Programari lliure

José María Barceló Ordinas
Jordi Íñigo Griera
Ramón Martí Escalé
Enric Peig Olivé
Xavier Perramon Tornil

71Z799C07MO

Xarxes de computadors

UOC
Formació de Postgrau

David Megías Jiménez

Coordinador

Enginyer d'Informàtica per la Universitat Autònoma de Barcelona. Màster en Tècniques Avançades d'Automatització de Processos per la Universitat Autònoma de Barcelona. Doctor en Informàtica per la Universitat Autònoma de Barcelona. Professor dels Estudis d'Informàtica i Multimèdia de la Universitat Oberta de Catalunya.

Jordi Mas i Hernández

Coordinador

Enginyer de programari en l'empresa de codi obert Ximian, on treballa en la implementació del projecte lliure Mono. Com a voluntari, col·labora en el desenvolupament del processador de textos Abiword i en l'enginyeria de les versions en català del projecte Mozilla i Gnome. És també coordinador general de Softcatalà. Com a consultor ha treballat per a empreses com Menta, Telépolis, Vodafone, Lotus, eresMas, Amena i Terra España.

Enric Peig Olivé

Coordinador

Doctor enginyer de Telecomunicacions per la Universitat Pompeu Fabra. Actualment treballa en l'especificació de metadades aplicades al comerç electrònic. És professor en els Estudis d'Informàtica de la Universitat Pompeu Fabra.

José María Barceló Ordinas

Autor

Doctor enginyer de Telecomunicacions per la Universitat Politècnica de Catalunya. Actualment treballa en l'avaluació de xarxes ATM i en la modelització de trànsit en xarxes informàtiques. És professor del Grup de Xarxes de Computadors a la Facultat d'Informàtica de Barcelona.

Jordi Íñigo Griera

Autor

Enginyer de Telecomunicacions per la Universitat Politècnica de Catalunya. Actualment és director de Desenvolupament de Programari de Safelayer Secure Communications, SA. Ha estat director tècnic de l'esCERT (Equip de Seguretat per a la Coordinació d'Emergències en Xarxes Telemàtiques) de la Universitat Politècnica de Catalunya. És professor del Grup de Xarxes de Computadors a la Facultat d'Informàtica de Barcelona.

Ramon Martí Escalé

Autor

Doctor enginyer de Telecomunicacions per la Universitat Politècnica de Catalunya. Actualment treballa en la seguretat en aplicacions distribuïdes de comerç electrònic d'informació multimèdia. És professor dels Estudis d'Enginyeria de Telecomunicació de la Universitat Pompeu Fabra de Barcelona.

Xavier Perramon Tornil

Autor

Doctor enginyer de Telecomunicacions per la Universitat Politècnica de Catalunya. Actualment treballa en el disseny i estandardització de sistemes de documentació multimèdia. És professor dels Estudis d'Informàtica de la Universitat Pompeu Fabra de Barcelona.

Primera edició: maig 2005

Av. Tibidabo, 39-43, 08035 Barcelona
Material realitzat per Eureca Media, SL

Dipòsit legal: B-15.566-2005
ISBN: 84-9788-298-9

Índex

Agraïments

Els autors agraeixen a la Fundació per a la Universitat Oberta de Catalunya (http://www.uoc.edu) el finançament de la primera edició d'aquesta obra, emmarcada en el Màster Internacional de Programari Lliure ofert per aquesta institució.

ANOTACIONS

Introducció

Les xarxes d'ordinadors actuals són una amalgama de dispositius, tècniques i sistemes de comunicació que han aparegut des de la darreria del segle XIX o, el que és el mateix, des de la invenció del telèfon. El telèfon, que es va desenvolupar exclusivament per a transmetre veu, avui s'utilitza, en molts casos, per a connectar ordinadors entre si. Des de llavors han aparegut les xarxes locals, les connexions de dades a llarga distància amb enllaços transoceànics o satèl·lits, la telefonia mòbil, etc. Mereix una menció especial la xarxa Internet dins d'aquest món de les comunicacions a distància. Ningú no dubta que avui en dia constitueix una xarxa bàsica de comunicació entre els humans.

Aquest curs ofereix una visió de les xarxes informàtiques en general i de la xarxa Internet en particular.

En la primera part, introduirem les idees i els conceptes bàsics de les xarxes d'ordinadors. Seguint un fil històric, presentarem els diferents mecanismes que s'han utilitzat i s'utilitzen per a comunicar-se a distància. Presentarem igualment el concepte d'arquitectura de protocols, fonamental en sistemes distribuïts, i el model de referència OSI com un exemple paradigmàtic d'això. Encara que avui en dia aquest model no gaudeixi d'una gran popularitat, les seves virtuts pedagògiques estan més que demostrades: a partir d'ell és fàcil estudiar i entendre altres arquitectures, com l'arquitectura Internet entorn de la qual gira tot el curs.

La segona part està dedicada a l'estudi de les xarxes d'àrea local. Presentem descriptivament els diferents tipus de xarxes que hi ha, les idees bàsiques del seu funcionament i la noció de cablatge estructurat, clau en el gran apogeu que han tingut últimament les xarxes d'àrea local.

En la tercera part es veuran els fonaments de la xarxa Internet. El que es coneix com *xarxa Internet* és un conjunt heterogeni de xarxes in-

Nota

Internet és una apòcope d'*internetworking* ('interconnectant xarxes').

terconnectades. Precisament, és la capacitat d'homogeneïtzar el que de fet és heterogeni, el que ha catapultat la xarxa Internet al seu estatus actual.

Els protocols que distingeixen la xarxa Internet com una unitat són l'IP (*Internet protocol*) i el TCP (protocol de control de transmissió o *transmission control protocol*). Aquests protocols no són els únics que es necessiten per a fer funcionar la xarxa Internet, però sí els més importants. Per aquest motiu, a tots en conjunt se'ls anomena normalment *pila TCP/IP* (*TCP/IP stack*).

En concret, en aquesta part es descriu el protocol IP i els seus col·laboradors més immediats (ARP i ICMP), i també els mecanismes d'accés a Internet de què disposem: per mitjà d'una xarxa d'àrea local o un enllaç telefònic: mitjançant PPP i un mòdem tradicional o, més recentment, mitjançant ADSL.

TCP/IP no és un estàndard *de iure*. Cap organisme internacional d'estandardització no s'ha encarregat d'emetre'l. Al contrari, el funcionament dels seus protocols està recollit en uns documents anomenats *RFC* (demanda de comentaris o *request for comments*), que són propostes que s'han fet sobre el funcionament d'un protocol concret, o d'una part. El procés és simple: una vegada feta pública una proposta, si ningú no hi posa cap objecció, ja es considera aprovada i llesta per a ser implementada.

Nota

Les RFC es poden consultar en l'adreça següent: http://www.ietf.org.

A més de consultar aquest material didàctic i la bibliografia recomanada, en què s'expliquen els protocols pedagògicament, es recomana llegir alguna RFC, encara que només sigui perquè es faci una idea del procés que ha seguit la xarxa des dels seus inicis.

En la quarta part, descriurem els protocols d'aplicació més utilitzats actualment a Internet i els programes més habituals que els implementen, com són la connexió remota (Telnet, rlogin), la transferència d'arxius (FTP), el correu electrònic (SMTP, POP, IMAP), les *news* (NNTP), el WWW (HTTP) i la missatgeria instantània.

Tots aquests programes es coneixen com *aplicacions distribuïdes*, ja que estan formades per diferents parts que es poden executar en màquines

ANOTACIONS

diferents. Aquesta dispersió de parts de programes obliga a definir una manera de dialogar entre elles.

Veurem, doncs, abans de començar la descripció de les diferents aplicacions, aquest concepte de programació distribuïda i el model client/servidor que és el que segueix majoritàriament.

Les aplicacions Internet permeten conèixer les màquines i els serveis per mitjà de noms, i no mitjançant números que és com treballen IP, TCP i UDP. Algú s'ha d'encarregar de l'associació dels noms amb les adreces numèriques i aquest algú és el servei DNS (sistema de noms de domini o *domain name system*). També tractarem d'aquest tema abans de descriure les aplicacions.

ANOTACIONS

Objectius

Amb els materials d'aquest curs es pretén que el lector assoleixi els objectius següents:

1. Conèixer les diferents tecnologies que s'utilitzen actualment per a transmetre informació a distància, i comprendre quan i per què van aparèixer.
2. Conèixer el model de referència OSI, les seves utilitats i les seves limitacions, i ser capaç d'entendre la motivació de cada un dels seus nivells.
3. Conèixer els principis bàsics de funcionament de les xarxes d'àrea local tant cablades com sense fil, les topologies possibles i les diferents polítiques d'accés al medi.
4. Conèixer el concepte de cablatge estructurat, entendre el paper que hi tenen els concentradors i saber diferenciar topologia física i topologia lògica.
5. Entendre els principis de funcionament del protocol de nivell de xarxa IP: l'assignació d'adreces i l'encaminament.
6. Aprendre el funcionament de les xarxes d'accés a Internet més comunes: accés LAN i accés per xarxa telefònica mitjançant PP o ADSL.
7. Entendre el funcionament dels protocols de transport i saber en quins principis es basen.
8. Conèixer algunes utilitats d'ús comú que permeten descobrir algunes interioritats dels protocols de xarxa i transport.
9. Comprendre el model client/servidor, que serveix com a base de la implementació d'aplicacions distribuïdes i el model d'igual a igual (*peer-to-peer*), complementari de l'anterior.
10. Comprendre el funcionament del DNS, el servei de noms de domini, que dóna suport a la resta d'aplicacions.

ANOTACIONS

11. Conèixer les aplicacions `telnet` i `rlogin`, que proporciona el servei de connexió remota a altres ordinadors (principalment en l'entorn GNU/Linux), i les aplicacions que proporcionen a Internet els serveis de transferència d'arxius, correu electrònic, *news*, WWW i missatgeria instantània, i sobretot els protocols que segueixen.

ANOTACIONS

I. Introducció a les xarxes de computadors

1. Breu història de les comunicacions

Des que l'ésser humà té capacitat de comunicar-se ha desenvolupat mecanismes i sistemes per a poder establir aquesta comunicació a distàncies superiors de les assolides pels seus propis mitjans.

Poc després d'aparèixer els ordinadors, es va sentir la necessitat d'interconnectar-los perquè es poguessin comunicar entre si com ho fem els humans.

En aquesta unitat ens plantegem repassar la història d'aquests sistemes de comunicació, pensats per a ser usats pels humans i que, després, han evolucionat per a interconnectar ordinadors.

Fixem l'inici d'aquest recorregut històric en el telèfon. El telèfon no va ser el primer sistema de telecomunicació, però sí el més antic dels que avui en dia s'utilitzen habitualment. Molt abans s'havien utilitzat sistemes òptics que, amb la llum del sol i jocs de miralls, permetien comunicar-se des de distàncies considerables. Posteriorment, a mitjan segle XIX, es va inventar el telègraf. Aquests sistemes, tanmateix, han caigut en desús (excepte usos marginals), mentre que la xarxa telefònica es manté com un sistema de comunicació de primer ordre.

1.1. El telèfon

El 1878, Alexander Graham Bell va mostrar la seva "màquina elèctrica parlant" i com podia mantenir una conversa a distància entre dos d'aquests aparells units per un fil elèctric.

Nota

Investigacions recents han fet sortir a la llum una història curiosa: sembla clar que l'inventor del telèfon va ser un italià anomenat Antonio Meucci, però no va patentar

Nota

Podeu trobar la història completa d'aquest episodi en l'adreça següent:

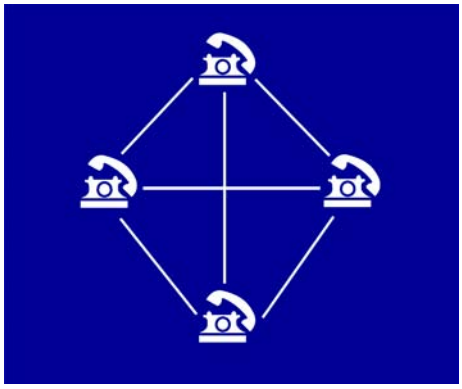
http://www.popular-science.net/history/meucci_bell.html.

el seu invent perquè no tenia suficients diners per a fer-ho. Bell es va apropiar de l'invent i el va patentar.

Al principi, els pocs telèfons que hi havia s'utilitzaven en entorns tancats, particulars. Servien per a interconnectar dos espais. A mesura que el nombre de telèfons instal·lats creixia, l'interès per mantenir comunicacions múltiples també ho feia: era necessari pensar en la manera d'interconnectar-los. Naixia la idea de la xarxa de comunicacions.

Una manera possible, bastant immediata, d'interconnectar tots els aparells, seria la que es pot observar en la figura següent:

Figura 1.

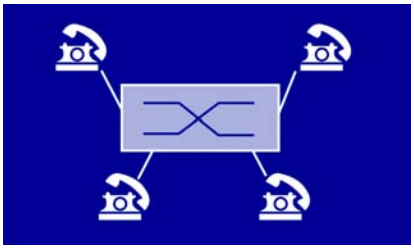


És evident que aquest model de connexió, "tots amb tots", és completament inviable: per a cada aparell nou que s'incorpora a la xarxa, es necessita un gran nombre de connexions noves. Per a fer-nos-en una idea, una xarxa "tots amb tots" de cinquanta telèfons necessita mil dues-centes vint-i-cinc línies de connexió i, a cada telèfon, un dispositiu que permeti quaranta-nou connexions.

Per a solucionar aquest problema, van aparèixer companyies que oferien un servei de **commutació**: feien arribar un cable fins a cada telèfon i connectaven els cables dels telèfons que volien establir una comunicació. D'aquesta manera, cada aparell disposava d'una sola connexió i no era necessari establir-hi cap variació per a incorporar aparells nous a la xarxa.

ANOTACIONS

Figura 2.



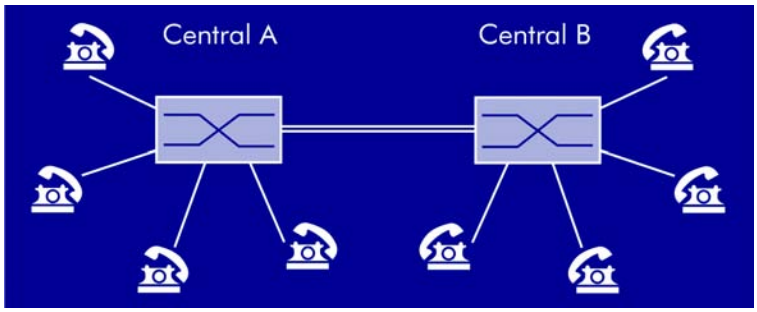
D'aquest fet provenen termes avui tan comuns com ***abonat*** (l'usuari que s'abona a una central), ***bucle*** d'*abonat* (el cable que uneix l'abonat amb la central) o *central de* ***commutació***.

La tasca de commutar les connexions, al principi, es feia a mà. Quan algú volia fer una trucada, despenjava i demanava a l'operadora que el connectés amb qui volia parlar. Una vegada finalitzada la comunicació, l'operadora desconnectava els cables i, així, les línies quedaven preparades per a rebre altres trucades.

Les operadores humanes van ser substituïdes progressivament per enginys electromecànics: les **centraletes**. Es va incorporar als telèfons un disc amb nombres per a "marcar" el número del destinatari de la trucada. La centraleta descodificava aquest número per saber entre quins dos cables era necessari establir la comunicació.

Aquest servei de commutació va començar en l'àmbit local: un barri, un poble, una ciutat. El pas següent va consistir a oferir connexions a llarga distància, connectant centrals locals entre si directament, o per mitjà de **centrals de trànsit**.

Figura 3. Comunicació entre dues centrals de commutació



Entre les dues centrals locals s'estableix un enllaç amb diferents cables independents, de manera que els abonats d'una d'aquestes poden, a

ANOTACIONS

més de connectar-se entre ells, connectar amb els abonats de l'altra: es tria un cable dels que formen l'enllaç, es connecta amb l'abonat local i es demana a l'altra central que connecti l'enllaç amb la destinació abonada, si no està ocupat amb cap altra trucada.

La connexió entre les dues centrals comporta un primer escull important: és necessari decidir amb quantes línies diferents es durà a terme.

Suposem que la central A de la figura anterior proporciona servei a cent abonats i la B, a dos-cents cinquanta. Sembla que, si es pretén donar el millor servei possible, es necessitin cent línies perquè tots els abonats de la central A puguin parlar de manera simultània amb altres punts de la central B.

No obstant això, la probabilitat que tots els abonats d'una central realitzin una trucada al mateix moment és molt baixa, ja que les trucades són, en general, curtes i esporàdiques. Per tant, és completament innecessari que la connexió entre les dues centrals tingui en compte totes les trucades possibles: aquesta situació no es donarà mai i té un cost exagerat.

Uns models matemàtics bastant complexos permeten calcular el nombre concret d'enllaços que es necessiten a partir de l'estadística de les trucades que serveixen les centrals (la freqüència d'aparició i la seva durada).

Nota

A.K. Erlang, enginyer danès de la primeria del segle XX, va establir els models matemàtics que s'utilitzen per a mesurar el trànsit telefònic. Es pot trobar molta informació sobre això en l'adreça següent: http://www.erlang.com.

Suposem que en l'exemple anterior aquests models ens donen vint-i-cinc enllaços. Si en un moment donat hi ha vint-i-cinc trucades en curs entre A i B i arriba una altra trucada, no tindrà cap camí disponible i, per tant, no es podrà establir. Aquesta situació es denomina *bloqueig*: tot i que l'abonat a qui es vol trucar no està ocupat, no es pot trobar un camí lliure a la xarxa per a establir la comunicació.

D'aquesta situació es desprenen dues idees fonamentals amb relació a la xarxa telefònica:

- La commutació de circuits requereix passar per tres fases per a cada comunicació:

– **Establiment de trucada**. Quan se sol·licita iniciar una conversa, és necessari esbrinar si el destinatari està disponible i, en cas afir-

ANOTACIONS

matiu, s'ha de buscar un camí lliure a la xarxa, que inclou commutadors dins de les centrals i enllaços entre elles.

– **Comunicació**. Una vegada establert el circuit, els interlocutors s'intercanvien informació.

– **Alliberament de recursos**. Acabada la comunicació, s'alliberen els recursos utilitzats (enllaços entre centrals i commutadors dins de les centrals).

- El fet que els recursos s'ocupin exclusivament mentre dura la comunicació fa que les companyies que ofereixen el servei cobrin segons la durada de la trucada: es penalitza l'ús extensiu dels recursos. D'aquesta manera, l'usuari s'afanya a acabar la comunicació i deixar els enllaços lliures, disminuint així la probabilitat de bloqueig.

La xarxa telefònica constitueix una xarxa de commutació de circuits. Per a dur a terme una comunicació, és necessari establir un circuit entre els dos extrems per mitjà de la xarxa. Mentre dura la comunicació, s'ocupen uns recursos en exclusiva, encara que no hi hagi intercanvi d'informació. Les companyies cobren l'ús dels recursos per temps d'ocupació.

Aviat, el sistema telefònic va passar a ser una qüestió nacional. Els estats desenvolupaven les seves xarxes segons els seus criteris i gustos. Es va crear un organisme, el CCITT (Comitè Consultiu Internacional de Telegrafia i Telefonia, Comité Consultatif International Télégraphique et Téléphonique), per a harmonitzar els sistemes nacionals i permetre les comunicacions entre països mitjançant centrals de trànsit internacionals.

Nota

El CCITT és un organisme internacional patrocinat per les operadores de telefonia, dedicat a tasques de normalització en l'àmbit de les telecomunicacions. L'1 de març de 1993 va passar a dir-se *ITU-T* (International Telecommunication Union Standardisation Sector).

Hem comentat que entre les centrals hi ha una sèrie de línies que permeten la connexió entre abonats de diferents centrals. Al principi era realment així: si es decidia que entre dues centrals era necessari disposar de cinquanta enllaços, es posaven cinquanta cables entre elles. Però, amb l'augment progressiu d'enllaços necessaris, aquest sistema aviat va ser totalment inviable i va ser necessari recórrer a una tècnica ja coneguda en radiodifusió: la multiplexació.

Nota

Multiplexar significa fer passar diferents comunicacions independents pel mateix medi de transmissió.

La tècnica de multiplexació que es va aplicar a la telefonia va ser la multiplexació en freqüència: es modulen els diferents canals d'entrada a diferents freqüències portadores, de manera que puguin viatjar pel mateix medi sense interferir-se. S'apliquen filtres a la recepció que permeten separar els diferents canals multiplexats.

Exemple

Fem el mateix en escoltar la ràdio o en veure la televisió. Fins a la nostra antena arriben tots els canals emesos; amb el dial i el selector de canals, respectivament, seleccionem el canal (la gamma de freqüències) corresponent a l'emissora que volem rebre. És a dir, el dial o el selector de canals de la televisió són els filtres que separen, en la recepció, els diferents canals multiplexats.

El nombre de canals diferents que poden viatjar per un medi multiplexat depèn de l'amplada de banda del senyal i de la capacitat del medi.

Pel que respecta a la capacitat del medi, no en té la mateixa un parell de fils que un cable coaxial o que una fibra òptica.

Quant a l'amplada de banda, en el cas de la veu, hauria de ser de 19.980 Hz (que és una amplada de banda considerable), ja que l'oïda humana és capaç de distingir freqüències entre els 20 Hz i els 20.000 Hz. No obstant això, arran d'estudis que es van dur a terme sobre les característiques de la veu humana, es va arribar a la conclusió que amb molt menys n'hi havia prou, ja que la intel·ligibilitat de la veu es concentra en una banda bastant estreta, entre els 300 Hz i els 3.400 Hz.

A partir d'aquesta conclusió, es va prendre una decisió que, a la llarga, ha condicionat molt l'ús de la xarxa telefònica: fer el canal de veu de 4 kHz (entre 300 Hz i 3.400 Hz, més unes bandes laterals de guarda).

Nota

Haver reduït el canal de veu a 4 kHz explica per què s'escolta tan malament la música pel telèfon: no hi ha ni greus ni aguts, només hi ha les freqüències del mig.

ANOTACIONS

També es van estandarditzar els diferents nivells de multiplexació. El nivell bàsic és l'agrupació de diferents canals de 4 kHz, el següent és una agrupació de multiplexats bàsics, etc.

Nota

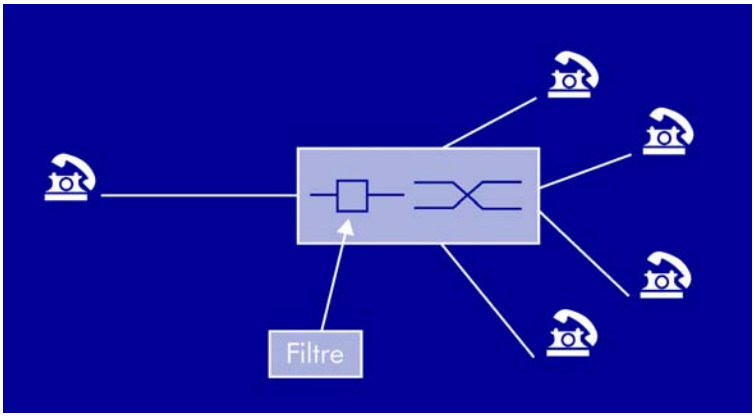
La jerarquia que va proposar la companyia americana AT&T, i que s'ha acabat estandarditzant, és la següent:

Taula 1.

Nom	Rang	Amplada de banda	Canals de veu
Group	60-108 kHz	48 kHz	12
Supergroup	312-552 kHz	240 kHz	60
Mastergroup	564-3.084 kHz	2,52 MHz	600
Jumbogroup	0,5-17,5 MHz	17 MHz	3.600

A l'entrada de la central local, un filtre elimina qualsevol freqüència per sobre dels 4 kHz. El seu senyal de sortida és el que es multiplexa, commuta i porta fins al destinatari.

Figura 4.



Amb tot això, ja podem dibuixar un panorama complet de la xarxa telefònica, tal com era fins als anys setanta.

La xarxa telefònica és analògica, ubiqua, treballa amb la tècnica de commutació de circuits, amb tarifació per

ANOTACIONS

temps d'ocupació, amb enllaços multiplexats en freqüència i amb canals limitats a 4 kHz.

1.2. Apareixen els primers ordinadors

Nota

En dir que eren màquines poc potents, evidentment, és comparant-les amb les actuals. Per a l'època, eren unes màquines fantàstiques.

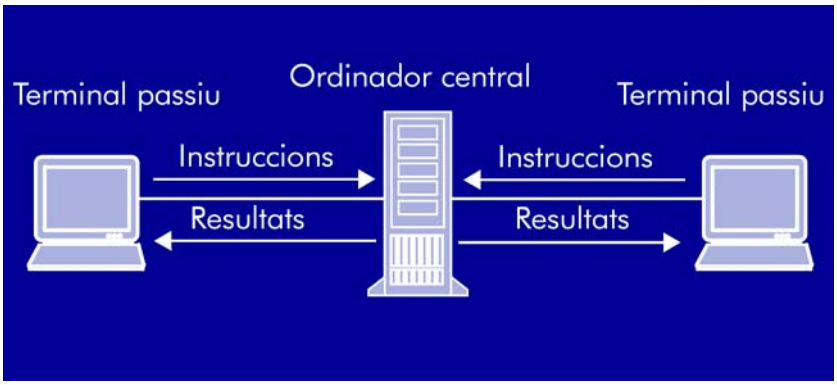
La dècada dels seixanta va conèixer l'aparició dels primers ordinadors comercials. Eren grans, cars i poc potents. Només organismes oficials, grans empreses o universitats els podien comprar, i el que és més normal és que només en compressin un (o alguns, però no un per a cada usuari, com avui dia estem acostumats a veure).

Nota

Als terminals passius, que col·loquialment s'anomenen *terminals "tontos"*, en anglès se'ls coneix com a *dumb terminal* ('terminal mut').

Per això, aquests ordinadors portaven sistemes operatius multitasca i multiusuari, perquè diferents usuaris, realitzant diferents treballs, poguessin utilitzar-los simultàniament. L'accés a aquests ordinadors es duia a terme per mitjà de terminals sense cap capacitat de procés, passius:

Figura 5.



1.2.1. Els mòdems

Nota

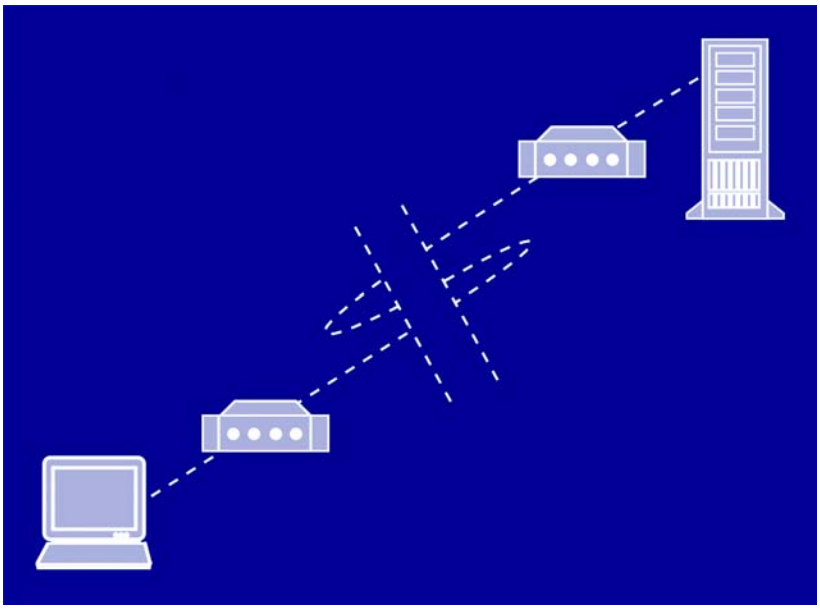
Mòdem és un acrònim de *modulator-demodulator*, que es refereix a la seva funció: modular (generar senyals audibles segons els valors dels bits) i desmodular (generar bits a partir dels senyals que rep de la xarxa telefònica).

No va trigar gaire a aparèixer la necessitat de poder allunyar els terminals de la unitat central per a connectar-se, per exemple, des de casa o des d'una delegació a l'ordinador central.

Per a poder realitzar aquest accés remot, la primera solució que van aportar els enginyers informàtics de l'època va ser utilitzar la xarxa

telefònica que, per la seva ubiqüitat, els estalviava generar infraestructures noves. Només es necessitava un aparell que adaptés els bits a la xarxa (recordeu que la xarxa telefònica només deixa passar sons entre uns marges de freqüència). Aquests aparells són els mòdems.

Figura 6.



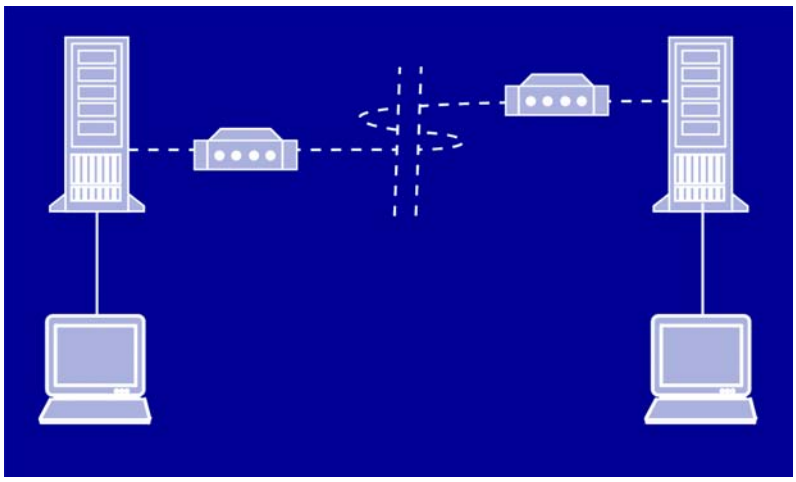
Els primers mòdems eren de 300 bps i generaven dos tons diferents: un per a l'1 lògic i un altre per al 0. Actualment, van a 56.000 bps, que és el màxim que permet la xarxa telefònica convencional actual.

Els 56.000 bps (56 k) de velocitat de transmissió només es poden aconseguir si un dels dos extrems té una connexió especial amb la seva centraleta (la majoria dels proveïdors d'Internet la tenen). De fet, amb línies telefòniques convencionals, la velocitat màxima és de 33.600 bps.

Els mòdems no solament servien per a poder allunyar els terminals passius dels ordinadors centrals, sinó que també permetien interconnectar ordinadors entre si.

ANOTACIONS

Figura 7.



Això ja és una xarxa de computadors!

La tecnologia de commutació de circuits es va desenvolupar en un origen per a les comunicacions telefòniques i una de les seves característiques fonamentals era l'ocupació en exclusiva dels recursos mentre durava la connexió, la qual cosa (com ja hem vist) justificava la tarifació per temps. Tanmateix, les comunicacions informàtiques no són curtes, intenses i esporàdiques com les de veu. En connectar un terminal a un ordinador central per mitjà de dos mòdems, no passen dades tot el temps que dura la connexió: hi pot haver llargs períodes de temps en què no passi cap bit i moments en què hi hagi un intercanvi de dades intens, encara que a una velocitat de transmissió molt més baixa que la que es pot mantenir entre el terminal i l'ordinador connectats directament. Les factures telefòniques van començar a ser astronòmiques, i desproporcionades, respecte de l'ús real de la xarxa.

1.2.2. Les xarxes de dades

Aviat les grans empreses van pressionar les companyies telefòniques del moment perquè desenvolupessin xarxes pensades per a transportar dades, amb un sistema de tarifació que s'ajustés al trànsit de dades real i permetés més velocitat que els escassos 300 o 1.200 bps que s'aconseguien utilitzant la xarxa telefònica. La resposta van ser les **xarxes de commutació de paquets**.

La tramesa de dades no necessàriament s'ha de dur a terme en temps real (les transmissions de veu, sí). Per tant, no és necessari es-

ANOTACIONS

tablir el camí entre els dos punts abans de començar la transmissió i mantenir-lo mentre dura l'intercanvi de dades. En lloc d'això, s'empaqueten els bits que s'han de transmetre i es donen a la central més pròxima perquè els enviï quan pugui a la següent, i així successivament fins que arribin a la destinació. Si quan un paquet arriba a una central tots els enllaços amb la següent s'estan ocupats, no passa res, el fa esperar posant-lo en una cua per a enviar-lo quan hi hagi un enllaç disponible.

La transmissió per paquets té l'avantatge que només ocupa els recursos quan en realitat s'utilitzen, no sempre. Com a contrapartida, és necessari suportar el retard que es pugui produir entre el moment en què els paquets surten de l'origen i arriben a la seva destinació, que és variable, ja que les esperes en les cues són aleatòries, depenen de l'estat de la xarxa. Però, com hem dit, en comunicació de dades aquest retard és fins a cert punt tolerable. Pel que fa a la qüestió econòmica, no té sentit que es cobri per temps de connexió: a les xarxes de dades es paga per bits transmesos.

Convé tenir present que les cues són limitades i, si arriba un paquet quan una ja és plena, no es podrà desar i es perdrà. És necessari preveure mecanismes que evitin aquestes pèrdues i regulin el flux d'informació entre els nodes de commutació.

Les companyies telefòniques van desenvolupar xarxes d'aquest tipus, i el CCITT va emetre un estàndard, l'X.25, que és el que s'ha adoptat fins fa molt poc temps.

Nota

A Espanya, la xarxa de dades es deia *Iberpac*.

Actualment, per a comunicacions de dades s'utilitza Frame Relay, l'evolució natural d'X.25.

1.2.3. Les xarxes d'àrea local

Quan va començar a ser habitual disposar de més d'un ordinador a la mateixa instal·lació, va aparèixer la necessitat d'interconnectar-los per a poder compartir els diferents recursos: dispositius cars, com ara impressores de qualitat, un disc dur que emmagatzemés les dades de l'empresa, un equip de cinta per a realitzar còpies de seguretat, etc.

ANOTACIONS

Nota

Amb freqüència s'utilitza la sigla anglesa *LAN* (*local area network*) per a identificar les xarxes d'àrea local, i la sigla *WAN* (*wide area network*) per a identificar les xarxes de gran abast.

El disseny de les xarxes d'àrea local va seguir camins completament diferents dels que es van seguir per a les xarxes de gran abast. A les xarxes d'àrea local es necessita, habitualment, establir comunicacions "molts a un" i "un a molts", cosa que és difícil d'aconseguir amb les xarxes de commutació, pensades per a interconnectar dues estacions. Per a aquest tipus de xarxes és més adequada la **difusió amb medi compartit**, en què els paquets que surten d'una estació arriben a tota la resta simultàniament. A la recepció, les estacions els accepten o ignoren depenent de si en són les destinatàries o no.

Difusió amb medi compartit

Es parla de *difusió* perquè els paquets s'envien per tot arreu, i de *medi compartit* perquè la difusió es duu a terme sobre un medi comú que les estacions comparteixen.

1.3. Arquitectures de protocols

De la dècada dels seixanta daten també els primers estàndards d'arquitectures de protocols. Convé tenir present que l'intercanvi d'informació entre ordinadors té tota una sèrie d'implicacions, entre les quals hi ha les següents:

- Aspectes elèctrics: els cables, els connectors, els senyals, etc.
- La manera d'agrupar els bits per a formar paquets i la de controlar que no es produeixin errors de transmissió.
- La identificació dels ordinadors dins de la xarxa i la manera d'aconseguir que la informació que genera un ordinador arribi a qui es pretén.

Atacar tots aquests aspectes d'una manera global no és viable: massa coses i massa diferents entre si. Per això, ja des del principi, es van desenvolupar models estructurats en nivells: en cada nivell es duu a terme una tasca i la cooperació de tots els nivells proporciona la connectivitat que volen els usuaris.

ANOTACIONS

Convé considerar que, en l'època que ens ocupa, la informàtica era a les mans de molt pocs fabricants i imperava la filosofia del servei integral: cada fabricant ho proporcionava tot (ordinadors, cables, perifèrics, sistema operatiu i programari). Per tant, quan una empresa es volia informatitzar, triava una marca i hi quedava vinculada per a tota la vida.

Nota

Parlem d'empreses com IBM (International Business Machines) o DEC (Digital Equipment Corporation). Quan aquestes empreses es van proposar oferir connectivitat entre els seus equips, local o remota, també ho van fer aplicant la filosofia de la separació per nivells: IBM va desenvolupar l'arquitectura SNA (*system network arquitecture*) i DEC, la DNA (*DEC network arquitecture*). Eren dos models complets, estructurats en nivells, però incompatibles entre si, segons la filosofia de la informàtica propietària.

En la dècada dels setanta el panorama va canviar radicalment, sobretot a causa de tres esdeveniments:

- La proposta del protocol Ethernet per a xarxes locals.
- L'aparició del sistema operatiu Unix, que no estava vinculat a cap marca comercial, compatible amb totes les plataformes de maquinari existents.
- La invenció dels protocols TCP/IP, embrió de l'actual Internet.

S'havia aplanat el camí per a l'aparició dels sistemes oberts: no era necessari vincular-se a cap marca per a tenir-ho tot. El maquinari podia ser d'un proveïdor, el sistema operatiu d'un altre, les aplicacions d'un altre i els protocols, públics.

Nota

TCP/IP són les sigles de *transmission control protocol / Internet protocol* (protocol de control de transmissió / protocol d'Internet).

TCP/IP va néixer a partir d'un encàrrec de la DARPA a la comunitat científica americana per a obtenir una xarxa mundial que fos reconfigurable amb facilitat i automàticament en cas de destrucció d'algun node o d'algun enllaç.

La pila TCP/IP era una jerarquia de protocols que oferia connectivitat i, malgrat que tenia poca relació amb les que ja existien, constituïa una

ANOTACIONS

opció més al mercat. Davant d'una oferta tan gran i dispar de protocols, l'ISO (Organització Internacional d'Estandardització, International Organization for Standardization) i el CCITT van proposar un model nou que intentava reunir d'alguna manera tot el que ja s'havia proposat i que pretenia ser complet, racional i molt ben estructurat (la TCP/IP té fama de ser una pila de protocols anàrquica), amb la intenció, per tant, que es convertís en un model de referència. Aquest model es coneix com ***pila de protocols OSI*** (*open systems interconnection*).

Internet, que va néixer i va créixer a les universitats, es va començar a popularitzar en la dècada dels noranta, a mesura que els qui coneixien la xarxa "l'ensenyaven", i la seva eclosió es va produir quan va saltar al món de l'empresa, en tots els seus vessants: com a aparador de productes o com a canalitzador de contactes comercials.

Tanmateix, l'origen universitari de la xarxa n'ha marcat l'evolució en molts sentits. Per exemple, el model client/servidor d'aplicacions distribuïdes. És un model senzill i, alhora, potent, i gairebé totes les aplicacions que s'utilitzen a Internet el segueixen. El Telnet, o obertura de sessió remota, la transferència de fitxers (FTP), el correu electrònic i, sobretot, el WWW (World Wide Web) constitueixen exemples clars d'aplicacions que segueixen aquest model. Les dues primeres han caigut una mica en desús, però tant el correu com el WWW són les actuals estrelles a Internet. Tímidament, apareixen noves propostes d'aplicacions; però, el WWW, que va néixer com un servei de pàgines estàtiques enllaçades amb hiperenllaços, es converteix en la interfície d'usuari de tota la xarxa, ja que actualment s'utilitza per servir pàgines dinàmiques (es creen en el moment que se serveixen) i, fins i tot, codi que s'executa a l'ordinador client (miniaplicació o *applet*).

1.4. La digitalització de la xarxa telefònica

En aquest moment tenim dues xarxes completament independents entre si, però d'alguna manera superposades:

- Una xarxa analògica, amb commutació de circuits, pensada per a veu.

ANOTACIONS

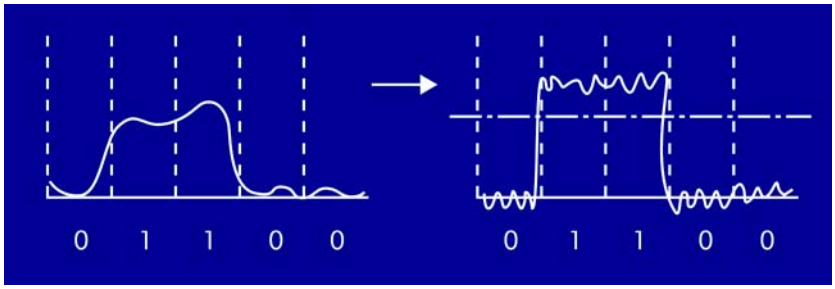
- Una xarxa digital, amb commutació de paquets, pensada per a dades.

La xarxa telefònica, tal com l'hem descrita fins ara, és completament analògica: el senyal electromagnètic que viatja des d'un telèfon fins a un altre és analògic (varia contínuament i en qualsevol moment pot adoptar qualsevol valor) i els circuits electrònics que componen la xarxa també ho són.

Els enllaços entre centrals de la xarxa telefònica es duien a terme mitjançant senyals analògics amb molts canals multiplexats en freqüència i, a vegades, havien de recórrer grans distàncies. L'atenuació del senyal inherent en la distància que calia recórrer s'havia de corregir per mitjà de repetidors que l'amplificaven, la qual cosa augmentava el soroll present en la línia. Sovint, el senyal rebut era d'una qualitat molt baixa perquè la transmissió analògica no permetia eliminar el soroll i les interferències en la recepció. No hi ha manera de saber amb exactitud què s'ha enviat des de l'origen i què és soroll afegit.

El 1972 es van fer públics els primers resultats del tractament digital del senyal aplicat a àudio, bàsicament orientat al seu emmagatzemament. El CD ja sortia a la llum. Convertir un so (una magnitud física que pot adoptar qualsevol valor en qualsevol moment) en una sèrie de 0 i 1 (dos únics valors, coneguts) permetia corregir amb facilitat qualsevol soroll afegit.

Figura 8.



En el cas del senyal analògic, veient el senyal rebut, no es pot deduir quin ha estat el senyal emès. En canvi, en el cas del senyal digital, com que es coneixen els valors enviats, s'estableix un llindar en el punt mitjà entre els dos valors i es decideix que tot el que estigui per sobre correspon a un 1 i tot el que estigui per sota, a un 0. Si el soroll que s'ha afegit és superior a la diferència entre el valor original i el llindar, es produeix un error de recepció: es decideix que s'havia enviat el valor equivocat. Les tècniques per a lluitar contra aquest tipus d'errors es veuran més endavant.

ANOTACIONS

El descobriment del processament digital del senyal, i també les seves aplicacions en els camps del so i la imatge, ha constituït una fita capital en el món de les comunicacions. Bàsicament, ha permès reduir dràsticament l'efecte del soroll, cosa que ha possibilitat, d'una banda, incrementar la qualitat de recepció dels senyals i, de l'altra, augmentar la velocitat de transmissió amb els mateixos mitjans.

Les companyies telefòniques van començar a substituir els enllaços interns (entre centrals) per enllaços digitals, però mantenint el bucle d'abonat (línia i terminal) analògic. La digitalització del senyal de so es duu a terme dins de la central local, després del filtre de 4 kHz, i es torna a passar a analògic a la central corresponent a l'altre extrem de la comunicació. La digitalització ha fet canviar substancialment els processos de commutació: ara s'ha de treballar amb bits i, per tant, les centrals electromecàniques s'han de substituir per ordinadors.

La digitalització de la part interna de la xarxa de veu va fer que, d'alguna manera, les dues xarxes, la telefònica i la de dades, confluïssin: els enllaços digitals entre centrals s'utilitzaven indistintament per a paquets de dades i per a transmissions de veu.

1.4.1. La xarxa digital de serveis integrats

Nota

La xarxa digital de serveis integrats (XDSI) correspon a la sigla en anglès *ISDN* (*integrated services digital network*).

Una vegada digitalitzada la xarxa telefònica, el pas següent havia de ser portar la transmissió de bits fins a les cases. Això permetia, d'una banda, oferir als usuaris a casa seva la transmissió de dades a més de la tradicional de veu i, de l'altra, un ventall de serveis nous associats a una comunicació enterament digital d'extrem a extrem. Aquest servei de transmissió digital per mitjà de la xarxa telefònica es coneix com xarxa digital de serveis integrats (XDSI). Ofereix dos canals independents de 64 kbps, que permeten parlar i connectar-se a Internet simultàniament, o, amb el maquinari adequat, aprofitar els dos canals junts per a navegar a 128 kbps.

1.5. La banda ampla

L'ús de la xarxa telefònica per a transmetre dades té una limitació important pel que respecta al màxim de bits per segon permesos, i les xarxes específiques de dades són molt cares per a l'ús domèstic. Des de la dècada dels noranta, s'han estudiat maneres de portar fins a les cases o les empreses un bon cabal de bits per segon (banda ampla) a un preu raonable, de manera que les noves aplicacions multimèdia puguin ser explotades al màxim.

Per a aconseguir aquesta banda ampla, s'han seguit dos camins completament diferents:

- S'han promogut cablatges nous amb fibra òptica, amb freqüència implementats per empreses amb afany competidor contra els monopolis dominants. Aquestes xarxes s'aprofiten per a proporcionar un servei integral: televisió, telèfon i dades.

- Les companyies telefòniques de tota la vida han volgut treure partit del cablatge que ja tenien fet, i per això s'han desenvolupat les tecnologies ADSL, que permeten la convivència al bucle d'abonat del senyal telefònic i d'un senyal de dades que pot arribar als 8 Mbps.

> ***Nota***
>
> La frontera entre banda estreta i banda ampla no és gaire clara. Els 128 kbps de l'XDSI es consideren banda estreta, i hi ha qui qualifica de banda ampla els 256 kbps de l'ADSL bàsica.
>
> Realment, es considera banda ampla a partir d'1 Mbps.

1.6. La telefonia mòbil

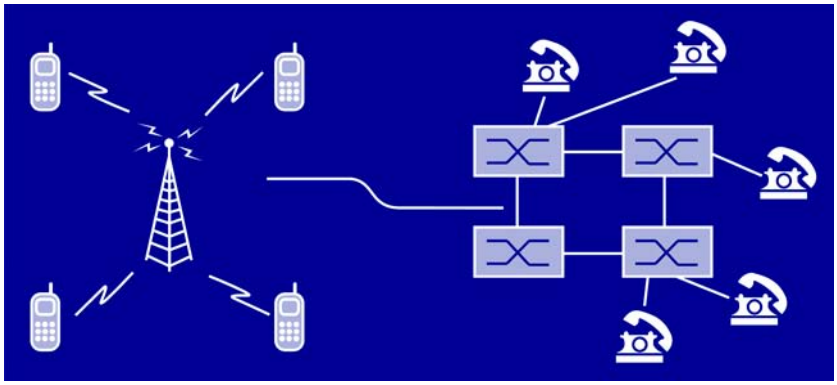
La telefonia mòbil, tot un fenomen sociològic al final del segle XX, ha viscut una evolució fulgurant: en menys de vint anys, ha passat del

ANOTACIONS

no-res a constituir una tecnologia d'ús diari per a més d'un 70% de la població.

Des del punt de vista de sistema de comunicació, hem de veure els mòbils com una extensió de la xarxa telefònica convencional.

Figura 9.



El sistema GSM, que constitueix l'actual estàndard europeu, permet l'accés a la xarxa de veu, canviant el bucle d'abonat: en lloc de ser un cable, és un enllaç radioelèctric entre una antena i el mòbil. Es tracta, per tant, d'una xarxa de commutació de circuits i es continua fixant la tarifa per temps de connexió.

L'estàndard GPRS permet el transport de bits, pagant per trànsit en lloc de per temps. Per tant, és aproximadament el clònic de les xarxes de dades amb fil.

L'estàndard UMTS, actualment encara en la fase prèvia al seu llançament comercial, permet transferències de l'ordre de megabits per segon, necessàries per a disposar d'aplicacions multimèdia al mòbil. Tanmateix, requereix noves antenes i terminals.

ANOTACIONS

2. Arquitectures de protocols: el model OSI

Com ja hem comentat, quan el CCITT i l'ISO van proposar la torre OSI, en el mercat hi havia moltes arquitectures de protocols, unes propietàries, altres obertes, però totes diferents. La torre OSI pretenia ser un model bàsic de referència, un marc per al desenvolupament d'estàndards que permetessin la interoperabilitat completa. Diferents raons han fet que aquest model, i també les normes que se'n deriven, no hagin tingut la repercussió que s'esperava, entre les quals destaquen les següents:

- La complexitat del model, innecessària en molts casos.
- La complexitat de les normes desenvolupades a partir del model.
- L'impuls del model Internet i la simplicitat dels seus estàndards.

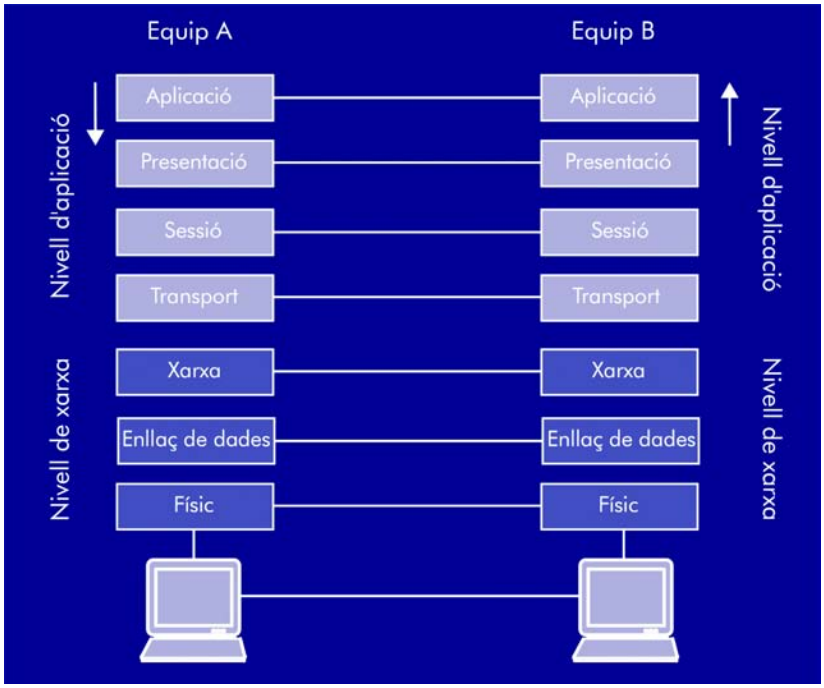
Malgrat que el model OSI no s'hagi imposat en els desenvolupaments, és molt útil com a referència per a explicar què s'ha de fer i com. El fet que sigui tan complet i cartesià ho fa molt interessant per a la pedagogia dels conceptes bàsics de xarxes, i les arquitectures que en realitat s'utilitzen s'expliquen establint una relació constant amb el model OSI. És per això que en aquest apartat expliquem els set nivells de la torre OSI. A partir del mòdul següent, tanmateix, ens centrarem en l'arquitectura TCP/IP, la que constitueix la xarxa Internet.

2.1. Definició

El model bàsic de referència OSI, o simplement model OSI, afronta el problema de les comunicacions de dades i les xarxes informàtiques dividint-lo en nivells. Cada participant de la comunicació n'incorpora com a mínim un, i els equips terminals els incorporen tots.

ANOTACIONS

Figura 10.



Els nivells de la torre es comuniquen en dues direccions:

- **Horitzontal**. La comunicació horitzontal només es dóna entre nivells homònims. Es podria pensar –i de fet és així– que tot el nivell constitueix un únic sistema distribuït que té un representant en cada un dels equips. Un **protocol de nivell *i*** (en el qual *i* és l'identificador del nivell corresponent) especifica el format, el significat i la temporització de la informació que circula entre els membres d'aquest sistema distribuït.
- **Vertical**. La comunicació vertical només es dóna entre nivells adjacents d'un mateix sistema. Aquest tipus de comunicació té un caràcter totalment local; és a dir, es pot materialitzar per mecanismes de programari (crides a llibreries, comunicació entre processos, etc.). De manera genèrica, denominarem aquests mecanismes *servei de nivell i* (en el qual *i* és l'identificador del nivell que proporciona el servei, i *i* + 1, el nivell que l'utilitza).

2.2. Els protocols

Amb els protocols es pretén la intercomunicació d'entitats situades en màquines diferents. Entenem per *entitat* un sistema electrònic i/o in-

ANOTACIONS

formàtic, ubicat dins d'un nivell del model OSI, que, en combinació amb les altres entitats del mateix nivell situades en altres sistemes, forma un tot (un sistema distribuït).

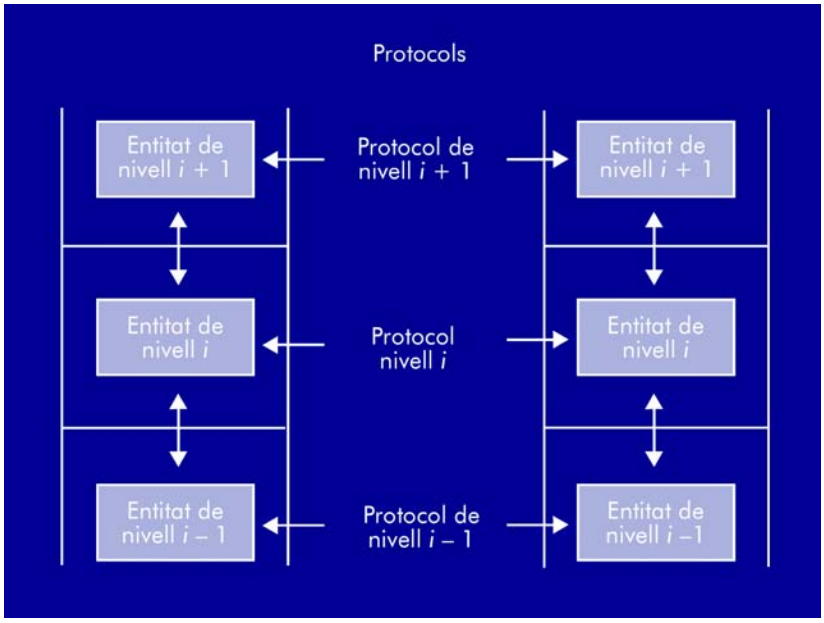
Per tant, l'**especificació del protocol** que utilitzem s'ha de dur a terme en un estàndard clarament definit que permeti a desenvolupadors que no treballen junts implementar-lo de manera totalment idèntica.

La recepció d'una seqüència de bits en un moment inesperat o d'una longitud incorrecta, o en una disposició imprevista, pot fer que l'entitat destinatària no reaccioni correctament i deixi tot seguit el nivell (les dues entitats que el formen) en una situació inestable.

Evidentment, aquesta situació no es pot permetre. És per això que la implementació del protocol ha de ser extremadament acurada i, per tant, també l'especificació de l'estàndard.

En un sistema trobem tants protocols com nivells el formin. Els sistemes a què es connecti directament hauran de tenir la mateixa especificació que els estàndards per a tots els nivells que implementi el protocol.

Figura 11.



ANOTACIONS

2.3. Els serveis

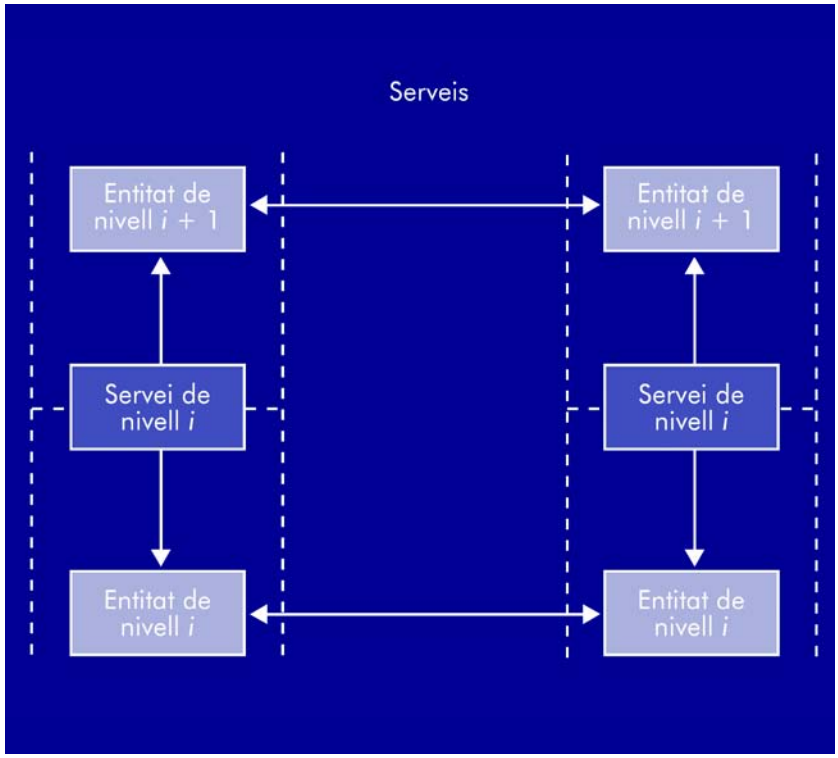
Nota

En terminologia OSI se sol dir que els serveis no s'especifiquen, sinó que es descriuen.

L'especificació d'un servei és sempre menys estricta que la d'un protocol. Per *servei* entenem la comunicació que es produeix dins d'una mateixa màquina i, per tant, dins d'un únic àmbit de responsabilitat. La funcionalitat de les interfícies de cada un dels nivells (i, per tant, de les entitats que la implementen), la determinaran els estàndards que utilitzin; la seva especificació precisa no és rellevant per als estàndards involucrats. Cada sistema individual les pot materialitzar d'una manera o altra segons convingui.

