carregarà en el nucli per mitjà de l'ordre `modprobe`, i el *microprogramari*, que és el codi que es carregarà a la targeta i que ens dóna el fabricant[28]. Com que estem parlant de

[28] Consulteu la pàgina d'Intel per a aquest model o el projecte Wireless Kernel: http://wireless.kernel.org/en/users/Drivers/iwlwifi.

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 a 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·lació del 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 documentació, es descarrega la versió del microprogramari compatible amb la versió del *driver*, es descomprimeix i s'instal·la a */usr/lib/hotplug/firmaware* (on X.X és la versió de microprogramari):

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

Amb això es copiaran tres arxius (*ipw2200-bss.fw*, *ipw2200-ibss.fw* i *ipw2200-sniffer.fw*). Després 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 a continuació, 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 descarrega el paquet `wirelesstools`, que conté `iwconfig` (per exemple, amb `aptget install wirelesstools`) i executem `iwconfig;` sortirà una cosa semblant al següent:

```
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 l'arxiu */etc/network/interfaces* i seguir el procediment indicat en l'apartat 4.6 del mòdul "Nivell usuari".

4.2. Configuració del sistema de resolució de noms

El pas següent és configurar el *name resolver*, que converteix noms com *nteum.remix.cat* en 192.168.110.23.

La manera en què actua el sistema de resolució de noms depèn de la línia *hosts* de l'arxiu de configuració */etc/nsswitch.conf*. Aquesta línia fa una llista dels serveis que haurien d'usar-se per a resoldre un nom, per exemple, `dns`, `files`, `nis`, `nisplus` (vegeu `man -s 5 nsswitch.conf`). Si s'utilitza el servei `dns`, el comportament del sistema de resolució també està determinat per l'arxiu de configuració */etc/resolv.conf*, que conté les adreces IP dels servidors de noms. El seu format és molt simple (una línia de text per sentència). Hi ha tres paraules clau per a aquest fi: *domain* (domini local), *search* (llista de dominis alternatius) i *name server* (l'adreça IP del *domain name server*).

4.2.1. Exemple de l'arxiu */etc/resolv.conf*

```
domain remix.cat
search remix.cat
name server 192.168.110.1
name server 192.168.110.65
```

Aquesta llista de servidors de nom sovint depèn 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 aquesta característica no sempre funciona adequadament i de vegades pot entrar en conflicte i generar configuracions errònies. El paquet `resolvconf` (instal·lat per defecte en algunes distribucions) soluciona de manera adequada el problema i permet una configuració simple dels servidors de nom de manera dinàmica. `resolvconf` està dissenyat per a funcionar sense que sigui necessària cap configuració addicional; no obstant això, pot requerir alguna intervenció per a aconseguir que funcioni adequadament.

Nota

Per a més informació sobre el paquet `resolvconf`, podeu consultar el web explicatiu:
http://packages.debian.org/unstable/net/resolvconf

En Debian és un paquet opcional i pot instal·lar-se amb els procediments habituals. Això modificarà la configuració de */etc/resolv.conf*, que serà reemplaçada per un enllaç a */etc/resolvconf/run/resolv.conf*, i el `resolvconf` utilitzarà un arxiu que es generarà dinàmicament a */etc/resolvconf/run/resolv.conf*. S'ha de tenir en compte que el `resolvconf` només és necessari quan necessitem modificar dinàmicament els *nameserver*, però si en la nostra xarxa els *nameservers* no canvien freqüentment el */etc/resolv.conf* pot ser el més adequat.

Quan `resolvconf` està instal·lat no és necessari modificar */etc/resolv.conf*, ja que aquest serà regenerat automàticament pel sistema. Si és necessari definir *nameservers* es podria agregar com a *dns-nameservers* a */etc/network/interfícies* (generalment després de la línia *de gateway*).

4.2.2. L'arxiu *host.conf*

L'arxiu */etc/host.conf* conté informació específica per a la resolució de noms. La seva importància resideix a indicar on es resol primer l'adreça o el nom d'un node. Aquesta consulta pot fer-se al servidor DNS o a taules locals dins de la màquina actual *(/etc/hosts)* i el seu format inclou les paraules clau *order*, *trim*, *multi*, nospoof, *spoof* i *reorder*, de les quals les més habituals són:

- ***order***: indica l'ordre de com es farà la cerca del nom (per exemple, *bind*, *hosts*, *nis*).

- ***multi***: que pot ser *on*, i retornarà totes les adreces vàlides per a un *host* que estigui *a /etc/hosts file*, o *off*, i solament retornarà el primer; si està en *on* pot causar retards importants quan */etc/hosts* sigui molt gran.

- ***nospoof/spoofalert/spoof*** (*on/off*): relacionat amb qüestions de seguretat en relació amb prevenir el *hostname spoofing* realitzat per algunes aplicacions.

Exemple de l'arxiu /etc/host.conf

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

Aquesta configuració indica que primer es verifica */etc/hosts* abans de sol·licitar una petició al DNS i també indica (2a. línia) que retorni totes les adreces vàlides que hi hagi a */etc/hosts*. Per això, l'arxiu */etc/hosts* és on es col·loquen les adreces locals i 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 els nodes locals i una entrada per a la interfície *loopback*.

4.2.3. L'arxiu */etc/hosts*

Aquest arxiu actua coma servidor de noms i és especialment útil en una xarxa local en què no hi hagi una alta variabilitat de les IP assignades als noms.

```
127.0.0.1 localhost
192.168.168.254 nteum.remix.local nteum
# Nodes xarxa privada
192.168.168.1   nodo1.remix.local nodo1
192.168.168.2   nodo2.remix.local nodo2
# The following lines are desirable for IPv6 capable hosts
::1      localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
```

```
ff02::2 ip6-allrouters
```

Per al nom d'una màquina es poden utilitzar àlies, fet que significa que la màquina es pot anomenar de diferents maneres per a la mateixa adreça IP. Amb referència a la interfície *loopback*, aquest és un tipus especial d'interfície que permet fer connexions amb si mateixa (per exemple, per a 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* connectarà amb la màquina mateixa). La configuració és molt fàcil (la fa generalment l'*script* d'inicialització de xarxa).

Com ja hem comentat, en la versió 2 de la biblioteca GNU hi ha una canvi important respecte a la funcionalitat de l'arxiu *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 ha estat centralitzada en l'arxiu */etc/nsswitch.conf*, el qual permet a l'administrador configurar l'ordre i les bases de dades de manera molt simple.

En aquest arxiu cada servei apareix en una línia amb un conjunt d'opcions, en què, per exemple, hi ha la resolució de noms de node. S'hi indica que l'ordre de consulta de les bases de dades per a obtenir la IP del node o el seu nom serà primer el servei de DNS, que utilitzarà l'arxiu */etc/resolv.conf* per a determinar la IP del node DNS, i en cas que no el pugui obtenir, utilitzarà el de les bases de dades locals (*/etc/hosts*). Altres opcions per a això podrien ser *nis, nisplus*, que són altres serveis d'informació que es descriuran en unitats posteriors. També es pot controlar per mitjà d'accions (entre []) el comportament de cada consulta, com per exemple:

```
hosts: files mdns4_minimal [NOTFOUND=return] dns mdns4
```

Això indica que quan es faci la consulta al mdns4, si no hi ha un registre per a aquesta consulta, retorni un zero al programa que la va fer. Es pot utilitzar el "!" per a negar l'acció, com per exemple:

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

4.3. Configuració de l'encaminament

Un altre aspecte que cal configurar és l'encaminament. Si bé hi ha el tòpic sobre la seva dificultat, generalment es necessiten uns requisits d'encaminament molt simples. En un node amb múltiples connexions, l'encaminament consisteix a decidir on cal enviar i què es rep. Un node simple (una sola connexió de xarxa) també necessita encaminament, ja que tots els nodes disposen d'un *loopback* i una connexió de xarxa (per exemple, Ethernet, PPP, SLIP...). Com es va explicar anteriorment, hi ha una taula anomenada *routing table*, que conté

Nota

Consulta de taules d'encaminament:

route -n
netstat -r

files amb diversos camps, però tres són summament importants: **adreça de destinació**, **interfície** per on sortirà el missatge i **adreça IP**, que efectuarà el pas següent a la Xarxa (passarel·la).

L'ordre `route` permet modificar aquesta taula per a fer les tasques d'encaminament adequades. Quan arriba un missatge, es mira la seva adreça de destinació, es compara amb les entrades a la taula i s'envia per la interfície en què l'adreça coincideix millor amb la destinació del paquet. Si s'especifica una passarel·la, el missatge s'envia a la interfície adequada.

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

- Primer la interfície:

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

- Més endavant, cal indicar que tots els paquets amb adreces 192.168.0. * han de ser enviats 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 ens volem connectar a un altre node fora d'aquest segment? Seria molt difícil tenir totes les entrades adequades per a totes les màquines a les quals ens volem connectar. Per a simplificar aquesta tasca, hi ha el *default route*, que s'utilitza quan l'adreça de destinació no coincideix a la taula amb cap de les entrades. Una possibilitat de configuració seria:

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

Nota

El *gw* és la IP o el nom d'una passarel·la o node encaminador.

Una forma alternativa de fer-ho és:

```
ifconfig eth0 inet down           deshabilito 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 nova 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 els manuals de les ordres `ifconfig`(8) i `route`(8).