Sigui com vulgui, la quantitat de paper que ocupa la descripció d'un servei sempre serà molt inferior a la que ocupa l'especificació d'un protocol.

Figura 12.



Activitat

Comenteu les diferències existents entre protocol i servei.

ANOTACIONS

2.4. Els set nivells del model OSI

2.4.1. Nivell físic

El nivell físic s'encarrega de les tasques de transmissió física dels senyals elèctrics (o electromagnètics) entre els diferents sistemes. Les limitacions del nivell físic (equips de transmissió i recepció, medis de transmissió, amplificadors, etc.) n'imposen d'altres a la resta del sistema: d'una banda, limiten la **velocitat de transmissió** (en bits per segon) i, de l'altra, fan aparèixer una **probabilitat d'error**, el percentatge de bits erronis que arriben a la destinació.

La primera limitació és gairebé insalvable partint d'un **medi de transmissió** donat, ja que els seus paràmetres físics imposen un límit superior no superable per mitjà d'una millora tecnològica. Els medis de transmissió tenen una **capacitat** de transmissió delimitada i l'electrònica que utilitzem per a dur a terme les transmissions pot millorar la velocitat de transmissió, però no superar aquest límit. Aquesta limitació ve determinada per l'amplada de banda, o amplada de l'espectre elèctric, que pot travessar el medi de transmissió (doblar l'amplada de banda significa que es pot doblar la velocitat de transmissió) i per la impossibilitat pràctica de rebre el senyal lliure de qualsevol interferència.

Nota

En el nivell físic som incapaços de corregir errors. Assumim una probabilitat d'error i n'encarreguem al nivell superior la correcció.

2.4.2. Nivell d'enllaç

El nivell d'enllaç és el primer de la torre OSI que es basa en programari, algorismes i protocols. La seva missió principal és donar fiabilitat a la transmissió dels senyals elèctrics o electromagnètics que proporciona el nivell físic, la qual cosa es pot aconseguir si les quotes d'error són inferiors a l'1%. S'afegeixen bits addicionals als que formen el missatge per poder detectar errors de transmissió i demanar-ne la retransmissió. Per a això, és necessari conferir una estructura als bits: s'agrupen en petits blocs denominats ***trames***, que contenen els bits de missatge, els bits afegits per a detectar errors i diferents camps de control, com ara el número de trama.

Nota

El fet que les trames siguin petits blocs de bits minimitza la probabilitat que hi hagi molts bits erronis dins dels blocs.

El transmissor calcula aquests bits addicionals a partir de la resta per mitjà d'una operació que el receptor també coneix i aplica. Si el re-

ANOTACIONS

ceptor detecta una discrepància entre els bits addicionals (redundants) i els que ha calculat a partir de la resta, detecta que el bloc és erroni i en demanarà la retransmissió.

L'addició dels bits redundants i la seva comparació en recepció es denomina *detecció d'errors*. Els procediments de correcció a partir de la detecció esmentada es coneixen com *control d'errors*.

A més del control d'errors, el nivell d'enllaç duu a terme una altra tasca important: el *control de flux*.

El receptor ha de processar les trames a mesura que les rep. En alguns casos, aquest procés comporta una despesa de temps mínim, tenint en compte la velocitat de transmissió (per exemple, desar les dades en disc); tanmateix, hi pot haver casos en els quals aquest procés sigui costós. En aquesta situació, el receptor necessita un mecanisme que notifiqui al transmissor que ha de detenir momentàniament la transmissió amb l'objectiu de disposar del temps necessari per a dur a terme aquesta tasca.

El nivell d'enllaç no solament serveix per a controlar línies punt a punt, sinó també per a controlar línies compartides per diferents terminals (xarxes d'àrea local).

2.4.3. Nivell de xarxa

El nivell de xarxa és el que permet que hi pugui haver més de dues màquines involucrades en les interconnexions. Si només es tingués el nivell d'enllaç, això no seria possible. El nivell d'enllaç s'ocupa que els bits arribin d'una costat a un altre, per tant, només permet interconnectar dues màquines. Per a poder interconnectar més de dues màquines, necessitem identificar-les i connectar-les d'alguna manera. Aquesta és la tasca del nivell de xarxa.

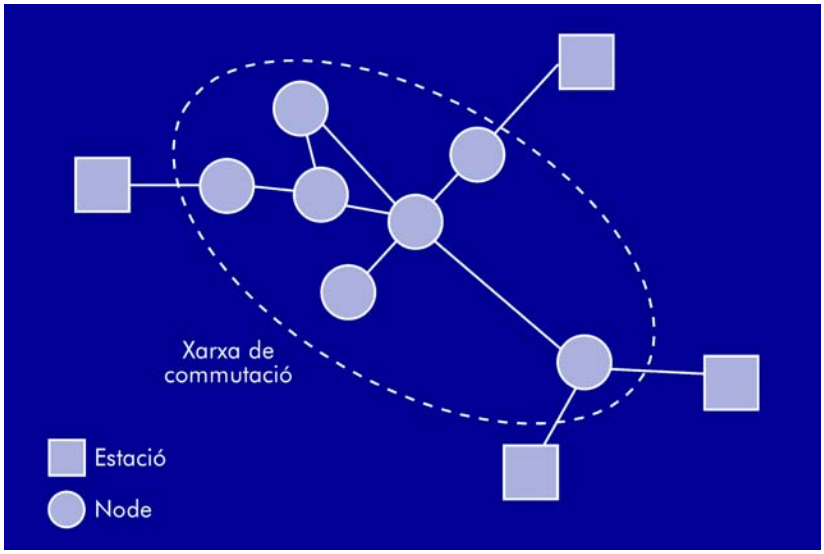
Ja hem vist que les xarxes de commutació de paquets constitueixen el tipus de xarxa més eficient per a transmetre dades des de diferents

ANOTACIONS

punts de vista: ús de recursos, cost, capacitat de mantenir diferents connexions simultànies, etc. El model OSI, per tant, només parla de xarxes de commutació de paquets.

En el nivell de xarxa es distingeix entre estacions terminals i nodes de commutació:

Figura 13.



La paraula *xarxa* prové d'aquesta imatge: els enllaços són els cordills que uneixen els nusos o sistemes.

Els nodes de commutació disposen de diferents enllaços cap a altres nodes o cap a terminals, i són els que permeten que els paquets viatgin per la xarxa des d'una estació terminal a una altra.

Hi ha dos tipus de xarxes de commutació de paquets:

- Xarxes que funcionen en mode **datagrama**. Podríem dir que aquest tipus de xarxes són les bàsiques, ja que incorporen la funcionalitat mínima perquè un grup de nodes i de terminals interconnectats puguin fer passar informació d'un punt a un altre. El problema de les xarxes en mode datagrama resideix en la dificultat de garantir el lliurament correcte i complet de la informació, ja que els diferents paquets que formen la transmissió no mantenen entre si un vincle conegut per la xarxa. Els paquets poden arribar fora d'ordre, duplicats, o fins i tot es poden perdre sense que la

ANOTACIONS

xarxa pugui fer gran cosa sobre això. Es deixa al terminal receptor la responsabilitat de restaurar els possibles danys que hagi tingut el paquet durant la transmissió.

- Xarxes que funcionen en mode **circuit virtual**. Aquestes xarxes poden garantir que el lliurament dels paquets sigui correcte i complet, i ho fan aportant el concepte de **connexió** propi de les xarxes de commutació de circuits. És el circuit virtual. Aquest circuit permet agrupar els paquets relacionats de manera que el receptor els rebi correctament sense problemes d'ordre, duplicació o pèrdua.

L'**assignació d'adreces** és un dels conceptes bàsics del nivell de xarxa. Li permet, com a sistema distribuït però únic, decidir quin dels múltiples terminals és el destinatari final de cada paquet.

L'encaminament constitueix el procediment que permet a aquest sistema distribuït conduir la informació pels diferents nodes d'origen a destinació, minimitzant el trajecte i el temps de trànsit, optimitzant recursos, etc.

2.4.4. Nivell de transport

El nivell de transport permet una connexió fiable sobre qualsevol tipus de xarxa (fiable o no). A les xarxes de commutació de paquets en mode datagrama és on aquest nivell revela la seva importància, ja que és el responsable de controlar les possibles deficiències de les transmissions.

La funció principal d'aquest nivell consisteix a assegurar la qualitat de transmissió entre els terminals que utilitzen la xarxa, la qual cosa implica recuperar errors, ordenar correctament la informació, ajustar la velocitat de transmissió de la informació (control de flux), etc.

2.4.5. Nivells de sessió, presentació i aplicació

Aquests tres nivells se solen explicar de manera conjunta, ja que hi ha pocs exemples pràctics de protocols de sessió i de presentació. A més, l'arquitectura Internet delega tots els treballs per sobre de trans-

ANOTACIONS

port a l'aplicació. No obstant això, en el model OSI es defineixen com tres nivells diferents i independents, amb atribucions pròpies.

El nivell de sessió és, en teoria, l'encarregat de gestionar les connexions de llarga durada, la recuperació de caigudes de xarxa de manera transparent i els protocols de sincronia entre aplicacions.

El nivell de presentació s'encarrega d'aconseguir que les diferents plataformes (sistemes operatius, processadors, etc.) es puguin entendre en connectar-se per mitjà d'una mateixa xarxa. Dit d'una altra manera, soluciona el problema de l'heterogeneïtat definint una manera universal de codificar la informació. Aquesta codificació pot tenir propietats d'eficiència (per mitjà de la compressió, per exemple), propietats de confidencialitat (per mitjà de la criptografia), etc.

En el nivell d'aplicació resideixen els programes. En aquest nivell podem trobar servidors, clients que accedeixen a aquests últims, aplicacions que treballen segons un model simètric (*peer-to-peer*), etc.

Activitat

Assigneu els diferents nivells de les xarxes que coneixeu a les funcions explicades en aquest apartat.

ANOTACIONS

II. Xarxes d'àrea local

3. Les xarxes d'àrea local

Una xarxa d'àrea local és un sistema que permet la interconnexió d'ordinadors propers físicament. Entenem per *proper* tot el que no sigui creuar una via pública: una habitació, un edifici, un campus universitari, etc.

En el moment en què una xarxa ha de creuar un carrer, o una via pública en general, és necessari que una companyia de telecomunicacions estableixi la comunicació, ja que són les úniques autoritzades per a passar línies per zones públiques.

Una definició més precisa de xarxa d'àrea local prescindeix de la distància entre les estacions i especifica que el seu caràcter distintiu resideix en el fet que els mecanismes d'enllaç entre estacions han d'estar completament sota el control de la persona o entitat que estableix aquesta xarxa.

Com comentàvem en la primera unitat, l'objectiu que es perseguia quan es van proposar les primeres xarxes d'àrea local era compartir recursos entre diferents ordinadors (un sistema d'emmagatzematge massiu, una impressora o un dispositiu de connexió cap a l'exterior, per exemple). Per a aquest tipus de comunicacions es va proposar una filosofia de disseny basada en la difusió de trames amb medi compartit, de manera que quan una estació posa una trama al medi, la resta d'estacions la puguin rebre. Els receptors reals de la trama se la queden i la resta, la ignora.

Nota

Les primeres xarxes d'àrea local només permetien que un dels ordinadors de la xarxa (el servidor) oferís recursos a la resta, que només podien actuar com a clients d'aquest servidor, sense capacitat d'oferir res. D'un temps ençà, el

ANOTACIONS

programari de xarxa que elaboren empreses com Novell, Microsoft o Apple permet que totes les estacions puguin actuar com a servidors i clients alhora.

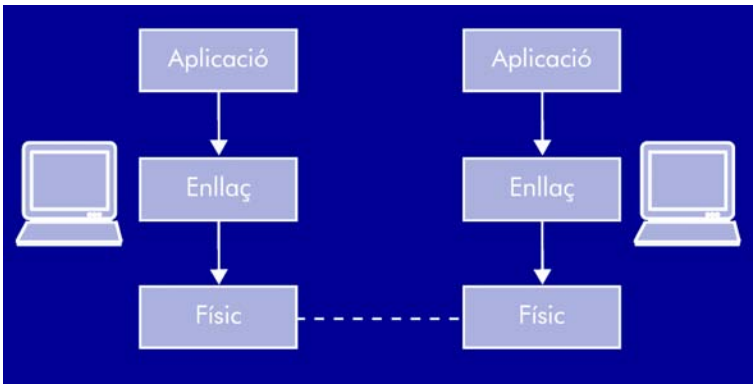
Últimament i com veurem més endavant, s'han aplicat tècniques de commutació a les xarxes d'àrea local, per a aconseguir millorar-ne el rendiment.

Una altra millora important ha estat l'aparició de les xarxes d'àrea local sense fil (*wireless LAN*), en les quals l'enllaç entre estacions no es duu a terme per mitjà de cables, sinó per mitjà d'enllaços radioelèctrics. Els avantatges d'aquest tipus d'enllaços, quant a mobilitat i facilitat d'instal·lació, són evidents.

En una xarxa és imprescindible identificar els ordinadors que en formen part. Quan un ordinador genera una trama per a un altre, a més de les dades que li vol enviar, li posa l'identificador de l'ordinador (o ordinadors) destinació i el seu, perquè qui rebi la trama pugui saber qui la hi ha enviat.

Per a construir una xarxa local, es necessiten bàsicament dues coses: maquinari (targetes, cables, connectors) i un programari que sigui conscient que hi ha diferents màquines connectades i ofereixi els serveis necessaris perquè les aplicacions els puguin aprofitar. El més lògic és que aquest programari s'integri al sistema operatiu i ofereixi a les aplicacions la visió de la xarxa com un recurs propi més. Aquests recursos de maquinari i programari necessaris es poden analitzar des del punt de vista de la torre OSI, com s'explicava en la unitat anterior:

Figura 14.



ANOTACIONS

Com es pot veure en la figura anterior, els nivells necessaris per a implementar una xarxa d'àrea local són els dos inferiors (físic i enllaç) i el superior (aplicació). En l'àmbit d'usuari no som conscients d'aquesta subdivisió perquè, com hem dit, el codi que implementa els serveis associats als nivells està integrat al sistema operatiu de les estacions.

Nota

Els nivells xarxa, transport, sessió i presentació tenen sentit en xarxes d'àrea extensa, com veurem en les unitats següents.

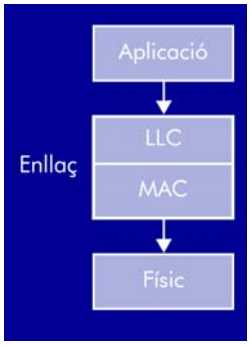
El nivell físic correspon al maquinari: a la targeta de xarxa, als senyals electromagnètics que viatgen pel medi de transmissió, als dispositius que generen aquests senyals a partir de bits, etc.

El nivell d'enllaç, com ja sabem, proporciona fiabilitat en l'intercanvi de trames entre les estacions: bàsicament control d'errors i control de flux. Però, pel fet d'usar un medi compartit, caldrà establir mecanismes perquè totes les estacions el puguin usar quan el necessitin, però sense molestar-se. Si dues estacions posen trames al medi de transmissió simultàniament, es barrejaran de manera que es convertiran en una cosa inintel·ligible. Aquesta situació es coneix com *col·lisió de trames*: necessitem mecanismes per a controlar l'accés al medi compartit de manera que no es produeixin col·lisions, o que si es produeixen, la xarxa es pugui recuperar i continuar funcionant.

La inclusió d'aquests mecanismes a la torre OSI es podia dur a terme afegint-hi un nivell més o, com al final va succeir, incloent-los en el nivell d'enllaç. Així, en contextos d'àrea local, el nivell d'enllaç inclou dos subnivells:

- MAC (control d'accés al medi o *medium access control*), que s'encarrega pròpiament de la política d'accés al medi.
- LLC (control de l'enllaç lògic o *logical link control*), que s'encarrega dels serveis típics d'enllaç: control d'errors i control de flux.

Figura 15.



ANOTACIONS

4. Topologies de les LAN

El primer que caracteritza una xarxa local és la manera en què es connecten les estacions; és a dir, la forma que adopta el medi que comparteixen. Bàsicament hi ha tres topologies possibles:

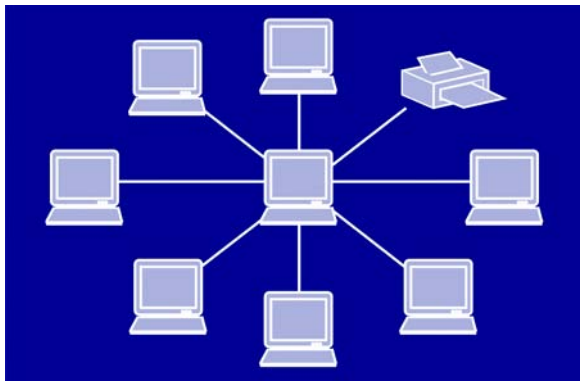
- Topologia en estrella.
- Topologia en bus.
- Topologia en anell.

4.1. Topologia en estrella

La topologia en estrella consisteix a connectar cada ordinador a un punt central, que pot ser tan senzill com una simple unió física dels cables.

Quan un ordinador posa una trama a la xarxa, aquesta apareix tot seguit en les entrades de la resta d'ordinadors.

Figura 16.



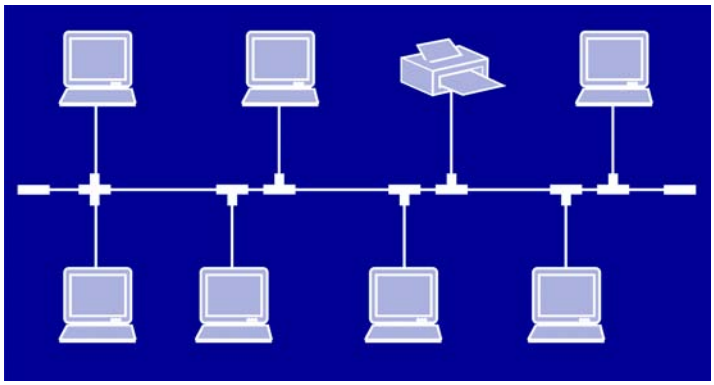
Encara que s'hagin definit estàndards per a aquest tipus de xarxes, actualment ja gairebé no existeixen, ja que no aporten cap avantatge sobre la resta i sí molts inconvenients.

ANOTACIONS

4.2. Topologia en bus

La topologia en bus consisteix en un cable al qual s'uneixen totes les estacions de la xarxa.

Figura 17.



Tots els ordinadors estan pendents de si hi ha activitat en el cable. Quan un ordinador hi posa una trama, tots els ordinadors l'agafen i miren si en són el destinatari. Si és així, se la queden, en cas contrari, la descarten.

Les primeres xarxes en bus utilitzaven un cable coaxial gruixut, connectors tipus BNC, i els ordinadors s'hi connectaven amb un dispositiu denominat *transceptor* (*transceiver*), que era exterior. Posteriorment, en va aparèixer una nova versió, amb un cable més fi (*thin-ethernet*) i amb uns transceptors més petits, de manera que es podien integrar a l'adaptador de xarxa i així no es veien.

Nota

Els capricis de l'electrònica exigeixen que el cable estigui "tapat" en els dos extrems, perquè els bits no es "perdin". Això es duu a terme amb una resistència de càrrega.

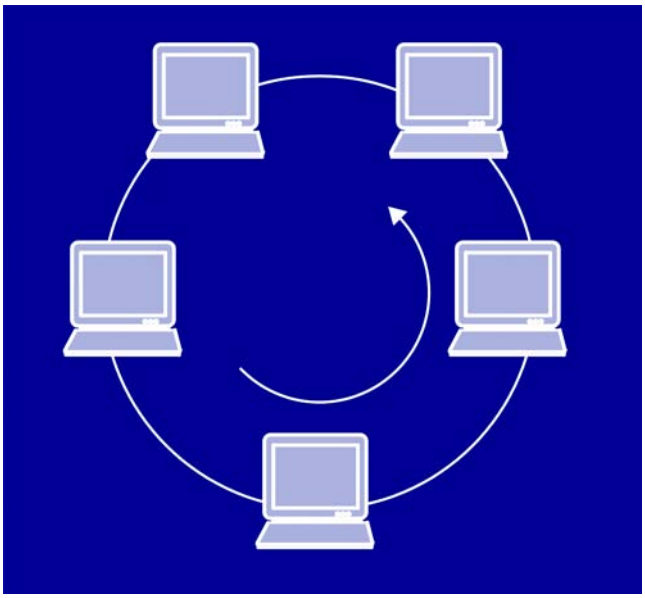
Quan un ordinador posa una trama al cable, aquesta recorre el cable per complet en els dos sentits fins als extrems, on és absorbida pels taps.

ANOTACIONS

4.3. Topologia en anell

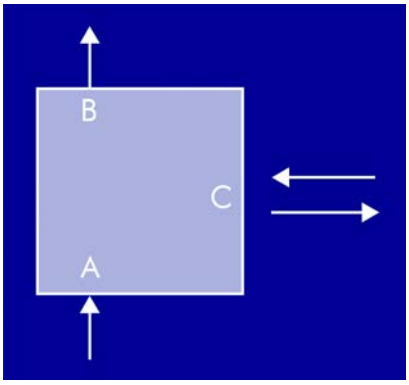
La topologia en anell consisteix a connectar cada ordinador a dos més, de manera que es formi un anell. Quan un ordinador vol enviar una trama a un altre, aquesta ha de passar per tots els ordinadors que hi hagi entre ells: la circulació per l'anell és unidireccional.

Figura 18.



El dispositiu que connecta l'ordinador a l'anell és el **repetidor**, un circuit amb tres connexions:

Figura 19.



- Connexió d'entrada de trames des de l'anell a l'ordinador (A).

ANOTACIONS

- Connexió de sortida de trames des de l'ordinador a l'anell (B).
- Connexió bidireccional, per la qual passen totes les trames que entren i surten de l'ordinador (C).

El repetidor té tres modes de treball:

- Mode escolta: el repetidor pren les trames que li arriben per A i les posa simultàniament a B i C, perquè continuïn per l'anell i perquè l'ordinador en rebi una còpia i l'analitzi. Si és el destinatari de la trama, se la queda, i en cas contrari, la descarta.
- Mode transmissió: l'ordinador envia informació a la xarxa. Posa una trama a C, de manera que creua el repetidor i surt per B cap a l'ordinador següent de l'anell.
- Mode curtcircuit: les trames que arriben per A es posen directament a B sense proporcionar-ne una còpia a l'ordinador. Aquest mode serveix perquè l'anell continuï funcionant si l'ordinador corresponent està inactiu.

4.4. Pseudotopologia de les xarxes sense fil

Parlar de topologia en una xarxa sense fil sembla fora de lloc, perquè no "veiem" cap medi de transmissió. Però en realitat l'"èter" per on viatgen les ones es considera un medi de transmissió, i si el comparem amb les tres topologies descrites, veiem que es pot comparar a la topologia en bus.

Nota

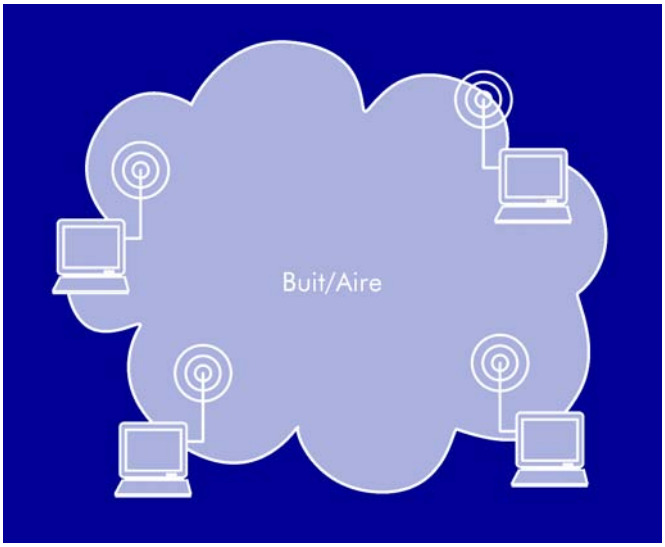
De fet, les ones electromagnètiques no necessiten cap suport físic per a ser transmeses. Es propaguen en el buit. Però fins que això no es va demostrar, els científics utilitzaven el terme *èter* per a designar una cosa que s'imaginaven que havia d'existir però eren incapaços de veure.

En un anell o en una estrella en realitat hi ha *n* medis independents que connecten una estació a una altra (o al punt central), mentre que

ANOTACIONS

en un bus tenim només un medi (un cable) a què es connecten totes les estacions, de la mateixa manera que en una xarxa sense fil tenim un sol medi (l'aire) on les estacions posen les seves trames.

Figura 20.



La topologia, com veurem més endavant, condiciona els mecanismes d'accés que es poden usar en una xarxa local. En el cas de les xarxes sense fil això és particularment determinant.

ANOTACIONS

5. Cablatge estructurat

Les topologies en bus i en anell comporten un seriós problema de cablatge a l'hora d'implementar-les. Encara que sigui relativament senzill muntar una xarxa en bus o en anell, és molt complicat mantenir-la i ampliar-la: quan falla un cable o una connexió, la xarxa sencera deixa de funcionar, i no és senzill localitzar el punt exacte on es troba l'error. És necessari comprovar la xarxa sencera, cosa que en nombroses ocasions és complicada, ja que els cables poden passar per falsos sostres o conduccions de difícil accés.

Aquest problema ha fet pensar en un nou disseny de xarxa més controlable: el cablatge estructurat.

El cablatge estructurat consisteix a fer una preinstal·lació de xarxa similar a la de les xarxes telefòniques. A cada punt de treball es fan arribar dues línies: una per al telèfon i una altra per a les dades. Tots els cables arriben a una habitació, on s'estableixen les connexions: els cables de telèfon es direccionen cap a la centraleta i els de les dades, cap a un dispositiu que permet la interconnexió en xarxa local.

El 1991 es va publicar l'EIA/TIA 568 sobre cablatge de telecomunicacions per a edificis comercials. El propòsit d'aquest estàndard és:

- Ser universal, tant en serveis suportats com en fabricants compatibles.
- Ser base per al desenvolupament d'altres estàndards de comunicacions (veu, imatge, LAN, WAN).
- Definir paràmetres que permetin definir i establir el cablatge de l'edifici fins i tot abans que algú l'ocupi. Es concep el cablatge com un servei més de l'edifici (llum, aigua, gas... i dades).

L'estàndard especifica els senyals que s'han d'usar, i també els aspectes mecànics dels cables, els connectors, els armaris, etc.

Nota

EIA/TIA:

Electronic Industries Association / Telecommunication Industry Association.

ANOTACIONS

Per norma general, es realitza un cablatge a dos nivells:

- Cablatge **horitzontal**: en cada planta (si és necessari cablar-ne diverses) es posen cables des d'un armari fins als punts terminals.
- Cablatge **vertical**: des de cada armari de planta es posen cables fins a una habitació de l'edifici on es troben els dispositius de xarxa, els encaminadors o *routers* cap a l'exterior, la centraleta telefònica, etc.

Nota

Parlant amb propietat direm que una xarxa té topologia física en estrella i topologia lògica en bus.

En cada planta necessitem crear una xarxa local en el punt on conflueixen els cables que provenen de cada una de les estacions. Sembla que una topologia en estrella seria la més adequada, però com hem comentat, tal com s'havia concebut, era la que oferia menys prestacions. La solució és combinar els avantatges de la topologia física en estrella amb el funcionament dels busos o els anells. És a dir, usar per a interconnectar les estacions un dispositiu, allotjat a l'armari de planta, que es comporti com un bus o com un anell. En el cas del bus (la topologia més utilitzada actualment), aquest dispositiu es coneix com a concentrador o, en anglès, *hub*.

Lectura complementària

Una bona aproximació al funcionament dels concentradors i els commutadors la trobareu a:

A.S. **Tanenbaum** (2003). *Redes de computadores*. Mèxic: Pearson Educación.

Una topologia així, on l'element central és un dispositiu actiu que simula un dispositiu passiu, va portar al desenvolupament de les LAN commutades. El raonament és el següent: quan el concentrador rep una trama, per a comportar-se com un bus l'ha de reenviar cap a la resta d'estacions. Però el concentrador té capacitat de procés: pot analitzar la trama i, en particular, pot esbrinar-ne el destinatari. Llavors, si el concentrador coneix els identificadors de les diferents estacions que té connectades, pot enviar la trama únicament al seu destinatari, cosa que disminueix el nombre de trames a la xarxa, i, per tant, n'augmenta l'eficiència. Els dispositius que es comporten així es denominen *commutadors* (en anglès, *switch*).

Nota

Parlarem de FastEthernet i Gigabit Ethernet en l'apartat següent.

Pel que es refereix al medi físic, s'usen tant parells de coure trenats com fibra òptica, encara que en molta major mesura els primers pel seu menor cost per a prestacions similars. S'han especificat categories de cables, cadascun amb unes capacitats i uns requisits mínims a complir. Avui en dia el més usat és el cable categoria 5e, que permet una amplada de banda de 100 MHz, el requerit per a les LAN d'alta velocitat, com Fast Ethernet i Gigabit Ethernet.

ANOTACIONS

Els costos d'instal·lació d'un sistema de cablatge estructurat són molt alts; però el seu manteniment és molt simple i barat.

Si falla un cable, només falla una estació de treball, no tota la xarxa, i, si falla tota la xarxa, és que s'ha espatllat el concentrador. Tant un cas com l'altre són molt ràpids de solucionar.

Activitat

Els que tingueu accés a una instal·lació amb cablatge estructurat, estudieu-la: observeu-ne les connexions, els cables, els armaris de planta, els commutadors, etc.

ANOTACIONS

6. Control d'accés al medi

Atès que qualsevol ordinador de la xarxa pot posar trames al medi compartit, cal establir mecanismes de control que regulin aquest accés de manera eficient, justa i fiable.

El control d'accés al medi (MAC) és un mecanisme que decideix quina estació té accés al medi de transmissió per a emetre una trama d'informació.

En general, els protocols d'accés es poden classificar en tres grans grups:

- Control d'accés estàtic
- Control d'accés dinàmic (centralitzat o distribuït)
- Control d'accés aleatori

Cada un d'aquests tipus d'accessos té avantatges i inconvenients, i s'apliquen a xarxes molt diferents. De fet, les polítiques d'accés al medi estan molt vinculades a la topologia utilitzada. D'aquesta manera, en una topologia en anell, la manera més natural de controlar l'accés és per **pas de testimoni** (*token passing*), que és un exemple de control dinàmic distribuït. En la topologia en bus, també es pot utilitzar aquest sistema; però, està molt més generalitzat l'ús de la tècnica CSMA/CD, que és de tipus aleatori. A les xarxes sense fil s'usa una política d'accés al medi que és una combinació de control estàtic i aleatori.

En aquesta unitat descriurem les dues polítiques més comunes avui en dia a les xarxes cablades: el pas de testimoni i CSMA/CD.

Lectura complementària

Podeu trobar la definició de tots els tipus de control d'accés al medi a:

A.S. **Tanenbaum** (2003). *Redes de computadores*. Mèxic: Pearson Educación.

Nota

CSMA és la sigla de *carrier sense multiple access* (accés múltiple per detecció de portadora) i *CD* és la sigla de *collision detection* (detecció de col·lisions).

6.1. Pas de testimoni

Com dèiem, la política de pas de testimoni és la més apropiada per a les xarxes en anell. Així doncs, per a descriure'n el funcionament

ANOTACIONS

assumirem que som en una xarxa d'aquesta topologia. En anglès aquestes xarxes es denominen *token-passing ring*, literalment 'anell amb pas de testimoni'.

El funcionament de la política de pas de testimoni és el següent:

Es defineix una trama especial, el testimoni. Quan una estació el rep, té permís per a posar una trama pròpia a la xarxa. Una vegada aquesta trama ha fet tota la volta a la xarxa, i després que els seus destinataris se n'hagin quedat una còpia, l'estació que l'ha posada la treu i allibera el testimoni que arribarà a l'estació següent de l'anell. Aquesta estació repeteix el procediment: treu el testimoni de la xarxa i posa una trama seva o, si no té res per a enviar, passa el testimoni a l'estació següent. Les estacions que tinguin informació per a transmetre han d'esperar a tenir el testimoni per a posar-la a la xarxa.

Aquest mecanisme de control del medi permet amb la mateixa facilitat l'emissió de trames tant a una sola estació com a moltes. La trama recorre tot l'anell, per tant tots els repetidors la veuen passar. Cada un comprova si en el camp "destinatari" de la capçalera de la trama apareix el seu identificador. En cas afirmatiu, se'n queda una còpia i la retransmet cap a l'estació següent. En cas contrari la retransmet sense quedar-se'n còpia.

Les velocitats de treball de les xarxes en anell amb testimoni estan normalitzades: 4, 16 i 100 Mbps. Si s'utilitza fibra òptica com a medi de transmissió, la xarxa, que es denomina *FDDI* (*fiber distributed data interface*), pot superar els 100 Mbps.

Les xarxes de pas de testimoni van ser inventades per IBM. Posteriorment, l'IEEE va elaborar l'estàndard 802.5, que en recollia tota la informació existent.

Nota

IEEE és la sigla de l'Institute of Electric and Electronic Engineers (Institut d'Enginyers Elèctrics i Electrònics).

6.2. CSMA/CD

Com ja hem comentat, CSMA/CD és una política d'accés al medi de tipus aleatori, la qual cosa vol dir bàsicament que les estacions no accedeixen al medi d'una manera prefixada, ho fan quan volen.

ANOTACIONS

D'aquesta manera s'aconsegueix augmentar l'eficiència de la xarxa respecte als sistemes de control estàtics. Òbviament farà falta controlar el cas en què dues estacions vulguin transmetre alhora.

Nota

Els mecanismes de control del tipus estàtic es basen en el fet de repartir l'accés al medi entre les estacions de manera fixa. Si quan a una estació li toca accedir al medi no ha de transmetre res, el lapse de temps que hi té assignat no pot ser aprofitat per cap altra estació i el medi queda desocupat.

De polítiques d'accés tipus aleatori n'hi ha diverses, però les "comercialment útils" són dues, CSMA/CD i CSMA/CA. La primera és la més indicada per a xarxes amb topologia en bus (tant amb un bus real, cablat, com amb un concentrador, en un entorn de cablatge estructurat). La segona és la que s'usa a les xarxes sense fil Wi-Fi, que com hem comentat, tenen una topologia assimilable a un bus.

Vegem en primer lloc com funciona CSMA, per després descriure la funcionalitat addicional de la detecció de col·lisions (CD).

La política d'accés CSMA (accés múltiple per detecció de portadora) funciona de la manera següent:

Els ordinadors escolten constantment el medi (miren si hi ha portadora). Quan tenen una trama per a transmetre, si detecten que no hi ha activitat, la hi posen; en cas contrari, esperen i continuen escoltant fins que quedi lliure, llavors transmeten la seva trama. Si no tenen res per a transmetre, quan detecten una trama al medi, la prenen i la processen.

Aquest algorisme presenta un inconvenient clar: hi ha la possibilitat que dues estacions vulguin enviar una trama en el mateix moment. Totes dues escolten el medi, no detecten activitat i emeten simultàniament. Llavors es produeix una col·lisió: els senyals electromagnètics es barregen i el resultat és una cosa inintel·ligible. El control d'errors que s'efectua en el subnivell LLC serà l'encarregat de detectar aquesta circumstància i sol·licitar la retransmissió de les trames que s'han corromput.

ANOTACIONS

Podem millorar la política CSMA afegint-hi un procediment addicional: quan una estació ja ha començat a transmetre, continua escoltant el medi per a detectar si es produeix una col·lisió; en aquest cas deixa de transmetre immediatament, per a reduir així el temps de col·lisió, i espera un temps aleatori (és una manera d'evitar una nova col·lisió) per a tornar a començar el procés. Aquesta variant rep el nom de *CSMA/CD*, per la detecció de les col·lisions.

Aquest mode de treball marca l'existència d'un paràmetre molt important a les xarxes CSMA/CD, que és la **longitud mínima de trama**.

Una trama en una xarxa local CSMA/CD ha de ser prou llarga perquè una col·lisió sigui detectada abans que en finalitzi la transmissió. Aquesta longitud mínima de trama depén únicament de la velocitat de transmissió dels senyals al medi. Aquest paràmetre, al seu torn, en marca un altre, també molt important en el disseny de xarxes, com és el **diàmetre de la xarxa**, o la distancia entre les estacions més allunyades.

La xarxa Ethernet és una topologia en bus que utilitza la política d'accés CSMA/CD i fixa com a longitud mínima de trama 512 bits. La va inventar Xerox, juntament amb Intel i Digital, i l'IEEE la va elevar a la categoria d'estàndard: IEEE-802.3.

La xarxa Ethernet treballa a 10 Mbps. Posteriorment, va aparèixer la FastEthernet, que treballa a 100 Mbps, i, més tard, la Gigabit Ethernet que, com el seu nom indica, treballa a 1 Gbps.

Activitat

Que una xarxa vagi a 10 Mbps no vol dir que totes les connexions es realitzin a aquesta velocitat. Si teniu accés a una xarxa d'àrea local, d'una banda, esbrineu a quina velocitat funciona, i d'una altra, feu una transferència d'informació entre dues estacions i mesureu la velocitat real a què s'ha realitzat (el quocient entre el nombre de bytes transmesos, multiplicats per vuit, i dividit pel temps que ha durat la transmissió, en segons). Com més gran sigui el fitxer que transmeteu, millor. Podeu fer la prova en diferents condicions de treball, com per exemple amb poques estacions actives, amb moltes transmissions simultànies, etc. Compareu els valors obtinguts amb la velocitat nominal de la xarxa.

ANOTACIONS

III. TCP/IP

7. Estructura de protocols a Internet

El model Internet gira entorn dels protocols TCP/IP. IP és un protocol que proporciona mecanismes d'interconnexió entre xarxes d'àrea local i TCP proporciona mecanismes de control de flux i errors entre els extrems de la comunicació.

No es tracta d'una arquitectura de nivells formal com la torre OSI, que ja hem vist en la unitat 1. De fet, podríem considerar que el model de la xarxa Internet consta només de quatre parts o nivells; és a dir, tot el que hi ha per sota de l'IP, l'IP, el TCP i tot el que hi ha per sobre del TCP:

1. **Per sota d'IP**. Aquest nivell, en l'entorn Internet, se l'anomena *nivell de xarxa local* o, simplement, *nivell de xarxa*. Per norma general, està format per una xarxa LAN, o WAN (de connexió punt a punt) homogènia. Tots els equips connectats a Internet implementen aquest nivell.

2. **Nivell IP o nivell Internet** (*nivell d'Internetworking*). Aquest nivell confereix unitat a tots els membres de la xarxa i, per tant, és el que permet que tots es puguin interconnectar, amb independència de si s'hi connecten per mitjà de línia telefònica, ISDN o una LAN Ethernet. L'encaminament i l'assignació d'adreces constitueixen les seves principals funcions. Tots els equips connectats a Internet implementen aquest nivell.

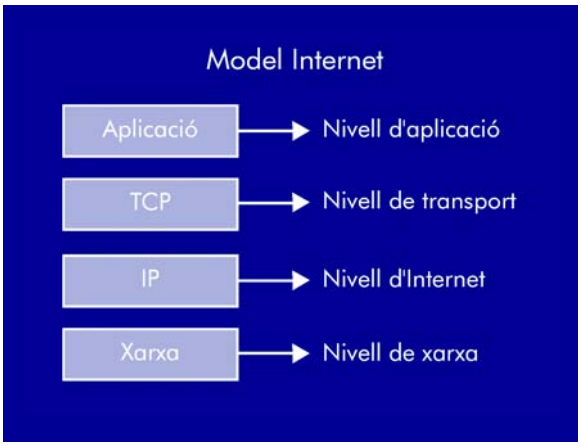
3. **Nivell TCP o nivell de transport**. Aquest nivell confereix fiabilitat a la xarxa. El control de flux i d'errors es duu a terme principalment dins d'aquest nivell, que només implementen els equips usuaris de la xarxa Internet o els terminals d'Internet. Els equips de commutació (encaminadors) no el necessiten.

4. **Per sobre de TCP**. Aquest nivell correspon a les aplicacions que utilitzen Internet, com ara clients i servidors de WWW, correu

ANOTACIONS

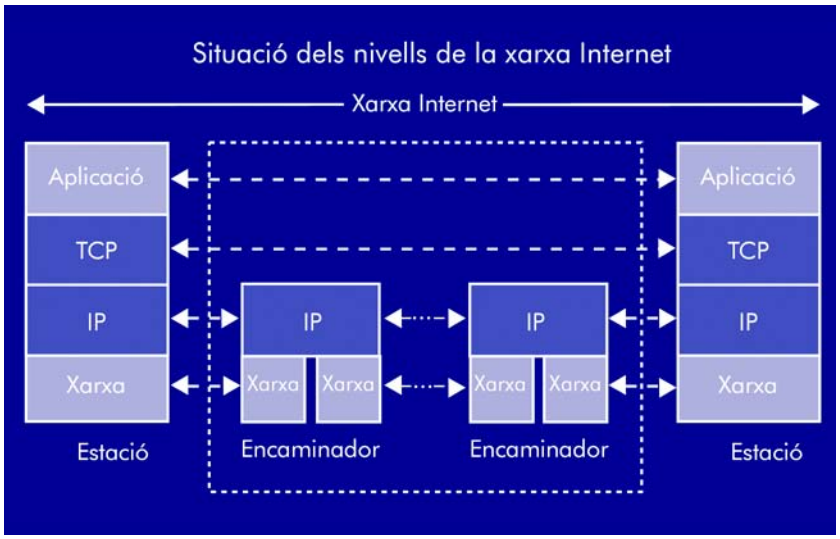
electrònic, FTP, etc. És per això que se'l denomina *nivell d'aplicació*. Només l'implementen pels equips usuaris de la xarxa Internet o els terminals d'Internet. Els equips de commutació no l'utilitzen.

Figura 21.



És important destacar que només els equips terminals implementen tots els nivells; els equips intermedis únicament implementen el nivell de xarxa i el nivell IP.

Figura 22.



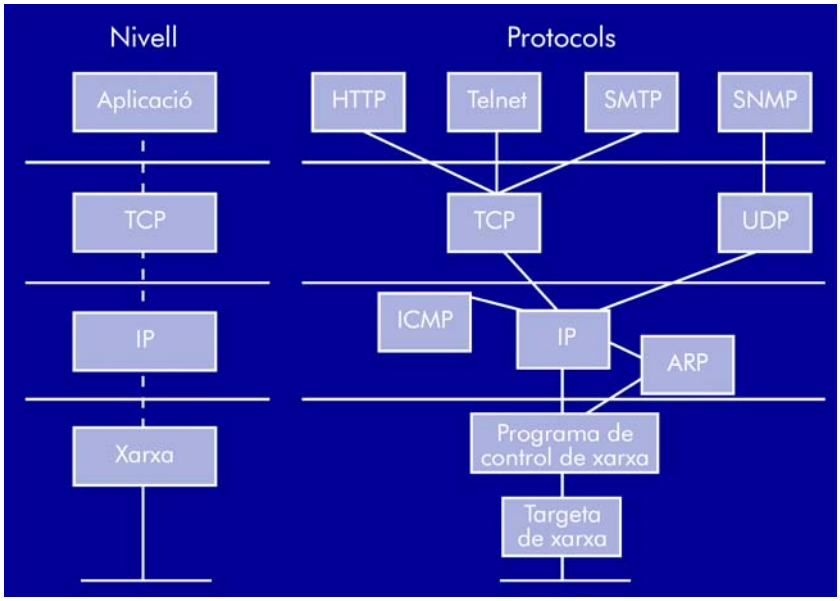
Internet se situa "damunt" de les xarxes locals ja existents. No redefineix el concepte ni proposa protocols de xarxa local nous.

ANOTACIONS

7.1. Protocols d'Internet

En els nivells intermedis hi ha altres protocols complementaris, a més de TCP i IP. La figura següent seria un mapa bastant complet dels protocols que s'usen a Internet:

Figura 23.



Com ja hem comentat, el concepte *nivell* no existeix a Internet. Aquest concepte s'utilitza en altres models de xarxa, com l'OSI. No obstant això, com que és un concepte útil, l'utilitzarem per a plantejar l'estudi dels diferents protocols de la manera següent:

a) En primer lloc, descriurem l'IP i els protocols que hi col·laboren: ARP i ICMP.

b) Després, estudiarem certs aspectes bàsics del nivell de xarxa per a les diferents possibilitats de connexió a Internet actuals: les LAN Ethernet i els accessos per línia punt a punt amb el protocol PPP o amb ADSL.

Deixarem per a unitats posteriors l'estudi del nivell de transport i del nivell d'aplicació.

ANOTACIONS

7.2. Encapsulació

En qualsevol arquitectura de nivells (sigui més o menys formal) en què hi hagi una comunicació vertical dins de cada màquina, les dades que es generen en el nivell superior (aplicació) travessen la resta de nivells per a "sortir" de la màquina pel nivell físic.

Cada un d'aquests protocols funciona amb unes estructures fonamentals que genèricament es coneixen com **PDU** (*protocol data units*). Però, en cada nivell s'utilitzen noms diferents per a denominar el que, de fet, té funcions equivalents. En el conjunt de protocols Internet tenim les PDU següents:

- Les PDU Ethernet o PPP es denominen *trames*.
- Les PDU del nivell d'interconnexió (IP o ARP) se solen denominar *paquets*, encara que les PDU ICMP se solen denominar *missatges*, segurament perquè viatgen en paquets IP.
- En el nivell de transport, es parla de *segments* en TCP, i de *datagrames* en UDP.
- En nivells superiors que utilitzen UDP, generalment s'utilitza la paraula *PDU* (SNMP-PDU, per exemple). En el cas del TCP, el servei que proporciona a les aplicacions és el flux de bytes sense estructura (*byte stream*). Per tant, el concepte *PDU* deixa de tenir sentit en el nivell superior a TCP.

El resultat de les diferents encapsulacions en cada nivell és que quan el nivell superior decideix transmetre certa informació, es provoca una cascada de PDU que descendeix fins al nivell inferior, que finalment és el que transmet físicament els bits que en resulten.

Nota

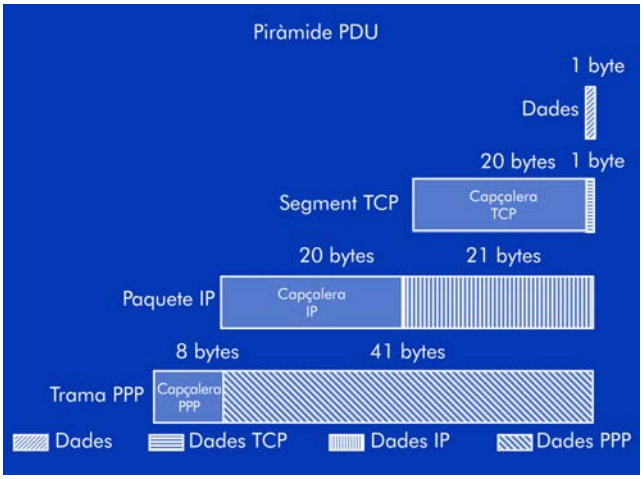
Per a calcular la longitud de trama necessària per a transmetre un byte d'informació, veurem, a tall d'exemple, l'emulació de terminal que efectua el programa `telnet` i el protocol que porta el mateix nom.

ANOTACIONS

Imaginem que premem una tecla sobre la finestra del programa `telnet`. Aquesta acció arriba en forma d'un byte únic al nivell TCP, que encapsula aquest byte en un segment TCP. Aquest segment tindrà una capçalera de 20 bytes i un contingut que serà el byte corresponent a la tecla premuda. Aquests 21 bytes es transportaran dins d'un paquet IP que, generalment, formarà un paquet amb 20 bytes de capçalera més el contingut ja esmentat de 21 bytes. Aquest paquet de 41 bytes anirà, al seu torn, dins d'una trama que el transportarà pel seu suport físic de transmissió. Si el suport de transmissió és una línia telefònica amb un mòdem i utilitzem PPP, pot ser que hi afegim 8 bytes més.

De tot això resulta una trama de 49 bytes de longitud (8 + 20 + 20 + 1) per a transmetre'n un de sol. La figura següent mostra aquesta estructura:

Figura 24.



ANOTACIONS

8. L'IP (*Internet protocol*)

IP i TCP són un parell de protocols ben compenetrats. L'IP és un protocol d'interconnexió de xarxa orientat a datagrama. Per tant, no disposa del concepte de circuit virtual, de manera que no és capaç de recuperar trames perdudes, ni de garantir que les trames es lliuraran en l'ordre correcte –atès que els paquets poden seguir camins diferents i, per tant, patir retards diferents–, ni que el ritme de recepció sigui l'adequat perquè el receptor processi convenientment les dades.

L'IP és del tipus *best effort* (podríem traduir-lo com 'amb la millor intenció', o 'qui fa el que pot no està obligat a més'). Evidentment, quan utilitzem una xarxa, no sempre ens podem conformar d'aconseguir que la informació arribi si la xarxa pot. Aquí intervé el TCP, que és responsable d'aconseguir que la informació arribi en les condicions de fiabilitat desitjades.

Nota

Hi ha detractors acèrrims i també defensors incondicionals de la filosofia *best effort*, aplicada a les xarxes i, segurament, tots dos grups tenen una part de raó. La xarxa X.25 és un exemple gairebé oposat a aquesta filosofia que proporciona als seus usuaris uns nivells de fiabilitat i flexibilitat raonablement elevats, gràcies a la utilització de la commutació de paquets amb circuits virtuals.

De fet, la història dóna la raó als defensors de la filosofia *best effort*, ja que l'X.25 actualment es considera una tecnologia gairebé obsoleta, mentre que l'IP està de moda. Les raons s'han de buscar, com sempre, en el cost. La filosofia IP permet implementar xarxes amb

Lectura complementària

Una bona descripció dels algorismes d'encaminament usats a Internet es pot trobar a:

W.R. Stevens (1994). *TCP/IP Illustrated. The protocols* (vol. 1). Wilmintong: Addison-Wesley.

ANOTACIONS

un cost mínim: penseu que el TCP només necessita cal implementar-la als terminals de la xarxa, i no als nodes de commutació. Per tant, els nodes de commutació IP són molt més simples que els d'X.25.

8.1. Adreces IP

Les adreces IP són úniques per a cada màquina. Per a ser precisos, cada adreça és única per a cada una de les interfícies de xarxa IP de cada màquina. Si una màquina disposa de més d'una interfície de xarxa, necessitarà una adreça IP per a cada una.

Les adreces IP tenen una longitud de 32 bits (4 bytes).

Nota

L'únic objectiu d'aquesta notació és la llegibilitat humana. No es pot perdre mai de vista que una adreça IP són 32 bits.

Per a representar una adreça, se sol escriure els 4 bytes en decimal i separats per punts. Per exemple:

212.45.10.89

La numeració en IP segueix una filosofia jeràrquica. Cada adreça està formada per dues parts: l'una correspon a la xarxa on és l'estació i l'altra, a la mateixa estació.

Per a aconseguir que no hi hagi cap adreça igual, Internet disposa d'una organització denominada *Internet Network Information Center* o *InterNIC* que es dedica a aquesta tasca. Actualment, aquesta entitat delega la responsabilitat de l'assignació d'adreces a entitats regionals. Les adreces s'assignen per grups o xarxes, no individualment.

Els tipus de xarxes que tenen cabuda a Internet es distingeixen per la quantitat d'estacions que poden suportar, i són les següents:

1) **Les xarxes de classe A** reserven el primer byte com a identificador de xarxa i els tres restants com a identificadors d'estació. El primer bit del primer byte val 0, per tant, a Internet només hi pot haver 128 xarxes de classe A (amb 2^{24} estacions cada una com a màxim). Fa molt temps que ja no en queda cap per a assignar.

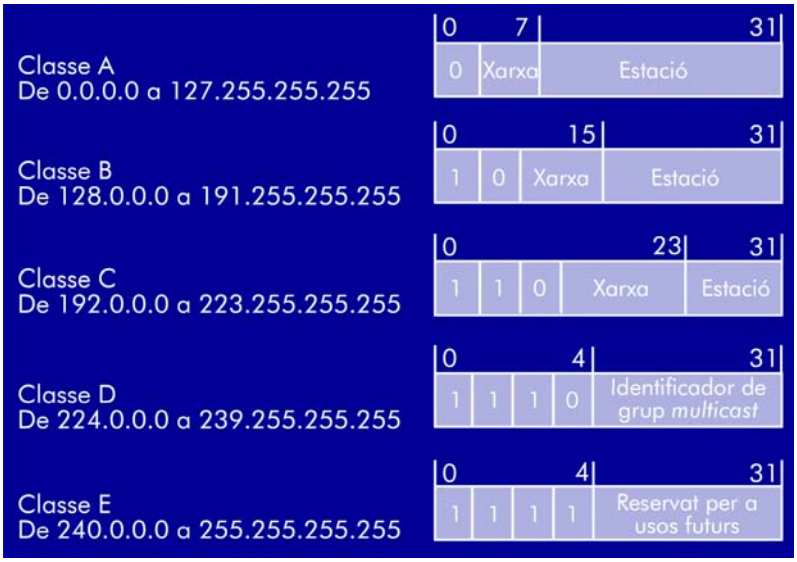
ANOTACIONS

2) **Les xarxes de classe B** tenen 16 bits per a cada camp; els dos primers bytes de l'identificador de xarxa valen 1 0, per tant, n'hi ha 16.384 (2^{14}) xarxes de, a tot estirar, 65.536 estacions. De classe B, no en queda cap per a assignar.

3) **Les xarxes de classe C** reserven 24 bits per a l'identificador de xarxa (amb els tres primers bits 1 1 0) i els 8 restants són per a l'identificador d'estació.

Una vegada que es coneix una adreça, és fàcil saber si correspon a una xarxa de classe A, B o C, com es pot veure en la figura següent:

Figura 25.



La classe A està pensada per a grans empreses o corporacions, amb molts terminals per a identificar; la classe B, per a corporacions mitjanes; la classe C, per a entorns molt més petits; la classe D està destinada al trànsit multicast IP, i la classe E, de moment, no té cap ús concret. L'adreça comentada anteriorment (212.45.10.89) és de classe C.

Nota

Classless interDomain Routing (CIDR)

L'espai d'adreces s'esgota a marxes forçades i, si bé hi ha tècniques relacionades amb la seguretat que ajuden a mitigar aquest inconvenient (com el NAT, *network address translation*), el fet d'utilitzar el concepte de "classe" ha agreujat el problema: algunes

ANOTACIONS

corporacions que han reservat una xarxa classe A o B i només n'han aprofitat una petita part, han deixat sense un bon tros de l'espai d'adreces la resta d'Internet.

Avui dia, en lloc de proporcionar xarxes classe C (de les altres ja no en queden) el que es fa és donar grups més grans (tècnicament, es podrien donar més petits, però no es fa) limitats per màscares intermèdies entre la classe B i la C. Per exemple, si algú vol una xarxa d'un miler d'adreces necessita quatre xarxes de classe C. En lloc de proporcionar-li aquestes quatre xarxes independents, se li dóna una màscara de 22 bits: en queden 10 per a adreces de terminal, la qual cosa permet 1.024 terminals.

8.1.1. Màscares de xarxa

Quan un administrador de sistemes rep l'encàrrec de gestionar un conjunt d'adreces, és possible que necessiti configurar internament diferents LAN amb aquest conjunt. Per això, el mecanisme per a distingir diferents xarxes (LAN) entre si no es pot basar exclusivament en els bits identificadors de classe que hem comentat anteriorment.

La màscara de xarxa és el mecanisme que ens permetrà aconseguir més flexibilitat. Per mitjà d'una màscara de 32 bits, definirem els bits que identifiquin la xarxa (bits en 1) i els que identifiquin l'estació (bits en 0). Per norma general, els bits 1 i els 0 són consecutius, però no necessàriament.

A continuació, definim de nou el concepte *identificador de xarxa*, adaptant-lo a la màscara: l'**identificador de xarxa** és la porció d'adreça IP que encaixa amb els bits 1 de la màscara.

El concepte *màscara* és capital per a comprendre el funcionament de les xarxes IP, permet a una estació decidir si la destinació a la qual ha de transmetre un paquet es troba dins de la mateixa xarxa d'àrea local o si, al contrari, es troba en una LAN remota i, per tant, ha de delegar-ne la transmissió a algun equip de la seva mateixa LAN (l'encaminador) perquè s'encarregui de fer arribar el paquet a la seva destinació.

ANOTACIONS

Totes les estacions d'una mateixa xarxa d'àrea local han d'utilitzar el mateix identificador de xarxa i totes les estacions han de tenir la mateixa màscara.

Nota

Si tenim dues estacions amb les adreces 147.83.153.100 i 147.83.153.200, podem deduir que estan interconnectades directament (per una LAN) si la màscara de la seva xarxa és 255.255.255.0. Deduiríem que no estan connectades amb la mateixa LAN si la màscara fos, per exemple, 255.255.255.128.

Nota

Una notació alternativa és proporcionar el nombre de bits 1 de la màscara. Així doncs, la màscara 255.255.255.0 és una màscara de 24 bits i la 255.255.255.128 és una màscara de 25 bits.

A vegades, podem veure una adreça amb l'afegit de la màscara; per exemple:

147.83.153.100/24.

Aquesta notació només és útil per a màscares amb 1 consecutius.