4.4. Configuració d'*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 aquesta mode el servei escolta el port assignat i sempre està actiu) o per mitjà d'`inetd`.

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

Dos arxius importants necessiten ser configurats de la manera següent: */etc/services* i */etc/inetd.conf*. En el primer s'associen els serveis, els ports i el protocol, i en el segon, quins programes servidors respondran davant d'una petició a un port determinat. El format de */etc/services* és *name port/protocol alias*, en què el primer camp és el nom del servei, el segon, el port on 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 l'arxiu */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
...
```

L'arxiu */etc/inetd.conf* és la configuració per al servei mestre de xarxa (*inetd server 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 a la primera columna de l'arxiu `/etc/services`, *socket_type* és el tipus de sòcol (valors possibles: *stream, dgram, raw, rdm*, o *seqpacket*), *proto* és el protocol vàlid per a aquesta entrada (ha de coincidir amb el de l'arxiu */etc/services*), *flags* indica l'acció per prendre quan hi ha una nova connexió sobre un servei que està atenent una altra connexió (*wait* diu a *inetd* que no posi en marxa un nou servidor, i *nowait* significa que *inetd* ha de posar en marxa un nou servidor). *user* serà l'usuari amb el qual s'identificarà qui ha posat en marxa el servei, *server_path* és el directori on es troba el servidor i *server_args* són arguments possibles que seran passats al servidor. Un exemple d'algunes línies de l'arxiu */etc/inetd.conf* són (cal recordar que després de # hi ha comentaris, per la qual cosa, 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, la funcionalitat d'`inetd` ha estat reemplaçada per `xinetd` (recomanable perquè presewnta més opcions pel que fa a la configuració de serveis i seguretat), el qual necessita l'arxiu de configuració */etc/xinetd.conf* (vegeu el final del mòdul).

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 a més:

Vegeu també

Per a veure més sobre la configuració típica dels serveis de xarxa en un entorn d'escriptori o servidor bàsic, vegeu el mòdul de servidors de l'assignatura *Administració avançada de sistemes GNU/Linux*.

- **ssh**: connexió interactiva segura com a reemplaçament de *telnet;* inclou dos arxius de configuració, */etc/ssh/ssh_config* (per al client) i */etc/ssh/sshd_config* (per al servidor).
- **exim**: agent de transport de correu (MTA), inclou els arxius de configuració: */etc/exim/exim.conf*, */etc/mailname*, */etc/alias* i */etc/email-addresses*.
- **fetchmail**: dimoni per a descarregar el correu d'un compte POP3, amb */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). Els directoris/nom poden variar en funció del paquet instal·lat.
- **CVS**: sistema de control de versions concurrents; */etc/cvs-cron.conf*, */etc/cvs-pserver.conf*.
- **NFS**: sistema d'arxius de xarxa; */etc/etc/exports*.
- **Samba**: sistema d'arxius de xarxa i compartició d'impressores en xarxes Windows; */etc/samba/smb.conf*.
- **CUPS**: sistema d'impressió, */etc/cups/**.
- **Apache** i **Apache2**: servidor de web; /etc/apache/* i /etc/apache2/*.
- **squid**: servidor de servidor cau; /etc/squid/*.

4.5. Configuració addicional: protocols i xarxes

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

Exemple de l'arxiu /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
```

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

Exemple de l'arxiu /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 mitjà de la xarxa. Ja se'n parlarà més sobre aquest tema en la unitat corresponent a seguretat; 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:

a) No activar serveis a */etc/inetd.conf* que no s'utilitzaran; inserir un # abans del nom per a evitar fonts de risc.

b) Modificar l'arxiu */etc/ftpusers* per a denegar que certs usuaris puguin tenir connexió via FTP amb la màquina.

c) Modificar l'arxiu */etc/securetty* per a indicar des de quins terminals (un nom per línia; per exemple, *tty1 tty2 tty3 tty4*) es permet la connexió del superusuari (*root*). Des dels terminals restants, *root* no es podrà connectar.

d) Utilitzar el programa `tcpd`. Aquest servidor és un *wrapper* que permet acceptar o negar un servei des d'un determinat node, i es col·loca a */etc/inetd.conf* com a mediador d'un servei. El **tcpd** verifica unes regles d'accés a dos arxius: */etc/hosts.allow* 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 mostrada abans a `inetd.conf`: *ftp stream tcp nowait root /usr/sbin/tcpd /usr/sbin/in.ftpd*). `tcpd` primer cerca */etc/*

hosts.allow i després */etc/hosts.deny*. L'arxiu *hosts.deny* conté la informació sobre quins són 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 a */etc/hosts.allow*.

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

f) En Debian és important configurar */etc/security/access.conf*, l'arxiu que indica les regles de qui i des d'on es pot connectar (*login*) a aquesta màquina. Aquest arxiu té una línia per ordre amb tres camps separats per ":" del tipus *permís:usuaris:origen*. El primer serà un + o – (accés 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 d'*access.conf*

Aquesta ordre no permet entrades com a *root* sobre *tty1:*

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

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

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

4.7. Opcions d'IP

Hi ha una sèrie d'opcions sobre el trànsit IP que és convenient esmentar. La configuració es fa per mitjà de la inicialització de l'arxiu corresponent en el directori */proc/sys/net/ipv4/*. El nom de l'arxiu és el mateix que el de l'ordre i per a activar-los s'ha de posar un 1 dins de l'arxiu, i un 0 per a desactivar-lo. Per exemple, si es vol activar *ip_forward*, s'hauria d'executar:

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

Els més utilitzats són *ip_forward*, utilitzat per a l'encaminament 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); i *ip_bootp_agent*, variable lògica (booleana) que accepta paquets (o no) amb adreça origen del tipus 0.b.c.d i destinació d'aquest node, difusió o multidestinació.

Aquestes opcions també es poden modificar estàticament des de l'arxiu */etc/sysctl.conf* (mireu a */etc/sysctl.d/* per a variables addicionals, i *sysctl.conf* (5) per a més informació).

En cas que no sigui necessari treballar amb IPv6 es poden deshabilitar fent:

```
echo "net.ipv6.conf.all.disable_ipv6 = 1" >> /etc/sysctl.conf
```

I per mirar els canvis:

```
sysctl -p
net.ipv6.conf.all.disable_ipv6 = 1
```

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

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

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

Per a verificar la connexió a la xarxa, es poden usar les ordres següents (cal tenir instal·lats `netkit-ping`, `traceroute`, `dnsutils`, `iptables` i `net-tools`):

```
ping uoc.edu                    # verificar la connexió a Internet
traceroute uoc.edu              # buscar paquets IP
ifconfig                        # verificar la configuració dels dispositius de xarxes
route -n                        # verificar la configuració de la ruta
dig [@dns.uoc.edu] www.uoc.edu  # verificar registres de www.uoc.edu
                                # sobre el servidor dns.uoc.edu
iptables -L -n |less            # verificar filtratge de paquets (nucli >= 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)
```

Entre tota ella ampliarem l'explicació de *netstat* (forma part del paquet net-tools), ja que és una de les eines que ens mostra un informe instantani de l'activitat de xarxa. Quan s'executa sense paràmetres ens mostrarà totes les connexions obertes, que pot ser una llista massa detallada, ja que inclou alguns punts de connexió que no són de xarxa (la comunicació de dbus per exemple). No obstant això, té una gran quantitat de paràmetres per a definir la sortida. Entre els més utilitzats podem enumerar:

-t només mostra connexions TCP; o ídem però UDP.
-a, ports en Listen (connexions entrants)
-n, solament IP, ports i ID de l'usuari
-p, processos involucrats (útil quan s'executa com a arrel)
-c, actualitzar dinàmicament la llista de connexions.

És habitual executar com `netstat -tupan` però és recomanable analitzar la resta de paràmetres per a aprofundir en els conceptes de monitoratge de la xarxa.