Quan la màscara no coincideix exactament amb l'estructura de classes definida, llavors es parla de *subnetting*, perquè creem subxarxes dins d'una xarxa.

Exemple

La màscara 212.45.10.0/ 27. Permet 6 subxarxes diferents dins de la xarxa de classe C 212.45.10.0.

8.1.2. Adreces de propòsit especial

Hi ha diferents adreces especials. Nosaltres exposarem a continuació només les més importants:

- **Adreça de xarxa**. Les adreces de xarxa s'expressen amb l'adreça que tindria qualsevol estació seva i amb tots els bits de l'identificador

ANOTACIONS

d'estació a zero. Per exemple, la xarxa en què es troba l'estació 147.83.153.100/24 és la 147.83.153.0/24 i la xarxa en què es troba l'estació 147.83.153.200/25 és la 147.83.153.128/25.

- **Adreça 0.0.0.0**. Aquesta adreça assenyala el mateix ordinador que l'envia. Té dues funcions bàsiques:

– Aparèixer com a adreça origen en paquets IP generats per estacions sense direcció IP assignada. Normalment només apareix mentre l'estació intenta esbrinar la seva adreça mitjançant protocols com ara RARP (*reverse address resolution protocol*), BOOTP (*bootstrap protocol*) o DHCP (*dynamic host central configuration protocol*).

– Servir al programari de gestió de direccionament per a indicar la ruta per defecte.

Exemple

L'adreça 127.03.87 és equivalent a l'adreça 127.0.0.1.

- **Adreça 127.0.0.1** (***loopback***). Aquesta adreça no és vàlida per als paquets IP. El programari de xarxa la utilitza per a transmetre paquets a la màquina local (de fet, els paquets no són enviats, sinó que són lliurats a la destinació pel mateix sistema operatiu). En realitat, els tres bytes de l'identificador d'estació són irrellevants. Aquesta adreça només té interès per a programar aplicacions; els sistemes de xarxa no veuran mai que cap paquet viatgi per la xarxa amb aquesta adreça com a origen o destinació.

- **Adreça 255.255.255.255** (***broadcast***). Aquesta adreça només és vàlida com a adreça de destinació d'un paquet. S'utilitza per a transmetre paquets a totes les estacions localitzades dins de la mateixa LAN que la màquina d'origen. Hi ha una versió equivalent, que és el *broadcast* dirigit. En aquest segon cas, el paquet és rebut per totes les màquines d'una LAN especificada per l'identificador de xarxa. L'identificador d'estació ha de ser tot 1.

  Per tant, per a enviar una difusió (*broadcast*) a la xarxa 147.83.153.0 amb la màscara 255.255.255.0 (o 147.83.153.0/24) podem utilitzar l'adreça 255.255.255.255 si som dins de la xarxa 147.83.153.0, o bé la 147.83.153.255 si som en una estació remota. El primer cas, l'anomenarem *difusió local* (*broadcast local*), i el segon, difusió remota (*broadcast remot*).

ANOTACIONS

- Totes les adreces d'aquests rangs:

– 10.0.0.0/8
– De la 172.16.0.0/16 a la 172.31.0.0/16.
– De la 192.168.0.0/24 a la 192.168.255.0/24.

Aquestes adreces, que corresponen respectivament a xarxes de classe A, B i C, no són assignades per Internet, i mai no ho seran. S'utilitzen en xarxes que treballen amb els protocols TCP/IP però no està previst que es connectin directament a Internet i, en cas que s'hi connectessin, estarien parcialment ocultes per *proxies* o tallafocs (*firewall*), que s'encarreguen de reduir la seva adreça a una altra que estigui en els rangs d'adreces públiques.

Per tant, a les xarxes de classe A, B i C no tenim 2^8, 2^{16} o 2^{24} estacions possibles, respectivament, sinó $2^8 - 2$, $2^{16} - 2$ i $2^{24} - 2$. Les adreces que tenen tots els bits corresponents a l'estació a 0 i les que els tenen tots a 1 no són adreces vàlides per a estacions.

Activitats

- Indiqueu a quina classe pertanyen, quina seria l'adreça de xarxa i l'adreça de difusió remota de cada una de les adreces IP següents:

– 169.5.10.10 / 16
– 124.127.122.123 / 8
– 199.134.167.175 / 27
– 201.201.202.202 / 24
– 129.11.189.15 / 20

- Una empresa vol muntar una xarxa d'ordinadors en un entorn TCP/IP. Per a això, ha decidit usar adreces reservades per a intranets. Concretament la xarxa 192.168.206.0, però només necessita espai d'adreces per a deu estacions, i es plantegen fer una segmentació de la xarxa.

– Quina seria la màscara més restrictiva per a aquest escenari?

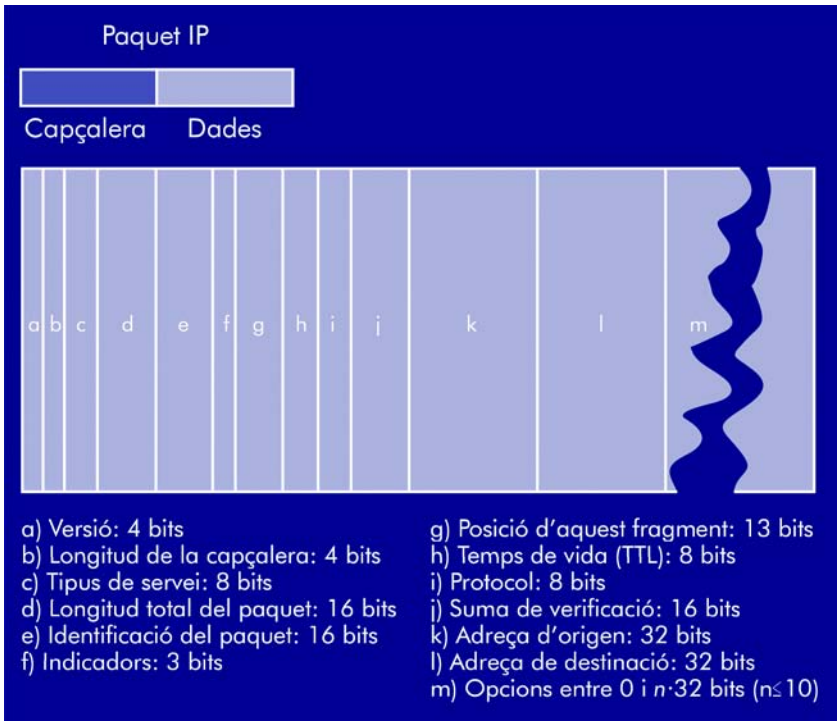
– Quin seria el rang de subxarxes possibles?

ANOTACIONS

– Suposant que l'adreça IP d'una estació sigui 192.168.206.100, quines serien les adreces de la resta d'estacions?

8.2. El format del paquet IP

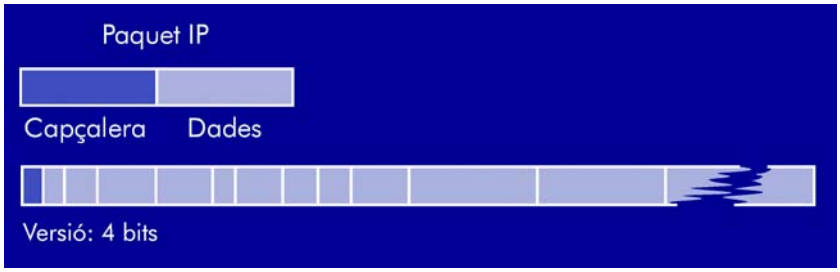
Figura 26.



A continuació descriurem els camps que componen el paquet IP:

- **Versió**: sempre val quatre (0100), per als paquets de la versió actual (IPv4).

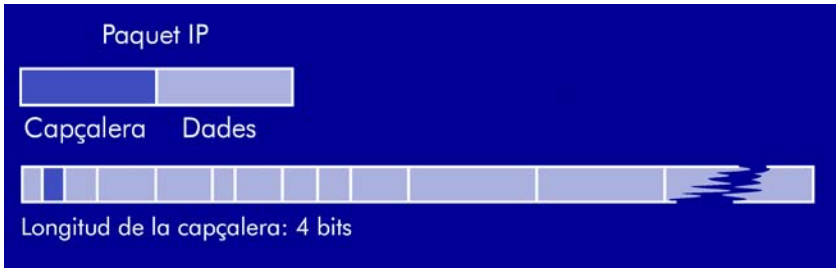
Figura 27.



ANOTACIONS

- **Longitud de la capçalera**: dóna la longitud de la capçalera en paraules de 32 bits (4 bytes). Per tant, el nombre de bytes de la capçalera ha de ser múltiple de 4. Així mateix, limita la longitud de la capçalera a 60 bytes (15 · 4), ja que 15 és el màxim que es pot expressar amb quatre dígits binaris. Si no hi ha camp d'opcions, la capçalera té 20 bytes; per tant, el camp en aquest cas valdria 5 (5 · 4 bytes = 20 bytes).

Figura 28.



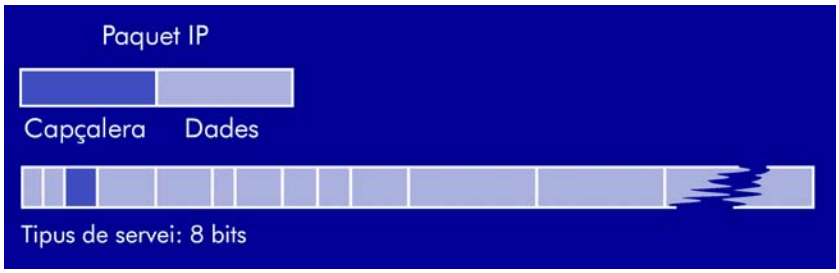
- **Tipus de servei**: aquest camp es troba dividit en diversos subcamps. Permet demanar un tracte especial per al paquet i rarament s'implementa.

Nota

El camp Tipus de servei permet definir dues variables:

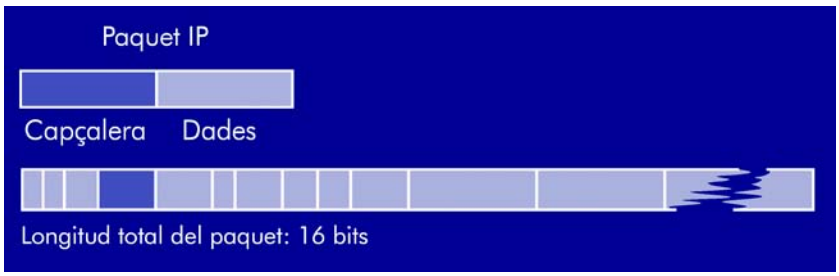
- El nivell de prioritat (de l'1 al 8).
- El tipus de qualitat aplicable al contingut (retard baix, velocitat alta, fiabilitat alta, cost baix, etc.).

Figura 29.



- **Longitud total del paquet**: dóna la longitud total del paquet (capçalera inclosa) en bytes. Com que aquest camp és de 16 bits, un paquet IP no pot tenir més de 65.535 bytes ($2^{16} - 1$).

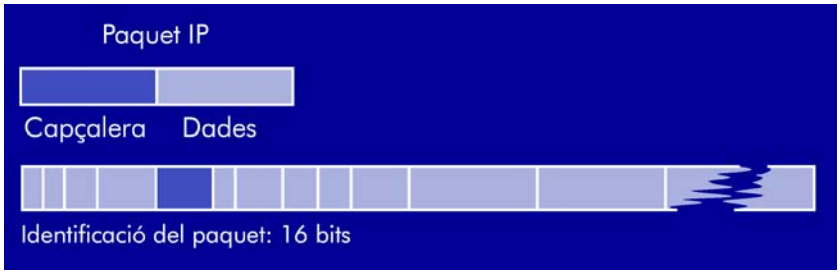
Figura 30.



ANOTACIONS

- **Identificació del paquet**: conté un identificador que és diferent per a cada paquet que genera l'estació.

Figura 31.

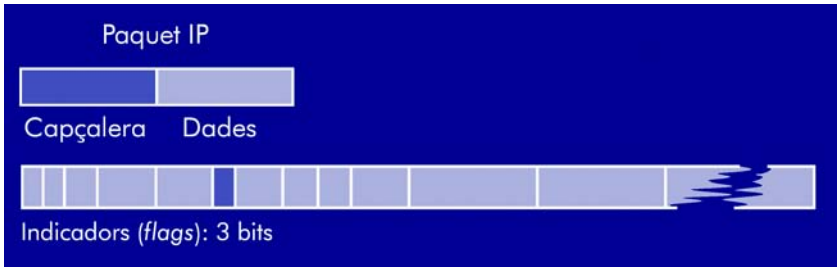


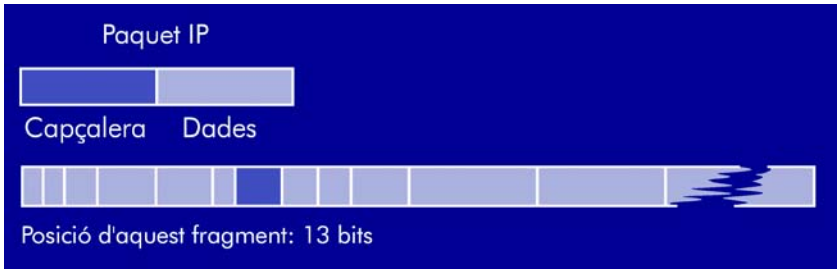
- **Indicadors** (*flags*) **i Posició d'aquest fragment**: aquests camps permeten gestionar la fragmentació de paquets.

Nota

Podeu veure la fragmentació de paquets en l'apartat 8.2.1 d'aquesta unitat.

Figura 32.

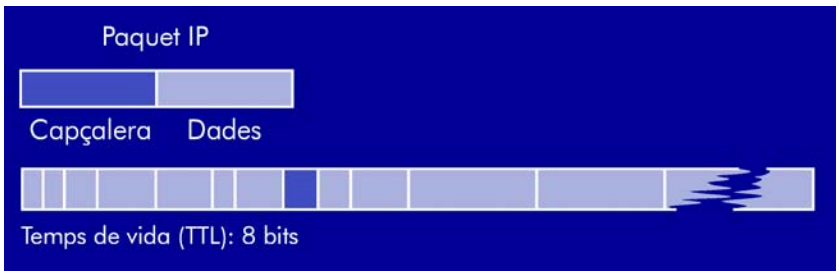




- **Temps de vida o TTL** (*time to live*): indica el nombre màxim d'encaminadors que pot creuar el paquet. S'utilitza per a evitar que un paquet pugui quedar fent voltes indefinidament dins de la xarxa en cas que hi hagi algun problema en lliurar-lo. Cada encaminador disminueix el camp en un; quan un d'aquests encaminadors detecta un paquet amb TTL = 1, l'elimina i envia a qui l'ha emès un missatge d'error per mitjà d'un missatge ICMP.

ANOTACIONS

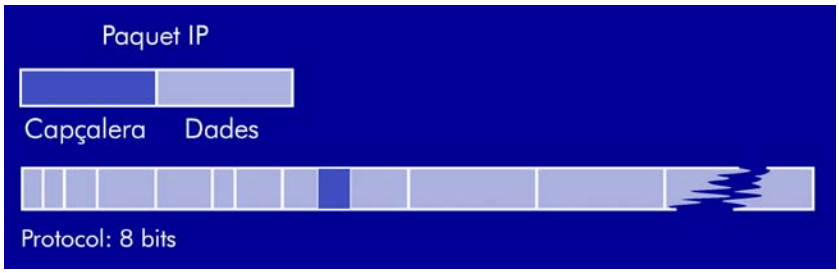
Figura 33.



- **Protocol**: identifica el tipus de protocol que transporta el paquet. Els valors més comuns dels diferents tipus de protocols són els següents:

TCP	6
UDP	17
ICMP	1

Figura 34.



- Suma de verificació (*checksum*): realitza el control d'errors a la capçalera. El camp de dades no queda protegit per cap *checksum*; és responsabilitat dels usuaris de l'IP (TCP, UDP, etc.) el control dels possibles errors en el seu contingut.

Figura 35.

Nota

Podeu veure una implementació en llenguatge C de l'algorisme *checksum* en l'annex 1.

ANOTACIONS

Nota

La suma de verificació consisteix bàsicament a sumar els bytes de la capçalera agrupats de dos en dos (si el resultat necessita més de 16 bits, se sumen els bits sobrants al mateix resultat).

Aquest algorisme, si bé és fàcil i ràpid de calcular, no es caracteritza per tenir unes grans qualitats per a la detecció d'errors. Això, sumat al fet que el contingut de l'IP no té suma de verificació i a altres factors (com ara que molts sistemes no calculen la suma de verificació dels paquets UDP), demostra que en el món d'Internet no hi ha un interès especial per la detecció d'errors. Amb freqüència, aquest ha estat un argument en què s'han basat els detractors del protocol IP per a atacar-lo.

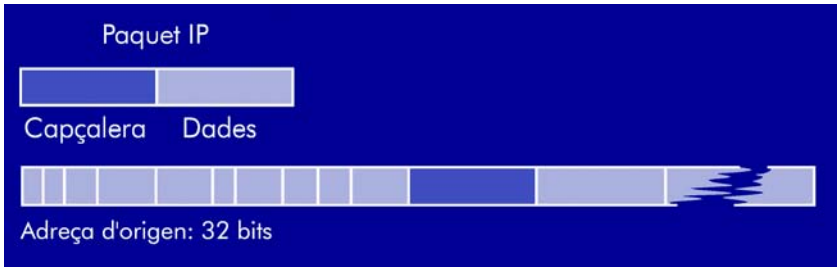
La detecció d'errors dins del camp d'informació és responsabilitat de qui entra la informació. Els usuaris més habituals de l'IP són els protocols TCP, UDP i ICMP, i tots protegeixen la informació amb un camp addicional de suma de verificació.

Activitat

Busqueu errors possibles que passarien inadvertits a l'algorisme *checksum*.

- **Adreça d'origen IP**: identifica la màquina que ha generat el paquet.

Figura 36.



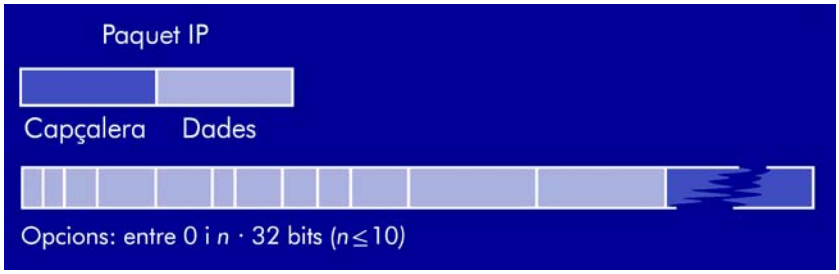
ANOTACIONS

- **Adreça de destinació IP**: identifica la màquina a la qual va destinat el paquet.

Figura 37.

- **Opcions**: hi ha diferents serveis assignats a aquest camp, però en general no s'utilitzen. En comentarem alguns quan parlem de l'ICMP. De tota manera, la longitud total d'aquest camp no pot excedir els 40 bytes ($15 \cdot 4 - 20$).

Figura 38.



8.2.1. Fragmentació

Els paquets IP van sempre inserits en trames de nivell d'enllaç o MAC. Uns exemples que ja hem vist són els protocols PPP i Ethernet.

La majoria de protocols de nivell d'enllaç fixen una longitud màxima de dades que es poden transmetre en una trama. Aquesta longitud es coneix com a MTU (*maximum transfer unit*) i està determinada per diferents factors segons el cas. En el cas d'Ethernet, l'MTU val 1.500 bytes.

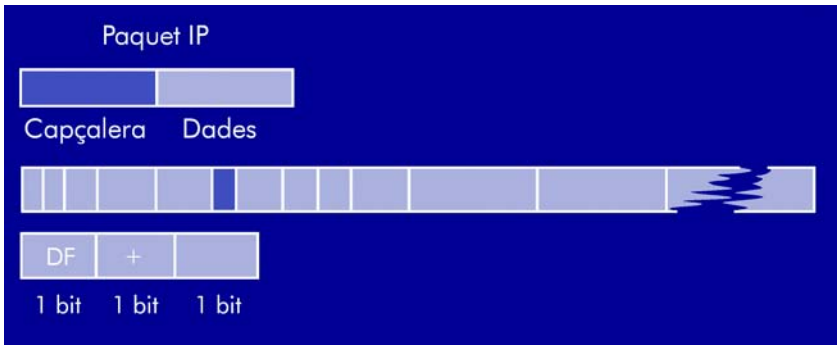
Quan una estació transmet un paquet IP, coneix l'MTU de la seva interfície de xarxa, i està restringida a transmetre paquets IP la longitud dels quals sigui inferior o igual que aquesta MTU. Si el paquet passa per un encaminador connectat a una xarxa amb una MTU inferior

ANOTACIONS

a la mida del paquet, caldrà fragmentar-lo. Els fragments resultants de l'operació són paquets normals, però amb les característiques següents:

- Tots els fragments que provenen d'un paquet fragmentat tenen el mateix identificador de paquet a les capçaleres IP respectives.
- Tots els fragments indiquen, amb el camp Posició d'aquest fragment, a quin byte correspon el primer byte de dades del fragment dins del paquet original. El primer fragment té un zero en aquest camp.
- La part de dades de cada fragment conté un nombre de bytes múltiple de 8. L'últim fragment no té per què complir aquest requisit.
- El bit + del camp Indicadors és 1 en tots els fragments excepte l'últim, que per això se sol denominar *n'hi ha més*.
- La resta dels camps de la capçalera es copia íntegrament del paquet original als fragments que resulten de la fragmentació, excepte l'indicador de longitud del paquet i el *checksum*, que s'han de calcular de nou.

Figura 39.



En principi, podríem suposar que, quan els fragments tornen a una xarxa amb una MTU suficient, l'encaminador que els rep els reunifica. Però, no es fa així, en primer lloc perquè la llei no escrita d'Internet segons la qual l'encaminador ha de dur a terme el mínim treball necessari no es compliria. En segon lloc (i més important), perquè ningú no garanteix que tots els fragments segueixin el mateix camí. L'única estació capaç de recompondre els fragments és la de destinació.

ANOTACIONS

L'estació de destinació, quan rep un fragment, reserva suficient memòria per a allotjar el paquet IP més gran possible (65.535 bytes), ja que no té manera de saber quina és la mida del paquet original. A partir d'aquest moment, posa en marxa un temporitzador i comença a recollir els fragments segons el seu identificador. Quan s'han rebut tots els fragments, es lliura el paquet a l'aplicació (protocol) corresponent. En cas que el temporitzador salti, es descarten tots els fragments arribats fins a aquell moment. El nivell superior (TCP) és el responsable de demanar una retransmissió quan convingui, ja que l'IP no disposa de cap mecanisme per a demanar-la.

Nota

Aquest temporitzador permet detectar la pèrdua d'un fragment pel camí i, per tant, la pèrdua de la memòria reservada.

Nota

Dins del camp Indicadors hi ha dos bits més que no hem comentat: un no s'utilitza i l'altre, el DF (*don't fragment*), especifica que el paquet no s'ha de fragmentar. Si fos necessari fragmentar-lo, el paquet seria descartat i s'enviaria un missatge ICMP indicant l'error a l'estació originadora del paquet. Usualment, s'activa el bit DF només quan la fragmentació no és desitjable com, per exemple, quan enviem paquets a estacions amb la pila TCP/IP implementada a ROM, ja que llavors s'implementa només una mínima part de la funcionalitat de TCP/IP.

Activitat

S'ha de transmetre un datagrama de 4.480 bytes i necessita ser fragmentat perquè passarà per una xarxa Ethernet que admet un màxim de 1.500 bytes de dades. Indiqueu els valors del camp Longitud del paquet, del camp Posició d'aquest fragment i del bit + (MF), per a cada un dels paquets IP que es generin.

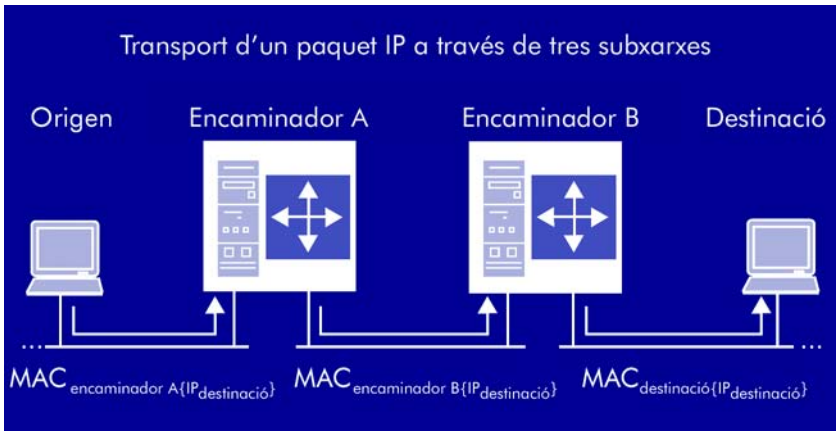
8.3. Encaminament i encaminadors

Les atribucions principals dels encaminadors són interconnectar dues o més subxarxes IP i encarregar-se d'encaminar el trànsit destinat a estacions remotes.

ANOTACIONS

Cada encaminador de la xarxa Internet ha de ser capaç de decidir (en una fracció de segon) on ha de dirigir un paquet basant-se exclusivament en l'adreça de destinació, i en el coneixement que té de la xarxa i de la seva posició (la de l'encaminador) dins d'Internet. De fet, cada encaminador no decideix la ruta sencera del paquet, sinó només el tros de ruta en què participa: el salt (*hop*) següent. Entre l'origen i la destinació tenim un nombre variable de salts, i en cada un d'ells participen un parell d'encaminadors (l'emissor i el receptor del paquet) a excepció del primer i l'últim, en el qual participen les estacions emissora i receptora, respectivament. Les connexions entre encaminadors, o bé entre encaminador i estació terminal, les componen LAN o enllaços punt a punt.

Figura 40.



En la figura anterior, podem veure un paquet IP que creua tres xarxes d'àrea local. En cada una el paquet va encapsulat dins d'una trama d'un tipus adequat a la subxarxa que creua (en la figura podrien ser tres LAN Ethernet). Les adreces d'origen i de destinació del nivell LAN en cada tram són diferents. En canvi, l'adreça d'origen i la de destinació de cada paquet IP no varien. La situació planteja dos problemes i, per a resoldre'ls, és necessari efectuar els passos següents:

- Esbrinar quina és la correspondència entre adreces IP i MAC (només en els casos en els quals l'enllaç entre els encaminadors sigui una LAN). Això ens permetrà enviar els paquets IP a un encaminador malgrat que el paquet IP no tingui la seva adreça. El mapat IP-MAC s'efectua de manera automàtica per mitjà de l'ARP.
- Decidir en cada moment quin és l'encaminador següent (el salt següent).

Nota

Podeu veure l'ARP en la unitat 9.

APLICACIONS

8.3.1. La taula d'encaminament

L'encaminament es duu a terme a partir de **taules d'encaminament** que disposen d'informació limitada, però suficient per a permetre la interconnexió de totes les subxarxes que componen Internet. Cada equip connectat a una xarxa IP necessita una taula d'encaminament.

Nota

Aquestes taules s'introdueixen dins de l'encaminador per mitjà d'un terminal que s'hi connecta directament (consola). Així mateix, hi ha protocols que permeten que les taules s'actualitzin automàticament quan es detecten canvis de topologia. Els més utilitzats són el RIP (*routing information protocol*) i l'OSPF (*open shortest path first*).

L'encaminador decideix la ruta dels paquets consultant la seva taula d'encaminament. Les estacions terminals també necessiten una taula d'encaminament, encara que només sigui per a poder descobrir si es comuniquen amb una estació local de la LAN (i, per tant, s'hi poden comunicar directament) o si el paquet IP ha d'anar a una estació remota (connectada a una altra LAN) i, per tant, l'han de deixar a les mans de l'encaminador.

Taula d'encaminament d'una estació amb una interfície única

Comencem estudiant la taula d'encaminament d'una estació connectada a una LAN (adreça 147.83.153.103/24) amb una única targeta Ethernet. Per norma general, una taula d'encaminament disposa també d'informació sobre les "qualitats" de cada una de les rutes descrites, que no apareixen en l'exemple:

Taula 2.

Taula d'encaminament de l'estació				
	Adreça	Màscara	Encaminador	Interfície
1	147.83.153.103	255.255.255.255	127.0.0.1	*Loopback*
2	127.0.0.0	255.0.0.0	127.0.0.1	*Loopback*
3	147.83.153.0	255.255.255.0	147.83.153.103	ether0
4	255.255.255.255	255.255.255.255	147.83.153.103	ether0
5	0.0.0.0	0.0.0.0	147.83.153.5	ether0

ANOTACIONS

Nota

L'ordre `route` permet consultar i modificar la taula d'encaminament d'una estació. Aquesta ordre és present, amb aquest nom, en la majoria de sistemes operatius, inclosos els GNU/Linux (i, en general, tots els sistemes Unix).

Aquesta és una taula típica que podem trobar en una estació connectada a Internet. Les files de la taula o regles es consulten no en l'ordre en què la comanda `route` ens les presenta, sinó en ordre de màscara decreixent (en primer lloc, les entrades amb 32 bits, etc.). De les entrades que formen la taula és necessari assenyalar el següent:

- La **primera entrada** i la **segona** permeten transmetre paquets IP a les adreces 147.83.153.103 i a totes les adreces que comencin per 127 (fixeu-vos en la màscara de les dues entrades). En tots dos casos els paquets s'envien a la interfície virtual *loopback* (la interfície amb la qual es coneix l'ordinador local des del mateix ordinador local). Cap dels paquets que s'encamini amb alguna d'aquestes dues regles no sortirà a la xarxa: serà curtcircuitat directament pel sistema operatiu i es lliurarà al procés de destinació sense que mai arribi a cap targeta de comunicacions.

  Nota

  Una interfície és una targeta de connexió física a la xarxa. Serien exemples d'interfície una targeta Ethernet (per exemple, l'Ether0) o un port sèrie al qual connectem un mòdem. *Loopback*, com ja s'ha comentat, és un cas especial d'interfície.

  Per a saber quines interfícies hi ha disponibles en un sistema GNU/Linux, es disposa de l'ordre `ifconfig`.

- La **tercera entrada** serà adoptada per tots els paquets destinats a la xarxa local. El camp Encaminador és el mateix que l'adreça local, cosa que significa que no se'n delegarà la transmissió a cap l'encaminador.

- La **quarta entrada** té una importància relativa. Ens indica que les difusions IP es restringiran a la xarxa local.

- La **cinquena entrada** (0.0.0.0/0) permet a l'estació comunicar-se amb estacions remotes. Noteu que la màscara no té cap bit a 1. Aquesta és la **ruta per defecte**. L'encaminador establert per a accedir a estacions remotes queda identificat en la taula amb l'adreça 147.83.153.5.

ANOTACIONS

Tota aquesta informació es calcula a partir de tres dades que s'introdueixen en la configuració de xarxa:

1) L'adreça local (en l'exemple: 147.83.153.103).
2) La màscara de la LAN local (en l'exemple: 255.255.255.0).
3) L'adreça de l'encaminador (en l'exemple: 147.83.153.5).

En casos més complexos (més d'una interfície d'accés a Internet, més d'un encaminador a la xarxa local, etc.), la informació s'haurà de modificar amb l'ordre `route`. Així mateix, hi ha altres mecanismes que no requereixen la intervenció humana i que permeten descobrir encaminadors que no s'han configurat prèviament per mitjà del protocol ICMP.

Activitat

Observeu la taula d'encaminament del vostre ordinador per mitjà de l'ordre `route`. Consulteu l'ajuda disponible en el sistema operatiu (en GNU/Linux feu `man route`).

En un encaminador, les taules d'encaminament tenen més entrades que les d'una estació, to i que el funcionament és el mateix. Observeu-ne una que connecta la xarxa 147.83.153.0/24 amb l'exterior per mitjà de la xarxa 147.83.30.0/24:

Taula 3.

Taula d'encaminament de l'encaminador				
	Adreça	Màscara	Encaminador	Interfície
1	147.83.153.5	255.255.255.255	127.0.0.1	*Loopback*
2	147.83.30.2	255.255.255.255	127.0.0.1	*Loopback*
3	127.0.0.0	255.0.0.0	127.0.0.1	*Loopback*
4	147.83.153.0	255.255.255.0	147.83.153.5	ether1
5	147.83.30.0	255.255.255.0	147.83.30.2	ether0
6	255.255.255.255	255.255.255.255	147.83.153.5	ether1
7	0.0.0.0	0.0.0.0	147.83.30.1	ether0

ANOTACIONS

Pràcticament no observem res de nou. Simplement s'han duplicat les entrades específiques d'interfície. L'encaminador també té una entrada per defecte. En aquest cas, tot el trànsit no destinat a les dues xarxes a què està directament connectat s'encamina cap a un altre encaminador situat a la xarxa 147.83.30.0 (el 147.83.30.1).

Activitat

Penseu si tots els encaminadors d'Internet tenen una ruta per defecte i raoneu-ho. Imagineu què succeeix quan enviem un paquet a una estació d'una xarxa inexistent (per exemple, a l'estació 10.0.1.1).

ANOTACIONS

9. L'ARP (*address resolution protocol*)

En descriure el funcionament dels encaminadors hem vist que necessiten saber l'adreça MAC corresponent a una adreça IP. L'ARP és l'encarregat de dur a terme la resolució automàtica del mapat entre adreces MAC.

Quan efectuem la transmissió d'un paquet entre dues estacions locals d'una mateixa LAN, ho fem indicant a l'aplicació corresponent només l'adreça IP. Per exemple, si des de 147.83.153.103 ens volem connectar a 147.83.153.100, farem el següent:

Nota

`telnet` és una aplicació d'ús comú per a l'accés a servidors remots. La veurem en la unitat 17.

```
$ telnet 147.83.153.100
```

Aplicant la màscara de xarxa a l'adreça IP sol·licitada, l'estació dedueix que es troba a la seva mateixa subxarxa. Per tant, no cal delegar en cap encaminador. Per a poder enviar-li les trames en les quals han d'anar els paquets IP que genera l'aplicació `telnet`, necessita conèixer la seva adreça MAC.

Per a això, emet una petició ARP. Es tracta d'un paquet encapsulat directament sobre una trama Ethernet (amb tipus = 0x0806). Com adreça de destinació, porta l'adreça de difusió (FF:FF:FF:FF:FF:FF) perquè arribi a totes les estacions de la LAN, i com contingut, l'adreça IP per a la qual es vol conèixer l'adreça MAC. L'estació que reconeix el seu IP en la petició ARP respon amb una resposta ARP dirigida a l'origen de la petició amb la seva adreça MAC. De fet, el contingut de la resposta ARP és irrellevant, l'únic important és l'adreça MAC d'origen de la trama.

Nota

No mostrarem el format dels paquets ARP, perquè no ens aportaran gaire més informació. Podeu trobar el format i els usos alternatius de l'ARP a:

D.E. Comer (1995). *Internetworking with TCP/IP. Principles, Protocols, and Architecture* (volum 1). Hertfordshire: Prentice Hall.

El mateix succeeix quan l'estació dedueix que l'IP sol·licitada per alguna aplicació correspon a una estació remota; és a dir, de fora de la seva LAN. En aquest cas, les trames s'han d'enviar a l'encaminador perquè s'encarregui de treure-les de la LAN cap a la seva destinació. Per a això, l'estació origen ha d'esbrinar l'adreça MAC de l'encaminador.

ANOTACIONS

Nota

En rigor, per a cada trama amb paquet IP que s'ha de generar, s'hauria de fer una petició ARP.

És fàcil deduir que, en el funcionament normal d'una estació, les peticions ARP poden ser molt habituals. Per a evitar-ho, cada estació disposa d'una taula de les parelles ja obtingudes, de manera que si una adreça IP es troba en aquesta taula, no cal enviar cap petició ARP. Aquesta taula es denomina *cau ARP*, i té l'aspecte següent:

Taula 4.

Taula cau ARP			
	Adreça IP	Adreça MAC	Interfície
1	147.83.153.103	08:00:00:10:97:00	ether0
2	147.83.153.5	00:0c:aa:00:0f:e0	ether0

Cada fila correspon a un mapat IP-MAC i a la interfície per a la qual s'ha sol·licitat.

Nota

En GNU/Linux, `arp -a` serveix per a mostrar la taula. Podeu consultar les pàgines del manual per a la resta d'opcions.

Aquesta taula s'omple automàticament a mesura que es realitzen peticions noves, però també es pot manipular amb la comanda `arp`, amb la qual es pot consultar, afegir o esborrar entrades.

La taula ARP es denomina *cau* perquè, de fet, actua com una memòria auxiliar que evita la consulta de la informació a la LAN mentre se'n tingui una còpia local. Pot semblar que la consulta ARP no hauria de ser massa freqüent, ja que al cap de cert temps tota la informació es trobaria dins de la memòria. Convé saber que les entrades caduquen al cap d'un període de temps relativament breu (entre un i uns quants minuts, segons el sistema).

Activitat

Imagineu què podria succeir si les entrades de la cau ARP no caduquessin. Us pot servir d'ajuda pensar en la reconfiguració d'adreces.

Alguns usos alternatius d'interès de l'ARP són els següents:

- **ARP innecessari** (*gratuitous-ARP*): s'utilitza quan una estació engega, per saber si hi ha alguna altra estació que utilitzi la seva mateixa adreça IP. Per mitjà d'una petició ARP, pot preguntar qui

ANOTACIONS

té la seva adreça IP (un conflicte d'aquest tipus podria deixar les dues estacions fora de combat).

- **ARP subsidiari** (*proxy-ARP*): s'utilitza en situacions en què un encaminador divideix una subxarxa sense que les estacions ni l'encaminador que els connecta a Internet modifiquin la seva màscara. Aquesta tècnica, malgrat no ser massa ortodoxa, s'aplica amb freqüència quan l'encaminador d'una xarxa privada es connecta a Internet per mitjà d'una xarxa aliena (un proveïdor d'Internet, per exemple).

Activitat

Consulteu la cau ARP d'algun sistema que tingueu a l'abast (Unix o MS) per mitjà d'`arp -a`. Comproveu quines opcions té disponibles.

ANOTACIONS

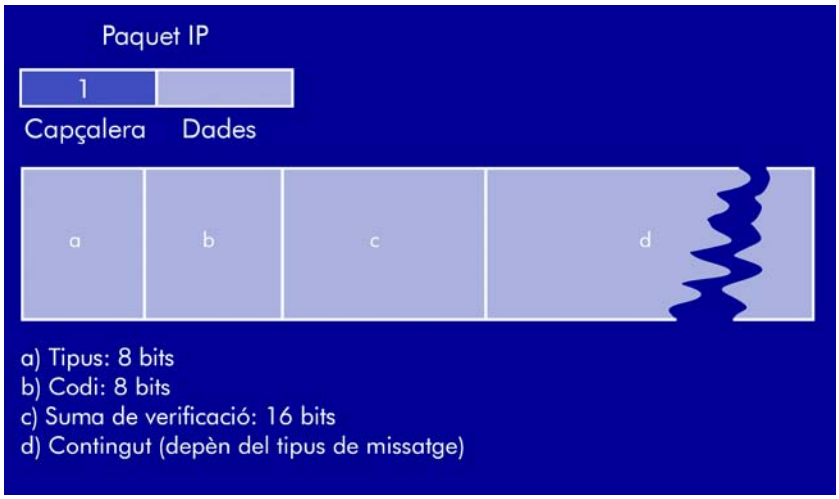
10. L'ICMP (*Internet control message protocol*)

La qüestió de si l'ICMP és un protocol, o si més aviat és una eina que utilitza el protocol IP per a notificar errors, genera molta polèmica. El cert és que l'**ICMP** constitueix el mecanisme bàsic per a gestionar les diferents incidències que poden ocórrer en una xarxa IP.

10.1. Missatges ICMP

Els missatges ICMP viatgen dins de paquets IP (al contrari del que succeïa amb els paquets ARP), en el camp de dades amb el camp Protocol igual a 1. El format del missatge presentat en la figura següent ens facilitarà l'estudi dels diferents usos del paquet ICMP:

Figura 41.



Hi ha tretze *tipus* de missatges ICMP en ús actualment, i al voltant d'una trentena de subtipus identificats amb el camp Codi.

Taula 5.

Tipus	Codi	Descripció	Classe
0	0	Resposta d'eco (*echo reply*)	Petició
8	0	Petició d'eco (*echo request*)	Petició

ANOTACIONS

Tipus	Codi	Descripció	Classe
3	0-15	Destinació inabastable (*unreachable destination*). Els diferents codis permeten definir si el que no es pot assolir és la subxarxa (0, 6, 9, 11), l'estació (1, 7, 10, 12), el protocol (2) o el port (3), i els motius	Error
4	0	Petició de control de flux (*source quench*)	Error
5	0-3	Reencaminament	Error
9	0	Publicació de rutes	Petició
10	0	Petició de rutes	Petició
11	0-1	El temps de vida ha expirat (*time exceeded*)	Error
12	0-1	Capçalera IP incorrecta	Error
13	0	Petició d'hora	Petició
14	0	Resposta d'hora (en mil·lisegons des de la mitjanit)	Petició
17	0	Petició de la màscara de la subxarxa	Petició
18	0	Resposta de la màscara de la subxarxa	Petició

L'última columna ens permet distingir missatges ICMP de notificació d'error de missatges que són part d'una petició (la petició o la resposta). Aquesta distinció és important, ja que els missatges d'error ICMP no poden generar altres missatges d'error ICMP. En particular, no es generen missatges d'error en resposta als paquets o missatges següents:

- Els missatges d'error ICMP.
- Un paquet IP destinat a una adreça de difusió (tant una difusió IP com una difusió MAC).
- Un fragment que no sigui el primer.
- Una adreça d'origen que no identifiqui una única estació. Per exemple, l'adreça d'origen vàlida 0.0.0.0 o l'adreça d'origen no vàlida 255.255.255.255.

En qualsevol de les situacions que acabem de descriure es poden provocar respostes en cascada que podrien afectar greument la xarxa.

Nota

Pedaços

Desgraciadament, no totes les implementacions TCP/IP han tingut en compte les excepcions existents en la gene-

ANOTACIONS

ració de missatges d'error, i alguns usuaris desaprensius han fet explotar aquests problemes per a bloquejar sistemes i xarxes remotes. Així mateix, de tant en tant es descobreixen excepcions noves que poden afectar els nostres sistemes.

Els fabricants de sistemes operatius publiquen regularment pedaços (*patch*) que permeten solucionar els problemes (les vulnerabilitats) descoberts des de la data de publicació de l'última versió.

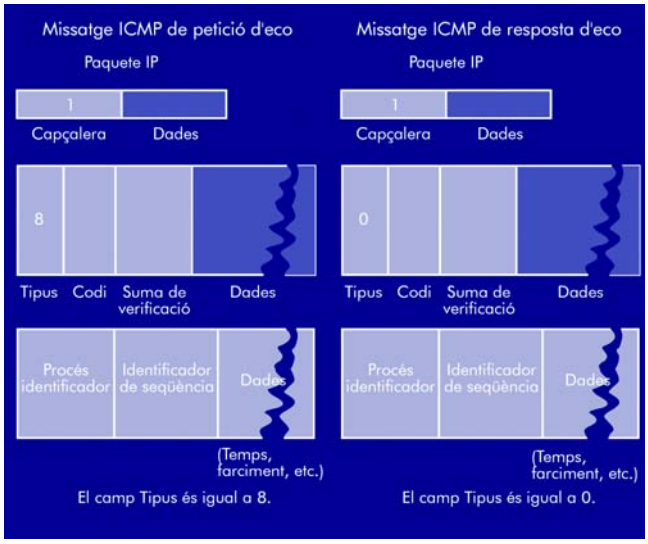
10.2. El programa `ping`

El programa `ping` permet descobrir si una estació es troba activa o no, simplement efectuant el següent:

```
$ ping <adreça_IP_destinació>
```

El programa `ping` envia un missatge ICMP del tipus 8 (petició d'eco) amb la destinació indicada. El receptor de la petició ha de respondre amb una resposta d'eco (ICMP tipus 0), i, quan el `ping` la rep, indica en pantalla que l'estació remota és activa. Evidentment, el programa s'hauria de dir ping-pong.

Figura 42.



ANOTACIONS

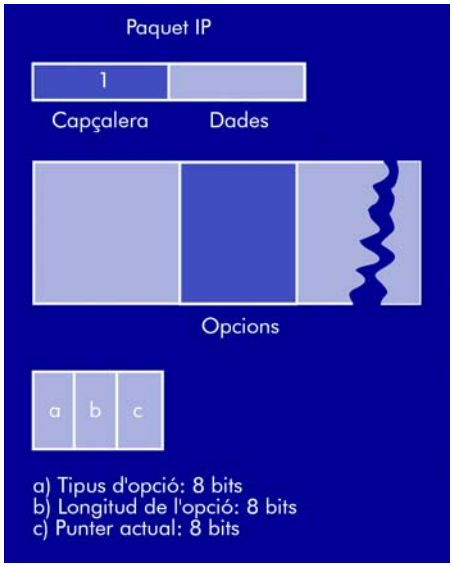
Activitat

Practiqueu l'ús del programa `ping`. Comproveu quins retards màxims es donen a Internet. Feu-ho a diferents hores del dia.

Amb el `ping` tenim altres opcions disponibles. En particular, l'opció de memorització de rutes (*record route* o RR; `ping -R`, en GNU/Linux), que no es reflecteix en cap dels camps del missatge ICMP, sinó que es troba dins de la mateixa capçalera IP, en el camp d'opcions.

Les diferents opcions que es troben disponibles a la capçalera s'identifiquen per mitjà del primer byte del camp d'opcions de la capçalera IP:

Figura 43.



Si l'originador, per exemple, vol activar l'opció RR, el primer byte ha de ser 7. En totes les opcions el primer byte indica el tipus, i el segon, la longitud en bytes de l'opció.

En aquest cas sempre es demana la longitud màxima possible, que són 39 bytes, ja que el camp següent té un byte (punter), al qual segueix una cadena de camps de 4 bytes (que són les adreces IP trobades). El camp Punter s'inicialitza a 4, i dins dels 4 bytes següents es desa l'adreça de l'estació en la qual executem el `ping`.

ANOTACIONS

Cada encaminador ha de mirar dins dels paquets IP (ICMP o no) per a veure si tenen l'opció RR activada. Si un encaminador troba un paquet amb aquesta opció activada, modifica la posició apuntada pel punter amb la seva adreça (per norma general, l'adreça de sortida de l'encaminador) i incrementa el valor del punter en quatre unitats. Quan torna el missatge de resposta d'eco, s'hi ha afegit una llista amb tots els salts que ha hagut de realitzar el paquet (tant d'anada, com de tornada).

El camp Opcions té limitacions de mida: 36 bytes (39 – 3) per a desar adreces IP. Com que cada adreça ocupa 4 bytes, només hi ha espai per a nou adreces IP. Si a això hi afegim que no tots els encaminadors comproven si hi ha opcions dins dels paquets IP, o no actualitzen l'opció RR, fa poc útil aquest tipus de ping al món real.

Activitat

Comproveu la ruta en diferents destinacions per mitjà d'un ping amb l'opció de memorització de rutes. Valoreu si la comprovació d'aquesta opció està molt estesa.

10.3. El programa `traceroute`

El programa `traceroute` permet trobar les rutes entre un origen i una destinació sense cap de les limitacions del `ping -R`. Utilitza un mecanisme bastant enginyós basat en missatges genèrics ICMP (i no en els específics petició d'eco i resposta d'eco del `ping`).

El funcionament es basa en l'explotació de dos missatges ICMP:

1. **Temps de vida esgotat** (*time-exceeded*): quan un encaminador rep un paquet, a part de les tasques primordials d'encaminament, ha de reduir en una unitat el valor del camp TTL de la capçalera IP.

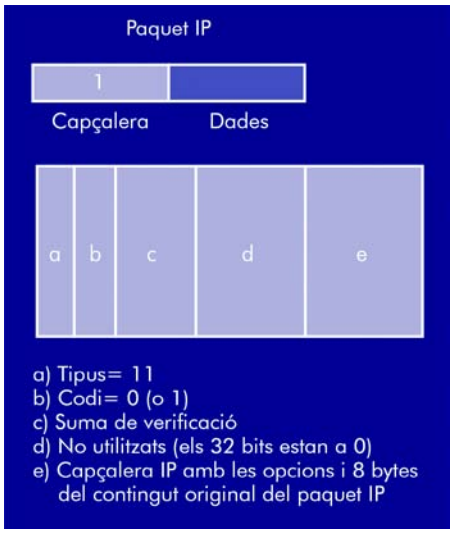
   En cas que el valor (després de la reducció) sigui zero, el paquet s'ha d'eliminar. Aquesta eliminació no és silenciosa, l'encaminament responsable n'envia una notificació a l'originador del

ANOTACIONS

paquet per mitjà d'un missatge ICMP tipus 11 i codi 0 (temps de vida esgotat).

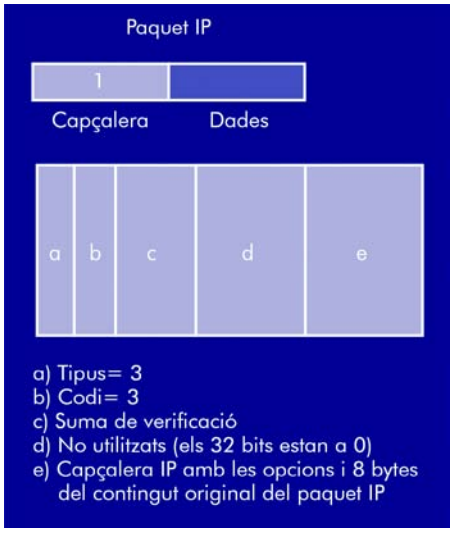
Aquest paquet ICMP conté la capçalera del paquet IP que s'ha eliminat i els primers 8 bytes del seu contingut (segurament la capçalera UDP o els primers bytes de la capçalera TCP).

Figura 44.



2. **Port inabastable** (*unreachable-port*): quan una estació rep un datagrama UDP o un segment TCP destinat a un port que la màquina no escolta, respon amb un missatge d'error de port inabastable (tipus 3 amb codi 3).

Figura 45.



ANOTACIONS

El programa `traceroute` simplement ha d'enviar paquets a la destinació amb TTL seqüencialment ascendents: el paquet (amb independència del tipus que sigui) que tingui el TTL = 1 serà rebutjat pel primer encaminador, el que tingui TTL = 2 ho serà pel segon, i així consecutivament. Cada un dels encaminadors tornarà un missatge ICMP "temps de vida esgotat", una pista del tot suficient perquè l'originador esbrini el camí que han seguit tots els paquets.

Quan el missatge arribi a la destinació, ha de tornar algun missatge per a saber que la seqüència ha finalitzat. Per norma general, el missatge serà "port inabastable" si el missatge enviat era un datagrama UDP a un port no usat, o bé una resposta d'eco si el que s'ha enviat són paquets ICMP de petició d'eco.

Activitats

Utilitzeu el programa `traceroute` per a descobrir els camins que segueixen diversos paquets fins a diferents destinacions.

Proveu què succeeix si ens connectem a un port no servit. Per exemple, connecteu-vos al port TCP 1234 (ho podeu fer amb `telnet localhost 1234`).

10.4. Missatge de reencaminament

És normal que els sistemes es connectin a Internet només configurant la seva adreça IP, la màscara de la LAN i l'encaminador que gestiona la comunicació remota. Quan hi ha més d'un encaminador a la LAN local, es pot donar el cas que per a algunes rutes sigui millor usar-ne un altre i no el que tenim configurat.

A aquest efecte, els encaminadors disposen del missatge ICMP de reencaminament (*redirect*), que actua de la manera següent:

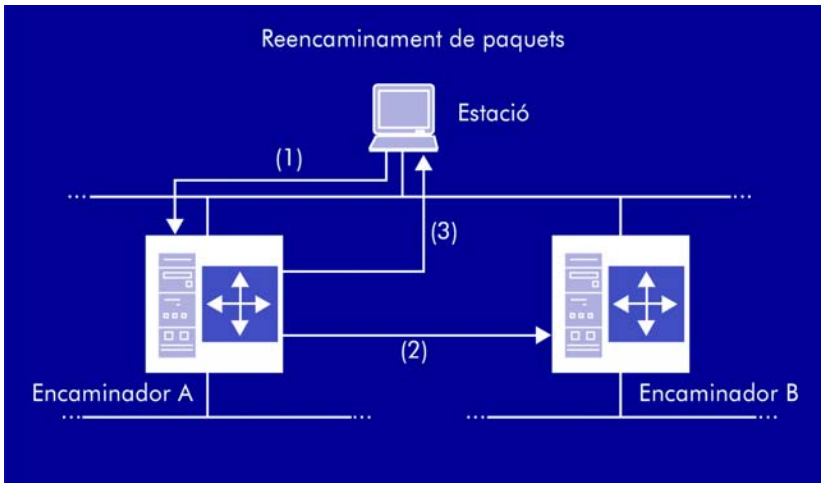
1. L'estació envia un paquet l'encaminador que té configurat (A). L'encaminador A descobreix que la millor ruta passa per utilitzar l'encaminador B.

ANOTACIONS

2. L'encaminador A direcciona el paquet cap a l'encaminador B.

3. Notifica a l'estació que modifiqui la seva taula d'encaminament.

Figura 46.



Noteu que, normalment, l'encaminador continua encaminant els paquets (pas 2). Encara que l'estació no fes cas del missatge ICMP de reencaminament (pas 3), continuaria tenint connectivitat amb l'exterior; evidentment, si en fa cas, millorarà el rendiment del sistema.

Quan una estació obeeix un ICMP de reencaminament, la seva taula d'encaminament queda convenientment actualitzada. La comanda `route` ens pot proporcionar alguna pista de si aquest fenomen té lloc.

ANOTACIONS

11. Xarxes d'accés a Internet

Les xarxes d'accés a Internet més habituals són la xarxa telefònica (per mòdem) que s'utilitza, sobretot, en l'àmbit domèstic, l'ADSL (*asymetric digital subscriber line*, línia d'abonament digital domèstica) que, encara que utilitza la infraestructura d'accés de la xarxa telefònica, no es pot dir que vagi sobre la línia telefònica, i l'Ethernet.

1. En accessos per mitjà de la **xarxa telefònica** i, en general, en accessos per mitjà de xarxes commutades (incloent-hi l'accés per l'XDSI), se sol utilitzar el PPP (*point-to-point-protocol*).

   Nota

   Si bé durant molt de temps s'han utilitzat l'SLIP (*serial line Internet protocol*) i el CSLIP (*compressed SLIP*), actualment s'han deixat pràcticament de banda en favor del PPP, que té més flexibilitat (permet gestionar automàticament certs paràmetres IP i multiplexar, dins de la mateixa connexió, diferents protocols d'interconnexió, a part de l'IP) i és més fiable (disposa de CRC en cada trama).

2. En **LAN**, el protocol que s'utilitza en més del 90% dels casos és l'Ethernet. Nosaltres ens centrarem només en els detalls de les adreces d'aquest protocol, ja que és l'únic que afecta la manera de funcionar de l'IP.

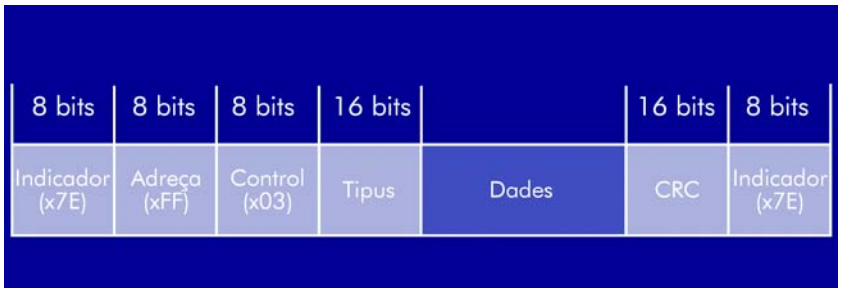
   Gairebé tots els protocols de LAN que componen el 10% restant (IEEE802.3 CSMA/CD, IEEE802.5 Token Ring, etc.) utilitzen una estructura d'adreces idèntica a la d'Ethernet. De fet, podríem parlar de compatibilitat, ja que l'assignació d'adreces es fa globalment per a totes aquestes LAN i la gestiona l'IEEE (Institute of Electric and Electronic Engineers).

ANOTACIONS

11.1. Accés telefònic: el PPP

El PPP és fonamentalment un protocol derivat de l'HDLC (*high-level data link protocol*) per a connexions balancejades (HDLC-ABM, *HDLC-asynchronous balanced mode*). El format de la trama PPP es representa en la figura següent:

Figura 47.



Els camps Indicador (*flag*), Adreça i Control estan fixats als valors de la figura anterior. El camp Adreça té el valor 11111111, que és el de difusió o *broadcast* en la majoria dels protocols (per exemple, en tots els HDLC). Això significa que aquest camp (com el camp Control) no s'utilitza. La seva utilització en el PPP només es pot justificar pel possible ús de targetes HDLC genèriques per a connexions PPP. Com qualsevol protocol HDLC, ha d'aplicar el mecanisme de transparència d'inserció de zeros (*bit stuffing*).

Nota

El PPP especifica una variant orientada a caràcter (els protocols de la família HDLC estan orientats a bit), que és la que més s'utilitza en enllaços per mitjà de mòdem (un contraexemple serien les connexions mitjançant XDSI).