5. Configuració del DHCP

DHCP són les sigles de *Dynamic Host Configuration Protocol*. La configuració és molt simple i serveix perquè, en lloc de configurar cada node d'una xarxa individualment, es pugui fer de manera centralitzada i l'administració sigui més fàcil. La configuració d'un client és molt fàcil, ja que només s'ha d'instal·lar un dels paquets següents: `dhcp-client` (en Debian, per exemple, `isc-dhcp-client`) 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 tots els clients DHCP (incloent-hi Windows). En algunes distribucions no està habilitada la possibilitat d'enviar missatges a l'adreça 255.255.255.255, per a provar-ho, executeu:

```
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 `255.255.255.255 dhcp` a */etc/hosts* i intentar-ho de nou:

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

En Debian, per exemple, la instal·lació és `apt-get install isc-dhcp-server`. La configuració de `dhcpd` es pot fer modificant l'arxiu */etc/dhcp/dhcpd.conf*. Un exemple d'aquest arxiu:

```
# Exemple de /etc/dhcpd/hcpd.conf:
default-lease-time 1200;
max-lease-time 9200;
option domain-name "remix.cat";
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 mart {
               hardware ethernet 00:00:95:C7:06:4C;
               fixed address 192.168.11.146;
               option host-name "mart";
               }
```

```
        host saturn {
              hardware ethernet 00:00:95:C7:06:44;
              fixed address 192.168.11.147;
              option host-name "saturn";
              }
    }
```

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

Per a posar en marxa el servidor `/etc/init.d/isc-dhcp-server start` o `service isc-dhcp-server start`. Amb `/usr/sbin/dhcpd -d -f` es podrà veure l'activitat del servidor sobre la consola del sistema.

6. Múltiples IP sobre una interfície

Hi ha algunes aplicacions en què és útil configurar múltiples adreces IP a un únic dispositiu de xarxa. Els ISP[29] fan servir freqüentment aquesta característica per a proveir de característiques personalitzades (per exemple, de World Wide Web i FTP) als seus usuaris. Per a això, el nucli ha d'estar compilat amb les opcions de *Network Aliasing* i *IP* (aquestes opcions ja estan activades per defecte en els nuclis actuals).

[29] De l'anglès *internet service providers*

Els àlies son annexats a dispositius de xarxa virtuals associats al nou dispositiu amb un format com:

```
dispositiu: nombre virtual
```

Per exemple:

```
eth0:0, ppp0:8
```

Una configuració habitual seria en */etc/network/interfaces* assignar per exemple tres IP a **eth0**:

```
auto eth0:0
allow-hotplug eth0:0
iface eth0:0 inet static
    address 192.168.1.43
    netmask 255.255.255.0
auto eth0:1
allow-hotplug eth0:1
iface eth0:1 inet static
    address 192.168.1.44
    netmask 255.255.255.0
```

També es pot fer mitjançant la línia d'ordres:

```
ifconfig eth0 192.168.1.42 netmask 255.255.255.0 up
ifconfig eth0:0 192.168.1.43 netmask 255.255.255.0 up
...
```

La qual cosa significa que tindrem dues IP (IP 192.168.1.42 i 192.168.1.43) per a la mateixa interfície. Per a esborrar un àlies, cal agregar un "-" al final del nom (per exemple, *ifconfig eth0:0-*).

Un cas típic és que es vulgui configurar una única targeta Ethernet perquè sigui la interfície de diferents subxarxes IP. Per exemple, suposem que tenim una màquina que es troba en una xarxa LAN 192.168.0.0/24 i volem connectar la màquina a Internet usant una adreça IP pública proporcionada amb DHCP, usant la targeta Ethernet existent.

Per exemple, es pot fer com en l'exemple anterior o també editar l'arxiu */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`.

Una interfície "àlies" podria no tenir passarel·la ni *dns-nameservers*. També es possible fer una assignació dinàmica d'IP.

7. IP Network Address Translation (NAT)

El NAT és un recurs perquè un conjunt de màquines puguin utilitzar una única adreça IP coma passarel·la. Això permet que els nodes en una xarxa local puguin sortir cap a Internet (és a dir, els que utilitzen una IP privada; per exemple, 198.162.10.1); però no poden acceptar crides o serveis de l'exterior directament, sinó per mitjà de la màquina que té la IP real.

L'IP *network address translation* (NAT) és el reemplaçament del que es coneixia com a *IP masquerade* i forma part del *kernel* (el qual ha d'estar configurat amb `IP_ADVANCED_ROUTER`, `IP_MULTIPLE_TABLES _IP_ROUTE_NAT`, `IP_FIREWALL` i `IP_ROUTE_FWMARK` però és la configuració habitual dels *kernels* actuals).

Avui dia són pocs els serveis que no poden executar-se en una xarxa privada amb accés de l'exterior i altres hauran de ser configurats en manera PASV (passiu) perquè funcionin. No obstant això, WWW, telnet o irc funcionen adequadament.

El cas més habitual, avui dia, és tenir un conjunt de màquines virtualitzades amb una fent de *gateway* i, per exemple, connectada al *host* mitjançant NAT, i les altres en una xarxa interna privada (que al seu torn faran NAT amb el *gateway*), però pot extrapolar-se a dispositius físics d'acord a la configuració del maquinari de què disposem.

Com hem vist, i d'acord amb l'RFC 1918, es poden utilitzar com a IP privades 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 han de ser ocultats (*masqueraded*) estaran dins d'aquesta segona xarxa. Cadascuna d'aquestes màquines hauria de tenir l'adreça de la màquina que fa la *masquerade* com a *default gateway* o encaminador. Sobre aquesta màquina (*gateway*) podem configurar una interfície perquè "miri" cap a la xarxa interna (per exemple, `eth1`) i una altra cap a la xarxa externa (per exemple, `eth0`):

- *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 eth0
```

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

```
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

- Per a automatitzar el procés i no haver d'executar-lo durant cada *boot* del sistema podem fer:
  - Modificar l'arxiu */etc/sysctl.conf* amb `net.ipv4.ip_forward=1` (o amb l'ordre `sysctl net.ipv4.ip_forward=1`)
  - Agregar la regla `iptables -t nat -A POSTROUTING -s 192.168.0.0/24 -o eth0 -j MASQUERADE` (es pot incloure o *source* –*s* o no).
  - Desar regles d'iptables: `iptables-save > /etc/default/iptables`
  - Descarregar *script* d'init 'iptables' (https://github.com/sirtea/iptables-init-debian).
  - Iniciar regles i test: `/etc/init.d/iptables start`
- Per a mirar les regles: `iptables –L –n` i `iptables –t nat –L –n`
- Per a verificar que tot estigui bé des d'una màquina interna es fa `ping google.com`
- Si és necessari instal·lar-les a *runlevel 2*: `update-rc.d [-n] iptables defaults` (cal mirar on són les regles */etc/default/iptables*)

És necessari recordar que s'hauran d'agregar les regles de *routing* adequades en cas que sigui necessari redirigir els paquets cap a interfícies internes. Una eina útil per a observar per on estan passant els paquets i detectar els problemes és el `tcpdump`, que és molt fàcil d'utilitzar.

Hi ha una altra manera de realitzar NAT amb el paquet `iproute2` (disponible a Debian). Aquest paquet (http://es.wikipedia.org/wiki/Iproute2) conté un conjunt d'eines que reemplaça les funcionalitats de `ifconfig`, `route` i `arp`. L'eina principal del paquet `iproute2` és `ip`, la seva sintaxi és `ip [opcions] objecte [ordre [arguments]]`. La manera més habitual és l'ordre `ip route`, que administra les entrades de `route` amb les taules d'encaminament que maneja el nucli [DR]. Per exemple:

- Agregar un camí cap a la xarxa 10.0.0/24 per la porta d'enllaç 192.168.0.2:
  `ip route add 10.0.0.0/24 via 192.168.0.2`

- Per a aplicar NAT abans de transmetre
  `ip route add nat 158.109.64.1 via 192.168.0.3`

És a dir `ip route add nat <extaddr>[/<masklen>] via <intaddr>` permetrà que un paquet d'entrada destinat a *ext-addr* (l'adreça visible des de fora d'Internet) transcrigui la seva adreça de destinació a *int-addr* (l'adreça de la seva xarxa interna per mitjà de la seva *gateway/firewall*). El paquet s'encamina d'acord amb la taula local de *route*. Es poden traslladar adreces simples o blocs.

- Si fem `ip route add nat 158.109.64.1 via 192.109.0.2` farà que l'adreça interna 192.109.0.2 sigui accessible com a 158.109.64.1

- Si fem `ip route add nat 158.109.64.32/24 via 192.109.0.0` farà que el bloc 192.109.0.0-31 es traslladi a 158.109.64.32-63.

Aquest nou paquet (`iproute2`) reemplaça les ordres del paquet original (`net-tools`) i les equivalències són:

Net-tools	Iproute2	Definició
ifconfig(8)	ip addr	(IP or IPv6) configuració d'un dispositiu
route(8)	ip route	entrada en la *table* de *routing*
arp(8)	ip neigh	entrada en la *table* d'ARP o NDISC
ipmaddr	ip maddr	adreça *multicast*
iptunnel	ip tunnel	tunel sobre IP
nameif(8)	ifrename(8)	redenominat de *network interfaces*
mii-tool(8)	ethtool(8)	paràmetres del dispositiu Ethernet

8. Udev - *Linux dynamic device management*

`udev` és el mecanisme que reemplaça l'antic *device file system (DevFS)* i permet identificar un dispositiu sobre la base de les seves propietats com per exemple marca, model, dispositiu i a més s'executa en espai d'usuari. `udev` permet regles per a especificar quin nom es dóna a un dispositiu i fer que aquest es visualitzi, per exemple, com un */dev/el_meu_nom*.

`udev` està format per uns serveis del *kernel services* (que hi transfereixen els esdeveniments) i el *daemon* `udevd` (que trasllada aquests esdeveniments a accions, les quals poden ser configurades mitjançant regles). La inicialització ocorre mitjançant */etc/rcS.d/udev*, la configuració s'emmagatzema a */etc/udev/udev.conf* i les regles s'obtenen de */run/udev/rules.d, /etc/udev/rules.d* o */lib/udev/rules.d* (Debian utilitza en la major part */lib/udev/rules.d*)

Un dels aspectes interessants de `udev` és que es va dissenyar per a respondre a esdeveniments de *hotplug* com els que es defineixen en les interfícies de xarxa amb el *tag allow-hotplug*, i les definicions de les regles per als dispositius de xarxa s'emmagatzemen a */etc/udev/rules.d/70-persistent-net.rules*. Aquest mecanisme té alguns inconvenients quan es treballa amb màquines virtuals clonades o es canvia un dispositiu, ja que la configuració del dispositiu original està emmagatzemada en aquest arxiu i el sistema la reconeixerà com un nou dispositiu i, per tant, la redenominarà a continuació. Per a solucionar aquest inconvenient simplement s'ha d'esborrar l'arxiu */etc/udev/rules.d/70-persistent-net.rules* i rearrancar el sistema. L'ordre per a gestionar les regles i el mecanisme `udev` és `udevadm` (per exemple, `udevadm info -a -n /dev/sda`).

Nota

Consulteu més informació a [HeMa]: http://debian-handbook.info/browse/stable/sect.hotplug.html i per a escriure regles http://www.reactivated.net/writing_udev_rules.html.

9. *Bridging*

El *bridging* d'una connexió a xarxa és un mètode per a compartir la connexió a Internet, per exemple, entre dos o més ordinadors. Això és útil si no es disposa d'un encaminador amb més d'un port d'Ethernet o està limitat per aquest nombre. Bàsicament s'utilitza perquè un ordinador que està connectat Internet pugui, per un altre dispositiu de xarxa, connectar-se a un segon ordinador que no té connexió a Internet i proveir el segon d'accés a Internet. En sistemes físics pot semblar estrany però és habitual quan tenim un conjunt de màquines virtualitzades i volem que totes tinguin accés a Internet.

L'ordre a utilitzar és `brctl` i està inclosa en el paquet `bridge-utils` que haurem d'instal·lar (`apt-get install bridge-utils`). Aquest ens permetrà configurar i utilitzar una nova interfície (*bridge*) de la mateixa manera que eth0, eth1, etc., la qual no existeix físicament (és virtual), i de manera transparent obtindrà els paquets d'una interfície i els traslladarà a l'altra.

La manera més simple de crear una interfície és `brctl addbr br0`, i es poden agregar els dispositius físics amb `brctl addif br0 eth0 eth1`.

Per a fer des de */etc/network/interfícies*:

```
iface eth0 inet manual
 iface eth1 inet manual
 # Bridge setup
 iface br0 inet dhcp
   bridge_ports eth0 eth1
```

Si necessitem configurar-la amb IP estàtica es pot canviar en br0:

```
iface br0 inet static
        bridge_ports eth0 eth1
        address 192.168.1.2
        broadcast 192.168.1.255
        netmask 255.255.255.0
        gateway 192.168.1.1
```

Nota

Podeu trobar més detalls sobre la configuració i paràmetres (libvirt, Wi-Fi, configuració avançada) a https://wiki.debian.org/BridgeNetworkConnections.

10. Xarxa privada virtual (VPN)

Una VPN[30] és una xarxa que utilitza Internet com a transport de dades, però impedeix que hi puguin accedir membres externs.

(30)De l'anglès virtual private network.

Tenir una xarxa amb VPN significa tenir nodes units per mitjà d'un túnel per on viatja el trànsit i on ningú no hi pot interactuar. S'utilitza quan es tenen usuaris remots que accedeixen a una xarxa corporativa per a mantenir la seguretat i privacitat de les dades. Per a configurar una VPN es poden utilitzar diversos mètodes SSH (SSL), CIPE, IPSec, PPTP, que es poden consultar en les referències [Bro, Wil] del projecte TLDP que, tot i que són una mica antigues, mostren els principis i els conceptes bàsics de l'estructura VPN.

Per a fer les proves de configuració, en aquest apartat s'utilitzarà **OpenVPN**, que és una solució basada en SSL VPN, i es pot usar 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 les capes OSI 2 o 3 amb protocols SSL/TLS i suporta autenticació basada en certificats, targetes (*smart cards*), i altres mètodes de certificació. Cal tenir present que OpenVPN no és un servidor intermediari (o *proxy*) d'aplicacions ni opera per mitjà d'un navegador web.

10.1. Instal·lació i prova en mode *raw*

En aquest apartat utilitzarem una màquina Debian i una altra Ubuntu però és similar en la resta de les distribucions. Primer cal instal·lar en les dues màquines OpenVPN: `apt-get install openvpn`. En funció de la distribució podrà donar alguns errors, ja que intenta arrencar el servei però com que encara no està configurat mostra alguns avisos. El segon és provar en mode *raw* si la connectivitat entre servidor i client funciona o hi ha algun impediment (per exemple, un tallafoc). Per a això executem, des del servidor:

```
openvpn --remote ubub --dev tun1 --ifconfig 10.9.8.1 10.9.8.2
```

des del client:

```
openvpn -remote deba -dev tun1 -ifconfig 10.9.8.2 10.9.8.1
```

És necessari que totes dues màquines estiguin connectades (en el nostre cas **ubub**, **10.9.8.2** és el nom de la màquina client –Ubuntu– i la seva IP en el túnel VPN i **deba**, **10.9.8.1** el nom de la màquina servidor i la IP en el túnel VPN). La sortida serà similar en totes dues màquines, per exemple, en el servidor:

```
Thu Jun 12 12:50:31 2014 OpenVPN 2.2.1 x86_64-linux-gnu [SSL] [LZO2] [EPOLL] [PKCS11] [eurephia]
      [MH] [PF_INET6] [IPv6 payload 20110424-2 (2.2RC2)] built on Jun 18 2013
Thu Jun 12 12:50:31 2014 IMPORTANT: OpenVPN's default port number is now 1194, based on
     an official port number assignment by IANA.  OpenVPN 2.0-beta16 and earlier used 5000
  as the default port.
Thu Jun 12 12:50:31 2014 NOTE: OpenVPN 2.1 requires '--script-security 2' or higher to call
     user-defined scripts or executables
Thu Jun 12 12:50:31 2014 ******* WARNING *******: all encryption and authentication features
     disabled -- all data will be tunnelled as cleartext
Thu Jun 12 12:50:31 2014 TUN/TAP device tun1 opened
Thu Jun 12 12:50:31 2014 do_ifconfig, tt->ipv6=0, tt->did_ifconfig_ipv6_setup=0
Thu Jun 12 12:50:31 2014 /sbin/ifconfig tun1 10.9.8.1 pointopoint 10.9.8.2 mtu 1500
Thu Jun 12 12:50:31 2014 UDPv4 link local (bound): [undef]
Thu Jun 12 12:50:31 2014 UDPv4 link remote: [AF_INET]172.16.1.2:1194
Thu Jun 12 12:50:34 2014 Peer Connection Initiated with [AF_INET]172.16.1.2:1194
Thu Jun 12 12:50:35 2014 Initialization Sequence Completed
```

On es pot veure els paràmetres importants de la connexió (port, túnel i IP). Si fem un `ping` sobre el servidor (des d'un altre terminal) veurem que totes dues puntes de túnel funcionen:

```
ping 10.9.8.1
ping 10.9.8.1 (10.9.8.1) 56(84) bytes of data.
64 bytes from 10.9.8.1: icmp_req=1 ttl=64 time=0.027 ms
--- 10.9.8.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.027/0.045/0.064/0.019 ms
ping 10.9.8.2
PING 10.9.8.2 (10.9.8.2) 56(84) bytes of data.
64 bytes from 10.9.8.2: icmp_req=1 ttl=64 time=0.597 ms
--- 10.9.8.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.597/1.131/1.666/0.535 ms
```

I també podrem veure la interfície com a tun1 amb `ifconfig`:

```
tun1      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
            inet addr:10.9.8.1  P-t-P:10.9.8.2  Mask:255.255.255.255
            UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
            RX packets:5 errors:0 dropped:0 overruns:0 frame:0
            TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:100
```