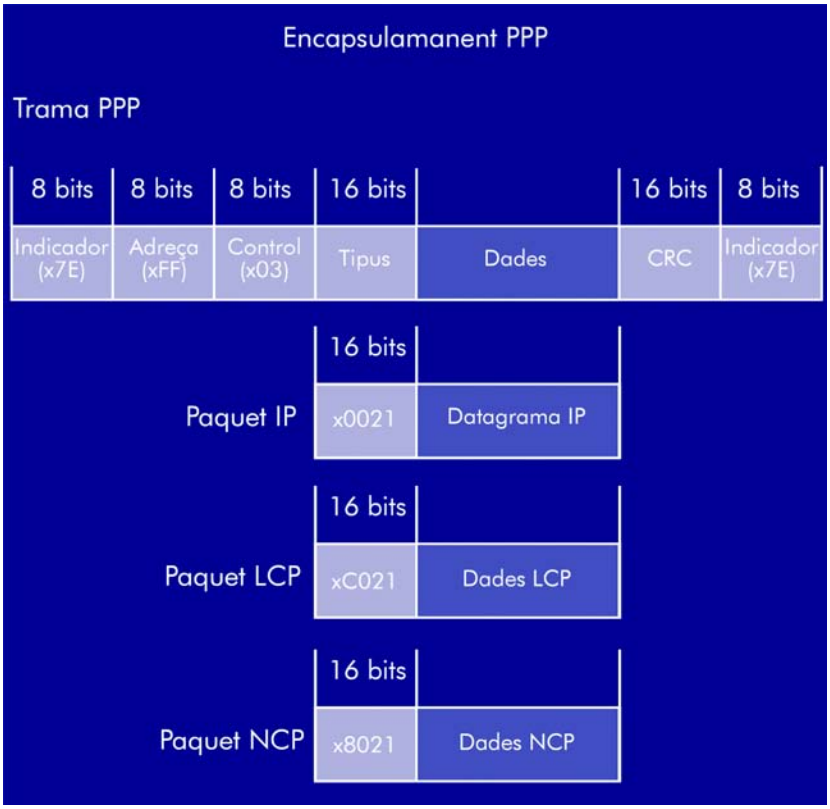
L'important és el que transporten les trames PPP. El mateix estàndard defineix la multiplexació de diferents protocols, que es distingiran per mitjà del camp Tipus. Els que ens interessen són els següents:

- **LCP** (*link control protocol*): és el protocol encarregat de realitzar el test de l'enllaç, el control de la connexió i la gestió de l'enllaç.

ANOTACIONS

- **Protocols de xarxa**: són trames que encapsulen paquets de nivell superior, com pot ser IP. Però també poden transportar NETBEUI, AppleTalk, Decnet, etc.
- **NCP** (*network control protocol*): és el protocol que s'utilitza per a tasques de gestió dels protocols de xarxa que transporta l'enllaç. En el cas de l'IP, permet que un dels terminals assigni l'adreça d'Internet a l'altre i configuri els diferents paràmetres de la xarxa (encaminador, màscara, etc.)

Figura 48.



11.1.1. Compressió de les capçaleres

Com ja hem vist, la piràmide de PDU provoca ineficiències en la transmissió, sobretot en protocols com Telnet, que amb freqüència envia blocs molt curts d'informació. Així mateix, el PPP s'ha utilitzat en enllaços telefònics que funcionen a velocitats màximes teòriques d'entre 9.600 bps (mòdems norma V.32) i 33.600 bps (mòdems norma V.34). Això fa que aquesta ineficiència sigui encara més greu.

ANOTACIONS

Nota

Si s'utilitza Telnet, per a enviar un caràcter (d'1 byte) hem d'enviar 8 + 20 + 20 + 1 = 49 bytes. Amb un mòdem de 9.600 bps, si utilitzéssim missatges de 8 bits de longitud més 1 bit d'arrencada i 1 bit de parada, podríem transmetre:

9.600 / (1 + 8 +1) = 960 caràcters/segon.

De fet, ningú no és capaç de mecanografiar tan ràpid. Tanmateix, no hem de perdre de vista que l'enllaç pot ser compartit per altres connexions (transferència de fitxers, consulta de la web, etc.).

La **compressió de capçaleres Van Jacobson** millora considerablement aquest problema. Aquest tipus de compressió es basa en el fet que en un accés a Internet amb PPP, en general no hi haurà massa connexions TCP simultànies. La compressió Van Jacobson permet mantenir una taula de fins a setze connexions simultànies TCP/IP, a les quals assignarà setze identificadors diferents. Com que la majoria dels camps de les dues capçaleres TCP/IP no varien durant el transcurs d'una mateixa connexió, n'hi ha prou de tenir entre 3 i 5 bytes de capçalera per a cada paquet (combinant PPP i TCP/IP) per a mantenir un funcionament correcte de la connexió. El guany en eficiència de la transmissió és prou important.

Per a saber amb exactitud quin guany s'aconsegueix, necessitem saber quina longitud poden tenir les trames PPP.

Per mitjà de l'LCP també es poden obtenir altres millores com, per exemple, l'eliminació dels camps Control i Adreça.

11.1.2. MTU

Com ja hem comentat, la majoria de protocols de nivell d'enllaç tenen una longitud màxima de transmissió, l'MTU, que condiciona la mida dels paquets de nivell superior que transporta. En el PPP, protocol derivat de l'HDLC, no es tracta d'un valor concret, sinó que el límit el fixarà la probabilitat d'error en un bit que tinguem: com més llarga sigui la trama, més probabilitat que sigui errònia.

ANOTACIONS

Tanmateix, hi ha un factor que limitarà l'MTU més que els límits imposats pels protocols: el **confort de la connexió**. S'ha demostrat que, en connexions en temps real (com una connexió Telnet), l'usuari ha de rebre una reacció a les seves accions en una dècima de segon com a màxim. Retards superiors provoquen cansament i donen la sensació que "la màquina va lenta", que tots hem experimentat alguna vegada.

Si pensem que dins de la mateixa connexió en podem tenir multiplexades d'altres de transferència (com FTP o web), ens trobarem que els paquets de les aplicacions en temps real, que solen ser curts, s'han d'esperar darrere dels paquets de longitud màxima que s'utilitzen en les aplicacions que tenen una taxa de transmissió elevada (suposarem que els paquets de l'aplicació en temps real tenen preferència sobre d'altres que prèviament estiguessin a la cua de sortida).

L'única manera de fer que una aplicació estàndard de transferència utilitzi paquets més petits és reduir l'MTU de la connexió. En el cas del PPP, si prenem com a referència un mòdem de 33.600 bps, tenim que un paquet d'una dècima de segon de durada tindria els bytes següents:

$$33.600 \text{ bps} \cdot (0{,}1 \text{ s}) \cdot (1 \text{ byte} / 8 \text{ bits}) = 420 \text{ bytes}.$$

En connexions PPP tindrem, doncs, l'MTU entre 250 i 500 bytes. Amb mòdems que disposin de compressió, els valors que obtindrem poden augmentar.

Nota

Les velocitats de transmissió dels mòdems estàndard són les següents: 9.600 bps (V.32), 14.400 bps (V.32bis) i 33.600 bps (V.34).

En general, els mòdems disposen de dispositius electrònics per a comprimir i descomprimir dades. Els estàndards V.42bis i MNP9 realitzen compressions de fins a 1:4, amb la qual cosa poden aconseguir velocitats de transmissió efectiva de fins a 134.400 bps (V.34 + V.42bis) en situacions favorables.

ANOTACIONS

Hi ha mòdems (els que segueixen l'estàndard V.90) que poden aconseguir velocitats de fins a 56 kbps. En realitat, no són mòdems en el sentit estricte, ja que necessiten que en l'altre extrem de la línia hi hagi un còdec connectat a una línia digital.

11.2. Accés ADSL

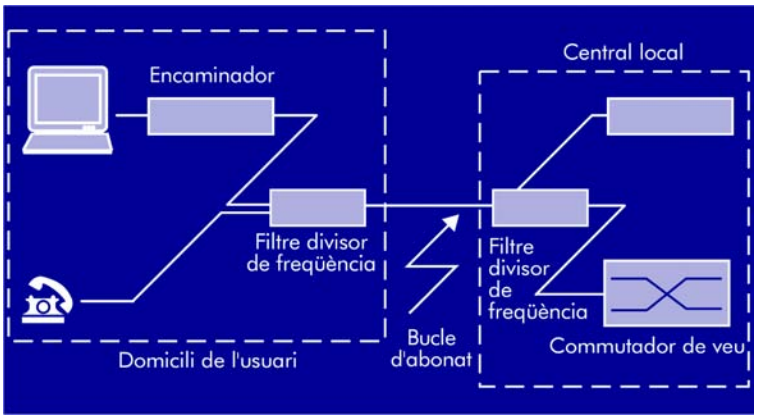
Si bé l'accés a Internet per mòdem per mitjà de PPP ha estat la manera habitual de connexió per part dels usuaris domèstics i les petites empreses durant la primera "dècada Internet", sembla que això canviarà i en la segona "dècada Internet" s'adoptaran sistemes que facilitin la connexió permanent.

Nota

La connexió per mòdem ocupa la línia de telèfon de l'abonat i, el que encara és pitjor, estableix una trucada durant un temps indefinit.

L'ADSL representa una solució a aquest problema, ja que, d'una banda (encara que utilitzi el cablatge de la línia telefònica, el bucle d'abonat), la línia telefònica queda lliure per a trucades mentre s'està connectat a Internet i, d'una altra, la connexió no consumeix recursos de la xarxa telefònica, ja que, quan el senyal arriba a la centraleta, s'extreu del bucle d'abonat i passa a la xarxa IP de l'operadora.

Figura 49.

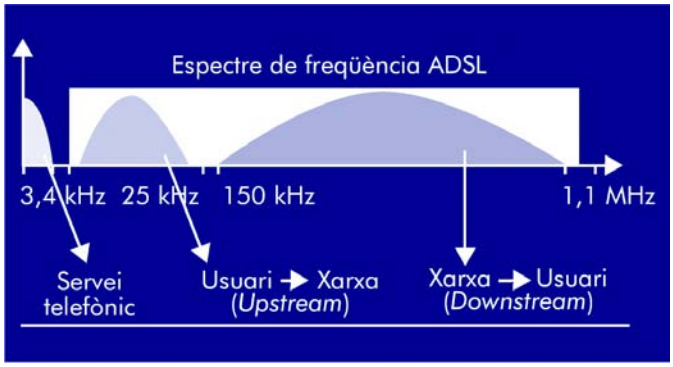


ANOTACIONS

En general, l'encaminador que hi ha al domicili de l'usuari ofereix una connexió Ethernet equivalent a un concentrador, la qual cosa permet que l'abonat connecti més d'un ordinador per mitjà de la línia ADSL.

Per a permetre que la línia telefònica convisqui amb la connexió a Internet permanent, es realitza una divisió de l'espectre: la part baixa continua ocupada pel canal de veu (fins a 4 kHz) i, a partir d'aquí, se situa l'espectre de la codificació ADSL (amb la limitació pròpia del parell de fils). El sistema és bidireccional: reserva l'espectre baix per a la sortida a la xarxa, i la resta, per a l'entrada:

Figura 50.



Generalment, no es pot arribar als 1,1 MHz que indica la figura, ja que la qualitat del parell de fils és molt variable (clima, longitud, edat, etc.), per la qual cosa s'utilitza una codificació adaptativa: els dos sentits del canal es divideixen en subbandes de 4 kHz independents (DMT, *discrete multitone*) que es codifiquen (en realitat, es modulen) amb procediments gairebé idèntics als utilitzats pels mòdems tradicionals. Amb això, s'aconsegueixen trenta-dos canals de sortida i fins a dos-cents cinquanta-sis d'entrada, amb una capacitat de 60 kbps cada un –modulats en QAM (*quadrature amplitude modulation*), com els mòdems telefònics que, acumulats, proporcionen un màxim de 15,36 Mbps d'entrada i 1,92 Mbps de sortida.

No obstant això, les millors connexions no aconsegueixen aprofitar correctament els canals superiors i es considera que el límit màxim de sortida és de 8 Mbps. Així mateix, les operadores no solen oferir connexions tan ràpides, la qual cosa fa que no es pugui assolir el límit teòric.

Nota

Alguns operadors, per a aprofitar millor línies dolentes, utilitzen una variant de l'estàndard que permet la transmissió bidireccional amb els dos canals a la mateixa banda.

Aquesta solució, encara que necessita DCE més cars, permet aprofitar línies amb pitjors condicions i/o obtenir velocitats més altes.

ANOTACIONS

Els protocols utilitzats dins de l'ADSL depenen de l'operadora i no estan definits per l'estàndard. Entre els possibles protocols per a l'ADSL, tenim l'ATM i el mateix PPP.

11.3. Accés LAN: el protocol Ethernet

Segurament, la simplicitat d'aquest protocol de xarxa, i no les seves prestacions teòriques, ha fet que sigui el més utilitzat en xarxes d'àrea local pràcticament des que DEC, Intel i Xerox van establir un estàndard (*de facto*) per a LAN amb control d'accés CSMA/CD, basant-se en una arquitectura similar desenvolupada en els anys setanta per Xerox.

Hi ha un principi que es compleix en totes les **xarxes d'àrea local**: el que una estació transmet és rebut per totes les altres. Una estació sap quan una trama li va destinada perquè llegeix totes les que li arriben i en comprova l'adreça de destinació. Ha de rebutjar totes les trames amb adreces que no siguin la seva. Tanmateix, també hi ha excepcions, i a vegades les estacions també han de capturar trames dirigides a adreces especials (com les *multicast* i les *broadcast*).

Nota

Un dels casos en què una estació no rebutja les trames amb adreces diferents de la seva és quan l'estació configura la targeta de xarxa en mode promiscu. En aquesta mode, la targeta inhabilita la seva capacitat de filtratge i llegeix totes les trames que passen per la xarxa. Equips que, generalment, funcionen en aquesta mode són els ponts (*bridges*, sistemes destinats a la interconnexió de LAN), els analitzadors de trànsit (o analitzadors de xarxa) o els commutadors (*switches*). No obstant això, gairebé totes les targetes es poden configurar en mode promiscu, cosa que amb freqüència és aprofitat pels lladres d'informació (*hackers*) per

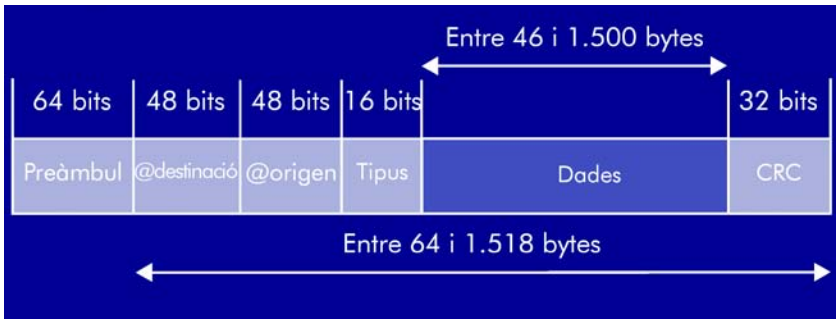
ANOTACIONS

a llegir i copiar informació interessant que viatgi per la LAN (principalment contrasenyes).

11.3.1. Format de la trama Ethernet

La manera de conèixer les principals característiques de la trama Ethernet és veure els diferents camps que la formen, que són els següents:

Figura 51.



a) **Preàmbul**: està format per 64 bits, alternativament 0 i 1. Els dos últims són 11. Això genera un senyal quadrat que permet als terminals sincronitzar adequadament els rellotges de sincronisme de bit. Els dos últims bits assenyalen on comença la trama (sincronisme de trama). La seva forma és idèntica en totes les trames. Nosaltres obviarem la seva presència en la resta de l'explicació, ja que només són un senyal per a marcar l'inici de la trama.

b) **Adreça de destinació**: porta l'adreça MAC del destinatari especificada de la mateixa manera (en el mateix format) que l'adreça d'origen. En aquest cas, tanmateix, tenim tres tipus d'adreces possibles: *unicast*, *multicast* i *broadcast*.

c) **Adreça d'origen**: porta l'adreça física o adreça MAC del transmissor de la trama. Són 48 bits diferents per a qualsevol terminal de la xarxa Ethernet.

d) **Tipus**: indica el tipus de contingut del camp de dades que porta la trama (les trames que transporten paquets IP porten un 0 x 800). Permet multiplexar diferents protocols dins d'una mateixa LAN. Xerox

ANOTACIONS

actualitza regularment la llista de protocols registrats (*Xerox Public Ethernet Packet Type*). Més endavant veurem les varietats de protocols d'Ethernet per a conèixer les variants d'Ethernet semicompatibles i saber com afecta la seva coexistència a la manera com l'Ethernet ha hagut de definir el camp Tipus.

Activitat

Consulteu la llista dels protocols registrats per a fer-vos una idea de la quantitat de protocols de xarxa (els superiors a Ethernet) que hi ha.

e) **Dades**: es refereix al format del camp de dades. Les restriccions sobre el tipus de dades que pot transportar l'Ethernet són les següents:

- La longitud de les dades ha de ser múltiple de 8 bits; és a dir, l'Ethernet transporta la informació en bytes. Això no és cap impediment, ja que els bits els envien sistemes que, per norma general, treballen amb bytes o múltiples de bytes.
- De la mateixa manera que PPP, l'Ethernet té limitada la longitud màxima d'informació transportable per la trama (MTU). En aquest cas, l'MTU és de 1.500 bytes. Aquesta limitació té com a objectiu evitar que una estació monopolitzi la LAN.
- El camp de dades ha de tenir com a mínim 46 bytes de longitud (en Gigabit Ethernet, 512). Això es deu al fet que és necessari que la trama mínima Ethernet tingui 64 bytes (512 bits). Es considera que les trames inferiors són resultat de col·lisions i els receptors les obvien.

Aquest problema no es planteja en la variant d'Ethernet IEEE802.3, ja que aquest protocol disposa d'un camp de longitud i un altre de farciment (*padding*) que permeten transmetre dades de fins a 1 byte de longitud, encara que la longitud que físicament es transmet continuï essent de 64 bytes.

Activitat

Imagineu què succeiria si una estació volgués enviar un fitxer d'1 GB per la LAN dins d'una sola trama.

f) **CRC**: és un codi de redundància cíclica de 32 bits (CRC-32) per a detectar errors. Inclou tota la trama a excepció del preàmbul.

ANOTACIONS

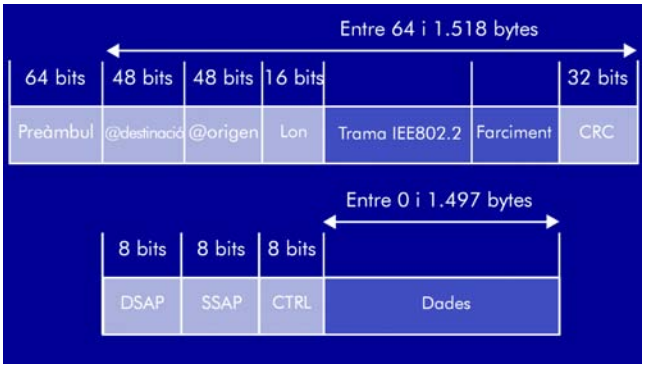
Les trames que no tenen un CRC correcte s'ignoren (com les trames de menys de 64 bytes i les que no són múltiples de 8 bits).

Nota

Des de mitjan anys vuitanta l'Ethernet ha conviscut amb una variant similar denominada *IEEE802.3* o *ISO8802.3*. Són estàndards establerts per organitzacions reconegudes (l'IEEE i l'ISO) dedicades a la normalització (estàndard *de iure*). Durant un temps es va pensar que l'IEEE802.3 acabaria substituint l'Ethernet original (també denominat *Ethernet-DIX*, en honor a DEC, Intel i Xerox), que no podia transmetre trames arbitràriament petites. Els protocols que treballen sobre Ethernet-DIX coneixen aquesta limitació i omplen la trama fins a ocupar els 46 bytes d'informació.

L'IEEE802.3 introdueix un camp Longitud (en la mateixa posició en la qual l'Ethernet té el camp Tipus) que permet saber quants bytes útils conté el camp Dades. Si la longitud no arriba als 46 bytes mínims, s'omple amb bytes (indefinits) fins que arribi al mínim. El receptor només ha de llegir el camp de longitud per a extreure'n la informació vàlida. El concepte de *tipus d'Ethernet* (és a dir, la coexistència de diferents protocols per sobre d'Ethernet/IEEE802.3) es delega a un protocol associat: l'IEEE802.2, protocol d'enllaç que es pot utilitzar en l'IEEE802.3 i en altres protocols de LAN i que té unes funcions similars a les de l'HDLC.

Figura 52.



ANOTACIONS

Com que físicament tots dos estàndards són totalment compatibles, ens podríem preguntar si poden coexistir trames Ethernet-DIX i IEEE802.3 dins d'una mateixa LAN. La resposta és que sí i que no alhora.

Fixeu-vos que tots els camps de la trama Ethernet i de la 802.3 són del mateix format i signifiquen el mateix, a excepció del camp de tipus (Ethernet) i de longitud (802.3). Els podríem distingir sempre que vigiléssim que cap tipus Ethernet no fos equivalent a una longitud vàlida de 802.3. Els tipus amb valors inferiors a 1.500 (0 x 5DC en hexadecimal) es poden confondre amb longituds vàlides.

Això, òbviament, no podia tenir-se en compte en l'Ethernet-DIX original, ja que és anterior a l'IEEE802.3. Per això, va aparèixer una addenda a la norma, coneguda com Ethernet-DIX-II, que elimina els identificadors de protocols per sota de 0 x 0600 (1.536 en decimal). Avui dia amb freqüència dins d'una mateixa LAN trobem trames Ethernet-DIX-II i trames IEEE802.3.

L'IP pot anar sobre qualsevol dels dos estàndards, encara que gairebé ningú no tria la possibilitat d'encapsular-lo sobre l'IEEE802.3. El parell de protocols IEEE802.3 + 802.2 s'utilitza en alguns dels sistemes operatius de xarxa apareguts en els anys vuitanta com, per exemple, l'IPX de Novell i el NETBEUI de Microsoft.

11.3.2. Tipus de medis físics en Ethernet

L'Ethernet s'ha adaptat a les necessitats del temps ampliant els subestàndards de nivell físic. A continuació, mostrem una llista dels estàndards més utilitzats:

- 10Base2: abast de 185 m, o 925 m amb repetidors, però amb coaxial prim, flexible i barat (és per això que durant molts anys

ANOTACIONS

aquesta xarxa s'ha denominat *Cheapernet*). Encara que avui dia es tendeix gradualment a deixar-lo de banda en favor de 10BaseT –que és molt més fiable–, milions de terminals a tot el món estan connectats amb Ethernet-10Base2. S'utilitza sobretot en topologies en bus.

- 10BaseT: connexió en estrella de les estacions a un nus central (concentrador) per mitjà de parell trenat; la distància màxima de l'estació al concentrador és de 100 m. La distància màxima entre estacions s'aconsegueix encadenant quatre concentradors, i és de 500 m.

  Representa una millora important respecte als estàndards anteriors, ja que se centralitzen en un sol punt la gestió i el monitoratge de l'estat de tota la LAN. Així mateix, amb les topologies en bus, el mal funcionament d'una estació podia comportar el bloqueig de tota la xarxa. Amb 10BaseT, una mala connexió d'un terminal és detectada pel concentrador, que simplement la desconnecta i indica que l'enllaç a l'estació està tallat o inactiu (amb un llum vermell, per exemple).

- 10BaseF: similar a 10BaseT; però, en lloc de parell trenat, utilitza fibra òptica (generalment, multimode), amb la qual cosa s'aconsegueix un abast molt més gran (fins a 2 km).

- 100BaseT i 100BaseF: similars a 10BaseT i 10BaseF, respectivament; però, funcionen a 100 Mbps. A causa del protocol de control d'accés CSMA/CD, se'n redueix molt l'abast (100 m entre estació i concentrador, sense possibilitat d'encadenar concentradors).

- Gigabit Ethernet: les variants més comunes són 1000BaseT, sobre cablatge de coure categoria 5 (equivalent al necessari per a 100BaseT) i 1000BaseSX, sobre fibra. Té el mateix abast que 100BaseT, 100 m.

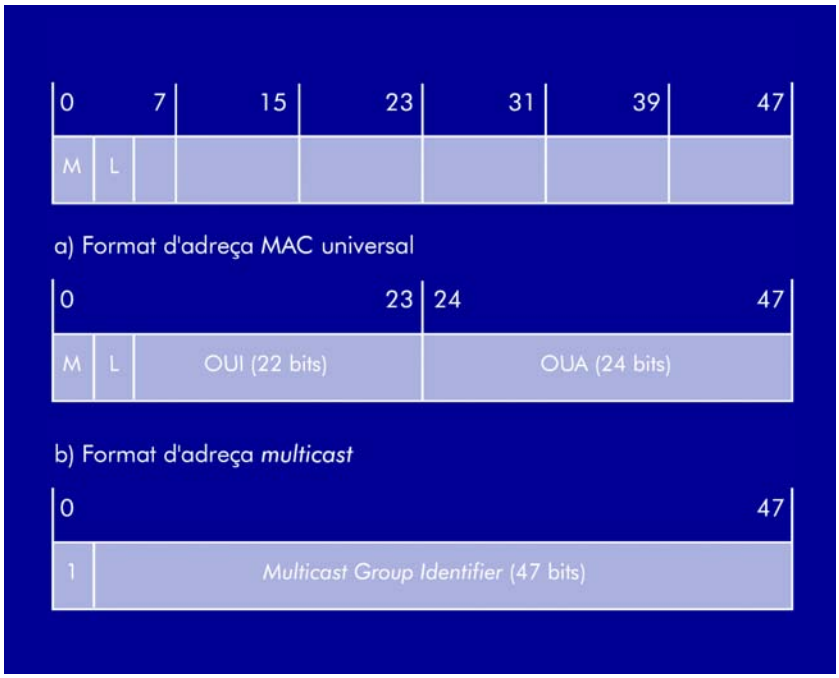
- 10 Gigabit Ethernet: actualització d'Ethernet per al segle XXI.

ANOTACIONS

11.3.3. Adreces LAN

Les adreces LAN estan dividides en diferents camps, com es pot observar en la figura següent:

Figura 53.



La meitat menys significativa de l'adreça (els bits del 2 al 23), assignada per l'IEEE a cada fabricant de manera fixa, és l'OUI (*organizational unique identifier*). Aquest últim, quan fabrica les targetes, programa (a ROM) l'adreça completa, que està formada per l'OUI més una part variable que el mateix fabricant assigna individualment per a cada targeta: l'OUA (*organizational unique address*).

Hi ha dos bits de l'OUI que sempre són zero quan es tracta de l'adreça d'origen: el bit *multicast* (M) i el bit *local* (L). Aquest últim no s'utilitza gairebé mai i, per tant, el considerarem sempre zero.

Tant l'adreça de destinació com la d'origen de la trama tenen el mateix format, amb l'única diferència que l'adreça de destinació també pot ser de tipus *multicast* (bit M = 1). En aquest cas, el nombre que porta no es refereix a una estació en particular, sinó que es dirigeix a un grup d'estacions, cada una de les quals coneix el grup o grups

ANOTACIONS

als quals està adscrita. Per norma general, cada un dels grups es refereix a grups d'estacions que comparteixen una mateixa aplicació o un mateix protocol. En l'IP, l'únic grup *multicast* Ethernet rellevant és el de difusió (*broadcast*).

Sobre paper, les adreces LAN s'escriuen en hexadecimal, separant els bytes amb dos punts i escrivint primer el byte menys significatiu, per exemple:

08:00:00:10:97:00

El primer byte (que és el menys significatiu) és sempre divisible per quatre en adreces no *multicast* que tenen els bits M i L a 0.

El grup *broadcast* és especial, en el sentit que, per defecte, totes les estacions li pertanyen; per tant, és una manera de difondre informació simultàniament a totes les estacions. El concepte de difusió (*broadcast*) no és exclusiu d'Ethernet, sinó que és comú a molts altres protocols de LAN i WAN (també en l'IP). Posar tots els bits de l'adreça a 1 constitueix la manera més habitual de representar l'adreça *broadcast*, i és la que utilitzen les LAN (FF:FF:FF:FF:FF:FF).

Nota

No és del tot cert que l'únic grup *multicast* Ethernet rellevant sigui el de difusió. La xarxa Internet disposa del protocol IGMP (*Internet group multicast protocol*), que també treballa sobre trames Ethernet *multicast*.

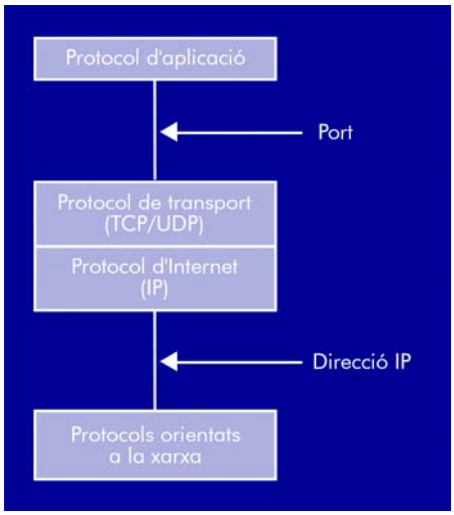
ANOTACIONS

12. Protocols del nivell de transport

L'objectiu principal del nivell de transport és establir una comunicació d'extrem a extrem per mitjà d'una xarxa. En altres paraules, actuar d'interfície entre els nivells orientats a l'aplicació i els nivells orientats a la xarxa de la jerarquia de protocols (tant OSI com TCP/IP).

El nivell de transport oculta als nivells alts del sistema el tipus de tecnologia (xarxa) a la qual està connectat el terminal. La figura següent descriu el posicionament del nivell de transport respecte a la resta dels nivells:

Figura 54.



En aquest apartat ens interessen els dos protocols del nivell de transport que es defineixen a la pila TCP/IP: UDP i TCP. UDP no és orientat a la connexió, mentre que TCP és orientat a la connexió.

En el nivell de transport es defineixen dues adreces que el relacionen amb els nivells superior i inferior:

- L'**adreça IP**, que ja coneixem, és l'adreça que identifica un subsistema dins d'una xarxa.

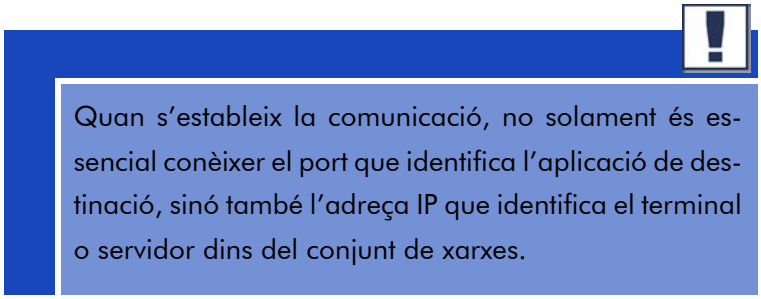
ANOTACIONS

- El **port** identifica l'aplicació que requereix la comunicació.

Per a identificar les diferents aplicacions, els protocols TCP/IP marquen cada paquet (o unitat d'informació) amb un identificador de 16 bits anomenat *port*.

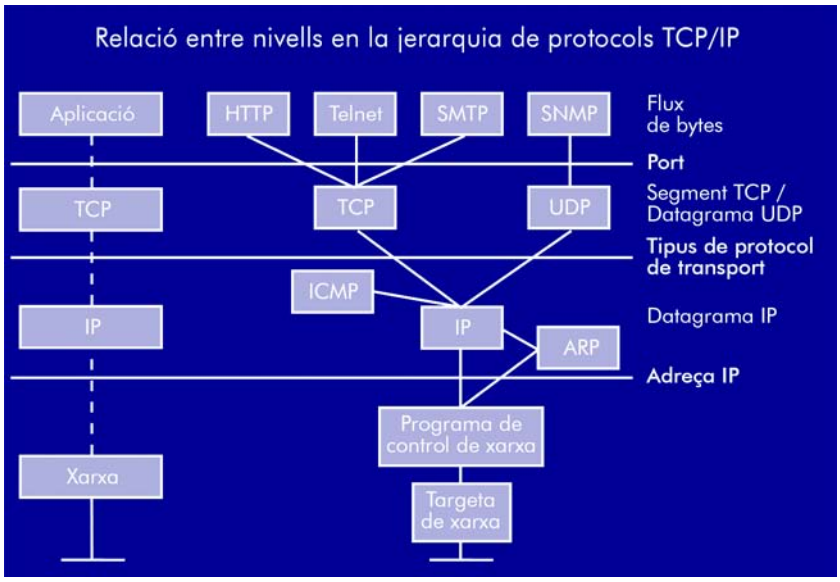
La vertadera utilitat dels ports és que permeten multiplexar aplicacions sobre protocols del nivell de transport. Això significa que un mateix protocol de transport porta informació de diferents aplicacions i aquestes són identificades pel port.

Si alguna aplicació que corre en un terminal vol establir una comunicació amb un servidor o amb un altre terminal, ha d'utilitzar un protocol de transport: el TCP o l'UDP. Com que la destinació es pot trobar en una xarxa remota, els protocols de transport necessiten el protocol Internet per a poder arribar al terminal o servidor remot.



Quan s'estableix la comunicació, no solament és essencial conèixer el port que identifica l'aplicació de destinació, sinó també l'adreça IP que identifica el terminal o servidor dins del conjunt de xarxes.

Figura 55.



Com podeu observar en la figura anterior, les aplicacions utilitzen un dels dos protocols de transport per a comunicar-se amb equips remots. Perquè un protocol d'aplicació es pugui comunicar amb un altre del mateix nivell situat en un terminal remot, li ha de transmetre un flux de bytes que és encapsulat pels protocols del nivell de transport.

El conjunt de bytes que transmet el nivell de transport TCP es coneix com a ***segment TCP***, mentre que el conjunt de bytes que transmet el protocol de transport UDP es diu ***datagrama UDP***.

En general, dues aplicacions es comuniquen seguint el model client/servidor. En una connexió és típic que una aplicació (el client) iniciï una comunicació demanant una informació a una altra aplicació (el servidor). Pensem en un ordinador que estigui connectat a una LAN i tingui assignada una adreça IP. Suposem que aquest ordinador actua com a servidor de correu electrònic, a més de com a servidor de noms. Un client connectat a Internet que sol·licita resoldre un nom necessita conèixer l'adreça IP assignada a aquest ordinador i el port que identifica l'aplicació servidor que resol noms.

Nota

Veurem el model client/servidor en la unitat 15.

El client necessita conèixer totes dues adreces, ja que el servidor estarà connectat a una xarxa i, per tant, tindrà una adreça IP que ha de ser coneguda perquè es pugui establir una comunicació amb aquesta màquina remota. Aquesta comunicació s'aconsegueix per mitjà de l'IP. Una vegada aconseguida, el servidor ha de ser capaç d'identificar l'aplicació amb què es vol comunicar el client entre les moltes que corren: servidor de noms, servidor de correu electrònic, etc.

El client coneix l'adreça IP d'origen (la seva), l'adreça IP de destinació (la del servidor) i el seu port d'origen (identifica l'aplicació client). Però, també ha de conèixer el número (port de destinació) que identifica l'aplicació desitjada al servidor, i ho fa per mitjà dels anomenats *ports coneguts* (*well-known port*).

ANOTACIONS

Un **port conegut** (*well-known port*) és un port (número) reservat que identifica una aplicació coneguda. Aquests ports són assignats per IANA (Internet Assigned Numbers Authority).

Exemple

Els valors de ports coneguts per a aplicacions que utilitzen l'UDP són els següents:

- Port 7 per al servidor d'eco.
- Port 53 per al servidor de noms (DNS, *domain name server*).
- Port 69 per al protocol de transferència de fitxers trivial (TFTP, *trivial file transfer protocol*).

I alguns valors de ports coneguts per a aplicacions que utilitzen el TCP són els següents:

- Ports 20 i 21 per al protocol de transferència de fitxers, FTP de dades i de control respectivament.
- Port 23 per al *Telnet Remote Login*.
- Port 80 per a l'HTTP.

Evidentment, el servidor no necessita conèixer *a priori* el port d'origen, ja que es limita a respondre a qualsevol petició de qualsevol client. Aquest últim informa en la unitat de dades de protocol (PDU) del nivell de transport (o bé un datagrama UDP, o bé un segment TCP) dels ports d'origen i de destinació, de manera que el servidor coneixerà el port d'origen una vegada hagi rebut una petició.

ANOTACIONS

13. L'UDP (*user datagram protocol*)

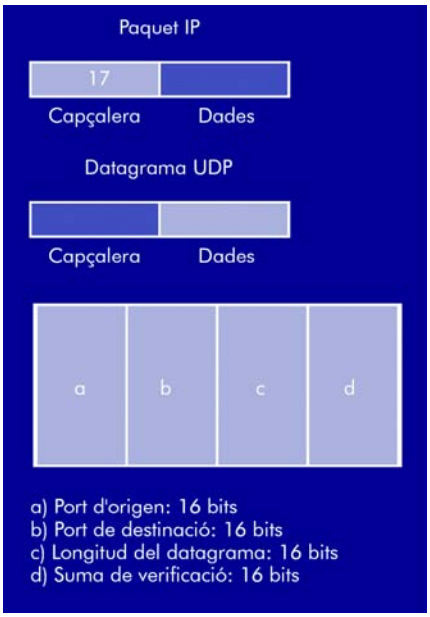
L'UDP és un protocol no orientat a la connexió, de manera que no proporciona cap tipus de control d'errors ni de flux, encara que utilitza mecanismes de detecció d'errors. En cas de detectar un error, l'UDP no lliura el datagrama a l'aplicació, sinó que el descarta. Convé recordar que, per sota, l'UDP està utilitzant l'IP, que també és un protocol no orientat a la connexió. Les característiques més importants de l'UDP són les següents:

Nota

L'UDP és un protocol no orientat a la connexió. Això significa que cada datagrama UDP existeix amb independència de la resta dels datagrames UDP.

- No garanteix la fiabilitat; és a dir, no ofereix la seguretat que cada datagrama UDP transmès arribi a la seva destinació; és un protocol *best-effort*: l'UDP fa tot el possible per a transferir els datagrames de la seva aplicació, però no en garanteix el lliurament.
- No preserva la seqüència de la informació que li proporciona l'aplicació. Com que està en mode datagrama i utilitza un protocol per sota com l'IP, que també està en mode datagrama, l'aplicació pot rebre la informació desordenada. L'aplicació ha d'estar preparada perquè hi hagi datagrames que es perdin, arribin amb retard o s'hagin desordenat.

Figura 56.



ANOTACIONS

La figura anterior mostra la unitat de dades del protocol UDP i la seva encapsulació en un paquet IP. Cada operació de sortida d'un datagrama UDP provoca la generació d'un paquet IP.

El datagrama UDP consta d'una capçalera i un cos per a encapsular les dades. La capçalera consta dels elements següents:

Nota

Hi ha moltes aplicacions que limiten la mesura de les seves memòries intermèdies de transmissió i recepció per sota de la mesura màxima d'un datagrama UDP. Per exemple, és típic trobar aplicacions que proporcionen, per defecte, mesures màximes del datagrama UDP de 8.192 bytes. Aquest valor prové de la quantitat de dades de l'usuari que l'NFS (*network file system*) pot llegir o escriure per defecte.

- Els **camps Port d'origen i Port de destinació**, que identifiquen les aplicacions en els terminals d'origen i de destinació. Cada port té 16 bits.
- El **camp Longitud** indica la longitud, en bytes, del datagrama UDP incloent-t'hi la capçalera UDP (és la diferència de la longitud del datagrama IP menys la capçalera IP). Com que la longitud màxima d'un datagrama IP és de 65.535 bytes, amb una capçalera estàndard de 20 bytes, la longitud màxima d'un datagrama UDP és de 65.515 bytes.
- El **camp Suma de verificació** (16 bits) és opcional i protegeix tant la capçalera com les dades UDP (recordeu que la suma de verificació del datagrama IP només cobreix la capçalera IP). Quan l'UDP rep un datagrama i determina que hi ha errors, el descarta i no el lliura a cap aplicació.

Nota

El càlcul de la suma de verificació en l'UDP és molt semblant al càlcul de la suma de verificació en l'IP (suma en complement a 1 de paraules de 16 bits), amb la particularitat que la longitud del datagrama UDP pot ser parella o senar. En cas que sigui senar, s'hi afegeix un 0 al final del datagrama per a calcular la suma de verificació (aquest 0 no es transmet).

Per a calcular la suma de verificació, l'UDP utilitza una pseudocapçalera de 12 bytes amb alguns dels camps IP. Aquesta última no es transmet; l'UDP només la utilitza per a calcular la suma de verificació i li serveix per a comprovar que la informació que li proporciona l'IP sigui realment per a ell.

Si l'UDP no aporta res a IP, es podria pensar que la seva presència és supèrflua. Però no és així, perquè no

ANOTACIONS

és estrictament cert que no aporti res. Aporta la multiplexació d'aplicacions sobre la mateixa comunicació de xarxa, per mitjà del concepte de port.

Les aplicacions que no requereixen la funcionalitat del TCP usen l'UDP com a protocol de transport. Podem posar dos exemples d'aquestes aplicacions:

- Aplicacions en temps real. Aquestes aplicacions requereixen poc retard (més ben dit, poca variabilitat en el retard), i TCP pot introduir retards considerables si ha d'esperar, per exemple, que li arribi un paquet que s'ha perdut.
- Aplicacions interessades a transmetre informació en manera *multicast* o *broadcast* (a un grup d'usuaris o a tots els d'una xarxa). En aquest cas, no té sentit establir una connexió com fa el TCP amb cada una de les estacions destinació.

ANOTACIONS

14. El TCP (*transmission control protocol*)

Com hem pogut observar, l'UDP no garanteix el lliurament de la informació que li proporciona una aplicació. Tampoc no reordena la informació en cas que li arribi en un ordre diferent d'aquell en què s'ha transmès. Hi ha aplicacions que no poden tolerar aquestes limitacions. Per a superar-les, el nivell de transport proporciona un protocol anomenat *TCP*.

El TCP proporciona fiabilitat a l'aplicació; és a dir, garanteix el lliurament de tota la informació en el mateix ordre en què ha estat transmesa per l'aplicació d'origen. Per a aconseguir aquesta fiabilitat, el TCP proporciona un servei orientat a la connexió amb un control de flux i d'errors.

14.1. El TCP proporciona fiabilitat

Per a proporcionar un servei fiable a l'aplicació, el TCP es basa en els principis següents:

1. **Transmissió lliure d'error**. El TCP ha de lliurar a l'aplicació de destinació exactament la mateixa informació que li ha lliurat l'aplicació d'origen. De fet, es tracta d'un lliurament "quasi lliure" d'errors, ja que n'hi pot haver alguns que un mecanisme de detecció d'errors no pugui detectar.
2. **Garantia de lliurament de la informació**. El TCP garanteix que tota la informació transmesa per l'aplicació d'origen es lliuri a l'aplicació de destinació. Si no és possible, el TCP ha d'avisar l'aplicació.
3. **Garantia de manteniment de la seqüència de transmissió**. El TCP garanteix el lliurament del flux d'informació en el mateix ordre en què li ha estat lliurat per l'aplicació d'origen.
4. **Eliminació de duplicats**. El TCP garanteix que només lliurarà una còpia de la informació transmesa a l'aplicació de destinació. En

ANOTACIONS

cas que rebi còpies a causa del funcionament de la xarxa o dels protocols que s'implementen per sota del nivell de transport, les elimina.

La fiabilitat de la transmissió que proporciona el TCP s'aconsegueix gràcies a les estratègies següents:

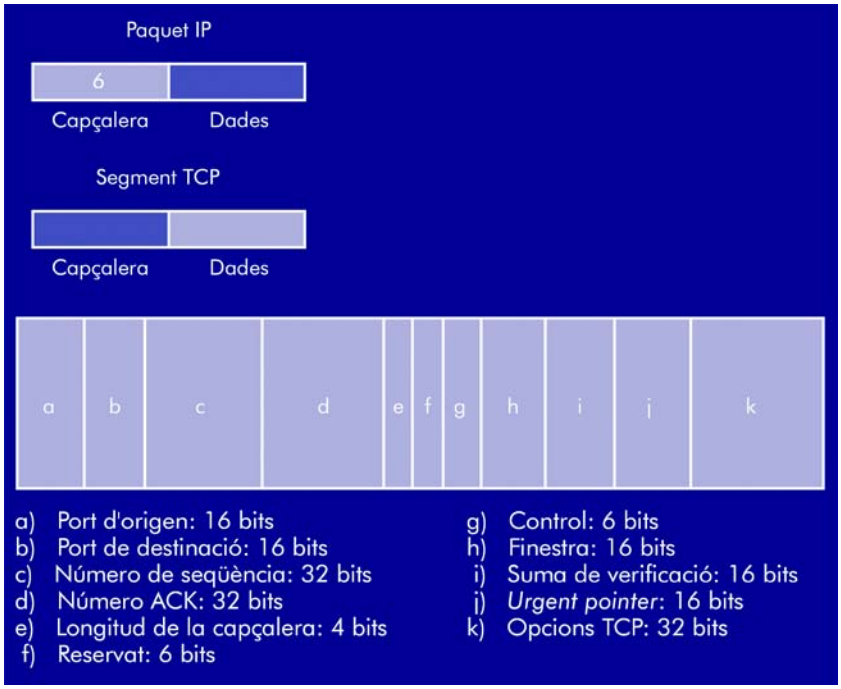
- El TCP està orientat a la connexió: té una fase d'establiment de la connexió, una de transmissió de dades i una de desconnexió.
- El TCP utilitza el concepte *buffered transfer*: quan es transfereix informació, el TCP divideix els fluxos de dades (*byte stream*) que li passa l'aplicació en segments de la mida que li convingui. El TCP decideix la mida dels segments tant si l'aplicació genera un byte d'informació, com si genera fluxos de grans dimensions. En el primer cas, el TCP pot esperar que la memòria intermèdia s'ompli més abans de transferir la informació, o bé la pot transferir tot seguit (mecanisme *push*). En cas que els fluxos siguin molt grans, el TCP pot dividir la informació en fragments més petits abans de transferir-los.
- El TCP utilitza una connexió *full duplex*: la transferència d'informació és en tots dos sentits. L'aplicació veu dos fluxos independents. En cas que l'aplicació tanqui un dels fluxos, la connexió passarà a ser *half duplex*. Això significa que un dels extrems (el que no ha tancat la connexió) pot continuar enviant informació pel canal, mentre que l'altre extrem (el que ha tancat la connexió) es limita a reconèixer-la. No obstant això, no és normal trobar aquest cas. El més habitual és que, si un extrem tanca la connexió, l'altre també la tanqui.

ANOTACIONS

14.2. Format del segment TCP

La unitat d'informació del protocol TCP s'anomena *segment TCP* i el seu format és el següent:

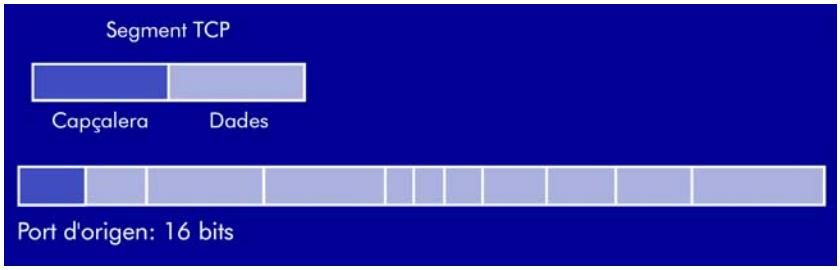
Figura 57.



El segment TCP consta d'una capçalera i un cos per a encapsular dades. La capçalera consta dels camps següents:

a) El camp **Port d'origen** identifica l'aplicació al terminal d'origen.

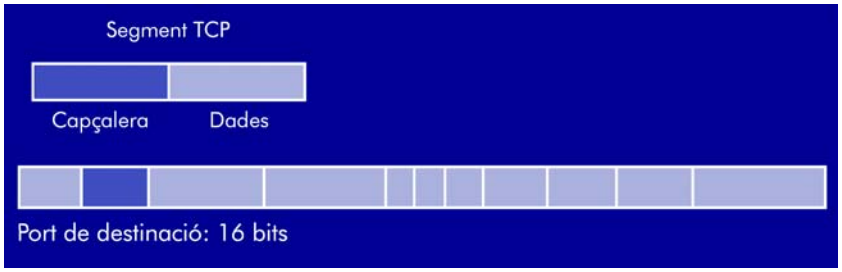
Figura 58.



ANOTACIONS

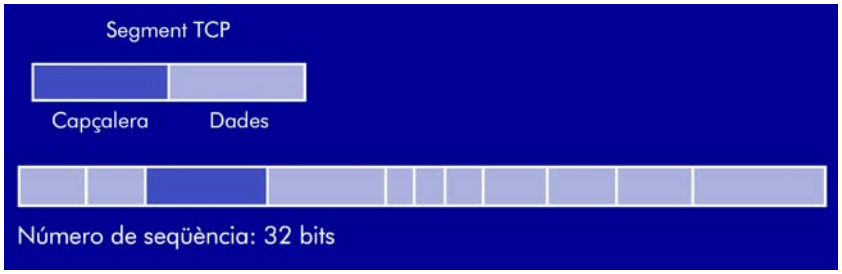
b) El camp **Port de destinació** identifica l'aplicació al terminal de destinació.

Figura 59.



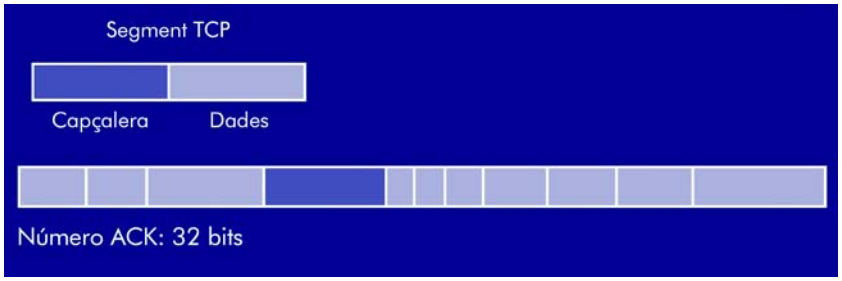
c) El camp **Número de seqüència** identifica el primer byte del camp de dades. En el TCP no es numeren segments, sinó bytes. Per tant, el número de seqüència identifica el primer byte de les dades que envia el segment: al principi de la connexió s'assigna un número de seqüència inicial (ISN, de l'anglès *initial sequence number*), a partir del qual el TCP numera els bytes consecutivament.

Figura 60.



d) El camp **Número ACK**. El TCP reconeix dades per mitjà de la tècnica de *piggybacking*. En activar un bit de la capçalera (el bit ACK), el TCP té en compte el número de seqüència ACK que indica a l'altre extrem TCP el pròxim byte que està disposat a rebre. Dit d'una altra manera, el número ACK menys un indica l'últim byte reconegut.

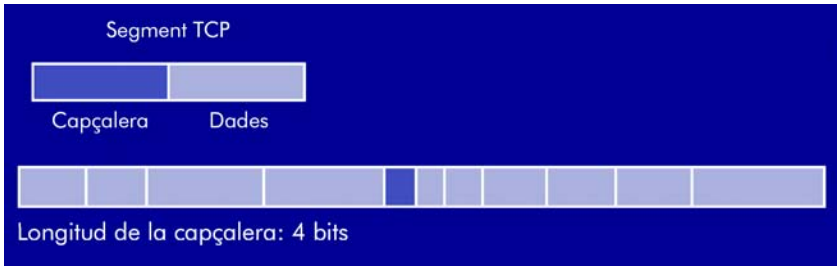
Figura 61.



ANOTACIONS

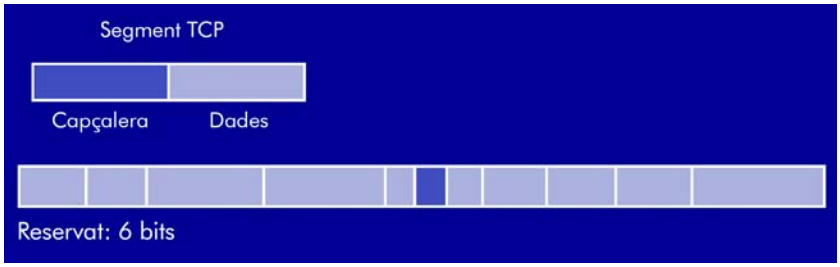
e) El camp **Longitud de la capçalera** indica la longitud de la capçalera, que pot ser variable. La longitud típica és de 20 bytes; tot i que, si el TCP utilitza el camp d'opcions, pot arribar a una longitud màxima de 60 bytes. D'aquesta manera, el TCP sap on comencen les dades.

Figura 62.



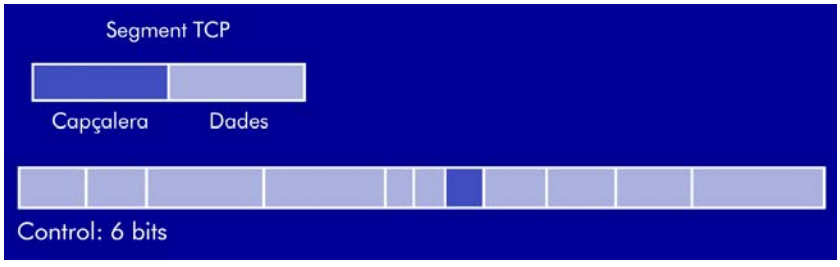
f) El camp **Reservat**, tal com el seu nom indica, és reservat i s'inicialitza amb zeros.

Figura 63.



g) El camp **Control** està format per sis indicadors independents, cada un dels quals assenyala una funció específica del protocol quan està actiu (a 1):

Figura 64.



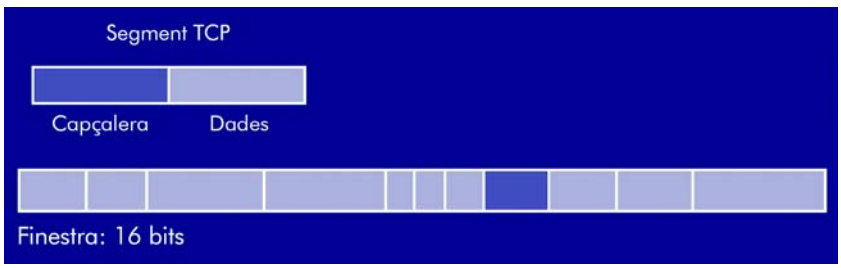
- **URG**: indica que hi ha dades urgents (i el camp *Urgent pointer* indica la quantitat de dades urgents existents en el segment).

ANOTACIONS

- **ACK**: quan aquest bit está actiu, el camp Número ACK indica el byte següent que espera rebre la connexió TCP. Si no està actiu, el camp Número ACK no té cap significat per al TCP.
- **PSH**: invoca la funció *push* en el protocol. Aquesta funció diu al receptor que lliuri a l'aplicació totes les dades que tingui disponibles en la memòria intermèdia de recepció sense esperar a completar-les amb dades addicionals. D'aquesta manera, les dades no esperen en la memòria intermèdia receptora fins a completar un segment de dimensió màxima.
- **RST**: reinicialitza la connexió.
- **SYN**: s'utilitza per a iniciar una connexió i també serveix per a resincronitzar els números de seqüència.
- **FIN**: indica que el transmissor ha acabat la connexió.

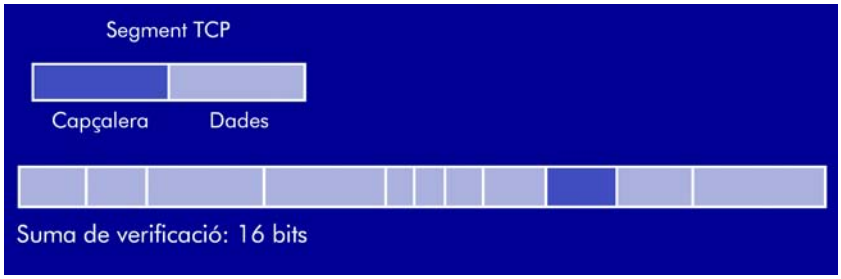
h) El camp **Finestra** indica quants bytes componen la finestra de transmissió del protocol de control de flux per finestra lliscant. A diferència dels protocols del nivell d'enllaç, que la finestra era constant i comptava trames, en el TCP la finestra és variable i compta bytes. Amb cada segment transmès, un extrem TCP anuncia a l'altre extrem la quantitat de dades que està disposat a rebre en cada moment. D'aquesta manera, l'extrem que rep un segment actualitza la mida de la seva finestra de transmissió.

Figura 65.



i) El camp **Suma de verificació** s'utilitza per a detectar errors.

Figura 66.



Nota

No s'ha de confondre PSH amb l'indicador URG, que indica que l'aplicació ha assenyalat una porció del segment com a urgent.

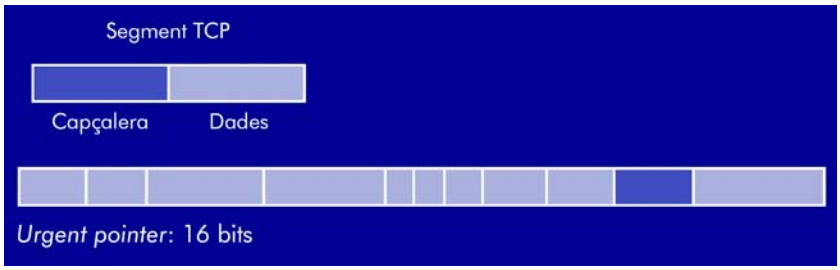
ANOTACIONS

j) El camp ***Urgent pointer*** té sentit quan el bit de control URG està actiu. Indica que les dades que envia l'origen són urgents i identifica l'últim byte d'aquest camp. L'aplicació és la que indica que aquests últims són urgents i ho sap perquè el TCP li ho indica en la recepció.

Exemple

Si el número de seqüència indica 1.000 i l'*Urgent pointer* indica 200, significa que els bytes del 1.000 al 1.200 es consideren urgents. A partir del byte 1.201 les dades es tornen a considerar normals.

Figura 67.

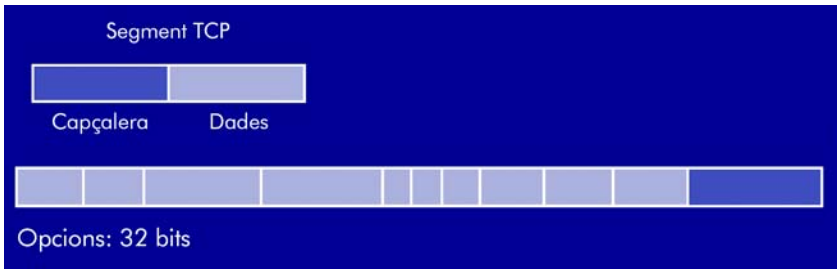


Nota

Algunes aplicacions que utilitzen l'*Urgent pointer* són, per exemple `telnet`, `rlogin` o `ftp`. A la llibreria de *sockets*, el trànsit urgent es denomina *trànsit fora de banda* (*out of band*).

k) El camp **Opcions TCP** permet afegir camps a la capçalera per a realitzar les operacions següents:

Figura 68.



- Marcar el temps (*timestamp*) en què es va transmetre el segment i, d'aquesta manera, poder monitoritzar els retards que experimenten els segments des de l'origen fins a la destinació.
- Augmentar la mida de la finestra.
- Indicar la mida màxima del segment (MSS, de l'anglès *maximum segment size*) que una estació està preparada per a rebre. Per tant, l'altre extrem no li pot transmetre segments per sobre d'aquest valor.

Nota

La mida màxima del segment TCP transmès s'especifica durant l'establiment de la connexió i defineix la màxima longitud de dades que enviarà el TCP.

Activitat

Quina és la mida d'un datagrama IP en funció d'MSS?

Solució

Si la mida de les dades TCP és MSS, serà necessari afegir-hi 20 bytes de la capçalera TCP més 20 bytes de la capçalera IP (tenint en compte les capçaleres bàsiques sense opcions). Això significa que la longitud del datagrama IP serà d'MSS + 40 bytes (sempre assumint que tant el TCP com l'IP no utilitzen els seus camps d'opcions).

Si no s'especifica la mida màxima durant la transmissió del segment SYN, es prenen per defecte 536 bytes (la mida per defecte d'un datagrama IP és de 576 bytes, menys els 40 bytes de les capçaleres IP i TCP).

Nota

Consulteu l'MTU en l'apartat 11.1.2.

El fet d'escollir l'MSS no és trivial. En general, com més gran sigui l'MSS, millor, ja que les capçaleres IP i TCP s'amortitzen més; si l'MTU és petita, caldrà fragmentar el datagrama IP (és a dir, el segment TCP); per tant, per norma general no interessa triar MSS més grans que l'MTU. En aquest cas, hi ha diferents possibilitats:

1. Buscar l'MTU local de la xarxa a la qual està connectada l'estació i, si hi ha MTU més petites fins a la destinació, hi haurà fragmentació.

2. Utilitzar *MTU discovery path*, un mecanisme de recerca per a esbrinar quina és l'MTU més petita des de l'origen fins a la destinació i utilitzar com a MSS aquesta mida menys els 40 bytes de capçaleres IP i TCP.

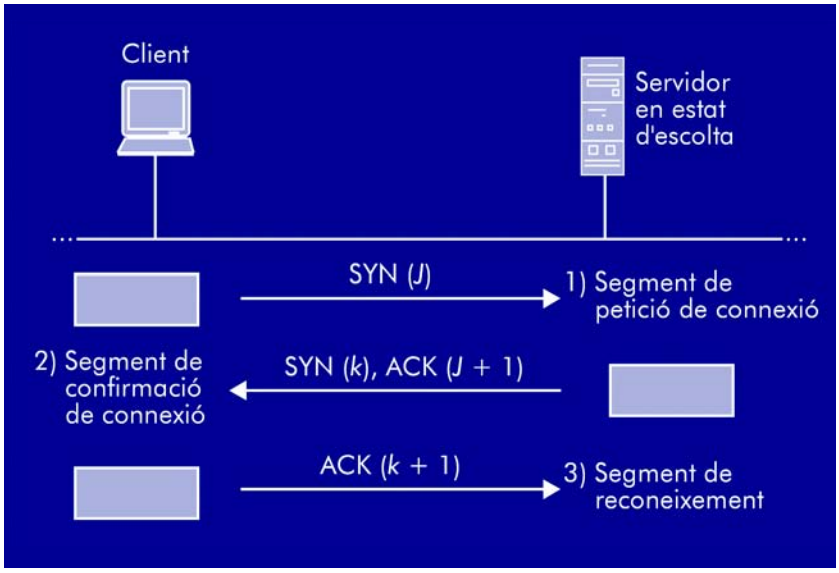
14.3. Establiment de la connexió

Per a establir una connexió, el TCP utilitza el **protocol *three-way handshake***, que necessita tres segments TCP per a poder establir la connexió.

ANOTACIONS

Considerem que el servidor està en un estat d'escolta, anomenat *listen*, i que el client vol establir una connexió amb el servidor. El TCP de la màquina client iniciarà la petició de connexió TCP, que contestarà el TCP de la màquina servidor.

Figura 69.



Perquè el client TCP pugui establir una connexió TCP amb el servidor, se segueixen aquests passos:

Nota

El segment SYN especifica més paràmetres, com ara el port del servidor a què es vol connectar el client, i sol especificar també la mida màxima del segment (MSS) que el client transmetrà.

1. **Petició de la connexió**

El TCP client envia un segment de petició de connexió al servidor. Aquest segment, que es coneix com a *segment SYN* perquè té activat el bit SYN en el camp Control de la capçalera del segment TCP, especifica el número de seqüència inicial TCP del client (ISN).

El número de seqüència inicial es tria a l'atzar. La raó és molt senzilla. Hi ha paquets que poden sobreviure a la xarxa una vegada s'ha tancat la connexió TCP (fins i tot si ha estat a causa d'una caiguda del sistema). Cal assegurar-se que una connexió nova escull un número de seqüència inicial que no existeixi. El TCP recomana utilitzar un número de seqüència inicial basat en una variable que s'incrementa una quantitat *x* cada *y* temps (per exemple, en 4.4 BSD hi ha un comptador que s'incrementa cada 8 ms).

ANOTACIONS

Si el sistema cau, passats uns quants segons torna a funcionar i immediatament s'estableix una connexió nova utilitzant el mateix port i la mateixa adreça IP, es podria interpretar que els segments TCP que han quedat retardats a la xarxa i que ja existien anteriorment a la caiguda de la màquina, pertanyen a la connexió nova, la qual cosa provocaria la confusió i el mal funcionament d'aquesta connexió. Això succeiria fins i tot amb independència del número de seqüència inicial elegit.

Amb l'objectiu de protegir-se d'aquesta situació, es combinen dues tècniques: una consisteix a triar el número de seqüència inicial de manera aleatòria i l'altra és l'anomenat *quiet time*, que consisteix en el fet que el TCP no creï cap connexió nova després d'un rebot de màquines fins que no transcorri un temps determinat denominat ***MSL*** (de l'anglès *maximum segment lifetime*: 'temps màxim de vida d'un segment'). D'aquesta manera, s'assegura que no rebrà segments antics d'altres connexions.

Nota

L'MSL depèn de la implementació; els valors normals són, aproximadament, de trenta segons, un minut o dos minuts. No obstant això, hi ha moltes implementacions que no tenen en compte aquesta situació, ja que consideren que un rebot de màquines dura més temps que l'MSL.

2. Confirmació de la connexió

El servidor respon a la petició d'establiment de la connexió amb un segment SYN que indica el número de seqüència inicial que utilitzarà.

Així mateix, aquest segment conté un reconeixement (ACK) del segment SYN del client que indica l'ISN del client més 1 (el número de seqüència inicial del client més 1).

Nota

Convé recordar que el TCP numera els ACK amb el número de seqüència del pròxim byte que espera rebre (en aquest cas, el servidor espera que el pròxim byte

ANOTACIONS

enviat pel client sigui $J + 1$). En la figura anterior, seria el segment SYN (K), ACK ($J + 1$), on K és l'ISN elegit pel TCP servidor.

3. **Reconeixement de la connexió**

El client reconeix el segment SYN (K) del servidor amb un reconeixement que conté l'ISN servidor més 1. En la figura anterior, seria el segment ACK ($K + 1$).

Es diu que qui envia el primer segment SYN (en aquest cas, el client) efectua una obertura activa (*active open*), mentre que qui rep el primer segment SYN i n'envia el següent (en aquest cas, el servidor) duu a terme una obertura passiva (*passive open*).

Es pot donar el cas que tots dos extrems efectuïn una obertura activa en el mateix moment. Aquesta situació es denomina *obertura simultània* (*simultaneous open*).

Després d'aquests tres passos, podem dir que ja s'ha establert la connexió entre el client i el servidor.

Nota

Utilitzarem el programa `tcpdump` per a veure com funciona el protocol d'establiment d'una connexió.

Assumim que ens hem connectat a una màquina anomenada *argos* i fem un `rlogin` a la màquina *helios*

```
argos % rlogin helios
```

Nota

En l'annex 2 podeu trobar una descripció del programa `tcpdump`.

Les primeres línies que obtenim amb el **tcpdump** són les següents:

- `15:56:54.796091 argos.1023 > helios.login: S 3541904332: 3541904332 (0) win 31744 <mss 1460>`
- `15:56:54.796091 helios.login > argos.1023: S 548133143: 548133143 (0) ack 33541904333 win 8760 <mss 1460>`
- `15:56:54.796091 argos.1023 > helios.login: . ack 548133144 win 31744`

ANOTACIONS

Interpretació

1. *argos*, des del port 1.023, envia a *helios* una petició de connexió per mitjà d'un segment SYN. El número de seqüència inicial (ISN) elegit per *argos* és el 3.541.904.332, i *argos* anuncia que pot rebre 31.744 bytes sense reconèixer-los i que espera rebre segments amb una mida màxima de 1.460 bytes.

2. *helios* respon amb un segment SYN, tria com a ISN el número 548.133.143 i respon amb un ACK amb el byte següent que espera rebre d'*argos*, el 3.541.904.333 (3.541.904.332 + 1). Anuncia que pot rebre 8.760 bytes i que espera rebre segments amb una mida màxima de 1.460 bytes.

3. *argos* reconeix el segment SYN amb un segment en el qual espera rebre el byte 548.133.144 (548.133.143 + 1), *argos* torna a advertir que està disposat a rebre fins a 31.744 bytes.

A continuació, començaria l'intercanvi d'informació entre el client i el servidor (per exemple peticions de connexió, contrasenya, *prompt* de la màquina, etc.).

Activitat

Utilitzeu el programa `tcpdump` per a veure l'establiment d'una connexió. Amb aquesta finalitat, estabiu una connexió amb aplicacions diferents (`telnet`, `ftp`, `rlogin`, etc.) i monitoreu la connexió. Observeu els segments d'inici de la connexió, el valor del número de seqüència inicial, el del número ACK inicial i la mida de la finestra.

14.4. Acabament de la connexió

Quan la transferència de la informació ha finalitzat, el TCP disposa d'un protocol d'acabament de la connexió per a tancar-la.

ANOTACIONS

En una connexió TCP *full duplex*, en la qual les dades flueixen en tots dos sentits, independents l'un de l'altre, qualsevol connexió s'ha de tancar independentment.

Cal tenir en compte que tant el client com el servidor poden tancar la connexió. La situació normal és que l'aplicació client iniciï la petició de connexió i tingui, possiblement, un usuari interactiu que li'n demani el tancament per mitjà, per exemple, d'una instrucció, que en Telnet seria `logout` i en un FTP seria `bye`. Per tant, suposem que és el client qui demana tancar la connexió (si fos el servidor, seria similar). Els passos que se segueixen per a tancar una connexió són els següents:

1. El client envia un segment TCP del tipus FIN amb el número de seqüència corresponent (*J*). A partir d'aquest moment no hi haurà més dades que flueixin en aquest sentit (client → servidor).
2. El servidor envia una confirmació del tancament per mitjà d'un ACK amb el número de seqüència rebut més 1 (*J* + 1).

Nota

El segment FIN rep aquest nom perquè té activat el bit FIN en el camp Control de la capçalera del segment TCP.

El TCP servidor indica a la seva aplicació que el client tanca la connexió. L'aplicació servidor indica al seu TCP que la tanqui a continuació.

3. El servidor envia un segment TCP del tipus FIN al client amb el número de seqüència corresponent (*K*).
4. El TCP client respon automàticament amb un ACK (*K* + 1).

Es diu que qui envia el primer segment FIN (en aquest cas el client) porta a terme un tancament actiu (*active close*), mentre que qui el rep (en aquest cas el servidor) realitza un tancament passiu (*passive close*).

La connexió que efectua el tancament actiu entra en un estat denominat *TIME_WAIT*, de manera que haurà d'esperar un temps (per norma general, una o dues vegades l'MSL) abans d'utilitzar de nou el mateix port. El més habitual és que sigui el client qui efectuï el tancament actiu. Com que els clients solen utilitzar ports locals efímers, aquest temps d'espera no els afecta. En canvi, si és el servidor que

ANOTACIONS

efectua el tancament actiu, ens podem trobar que no es pugui reinicialitzar durant un o dos minuts. Això succeeix perquè el servidor utilitza ports coneguts que no es poden tornar a reassignar fins que no acabi el procediment *quiet time* i se surti de l'estat *TIME_WAIT*.

Lectura complementària

Podeu trobar una secció dedicada a aquest tema en el llibre següent:

W.R. Stevens (1998). *TCP/IP Illustrated* (vol. 1: "The Protocols", cap. 19, pàg. 252). Wilmington: Addison-Wesley, 1994.

És possible que només tanqui la connexió (sortida de dades) un dels extrems, mentre que l'altre (recepció) es manté obert. Aquesta situació es denomina *half-close*, però hi ha poques aplicacions que se n'aprofitin. El més normal és que totes dues aplicacions tanquin els seus canals de comunicacions. Així mateix, es pot donar el cas que els dos extrems efectuïn un tancament actiu. Aquesta situació es denomina *tancament simultani* (*simultaneous close*).

Monitoratge de l'acabament d'una connexió utilitzant el programa `tcpdump`

Utilitzarem el programa `tcpdump` per a veure com funciona el protocol d'acabament de la connexió. Assumim que en el rlogin de l'exemple d'establiment de la connexió *helios* fa un `logout` (demana el tancament de la connexió).

```
helios % logout
```

Les línies que obtenim amb el `tcpdump` són les següents:

```
15:57:01.616091 helios.login > argos.1023: F 1417: 1417 (0)
ack 41 win 8760

15:57:01.616091 argos.1023 > helios.login: .ack 1418 win
31744

15:57:01.616091 argos.1023 > helis.login: F 41:41 (0) ack
580 31744

15:57:01.616091 helios.login > argos.1023: .ack 42 win 8760
```

Nota

La notació dels números de seqüència i números ACK s'estableix a partir de l'ISN; és a dir, un número de seqüència 1.417 indica un número de seqüència ISN + 1.417.

Interpretació

1. *helios* envia un segment amb l'indicador F (FIN). El número de seqüència és el 1.417 i envia 0 bytes de dades. Espera rebre el byte 41 d'*argos* i anuncia una finestra de 8.760 bytes.

2. *argos* envia un reconeixement per mitjà d'un segment amb ACK 1.418 (1.417 + 1) i anuncia una finestra de 31.744 bytes.

3. Ara li toca a *argos* tancar la seva connexió TCP. Amb aquesta finalitat, envia un segment amb l'indicador F (FIN) a *helios*. El número de seqüència és el 41 (és el que esperarà *helios*) i envia 0 bytes de dades. Anuncia una finestra de 31.744 bytes.

4. *helios* rep el segment, el reconeix amb l'ACK numerat com a 42 (41 + 1) i anuncia una finestra de 8.760 bytes.

helios ha efectuat un tancament actiu, mentre que *argos* ha efectuat un tancament passiu.

Activitat

Utilitzeu el programa `tcpdump` per a veure el tancament d'una connexió. Amb aquesta finalitat, establiu una connexió amb diferents aplicacions (`telnet`, `rlogin`, etc.) i superviseu-la.

14.5. Diagrama d'estats del TCP

En el diagrama d'estats del TCP es descriuen els diferents estats pels quals passa una connexió des del seu establiment fins al seu acabament, incloent-hi l'etapa de transferència de la informació. Els noms dels estats TCP són els mateixos que es poden consultar amb la crida al sistema *netstat*.

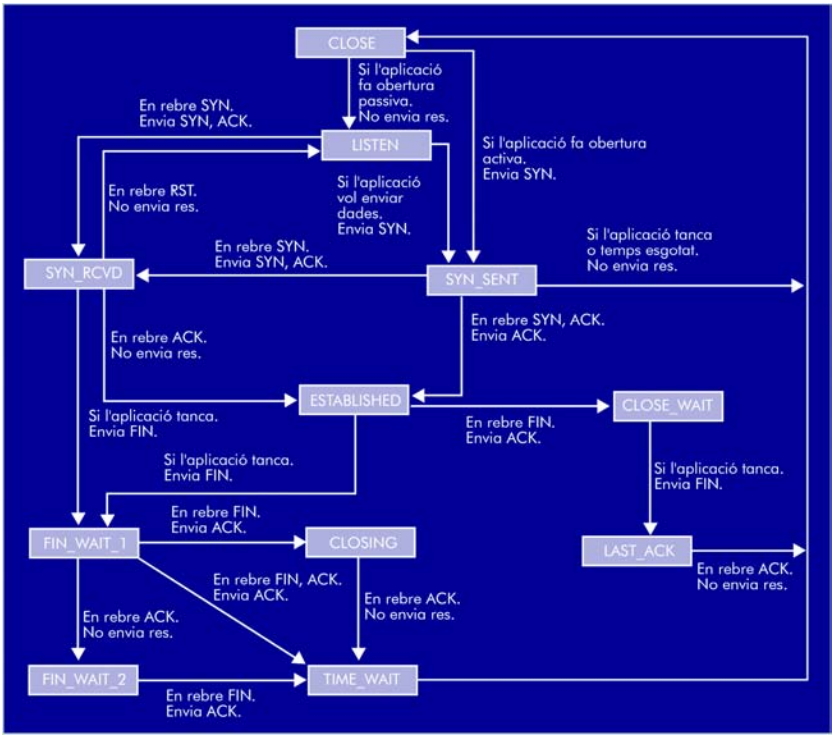
Lectura complementària

W.R. **Stevens** (1998). *TCP/IP Illustrated* (vol. 1, "The Protocols"). Wilmington: Addison-Wesley, 1994.

L'estat ESTABLISHED es correspon amb la transferència de la informació. La resta dels estats es corresponen amb l'establiment i l'acabament de la connexió, tenint en compte totes les maneres possibles d'establir i tancar una connexió en el TCP. Els símbols SYN, RST, FIN i ACK es corresponen amb els bits d'indicació de la capçalera TCP.

ANOTACIONS

Figura 70.



Un exemple de com s'interpreta aquest diagrama seria el protocol d'acabament d'una connexió, en la qual ja hem vist que l'extrem TCP que demana el tancament efectua un tancament actiu. Això significa que passaria de l'estat ESTABLISHED a l'estat FIN_WAIT_1 enviant un segment FIN.

Des d'aquí pot passar a un dels tres estats que descriuen com es pot fer un tancament actiu depenent de com es tanqui la connexió:

- Amb un tancament simultani (*simultaneous close*), passa a CLOSING.
- Amb la recepció d'un ACK, passa a FIN_WAIT_2, on espera rebre un FIN.
- Amb la recepció d'un FIN i un ACK, passa a TIME_WAIT, on espera dues vegades l'MSL abans d'alliberar el port.

Ja hem vist que l'extrem TCP que rep un FIN duu a terme un tancament passiu. Per tant, passa de l'estat ESTABLISHED a l'estat CLOSE_WAIT enviant els indicadors ACK i FIN corresponents per a acabar la connexió.

Nota

Consulteu l'MSL en l'apartat 15.3 d'aquesta unitat.

ANOTACIONS

La fase d'establiment també es pot seguir amb facilitat per mitjà del diagrama d'estats, tenint en compte quin extrem efectua el tancament actiu i quin, el tancament passiu.

Activitats

- Utilitzeu la comanda `netstat` per a veure l'estat de les connexions TCP que tingueu en aquest moment. Si no teniu cap aplicació a la xarxa, connecteu-vos a algun servidor amb la web feu un `ftp` o un `telnet` a alguna màquina.
- Suposeu una interacció entre un client i un servidor en la qual el client faci una obertura activa i el servidor, una obertura passiva; a continuació, s'intercanvien un segment de dades, amb els seus corresponents reconeixements, i finalitzen amb un tancament actiu per part del servidor, passiu per part del client. Dibuixeu el diagrama de temps que mostri l'intercanvi de segments i, en paral·lel, els estats on es troben les dues entitats en cada moment.

14.6. Transferència de la informació

Una vegada establerta la connexió, el TCP pot començar la transferència de segments TCP en tots dos sentits. Per a transmetre informació de manera fiable, el TCP implementa protocols de control d'errors i de flux. Els passos que segueix el TCP per a transferir la informació són els següents:

1. Quan el TCP envia dades, manté un **temporitzador** (***timeout***) fins que rep un reconeixement (ACK) del receptor. Si el temporitzador salta, el TCP retransmet les dades.
2. Quan el TCP rep un segment de dades, envia un reconeixement. Aquest últim es pot retornar retardat (no tot seguit) si el TCP ho considera necessari.
3. Si un segment rebut és incorrecte (el *checksum* ho indica), el TCP el descarta i no ha d'enviar la informació. De fet, com que el

ANOTACIONS

TCP utilitza la tècnica de *piggybacking* (aprofita els segments de dades que viatgen en sentit contrari), el que fa és retornar un segment amb el mateix número d'ACK que havia reconegut l'última vegada. El transmissor veurà un ACK amb un número repetit i interpretarà que no li reconeixen la informació. Aquest número es denomina *ACK duplicat*.

En cas que no tingués dades per a enviar en sentit contrari, el TCP pot enviar un segment que no contingui informació (amb un camp de dades de zero bytes). Aquest segment tindria l'indicador ACK activat i reconeixeria els bytes pertinents en el camp Número d'ACK. El número de seqüència no s'hauria incrementat, ja que no envia dades.

4. Si els segments arriben desordenats (per sota hi ha l'IP, no orientat a la connexió), el TCP reordena els segments i passa les dades (bytes) correctament ordenades a l'aplicació. Si rep segments duplicats, el TCP descarta les còpies.

5. Com que el TCP té una memòria limitada, ha d'efectuar un control de flux. Amb aquesta finalitat, cada extrem avisa de les dades que està disposat a rebre en cada moment utilitzant el camp Finestra (es tracta d'un mecanisme de finestra lliscant basat en bytes que explicarem més endavant).

Exemple

- Dades interactives: les que transmeten aplicacions com ara `telnet` o `rlogin`.
- *Bulk data*: les que transmeten aplicacions com ara correu electrònic o `ftp`.

El tipus d'informació que s'envia es pot dividir en dades interactives i dades de gran volum o *bulk data*. La diferència entre ells resideix en la quantitat d'informació que es transmet. Les dades interactives transmeten pocs bytes d'informació (al voltant de 10), mentre que les dades de gran volum transmeten gran quantitat de dades (ocupen la totalitat de la mida del segment TCP). Convé considerar que no és el mateix carregar la xarxa amb paquets petits d'informació que amb paquets grans. El TCP pot aplicar en cada cas tècniques diferents de manera automàtica per a aprofitar la xarxa al màxim.

14.6.1. Transmissió de dades interactives

En una transmissió de dades interactives és normal enviar poques dades. En una aplicació del tipus `telnet`, per exemple, un usuari cli-

ANOTACIONS

ent podria executar l'ordre de Unix `ls` i obtenir una llista d'un directori per part del servidor. En aquesta transferència d'informació intervenen pocs bytes des de l'origen (client) fins a la destinació (servidor) i s'utilitzen conjuntament dues tècniques per a obtenir un millor aprofitament de la xarxa:

- Reconeixements retardats.
- Algorisme de Nagle.

Reconeixements retardats

En aquest tipus de transferència, és normal que el TCP no enviï els reconeixements ACK immediatament després de rebre les dades, sinó que estigui esperant un temps que hi hagi dades per a enviar en sentit contrari. D'aquesta manera, pot utilitzar la tècnica *piggybacking* i enviar el reconeixement encapsulat en les dades que retornen al client.

És possible que el servidor s'estalviï enviar un segment que només reconeix, però que no conté dades. És típic que el TCP esperi (utilitza un temporitzador) uns 200 ms per si hi ha dades per a transmetre abans d'enviar l'ACK. Una vegada ha transcorregut aquest temps, el TCP reconeix les dades rebudes fins al moment amb un segment de dades, si disposa de dades per a enviar en sentit contrari (*piggybacking*), o amb un segment sense dades (el número de seqüència no varia). En qualsevol dels dos casos, l'indicador ACK estarà activat i el número ACK reconeixerà les dades pertinents.

Algorisme de Nagle

Moltes vegades un client té molt poques dades per a enviar (per exemple, només 1 byte). En aquest cas el TCP enviaria un segment només amb 1 byte de dades i amb 20 bytes de capçalera TCP. L'IP afegiria 20 bytes més de capçalera, la qual cosa proporciona un total de 40 bytes de control i 1 de dades. Si es transmeten molts segments d'aquest tipus, l'eficiència de la transmissió és molt baixa. Una solució a aquesta baixa eficiència de transmissió és aplicar l'algorisme de Nagle.

ANOTACIONS

Utilitzant l'algorisme de Nagle, una connexió TCP només pot tenir un segment de mida petita (pocs bytes) sense que s'hagi reconegut; és a dir, només hi pot haver un únic segment de mida petita viatjant per la xarxa (en vol). La resta dels segments de mida petita no es poden transmetre fins que l'ACK del segment petit que viatgi per la xarxa hagi arribat.

Així, els segments que esperen per a ser transmesos s'emmagatzemen fins que es rep l'ACK del segment en vol. Quan aquest últim arriba, la connexió TCP pot enviar un segment que contingui totes les dades emmagatzemades fins a aquest moment, formant un segment més gran.

L'algorisme de Nagle funciona quan els retards a la xarxa són grans; és a dir, si la connexió creua una WAN. En cas que la connexió sigui local, en una LAN, és difícil que s'apliqui aquest algorisme a causa de l'alta velocitat de la xarxa.

Nota

A la llibreria de *sockets* API, l'indicador que desinhibeix l'algorisme de Nagle és el TCP_NODELAY.

A vegades, és interessant desinhibir l'algorisme de Nagle, ja que l'aplicació no pot esperar. El moviment del ratolí en *X Windows System* provoca segments petits. Aquests moviments del ratolí s'han de lliurar sense retards perquè l'usuari interactiu no ho noti. Les llibreries de *sockets* han de permetre, activant indicadors, desinhibir l'algorisme de Nagle.

14.6.2. Transmissió de dades de gran volum. Control de flux per finestra lliscant

En les comunicacions en què s'envia una ingent quantitat de dades de gran volum (correu electrònic, transferències FTP, etc.), com que les memòries intermèdies de recepció es poden omplir, és necessari un protocol de finestra lliscant (*sliding window*) per a controlar el flux de dades, amb la diferència, respecte dels protocols del nivell d'enllaç, que en el TCP la finestra de transmissió és variable.

La idea és que cada extrem TCP reguli la quantitat de dades que l'altre extrem pot transmetre. Amb aquesta finalitat, cada extrem TCP notifica a l'extrem oposat, cada vegada que envia un segment, la finestra que pot acceptar en aquest moment. El TCP oposat actualitza la seva finestra de transmissió d'acord amb aquest valor.

ANOTACIONS

Mentre que el TCP transmissor marca els bytes que ha transmès amb un número de seqüència, el TCP receptor agafa els bytes que rep i els reconeix amb un ACK. Els reconeixements ACK especifiquen sempre el número de seqüència del pròxim byte que el receptor espera rebre.

En el TCP es reconeixen posicions de bytes en el flux de dades fins a l'última posició que ha rebut correctament, sense tenir en compte el segment al qual pertanyen.

El TCP només activa un temporitzador de retransmissions que reprograma quan rep un reconeixement o quan salta el temporitzador. Més endavant veurem com el TCP programa el temporitzador de retransmissions. La capçalera del segment TCP especifica tres paràmetres essencials en el funcionament del protocol de finestra lliscant:

- El **número de seqüència**, que indica a la seva connexió oposada el primer byte de dades que conté el segment transmès.
- El **número de reconeixement (número ACK)**, que indica a la seva connexió oposada el pròxim byte que espera rebre i, per tant, l'últim byte rebut correctament.
- La **finestra**, que indica a la seva connexió oposada la mida de la memòria intermèdia de recepció i, per tant, la mida de la finestra que el transmissor ha d'utilitzar.

Nota

Recordeu que el TCP és bidireccional i que un segment TCP reconeix, per mitjà de *piggybacking*, les dades que rep amb un ACK que ha d'estar numerat.

Activitat

Assumim que un extrem client TCP ha elegit el 28.325 com a número de seqüència inicial (ISN), mentre que l'extrem servidor TCP ha elegit com a ISN el 12.555. Què indica un segment client TCP amb número de seqüència 29.201, número ACK 12.655 i finestra 1.024?

Solució

El número de seqüència indica que el client ja ha transmès des del byte 28.325 fins al byte 29.200 (875 bytes en total) i que en aquest segment transmetrà a partir del byte 29.201. El número ACK indicarà el servidor

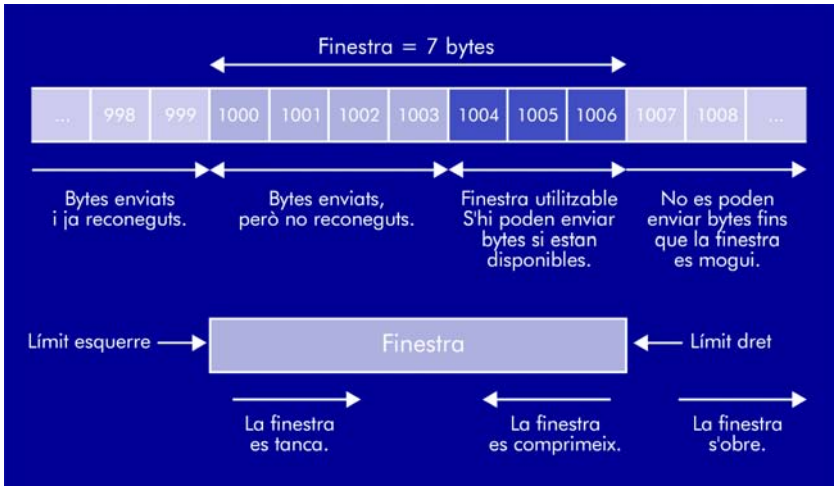
ANOTACIONS

que el client ha rebut correctament fins al byte 12.654 i que espera rebre a partir del 12.655. La finestra indica al servidor que el client només pot acceptar 1.024 bytes abans de confirmar-los. Per tant, el servidor TCP actualitzarà la seva finestra de transmissió a 1.024.

Amb l'objectiu d'estudiar el mecanisme de finestra lliscant, analitzarem un cas senzill. Assumirem que ja s'ha establert la connexió i s'ha assignat l'ISN per a tots dos extrems.

En la figura següent, podem veure com funcionaria el protocol de finestra lliscant per al TCP transmissor. El TCP receptor li ha indicat que està disposat a rebre 7 bytes. Per tant, la finestra de transmissió de està transmissor és de 7 bytes:

Figura 71.



Podem interpretar la finestra lliscant de la manera següent:

1. El TCP ha enviat bytes fins al número de seqüència 1.003. D'aquests bytes, el TCP receptor li n'ha reconegut fins al 999; falten per a conèixer els bytes del 1.000 al 1.003.
2. Com que la finestra de transmissió és de 7 bytes i ja n'ha transmès 4, el TCP encara pot transmetre 3 bytes abans d'esgotar-la (bytes 1.004 a 1.006).

ANOTACIONS

3. El TCP només podrà transmetre del byte 1.007 en endavant en els casos següents:

- Si el TCP receptor li reconeix els bytes a partir del 1.000, de manera que el límit esquerre de la finestra es mourà cap a la dreta.
- Si el TCP receptor li anuncia una finestra superior a 7, de manera que el límit dret de la finestra es mourà cap a la dreta.
- Una combinació dels dos casos anteriors.

Com podeu observar, el TCP receptor pot advertir una nova finestra de transmissió. Cada vegada que reconegui dades, avisarà de la nova finestra que està disposada a rebre. El TCP transmissor l'actualitzarà.

La finestra pot experimentar tres tipus de moviment:

1. La **finestra es tanca** en moure's el límit esquerre cap a la dreta quan les dades enviades són reconegudes.
2. La **finestra s'obre** en moure's el límit dret cap a la dreta i això permet que el TCP enviï més dades. Aquesta obertura té lloc quan el receptor allibera espai de la seva memòria i pot advertir una finestra nova.
3. La **finestra es comprimeix** en moure's el límit dret cap a l'esquerra.

Alguns punts que podem resumir de la figura de la finestra lliscant són els següents:

- Si el límit esquerre assoleix el límit dret, es diu que la finestra val zero (*zero window*). Això fa que el transmissor aturi la tramesa de dades.
- Es recomana que el TCP transmissor no comprimeixi la finestra de transmissió.

ANOTACIONS

- Cal diferenciar el fet que la finestra es comprimeixi (el límit dret es mou cap a l'esquerra) del fet que la finestra disminueixi de mida (s'anuncia una finestra més petita, però el límit dret no es mou cap a l'esquerra).

> ***Nota***
>
> Suposem una finestra de 7 bytes com en la figura de la finestra lliscant. El receptor reconeix els bytes del 1.000 al 1.003 i anuncia una finestra de 5 bytes. Com podeu deduir, el límit esquerre val ara 1.004; el límit dret, 1.008 (s'ha mogut cap a la dreta), i la nova finestra, 5. En aquest cas, la finestra de recepció s'ha reduït, però no s'ha comprimit.
>
> En canvi, si el receptor només reconeix 1 byte (el byte 1.000) i anuncia una finestra d'1 byte, el transmissor es trobarà amb un problema. Una finestra d'1 byte significa que només podia haver-ne transmès 1 (el 1.001), però ja n'havia transmès 3, incloent-hi el reconegut (del 1.000 al 1.003). Per tant, el receptor s'ha d'assegurar d'advertir almenys tants bytes com el transmissor pot haver enviat amb la finestra anterior. Si només reconeix 1 byte, la finestra advertida ha de ser de 6 bytes; si reconeix els 4 bytes, aquesta última ha de ser, almenys, de 3 bytes, ja que el transmissor ja els podria haver transmès.

Exemple

Utilitzarem el programa `tcpdump` per a observar com funciona el protocol de finestra lliscant. Assumim que hem efectuat un `rlogin` d'*argos* a *helios* (*argos* % *`rlogin helios`*) i ja estem connectats a *helios*. Una vegada ens trobem a *helios*, executem l'ordre `ls`. Aquesta última retorna per sortida estàndard la llista de directoris del directori de l'usuari (*home directory*) a *helios* que ocupa 811 caràcters (representa la tramesa de 811 bytes).

```
helios % ls
```

ANOTACIONS

Les línies que obtenim amb el programa `tcpdump` (numerades de l'1 al 13) són les següents:

```
1) 15:56:59.506091 argos.1023 > helios.login: P 37:38 (1)ack 596 win 31744
2) 15:56:59.516091 helios.login > argos.1023: P 596:597 (1) ack 38 win 8760
3) 15:56:59.526091 argos.1023 > helios.login: .ack 597 win 31744
4) 15:56:59.846091 argos.1023 > helios.login: P 38:39 (1)ack 597 win 31744
5) 15:56:59.856091 helios.login > argos.1023: P 597:600 (3) ack 39 win 8760
6) 15:56:59.866091 argos.1023 > helios.login: .ack 600 win 31744
7) 15:57:00.116091 argos.1023 > helios.login: P 39:40 (1)ack 600 win 31744
8) 15:57:00.126091 helios.login > argos.1023: P 600:603 (3) ack 40 win 8760
9) 15:57:00.136091 argos.1023 > helios.login: .ack 603 win 31744
10) 15:57:00.146091 helios.login > argos.1023: P 603:658 (55) ack 40 win 8760
11) 15:57:00.156091 argos.1023 > helios.login: .ack 658 win 31744
12) 15:57:00.166091 helios.login > argos.1023: P 658:1414 (756) ack 40 win 8760
13) 15:57:00.176091 argos.1023 > helios.login: .ack 1414 win 31744
```

La interpretació d'aquestes línies és la següent: *argos* ja ha enviat 36 bytes, mentre que *helios* ja n'ha enviat 595 (informació que tots dos han intercanviat des del principi de la connexió, com poden ser *logins*, *usernames*, etc.). Deduïm aquesta informació de la primera línia de l'exemple.

1. *argos* envia el caràcter 'l'. L'indicador P assenyala PUSH. El número de seqüència avança de 37 a 38.

2. *helios* retorna un eco del caràcter 'l'. El seu número de seqüència avança de 596 a 597 i reconeix el byte rebut (ACK = 37 + 1 = 38).

3. *argos* reconeix l'eco: ACK = 597 + 1 = 598.

4. *argos* envia el caràcter 'segon'. El número de seqüència avança de 38 a 39. L'ACK no reconeix res perquè val igual que abans: ACK = 597.

5. *helios* fa un eco que ocupa 3 bytes (BS +1 + s). El número de seqüència avança tres posicions (de 597 a 600) i reconeix el caràcter 'segon', ja que ACK = 38 + 1 = 39.

6. *argos* reconeix l'eco amb un ACK = 600.

7. *argos* envia el retorn de carro (CR). El número de seqüència avança una posició.

8. *helios* fa un eco del CR i, així mateix, retorna un altre CR seguit d'un LF. Això significa la tramesa de 3 bytes. Reconeix el CR, ja que ACK = 40.

9. *argos* reconeix aquests tres caràcters.

Nota

Recordeu que PUSH indica al receptor que passi les dades immediatament a l'aplicació; és a dir, que no les deixi durant un quant temps en la memòria intermèdia de recepció.

ANOTACIONS

10. *helios* respon a 'ls' enviant 55 dels 811 bytes que ha d'enviar. El número de seqüència avança de 603 a 658. L'ACK es manté a 40.

11. *argos* reconeix aquests 55 bytes enviant un ACK de 659.

12. *helios* transmet la resta dels 811 bytes, és a dir, 756 bytes.

13. *argos* reconeix aquests bytes avançant l'ACK a 1.414.

Com podem observar en aquest exemple, el TCP divideix la informació que cal enviar en dos segments: un segment de 55 bytes i un altre de 756 bytes. Convé remarcar que `rlogin` envia les ordres caràcter a caràcter i que, a més, l'aplicació remota fa un eco d'aquests caràcters. És per això que en les primeres línies s'envia primer l''l', després l''s', a continuació el retorn de carro, etc. El que ens interessa d'aquest exemple és veure com avancen les finestres en emetre i en reconèixer bytes. Per tant, no justificarem per què `rlogin` retorna ecos, ni per què afegeix un caràcter LF al retorn de carro.

14.6.3. Temporitzadors i retransmissions

El TCP activa fins a quatre temporitzadors de diferent tipus per a aconseguir un lliurament fiable de la informació. En aquest subapartat ens centrarem únicament en el **temporitzador de retransmissions** o **RTO** (de l'anglès, *retransmission time out*).

Convé recordar que tant els segments com els reconeixements es poden perdre durant la transmissió, de manera que cal utilitzar un temporitzador de retransmissions.

Com ja hem esmentat, el temporitzador es programa cada vegada que es rep un reconeixement, o bé quan salta perquè el reconeixement no ha arribat a temps (o, simplement, no ha arribat).

Definim el temps d'anada i tornada o RTT (de l'anglès, *round trip time*) com el temps que transcorre des que es

ANOTACIONS

transmet un segment fins que és reconegut (l'ACK torna al transmissor). L'RTT es pot mesurar restant l'instant en què el TCP transmissor emet el segment de l'instant en què rep l'ACK.

El més lògic seria activar el temporitzador de retransmissió en el temps d'anada i tornada (RTO = RTT). Tanmateix, és evident que els retards que experimenten els segments són variables: si s'activa l'RTO a l'RTT que ha experimentat el segment anterior, no es pot assegurar que aquest segment no tingui un retard superior i que el temporitzador no salti abans de temps. D'altra banda, si el temporitzador es programa a un valor molt més gran que RTT, cada vegada que hi hagi una pèrdua esperarem debades una bona estona, amb la pèrdua d'eficiència consegüent.

Hi ha diferents alternatives per a programar el temporitzador, totes orientades a trobar un valor adequat d'RTT; és a dir, que no sigui massa curt ni massa llarg.

Lectura complementària

Podeu trobar els diferents algorismes proposats per al càlcul d'RTT en l'obra següent:

W.R. Stevens (1994). *TCP/IP Illustrated* ("The protocols", vol. 1) Willmington: Addison-Wesley.

ANOTACIONS

IV. Aplicacions Internet

15. El model client/servidor

Les xarxes de computadors han fet aparèixer un concepte nou en el món de la programació: la **programació distribuïda**. Amb aquesta última es pretén aprofitar la potència i els recursos dels ordinadors interconnectats per a dur a terme una tasca de forma cooperativa.

Una aplicació distribuïda està formada per diversos programes que s'executen en ordinadors diferents i que es comuniquen per mitjà de la xarxa que uneix els ordinadors.

Convé destacar que cada programa, per si sol, no pot fer res. És necessària la **col·laboració** de tots perquè l'aplicació en conjunt produeixi resultats útils.

La cooperació dels diferents trossos de codi que formen l'aplicació ha de seguir un **protocol**. Aquest últim es pot elaborar a mida per a cada aplicació que es desenvolupi, o es pot definir un estàndard que tothom pugui seguir i així estalviar-se, d'aquesta manera, els dissenys particulars. Així mateix, l'existència d'un protocol estàndard garanteix la possibilitat d'interactuar amb productes de diferents fabricants.

Hi ha diversos estàndards d'aplicacions distribuïdes; tanmateix, sense cap dubte, el que ha tingut més èxit és el que segueix el model client/servidor.

En el model client/servidor, l'aplicació es divideix en dues parts, amb dos rols clarament diferenciats:

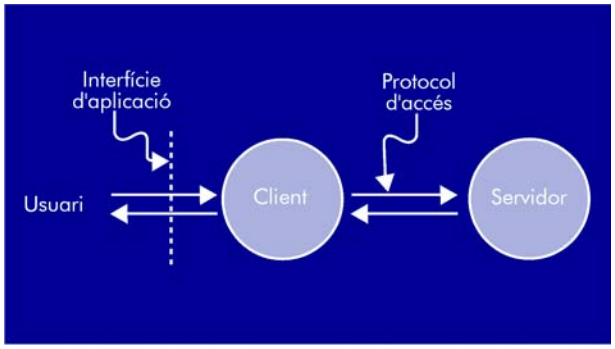
Servidor: ofereix un servei que pot ser l'accés a un recurs, l'execució d'operacions matemàtiques complexes, processament de dades, etc.

ANOTACIONS

Client: realitza una petició al servidor i n'espera un resultat.

Per norma general (almenys és el que diu el model), els usuaris accedeixen a l'aplicació per mitjà del client.

Figura 72.

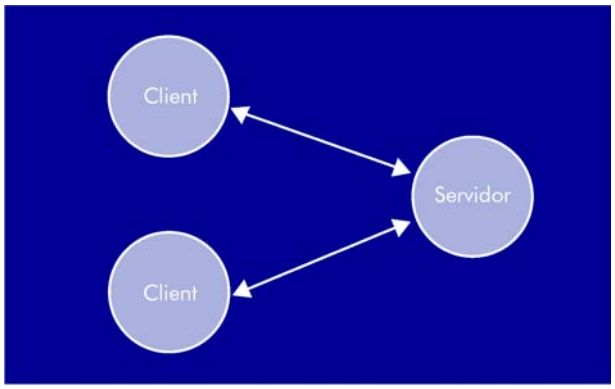


L'usuari de les aplicacions pot ser una persona, però també una altra aplicació. En el primer cas, el client ha de disposar d'una **interfície d'usuari** que permeti el diàleg, canalitzant les peticions i mostrant-li els resultats. En el segon, l'accés se sol implementar amb crides a funcions.

El model es pot complicar si s'afegeixen més entitats en l'aplicació distribuïda. Vegem-ne dos exemples:

- Múltiples clients:

Figura 73.

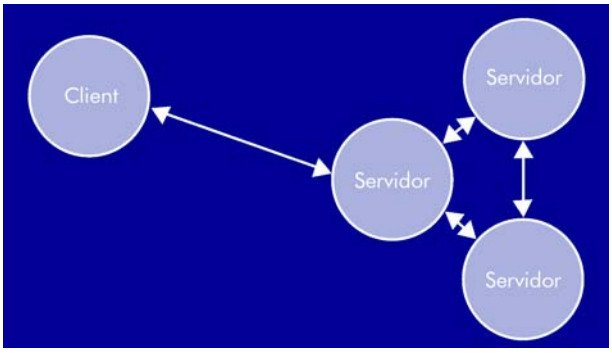


El servidor ha d'estar preparat per a rebre múltiples connexions, que poden ser simultànies.

ANOTACIONS

- Múltiples servidors:

Figura 74.



Diversos servidors ofereixen serveis complementaris. Si el servidor no pot satisfer la petició del client, però està en contacte amb un altre servidor que sí que ho pot fer, llavors li ho sol·licita. Els servidors poden dialogar entre si amb protocols propis o seguint el model client/servidor.

Des del punt de vista del client, aquesta multiplicitat de servidors és completament transparent: realitza una petició a un servidor i n'espera una resposta.

Quan diferents servidors cooperen per a oferir un servei, parlem de *sistema servidor*.

La gran majoria de les aplicacions que s'utilitzen a Internet segueixen aquest model client/servidor. D'aquest fet provenen els noms que s'empren amb assiduïtat: servidor web, client de correu, servei de noms, etc.

El disseny d'una aplicació distribuïda que segueixi el model client/servidor ha d'incloure bàsicament **l'especificació dels serveis** que ofereix el servidor i **l'especificació del protocol d'accés**, on es descriu com es demanen aquests serveis i com s'ha de retornar el resultat.

15.1. El model d'igual a igual

El model client/servidor implica una certa asimetria. Tendim a veure els servidors com a sistemes potents que ofereixen els seus serveis a

ANOTACIONS

màquines molt més senzilles, els clients. El gran desenvolupament d'Internet en el seu vessant comercial ha contribuït a aquesta visió. Els usuaris d'Internet, des dels seus PC, tenen un clar comportament de client, accedint a pàgines web allotjades en servidors remots, o enviant correus electrònics a altres usuaris.

A mesura que l'ús d'Internet ha madurat, els usuaris de la xarxa han sentit la necessitat d'exercir un paper més actiu, i convertir les seves màquines en servidors que d'altres puguin aprofitar, o desenvolupar models nous d'aplicació on els rols no estan tan clarament diferenciats. Un d'aquests models és el que es coneix com *d'igual a igual*, i les aplicacions més típiques que segueixen aquest model de comportament són els sistemes de compartició d'arxius o la missatgeria instantània.

Nota

Diferents característiques de la xarxa que s'han afegit per qüestions de seguretat o d'eficiència perjudiquen molt el desenvolupament d'entorns d'igual a igual. Podem esmentar com les més paradigmàtiques els tallafocs, les adreces IP dinàmiques, el NAT (*Network Address Translation*) i l'amplada de banda asimètrica que ofereixen en molts casos els proveïdors d'Internet en els dos sentits de transmissió. La majoria d'aplicacions s'intenten sobreposar a aquests obstacles amb estratègies més o menys afortunades. De tota manera, és clar que en un futur han de canviar coses en l'àmbit de xarxa per a no limitar les enormes possibilitats que tenim en l'àmbit de les aplicacions.

Nota

El servei DNS i el servei News (o Usenet o NNTP), presents a Internet des dels seus inicis, són clarament d'igual a igual.

De fet, Internet en el seu origen era d'igual a igual. Els primers ordinadors connectats a la xarxa eren semblants, i una aplicació com `ftp` permetia la transferència d'arxius entre qualssevol d'aquestes màquines. Dit d'una altra manera, que el protocol que usa l'aplicació segueixi el model client/servidor no és incompatible amb el fet que la xarxa estigui formada per ordinadors de potència semblant entre ells es vegin com una xarxa d'igual a igual.

Perquè en el fons, si analitzem com funcionen les aplicacions típiques d'entorns d'igual a igual, com l'extingit Napster o Gnutella o ICQ o

Jabber, veuríem que segueixen protocols típics client/servidor, tipus "jo et demano, tu em dónes". El que passa és que en aquest tipus d'entorns qualsevol pot fer alhora de client i de servidor. Per tant, és erroni considerar tots dos models excloents.

Un sistema pot ser d'igual a igual si presenta una certa simetria en el sentit que les diferents màquines que el componen són semblants i poden exercir els mateixos rols, i usar un model client/servidor en el desenvolupament de les aplicacions.

D'altra banda, hi ha aplicacions considerades d'igual a igual perquè permeten la comunicació directa entre iguals, però que necessiten un servidor extern (o diversos) on se centralitza el control de la xarxa (Napster era un clar exemple d'aquest model). Hi ha autors que són reticents a considerar aquests entorns realment d'igual a igual, i prefereixen guardar el terme per a aquelles xarxes on només intervenen les màquines propietat dels mateixos usuaris. No és objectiu d'aquest material didàctic entrar en aquesta discussió.

ANOTACIONS

16. Servei de noms Internet

La xarxa Internet permet l'accés a una ingent quantitat d'ordinadors i, en general, de recursos, la majoria dels quals es poden referenciar mitjançant un nom. Per motius d'eficiència i simplicitat (poder saber amb facilitat quin recurs correspon a cada nom, poder assignar-ne altres de nous sense perill de duplicitats o ambigüitats, etc.), tots els noms de la xarxa estan organitzats d'una manera jeràrquica, formant un **sistema de dominis**.

Per a obtenir informació referent a qualsevol nom, s'utilitza un servei que, funcionalment, és com una base de dades: s'hi fan consultes, que poden incloure una sèrie de criteris de selecció, i respon amb la informació sol·licitada. En un inici, quan el nombre d'ordinadors connectats a la xarxa era relativament petit, aquesta base de dades la gestionava una única autoritat central, el Network Information Center (NIC).

A mesura que s'expandia la xarxa, aquest model de gestió es feia cada vegada més inviable, tant per l'enorme trànsit que generaven les consultes, com per la dificultat de mantenir la informació actualitzada. Va ser llavors quan va néixer el **servei de noms de domini** (DNS), que segueix el model d'una base de dades distribuïda descentralitzada.

Si voleu més informació sobre el DNS, podeu consultar les obres següents:

P.V. Mockapetris (1987,1 de novembre). *RFC 1034- Domain names - concepts and facilities*.

P.V. Mockapetris (1987, 1 de novembre). *RFC 1035- Domain names - implementation and specification.*

L'especificació del DNS és en els documents RFC 1034, que defineix els conceptes i l'organització del sistema de dominis, i RFC 1035, que defineix el protocol d'accés al servei.

ANOTACIONS

16.1. El sistema de noms de domini

El sistema de noms de domini, en què es basa el DNS, proporciona un espai de noms per a referenciar recursos que, per norma general, són ordinadors connectats a la xarxa, però que també poden ser, per exemple, bústies de correu electrònic.

En el sistema de noms de domini, els noms estan organitzats jeràrquicament en forma d'arbre. Cada node d'aquest últim té una etiqueta que el distingeix dels seus nodes "germans".

El **nom de domini** corresponent a un node es defineix com la seqüència formada per les etiquetes existents en el camí entre aquest node i l'arrel.

Cada etiqueta constitueix una cadena de caràcters de longitud compresa entre 0 i 63 caràcters. L'etiqueta amb longitud 0 es reserva per al node arrel: cap altre node no pot tenir una etiqueta buida.

Nota

A l'hora de comparar noms, les lletres majúscules i minúscules es consideren equivalents. No obstant això, per a respondre les consultes, els servidors DNS han d'enviar els noms tal com estan emmagatzemats a la base de dades, respectant les majúscules i minúscules, en previsió del cas que en alguna versió futura del servei es considerin diferents.

El protocol d'accés al DNS defineix el format per a representar els noms de domini quan s'han d'enviar en les consultes i les respostes, com veurem més endavant. Aquest format no imposa restriccions en els caràcters que hi pot haver en una etiqueta; tanmateix, estableix una "sintaxi preferida" per a facilitar les implementacions d'altres serveis. D'acord amb aquesta última, els únics caràcters vàlids en una etiqueta són les lletres, majúscules o minúscules, els dígits decimals i el símbol "-" amb l'excepció que el primer caràcter només pot ser una lletra, i l'últim no pot ser "-".

A part de la representació definida en el protocol, quan es vol utilitzar una notació textual per a expressar un nom de domini, se segueix la convenció de concatenar les etiquetes que el formen, separant-les amb el caràcter ".". En l'ordre utilitzat, les etiquetes van des del nivell jeràrquic més baix fins al més alt. Per tant, l'última etiqueta és la del node arrel que, com acabem d'observar,

és buit, de manera que els noms acaben en "." com en l'exemple següent:

`campus.uoc.edu.`

Noteu la presència del punt final.

Aquest exemple és el que es denomina un *nom de domini absolut*, però també es poden utilitzar noms relatius, referents a un node origen implícit. Així, dins del domini `uoc.edu.` podem utilitzar el nom `tibet` per a fer referència al nom de domini `tibet.uoc.edu.` Moltes vegades, el node origen dels noms relatius és l'arrel, de manera que l'exemple anterior també es pot escriure de la manera següent:

`campus.uoc.edu`

Nota

Al Regne Unit abans era usual utilitzar l'ordre invers en la representació textual dels noms de domini, per exemple: `uk.ac.ic.doc.src` en lloc de `src.doc.ic.ac.uk`; avui dia aquest costum es troba en desús.

En un inici, els TLD (*top level domain*, 'dominis de nivell superior'), és a dir, els subordinats directament a l'arrel, s'utilitzaven per a agrupar els diferents tipus d'organitzacions a què pertanyien recursos com, per exemple, els que presentem en la taula següent:

Taula 6.

Tipus d'organització	Nom del domini
Empreses i organitzacions comercials	com
Universitats i centres educatius	edu
Organismes governamentals	gov
Organitzacions militars	mil
Altres tipus d'organitzacions	org

Nota

En aquests moments, l'organisme encarregat de gestionar els dominis és l'ICANN. Podeu consultar el seu web: http://www.icann.org.

Des de llavors fins avui, s'han afegit al nivell superior del sistema de noms els elements següents:

a) D'una banda, noves classes d'organitzacions, com ara `net` (centres relacionats amb la gestió de la xarxa), `int` (organitzacions internacionals) o, més recentment, `biz`, `info`, `name`, `pro`, `aero`, `coop` i `museum`.

b) D'altra banda, cada país té assignat un nom de domini que coincideix amb el codi corresponent de dues lletres especificat en l'estàndard ISO 3166, per exemple: `es` (Espanya), `fr` (França), de

ANOTACIONS

(Alemanya), etc. Això significa que, en realitat, les diferents branques de l'arbre es poden encavallar i, de fet, hi ha noms pertanyents a diferents dominis d'alt nivell que corresponen a una mateixa organització.

Així mateix, és possible que un recurs determinat (ordinador, bústia de correu, etc.) tingui assignats diversos noms, que poden ser en la mateixa branca de l'arbre o en branques completament independents. En aquest cas, es considera que un dels noms és el denominat *nom canònic* i els altres, *àlies*.

16.2. Model del DNS

Els agents que intervenen en el DNS són els següents:

Nota

Consulteu la definició de sistema servidor en la unitat 15.

a) Els **servidors**, que reben les consultes i envien les respostes corresponents. A més de l'accés directe a una base de dades local, en la qual es troba una part de la base de dades total del sistema de noms, el servidor té accés indirecte a la resta de la base de dades per mitjà d'altres servidors. El conjunt de servidors DNS constitueix, doncs, un sistema servidor.

b) Els **resolvedors**, que actuen com a clients del servei. Per norma general, un resolvedor és un programa o una llibreria que rep peticions de les aplicacions d'usuari, les tradueix a consultes DNS i extreu de les respostes la informació sol·licitada. El resolvedor ha de tenir accés directe almenys a un servidor DNS.