```
              RX bytes:420 (420.0 B)  TX bytes:420 (420.0 B)
```

Per a acabar l'aplicació simplement cal fer un Crtl+C (i veureu com també desapareix la interfície *virtul tun1*). El pas següent després de verificar la connectivitat és fer la configuració.

10.2. VPN amb intercanvi de claus estàtiques

Per a analitzar aquest servei utilitzarem una opció d'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 simplicitat, i que no és necessari un certificat X509 PKI[31] per a mantenir la VPN. Els desavantatges són que només permet un client i un servidor. En no utilitzar el mecanisme PKI (clau pública i clau privada), hi pot haver igualtat de claus amb sessions anteriors, i hi ha d'haver, doncs, una clau en mode text en cada igual (*peer*), i la clau secreta ha de ser intercanviada anteriorment per un canal segur.

[31] De l'anglès *public key infrastructure*.

Recordem que el nostre túnel VPN tindrà sobre el servidor IP=10.9.8.1 i el client amb IP=10.9.8.2. La comunicació serà encriptada entre el client i el servidor sobre UDP port 1194 (que és el port per defecte d'OpenVPN). Després d'instal·lar el paquet s'haurà de generar la clau estàtica a */etc/openvpn* i copiar-la de manera segura al client:

```
cd /etc/openvpn
openvpn --genkey --secret static.key
scp static.key ubub:/etc/openvpn
```

En el nostre cas hem utilitzat l'ordre *secure copy* (`scp`) per a transferir l'arxiu *static.key* al client `/ubub` sobre un canal segur.

L'arxiu de configuració del servidor */etc/openvpn/tun0.conf*:

```
dev tun0
ifconfig 10.9.8.1 10.9.8.2
secret /etc/openvpn/static.key
```

L'arxiu de configuració del client, */etc/openvpn/tun0.conf*:

```
remote deba
dev tun0
ifconfig 10.9.8.2 10.9.8.1
secret /etc/openvpn/static.key
```

Abans de verificar el funcionament de la VPN, ha d'assegurar-se en el tallafoc que el port 1194 UDP està obert sobre el servidor i que la interfície virtual tun0 usada per OpenVPN no està bloquejada ni sobre el client ni sobre el servidor. Tinguem present que el 90% dels problemes de connexió trobats per usuaris nous d'OpenVPN estan relacionats amb el tallafoc.

Per a verificar l'OpenVPN entre dues màquines, (recordem canviar les IP i el domini pel qual tingui la seva configuració) i després executar tant del costat servidor com del client (--verb 6 mostrarà informació addicional a la sortida i es pot evitar en execucions posteriors):

```
openvpn --config /etc/openvpn/tun0.conf --verb 6
```

El qual donarà una sortida similar a:

```
Thu Jun 12 13:59:58 2014 us=359502 OpenVPN 2.2.1 x86_64-linux-gnu [SSL] [LZO2] [EPOLL] [PKCS11]
      [eurephia] [MH] [PF_INET6] [IPv6 payload 20110424-2 (2.2RC2)] built on Jun 18 2013
.
Thu Jun 12 13:59:58 2014 us=361008 TUN/TAP device tun0 opened
Thu Jun 12 13:59:58 2014 us=361049 TUN/TAP TX queue length set to 100
Thu Jun 12 13:59:58 2014 us=361081 do_ifconfig, tt->ipv6=0, tt->did_ifconfig_ipv6_setup=0
Thu Jun 12 13:59:58 2014 us=361113 /sbin/ifconfig tun0 10.9.8.1 pointopoint 10.9.8.2 mtu 1500
Thu Jun 12 13:59:58 2014 us=363861 Data Channel MTU parms [ L:1544 D:1450 EF:44 EB:4 ET:0 EL:0 ]
Thu Jun 12 13:59:58 2014 us=363927 Local Options String: 'V4,dev-type tun,link-mtu 1544,tun-mtu
     1500,proto UDPv4,ifconfig 10.9.8.2 10.9.8.1,cipher BF-CBC,auth SHA1,keysize 128,secret'
Thu Jun 12 13:59:58 2014 us=363947 Expected Remote Options String: 'V4,dev-type tun,link-mtu
     1544,tun-mtu 1500,proto UDPv4,ifconfig 10.9.8.1 10.9.8.2,cipher BF-CBC,auth SHA1,keysize 128,secret'
..
Thu Jun 12 14:00:37 2014 us=657976 Peer Connection Initiated with [AF_INET]172.16.1.2:1194
Thu Jun 12 14:00:37 2014 us=658000 Initialization Sequence Completed
```

Per a verificar-ne el funcionament es pot executar, per exemple, un `ping 10.9.8.1` des del client, amb la qual cosa veurem la seva resposta i a més en la consola del servidor un missatge de l'activitat del túnel similar a:

```
Thu Jun 12 14:06:54 2014 us=604089 TUN READ [84]
Thu Jun 12 14:06:54 2014 us=604113 UDPv4 WRITE [124] to [AF_INET]172.16.1.2:1194:  DATA len=124
Thu Jun 12 14:06:55 2014 us=604293 UDPv4 READ [124] from [AF_INET]172.16.1.2:1194:  DATA len=124
Thu Jun 12 14:06:55 2014 us=604342 TUN WRITE [84]
Thu Jun 12 14:06:55 2014 us=604370 TUN READ [84]
Thu Jun 12 14:06:55 2014 us=604395 UDPv4 WRITE [124] to [AF_INET]172.16.1.2:1194:  DATA len=124
```

Per a agregar compressió sobre l'enllaç, ha d'afegir-se la línia següent als dos arxius de configuració `comp-lzo`, i per a protegir la connexió amb un NAT *router/firewall* i seguir els canvis d'IP mitjançant un DNS, si un dels *peers* canvia, cal agregar als dos arxius de configuració:

Nota

Podeu trovar més informació al web d'OpenVPN: http://openvpn.net/index.php/open-source/documentation/miscellaneous/78-static-key-mini-howto.html.

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

Com hem vist, l'execució anterior és en consola i per això la consola queda bloquejada amb l'execució de la VPN, però una vegada depurada és necessari engegar com a *daemon* i per a això és necessari agregar a l'arxiu de configuració */etc/openvpn/tun0.conf* la paraula *daemon* i canviar en */etc/default/openvpn* la línia d'AUTOSTART="tun0" per a indicar-li el nom de la nostra VPN (també es pot posar "all", que executarà tots els arxius de configuració */etc/openvpn/*.conf*). Després queda engegar-la a tots dos costats (`service openvpn start`) i provar amb un `ifconfig` que hi ha el `ping` la seva funcionalitat.

Una vegada establerta la connexió, la podem utilitzar per a connectar-nos com si es tractés de qualsevol altra IP, per exemple des del servidor podem fer un `ssh 10.9.8.2`, la qual cosa ens connectarà amb el client per `ssh` però a través del túnel encriptat (amb el `ssh` no és necessari, ja que el mateix encripta la comunicació però s'utilitza a tall d'exemple).

10.3. VPN amb TLS

Si ben com a prova de concepte una VPN amb clau estàtica és molt adequada perquè té una configuració simple i no necessita infraestructura PKI, té els inconvenients de l'escalabilitat (un servidor i un client). La clau, que és la mateixa, existeix en text en cada *peer* i ha de ser copiada per un canal previ segur. La configuració amb TLS ens permetrà resoldre aquests inconvenients si bé la seva configuració i posada en marxa serà una mica més complexa.

En primer lloc, hem de copiar l'*script* que ens permetrà generar les claus:

```
cd /etc/openvpn
mkdir easy-rsa
cp -R /usr/share/doc/openvpn/examples/easy-rsa/2.0/* easy-rsa/
```

Modificar */etc/openvpn/easy-rsa/vars* per a reflectir-ne la localització:

```
export KEY_COUNTRY="SP"
export KEY_PROVINCE="BCN"
export KEY_CITY="Bellaterra"
export KEY_ORG="Nteum"
```

```
export KEY_EMAIL="root@deba"
```

Executar:

```
cd /etc/openvpn/easy-rsa;
mkdir keys; touch keys/index.txt; echo 01 > keys/serial
. ./vars
./clean-all
```

Cal recordar que els arxius *.key són confidencials, els arxius *.crt i *.csr poden ser enviats per canals insegurs (per exemple, correu electrònic) i que cada ordinador haurà de tenir el seu parell *certificate/key*.

Amb això ja podem crear l'entitat certificadora (CA) que generarà els arxius */etc/openvpn/easy-rsa/keys/ca.crt* i */etc/openvpn/easy-rsa/keys/ca.key*:

```
./build-ca
```

A continuació, generarem una *inntermediate certificate authority certificate/key* que crearà els arxius *server.crt* i *server.key* a */etc/openvpn/easy-rsa/keys/* signats per la CA:

```
./build-key-server server
```

A continuació s'han de generar els arxius Diffie-Hellamn (necessaris en la comunicació SSL/TLS) i seguidament generarem les claus per al client (**ubub** en el nostre cas):

```
./build-dh
./build-key ubub
```

Aquest últim pas generarà en el directori */etc/openvpn/easy-rsa/keys/* els arxius *ubub.crt* i *ubub.key*, els quals s'hauran de copiar en el mateix directori *(/etc/openvpn/easy-rsa/keys/)* del client i també el certificat de la CA (*ca.crt*). Amb això ja podem provar la connexió des de la consola (la primera ordre s'ha d'executar en el servidor i la segona en el client –cal tenir en compte que és una única línia tant per al servidor com per al client):