Nota

Utilització del resolvedor

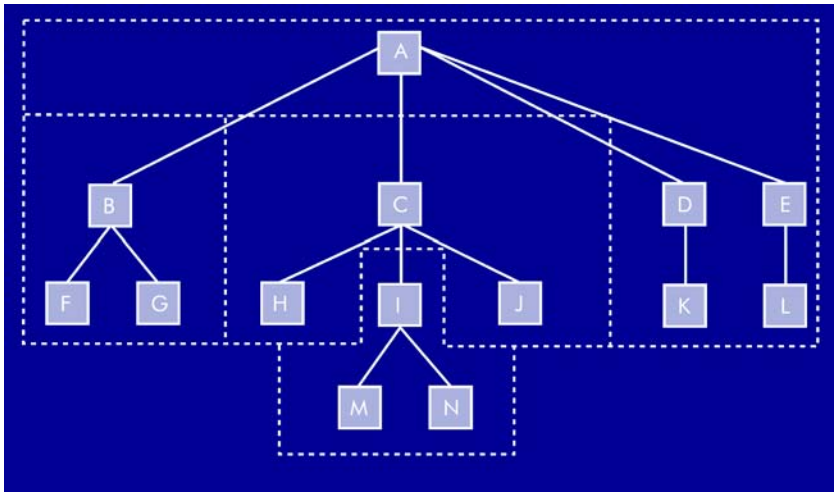
Una aplicació que serveixi per a copiar fitxers emmagatzemats en altres ordinadors pot acceptar que l'usuari indiqui a quin ordinador vol accedir per mitjà del seu nom. Per tant, per a poder-s'hi comunicar, l'aplicació haurà de cridar el resolvedor per a obtenir-ne l'adreça a partir del nom. Com que l'aplicació i el resolvedor, en general, es trobaran ubicats en el mateix sistema, no cal establir cap protocol de comunicació entre ells.

ANOTACIONS

Per a fer manejables la gestió i administració del sistema de noms de domini, els seus nodes estan agrupats en zones. Cada **zona** està formada per un node que es diu *superior* i tots els que es trobin en els nivells jeràrquicament inferiors fins a arribar als nodes terminals (les fulles de l'arbre) o a nodes d'altres zones que siguin superiors.

La figura següent mostra un exemple de divisió d'un arbre en quatre zones:

Figura 75.



Nota

Com que cada zona se sol denominar amb el nom de domini del seu node superior, podem dir que les zones de la figura són A, B.A, C.A i I.C.A.

L'objectiu d'agrupar els nodes en zones consisteix a assignar a una autoritat la responsabilitat de gestionar tots els nodes de cada zona.

L'administrador designat per a aquesta autoritat pot afegir o esborrar nodes dins de la seva zona, modificar la informació dels seus nodes o crear noves subzones i delegar la seva gestió en altres autoritats.

La informació referent a una zona ha d'estar emmagatzemada en la base de dades local d'un servidor, del qual es diu que és un servidor amb autoritat sobre aquesta zona. Aquest servidor pot contestar directament les consultes que rebi sobre els nodes de la seva zona, sense necessitat d'accedir a altres servidors. És a dir, en aquest cas, el servidor enviarà respostes amb autoritat.

Exemple

Una zona pot correspondre a un país i les seves subzones poden correspondre a organitzacions d'aquest país. Cada organització, al seu torn, pot crear subzones per a diferents departaments, etc.

Si una consulta es refereix a una altra zona, hi ha dues maneres possibles de generar la resposta:

- En el **mode no recursiu**, la resposta únicament inclou una referència a algun servidor que pot proporcionar més informació. El client s'ha de preocupar de continuar realitzant consultes fins a trobar la resposta definitiva.
- Si el client sol·licita el **mode recursiu**, el servidor s'ocupa de buscar la informació on sigui necessari, i només pot retornar la resposta final o un error, però no referències a altres servidors.

L'especificació DNS estableix que tots els servidors han de suportar el mode no recursiu, i que el mode recursiu és opcional, encara que a la pràctica, la majoria de les consultes dels clients són en mode recursiu.

Quan un servidor ha de respondre a una consulta en mode recursiu, demana la informació a altres servidors i envia la resposta rebuda a qui ha fet la consulta. És habitual que el servidor afegeixi la informació obtinguda a la seva base de dades, en el que s'anomena *cau*. D'aquesta manera, si rep una altra consulta (del mateix client o d'un altre) en què se sol·licita la mateixa informació, no li caldrà tornar-la a demanar a altres servidors, sinó que pot aprofitar la que hi ha a la cau. En aquest cas, el servidor ha d'avisar al client que li envia una resposta sense autoritat, perquè és possible que les dades s'hagin modificat en el servidor d'origen des que es van sol·licitar per primera vegada. D'altra banda, els resolvedors també solen desar en una memòria pròpia les respostes que reben dels servidors.

Per a oferir una alta fiabilitat en el servei, l'especificació DNS requereix que per a cada zona hi hagi com a mínim dos servidors amb autoritat que, per norma general, responguin a les funcions següents:

a) Un actua com a **servidor primari** i té els fitxers originals de la base de dades corresponent a la zona; és a dir, aquells que l'administrador ha d'actualitzar directament cada vegada que hi hagi una modificació en els seus nodes.

ANOTACIONS

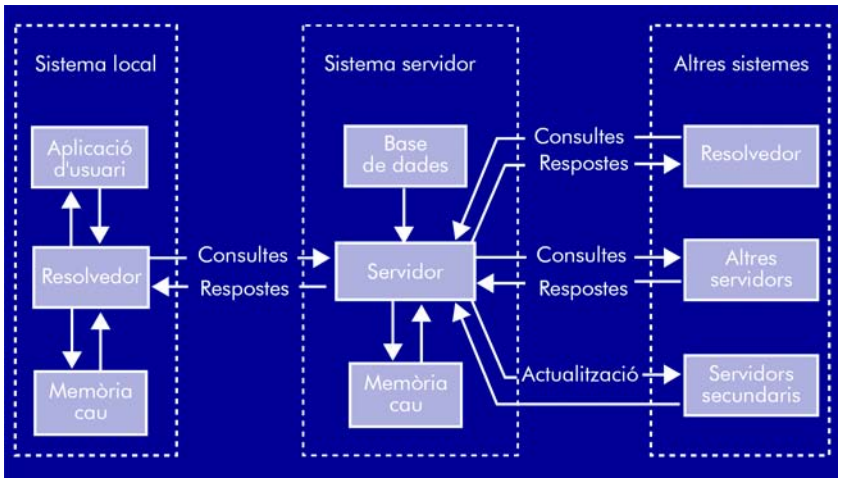
b) Els altres actuen com a **servidors secundaris** i actualitzen automàticament les seves bases de dades a partir de la del primari; per exemple, per mitjà de consultes periòdiques per a saber si s'hi ha produït algun canvi. D'aquesta manera, si un servidor primari és temporalment inaccessible (per una caiguda de la xarxa, del mateix servidor, etc.), els clients poden enviar les seves consultes a un dels servidors secundaris.

Nota

En l'argot DNS, *refrescar les dades* significa actualitzar-les fent consultes periòdiques.

La figura següent mostra una possible configuració del servei DNS en un ordinador:

Figura 76.



A la pràctica, es poden trobar moltes variants d'aquest esquema; per exemple, que el servidor DNS es trobi al mateix ordinador que el resolvedor i que tots dos comparteixin la memòria, que el resolvedor tingui accés a diferents servidors DNS, etc.

16.3. Base de dades DNS: els registres de recurs

Com hem vist en els subapartats anteriors, un nom del sistema de dominis identifica un node, i per a cada node hi pot haver certa informació emmagatzemada en els servidors corresponents. L'objectiu del servei DNS consisteix a permetre l'obtenció d'aquesta informació a partir del nom de domini.

ANOTACIONS

La informació associada a un node consta d'un conjunt de registres de recurs. Els registres de recurs de tots els nodes formen la **base de dades DNS**.

Cada registre de recurs consta dels camps següents, que s'expliquen a continuació:

- Nom
- Tipus
- Classe
- Temps de vida (TTL)
- Dades del recurs (RDATA)

a) **Nom**. Conté el nom de domini del node al qual està associat el registre.

b) **Tipus**. Indica quin tipus d'informació conté el registre. Els valors que podem trobar en aquest camp i els tipus d'informació que representen són els següents:

- A: adreça d'un ordinador.
- CNAME: nom canònic d'un àlies.
- HINFO: informació sobre el tipus d'ordinador.
- *MX* (*mail exchanger*): nom d'un servidor de correu electrònic per a un domini.
- NS: nom d'un servidor DNS amb autoritat per a una zona.
- PTR: nom d'un domini que conté informació relacionada amb un node.
- SOA (*start of authority*): informació sobre el node superior d'una zona.

Altres valors possibles del camp Tipus són els següents:

- MB: nom d'una bústia de correu.
- MG: membre d'un grup de correu.

ANOTACIONS

- MR: nom nou d'una bústia de correu.
- MINFO: informació sobre una bústia o una llista de distribució de correu.
- WKS (*well known services*): llista de serveis que proporciona un ordinador.
- TXT: text descriptiu.
- NULL: registre buit.

c) **Classe**. Indica la família de protocols utilitzada en l'espai de noms. Per norma general, serà la família de protocols Internet, però n'hi pot haver d'altres.

d) **Temps de vida** (**TTL**). Indica el temps màxim que un servidor o un resolvedor pot guardar el registre en la seva memòria cau.

e) **Dades del recurs** (**RDATA**). El valor d'aquest camp depèn del tipus de registre:

- Si el registre és de tipus A, en la classe Internet, el valor és un número de 32 bits que correspon a una adreça IP.
- Si el registre és de tipus CNAME, el valor és un nom de domini que correspon al nom canònic de l'àlies associat amb el registre. Si un node de l'arbre de noms és un àlies, l'única informació que pot tenir associada és un registre CNAME.
- Si el registre és de tipus HINFO, el valor és una cadena de caràcters.
- Si el registre és de tipus MX, el valor té dos subcamps, el primer és un número que representa una preferència (com més petit sigui el número, més preferència) i el segon és el nom d'un ordinador que està disposat a acceptar missatges destinats al domini corresponent al registre.

Nota

El sistema de noms de domini proporciona un conjunt d'espais de noms, un per a cada classe, tots sobre un mateix arbre. Les dades corresponents a un node en una classe poden ser diferents de les del mateix node en una altra classe (un node pot no tenir cap registre de recurs d'alguna classe), ja que les bases de dades (com les divisions en zones) són separades i independents.

Nota

L'estàndard RFC 974 especifica com s'han d'utilitzar els registres MX per a direccionar un missatge de correu electrònic segons l'adreça del destinatari (per exemple, usuari@uoc.edu). D'aquesta adreça, se'n separa la part corresponent al domini de correu (uoc.edu), es consulta el sistema DNS per a saber quins servidors accepten correu per a aquest domini i se'n tria un tenint en compte la seva preferència. Si la

ANOTACIONS

comunicació amb aquest servidor no té èxit, s'intenta amb els altres per ordre de preferència.

Si no hi ha cap registre MX associat al domini, s'entén que només disposa d'un servidor de correu: l'ordinador que tingui per nom el del domini.

- Si el registre és de tipus NS, el valor és un nom d'ordinador.
- Si el registre és de tipus PTR, el valor és un nom de domini.
- Si el registre és de tipus SOA, el valor del camp RDATA en un registre d'aquest tipus està format pels set subcamps següents:

– MNAME: nom del servidor primari de la zona.

– RNAME: nom de domini corresponent a la bústia del responsable de la zona.

– SERIAL: nombre que s'ha d'augmentar cada vegada que hi hagi una modificació en les dades de la zona.

– REFRESH: temps que ha de transcórrer perquè els servidors secundaris refresquin les seves dades.

– RETRY: temps que cal esperar per a tornar a intentar un refresc si no s'ha aconseguit contactar amb el servidor primari.

– EXPIRE: temps màxim a partir del qual les dades d'un servidor secundari es consideraran sense autoritat si no s'han refrescat.

– MINIMUM: valor mínim del camp TTL en els registres de la zona.

Nota

Refresc d'una zona en un servidor secundari

El procés de refresc d'una zona en un servidor secundari consisteix a demanar al primari el seu registre SOA i comprovar si ha variat el camp SERIAL. Si ha variat, és necessari dur a terme una transferència de les dades de la zona (per exemple, amb una petició AXFR del protocol DNS, per mitjà del protocol FTP, etc.) i, si no, no és necessari fer res.

Nota

Cada node que sigui el superior d'una zona ha de tenir associat un registre SOA.

Nota

El nom de la bústia es pot expressar com un nom de domini seguint la sintaxi general de separar els noms dels nodes amb "." (en aquest cas, la part corresponent a l'usuari seria el node inferior). Per exemple, a la bústia `usuari@uoc.edu` li correspon el nom de domini `usuari.uoc.edu`.

ANOTACIONS

El protocol d'accés al DNS, com veurem en el subapartat següent, defineix el format de representació dels registres de recurs en les respostes dels servidors. Tanmateix, també és habitual utilitzar una representació textual; per exemple, en els fitxers de la base de dades que l'administrador d'una zona pot editar i actualitzar. La sintaxi típica d'aquesta representació textual és la següent:

```
[nom] ( [classe] [TTL] | [TTL] [classe] ) tipus RDATA
```

Cada registre s'escriu en una línia, tot i que si és massa llarg, es poden utilitzar parèntesis, dins dels quals s'ignoren els salts de línia. El símbol ";" serveix per a introduir-hi comentaris. El camp Nom és opcional (per defecte es pren el del registre anterior), com també ho són Classe (el valor del camp Classe, per norma general, és `IN`, que correspon a la classe Internet) i TTL, que es pot escriure en qualsevol ordre. Els camps que representen dominis poden ser absoluts (acabats en ".") o relatius (respecte a un origen preestablert), i els que representen temps s'expressen en segons.

El camp Nom pot començar amb l'etiqueta "*" per a indicar que la informació del registre s'aplica als dominis que tinguin qualsevol etiqueta o seqüència d'etiquetes al lloc del "*" i no figurin en cap altre registre.

Nota

Línies del fitxer de dades d'un servidor `acme.com`

Aquestes podrien ser algunes línies del fitxer de dades d'un hipotètic servidor de la zona acme.com:

```
acme.com.   IN   SOA   servidor.acme.com.   admin.acme.com. (
                          38      ; SERIAL
                          7200    ; REFRESH
                          600     ; RETRY
                          3600000 ; EXPIRE
                          60 )    ; MINIMUM
                       NS  servidor.acme.com.
                       NS  servidor2.acme.com.
                       NS  dns.competencia.com.
                       MX  10  correu.acme.com.
                       MX  20  servidor.acme.com.
  *.acme.com.          MX  10  correu.acme.com.
  servidor.acme.com.   A   128.52.46.32
                       A   128.32.11.99
  servidor2.acme.com.  A   128.52.46.33
  correu.acme.com.     A   128.52.46.34
```

ANOTACIONS

Un ús habitual dels registres PTR és facilitar l'obtenció del nom d'un ordinador a partir de la seva adreça per mitjà d'un domini especial denominat `IN-ADDR.ARPA`. El DNS proporciona una operació per a efectuar consultes inverses; és a dir, donat un registre, retorna el nom de domini al qual correspon. Però, aquesta operació pot ser molt costosa per al servidor, i no es pot garantir que la resposta sigui única o completa, ja que hi pot haver altres servidors que continguin informació relacionada.

No obstant això, com que la traducció d'adreces IP a noms és útil i, fins i tot, necessària, en alguns protocols (per exemple, rlogin i rsh), s'ha adoptat un mecanisme per a facilitar-la que consisteix a introduir registres addicionals que actuïn com a índex. D'aquesta manera, cada vegada que es modifiqui una adreça en la base de dades DNS, o se n'afegeixi una, serà necessari actualitzar dos registres:

- El registre A que té per nom el de l'ordinador.
- Un registre PTR que té un nom pertanyent al domini `IN-ADDR.ARPA.`

Els noms del domini `IN-ADDR.ARPA` poden tenir fins a quatre etiquetes (sense comptar `IN-ADDR i ARPA`), que representen els valors possibles dels bytes d'una adreça IP en decimal, entre 0 i 255. D'aquestes etiquetes, la de nivell més alt correspon al primer byte de l'adreça, i la de nivell més baix, a l'últim. D'aquesta manera, es pot tenir una estructura de zones i autoritats delegades similar a la de la resta dels dominis. Un nom del domini `IN-ADDR.ARPA` que tingui les quatre etiquetes numèriques correspondrà a l'adreça d'un ordinador, i se li associarà un registre PTR que apunti al nom d'aquest ordinador. Els noms que disposin de menys de quatre etiquetes numèriques seran adreces de xarxes i podran tenir associats registres PTR apuntant al nom de les passarel·les connectades a aquestes xarxes.

Nota

Una excepció o la sintaxi preferida per a les etiquetes la tenim en els noms del domini `IN-ADDR.ARPA`, que han de començar amb un dígit.

Nota

Registres de traducció

Seguint l'exemple anterior, els registres del domini `INADDR.ARPA` següents podrien servir per a dur a terme la traducció d'adreces a noms:

```
32.46.52.128.IN-ADDR.ARPA.  PTR  servidor.acme.com.
```

```
33.46.52.128.IN-ADDR.ARPA.  PTR servidor2.acme.com.
34.46.52.128.IN-ADDR.ARPA.  PTR correu.acme.com.
99.11.32.128.IN-ADDR.ARPA.  PTR servidor.acme.com.
46.52.128.IN-ADDR.ARPA.      NS  servidor.acme.com.
                             NS  servidor2.acme.com.
                            PTR servidor.acme.com.
11.32.128.IN-ADDR.ARPA.      NS  servidor.acme.com.
                             NS  servidor2.acme.com.
                            PTR servidor.acme.com.
```

16.4. Protocol

16.4.1. Mecanismes de transport

Per a accedir al DNS es poden utilitzar els protocols de transport UDP o TCP. Per norma general, s'utilitza UDP en les consultes dels clients per la seva simplicitat i pels pocs recursos que requereix. Si no arriba la resposta d'un datagrama en un temps determinat, simplement es retransmet (fins que hagi transcorregut un temps màxim). En canvi, s'utilitza TCP quan convé assegurar una transmissió fiable; per exemple, en les transferències de dades d'una zona d'un servidor a l'altre. Tant en un cas com en l'altre, el número de port utilitzat és el 53.

Les consultes i les respostes s'intercanvien per mitjà de missatges, el format dels quals depèn del protocol que s'utilitzi:

- En UDP, cada missatge s'envia en un datagrama, amb una longitud màxima de 512 bytes.

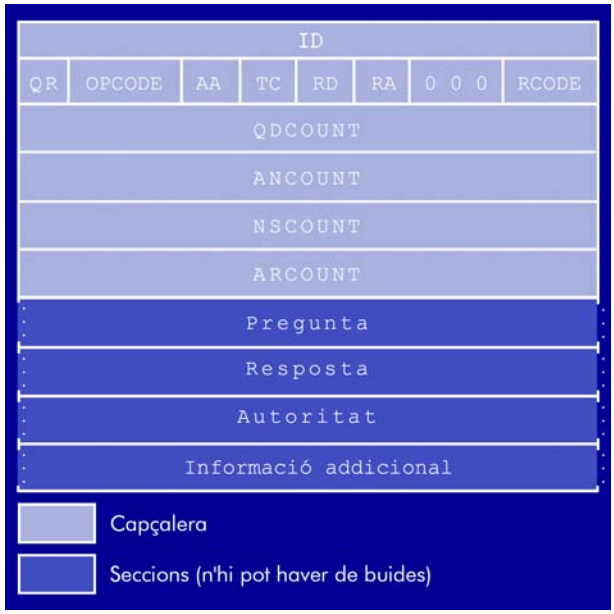
- En TCP, els missatges s'envien prefixats amb 2 bytes, que n'indiquen la longitud, per saber on acaben, ja que el més normal és intercanviar-ne més d'un en una mateixa connexió.

ANOTACIONS

16.4.2. Missatges

L'esquema següent mostra l'estructura d'un missatge DNS:

Figura 77.



Cada missatge està format per una capçalera i quatre seccions (pregunta, resposta, autoritat i informació addicional).

La **capçalera** consta dels camps següents:

- **ID** és un nombre de 16 bits que assigna qui realitza la consulta i que es copia en la resposta perquè se sàpiga a quina consulta correspon.
- **QR** (*query/response*) és un bit que indica si el missatge és una consulta (0) o una resposta (1).
- **OPCODE** és un codi d'operació de 4 bits. Actualment, hi ha definits els de consulta directa (QUERY, codi 0), consulta inversa (IQUERY, codi 1) i petició d'estatus (STATUS, codi 2). Per norma general, el valor d'aquest camp serà QUERY, ja que l'operació IQUERY és opcional i poc suportada, i l'operació STATUS no està definida en l'especificació RFC 1034.
- **AA** (*authoritative answer*) és un bit que informa que el missatge és una resposta amb autoritat.

ANOTACIONS

- **TC** (*truncation*) és un bit que avisa que el missatge s'ha truncat perquè no hi cap en un datagrama. Per a saber quin és el seu contingut complet, cal utilitzar el protocol TCP.
- **RD** (*recursion desired*) és un bit que indica que el client sol·licita resposta en mode recursiu.
- **RA** (*recursion available*) és un bit que informa que el servidor suporta el mode recursiu. Si els bits RD i RA valen 1, el missatge és una resposta recursiva.
- **RCODE** és un codi de resposta de 4 bits. Els codis possibles són: cap error (0), error de format (1), error intern del servidor (2), domini inexistent (3), operació no implementada (4) o consulta rebutjada (5).
- **QDCOUNT**, **ANCOUNT**, **NSCOUNT**, **ARCOUNT** són nombres de 16 bits que indiquen quantes entrades hi ha en cada secció del missatge: pregunta, resposta, autoritat i informació addicional, respectivament.

Secció de pregunta

Els missatges corresponents a les consultes disposaran d'una o més entrades (en general, una) en la secció de pregunta. Cada entrada està formada per tres camps:

- **QNAME**: nom de domini del qual el client vol obtenir informació.
- **QTYPE**: nombre de 16 bits que indica el tipus de registre de recurs que el client vol obtenir com a resposta. El valor d'aquest camp pot ser qualsevol dels nombres que es poden trobar en el camp TYPE d'un registre, o un dels següents:

– **XFR** (codi 252): s'utilitza per a sol·licitar la transferència de tots els registres d'una zona (quan un servidor secundari vol refrescar la seva base de dades).

– **MAILB** (codi 253): serveix per a sol·licitar tots els registres relacionats amb bústies (MB, MG i MR).

– * (codi 255): serveix per a demanar tots els registres de qualsevol tipus corresponents a un nom.

ANOTACIONS

- **QCLASS**: nombre de 16 bits que indica la classe de registres desitjats. El seu valor pot ser el del camp CLASS dels registres o el valor especial "*" (codi 255), que representa totes les classes.

Secció de resposta

Els missatges corresponents a les respostes contenen en la secció de resposta el conjunt de registres de recurs que satisfan els criteris expressats als camps de la consulta. És a dir, el servidor envia en la resposta els registres corresponents al nom sol·licitat que concorden amb el tipus o amb els tipus requerits i en la classe o classes requerides.

Un cas especial és el dels àlies. Si el nom sol·licitat correspon a un àlies i el camp QTYPE és diferent de CNAME, el servidor ha de repetir automàticament la consulta substituint el nom original pel nom canònic. D'aquesta manera, el client obtindrà directament les dades desitjades tant si utilitza un àlies, com el nom canònic.

La resposta a una petició AXFR consistirà en una seqüència de missatges. El primer ha de contenir el registre SOA de la zona; en els següents hi ha d'haver la resta dels registres, i l'últim ha de ser una altra vegada el registre SOA, perquè el receptor sàpiga que ja ha acabat la transferència.

Secció d'autoritat

En una resposta, la secció d'autoritat pot contenir registres que referenciïn un servidor amb autoritat per a respondre la consulta (per al cas de les respostes en mode no recursiu).

Secció d'informació addicional

La secció d'informació addicional pot contenir altres registres de recurs relacionats amb la consulta, però que no formen part de la resposta.

16.4.3. Representació dels registres de recurs

Quan un missatge ha de contenir una cadena de caràcters, es representa per mitjà d'1 byte, que n'indica la longitud i, a continua-

ANOTACIONS

ció, els caràcters de la cadena. Un nom de domini es representa com una concatenació de cadenes, una per a cada una de les etiquetes que el formen. Com que els noms de domini acaben amb l'etiqueta del node arrel, una cadena que tingui per longitud 0 (i, per tant, cap caràcter a continuació) indica el final del nom. Per a simplificar la implementació del servei, l'especificació RFC 1035 requereix que la longitud total d'un nom de domini no sigui superior a 255 bytes.

Nota

Representació comprimida dels noms de domini

L'especificació RFC 1035 també defineix una representació comprimida dels noms de domini quan n'hi ha molts d'iguals, o que acaben igual, en un missatge (com ara els que contenen registres SOA). En lloc d'una etiqueta, en un nom hi pot haver 1 byte amb els 2 bits de més pes a 1 (per tant, no es pot confondre amb la longitud d'una etiqueta, que com a màxim pot ser 63). Els altres 6 bits i els 8 del byte següent indiquen una posició dins del missatge en què es troba la continuació del nom.

La representació dels registres de recurs que poden aparèixer en les seccions de resposta, autoritat i informació addicional segueix el format següent:

Figura 78.

ANOTACIONS

Els camps que formen els registres de recurs dins dels missatges DNS són els següents:

- El **camp NAME** és el nom de domini corresponent al registre.
- El **camp TYPE** és un nombre de 16 bits que indica el tipus de registre, segons la taula següent:

Taula 7.

Nombre	Tipus de registre
1	A
2	NS
5	CNAME
6	SOA
7	MB
8	MG
9	MR
10	NULL
11	WKS
12	PTR
13	HINFO
14	MINFO
15	MX
16	TXT

- El **camp CLASS** és un nombre de 16 bits que indica la classe del registre. Per norma general, el seu valor és 1, que correspon a la classe Internet.
- El **camp TTL** és un nombre de 32 bits que indica el temps de vida del registre en segons.
- El **camp RDLENGTH** és un nombre de 16 bits que indica quants bytes hi ha en el camp RDATA.
- El **camp RDATA** conté les dades del registre, i el seu format depèn del valor del camp TYPE:

– En els registres A (de la classe Internet), és un nombre de 32 bits.

– En els registres CNAME, NS, PTR, MB, MG i MR, és un nom de domini.

ANOTACIONS

– En els registres SOA, és una seqüència de dos noms de domini (MNAME i RNAME) seguida de cinc nombres de 32 bits (SERIAL, REFRESH, RETRY, EXPIRE i MINIMUM).

– En els registres MX, és un enter de 16 bits (la preferència) seguit d'un nom de domini.

– En els registres HINFO, és la concatenació de dues cadenes de caràcters. La primera indica el tipus d'UCP de l'ordinador, i la segona, el sistema operatiu.

– En els registres WKS, és un nombre de 32 bits, com en els registres A, seguit d'un nombre de 8 bits que identifica un protocol (per exemple, 6 = TCP; 17 = UDP) i una seqüència de bits en la qual cada bit indica si l'ordinador ofereix un determinat servei o no.

– En els registres MINFO, és la concatenació de dos noms de domini. El primer constitueix l'adreça del responsable de la bústia o llista de distribució, i el segon, l'adreça on s'han d'enviar les notificacions d'error.

– En els registres TXT, és una cadena de caràcters.

– En els registres NULL, pot ser qualsevol cosa.

16.5. Implementacions del DNS

Pràcticament tots els ordinadors connectats a la xarxa Internet incorporen alguna implementació d'un client DNS (per norma general, una llibreria de funcions cridades per les aplicacions) per a poder accedir a altres ordinadors coneixent-ne els noms. Aquest és el cas, per exemple, dels sistemes Unix o GNU/Linux, que proporcionen funcions com ara `gethostbyname` o `gethostbyaddr`.

En molts sistemes Unix o GNU/Linux també es troba disponible una utilitat denominada `nslookup`, que serveix per a efectuar consultes directament a un servidor DNS. `nslookup`, en el mode de funcionament bàsic, accepta ordres com les següents:

- *nom*: envia una consulta utilitzant el nom especificat com a valor del camp QNAME i mostra els resultats obtinguts, indicant si la

ANOTACIONS

resposta és sense autoritat. En cas de ser-ho, informa de quins registres ha enviat el servidor a la secció d'autoritat de la resposta.

- `server` *`servidor`*: canvia el servidor per defecte al qual s'envien les consultes.
- `ls` *`domini`*: mostra una llista de nodes del domini i informació associada; si la llista és molt llarga, es pot desar en un fitxer afegint-hi `> fitxer`.
- `set querytype=`*`tipus`*: permet canviar el camp QTYPE de les consultes que, per defecte, és A, a qualsevol altre valor (NS, MX, PTR, CNAME, etc.).
- `set q=`*`tipus`*: és equivalent a `set querytype`.
- `set domain=`*`origen`*: canvia el domini origen que s'ha d'utilitzar quan s'especifica un nom relatiu.
- `set` *`opció`*: canvia el valor d'alguna opció utilitzada pel programa. Si el valor és `recurse`, les consultes s'han de fer en mode recursiu. Si el valor és `norecurse`, en mode no recursiu. Si el valor és `vc`, s'han d'utilitzar "circuits virtuals"; és a dir, el TCP. Si el valor és `novc`, s'han d'utilitzar datagrames, o el que és el mateix, l'UDP, etc.
- `set all`: mostra el valor actual dels paràmetres amb què treballa el programa (servidor per defecte, opcions, llista de nodes origen que s'han d'utilitzar quan es consulten noms de domini relatius, etc.).
- ?: mostra un missatge d'ajuda.
- `exit`: acaba el programa.

Activitat

Utilitzeu el programa `nslookup` per a respondre aquestes preguntes:

a) Quina és l'adreça de l'ordinador `www.uoc.es`?

b) Quin ordinador o ordinadors tenen per àlies `ftp.uoc.es`?

ANOTACIONS

c) Quin és el servidor DNS del domini `uoc.es`? I el del domini `uoc.edu`?

d) Quin és el servidor del domini de correu `campus.uoc.es`?

ANOTACIONS

17. Serveis bàsics d'Internet

17.1. Terminal virtual: el protocol Telnet

Un dels serveis bàsics per als quals es pot utilitzar una xarxa de comunicacions entre computadors és l'accés per mitjà d'un terminal. Quan es van desenvolupar els sistemes multiusuari, la manera habitual de treballar-hi consistia a utilitzar un equip terminal (en un inici, un teletip i, més endavant, un dispositiu amb teclat i pantalla) connectat directament a l'ordinador, en general per mitjà d'una línia sèrie. L'ordinador havia de saber quins terminals tenia connectats i com els podia controlar.

Nota

El fet que en una primera època s'utilitzessin teletips és el motiu pel qual en UNIX es va utilitzar l'abreviació `tty` (*teletype*) per a designar els terminals (bàsicament, impressores amb teclat).

Amb l'arribada de les xarxes de comunicacions, que permetien interconnectar diferents ordinadors, un dels seus usos naturals era mantenir una sessió de treball interactiva amb un ordinador remot sense necessitat d'utilitzar els terminals que hi estiguessin connectats directament. Per a això, era necessari establir un protocol que, d'una banda, fes possible que l'ordinador remot i el sistema en el qual volia treballar l'usuari es poguessin comunicar i s'entenguessin (és a dir, un protocol en l'àmbit d'aplicació, ja que el problema del transport se suposa que ja estava resolt) i, d'altra banda, facilités el control del terminal de l'usuari des de l'ordinador remot.

Un ordinador pot permetre que s'hi accedeixi des de qualsevol terminal de qualsevol altre ordinador connectat a la xarxa; tanmateix, no és pràctic que hagi de saber com es controlen tots els tipus possibles de terminal. La solució general a aquests requisits es basa en l'ús d'un protocol de terminal virtual.

Un terminal virtual és un dispositiu imaginari per al qual es defineixen unes funcions de control canòniques, de manera que es pot establir una correspondència entre elles i les de cada tipus de terminal real.

ANOTACIONS

Lectura complementària

Si voleu més informació sobre el protocol Telnet, consulteu l'obra següent:

J. Postel; J.K. Reynolds (1983, 1 de maig). *RFC 854 - Telnet Protocol Specification.*

En l'entorn Internet, el protocol de terminal virtual més utilitzat és Telnet, l'especificació del qual es va publicar el 1983 en l'estàndard RFC 854.

17.1.1. Principis bàsics del protocol Telnet

El protocol Telnet es basa en el protocol de transport TCP. En una comunicació Telnet, per norma general se segueix el model client/servidor; és a dir, el sistema usuari estableix una connexió amb el sistema proveïdor, que espera peticions de connexió en un port determinat. Es pot utilitzar qualsevol número de port per a les connexions i, de fet, hi ha moltes aplicacions que utilitzen el protocol Telnet per a la comunicació, cada una amb el seu propi número. L'aplicació bàsica consisteix a establir una sessió de treball interactiva amb el sistema servidor i, en aquest cas, el número de port utilitzat és el 23.

Per a resoldre el problema del control del terminal en l'aplicació de sessions interactives, en el protocol Telnet s'utilitza el concepte de **terminal virtual de xarxa** o **NVT**. Un NVT és un terminal virtual amb una funcionalitat molt bàsica, definida en la mateixa especificació del protocol.

Nota

L'usuari del terminal pot ser una persona o un procés.

Les dades llegides en el primer cas, per norma general, seran els caràcters teclejats i, en el segon, els que generi el procés. La presentació de les dades rebudes pot consistir a escriure-les a la pantalla, enviar-les a la impressora o passar-les al procés usuari.

Quan s'estableix la connexió entre el client i el servidor, en un inici se suposa que la comunicació es produeix entre dos NVT. Això significa que tant el sistema client com el sistema servidor han d'aplicar les seves característiques en les d'un NVT i suposar que en l'altre extrem de la connexió hi ha un altre NVT. En el mode d'operació normal, cada terminal accepta dades de l'usuari i les envia per mitjà de la connexió establerta en l'altre terminal, i també accepta les dades que arriben per la connexió i les presenta a l'usuari.

Les característiques principals dels NVT són les següents:

a) Les dades s'intercanvien en bytes de 8 bits. Quan representen caràcters, s'utilitza la codificació ASCII (per tant, s'envia un codi de 7 bits en cada byte).

ANOTACIONS

b) Hi ha tres caràcters de control que l'NVT sempre interpreta de la manera següent:

- NUL (caràcter de control nul, codi 0): no és necessari fer res.
- LF (avançament de línia, codi 10): moure la posició d'escriptura actual (en una impressora, el carro, i en una pantalla, el cursor) una línia endavant mantenint la posició horitzontal.
- CR (retorn de carro, codi 13): moure la posició d'escriptura enrere fins al principi de la línia actual.

Per tant, un final de línia (és a dir, un salt al principi de la línia següent) es podria representar amb les seqüències CR-LF o LF-CR. Tanmateix, per a facilitar el mapatge en terminals que no tenen funcions separades per a l'avançament de línia i el retorn de carro, el protocol Telnet especifica que els finals de línia s'han de representar amb la seqüència CR-LF, i els retorns de carro sense moure la posició vertical, amb la seqüència CR-NUL. D'aquesta manera, quan el terminal rebi un caràcter CR, podrà esperar que arribi el següent, que haurà de ser LF o NUL, per saber quina acció ha de dur a terme.

c) Hi ha cinc caràcters més que opcionalment pot interpretar l'NVT i el significat dels quals és el següent:

- BEL (timbre, codi 7): generar un senyal que, en general, sigui audible, però que alternativament pugui ser visible, sense moure la posició actual.
- BS (retrocés, codi 8): moure la posició horitzontal un espai enrere.
- HT (tabulació horitzontal, codi 9): moure la posició horitzontal cap endavant fins a la posició de tabulació següent.
- VT (tabulació vertical, codi 11): moure la posició vertical cap endavant fins a la posició de tabulació vertical següent.
- FF (avançament de pàgina, codi 12): moure la posició vertical fins al principi de la pàgina següent, sense moure la posició horitzontal.

Qualsevol altre caràcter de control es considera equivalent a NUL.

ANOTACIONS

d) El dispositiu NVT és *full duplex*; tot i que, per norma general, treballa en manera *half duplex*; és a dir, només envia les dades de l'usuari quan ha llegit una línia completa o quan rep algun altre senyal local que provoqui la transmissió.

e) Si l'usuari, de moment, no disposa de més dades per a enviar, i s'han processat totes les dades rebudes, el terminal pot transmetre al sistema remot un senyal, denominat *Go Ahead* (GA), per a indicar-li que és el seu torn d'enviar dades.

Nota

Només s'hauria d'utilitzar el senyal GA quan l'altre sistema no tingués cap altra manera de saber que no ha d'esperar més dades. Una situació típica és que l'usuari (client) generi línies d'una en una i el sistema remot (servidor) enviï respostes de zero, una o més línies a cada una. En aquest cas, només serà necessari que el servidor enviï el senyal GA per a notificar que el control passa a l'usuari del terminal, i no serà necessari que el client li ho enviï després de cada línia.

f) L'NVT també pot generar, a instàncies de l'usuari, dos senyals més, denominats *Break* (BRK) i *Synch*. El significat del primer depèn de l'aplicació. El segon serveix per a cancel·lar les dades que hagi enviat el terminal i encara no hagin estat processades pel sistema remot.

g) Així mateix, hi ha cinc funcions de control que pot generar l'NVT a instàncies de l'usuari: *Interrupt Process* (IP), *Abort Output* (AO), *Are You There* (AYT), *Erase Character* (EC) i *Erase Line* (EL).

Nota

Veurem el significat de les funcions de control en l'apartat 18.1.2.

Si el client i el servidor suporten una funcionalitat més avançada que la d'un NVT, que és el més habitual, el protocol permet que duguin a terme una negociació per a posar-se d'acord sobre quines característiques diferents de les d'un NVT utilitzaran en la comunicació.

La **negociació** consisteix a intercanviar codis que indiquen les opcions del protocol que cada part vol o està disposada a utilitzar.

ANOTACIONS

Els quatre codis que s'utilitzen per a implementar la negociació són els següents:

- DO és una petició que s'envia per a demanar que s'utilitzi una opció determinada.
- WILL és un oferiment que s'envia per a indicar que el sistema està preparat per a utilitzar una opció determinada.
- DON'T significa que el sistema no vol que s'utilitzi l'opció indicada.
- WON'T significa que el sistema no està preparat per a utilitzar l'opció indicada.

Nota

El mecanisme de negociació és simètric, de manera que qualsevol de les dues parts pot enviar un DO o un WILL. Després d'enviar un DO, es pot rebre un WILL com a resposta positiva o un WON'T com a resposta negativa. De la mateixa manera, després d'enviar un WILL, es pot rebre un DO com a resposta positiva o un DON'T com a resposta negativa. Això significa que un DO pot actuar com a petició o com a confirmació si l'altra part ha enviat un WILL, i viceversa.

Per norma general, la negociació de les opcions es produeix a l'inici de la connexió, però també és possible efectuar-la en qualsevol altre moment. Cada sistema ha de començar a utilitzar una opció quan envia o rep la resposta positiva corresponent.

Aquest mecanisme de negociació està dissenyat perquè sigui fàcilment extensible. Si un sistema no reconeix una opció determinada que se li demana o se li ofereix, simplement ha de contestar amb WON'T o DON'T.

A vegades es necessita més informació per a negociar una opció, com el valor d'algun paràmetre. En aquest cas, en primer lloc s'han de posar d'acord les dues parts amb el mecanisme bàsic dels DO/DON'T/WILL/WON'T per a utilitzar l'opció desitjada i, una vegada s'hagi acceptat, s'ha de dur a terme una **subnegociació** dels valors dels paràmetres utilitzant la sintaxi específica de cada opció.

Nota

Les opcions telnet possibles no estan definides en l'especificació del protocol, sinó en especificacions independents.

Nota

Moltes opcions defineixen les seves ordres per a la subnegociació, com ara IS (per a enviar el valor d'un paràmetre), SEND (per a demanar que l'altra part l'enviï), etc.

17.1.2. Ordres del protocol Telnet

En les dades intercanviades entre dos sistemes que es comuniquen amb el protocol Telnet, es poden intercalar certes ordres pròpies del protocol, com ara el senyal GA o els codis de negociació. Per a distingir les ordres de les dades normals, les primers han d'anar prefixades amb un codi especial anomenat *IAC* (Interpret As Command), que es representa amb un byte igual a 255.

Per tant, cada ordre es representa amb una seqüència de dos bytes, en la qual el primer és el codi IAC, i el segon, el codi mateix de l'ordre, excepte les ordres de negociació, que disposen d'un tercer byte que serveix per a indicar a quina opció es refereixen. Per a representar un byte normal de dades que sigui igual a 255, és necessari prefixar-lo amb un codi IAC. Qualsevol altre byte de dades es representa directament amb el seu codi.

Les ordres definides en el protocol Telnet són les següents:

- NOP (*No Operation*, codi 241): operació nul·la.
- GA (*Go Ahead*, codi 249): senyal GA.
- BRK (*Break*, codi 243): senyal BRK.
- DO (codi 253) + codi opció: codi de negociació DO.
- DON'T (codi 254) + codi opció: codi de negociació DON'T.
- WILL (codi 251) + codi opció: codi de negociació WILL.
- WON'T (codi 252) + codi opció: codi de negociació WON'T.
- SB (*Subnegotiation Begin*, codi 250) + codi opció: les dades que vinguin a continuació corresponen a la subnegociació d'una opció.
- SE (*Subnegotiation End*, codi 240): indica el final de les dades de subnegociació.
- DM (*Data Mark*, codi 242): indica en quin punt de la seqüència de dades s'ha enviat un senyal *Synch*.
- IP (*Interrupt Process*, codi 244): codi de funció que pot enviar l'usuari per a indicar al sistema remot que interrompi el procés que està executant.

ANOTACIONS

- AO (*Abort Output*, codi 245): codi de funció que serveix per a indicar que es continuï executant el procés remot, però que deixi d'enviar a l'usuari les dades que genera.

- AYT (*Are You There*, codi 246): codi de funció que demana al sistema que, quan el rebi, contesti amb algun missatge que serveixi a l'usuari per a comprovar que la connexió continua establerta (per exemple, quan el procés que executa triga a generar una resposta).

- EC (*Erase Character*, codi 247): codi de funció que serveix per a esborrar l'últim caràcter escrit al terminal.

- EL (*Erase Line*, codi 248): codi de funció que serveix per a esborrar tota la línia actual.

El protocol Telnet també proporciona un mecanisme per transmetre el senyal Synch. Aquest últim no es representa amb una ordre com les altres, ja que ha de tenir un efecte immediat. En aquest cas, s'envien dades urgents TCP acabades amb un codi DM i, en la seqüència de dades normals, s'insereix un altre codi DM. En rebre les dades urgents, el sistema remot passarà a processar les dades normals en mode urgent (que pot consistir a descartar tot el que hi hagi en aquestes dades, llevat de les ordres especials) fins que trobi el DM.

El terminal pot generar automàticament un senyal *Synch* quan l'usuari enviï alguna de les funcions que requereixin atenció immediata, com ara IP, AO o AYT (però no EC o EL).

17.1.3. Implementacions del protocol Telnet

Hi ha implementacions de servidors Telnet pràcticament en tots els sistemes multiusuaris que utilitzen el protocol TCP. Els clients Telnet són més nombrosos encara, perquè també n'hi ha per a sistemes uniusuari.

Un exemple d'implementació de client Telnet és la utilitat del sistema operatiu GNU/Linux denominada precisament `telnet`. Si es crida sense arguments, entra en mode comanda. Amb un argument, que

ANOTACIONS

pot ser una adreça IP o un nom de servidor, estableix una connexió amb el port Telnet (el 23) d'aquest servidor i entra en mode connexió. Amb un segon argument, que pot ser un número de port o un nom de servei, estableix la connexió amb aquest port.

Quan el número de port que s'utilitza és el 23, el client inicia automàticament el procés de negociació enviant els codis `DO` i `WILL` corresponents a les opcions que suporta. Amb qualsevol altre port, per norma general no s'envia cap codi de negociació, llevat que se'n rebi algun del sistema remot.

Quan el programa està en mode connexió, envia al sistema remot cada caràcter que tecleja l'usuari. Des del mode connexió es pot passar al mode comanda per mitjà del caràcter d'escapada, que sol ser "`^]`". En aquest mode, el programa admet, entre altres, les comandes següents:

Nota

Els noms de les ordres es poden abreujar sempre que l'abreviatura no generi ambigüitats.

- `open`: estableix una connexió amb el servidor, i opcionalment amb el port, indicat en els arguments de la comanda.
- Comanda nul·la (línia buida): si hi ha una connexió establerta, surt del mode comanda i torna al mode connexió.
- `send`: envia al sistema remot el codi `Telnet` indicat per l'argument, que pot ser `ao`, `ayt`, `brk`, `ec`, `el`, `ga`, `ip`, `synch`, etc.
- "`^]`" (o el caràcter que actuï com a caràcter d'escapada): envia aquest últim al sistema remot (equival a `send escape`).
- `?`: mostra un missatge d'ajuda.
- `quit`: tanca la connexió, si n'hi ha cap d'establerta, i acaba el programa (que també acaba quan el sistema remot tanca la connexió).

Activitat

En algunes implementacions Unix del client Telnet, es pot utilitzar l'ordre `toggle netdata` per a veure el diàleg de la negociació. Proveu aquesta altra opció en una sessió interactiva (al port per defecte 23) i en una connexió a un altre port (per exemple, `echo`).

ANOTACIONS

17.2. Terminal virtual en GNU/Linux: el protocol rlogin

Quan es vol establir una sessió de treball interactiva des d'un sistema Unix amb un altre sistema Unix, a més d'utilitzar el protocol Telnet, també hi ha la possibilitat d'utilitzar el protocol anomenat **rlogin**. Aquest protocol es va desenvolupar en un inici en l'entorn de les utilitats de comunicacions de la variant de Unix anomenada *BSD*; avui dia n'hi ha implementacions disponibles en totes les variants, i també en altres sistemes operatius. L'especificació del protocol rlogin està publicada en el document *RFC 1282*.

Les característiques principals que diferencien el protocol rlogin del protocol Telnet són les següents:

Lectura complementària

Per a obtenir més informació sobre el protocol rlogin, consulteu l'obra següent:

B. Kantor (1991, desembre). *RFC 1282 - BSD Rlogin*.

a) El servidor sempre és informat del tipus de terminal amb què treballa el client sense necessitat de negociació.

b) El servidor també és informat de la identitat de l'usuari en el sistema client, cosa que es pot utilitzar per a automatitzar el procés d'autenticació.

c) Si canvien les mides de la finestra de presentació del terminal, es pot enviar automàticament un senyal al servidor per a notificar-li-ho.

d) En la transmissió de dades, sempre es poden utilitzar els 8 bits de cada byte.

17.2.1. Conceptes bàsics del protocol rlogin

El protocol rlogin utilitza connexions TCP. El nom oficial del servei proporcionat per aquest protocol és `login` i el número de port assignat a aquest servei és el 513.

Quan el client estableix la connexió amb el servidor, en primer lloc li envia quatre cadenes de caràcters, cada una acabada amb un caràcter NUL (codi 0). Són les cadenes següents:

- Una cadena buida (només conté el caràcter NUL).

ANOTACIONS

- El nom de l'usuari en el sistema client.
- El nom d'usuari amb què es vol establir la sessió de treball en el sistema servidor.
- El tipus de terminal i, separat amb /, la seva velocitat (per exemple, vt100/9600).

Quan ha rebut aquestes quatre cadenes, el servidor envia un caràcter NUL i comença la transferència de dades entre client i servidor en mode control de flux. En aquest mode, el client envia els caràcters que tecleja l'usuari tal com li arriben, excepte els caràcters de control DC3 ("^S") i DC1 ("^Q"), que signifiquen 'suspendre la transmissió' i 'reprendre-la', respectivament.

D'altra banda, el client presenta les dades que envia el servidor tal com li arriben. El client també pot rebre missatges de control del servidor, que es representen amb un codi d'un byte enviat en dades urgents TCP. El client ha de respondre aquests missatges de manera immediata i suspendre temporalment el processament d'altres dades que pugui haver rebut.

17.2.2. Implementació del protocol rlogin

En els sistemes GNU/Linux hi ha una utilitat anomenada `rlogin` que actua com a client del protocol rlogin. Cal especificar-li el nom del servidor remot i, optativament, el nom d'usuari que ha d'utilitzar en aquest servidor, amb l'opció `-l` (per defecte, s'utilitza el mateix nom d'usuari que en el sistema local).

La majoria dels servidors rlogin implementen l'autenticació automàtica de la manera següent:

- Si l'usuari remot té en el seu directori un fitxer anomenat `.rhosts` amb una línia en la qual hi ha el nom de l'ordinador client i, opcionalment, el nom de l'usuari client (en absència d'aquest nom d'usuari, es pren el de l'usuari al servidor), s'inicia directament la sessió interactiva sense necessitat de demanar cap contrasenya.

ANOTACIONS

- Si no disposa d'aquest fitxer, se'n consulta un altre, el `/etc/hosts.equiv`, per a veure si existeix alguna línia amb el nom de l'ordinador client (en aquest cas, ha de coincidir el nom de l'usuari en el client i en el servidor) i, si hi és, també es dóna l'usuari per autenticat.

- En qualsevol altre cas, se segueix el procediment normal d'autenticació mitjançant contrasenya.

> ***Nota***
>
> Per motius de seguretat, els servidors rlogin solen requerir que el fitxer `.rhosts` tingui permís de lectura només per a l'usuari a qui pertany. D'altra banda, els noms de servidors als fitxers `.rhosts` i `/etc/hosts.equiv` no poden ser àlies, han de ser els noms oficials. I com a precaució addicional, el fitxer `/etc/hosts.equiv` només s'utilitza quan el nom de l'usuari al servidor és diferent de `root`.

Abans de consultar els fitxers `.rhosts` i `/etc/hosts.equiv`, el servidor sempre comprova que el client s'hagi connectat des d'un port reservat (amb un número inferior a 1.024). En cas contrari, no es permet l'autenticació automàtica, ja que la connexió podria provenir d'un altre usuari que executés un programa que seguís el protocol rlogin, però que enviés informació falsa sobre la seva identitat.

L'autenticació automàtica pot resultar còmoda perquè estalvia teclejar les contrasenyes, però també pot ser una font de problemes si no s'utilitza amb precaució.

A més de llegir els caràcters teclejats i enviar-los al servidor, el client també pot rebre comandes de l'usuari. Per a distingir-los dels caràcters normals, es representen amb un caràcter precedit del símbol "~". Aquesta seqüència només es reconeix a l'inici de les línies (és a dir, immediatament a continuació d'un salt de línia). El caràcter de la comanda pot ser, per exemple, el següent:

- "^Z" (o el caràcter corresponent a la funció SUSP del terminal): suspèn l'execució del programa client, amb la possibilitat de re-

ANOTACIONS

prendre-la més endavant (la manera de fer-ho depèn de l'intèrpret de comanda o *shell*, però en general és amb l'ordre `fg`).

- "^Y" (o el caràcter corresponent a la funció DSUSP del terminal): suspèn la tramesa de dades al servidor i, per tant, el teclat queda disponible per a altres processos, tot i que la recepció continua (si es reben dades noves, es mostraran quan arribin).
- "~": envia el caràcter ~.
- "`.`": tanca la connexió.

Si el caràcter no correspon a cap ordre reconeguda pel client, s'envia al servidor tal com ha arribat, precedit pel caràcter "~."

17.3. Altres serveis

17.3.1. Execució remota amb autenticació automàtica: rsh

Paral·lelament al servei de terminal virtual, que permet mantenir una sessió de treball en un sistema remot, en els sistemes Unix BSD es va desenvolupar un altre servei que permet executar una ordre en el sistema remot. El programa que actua com a client d'aquest servei s'anomena **rsh** (*remote shell*).

El nom oficial d'aquest servei és `shell`, el protocol de transport utilitzat és el TCP i el número de port assignat és el 514.

A continuació presentem els passos que se segueixen en aquest protocol:

1. El servidor comprova que el client s'hagi connectat des d'un port reservat. En cas contrari, tanca la connexió.
2. El client envia al servidor una cadena acabada en `NUL` que conté un nombre decimal:

- Si és diferent de zero, s'interpreta com un número de port i el servidor estableix en aquest port una connexió secundària (aquesta segona connexió s'estableix també des d'un port reservat).

ANOTACIONS

- Si és zero, la mateixa connexió principal actua també com a secundària.

3. El client envia tres cadenes més acabades en NUL amb el nom de l'usuari en el sistema servidor, el nom en el sistema client i la comanda que s'ha d'executar.

4. El servidor comprova en els fitxers `.rhosts` i/o `/etc/hosts.equiv` que l'usuari estigui autoritzat per a l'autenticació automàtica.

5. Si ho està, el servidor envia un caràcter NUL per la connexió secundària; en cas contrari, tanca les connexions.

6. El servidor executa la comanda indicada amb la identitat de l'usuari indicat. Les dades que escrigui aquesta comanda per la sortida estàndard (`stdout` en la nomenclatura del llenguatge C) s'enviaran per la connexió principal, i els que escrigui per la sortida d'error (`stderr`), per la connexió secundària. Les que enviï el client seran accessibles per l'entrada estàndard (`stdin`).

7. Quan acabi l'execució de la comanda, el servidor tancarà la connexió.

17.3.2. Execució remota: rexec

Amb rsh, l'usuari ha d'estar autoritzat per mitjà dels fitxers `.rhosts` o `/etc/hosts.equiv` del sistema remot per a poder-hi executar una comanda. Hi ha un altre servei molt semblant en què no s'utilitza l'autenticació automàtica, sinó que sempre requereix que l'usuari proporcioni la seva contrasenya en el sistema remot. En alguns sistemes Unix, hi ha un programa anomenat **`rexec`** que actua com a client d'aquest servei. En d'altres, només hi ha una funció de la llibreria estàndard, `rexec`, que implementa el protocol.

El nom oficial d'aquest servei és `exec`, el protocol de transport utilitzat és el TCP i el número de port assignat és el 512.

Aquest protocol és molt semblant a l'anterior: les úniques diferències són que, en lloc d'enviar el nom de l'usuari, en el sistema client s'envia la contrasenya de l'usuari en el sistema servidor (els fitxers `.rhosts` i `/etc/hosts.equiv` no es consulten) i que no és necessari comprovar que les connexions provinguin de ports reservats.

ANOTACIONS

17.3.3. Serveis trivials

La majoria dels sistemes Unix i GNU/Linux proporcionen una sèrie de serveis denominats *trivials*, que es poden utilitzar per a dur a terme proves de funcionament dels mòduls que implementen els protocols de transport. Tots són accessibles per mitjà de TCP i UDP, i el número de port utilitzat en tots dos casos és el mateix. Alguns exemples de serveis trivials són els següents:

- `echo` (port 7): retorna tots els bytes (en TCP) o datagrames (en UDP) que rep (RFC 862).
- `discard` (port 9): llegeix dades (bytes o datagrames), però no fa res més (RFC 863).
- `chargen` (port 19): en TCP, quan s'estableix la connexió, el servidor comença a enviar una seqüència de caràcters fins que el client la tanca i, en UDP, quan el servidor rep un datagrama, respon amb un altre que conté un nom arbitrari de caràcters (RFC 864).
- `daytime` (port 13): quan s'estableix la connexió, o es rep un datagrama, el servidor envia la data i l'hora actual en format llegible per als humans (RFC 867).
- `time` (port 37): quan s'estableix la connexió, o es rep un datagrama, el servidor envia un número de 4 bytes que representa el nombre de segons transcorreguts des de l'1 de gener de 1970 a les 0 hores GMT (RFC 868).

Finger

`Name/Finger` és un altre exemple de servei trivial que permet obtenir informació d'un usuari en un sistema remot. L'especificació del protocol per a aquest servei està recollida en el document *RFC 742*.

El nom oficial d'aquest servei és `finger`, el protocol de transport utilitzat és el TCP i el número de port assignat és el 79.

El programa client s'anomena *finger* i, per norma general, admet arguments de la forma `usuari`, `usuari@host` o `@host`:

- En el primer cas, proporciona informació sobre un usuari en el sistema local sense necessitat d'utilitzar cap protocol.

ANOTACIONS

- En el segon, informa sobre un usuari d'un sistema remot.
- En el tercer, normalment ofereix una informació resumida dels usuaris que estan actualment connectats amb sessions interactives en el sistema remot.

Quan s'estableix la connexió, el client simplement envia una línia al servidor acabada amb CR-LF. En aquesta línea es troba el nom de l'usuari de qui es vol obtenir informació. Si no hi ha cap nom, s'entén que la sol·licitud es refereix a tots els usuaris que estiguin connectats.

El format de la resposta depèn de la implementació del servidor i pot incloure informació com el nom de l'usuari, el seu nom real, quan es va connectar per última vegada, quan va rebre o va llegir el correu per última vegada, etc., i, si està connectat actualment, des d'on, a quin terminal, quant de temps fa que no hi ha activitat al seu terminal, etc. La informació retornada normalment és un subconjunt de tots aquests camps; tanmateix, si la línia de petició comença amb els caràcters "/W", el servidor pot retornar informació més completa.

ANOTACIONS

18. Transferència de fitxers

18.1. FTP: protocol de transferència de fitxers

Una de les primeres aplicacions bàsiques desenvolupades en l'entorn del que després seria la xarxa Internet va ser la transferència de fitxers entre diferents sistemes. Al principi dels anys setanta ja es van elaborar les primeres especificacions del protocol més utilitzat per a aquesta finalitat, l'**FTP**. Després d'algunes modificacions i millores, l'especificació oficial del protocol es va publicar el 1985 en el document *RFC 959*.

L'FTP es basa en el model client/servidor i permet la transferència de fitxers tant del servidor al client, com del client al servidor. Així mateix, permet que un client efectuï transferències directes d'un servidor a un altre, amb la qual cosa s'estalvia la necessitat que s'hagin de copiar els fitxers del primer servidor al client i, després, passar-los del client al segon servidor.

Lectura complementària

Per a més informació sobre l'FTP, consulteu l'obra següent:

J. Postel; J. Reynolds (1985, octubre). *RFC 959 - File Transfer Protocol.*

El protocol proporciona també operacions perquè el client pugui manipular el sistema de fitxers del servidor: esborrar fitxers o canviar-los el nom, crear i esborrar directoris, llistar-ne els continguts, etc.

Un dels objectius principals d'aquest protocol consisteix a permetre la interoperabilitat entre sistemes molt diferents, amagant els detalls de l'estructura interna dels sistemes de fitxers locals i de l'organització dels continguts dels fitxers.

Nota

Alguns dels sistemes utilitzats en l'època de desenvolupament de l'FTP actualment són obsolets; per exemple, els que utilitzen paraules de 36 bits o codis

ANOTACIONS

de caràcters incompatibles amb l'estàndard ASCII o els que emmagatzemen els fitxers en "pàgines" no necessàriament contigües. De fet, moltes de les implementacions actuals del protocol no suporten aquestes característiques especials previstes en l'especificació.

18.1.1. El model de l'FTP

En el model general descrit en l'especificació de l'FTP, tant en el servidor com en el client, hi ha dues entitats que intervenen en la transferència de fitxers:

a) L'**intèrpret de protocol** s'encarrega de l'intercanvi de les comandes del protocol. En la part del client, les operacions que l'usuari sol·licita per mitjà de la interfície d'usuari, l'intèrpret les converteix en una seqüència adequada de comandes FTP i s'envien al servidor.

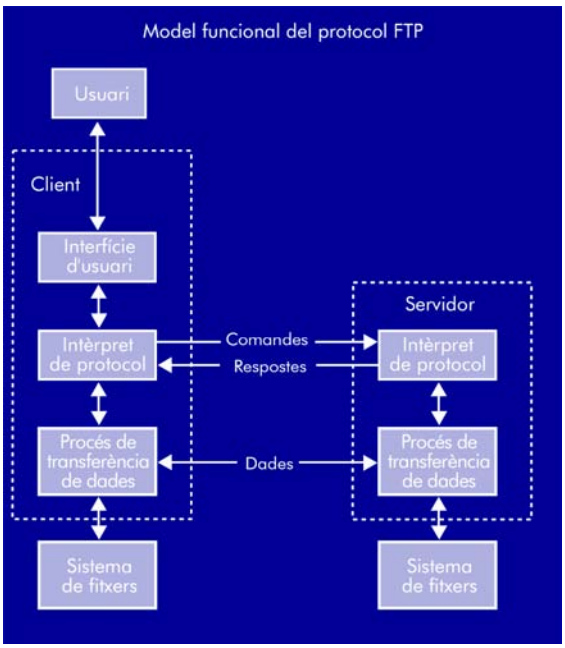
   A la part del servidor s'interpreten les comandes rebudes, es generen les respostes corresponents i s'envien al client. Per tant, l'intèrpret client ha d'analitzar aquestes respostes, per exemple, per informar l'usuari del resultat de l'operació o per a prosseguir la seqüència de comandes FTP.

b) El **procés de transferència de dades**, que està sota el control de l'intèrpret de protocol, s'encarrega d'intercanviar les dades que s'han de transferir; és a dir, els continguts dels fitxers o bé les llistes dels directoris.

   Tant en la part del client com en la del servidor, aquest procés interactua directament amb el sistema de fitxers locals per a llegir les seves dades o per a emmagatzemar-les-hi.

ANOTACIONS

Figura 79.



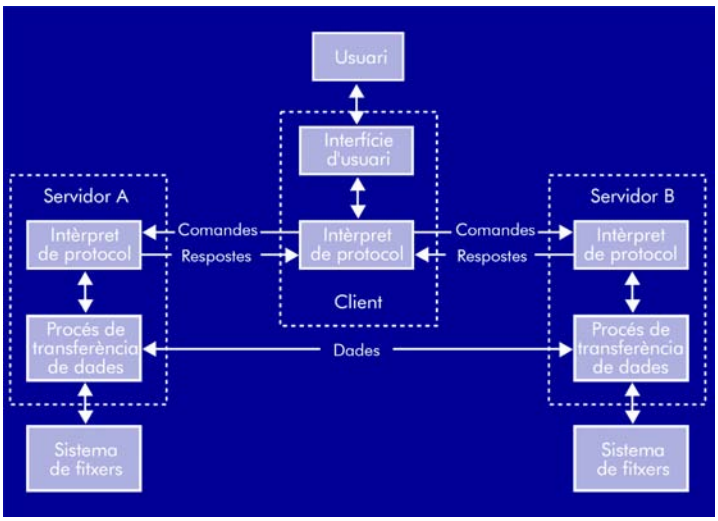
Els dos intèrprets de protocol es comuniquen mitjançant una connexió TCP anomenada *connexió de control*. Quan s'han de transferir dades d'un sistema a l'altre, s'estableix una altra connexió TCP entre els dos processos de transferència de dades denominada *connexió de dades*. Generalment, la part activa d'aquesta connexió (la que la inicia) constitueix el procés de transferència del servidor, i la part passiva, el del client.

La figura següent mostra la variació del model general per al cas en què un client controli una transferència entre dos servidors (uns dels quals actuarà com a servidor actiu i l'altre, com a servidor passiu):

Nota

L'FTP permet efectuar més d'una transferència en una única sessió. Segons el mode de transmissió, és possible que s'obrin i es tanquin diferents connexions de dades durant una mateixa connexió de control, o bé que s'utilitzi la mateixa connexió de dades per a les diferents transferències.

Figura 80.



ANOTACIONS

18.1.2. Conceptes bàsics de l'FTP

L'FTP es basa en connexions TCP. L'intèrpret de protocol del servidor ha d'estar preparat per a rebre peticions de connexió en un port TCP que, per defecte, és l'assignat oficialment al servei FTP: el número 21.

L'intèrpret de protocol del client estableix una connexió de control amb el port de l'intèrpret servidor. En aquesta connexió s'utilitzen les regles especificades en el protocol Telnet. Això significa que, per defecte, els intèrprets de protocol s'intercanvien missatges codificats amb bytes de 8 bits, segons el joc de caràcters ASCII, i representen els finals de línia amb la seqüència <CRLF>.

Nota

Consulteu la descripció de les regles especificades pel protocol Telnet (RFC 854) en l'apartat 18.1 de la unitat anterior.

Les **comandes FTP** són els missatges que envia l'intèrpret client, i les que envia l'intèrpret servidor són respostes a aquestes comandes. Les respostes es generen en l'ordre en què el client envia les comandes, ja que en general el servidor efectua les operacions de manera seqüencial (no comença una nova operació fins que no ha acabat l'anterior). A continuació, analitzem separadament les comandes i les respostes:

Nota

Les excepcions a la regla de la resposta seqüencial són les comandes obtenir l'estat actual d'una transferència, avortar una operació i tancar la sessió.

1. **Comandes FTP**: una comanda FTP es representa per mitjà d'un codi de comanda de fins a quatre lletres (que poden ser indistintament majúscules o minúscules), seguit d'una llista de zero o més arguments, separats per espais, acabada amb un final de línia.

Nota

Algunes implementacions de servidors FTP suporten altres codis de comandes no estàndards, que poden tenir més de quatre lletres.

L'especificació RFC 959 defineix trenta-tres comandes agrupades en tres categories diferents (representem entre parèntesis el codi corresponent a cada comanda):

a) **Comandes de control d'accés**: nom d'usuari (USER), contrasenya (PASS), compte (ACCT), canviar de directori (CWD), canviar al directori pare (CDUP), muntar un sistema de fitxers (SMNT), reinicialitzar (REIN), acabar la sessió (QUIT).

b) **Comandes de paràmetres de transferència**: estructura de fitxer (STRU), mode de transmissió (MODE), tipus de representació (TYPE), port de dades (PORT), port passiu (PASV).

ANOTACIONS

c) **Comandes de servei FTP**: obtenir (RETR), emmagatzemar (STOR), emmagatzemar amb nom únic (STOU), afegir (APPE), fer una llista (LIST), fer una llista de noms (NLST), mostrar el directori actual (PWD), reservar espai (ALLO), avortar (ABOR), reprendre (REST), esborrar (DELE), nom antic (RNFR), nom nou (RNTO), crear un directori (MKD), esborrar un directori (RMD), estatus (STAT), sistema (SYST), serveis addicionals (SITE), ajuda (HELP) i operació nul·la (NOOP).

2. **Respostes FTP**: les respostes consten d'un codi numèric de tres dígits decimals, que permet a l'intèrpret del protocol client saber quin és el resultat de l'operació, seguit d'un text explicatiu destinat a l'usuari humà.

El text explicatiu del missatge de resposta pot tenir una línia o més:

- Si té una línia, la resposta es representa amb el codi numèric seguit del caràcter espai i, a continuació, el text i un final de línia.
- Si té més d'una línia, la primera es representa amb el codi numèric seguit del caràcter "-" i, a continuació, les línies de text, acabades amb finals de línia, l'última de les quals ha de començar amb el mateix codi numèric de la primera línia seguit del caràcter espai.

Nota

D'aquest grup de comandes de servei, les sis següents són les comandes de transferència; és a dir, les que utilitzen la connexió de dades:

- RETR
- STOR
- STOU
- APPE
- LIST
- NLST

Exemple

Exemple de resposta d'una sola línia:

```
220 Sistema preparat. Introduïu nom
d'usuari i contrasenya.
```

Exemple de resposta amb múltiples línies:

```
220- Sistema preparat.
     Introduïu el nom d'usuari i la
     contrasenya per accedir als fitxers
     restringits, o utilitzeu el nom
     "anonymous"
220 per a accedir al directori públic.
```

ANOTACIONS

Nota

L'estructura dels codis de resposta i el significat de cada dígit són comuns a tots els protocols. En podeu trobar una explicació en l'annex 4.

L'especificació RFC 959 defineix els codis de resposta possibles per a cada una de les comandes i el seu significat:

Taula 8.

Codis de resposta	
Codi	**Significat**
110	Marca de represa.
120	Esperar fins que el servidor estigui preparat.
125	El servidor transfereix dades (la connexió de dades ja és oberta).
150	El servidor estableix la connexió de dades i inicia la transferència.
200	Operació efectuada.
202	Comanda innecessària en aquest servidor.
211	Informació d'estat del sistema o missatge d'ajuda del sistema.
212	Informació d'estat del directori.
213	Informació d'estat del fitxer.
214	Missatge d'ajuda (destinat a l'usuari humà).
215	Tipus de sistema.
220	Servidor preparat.
221	El servidor tanca la connexió de control.
226	Operació efectuada i connexió de dades tancada.
227	El servidor està en mode passiu (*h1,h2,h3,h4,p1,p2*).
230	Procés d'autenticació completat satisfactòriament.
250	Operació de fitxer efectuada.
257	El directori és el resultat de l'operació.
331	Enviar contrasenya.
332	Enviar compte.
350	Enviar la comanda següent per a completar l'operació.
421	Servei no disponible (per exemple, per *time out*...), connexió de control tancada.
425	No es pot establir la connexió de dades.
426	Transferència avortada i connexió de dades tancada.
450	No es pot efectuar l'operació sobre el fitxer (per exemple, fitxer ocupat).
452	Espai de disc insuficient: transferència no iniciada.
500	Error de sintaxi en la comanda.

ANOTACIONS

Codis de resposta	
Codi	**Significat**
501	Error en els arguments.
502	Comanda no implementada.
503	Error en la seqüència d'ordres (aquesta comanda no correspon).
504	Argument no suportat.
530	L'usuari no s'ha autenticat correctament.
532	Es necessita compte per a aquesta operació.
550	No es pot accedir al fitxer o directori (no existeix, accés denegat, etc.).
552	Espai de disc insuficient: transferència avortada.
553	Nom de fitxer no permès.

El client, un cop establerta la connexió de control amb l'intèrpret de protocol del servidor i abans d'enviar cap comanda, ha d'esperar a rebre una resposta, que pot tenir els codis 220, 421 o 120. Si el codi és 120, el client ha d'esperar fins que el servidor enviï una resposta 220. Quan el servidor indiqui que està preparat, pot començar l'intercanvi de comandes i respostes. Per norma general, el servidor tancarà la connexió de control quan li ho sol·liciti el client (amb la comanda d'acabar sessió).

Nota

El text explicatiu de les respostes és de format lliure, excepte en el cas de les respostes 110 (comanda RETR), 227 (comanda PASV) i 257 (comanda PWD i MKD).

Si el client vol realitzar **operacions de transferència**, el seu procés de transferència de dades hauria d'estar preparat per a rebre peticions de connexió en un port TCP que, per defecte, és el mateix des del qual l'intèrpret ha iniciat la connexió de control.

L'encarregat d'establir les connexions de dades és el procés de transferència de dades del servidor, des del seu port de dades. Per defecte, aquest port és $L - 1$ (on L és el número de port corresponent a la connexió de control). És a dir, si el servidor ha rebut la connexió de control al port 21, el seu port de dades per defecte serà el 20. Evidentment, el client pot sol·licitar l'ús de ports diferents dels predeterminats per mitjà de les comandes adequades.

Generalment, les connexions de dades les tanca el servidor, excepte quan sigui l'altra part (el client o un servidor passiu) qui enviï les dades i el mode de transmissió utilitzat requereixi el tancament de la connexió per indicar el final del fitxer.

ANOTACIONS

18.1.3. Funcionalitat de l'FTP

Les accions que duu a terme el servidor per a cada una de les comandes definides en l'especificació RFC 959 són les següents:

1. Nom d'usuari (**USER**)

L'argument d'aquesta comanda és el nom que cal donar al sistema servidor per a identificar l'usuari amb l'objectiu d'accedir als fitxers. Aquesta sol ser la primera comanda que envia el client. Així mateix, és possible enviar-la al mig d'una sessió; llavors el servidor s'oblida de la identitat de l'usuari anterior i realitza les noves operacions sota el nom de l'usuari nou.

Si el servidor envia la resposta 230, significa que el procés d'autenticació ja s'ha completat; si envia la 530, significa que aquest usuari no és admissible (per exemple, perquè no hi ha cap usuari amb aquest nom), i si envia la 331 o la 332, llavors es necessita una contrasenya o un compte, respectivament.

> ***Nota***
>
> En molts sistemes, un intent de connexió amb un nom d'usuari inexistent donarà com a resultat una resposta 331, per a no proporcionar informació als usuaris aliens al sistema sobre quins noms són vàlids i quins no.
>
> El codi 332 indica que l'operació sol·licitada es portarà a terme tan aviat com es rebi l'ordre ACCT.

2. Contrasenya (**PASS**)

L'argument d'aquesta comanda és la contrasenya que necessita el sistema servidor per a comprovar la identitat de l'usuari. Si no se'n necessita cap, la resposta serà 202; si se'n necessita però és incorrecta, 530, i si és correcta, pot ser 230 (autenticació completada) o 332 (es necessita un compte).

3. Compte (**ACCT**)

Alguns sistemes poden requerir que l'usuari proporcioni una identificació de compte. Aquesta identificació pot ser necessària per al procés

ANOTACIONS

d'autenticació inicial, com per a efectuar determinades operacions, com ara emmagatzemar fitxers. El servidor farà saber que necessita aquesta informació enviant una resposta 332 en el procés d'autenticació, o una resposta 332 o 532 després d'una operació determinada.

4. Estructura de fitxer (STRU)

Aquesta comanda serveix per a especificar com estaran estructurats els fitxers que s'hagin de transferir. El tipus d'estructura afecta el mode de transmissió, com veurem a continuació. Els valors possibles de l'argument són els tres següents:

- R: el fitxer consta d'una seqüència de registres.
- P: l'estructura es basa en pàgines indexades. (L'opció P era útil en els sistemes de l'època en què es va desenvolupar l'FTP; avui dia està pràcticament en desús.)
- F: el fitxer es considera simplement una seqüència de bytes (en aquest cas es considera que només hi ha estructura de fitxer).

Si no s'utilitza la comanda STRU, el tipus d'estructura per defecte és F.

5. Mode de transmissió (MODE)

L'argument indica com es transmetran els fitxers. Pot tenir els valors següents:

- B: la transmissió es duu a terme en blocs de dades, cada un precedit d'una capçalera amb la longitud del bloc (2 bytes) i un codi descriptor (1 byte). Aquest últim serveix per a indicar, per exemple, si el bloc és l'últim d'un registre o del fitxer.
- C: la transmissió es realitza en blocs més compactes, amb capçaleres d'un sol byte, i permeten codificar una seqüència de fins a 64 bytes repetits en un bloc de 2 bytes.
- S: la transmissió s'efectua en mode *stream*; és a dir, com una simple seqüència de bytes. Si s'utilitza amb el tipus d'estructura en registres, s'insereixen codis de control en la seqüència per a assenyalar els finals de registre i de fitxer. Si el tipus d'estructura és de fitxer i la

Nota

La combinació mode *stream* i estructura de fitxer és l'única que requereix tancar la connexió de dades després de cada transferència. En els altres casos, el servidor pot mantenir-la oberta o tancar-la, i n'informa el client amb una resposta 250 o 226, respectivament.

transmissió és en tipus *stream*, l'única manera d'indicar el final de fitxer és tancant la connexió de dades.

Si no s'utilitza la comanda MODE, el mode de transmissió per defecte és S.

6. Tipus de representació (TYPE)

Amb aquesta comanda, s'especifica com es representaran les dades dels fitxers que s'hagin de transferir. El procés que les envia ha de convertir el contingut dels fitxers a la representació especificada, i el procés que les rep, les ha de convertir a la seva representació local. Els valors possibles de l'argument són els següents:

- A: s'utilitza l'especificació NVT definida en el protocol Telnet (caràcters de 8 bits codificats en ASCII i finals de línia representats amb la seqüència <CRLF>).
- E: s'utilitzen caràcters de 8 bits codificats en EBCDIC.

Nota

En els tipus de representació A i E, un segon argument opcional permet indicar com s'especifica la informació de control per al format vertical del text (separació entre paràgrafs, canvis de pàgina, etc.) i pot valer N (no hi ha informació de format), T (hi ha caràcters de control Telnet: ASCII\EBCDIC) o C (s'utilitza el format de l'estàndard ASA, RFC 740).

- I: s'envia una imatge del fitxer consistent en una seqüència arbitrària de bits, codificats en bytes (8 bits), que el receptor ha d'emmagatzemar tal com li arriba (o afegint zeros al final si el sistema local necessita que la longitud total sigui múltiple d'alguna quantitat).
- L: s'envia una seqüència de bytes lògics de *n* bits, essent *n* el valor del segon argument (obligatori en aquest cas), amb tots els bits consecutius agrupats en bytes de 8 bits. Segons el valor de *n*, és possible que el receptor necessiti aplicar alguna transformació (reversible) per a emmagatzemar les dades.

Nota

Independentment del tipus de representació, la transferència sempre es fa en bytes de 8 bits.

Si no s'utilitza la comanda TYPE, el tipus de representació per defecte és A (i sense informació de format vertical).

7. Canviar de directori (CWD)

Normalment, cada usuari té assignat un directori per defecte en el sistema de fitxers del servidor. Aquesta comanda serveix per a canviar el directori de treball pel directori que indica l'argument.

8. Canviar al directori pare (CDUP)

Aquesta comanda no rep cap argument. Es tracta d'un cas particular de l'anterior que el protocol proporciona per a facilitar l'accés al directori pare de l'actual amb independència de la sintaxi utilitzada en el sistema de fitxers per a referenciar-lo.

9. Muntar un sistema de fitxers (SMNT)

L'argument d'aquesta comanda és un nom que, en la sintaxi del sistema de fitxers del servidor, permet seleccionar un nou grup de fitxers (per exemple, un altre sistema de fitxers, una altra partició del disc, un altre disc, etc.).

10. Mostrar el directori actual (PWD)

En la resposta a aquesta comanda (codi 257), el servidor informa del directori actual. Per a facilitar la interpretació de la resposta, el nom del directori ha de ser la primera paraula que vingui després del codi numèric, i ha d'anar delimitat pel caràcter "".

11. Port de dades (PORT)

Amb aquesta comanda, el client indica quin és el seu port de dades, en cas que sigui diferent del port per defecte. Per a efectuar les transferències, el servidor haurà d'establir la connexió de dades al port especificat. Si ja hi havia una connexió de dades establerta amb un altre port quan es rep aquesta comanda, el servidor l'haurà de tancar.

L'argument ha de tenir aquesta forma: *h1,h2,h3,h4,p1,p2* (cada un dels elements és un nombre decimal entre 0 i 255). Els quatre primers nombres formen l'adreça IP, i els dos últims, el número de port (que es representa de més a menys pes).

Nota

Recordeu que el port de dades per defecte del client és el mateix des del qual ha establert la connexió de control.

ANOTACIONS

Nota

Canvi de port

Quan el mode de transmissió requereix tancar la connexió després de cada transferència, se sol utilitzar aquesta comanda per a variar el número de port i evitar així les demores que hi pot haver quan s'intenti reutilitzar un port TCP que s'acaba de tancar.

Nota

Consulteu aquesta figura en l'apartat 18.1.1.

El client pot especificar una adreça IP diferent de la seva; d'aquesta manera, es pot efectuar una transferència entre dos servidors, com il·lustra la figura del model FTP, en la qual un client controla la transferència de dades entre dos servidors.

12. Port passiu (PASV)

Aquesta comanda serveix per a indicar al servidor que, quan se li enviï una comanda de transferència, en lloc d'establir de manera activa la connexió de dades, ha de quedar preparat per a rebre-la de manera passiva. En la resposta (codi 227), el servidor torna l'adreça en la qual esperarà les peticions de connexió, i utilitzarà la mateixa notació de l'argument de la comanda PORT.

La comanda PASV s'utilitza en les transferències entre servidors. El client ha d'establir connexions de control amb els dos servidors, enviar una comanda PASV a un dels servidors i passar l'adreça tornada a l'altre amb una comanda PORT. Llavors ha d'enviar la comanda de transferència corresponent (llegir o emmagatzemar) al servidor passiu, i la comanda complementària a l'actiu.

13. Reservar espai (ALLO)

Nota

Recordeu que cada byte lògic té *n* bits, on *n* és l'argument de la comanda TYPE L o, per defecte, 8.

Alguns sistemes poden requerir que s'especifiqui la longitud d'un fitxer abans d'emmagatzemar-lo. L'argument constitueix el nombre de bytes lògics que s'han de reservar. Si és necessari, el primer argument pot anar seguit de la cadena R *n*, on n indica la longitud màxima dels registres o pàgines (per a fitxers amb tipus d'estructura R o P).

14. Obtenir (RETR)

Aquesta és l'operació de transferència de fitxers del servidor cap al client (o cap al servidor passiu). L'argument és el nom del fitxer que s'ha de transferir.

Tant en aquesta operació com en les d'emmagatzemar i afegir, si el mode de transmissió és B o C, el procés que envia les dades pot inserir un tipus especial de bloc denominat *marca de represa* (el seu contingut és un identificador de la posició actual del fitxer), que s'haurà d'utilitzar en cas d'error de la transferència. Quan troba la marca, el receptor associa un identificador propi a la posició actual i ho notifica a l'usuari. Si qui actua de receptor és el servidor, actiu o passiu, la notificació es duu a terme per mitjà d'una resposta amb el codi 110 i el text MARK $c = s$ (c i s són els identificadors de la posició del client i del servidor, respectivament).

15. Emmagatzemar (STOR)

Aquesta és l'operació de transferència de fitxers del client cap al servidor. L'argument és el nom del fitxer en el qual el servidor ha d'emmagatzemar les dades. Si aquest fitxer no existeix, es crea; si ja existeix, se'n descarta el contingut i se substitueix per les dades rebudes.

16. Emmagatzemar amb nom únic (STOU)

Aquesta operació és com l'anterior, però sense argument: el servidor triarà un nom per al fitxer (en el directori actual) que no coincideixi amb cap dels ja existents. En la resposta s'ha de notificar el nom elegit.

17. Afegir (APPE)

Aquesta operació també és com la d'emmagatzemar, excepte que, si el fitxer ja existeix, les dades rebudes s'afegiran al final del seu contingut.

18. Fer una llistar (LIST)

L'argument d'aquesta comanda és opcional. Si no hi ha argument, el servidor envia per la connexió de dades una llista dels fitxers del

ANOTACIONS

directori actual i els seus atributs, en general en un format propi del sistema. Si hi ha argument, la llista es refereix al fitxer o grup de fitxers especificat, o als fitxers continguts en el directori especificat.

19. Fer una llista de noms (NLST)

Nota

Com a precaució, el client s'hauria d'assegurar que el tipus de representació és A o E abans d'enviar la comanda LIST o NLST.

Aquesta comanda és com l'anterior, però amb el format de la llista normalitzat, cosa que significa que consta només dels noms dels fitxers, separats per finals de línia, que permet que la llista tornada sigui processada, per exemple, perquè el client pugui implementar una operació per a obtenir un grup de fitxers.

20. Avortar (ABOR)

Aquesta comanda fa que el servidor interrompi la transferència en curs (si n'hi ha) i, després, tanqui la connexió de dades. Si no, simplement tanca la connexió de dades. En el primer cas, el servidor envia un codi 426 com a resposta a l'ordre de transferència i, a continuació, un codi 226 en resposta a l'ordre ABOR. En el segon cas, només envia la resposta 226.

Nota

Per a indicar que el servidor ha d'atendre una comanda (ABOR, STAT, QUIT) mentre es dugui a terme una transferència, l'especificació RFC 959 suggereix que el client enviï per a la connexió de control el codi de funció IP del protocol Telnet, seguit d'un senyal *synch* (és a dir, una tramesa de dades urgents TCP combinada amb el codi DM), i a continuació la comanda FTP.

21. Reprendre (REST)

Nota

Recordeu que només hi pot haver marques de represa en els modes de transmissió B o C.

Si una transferència ha estat interrompuda per qualsevol causa (caiguda del servidor, del client, de la xarxa, etc.) és possible reprendre-la a partir d'un punt indicat per una marca de represa. El client ha d'enviar l'ordre REST amb un argument igual que l'identificador de posició del servidor corresponent a la marca desitjada i, a continuació, la comanda de transferència que vol reprendre.

ANOTACIONS

22. Esborrar (DELE)

L'efecte d'aquesta comanda és esborrar el fitxer especificat en l'argument.

23. Nom antic (RNFR)

Per a canviar el nom d'un fitxer, en primer lloc el client ha d'enviar aquesta comanda, amb el nom actual en l'argument i, immediatament, l'ordre RNTO.

24. Nom nou (RNTO)

L'argument d'aquesta comanda és el nom nou que s'ha de conferir al fitxer. Només es pot utilitzar immediatament després d'una comanda RNFR.

25. Crear un directori (MKD)

El servidor crea el directori indicat per l'argument. Si l'operació té èxit, el format del codi de resposta (codi 257) és idèntic al de la comanda PWD.

Nota

Segons l'especificació RFC 959, el nom retornat en la resposta 257 ha de ser apte per a poder-lo utilitzar com a argument de la comanda CWD (canviar directori). Com que en alguns sistemes (obsolets) aquest argument no podia ser un nom relatiu al directori actual, sinó que havia de ser un nom absolut (i el de l'ordre MKD, en canvi, sí que podia ser relatiu), per norma general els servidors responen a l'ordre MKD amb el nom complet del directori creat.

26. Esborrar un directori (RMD)

El servidor esborra el directori indicat per l'argument.

27. Estatus (STAT)

Si s'envia aquesta comanda durant una transferència, el servidor inclou en la resposta informació sobre l'estat actual de la transferència.

ANOTACIONS

Si no, se li pot passar un nom de fitxer com a argument perquè el servidor enviï informació similar a la de la comanda LIST, però per la connexió de control, o bé es pot utilitzar sense argument per a obtenir informació general sobre el procés FTP.

28. Sistema (SYST)

El servidor envia una resposta informant sobre quin tipus de sistema és. La primera paraula del text ha de ser un dels noms de sistema normalitzats de la llista oficial de números assignats.

Nota

La IANA (Autoritat per a l'Assignació de Números d'Internet, Internet Assigned Numbers Authority) és l'encarregada de publicar la llista de números assignats. En el moment d'elaborar aquesta unitat, l'última versió de la llista es pot trobar en l'especificació RFC 1700. ftp://ftp.isi.edu/in-notes/iana/assignments/

Exemple

Un client Unix pot utilitzar aquesta ordre per a saber si el servidor també és Unix i, si ho és, invocar automàticament la comanda TYPE I per a millorar l'eficiència de les transmissions.

29. Serveis addicionals (SITE)

Si el servidor ofereix serveis addicionals a més de les operacions estàndard del protocol, el client els pot invocar per mitjà dels arguments de la comanda SITE, la sintaxi de la qual depèn del servidor.

30. Ajuda (HELP)

Aquesta comanda, mitjançant la qual el servidor envia informació general sobre la implementació del protocol (per exemple, quines comandes suporta i quines no), es pot utilitzar sense arguments. No obstant això, el servidor també pot proporcionar informació més específica si se li passen arguments (per exemple, un nom de comanda).

31. Operació nul·la (NOOP)

El servidor simplement envia una resposta 200.

32. Reinicialitzar (REIN)

El servidor reinicialitza la sessió oblidant la identitat de l'usuari i tornant a donar als paràmetres de transmissió els valors per defecte. La

ANOTACIONS

sessió queda en el mateix estat en què es trobava just després d'establir la connexió de control.

33. Acabar la sessió (QUIT)

El servidor dóna la sessió per finalitzada i tanca la connexió de control (si hi ha una transferència en curs, després d'enviar la resposta corresponent).

La taula següent resumeix la sintaxi de les comande i els possibles codis de resposta a cada una d'elles segons l'especificació RFC 959:

Taula 9.

Sintaxi de les comandes i els codis de resposta	
Comanda	**Resposta**
Establiment de connexió	220, 120, 421
USER *usuari*	230, 331, 332, 530
PASS *contrasenya*	230, 202, 332, 530
ACCT *compte*	230, 202, 530
STRU F|R|P	200, 504
MODE S|B|C	200, 504
TYPE A|E|I|L [N|T|C|*n*]	200, 504
CWD *directori*	250, 550
CDUP	250, 550
SMNT *sist-fitxers*	250, 202, 550
PWD	257, 550
PORT *h1,h2,h3,h4,p1,p2*	200
PASV	227
ALLO *long-1* [R *long-2*]	200, 202, 504
RETR *fitxer*	226, 250, 110, 125, 150, 550
STOR *fitxer*	226, 250, 110, 125, 150, 452, 532, 552, 553
STOU	226, 250, 110, 125, 150, 452, 532, 552, 553
APPE *fitxer*	226, 250, 110, 125, 150, 452, 532, 550, 552, 553
LIST [*nom*]	226, 250, 125, 150

ANOTACIONS

Sintaxi de les comandes i els codis de resposta	
Comanda	**Resposta**
NLST [*nom*]	226, 250, 125, 150
ABOR	226
REST *marca*	350
DELE *fitxer*	250, 550
RNFR *fitxer*	350, 450, 550
RNTO *fitxer*	250, 503, 532, 553
MKD *directori*	257, 550
RMD *directori*	250, 550
STAT [*nom*]	211, 212, 213
SYST	215
SITE *cadena*...	200, 202
HELP [*cadena*]	211, 214
NOOP	200
REIN	220, 120, 421
QUIT	221

Nota

La resposta 502 només és vàlida per a les comandes no bàsiques; és a dir, les que no pertanyen al grup de la "implementació mínima".

A més dels codis de resposta específics continguts en aquesta taula, el client també pot rebre codis de resposta generals a cada comanda, com ara 500, 501, 502, 530 i 421, o bé, en el cas de les comandes de transferència, 425, 426 i 450.

18.1.4. Implementacions de l'FTP

A l'hora d'implementar el protocol FTP, hem de distingir servidor de client:

1. Servidors FTP

Segons l'especificació RFC 959, la implementació mínima d'un servidor FTP ha de suportar les comandes següents:

- USER
- RETR
- TYPE A N
- NOOP
- STOR

ANOTACIONS

- `MODE S`
- `QUIT`
- `PORT`
- `STRU F`
- `STRU R`

Nota

Molts servidors suporten l'accés públic (possiblement restringit) a una part del seu sistema de fitxers sense necessitat de seguir el procediment normal d'autenticació.

Aquest tipus de servei es coneix com a *FTP anònim*. Per norma general, s'hi accedeix utilitzant el nom d'usuari `anonymous` com a argument de la comanda `USER`. Si el servidor demana una contrasenya, sol acceptar qualsevol cadena. En aquest cas, és costum proporcionar l'adreça electrònica de l'usuari com a contrasenya.

2. Clients FTP

Actualment, hi ha moltes implementacions de clients FTP per a una gran varietat de sistemes diferents (incloent-hi els navegadors WWW); però, el client més utilitzat durant molt de temps ha estat la utilitat del sistema operatiu Unix i de les distribucions GNU/Linux denominada precisament `ftp`.

Aquest client presenta a l'usuari la majoria de les respostes del servidor, incloent-hi els codis numèrics. Les principals comandes que ofereix són els següents:

- `open`: permet especificar el nom del servidor al qual cal connectar-se, si no s'ha passat com a argument del programa, i llavors demana automàticament el nom d'usuari i, si escau, la contrasenya.

- `cd, pwd, dir`: envien les comandes `CWD, PWD` i `LIST` al servidor.

- `ascii, binary`: envien les comandes `TYPE A` i `TYPE I` al servidor.

- `get`: efectua la seqüència `PORT-RETR` per a copiar un fitxer del servidor.

Nota

Recordeu que els noms de les comandes es poden abreujar sempre que no generin ambigüitats.

ANOTACIONS

- `put`: efectua la seqüència `PORT-STOR` per a copiar un fitxer al servidor.
- `^C` (o el caràcter que generi el senyal d'interrupció): envia la comanda `ABOR`.
- `delete, mkdir, rmdir`: envien les comandes `DELE, MKD` i `RMD`.
- `rename`: envia la seqüència `RNFR-RNTO` necessària per a canviar el nom d'un fitxer.
- `mget`: envia la comanda `NLST` per a saber quins fitxers concorden amb un patró, i després una sèrie de comandes `RETR` per a poder-los copiar.
- `mput`: envia una sèrie de comandes `STOR`.
- `mdelete`: envia la comanda `NLST` i, a continuació, una sèrie de comandes `DELE`.
- `quote`: permet enviar una comanda directament al servidor (per exemple, `quote syst`).
- `?`: mostra un missatge d'ajuda.
- `quit` o `bye`: envia la comanda `QUIT` i tanca el programa.

18.1.5. Exemple de sessió FTP

A continuació, mostrem els missatges intercanviats entre un client i un servidor en una hipotètica sessió de transferència per mitjà de la utilitat `ftp`.

Després de cada comanda d'usuari, es mostren els missatges enviats, i se n'indica l'origen i la destinació (`C` per al client, `S` per al servidor), i el número de port:

```
open ftp.uoc.es
C:4695 → S:21     (establiment de connexió)
S:21 → C:4695     220 tibet FTP server (5.6) ready.<CRLF>
(usuari) anonymous
C:4695 → S:21     USER anonymous<CRLF>
S:21 → C:4695     331 Guest login ok, send ident
                  as password.<CRLF>
```

ANOTACIONS

```
(contrasenya) usuari@acme.com
C:4695 → S:21    PASS usuari@acme.com<CRLF>
S:21 → C:4695    230 Guest login ok, access restrictions
                     apply.<CRLF>
cd pub
C:4695 → S:21    CWD pub<CRLF>
S:21 → C:4695    250 CWD command successful.<CRLF>
dir
C:4695 → S:21    PORT 193,146,196,254,18,88<CRLF>
S:21 → C:4695    200 PORT command successful.<CRLF>
C:4695 → S:21    LIST<CRLF>
S:20 → C:4696    (establiment de connexió)
S:21 → C:4695    150 ASCII data connection for /bin/ls
                 (193.146.196.254,4696) (0 bytes).<CRLF>
S:20 → C:4696    total 20<CRLF>
                 drwxr-xr-x 7 0 other 512 Aug 3 07:10
                 .<CRLF>
                 ...
S:20 → C:4696    (tancament de connexió)
S:21 → C:4695    226 ASCII Transfer complete.<CRLF>
cd doc
C:4695 → S:21    CWD doc<CRLF>
S:21 → C:4695    250 CWD command successful.<CRLF>

dir
C:4695 → S:21    PORT 193,146,196,254,18,89<CRLF>
S:21 → C:4695    200 PORT command successful.<CRLF>
C:4695 → S:21    LIST<CRLF>
S:20 → C:4697    (establiment de connexió)

S:21 → C:4695    150 ASCII data connection for /bin/ls
                 (193.146.196.254,4697) (0 bytes).<CRLF>
S:20 → C:4697    total 886<CRLF>
                 drwxr-xr-x 2 0 other 512 Oct 24 1996 .<CRLF>

                 ...
S:20 → C:4697    (tancament de connexió)
S:21 → C:4695    226 ASCII Transfer complete.<CRLF>
get README
C:4695 → S:21    PORT 193,146,196,254,18,91<CRLF>
S:21 → C:4695    200 PORT command successful.<CRLF>
C:4695 → S:21    RETR README<CRLF>
S:20 → C:4699    (establiment de connexió)
S:21 → C:4695    150 ASCII data connection for README
                 (193.146.196.254,4699) (95 bytes).<CRLF>
S:20 → C:4699    (contingut del fitxer)

S:20 → C:4699    (tancament de connexió)
S:21 → C:4695    226 ASCII Transfer complete.<CRLF>

bye
C:4695 → S:21    QUIT<CRLF>
S:21 → C:4695    221 Goodbye.<CRLF>
S:21 → C:4695    (tancament de connexió)
```

ANOTACIONS

18.2. El TFTP

Hi ha situacions en què es necessita transferir fitxers d'un ordinador a un altre i l'FTP no és apropiat, principalment a causa de la seva complexitat.

Nota

Un exemple típic en què es creu necessari transferir fitxers d'un ordinador a un altre és el de les estacions de treball sense disc, que carreguen el sistema operatiu per mitjà de la xarxa. En aquest cas, hi ha d'haver un ordinador que funcioni com a "servidor d'arrencada" de l'estació, proporcionant-li el fitxer o fitxers en què es troba el codi executable del sistema operatiu.

Un petit programa resident a la memòria ROM de l'estació ha de controlar la transferència dels fitxers.

Per a aquesta operació l'FTP no és adequat, ja que requereix implementar el protocol de transport TCP, establir-hi diferents connexions simultànies, etc.

Per a satisfer les necessitats de transmissions simplificades, s'ha definit el TFTP, l'última versió del qual està especificada en l'estàndard RFC 1350.

Aquest protocol es basa en datagrames, només proporciona dues operacions (llegir i escriure fitxers) i no hi ha cap tipus d'identificació ni autenticació d'usuari.

Lectura complementària

Si voleu més informació sobre el TFTP, consulteu l'obra següent:

K. Sollins (1992, juliol). *RFC 1350 - The TFTP protocol.*

18.2.1. Conceptes bàsics del TFTP

Com que el TFTP es basa en datagrames, generalment s'utilitza amb el protocol de transport UDP. El número de port al qual el client ha d'enviar les peticions de servei és el 69.

Una **transferència TFTP** s'inicia amb un datagrama enviat pel client en qui se sol·licita l'operació desitjada: llegir o escriure un fitxer. A partir de llavors, s'estableix un diàleg en què cada part envia un datagrama de manera alternada. Cada un d'aquests datagrames serveix de confirmació de recepció de l'anterior. Això significa que en cada moment només hi pot haver un datagrama pendent de ser confirmat.

D'aquesta manera, la recuperació dels errors de transmissió és molt simple: si passa un quant temps sense que arribi la resposta a un datagrama, es reenvia, i si es rep un datagrama que ja s'havia rebut anteriorment, s'ignora. Per tant, n'hi ha prou de guardar l'últim datagrama enviat per si s'ha de retransmetre.

El client ha d'enviar el seu primer datagrama des d'un port C al port 69 del servidor. Quan el rep, el servidor tria un número de port S que hauria de canviar en cada transferència. Tots els datagrames que enviï el servidor tindran com a port d'origen el número S i com a port de destinació el número C; tots els datagrames que enviï el client, excepte el primer, tindran com a port d'origen el número C i com a port de destinació el número S. Això permet detectar una possible duplicació del primer datagrama: si el servidor el rep dues vegades o més, ha d'enviar totes dues respostes des de ports diferents; el client acceptarà la primera i enviarà missatges d'error als ports dels quals provinguin les altres.

En el transcurs de la transferència, una de les parts envia les dades del fitxer i l'altra només envia confirmacions. Les dades s'envien en blocs de longitud fixa, 512 bytes, excepte l'últim bloc, que tindrà entre 0 i 511 bytes. Cada bloc s'envia en un datagrama.

Nota

Hi haurà un últim bloc de zero bytes només quan la longitud del fitxer que s'hagi de transmetre sigui múltiple de 512.

La transferència s'acaba quan el receptor rep l'últim bloc de dades i envia la confirmació corresponent. En aquest moment, pot donar per finalitzada la comunicació. Opcionalment, pot esperar per si torna a rebre l'últim bloc, la qual cosa significaria que l'última confirmació no ha arribat a l'emissor. Si succeeix això, només cal reenviar aquesta confirmació.

Per la seva banda, l'emissor donarà per acabada la transferència quan rebi l'última confirmació o quan hagi transcorregut cert temps

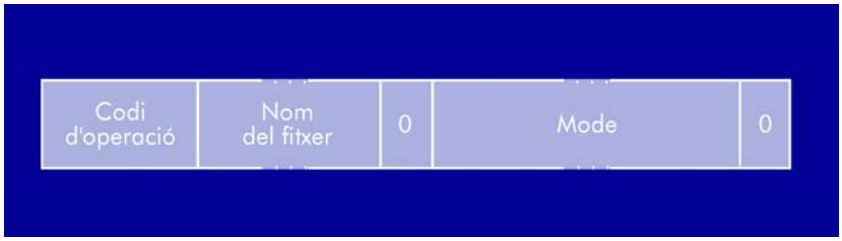
ANOTACIONS

retransmetent l'últim bloc de dades i no rebi resposta. En aquest últim cas, podria ser que la transferència s'hagués efectuat correctament i que l'únic problema estigués en la transmissió de l'última confirmació.

18.2.2. Funcionalitat del TFTP

El TFTP ofereix dues operacions bàsiques: llegir fitxers del servidor i escriure fitxers en el servidor. El datagrama inicial indica l'operació que el client vol dur a terme i té el format següent:

Figura 81.



El codi d'operació és un nombre de dos bytes, que pot ser `RRQ` o `WRQ` si el client sol·licita llegir o escriure un fitxer, respectivament. A continuació, hi ha dues cadenes de caràcters, acabades amb un byte igual a zero: la primera és el nom del fitxer i la segona és el mode de transferència (l'equivalent al tipus de representació en FTP). Aquesta segona cadena pot ser `netascii` o `octet` (en caràcters ASCII, i en qualsevol combinació de majúscules i minúscules). El primer valor indica que les dades són caràcters ASCII tal com s'usen en el protocol Telnet, i el segon indica que les dades són bytes arbitraris de 8 bits.

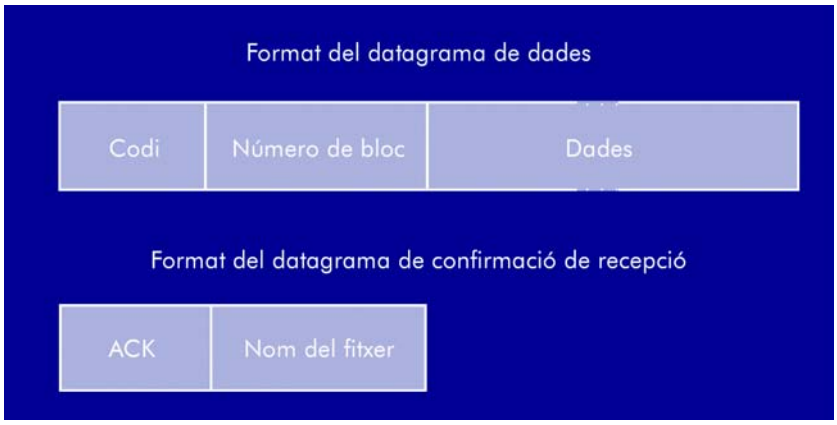
Nota

En versions anteriors del protocol, hi havia una tercera manera de transferència anomenada `mail`, només aplicable a les operacions d'escriptura, en la qual el nom del fitxer era substituït pel nom d'un usuari que havia de rebre les dades per correu.

ANOTACIONS

Els datagrames que contenen les dades i les confirmacions de recepció tenen els formats següents:

Figura 82.



El primer camp és el codi d'operació i el segon és el número de bloc que s'envia o es confirma (tots dos camps són de dos bytes). Cada bloc del fitxer té un número correlatiu, començant per 1, que serveix per a distingir les confirmacions duplicades.

Si el client envia un datagrama inicial RRQ, el servidor contesta amb un datagrama DATA amb número de bloc igual a 1 i, si el client envia un datagrama inicial WRQ, el servidor contesta amb un datagrama ACK amb número de bloc igual a 0 i, a continuació, el client envia el primer datagrama de dades. Llavors, client i servidor s'intercanvien datagrames ACK i DATA de manera alternada, amb les retransmissions necessàries si no arriba el datagrama que correspon en cada moment.

El TFTP també preveu l'acabament de la transferència si es produeix algun error. Quan es detecta l'error, s'envia un datagrama amb el format següent:

Figura 83.

Els dos primers camps són el codi d'operació i el codi d'error (cada un de dos bytes). A continuació, hi ha una cadena de caràcters, acabada en 0, que pot servir per a descriure a un usuari humà la causa de l'error.

Nota

Si arriba un datagrama del servidor amb un número de port d'origen incorrecte (probablement a causa d'un datagrama inicial duplicat), la transferència amb aquest port queda interrompuda, però la que utilitza el port correcte ha de continuar amb normalitat.

Un datagrama d'error indica que es considera acabada la transferència i no s'ha de confirmar ni, per tant, retransmetre. Ara bé, si per alguna raó es perd aquest datagrama, l'altra part interpretarà que la transferència ha acabat prematurament quan hagi transcorregut cert temps retransmetent sense rebre res.

En les taules següents es presenta una relació dels codis numèrics assignats a cada operació i a cada tipus d'error TFTP:

Taules 10 i 11.

Codi	Operació
1	RRQ
2	WRQ
3	DATA
4	ACK
5	ERROR

Codi	Error
0	Error indefinit (vegeu el missatge)
1	No s'ha trobat el fitxer
2	Accés denegat
3	Disc ple
4	Operació no vàlida
5	Número de port incorrecte
6	Ja existeix el fitxer
7	Usuari incorrecte (en mode `mail`)

18.2.3. Implementacions del TFTP

Nota

Per norma general hi ha un directori anomenat `/tftpboot` en què es desen les imatges dels sistemes operatius dels clients sense disc, i només es permet a aquest directori l'accés per mitjà del TFTP.

En molts sistemes Unix i distribucions GNU/Linux hi ha un servidor del TFTP anomenat `tftpd`. Com que no existeix cap tipus d'identificació d'usuari, els clients poden accedir en principi a qualsevol fitxer amb permisos d'accés públic. En alguns sistemes, tanmateix, es pot restringir l'accés a un directori o uns directoris determinats. No obstant això, el servei TFTP sol estar inhabilitat, i només l'ofereixen els sistemes en què es necessita per algun motiu concret (per exemple, els servidors d'arrencada de les estacions sense disc).

Així mateix, hi ha un programa client anomenat `tftp` que funciona de manera similar a la utilitat `ftp`: admet comandes com ara `get`, `put`, `ascii`, `binary` o `quit` (però no `cd`, `dir`, `delete`, `rename`, etc., perquè el protocol no les suporta).

ANOTACIONS

19. Correu electrònic Internet

El correu electrònic és l'aplicació distribuïda que permet enviar missatges electrònics per mitjà de sistemes informàtics.

En especificar aquesta aplicació, es va considerar adequat que totes les seves característiques seguissin les idees bàsiques del correu postal:

- Les **operacions** permeten bàsicament enviar missatges i rebre'ls.
- Els **elements** són equivalents als missatges, els usuaris, les oficines de correu i les bústies.
- La **funcionalitat** es basa en la filosofia de l'emmagatzematge i el reenviament: quan un missatge arriba a una oficina de correus, aquesta l'emmagatzema i no el reenvia fins al moment en què ho considera oportú. D'aquesta manera, els missatges van d'oficina de correus a oficina de correus fins que arriben al destinatari.

Nota

La filosofia de l'emmagatzematge i el reenviament permet superar problemes com les caigudes de la xarxa; en aquest cas, els missatges no es perden, ja que en cada moment hi ha una oficina responsable del missatge.

Per a dur a terme aquesta funcionalitat, es van definir els protocols següents:

- **SMTP** (*simple mail transfer protocol*), per a la **transferència de missatges**.
- **POP3** (*postoffice protocol*), per a l'**accés simple a bústies de correu**.
- **IMAP4rev1** (*Internet message access protocol*), per a l'**accés complex a bústies de correu**.

També va ser necessari definir un **format de missatge**, l'**RFC 822**, que posteriorment es va ampliar i va donar lloc al format **MIME**.

ANOTACIONS

19.1. Format dels missatges: l'RFC 822

Lectura complementària

Si voleu més informació sobre el format dels missatges de correu Internet, l'RFC 822, consulteu l'obra següent:

D. Crocker (1982, 13 d'agost). *RFC 822 - Standard for the format of ARPA Internet text messages*.

Nota

Consulteu en l'annex 3 la notació que s'utilitza per a descriure els camps dels missatges.

Abans de descriure els diferents protocols relacionats amb el correu electrònic, cal veure quin és el format o la norma dels missatges que s'hi utilitza. Aquest format rep el nom de l'estàndard en què s'especifica, **RFC 822**, i es basa en els elements típics de les cartes postals; és a dir, el sobre, que conté la informació del destinatari i del remitent de la carta, i el contingut, que és el missatge en si mateix.

L'estàndard especifica que els missatges de correu electrònic estan formats per les dues parts següents:

- La **capçalera**, que recull la informació general del missatge. Equival al sobre de la carta postal i està formada per una sèrie de camps de capçalera, cada un dels quals inclou un tipus concret d'informació.
- El **cos del missatge**, que conté el missatge en si mateix. Correspon al contingut de la carta postal. Aquesta part és opcional.

Cada camp de capçalera consta d'un **nom del camp** (que identifica el tipus d'informació) seguit pel caràcter ":" opcionalment acompanyat per un cos del camp (que inclou la informació del camp), i acaba amb un caràcter <CRLF>. El format dels camps és el següent:

NomdelCamp:[*CosdelCamp*]<CRLF>

Per norma general, cada camp de capçalera es representa en una sola línia, començant des del primer caràcter de la línia. En el cas que se'n necessiti més d'una, cal codificar aquestes línies addicionals començant per un caràcter, o més, d'espai o tabulador.

El cos del missatge és simplement una seqüència de línies amb caràcters ASCII. El cos està separat de la capçalera per una línia nul·la (és a dir, per la seqüència <CRLF><CRLF>).

Alguns sistemes de correu electrònic permeten reenviar missatges als usuaris. Per aquest motiu, s'hi inclouen uns camps de capçalera amb informació sobre el reenviament. Aquests camps són els que porten el pre-

ANOTACIONS

fix `Resent-` i tenen la mateixa semàntica que els camps corresponents que no el porten. No obstant això, sempre convindrà tenir present que, dins d'un missatge, els camps que porten prefix són els més recents.

19.1.1. Informació de la capçalera

Els camps de capçalera del missatge han de proporcionar informació general del missatge, incloent-hi la informació equivalent a la d'un sobre postal; d'aquesta manera, trobem els camps següents:

Nota

Consulteu el format de les adreces per a identificar les bústies d'usuari en l'apartat 19.2.2.

1. Originador (**`From/Resent-From`**)

La identitat i l'adreça de la bústia de la persona o les persones que originen el missatge es pot incloure en el camp `From` o `Resent-From`. En aquest camp, es pot introduir l'adreça de la bústia d'un originador o més.

```
From: 1#adreça
Resent-From: 1#adreça
```

2. Destinatari (**`To/Resent-To`**)

L'RFC 822 identifica el destinatari o destinataris principals del missatge per mitjà del camp `To` o `Resent-To`. En aquest camp, es pot introduir l'adreça d'un destinatari o més.

```
To: 1#adreça
Resent-To: 1#adreça
```

3. Destinatari de còpia (**`Cc/Resent-cc`**)

Així mateix, hi ha la possibilitat d'enviar còpies d'un missatge. En aquest cas, la identitat del receptor o receptors secundaris del missatge s'especifica amb el camp `Cc` o `Resent-cc`. En aquest últim, es pot introduir l'adreça d'un destinatari de còpia o més.

```
Cc: 1#adreça
Resent-cc: 1#adreça
```

ANOTACIONS

4. Destinatari addicional (o de còpia cega) (**Bcc/Resent-bcc**)

Quan es vol enviar una còpia del missatge a destinataris addicionals, sense que cap d'ells (els principals, els de còpia i els de còpia cega) sàpiguen que altres destinataris l'han rebut, s'utilitza el camp `Bcc` o `Resent-bcc`, que no veu cap d'ells.

```
Bcc: 1#adreça
Resent-bcc: 1#adreça
```

5. Destinatari de resposta (**Reply-To/Resent-Reply-To**)

La identificació del destinatari o destinataris a qui s'han d'enviar les respostes es pot dur a terme per mitjà del camp `Reply-To` o `Resent-Reply-To`; en aquest últim es pot introduir l'adreça d'un destinatari de la resposta o més.

```
Reply-To: 1#adreça
Resent-Reply-To: 1# adreça
```

6. Assumpte (**Subject**)

Una altra possibilitat que ofereix l'RFC 822 és enviar un text explicatiu de l'assumpte del missatge amb el camp `Subject`.

```
Subject: text
```

7. Data (**Date/Resent-Date**)

Dins d'un missatge també es pot incloure l'hora i la data en què s'envia per mitjà del camp `Date` o `Resent-Date`. El primer sistema de correu que rep el missatge genera automàticament aquest camp.

```
Date: data-hora
Resent-Date: data-hora
```

8. Sistema remitent (**Sender/Resent-Sender**)

Algunes vegades l'usuari que envia el missatge no és el mateix que l'autor. En aquests casos, la identitat de l'agent (persona, sistema o

ANOTACIONS

procés) que envia en realitat el missatge s'identifica amb el camp `Sender` o `Resent-Sender`.

```
Sender: bústia
Resent-Sender: bústia
```

9. Camí de retorn cap a l'originador (`Return-path`)

El missatge pot incloure la identificació del camí de retorn cap a l'originador del missatge amb el camp `Return-path`. Aquest camp l'afegeix el sistema de transport final que lliura el missatge al receptor.

```
Return-path: addr-ruta
```

10. Informació de sistemes intermedis (`Received`)

Hi ha la possibilitat que cada sistema de transport intermedi pel qual passa el missatge inclogui informació dels sistemes emissor (`from`) i receptor (`by`), del mecanisme físic (`via`), de l'identificador del missatge (`id`), de les especificacions originals (`for`) i de l'hora de recepció per mitjà del camp `Received`. Cada sistema intermedi afegeix una còpia d'aquest camp al missatge.

```
Received:
  [from domini]
  [by domini]
  [via atom]
  *(with atom)
[id id-msg]
  [for addr-spec]
  ; data-hora
```

11. Identificador del missatge (`Message-ID/Resent-Message-ID`)

Cada missatge ha d'incloure, en el camp `Message-ID` o `Resent-Message-ID`un, identificador únic del missatge, que es pugui contestar o referenciar posteriorment. El sistema que genera l'identificador és l'encarregat d'assegurar que l'identificador sigui únic.

```
Message-ID: id-msg
Resent-Message-ID: id-msg
```

ANOTACIONS

12. Identificador del missatge contestat (`In-Reply-To`)

Quan un missatge constitueix la resposta d'un missatge anterior, el missatge original es pot referenciar per mitjà del seu identificador dins del camp `In-Reply-To`.

```
In-Reply-To:
  *(frase | id-msg)
```

13. Identificador de missatges referenciats (`References`)

Si es vol enviar un missatge que es refereix a altres missatges, l'identificador del missatge referenciat es pot incloure dins del camp `References`.

```
References:
  *(frase | id-msg)
```

14. Paraules clau (`Keywords`)

Les paraules clau o frases referents al missatge es poden incloure, separades per comes, dins del camp `Keywords`.

```
Keywords: #frase
```

15. Comentaris (`Comments`)

Es pot incloure un text de comentari en el missatge, sense que interfereixi en el seu contingut, per mitjà del camp `Comments`.

```
Comments: text
```

16. Identificació del mecanisme de xifratge (`Encrypted`)

L'RFC 822 permet xifrar el cos del missatge. En aquest cas, és convenient enviar una o dues paraules que identifiquin el mecanisme de xifratge que s'ha aplicat utilitzant el camp `Encrypted`.

```
Encrypted: 1#2paraules
```

ANOTACIONS

Nota

L'RFC 822 només defineix la sintaxi per a especificar un mecanisme de xifratge, però no especifica el mecanisme ni la manera d'utilitzar-lo.