```
openvpn --dev tun1 --ifconfig 10.9.8.1 10.9.8.2 --tls-server --dh /etc/openvpn/easy-rsa/keys
    /dh1024.pem --ca /etc/openvpn/easy-rsa/keys/ca.crt --cert /etc/openvpn/easy-rsa/keys
 /server.crt --key /etc/openvpn/easy-rsa/keys/server.key --reneg-sec 60 --verb 5

openvpn --remote deba --dev tun1 --ifconfig 10.9.8.2 10.9.8.1 --tls-client --ca /etc/openvpn
    /easy-rsa/keys/ca.crt --cert /etc/openvpn/easy-rsa/keys/ubub.crt --key /etc/openvpn
 /easy-rsa/keys/ubub.key --reneg-sec 60 --verb 5
```

La sortida serà equivalent a la que hem vist en el punt anterior i ara amb l'`ifconfig` tindrem un dispositiu tun1. Des d'una altra terminal podem realitzar un `ping` per a comprovar-ne la funcionalitat. Fixem-nos en el text que s'haurà de reemplaçar amb els arxius que generem per a la seva configuració.

A continuació crearem els arxius de configuració en el servidor i el client perquè arrenqui de manera automàtica. Primer crearem el directori de *log* `mkdir -p /etc/openvpn/log/`; `touch /etc/openvpn/log/openvpn-status.log` i després editarem */etc/openvpn/server.conf*:

```
port 1194
proto udp
dev tun
ca      /etc/openvpn/easy-rsa/keys/ca.crt    # generated keys
cert    /etc/openvpn/easy-rsa/keys/server.crt
key     /etc/openvpn/easy-rsa/keys/server.key  # keep secret
dh      /etc/openvpn/easy-rsa/keys/dh1024.pem
server 10.9.8.0 255.255.255.0  # internal tun0 connection IP
ifconfig-pool-persist ipp.txt
keepalive 10 120
comp-lzo         # Compression - must be turned on at both end
persist-key
persist-tun
status log/openvpn-status.log
verb 3  # verbose mode
client-to-client
```

Després modificarem l'arxiu */etc/default/openvpn* per modificar l'entrada `AUTOSTART="server"` i engegar el servei (`service openvpn start`). Sobre el client crearem l'arxiu */etc/openvpn/client.conf*:

```
client
dev tun
port 1194
proto udp
remote deba 1194             # VPN server IP : PORT
nobind
ca /etc/openvpn/easy-rsa/keys/ca.crt
cert /etc/openvpn/easy-rsa/keys/ubub.crt
key /etc/openvpn/easy-rsa/keys/ubub.key
comp-lzo
persist-key
persist-tun
verb 3
```

Aquí també modificarem l'arxiu */etc/default/openvpn* per modificar l'entrada `AUTOSTART="client"` i engegar el servei (`service openvpn start`). Veurem que s'ha creat la interfície tun0 i que respon al `ping` a les diferents ordres sobre la IP del túnel del servidor. A partir d'aquest servidor i amb petits canvis podrem utilitzar les Apps OpenVPN per a IOS i per a Android o per a reenviar el trànsit IP pel túnel VPN (vegeu detalls a https://wiki.debian.org/OpenVPN).

De vegades és necessari que els clients puguin configurar-se la connexió a la VPN, per la qual cosa una interfície gràfica pot ajudar. Hi ha dos paquets com *pluguins* al NetworkManager que redueixen la complexitat a l'hora de configurar els clients (`network-manager-openvpn`) i per als quals treballen escriptoris Gnome `network-manager-openvpn-gnome`. Primer des del servidor s'han de generar els certificats per al nou client i enviar-los-hi per un canal segur (sobretot l'arxiu .key), posteriorment s'ha d'instal·lar un dels dos paquets (cal tenir en compte que el segon reemplaçarà el primer si aquest està instal·lat). Amb aquests *pluguins* i una vegada arrencat el NetworkManager serà molt fàcil configurar el client perquè aquest gestioni la connexió a la VPN (https://wiki.gnome.org/Projects/NetworkManager/Admins).

11. 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 amigable.

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, no per part de l'usuari local, sinó per mitjà d'una suplantació d'identitat) o bé ajudar l'usuari novell a configurar adequadament els serveis.

En aquest sentit, és necessari preveure:

1) Configuració avançada de TCP/IP: per mitjà de l'ordre *sysctl* és possible modificar els paràmetres del nucli durant l'execució o en l'inici per a 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 per mitjà de l'arxiu de configuració */etc/syscntl.conf*. Després de la modificació, s'ha de tornar a engegar la xarxa:

Nota

Recordeu que és important fer una còpia dels fitxers de configuració (per exemple, `cp /etc/syscntl.conf`, `/etc/syscntl.conf.org`) per a retornar la configuració al seu estat anterior en cas de tenir problemes.

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

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

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

No respon 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 la difusió (*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.

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

Permet rebutjar un atac DoS de paquets SYNC que consumiria tots els recursos del sistema i forçaria a fer un reinici de la màquina.

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

Útil per a evitar atacs amb CMP *redirect acceptance* (aquests paquets són utilitzats quan l'encaminament no té una ruta adequada) en totes les interfícies.

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

Envia alertes sobre tots els missatges erronis a 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 registres sobre tots els *spoofed packets*, *source routed packets* i *redirect packets*.

Els paràmetres següents permetran que el sistema pugui atendre 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).
```

2) Iptables: Netfilter és un conjunt de funcionalitats integrades en *el kernel* de Linux per a interceptar i gestionar els paquets de xarxa. El component principal d'aquest és `iptables`, que funciona com una eina de tallafoc (*firewall*) i permet no solament les accions de filtre sinó també de translació d'adreces de xarxa (NAT) –com ja s'ha comentat en l'apartat corresponent d'aquest mòdul– o redireccions/registre de les comunicacions. Aquest tema es veurà detalladament en l'assignatura *Administració avançada de sistemes GNU/Linux*, però aquí donarem uns mínims conceptes sobre aquest paquet. Amb l'ordre `iptables` podem gestionar les regles fent, per exemple:

```
iptables -L     per posar les regles en una llista
iptable -t nat -L  ídem anerior però les de NAT
iptables -A Type -i Interf -p prot -s SrcIP --source-port Ps -d DestIP --destination-port Pd -j Action
  Inserir una regla
iptables-save > /etc/iptables.rules
  Salvar les regles definides
iptables-restore < /etc/iptables.rules
  Restaurar regles salvades prèviament
```

Un exemple de l'arxiu */etc/iptables.rules* podria ser (# indica comentari):

```
*filter
# Permetre tot el trànsit de loopback (el0) i denegar la resta de 127/8
-A INPUT -i lo -j ACCEPT
-A INPUT ! -i lo -d 127.0.0.0/8 -j REJECT
# Acceptar totes la connexions entrants establertes prèviament
-A INPUT -m state --state ESTABLISHED, RELATED -j ACCEPT
# Acceptar tot el trànsit sortint
-A OUTPUT -j ACCEPT
# Permetre HTTP i HTTPS des de qualsevol lloc
-A INPUT -p tcp --dport 80 -j ACCEPT
-A INPUT -p tcp --dport 443 -j ACCEPT
# Permetre les connexions d&#8217;SSH
# Normalment utilitza el port 22, verificar-ho en l'arxiu /etc/ssh/sshd_config.
 -A INPUT -p tcp -m state --state NEW --dport 22 -j ACCEPT
# Respondre al ping icmp
-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
# Rebutjar tot el trànsit restant d'entrada.
-A INPUT -j REJECT
-A FORWARD -j REJECT
COMMIT
```

Com es pot veure, la sintaxi pot semblar una mica estranya però és fàcil d'aprendre. En aquest cas, les regles s'han posat perquè actuïn com a tallafoc i, com que hauran de ser carregades per l'ordre `iptables-restore`, no inclouen l'ordre `iptables` a l'inici de cada regla i és necessari el *commit* al final. En aquest exemple, solament acceptem trànsit entrant de *ping*, *http*, *htpps* i

Nota

Podeu trobar més informació sobre aquest tema a https://wiki.debian.org/es/iptables.

ssh i bloquegem tot el trànsit restant d'entrada. Quant al trànsit de sortida es deixa que surti tot sense restriccions. Es poden salvar les regles en un arxiu, carregar-les i, per exemple, des d'una altra màquina provar la connectivitat o executar algun programa com `nmap` que ens mostrarà els ports oberts en la màquina configurada amb `iptables`.

3) GnuPG: GnuPG és una implementació completa de l'estàndard OpenPGP definida per l'RFC 4880. GnuPG (o GPG) permet xifrar i signar dades de tot tipus i disposa d'un sistema de gestió de claus molt versàtil i dels mòduls d'accés per a tot tipus de directoris de claus públiques. GPG pot funcionar en línia d'ordres o integrat a eines per mitjà de les seves biblioteques, i també proporciona suport per a S/ MIME (*secure/multipurpose Internet mail extensions*).

Per a crear el parell de claus s'ha d'executar `gpg --gen-key` responent a les preguntes que ens farà.

Per a veure les claus creades, s'utilitza `gpg --list-keys` o `gpg -v -fingerprint`, la qual cosa ens donarà un resultat com:

```
pub   2048R/119EDDEA 2014-06-13
Key fingerprint = 6644 5BF2 9926 441C A211  B684 DA35 616A 119E DDEA
uid   Remo Suppi (Universitat Autònoma de Barcelona) <Remo.Suppi@uab.cat>
sub   2048R/C8A79D4F 2014-06-13
```

El següent és crear un certificat de revocació, ja que si bé ara no és important, quan la clau pública estigui en un servidor i es vulgui destruir (per diferents qüestions), l'única manera és revocar-la. Per a això, executem `gpg -a --gen-revoke`, copiem la informació entre [BEGIN] i [END] en un arxiu i el posem a resguard, ja que amb aquest es podrà anul·lar la clau pública.

El següent és fer "pública" la clau pública en algun servidor com per exemple *pgp.mit.edu*; per a això executarem `gpg --keyserver=x-hkp://pgp.mit.edu -a --send-keys 119EDDEA`, en què l'últim número amb el mecanisme de claus públiques i privades, però amb això apareix un nou problema: si rebem (per exemple des d'un servidor de claus) una clau pública d'algú, com sabem que aquesta clau pertany realment a la persona a qui diu que pertany? Qualsevol podria crear una clau en nom nostre i usar-la i ningú no s'adonaria que no som nosaltres! és per això que s'ha de ser prudent en acceptar altres claus i garantir que aquesta clau és d'una determinada persona (comprovant-ho amb el *Key-ID* i el *fingerprint*). Un mecanisme fàcil és signar una clau i com més usuaris coneguts signin aquesta clau més (tots) estarem segurs que pertany a l'usuari que coneixem (és el que es denomina *key-signing*). Jo puc signar una altra clau (amb la meva) per a garantir que aquesta clau pertany a qui estic segur que és, i també si rebo una clau pública d'algú que no sé qui és però està signada per diverses persones que conec llavors puc confiar que aquesta clau pertany a aquesta persona (confiança).

Per a signar una clau pública ha d'estar en el *key-ring* i s'ha d'executar `gpg --edit-key <description>`, on *description* pot ser el nom/correu o qualsevol dada o part de la clau que volem signar. Després entrarem en mode ordre i indiquem `sign -> grau de confiança (0-3) -> I -> passwd -> save`.

Ara s'hauria de pujar novament aquesta clau pública al servidor amb la nova signatura: `gpg -a --send-keys <key-ANEU>`. Per a actualitzar les claus que tenim en el nostre *key-ring* hem de fer `gpg --refresh-keys`. Per a encriptar un arxiu hauríem de fer `gpg -e -a -r <description> file`, el qual es dirà *file.asc*, ja que hem inclòs el *–a* que indica que la sortida ha de ser ASCII, si no existeix es generarà en binari com a *file.gpg*. Per a desencriptar s'haurà de fer `gpg -d -o arxivi_sortida`.

Per a utilitzar-ho en el correu o altres aplicacions és convenient integrar `gpg` amb l'aplicació utilitzada. Per exemple, per a incloure `gpg` a *Thunderbird* i poder signar/encriptar correus haurem d'instal·lar una extensió anomenada *enigmail*.

Nota

Podeu trobar més informació sobre *enigmail* a https://www.enigmail.net/home/index.php.

4) Logcheck: una de les activitats d'un administrador de xarxa és verificar diàriament (més d'una vegada per dia) els arxius de registre (*log*) per a detectar possibles atacs, intrusions o esdeveniments que puguin donar indicis sobre aquestes qüestions. Aquesta eina selecciona (dels arxius de registre) informació condensada de problemes i riscos potencials, i després envia aquesta informació al responsable, per exemple, per mitjà d'un correu. El paquet inclou utilitats per a executar-se de manera autònoma i recordar l'última entrada verificada per a les execucions subsegüents.

La llista d'arxius a monitorar s'emmagatzema a */etc/logcheck/logcheck.logfiles*, i la configuració per defecte és adequada si no es va modificar gran part de l'arxiu */etc/syslog.conf*. logcheck pot funcionar en tres modalitats: *paranoid* (molt detallat i hauria de limitar-se a casos específics com, per exemple, *firewall*), *server* (el que apareix per defecte, recomanat per a la majoria dels servidors) i *workstation* (adequat per a estacions d'escriptori). Aquesta eina permet una configuració total dels filtres i de les sortides si bé pot ser complicat reescriure'ls. Les regles es poden classificar en "intent d'intrusió" (*cracking*), emmagatzemades a */etc/logcheck/cracking.d/*, les d'"alerta de seguretat", emmagatzemades a */etc/logcheck/violations.d/* i les aplicades a la resta dels missatges. [HeMa].

5) PortSentry i **Tripwire:** PortSentry forma part d'un conjunt d'eines que proporcionen serveis de seguretat de nivell de *host* per a GNU/Linux. PortSentry, LogSentry i hostsentry protegeixen contra escanejos de ports i detecten indicis d'activitat sospitosa. Tripwire és una eina que ajuda l'administrador notificant possibles modificacions i canvis en arxius per a evitar possibles danys (majors). Aquesta eina compara les diferències entre els arxius 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 d'arxius vitals com, per exemple, en arxius de configuració.

6) Xinetd: aquesta eina millora notablement l'eficiència i les prestacions d'*inetd* i *tcp-wrappers*. Un dels grans avantatges de **xinetd** és que pot fer front a atacs de DoA[32] per mitjà de 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, SSH és millor que s'executi només com a dimoni), ja que molts generen una gran sobrecàrrega al sistema i disposen de mecanismes d'accés segurs que no creen interrupcions en la seguretat del sistema.

(32) De l'anglès *denial-of-access*.

La configuració és molt simple, es fa mitjançant l'arxiu */etc/xinetd.conf* (que pot incloure altres arxius del directori */etc/xinetd.d/* per a una millor estructuració). En aquest arxiu hi ha una secció **defaults**, que és on es troben els paràmetres que s'aplicaran a tots els serveis, i una **service** (o arxius que les contenen de *service*), que seran els serveis que es posaran en marxa per mitjà de *xinetd*.

Un exemple típic de la configuració podria ser el següent. Tot i que els **defaults** es col·loquen a *xinetd.con* i els **services** en arxius separats del directori */etc/xinetd.d/*, en aquest exemple ho hem posat tot junt:

```
# 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 ha de ser a /etc/services per a obtenir el port correcte
# Si es tracta d'un servidor/port no estàndard, useu "port = X"
service ftp
{
   socket_type  = stream
   protocol   = tcp
   wait   = no
   user   = root
   server   = /usr/sbin/proftpd
   }

service ssh
{
   socket_type  = stream
```

```
    protocol   = tcp
    wait   = no
    user   = root
    port   = 22
    server   = /usr/sbin/sshd server_args = -i
    }
# This is the tcp version.
service echo
{
    disable  = yes
    type  = INTERNAL
    id  = echo-stream
    socket_type = stream
    protocol = tcp
    user  = root
    wait  = no
    }

# This is the udp version.
service echo
{
    disable  = yes
    type  = INTERNAL
    id  = echo-dgram
    socket_type = dgram
    protocol = udp
    user  = root
    wait  = yes
    }
```

Els serveis comentats (#) no estaran disponibles. A la secció **defaults** es poden inserir paràmetres com ara 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 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
    cps                             = 25 30
    enabled = imaps
    }
```

Un paràmetre interessant és *cps*, que limita el nombre de les connexions entrants amb dos arguments: el nombre de connexions per segon a manejar pel servei i el nombre de segons que quedarà deshabilitat el servei en cas de sobrepassar aquest nombre. Per defecte, cinquanta connexions i un interval de deu segons es consideren paràmetres adequats per a contenir una atac de *denial of service* (DoS)

La secció **service**, (una per cada servei) pot contenir paràmetres específics –i de vegades molt detallats– del 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.
}
```

7) Tcpdump i **Wireshark**: una eina molt potent que és en totes les distribucions i que ens serveix per a analitzar els paquets de xarxa és `tcpdump`. Aquest programa permet la visualització del trànsit de xarxa d'una xarxa i pot analitzar la majoria dels protocols que s'utilitzen avui dia (IPv4, ICMPv4, IPv6, ICMPv6, UDP, TCP, SNMP, AFS BGP, RIP, PIM, DVMRP, IGMP, SMB, OSPF, NFS, etc.). `Wireshark` és una altra eina (més complexa) que disposa d'una interfície gràfica per a analitzar paquets i també permet descodificar-los i analitzar-ne el contingut (actua com a *network sniffer*). Totes dues eines s'instal·len seguint el procediment habitual i estan preconfigurades en gairebé totes les distribucions.

Atesa la importància d'analitzar els paquets i saber on van, mostrarem algunes ordres habituals de `tcpdump`:

```
tcpdump          paràmetres per defecte -v o -vv per al nivell d'informació mostrada,
                 -q sortida ràpida
tcpdump -D       interfícies disponibles per a la captura
tcpdump -n       mostra IP en lloc d'adreces
tcpdump -i eth0       captura el trànsit d'eth0
```

```
tcpdump udp            solament els paquets UDP
tcpdump port http      solament els paquets del port 80 (web)
tcpdump -c 20          solament els 20 primers paquets
tcpdump -w capture.log envia les dades a un arxiu
tcpdump -r capture.log llegeix les dades d'un arxiu
```

Les dades capturades no són text, per la qual cosa es pot o utilitzar el mateix per a la seva lectura o una altra eina com wireshark per a llegir-les i descodificar-les:

```
tcpdump host www.uoc.edu    solament els paquets que continguin aquesta adreça
tcpdump src 192.168.1.100 and dst 192.168.1.2 and port ftp
           Per mostrar els paquets ftp que vagin des de 192.168.1.100 a 192.168.1.2:
tcpdump -A           mostra el contingut dels paquets
```

8) Webmin: és una eina que permet a través d'una interfície web configurar i afegir aspectes relacionats amb la xarxa. En l'URL indicat trobareu els detalls de la seva descàrrega i instal·lació de Debian, per exemple. Si bé es continua el seu desenvolupament en moltes distribucions, no s'inclou per defecte. Per a executar-la, una vegada instal·lada des d'un navegador cal cridar l'URL *https://localhost:10000*, que sol·licitarà l'acceptació del certificat SSL i l'usuari (inicialment `root`) i la seva clau (`passwd`).

9) system-config-*: a Fedora (i també en menys mesura a Debian) hi ha una gran varietat d'eines gràfiques que es diuen `system-config-"algunacosa"` i en què "alguna-cosa" és allò per a la qual cosa estan dissenyades. En general, si s'està en un entorn gràfic, es pot arribar a cadascuna per mitjà d'un menú; no obstant això, cadascuna d'aquestes eines implica un menú a recordar.

10) Altres eines:

- **Nmap**: explorar i auditar amb finalitats de seguretat una xarxa.
- **OpenVas**: anàlisi de vulnerabilitats.
- **Snort**: sistema de detecció d'intrusos, IDS.
- **Netcat**: utilitat simple i potent per a depurar i explorar una xarxa.
- **MTR**: programa que combina la funcionalitat de traceroute i ping i és molt adequada com a eina de diagnòstic de xarxa.
- **Hping2**: genera i envia paquets d'ICMP/UDP/TCP per analitzar el funcionament d'una xarxa.

Vegeu també

Algunes d'aquestes eines seran tractades en el mòdul d'administració de seguretat en l'assignatura *Administració avançada de sistemes GNU/Linux*, que tracta sobre seguretat.

Activitats

1. Definiu els escenaris de xarxa següents:

a) Màquina aïllada.
b) Petita xarxa local (4 màquines, 1 passarel·la).
c) 2 xarxes locals segmentades (2 conjunts de 2 màquines, un encaminador cada una i una passarel·la general).
d) 2 xarxes locals interconnectades (dos conjunts de 2 màquines + passarel·la cada una).
e) Una màquina amb dues interfícies connectada a Internet amb NAT a un encaminador i a una xarxa privada1, una segona màquina amb dues interfícies connectada a xarxa privada1, i l'altra a xarxa privada2, una tercera màquina connectada a xarxa privada2.
f) 2 màquines connectades per mitjà d'una xarxa privada virtual.

Indiqueu els avantatges i desavantatges de cada configuració, per a quin tipus d'infraestructura són adequades i quins paràmetres rellevants es necessiten, tant per a IPv4 com per a IPv6.

2. Utilitzant màquines virtuals realitzar la configuració, monitoratge i test de connexió (per exemple, `ping`, `dig` i `apt-get update`) de les propostes del punt anterior i en cada màquina de l'arquitectura proposada.

3. Fer els experiments anteriors sobre IPv6 utilitzant `ping6` i un túnel (per exemple, http://www.gogo6.net/freenet6/tunnelbroker) per a mostrar la connectivitat cap a Internet.

Bibliografia

[Bro] Bronson, Scott (2001). "VPN PPP-SSH". *The Linux Documentation Project*, 2001.

[Cis] Cisco (2005). "TCP/IP Overview White Paper" (visitat 10/6/2014).

[Com] Comer, Douglas (2013). *Internetworking with TCP/IP Volume One*. Addison-Wesley.

[daCos] Costa, F. da. "Intel Technical Books, Rethinking the Internet of Things" (visitat 10/6/2014).

[DR] Debian. "Debian Reference Manual. Network Setup Chapter 5".

[Gar98] Garbee, Bdale (1998). *TCP/IP Tutorial*. N3EUA Inc.

[Gnu] Gnupg.org. GnuPG Web Site.

[GRD] Debian. "Guía de referencia Debian Capítulo 10. Configuración de red" (visitat 11/6/2014).

[HeMa] Hertzog, R.; Mas, R. (2013). "El libro del administrador de Debian 7". ISBN: 979-10-91414-02-9 (visitat 10/6/2014).

[IET] IETF "Repositorio de Request For Comment desarrollado por Internet Engineering Task Force (IETF) en el Network Information Center (NIC)".

[KD] Kirch, Olaf; Dawson, Terry (2000). *Linux Network Administrator's Guide*. O'Reilly Associates [com *e-book* (*free*) en Free Software Foundation, Inc].

[LeCe] Leiner, B.; Cerf, V.; Clark, D. i altres. "Brief History of the Internet" (visitat 10/6/2014).

[Mal] Mallett, Fred (1996). *TCP/IP Tutorial*. FAME Computer Education.

[Mou01] Mourani, Gerhard (2001). *Securing and Optimizing Linux: The Ultimate Solution*. Open Network Architecture, Inc.

[OVPN] OpenVPN "OpenVPN: Instalación y configuración".

[Wil02] Wilson, Matthew D. (2002). "VPN". *The Linux Documentation Project*.

Annex

Controlar els serveis vinculats a xarxa a FCx

Un aspecte important de tots els serveis és com es posen en marxa. Fcx (des de FC6) inclou una sèrie d'utilitats per a gestionar els serveis –dimonis– (incloent-hi els de xarxa). Com ja s'ha vist en l'apartat d'administració local, el *runlevel* és el mode d'operació que especifica quins dimonis s'executaran. A FC podem trobar: *runlevel* 1 (monousuari), *runlevel* 2 (multiusuari), *runlevel* 3 (multiusuari amb xarxa), *runlevel* 5 (X11 més *runlevel* 3). Típicament s'executa el nivell 5 o 3 si no es necessiten interfícies gràfiques. Per a determinar quin nivell s'està executant, 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 */etc/inittab* per a canviar el valor per defecte).

Per a visualitzar els serveis que s'estan executant, 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 amb `/sbin/service crond status` per a saber com està cada servei. I també gestionar-lo (*start, stop, status, reload, restart*); per exemple, `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 el que s'està fent): *acpid, haldaemon, messagebus, klogd, network, syslogd*. Els serveis més importants vinculats a la xarxa (encara que no es recullen tots, sí la majoria en aquesta llista no exhaustiva) són:

- **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 és una xarxa sense fil per a dispositius portàtils (*no és* Wi-Fi, 802.11). Per exemple, teclats, ratolins, telèfons, altaveus, auriculars, etc.

- **Capi, isdn:** xarxa basada en maquinari ISDN.

- **Iptables:** és el servei de tallafocs estàndard de Linux. És totalment necessari per seguretat si es té connexió a xarxa (*cable, DSL, T1*).

- **Ip6tables:** és el servei de tallafocs estàndard de Linux per a protocols i xarxes basats en IPv6.

- **Netplugd:** pot monitorar la xarxa i executar instruccions quan en canviï l'estat.

- **Netfs:** s'utilitza per a muntar automàticament sistemes d'arxius per mitjà de la xarxa (NFS, Samba, etc.) durant l'arrencada.

- **Nfs, nfslock:** són els dimonis estàndard per a compartir sistemes d'arxius per mitjà de la xarxa en sistemes operatius d'estil Unix/Linux/BSD.

- **Ntpd:** servidor d'hora i data a través de la xarxa.

- **Portmap:** és un servei complementari per a NFS (*file sharing*) o NIS (*authentication*).

- **Rpcgssd, rpcidmapd, rpcsvcgssd:** s'utilitza per a NFS v4 (nova versió d'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 altres usuaris connectar-se interactivament de manera segura a la màquina local.

- **Yum-updatesd:** servei d'actualitzacions per xarxa d'FC.

- **Xinetd:** servei alternatiu d'**inetd** que presenta un conjunt de característiques i millores, com per exemple llançar múltiples serveis per al mateix port (aquest servei pot no estar instal·lat per defecte).