17. Camps d'ús personal

L'estàndard permet també que els usuaris defineixin camps per a ús personal. El nom dels camps creats per un usuari ha de començar obligatòriament per la cadena `X-`.

```
X-camp-ús-personal
```

18. Camps d'extensió

En el futur es poden estandarditzar nous camps de capçalera. Perquè no hi hagi conflictes amb els camps d'ús personal, els seus noms mai no començaran per la cadena `X-`.

```
camp-extensió
```

L'estàndard RFC 822 estableix que els únics camps de capçalera que són obligatoris en un missatge són els següents: data (`Date`), originadors (`From`), i destinatari (`To`) o destinatari de còpia cega (`Bcc`).

19.1.2. Exemple

En aquest apartat es presenta un exemple de missatge RFC 822 en què es poden apreciar els camps de capçalera més típics, i també un cos breu del missatge. Convé recordar que és un missatge en ASCII de 7 bits.

```
Date: 25 Jun 2003 0932 PDT
From: Jordi Inyigo <jinyigo@uoc.edu>
Subject: Exemple missatge RFC 822
Sender: jmmarques@uoc.edu
Reply-To: jinyigo@uoc.edu
To: Ramon Marti <rmarti@uoc.edu>,
    xperramon@uoc.edu
```

ANOTACIONS

```
Cc: Llorenc Cerda <lcerda@uoc.edu>,
    Jose Barcelo <jbarcelo@uoc.edu>,
    epeig@uoc.edu
Comment: us envio aquesta informació que us pot interessar
In-Reply-To: <1234321.567898765@uoc.edu>
Received: from uoc.edu by peru.uoc.es
      (8.8.5/8.8.5) with ESMTP id SAA14826
      for <rmarti@uoc.edu >; Fri, 20 Jun 2003
      18:35:52 +0200 (MET DST)
Received: from rectorat.uoc.edu(147.83.35.35)
      by uoc.es via smap (V2.0) id xma020193;
      Mon, 20 Jun 2003 18:38:50 +0200 for
      rmarti@uoc.edu
Message-Id: <199809211639.SAA20364@uoc.edu>

Aquest missatge té format RFC 822 i inclou
alguns dels camps de capçalera més utilitzats.
```

Activitat

Agafeu un missatge de correu electrònic que hàgiu rebut i analitzeu-ne la capçalera. Identifiqueu tots els camps i tracteu d'esbrinar qui és el responsable de crear cada un dels camps.

19.2. L'SMTP

Lectura complementària

Si voleu més informació sobre l'SMTP, consulteu l'obra següent:

J. Postel (1982, 1 d'agost). *RFC 821 - Simple Mail Transfer Protocol*.

L'SMTP és el protocol més utilitzat a Internet per a transferir missatges de correu electrònic. Proporciona la funcionalitat necessària per a aconseguir una transferència fiable i eficient de missatges de correu entre ordinadors que actuen com a oficina de correus. Seguint les idees del correu postal, l'SMTP es basa en l'emmagatzematge i el reenviament. És a dir, quan un missatge arriba a una oficina, hi queda emmagatzemat cert temps abans de ser lliurat a una altra oficina o al destinatari final. Convé assenyalar, així mateix, que cada usuari ha de disposar d'una bústia per a rebre missatges, la qual sempre ha d'estar associada a una oficina determinada.

19.2.1. Model de l'SMTP

Nota

L'agent d'usuari, definit en la majoria dels estàndards, és equivalent a l'element usuari especificat en el model client/servidor. Pot ser tant una altra aplicació, com una persona per mitjà d'una interfície d'usuari.

Des del punt de vista del model, l'SMTP ha de proporcionar els elements necessaris per a transferir missatges. Per això, es defineixen els elements següents:

- **Agent d'usuari**: s'encarrega d'introduir els missatges en el sistema de correu SMTP.

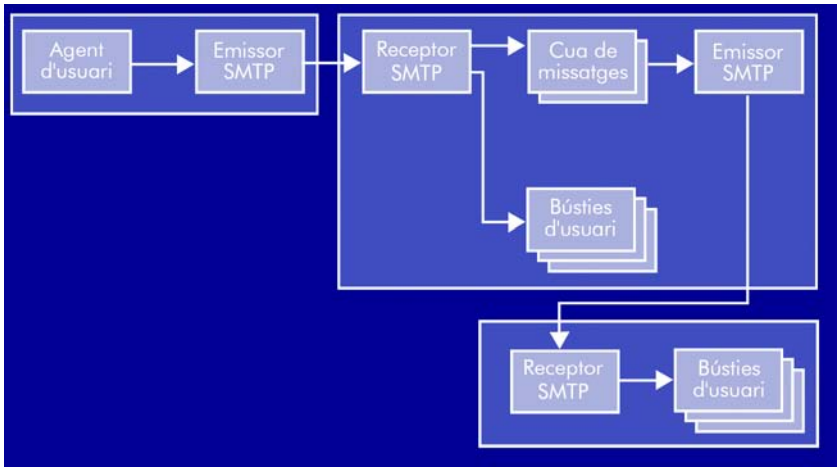
- **Emissor SMTP**: s'ocupa de fer les connexions i d'enviar missatges a receptors SMTP a partir de peticions dels usuaris. Generalment, cada emissor SMTP té associada una cua de missatges, en la qual s'emmagatzemen els missatges abans de ser reenviats (seguint la filosofia de l'emmagatzematge i el reenviament).

- **Receptor SMTP**: s'encarrega de rebre els missatges. Si el missatge va destinat a un usuari associat al mateix sistema en què es troba el receptor SMTP, aquest diposita el missatge a la bústia del destinatari. En cas contrari, el receptor SMTP diposita el missatge en la cua de missatges de l'emissor SMTP associat, que el recupera i el reenvia cap al receptor SMTP d'una altra oficina més pròxima al destinatari.

Convé destacar que els sistemes que actuen com a oficina han de disposar al mateix temps d'un receptor SMTP (per a rebre els missatges), de bústies dels usuaris i d'un emissor SMTP (per a poder reenviar els missatges) amb una cua de missatges.

La figura següent ens mostra el model d'un sistema SMTP:

Figura 84.



Nota

Noteu que aquest model, i també els altres que s'estudiaran en aquest mòdul, segueix el model client/servidor definit anteriorment. Així mateix, convé remarcar que cada estàndard proporciona un nom diferent dels elements del model client/servidor. En l'estàndard que ens ocupa, l'usuari equival a l'agent d'usuari, el client equival a l'emissor SMTP i el servidor equival al receptor.

ANOTACIONS

19.2.2. Adreces de correu

Nota

El nom de domini sol estar format per la seqüència de noms dels subdominis dels quals depèn jeràrquicament separats pel caràcter ".".

Perquè el sistema de correu sigui capaç de lliurar un missatge, es necessita algun mecanisme que permeti definir adreces per a les bústies dels usuaris. En els protocols Internet, l'adreça de bústia està formada per una cadena que identifica un usuari (persona, sistema o procés) i una cadena que identifica el sistema (domini) en què es troba la bústia.

```
adreça = usuari@domini
domini = subdomini*(.subdomini)
```

Amb aquest tipus de direccionament electrònic, es té la funcionalitat següent:

- El missatge s'envia al sistema identificat pel nom de domini que es troba en l'adreça a la dreta del signe @ (és a dir, `domini`).
- Una vegada en el sistema, el missatge es lliura a la bústia de l'usuari identificat en l'adreça a l'esquerra del signe @ (és a dir, `usuari`).
- L'SMTP proporciona també la possibilitat de definir **llistes de correu**; és a dir, llistes de destinataris identificades per una única adreça. Aquesta última és la que s'utilitza per a enviar missatges a la llista, i el sistema SMTP s'encarrega d'expandir-la i enviar el missatge a tots els usuaris que en siguin membres en aquell moment. Aquest mètode permet la modificació de la llista sense necessitat de canviar-ne l'adreça.

Nota

Cada ordinador que actua com a oficina de correus sol definir un domini de correu, que s'identifica amb un nom de domini. Aquest nom és el que es troba en la part *domini* de l'adreça electrònica dels usuaris que tenen les bústies en aquestes màquines.

19.2.3. Tramesa de correu i missatges a terminals

A més de la tramesa de missatges de correu a bústies (en l'estàndard, `mail`), que és el seu propòsit principal, l'SMTP també suporta el lliurament de missatges al terminal del destinatari (en l'estàndard, `send`). Com que la implementació d'aquests dos mètodes és molt similar, l'SMTP defineix comandes que els combinen.

19.2.4. Conceptes bàsics de l'SMTP

L'SMTP es basa en connexions TCP i el port que té assignat és el 25.

ANOTACIONS

Com hem comentat en descriure el model, l'emissor SMTP fa arribar missatges de correu al receptor. Per a aconseguir-ho, s'estableix un diàleg entre els dos amb **comandes** que envia a l'emissor i **respostes** amb què contesta al receptor.

Tant les comandes com les respostes SMTP segueixen les regles específiques del protocol Telnet. És a dir, són cadenes de caràcters codificats amb bytes segons el codi ASCII i s'utilitza la seqüència <CRLF> per a representar el final de línia.

- Les comandes SMTP estan formades per un codi constituït per quatre caràcters alfanumèrics i, segons les comandes, un espai i una sèrie de paràmetres.

- Les respostes SMTP estan formades per un codi numèric de tres dígits que, habitualment, va seguit d'un text descriptiu.

En els apartats següents es descriuran les comandes definides i les respostes possibles.

19.2.5. Funcionalitat de l'SMTP

Cal diferenciar entre funcionalitat bàsica i funcionalitat addicional.

Funcionalitat bàsica

Una vegada connectat, l'emissor SMTP s'identifica davant del receptor SMTP amb l'ordre `HELO`.

```
HELO domini
```

Quan es vol iniciar la tramesa d'un missatge de correu, s'utilitza La comanda `MAIL`, que inclou la identificació del sistema des del qual s'envia el missatge. Aquesta comanda dóna lloc al camp `FROM:` del missatge.

```
MAIL FROM: originador
```

ANOTACIONS

Amb la comanda RCPT s'identifiquen els receptors del missatge. Se n'ha d'utilitzar una per a cada receptor, i cada crida dóna lloc a un camp de capçalera TO: en el missatge.

```
RCPT TO: receptor
```

La comanda DATA indica l'inici de la tramesa del cos del missatge. Les línies següents a aquesta comanda es tracten com a contingut del missatge. El missatge s'acaba amb una línia que només inclou un punt; és a dir, amb la seqüència <CRLF>.<CRLF>.

```
DATA
```

Les dades que s'envien dins d'aquest camp són missatges RFC 822, per la qual cosa poden incloure camps de capçalera a l'inici. Quan això succeeix, entre els camps de capçalera i el cos del missatge hi ha d'haver una línia en blanc; és a dir, la seqüència <CRLF><CRLF>.

Una vegada iniciada la transacció de tramesa de missatge, i abans d'acabar-la, l'emissor SMTP sempre pot interrompre-la per mitjà de la comanda RSET.

```
RSET
```

La comanda NOOP serveix perquè el receptor SMTP enviï una resposta afirmativa per a informar que la connexió encara és oberta.

```
NOOP
```

Per a tancar el canal de transmissió, l'SMTP proporciona la comanda QUIT. Quan aquesta comanda arriba al receptor SMTP, aquest envia una resposta afirmativa i tanca el canal de transmissió.

```
QUIT
```

Funcionalitat addicional

El protocol STMP possibilita també una funcionalitat addicional amb l'objectiu d'enviar missatges a terminals.

ANOTACIONS

Per a iniciar el lliurament d'un missatge, o diversos, a terminals, l'emissor SMTP disposa de la comanda SEND.

```
SEND FROM: originador
```

La comanda SOML permet iniciar el lliurament d'un missatge, o diversos, a terminals si l'usuari es troba al terminal, o, en cas contrari, permet lliurar el correu a la bústia o a les bústies.

```
SOML FROM: originador
```

La comanda SAML permet iniciar el lliurament d'un missatge, o diversos, a terminals i, alhora, a la bústia o a les bústies.

```
SAML FROM: originador
```

L'emissor SMTP també pot demanar la confirmació que una cadena identifiqui un usuari per mitjà de la comanda VRFY.

```
VRFY cadena
```

La comanda EXPN permet sol·licitar confirmació per a una cadena que identifiqui una llista de correu i, en cas de resposta positiva, requerir la relació dels membres de la llista.

```
EXPN cadena
```

La comanda HELP permet demanar ajuda al receptor SMTP sobre les comandes que suporta. En cas d'incloure una cadena com a argument, aquesta última pot identificar una comanda, de la qual es retorna informació específica.

```
HELP [cadena]
```

La comanda TURN permet canviar el paper d'emissor SMTP i receptor SMTP. El resultat de la petició pot ser que el receptor SMTP enviï una resposta afirmativa i faci el paper d'emissor SMTP, o que enviï una resposta negativa i mantingui el seu paper.

```
TURN
```

ANOTACIONS

19.2.6. Codis de resposta

L'SMTP defineix una sèrie de codis de resposta per a les diferents comandes, tant en cas d'èxit (E), com en cas de resultat intermedi (I), de fallada (F) i d'error (X). La taula següent recull aquests codis:

Taula 12.

Codis de resposta															
	Establiment de connexió	HELO	MAIL	RCPT	DATA	RSET	SEND	SOML	SAML	VRFY	EXPN	HELP	NOOP	QUIT	TURN
211 Estatus del sistema o resposta a petició d'ajuda												E			
214 Missatge d'ajuda												E			
220 El servidor està preparat	E														
221 El servidor tanca el canal de transmissió														E	
250 Acció completada correctament		E	E	E	E	E	E	E	E	E	E	E	E		E
251 Usuari no local, el missatge es renviarà				E						E					
354 Principi d'entrada de missatge; és necessari acabar amb <CRLF>.<CRLF>					I										
421 Servei no disponible; tancament del canal de transmissió	F	X	X	X	X	X	X	X	X	X	X		X		
450 Acció no executada: bústia no accessible				F											
451 Acció avortada per error local			F	F	F		F	F	F						
452 Acció avortada per capacitat d'emmagatzematge insuficient			F	F	F		F	F	F						
500 Error de sintaxi; comanda no reconeguda		X	X	X	X	X	X	X	X	X	X	X	X	X	X
501 Error de sintaxi en els paràmetres o arguments		X	X	X	X	X	X	X	X	X	X	X			
502 Comanda no implementada							X	X	X	X	X	X			F
503 Seqüència de comandes errònia				X	X										X
504 Paràmetre de comanda no implementat		X				X				X	X	X			

Codis de resposta															
	Establiment de connexió	HELO	MAIL	RCPT	DATA	RSET	SEND	SOML	SAML	VRFY	EXPN	HELP	NOOP	QUIT	TURN
550 Acció no executada: bústia no trobada				F						F	F				
551 Usuari no local; intentar una altra adreça				F						F					
552 Acció avortada per excés de capacitat emmagatzemada			F	F	F		F	F	F						
553 Acció no executada; nom de bústia no permès				F						F					
554 Fallada en la transacció					F										

L'estàndard estableix el conjunt de comandes mínim que han de suportar tots els receptors SMTP:

- `HELO` *domini*
- `MAIL FROM:` *originador*
- `RCPT TO:` *receptor*
- `DATA`
- `RSET`
- `NOOP`
- `QUIT`

19.2.7. Extensions SMTP per a missatges de 8 bits

Es va considerar oportú afegir un mecanisme per a estendre l'SMTP. Quan un client SMTP vol utilitzar les extensions SMTP, i també les extensions per a missatges de 8 bits, ho ha de sol·licitar al receptor SMTP amb la comanda `EHLO` en lloc de la comanda `HELO`:

`EHLO` *domini*

Si el receptor SMTP suporta les extensions, retorna una resposta o més d'èxit (250) i indica les extensions de fallada (550) o d'error

ANOTACIONS

(501) que suporta. Si el receptor no les suporta, retorna una resposta d'error (500, 502, 504 o 421).

Lectura complementària

Per a saber més sobre les extensions SMTP per a missatges de 8 bits, consulteu les obres següents:

J. Klensin; N. Freed; M. Rose; E. Stefferud; D. Crocker (1995, novembre). *RFC 1869 - SMTP Service Extensions*.

J. Klensin; N. Freed; M. Rose; E. Stefferud; D. Crocker (1994, juliol). *RFC 1652 - SMTP Service Extension for 8bit- MIME transport*.

Una de les funcionalitats addicionals és la possibilitat d'enviar missatges de 8 bits. Quan el servidor accepta missatges de 8 bits, la resposta d'èxit (250) inclou la cadena `8BITMIME`. Quan es vulgui enviar el missatge, s'ha d'indicar que és de 8 bits. Això s'aplica a les ordres `MAIL`, `SEND`, `SAML` o `SOML` de la manera següent:

```
MAIL FROM: originador BODY = valor-cos
SEND FROM: originador BODY = valor-cos
SAML FROM: originador BODY = valor-cos
SOML FROM: originador BODY = valor-cos
valor-cos = 7BIT | 8BITMIME
```

19.2.8. Exemple

En aquest apartat es presenta un exemple en què es pot veure la seqüència de comandes que envia un emissor SMTP (en negreta) i les respostes del receptor SMTP en una transacció de tramesa de correu:

```
220 peru.uoc.es SMTP/smap Ready.
HELO campus.uoc.es
250 (campus.uoc.es) pleased to meet you.
HELP
214-Commands
214-HELO    MAIL    RCPT    DATA    RSET
214 NOOP    QUIT    HELP    VRFY    EXPN
NOOP
220 OK
EXPN xc@campus.uoc.es
250-Jordi Inyigo <jinyigo@uoc.edu>
250-Jose Barcelo<jbarcelo@uoc.edu>
250-Llorenc Cerda <lcerda@uoc.edu>
250-Ramon Marti <rmarti@uoc.edu>
250-Enric Peig <epeig@uoc.edu>
250 Xavier Perramon <xperramon@uoc.edu>
MAIL FROM: jinyigo@uoc.edu
250 jinyigo@uoc.edu... Sender OK
```

```
RCPT TO: rmarti@uoc.edu
250 rmarti@uoc.edu OK
RCPT TO: rmarti@uoc.edu
501 Syntax error
RCPT TO: xperramon@uoc.edu
250 xperramon@uoc.edu OK

DATA
354 Enter mail, end with "." on a line by itself
Subject: Master de programari lliure
Date: 20 Jun 2003
Això és un missatge de correu d'exemple.
.
250 Mail accepted
QUIT
250 Closing connection
```

Nota

`telnet` com a client genèric dels protocols

El programa `telnet` està pensat per a fer de client de servidors del protocol Telnet. Tanmateix, si es considera que el que fa és enviar al servidor cadenes de caràcters tal com s'introdueixen pel teclat i tornar per pantalla el que rep del servidor, també es pot utilitzar com a client SMTP, ja que, com hem comentat en descriure'l, els missatges que s'intercanvien en aquest protocol són cadenes de caràcters ASCII. Aquest mateix raonament es pot aplicar a la resta de protocols que veurem: POP3, IMAP, NNTP i HTTP.

D'aquesta manera, si s'utilitza el programa `telnet` com a client SMTP, es poden enviar comandes SMTP al servidor escrivint-los pel teclat i veient les seves respostes per la pantalla. Per a això, el programa `telnet` admet com a segon paràmetre el número de port on s'ha de connectar.

Per exemple, el diàleg anterior es podria haver realitzat fent:

```
telnet peru.uoc.es 25
```

ANOTACIONS

19.3. Accés simple a les bústies de correu: el POP3

En sistemes petits no és pràctic, ni usual, suportar l'SMTP, ja que implica tenir el sistema connectat i disposat a rebre missatges en qualsevol moment. Per aquest motiu, es va creure necessari definir un protocol que permetés la recuperació de missatges de bústies de correu remots i es va definir el **POP3**.

Nota

El POP i el POP2, dos protocols definits en les RFC 918 i RFC 937 respectivament, parteixen de la mateixa filosofia que el POP3; tanmateix, disposaven d'un conjunt de comandes més reduït.

No obstant això, en aquest protocol cal disposar de sistemes en què es trobin les bústies –un servidor POP3–, i han d'estar connectats en tot moment, tant per a rebre els missatges, com per a rebre les peticions d'accés a les bústies. Pel que respecta als clients POP3, només cal que es connectin quan vulguin accedir al seu correu.

Lectura complementària

Si voleu més informació sobre POP3, consulteu l'obra següent:

J. Myers; M. Rose (1996, maig). *RFC 1939 - Post Office Protocol - Version 3.*

El POP3 no especifica cap mètode per a la tramesa de correu; altres protocols de transferència de correu, com l'SMTP, proporcionen aquesta funcionalitat.

19.3.1. Model del POP3

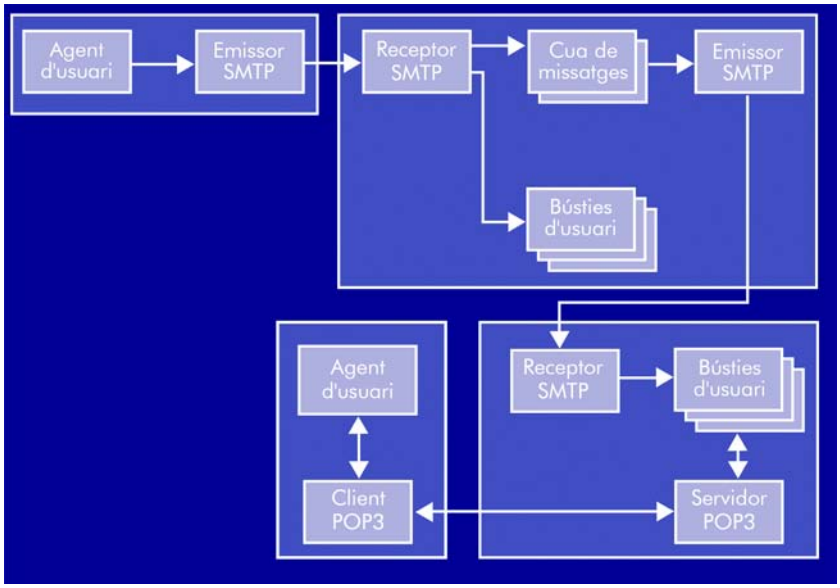
El model funcional del POP3 es basa en els elements següents:

- **Agent d'usuari**: utilitza el client POP3 per a accedir al seu correu.
- **Client POP3**: es comunica amb el servidor POP3 per mitjà del protocol POP3 per a accedir a la seva bústia de correu.
- **Servidor POP3**: rep peticions dels clients POP3 i els les serveix accedint a les bústies corresponents.

ANOTACIONS

En la figura següent es presenten els elements del model funcional del POP3 integrats en un sistema en què s'utilitza l'SMTP per a enviar el correu, i el POP3 per a accedir a les bústies:

Figura 85.



19.3.2. Conceptes bàsics del POP3

El POP3 es basa en comunicacions TCP sobre el port 110.

El mecanisme normal d'utilització d'aquest protocol és el següent: quan el client POP3 necessita accedir a la bústia, es connecta amb el servidor POP3, recupera la informació que li interessa i tanca la connexió.

Cada vegada que sigui necessari tornar a accedir a la bústia, s'estableix una nova connexió.

- Les **comandes POP3** constitueixen cadenes de caràcters ASCII imprimibles acabats amb <CRLF>. Totes les comandes inclouen un codi alfanumèric de quatre caràcters que identifica la comanda, seguit de zero o més paràmetres.
- Les **respostes POP3** també són cadenes de caràcters ASCII, i es representen amb un indicador d'estat positiu (+OK) o negatiu

ANOTACIONS

(–ERR) i, possiblement, informació addicional, de la manera següent:

```
+OK la comanda s'ha executat amb èxit
–ERR la comanda no s'ha executat amb èxit
```

Estats

La norma defineix tres estats pels quals ha de passar tota sessió POP3:

- Una vegada s'ha obert la connexió, la sessió entra en l'**estat d'autorització**, quan el client s'ha d'identificar davant del servidor POP3.
- Una vegada autoritzat, la sessió passa a l'**estat de transacció**. En el qual, el client demana accions al servidor POP3 amb les comandes necessàries.
- Quan el client crida la comanda `QUIT`, la sessió entra en l'**estat d'actualització**. El servidor allibera els recursos, s'acomiada i tanca la connexió TCP.

19.3.3. Funcionalitat del POP3

Des del punt de vista de funcionalitat, i reprenent els plantejaments del correu postal, el POP3 proporciona les comandes necessàries per a accedir a bústies de correu. A continuació, descriurem les comandes corresponents a cada un dels estats pels quals ha de passar una sessió POP3:

1. **Estat d'autorització**. En aquest estat el client s'identifica davant del servidor POP3. Per a dur a terme el procés d'identificació, es disposa de les comandes següents:

a) **Identificació d'usuari (`USER`)**

El primer que ha de fer un client POP3 és identificar-se davant del servidor POP3. Un dels mètodes és fer-ho mitjançant la comanda USER, dins de la qual s'envia el nom que identifica l'usuari.

USER *nom*

b) **Tramesa de la contrasenya (PASS)**

Una vegada identificat l'usuari mitjançant la comanda USER, s'ha de dur a terme l'autenticació per mitjà de la tramesa d'una contrasenya amb la comanda PASS. El servidor utilitza la cadena enviada en aquesta comanda juntament amb el nom d'usuari per a donar accés a l'usuari a la bústia o denegar-l'hi.

```
PASS contrasenya
```

c) **Identificació i autenticació de l'usuari amb seguretat (APOP)**

El mètode d'identificació i autenticació mitjançant les comandes USER i PASS té el problema que tant el nom com la contrasenya viatgen per la xarxa sense cap mecanisme de seguretat. Un mètode alternatiu d'identificació i autenticació de l'usuari consisteix a utilitzar la comanda APOP, que incorpora mecanismes de seguretat en la tramesa de la contrasenya.

```
APOP nom resum
```

Nota

La comanda APOP actua de la manera següent:

En connectar-se, el servidor envia una cadena al client. El client concatena la cadena rebuda a la contrasenya i en crea un resum per mitjà d'un algorisme de creació de resums (*hash* criptogràfic). Aquest resum s'envia juntament amb el nom identificatiu de l'usuari com a arguments de la comanda APOP.

El servidor, per a comprovar l'usuari, genera el mateix resum i el compara amb el que ha rebut.

2. **Estat de transacció**. En aquest estat, el client sol·licita accions al servidor POP3. Les accions que pot demanar són les següents:

a) **Estat (STAT)**

Una vegada autenticat l'usuari, el client POP3 pot requerir informació sobre l'estat de la bústia de l'usuari per mitjà de la comanda

ANOTACIONS

STAT que no té arguments i torna el nombre de missatges i els bytes que ocupa la bústia de l'usuari.

```
STAT
```

b) Llista (LIST)

Quan ja se sap el nombre de missatges que hi ha a la bústia, el pas següent és la petició d'informació sobre un dels missatges o sobre tots. El POP3 proporciona la comanda LIST per a aquesta tasca. Aquesta comanda torna, per a cada missatge, un número de missatge i els bytes que ocupa.

```
LIST [missatge]
```

c) Recuperació de missatges (RETR)

Una vegada es coneixen els missatges que hi ha a la bústia, cal recuperar els que es vol llegir. Això ho pot fer el client POP3, per a cada missatge, amb la comanda RETR, que inclou com a argument l'identificador del missatge que es vol recuperar.

```
RETR missatge
```

d) Esborrament de missatges (DELE)

Després de llegir un missatge, pot interessar esborrar-lo. La comanda DELE indica al servidor POP3 que marqui com a missatge a esborrar l'identificat com a tal en l'argument; tanmateix, el missatge no s'esborrarà fins que no s'entri en l'estat d'actualització.

```
DELE missatge
```

e) Operació nul·la (NOOP)

La comanda NOOP serveix per a saber si la connexió amb el servidor encara és oberta. Amb aquesta comanda, el servidor POP3 no fa res, excepte tornar una resposta afirmativa.

```
NOOP
```

ANOTACIONS

f) Desmarcatge de missatges per a esborrar (`RSET`)

Una vegada s'ha fet una crida a la comanda DELE, i abans d'entrar en l'estat d'actualització, l'usuari es pot fer enrere i demanar al servidor POP3, mitjançant la comanda RSET, que desmarqui tots els missatges marcats per a esborrar.

```
RSET
```

g) Recuperació de la part superior d'un missatge (`TOP`)

A vegades, pot interessar recuperar només la part superior d'un missatge per a decidir si val la pena recuperar-lo tot o no. La comanda TOP permet recuperar la capçalera i *n* línies del missatge identificat en l'argument.

```
TOP missatge n
```

h) Llista d'identificadors únics (`UIDL`)

Tots els missatges de la bústia tenen un identificador únic (*uniqueid*) permanent (a diferència del número de missatge, que és únic dins de la bústia, però que pot variar). La comanda UIDL permet obtenir el número de missatge i l'identificador únic d'un dels missatges de la bústia o de tots.

```
UIDL [missatge]
```

i) Pas a l'estat d'actualització (`QUIT`)

Una vegada finalitzades les transaccions, el client POP3 ha de cridar La comanda QUIT per a passar a l'estat d'actualització.

```
QUIT
```

3. **Estat d'actualització**. En aquest estat, el servidor POP3, en primer lloc, esborra tots els missatges marcats per a esborrar i, pos-

ANOTACIONS

teriorment, allibera tots els recursos i tanca la connexió TCP. L'estat d'actualització no disposa de cap comanda associada.

L'estàndard estableix que els servidors POP3 han de suportar com a mínim les comandes següents:

USER *nom*
PASS *cadena*
STAT
LIST [*missatge*]
RETR *missatge*
DELE *missatge*
NOOP
RSET
QUIT

19.3.4. Exemple

A continuació, es presenta un exemple de les comandes d'accés d'un client POP3 (en negreta) a un servidor POP3 per a recuperar el missatge enviat en l'exemple anterior:

```
+OK QPOP (version 2.52) at pop.uoc.es starting.
USER rmarti
+OK Password required for rmarti.
PASS prueba
-ERR Password supplied for "rmarti" is incorrect.
+OK Pop server at pop.uoc.es signing off.
PASS password
+OK rmarti has 6 message(s) (190885 bytes).
STAT
+OK 6 190885
LIST
+OK 6 messages (190885 bytes)
1 3140
2 3326
3 1911
4 180846
5 861
```

ANOTACIONS

```
6 801
.
RETR 6
+OK 801
Received: from campus.uoc.es by peru.uoc.es
   (8.8.5/8.8.5) with ESMTP id SAA14826
   for <rmarti@uoc.edu>; Fri, 27 Jun 2003
   18:35:52 +0200 (MET DST)
From: Jordi Inyigo <jinyigo@uoc.edu>
Message-Id:
   <199809211639.SAA20364@peru.uoc.es>
To: rmarti@uoc.edu, xperramon@uoc.edu
Subject: Master de programari lliure
Date: 27 Jun 2003
Content-Type: text
Status: RO

Això és un missatge de correu d'exemple.
.
QUIT
+OK Pop server at dns signing off
```

19.4. Accés complex a les bústies de correu: l'IMAP4rev1

El protocol d'accés a missatges Internet, versió 4 rev 1, IMAP4rev1, permet al client accedir als missatges de correu electrònic d'un servidor i manipular-los.

L'IMAP4rev1 (a partir d'ara l'anomenarem *IMAP4*) permet a l'usuari disposar de diferents bústies estructurades de manera jeràrquica i, alhora, poder-les manipular de manera remota, tal com fa amb les bústies locals.

L'IMAP4 també proporciona als clients la capacitat de resincronització amb el servidor.

L'IMAP4 no especifica cap mètode per a la tramesa de correu; altres protocols de transferència de correu, com l'SMTP, proporcionen aquesta funcionalitat.

Nota

L'IMAP4rev1 va sorgir de l'evolució de les especificacions IMAP2 [RFC 1176], IMAP3 [RFC 1203] i IMAP4 [RFC 1730].

19.4.1. Model de l'IMAP4

El model funcional de l'IMAP4 es basa en els elements que presentem a continuació:

- **Agent d'usuari**: utilitza el client IMAP4 per a llegir el correu de la seva bústia.
- **Client IMAP4**: es comunica amb el servidor IMAP4 per mitjà de l'IMAP4 per a accedir a la seva bústia de correu.
- **Servidor IMAP4**: rep peticions dels clients IMAP4 i els les serveix accedint a les bústies corresponents.

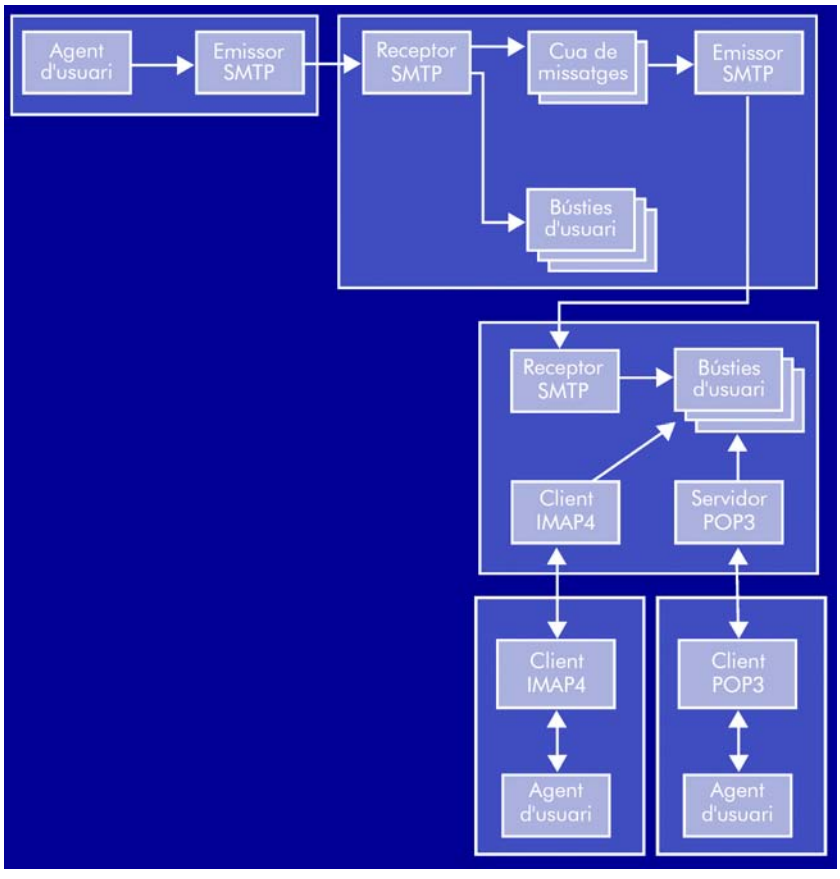
Lectura complementària

Si voleu més informació sobre l'IMAP4rev1, consulteu l'obra següent:

M. **Crispin** (1996, desembre). *RFC 2060 - Internet Message Access Protocol - Version 4 rev 1.*

La figura següent presenta els elements del model funcional de l'IMAP4 integrats en un sistema en què s'utilitza SMTP per a enviar el correu i IMAP4 per a accedir a les bústies:

Figura 86.



19.4.2. Conceptes bàsics de l'IMAP4

L'IMAP4 es pot utilitzar amb qualsevol protocol de transport fiable. Per norma general, s'utilitza el TCP i, en aquest cas, s'utilitza el port 143. Totes les interaccions entre client i servidor es duen a terme en forma de línies ASCII acabades amb un caràcter <CRLF>.

Cada comanda del client comença amb un identificador (en general, una cadena alfanumèrica curta) anomenat *etiqueta* o ***tag***. El client ha de generar una etiqueta diferent per a cada comanda.

El servidor pot enviar dades tant en resposta a una comanda del client, com de manera unilateral, i el client ha d'estar a punt per a rebre-les en tot moment. Les dades transmeses pel servidor cap al client i les respostes d'estatus que no impliquen l'acabament de la comanda comencen amb el *token*. La resposta final de culminació de la comanda comença amb la mateixa etiqueta que la comanda del client que ha donat lloc a la resposta.

A més de les respostes específiques de cada comanda, gairebé totes les comandes disposen, com a mínim, dels resultats d'estatus següents:

```
OK [Paràmetres]: l'ordre s'ha executat.
NO [Paràmetres]: l'ordre no s'ha executat.
BAD [Paràmetres]: l'ordre és desconeguda o els arguments són
invàlids.
```

Exemple

```
a006 logout
* BYE IMAP4rev1 server terminating connection
a006 OK LOGOUT completed
```

La primera línia és la crida a la comanda precedida de l'etiqueta (a006). En aquest cas, l'ordre LOGOUT. Després, es pot veure la resposta, que està formada per dues línies. L'última línia de resposta comença amb la mateixa etiqueta que la cridada, mentre que les línies anteriors ho fan amb el *token*.

Nota

Consulteu l'ordre LOGOUT en l'apartat 19.4.3.

El client pot enviar una comanda després d'una altra sense esperar el resultat de la primera. De manera similar, un servidor pot començar a processar una comanda abans d'acabar de processar la que s'executa en aquell moment.

ANOTACIONS

A més del text, cada missatge té diferents atributs associats que es troben emmagatzemats dins de la bústia i que són els següents:

1. **Identificador únic (UID, *unique identifier*)**: a cada missatge se li associa un valor de 32 bits que, quan s'utilitza conjuntament amb els valors de validesa d'identificador únic (*unique identifier validity value*), forma un valor de 64 bits que identifica de manera única un missatge dins d'una bústia. Els UID s'assignen de manera ascendent dins de la bústia, però no necessàriament de manera contigua.

2. **Número de seqüència del missatge (*message sequence number*)**: aquest atribut proporciona la posició relativa del missatge dins de la bústia, des de l'1 fins al nombre total de missatges.

Aquesta posició s'ha d'ordenar seguint els UID ascendents. Els números de seqüència del missatge es poden reassignar durant la sessió (per exemple, quan s'elimina un missatge de la bústia).

3. **Indicadors**: cada missatge té associada una llista de zero indicadors, o més, que informen de l'estat:

- `\Seen`: missatge llegit
- `\Answered`: missatge contestat
- `\Flagged`: missatge marcat per atenció urgent/especial
- `\Deleted`: missatge marcat per a ser esborrat per un *Expunge* posterior
- `\Draft`: missatge no editat del tot
- `\Recent`: missatge acabat d'arribar en aquesta sessió

4. **Data interna (*internal date*)**: data i hora que cada missatge porta associades de manera interna dins del servidor, que reflecteixen quan ha arribat el missatge al servidor. No és la data del missatge RFC 822.

5. **Longitud [RFC 822] ([RFC 822] *size*)**: és el nombre de bytes del missatge expressat en el format RFC 822.

6. **Estructura del sobre (*envelope structure*)**: representació analitzada de la informació del sobre RFC 822.

7. **Estructura del cos (*body structure*)**: representació analitzada de la informació de l'estructura del cos MIME.

ANOTACIONS

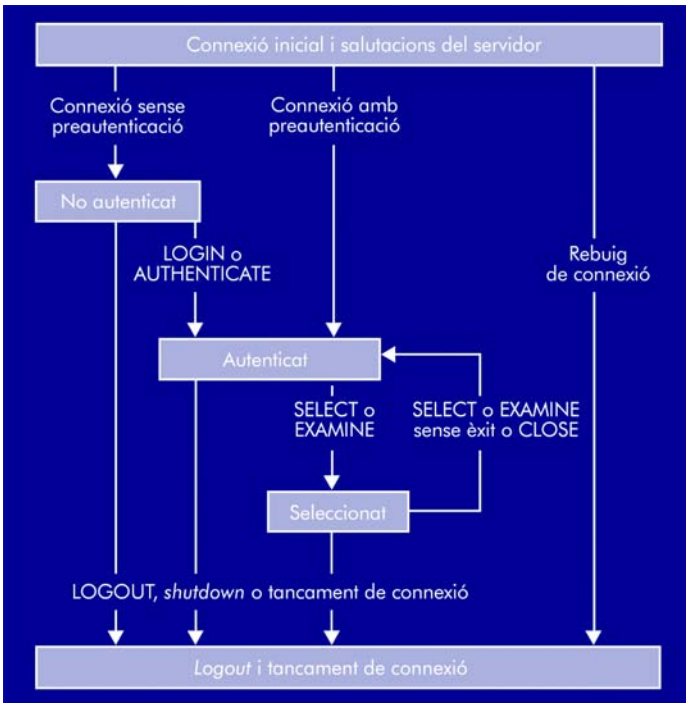
8. **Textos de missatge**: a més de permetre la recuperació dels missatges RFC 822 sencers, amb l'IMAP4 també es poden efectuar recuperacions parcials. En concret, permet recuperar la capçalera i/o el cos del missatge RFC 822, una part del cos MIME o una capçalera MIME.

L'IMAP4 especifica quatre estats:

- **Estat no autenticat**: el client ha de proporcionar les seves credencials. S'arriba a aquest estat quan es comença una connexió, llevat que ja s'hagi preautenticat.
- **Estat autenticat**: el client ha de seleccionar una bústia a la qual accedirà abans de tenir permís per a efectuar comandes sobre els missatges.
- **Estat seleccionat**: s'entra en aquest estat quan s'ha seleccionat amb èxit una bústia.
- **Estat de *logout***: la connexió s'acaba i el servidor la tancarà. Es pot entrar en aquest estat com a resultat d'una petició del client o de manera unilateral per part del servidor.

La figura següent mostra el diagrama de flux entre els diferents estats definits per l'IMAP4:

Figura 87.



ANOTACIONS

Cada usuari té el seu correu al servidor, dipositat en un conjunt de bústies estructurades jeràrquicament que es poden manipular remotament per mitjà de l'IMAP4.

Nota

Encara que l'IMAP4 utilitzi la paraula *bústia* per a identificar tots els llocs on es poden desar missatges, en realitat el que s'entén per *bústia on es reben els missatges* és la safata d'entrada (bústia *inbox*), mentre que les altres bústies són més aviat carpetes en què es classifiquen els missatges rebuts.

Cada bústia s'identifica per un nom relatiu, una cadena de caràcters que la diferencia de les bústies germanes. Així mateix, cada bústia disposa d'un nom que l'identifica dins de l'estructura formada per la seqüència dels noms relatius de les bústies que defineixen els nivells de jerarquia superiors, d'esquerra a dreta, separats amb un únic caràcter (per norma general, el caràcter "/"). S'ha d'utilitzar el mateix separador en tots els nivells de la jerarquia dins d'un nom.

19.4.3. Funcionalitat de l'IMAP4

La funcionalitat proporcionada per l'IMAP4, com la del POP3, imita la que es requereix per a accedir a les bústies de correu postal. Com a millora respecte del POP3, l'IMAP4 proporciona una estructura jeràrquica de la bústia en forma de carpetes, i també facilitats de subscripció, la qual cosa dóna lloc a comandes noves que permeten gestionar tots aquests elements.

Les funcions que s'utilitzaran segons l'estat en què es trobi la comunicació seran les següents:

1. **Qualsevol estat (comandes / estats universals)**. Independentment de l'estat de la connexió, es poden utilitzar les comandes següents:

a) **Petició de capacitats (`CAPABILITY`)**

La comanda `CAPABILITY`, que no té cap argument, serveix per a sol·licitar la llista de capacitats que suporta el servidor

```
CAPABILITY
```

ANOTACIONS

b) Operació nul·la (**NOOP**)

La comanda NOOP permet al client IMAP4 esbrinar si la connexió amb el servidor encara és oberta.

```
NOOP
```

c) Acabament de connexió (**LOGOUT**)

La comanda LOGOUT permet al client notificar al servidor que vol acabar la connexió.

```
LOGOUT
```

2. **Estat no autenticat**. En aquest estat, un client proporciona les seves credencials; per a això, s'utilitzen les comandes següents:

a) Indicador d'autenticació (**AUTHENTICATE**)

La comanda AUTHENTICATE serveix per a indicar al servidor IMAP4 un mecanisme d'autenticació; és a dir, quin dels diferents mecanismes d'autenticació possibles utilitza el client.

```
AUTHENTICATE tipus_autenticació
```

b) Identificació d'usuari (**LOGIN**)

Com tot protocol, l'IMAP4 proporciona una comanda per a permetre que el client s'identifiqui davant del servidor per mitjà de la tramesa d'un identificador d'usuari i una contrasenya. Aquesta comanda és LOGIN.

```
LOGIN id_usuari contrasenya
```

3. **Estat autenticat**. En aquest estat l'IMAP4 proporciona algunes funcionalitats noves:

a) Selecció d'una bústia (**SELECT**)

Una vegada autenticat, el client IMAP4 pot seleccionar una bústia per a accedir als seus missatges. La comanda SELECT proporciona aquesta funcionalitat.

ANOTACIONS

```
SELECT bústia
```

Aquesta comanda, a més dels resultats d'estatus, retorna les respostes obligatòries sense etiqueta següents:

```
FLAGS:  informa dels identificadors que es poden aplicar a
        la bústia de la comanda.
EXISTS: nombre de missatges existents a la bústia.
RECENT: nombre de missatges amb l'indicador \Recent.
```

També pot retornar les respostes OK sense etiqueta:

```
NSEEN: número del primer missatge sense l'indicador \Seen.
PERMANENTFLAGS: llista d'indicadors que es poden modificar
permanentment.
```

b) Examen d'una bústia (**EXAMINE**)

La comanda EXAMINE permet un accés a la bústia similar a la de la comanda SELECT, però de manera que la bústia sigui només de lectura. Aquesta comanda, a més dels resultats d'estatus, pot retornar les mateixes respostes que la comanda SELECT.

```
EXAMINE bústia
```

c) Creació d'una bústia (**CREATE**)

A l'usuari li pot interessar crear una bústia amb un nom determinat. L'IMAP4 proporciona la comanda CREATE per a això.

```
CREATE bústia
```

d) Esborrament d'una bústia (**DELETE**)

L'IMAP4 també facilita l'eliminació permanent d'una bústia amb un nom determinat per mitjà de la comanda DELETE.

```
DELETE bústia
```

ANOTACIONS

e) Reanomenament d'una bústia (**RENAME**)

A vegades, l'usuari vol canviar el nom d'una bústia per un de nou. El client IMAP4 pot sol·licitar aquesta acció al servidor per mitjà de la comanda RENAME.

```
RENAME bústia novabústia
```

f) Subscripció d'una bústia (**SUBSCRIBE**)

En l'IMAP4, no cal que totes les bústies estiguin sempre actives. Amb la comanda SUBSCRIBE, el nom de la bústia passa a formar part de la llista de bústies actives o subscrites.

```
SUBSCRIBE bústia
```

g) Eliminació de la subscripció d'una bústia (**UNSUBSCRIBE**)

La comanda UNSUBSCRIBE permet desactivar una bústia eliminant-ne el nom de la llista de bústies actives o subscrites.

```
UNSUBSCRIBE bústia
```

h) Llista de bústies (**LIST**)

La comanda LIST permet obtenir els noms de bústies que compleixen els criteris de recerca desitjats dins d'una bústia de referència.

```
LIST bústia 1*criteri_cerca
```

Aquesta comanda, a més dels resultats d'estatus, pot retornar la resposta sense etiqueta següent:

```
LIST:  nom de la bústia que compleix els criteris de recerca
       desitjats.
```

Hi pot haver més d'una resposta LIST.

ANOTACIONS

i) **Llista de bústies subscrites (LSUB)**

La comanda LSUB permet obtenir els noms de les bústies actives o subscrites que compleixen els criteris de recerca desitjats dins d'una bústia de referència.

```
LSUB bústia 1*criteri_cerca
```

Aquesta comanda, a més dels resultats d'estatus, pot retornar la resposta sense etiqueta següent:

```
LSUB:  nom de la bústia que compleix els criteris de recerca
       desitjats.
```

Hi pot haver més d'una resposta LSUB.

j) **Estat de la bústia (STATUS)**

La comanda STATUS permet conèixer l'estat d'una bústia.

Els atributs d'estat definits són els següents:

- MESSAGE: nombre de missatges a la bústia.
- RECENT: nombre de missatges amb l'indicador \Recent.
- UIDNEXT: UID que s'assignarà al missatge següent que arribi a la bústia.
- UIDVALIDITY: valor de l'UID de la bústia.
- UNSEEN: nombre de missatges sense l'indicador \Seen.

```
STATUS bústia (1#atribut_estat)
```

Aquesta comanda, a més dels resultats d'estatus, pot retornar la resposta sense etiqueta següent:

```
STATUS: nom de bústia que compleix l'estatus desitjat i la
        informació d'estatus requerida.
```

ANOTACIONS

k) Afegit d'un missatge a la bústia (**APPEND**)

La comanda APPEND permet al client de l'IMAP4 afegir un text literal com a nou missatge al final d'una bústia seleccionada, amb la data, l'hora i els indicadors desitjats.

```
APPEND bústia[llista_flags]
  [data_hora] literal
```

4. **Estat seleccionat**. En l'estat seleccionat, l'IMAP4 proporciona noves funcionalitats, a més de les comandes universals i les de l'estat autenticat:

a) Control de la bústia (**CHECK**)

La comanda CHECK permet al client IMAP4 demanar al servidor un punt de control de la bústia seleccionada en un moment determinat.

```
CHECK
```

b) Tancament de la bústia (**CLOSE**)

La comanda CLOSE permet tancar una bústia seleccionada i eliminar permanentment tots els seus missatges que tenen l'indicador `\Deleted`.

```
CLOSE
```

c) Eliminació de missatges (**EXPUNGE**)

L'ordre EXPUNGE permet eliminar de manera permanent tots els missatges que tenen l'indicador `\Deleted` de la bústia seleccionada i sense necessitat de tancar-lo.

```
EXPUNGE
```

Aquesta comanda, a més dels resultats d'estatus, pot retornar la resposta sense etiqueta següent:

```
EXPUNGE: número de seqüència de missatge especificat que
         ha estat eliminat permanentment.
```

ANOTACIONS

d) Recerca de missatge (**SEARCH**)

La comanda SEARCH permet al client IMAP4 buscar dins de la bústia els missatges que contenen un conjunt de caràcters especificat i que compleixen uns criteris de recerca desitjats.

```
SEARCH [CHARSET
  conjunt_caràcters]
  1#criteri_cerca
```

Aquesta comanda, a més dels resultats d'estatus, pot retornar la resposta sense etiqueta següent:

```
SEARCH: un número de seqüència o més dels missatges que com-
        pleixen el criteri de recerca desitjat.
```

e) Recuperació de missatges (**FETCH**)

La comanda FETCH permet recuperar un conjunt de missatges (totalment o parcialment), especificant uns atributs de recuperació.

```
FETCH conjunt_missatges ALL | FULL | FAST |
atrib_recup | (1#atrib_recup)
```

Aquesta comanda, a més dels resultats d'estatus, pot retornar la resposta sense etiqueta següent:

```
FETCH: informació del missatge que presenta els atributs de
       recuperació especificats.
```

f) Modificació de magatzem (**STORE**)

La comanda STORE permet modificar els indicadors del magatzem que proporcionen els atributs a un conjunt de missatges de la bústia.

```
STORE conjunt_missatges
indicadors_atrib_magatzem
```

Aquesta comanda, a més dels resultats d'estatus, pot retornar la resposta sense etiqueta següent:

```
FETCH: informació dels missatges.
```

ANOTACIONS

g) Còpia de missatge(s) (`COPY`)

La comanda `COPY` permet copiar un conjunt de missatges al final d'una bústia especificada.

```
COPY conjunt_missatges bústia
```

h) Retorn d'identificador únic (`UID`)

La comanda `UID`, seguida dels paràmetres `COPY`, `FETCH`, `STORE` o `SEARCH`, en lloc de retornar el número o números de seqüència de missatge, retorna els seus identificadors únics.

```
UID (COPY... | FETCH... |
  SEARCH... | STORE ...)
```

Aquesta ordre, a més dels resultats d'estatus, pot retornar les respostes sense etiqueta següents:

```
FETCH:  informació del missatge que presenta els atributs
        de recuperació especificats.
SEARCH: un UID o més indiquen els missatges que compleixen
        el criteri de recerca desitjat.
```

5. Experimental/expansió

a) Comanda experimental (`X<atom>`)

L'IMAP4 permet especificar comandes experimentals que no són una part de l'especificació, sempre que el seu nom comenci amb el prefix `X`.

```
X1*caràcter
```

19.4.4. Exemple

A continuació, es presenta un exemple de les comandes d'un client IMAP4 (en negreta) que accedeix a un servidor IMAP4 per a entrar a les seves bústies. En l'exemple es veu com es recupera la informació del missatge 12 (`fetch 12 full`). En la informació retornada pel

ANOTACIONS

servidor, es veu la informació "analitzada" de la capçalera. A continuació, es recupera tota la capçalera del mateix missatge (`fetch12 body [header]`). Al final, abans de tancar la connexió, es marca el missatge perquè sigui esborrat.

```
* OK IMAP4rev1 Service Ready
a001 login rmarti secret
a001 OK LOGIN completed
a002 select inbox
* 18 EXISTS
* FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
* 2 RECENT
* OK [UNSEEN 17] Message 17 is the first unseen message
* OK [UIDVALIDITY 3857529045] UIDs valid
a002 OK [READ-WRITE] SELECT completed
a003 fetch 12 full
* 12 FETCH (FLAGS (\Seen) INTERNALDATE "27-Jun-2003
02:44:25 -0700" RFC 822.SIZE 4286 ENVELOPE
("Fri, 27 Jun 2003 02:23:25 -0700 (PDT)"
"Exemple d'accés IMAP4rev1"
(("Jordi Inyigo" NIL "jinyigo" "uoc.edu"))
(("Jordi Inyigo" NIL "jinyigo" "uoc.edu"))
(("Jordi Inyigo" NIL "jinyigo"
((NIL NIL "rmarti" "uoc.edu"))
((NIL NIL "xperramon" "uoc.edu")
("JM Marques" NIL "jmmarques" "uoc.edu"))
NIL NIL
"<B27397-0100000@uoc.edu>")
BODY ("TEXT" "PLAIN" ("CHARSET" "US-ASCII") NIL NIL
   "7BIT" 3028 92))
a003 OK FETCH completed
a004 fetch 12 body[header]
* 12 FETCH (BODY[HEADER] {350}
Date: Fri, 27 Jun 2003 02:23:25 -0700 (PDT)
From: Jordi Inyigo <jinyigo@uoc.edu>
Subject: Exemple d'accés IMAP4rev1
To: rmarti@uoc.edu
cc: xperramon@uoc.edu,
JM Marques <jmmarques@uoc.edu>
Message-Id: <B27397-0100000@uoc.edu>
MIME-Version: 1.0
Content-Type: TEXT/PLAIN; CHARSET=US-ASCII

)
a004 OK FETCH completed
a005 store 12 +flags \deleted
* 12 FETCH (FLAGS (\Seen \Deleted))
a005 OK +FLAGS completed
a006 logout
* BYE IMAP4rev1 server terminating connection
a006 OK LOGOUT completed
```

19.5. Extensions multimèdia: el format MIME

La norma RFC 822 defineix un format de missatge i un contingut amb una única part de text en ASCII de 7 bits. Es va considerar que

ANOTACIONS

aquest format era molt pobre i que calia algun mètode per a superar-ne les limitacions.

El format **MIME** (*multipurpose Internet mail extensions*) redefineix el format del missatge per a permetre, sense perdre la compatibilitat amb el format definit per l'RFC 822, les característiques següents:

- Contingut de text no solament ASCII de 7 bits.
- Contingut no text.
- Contingut amb múltiples parts.
- Capçaleres amb text no solament ASCII de 7 bits.

19.5.1. Nous camps de capçalera

Per a permetre totes les noves extensions, el MIME defineix nous camps de capçalera: `MIME-Version`, `Content-Type`, `Content-Transfer-Encoding`, `Content-ID` i `Content-Description`.

Indicador de versió (`MIME-Version`)

El camp de capçalera `MIME-Version` indica la versió MIME que s'utilitza en el missatge (la versió de MIME que s'utilitza actualment, i que especifiquem en aquest apartat, és l'1.0). Aquest camp és útil perquè el receptor del missatge pugui interpretar els camps de capçalera MIME.

```
MIME-Version: 1*digit.1*digit
```

Indicador del tipus i subtipus de dades (`Content-Type`)

El camp de capçalera `Content-Type` permet indicar els tipus i subtipus de les dades que conté el missatge. D'aquesta manera, el receptor podrà conèixer els tipus de dades que conté el missatge i les podrà visualitzar adequadament. Segons el tipus o subtipus de dada, pot incloure algun paràmetre addicional.

```
Content-Type: tipus/subtipus *(; paràmetre)
```

ANOTACIONS

```
tipus = tipus-discret | tipus-compost
tipus-discret = text | image | audio | video |
application | extension-token
tipus-compost = message | multipart |
extension-token
```

Per a obtenir més informació sobre el format de missatges MIME, podeu consultar les obres següents:

N. Freed; **N. Borenstein** (1996, novembre). *RFC 2045 - Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies.*

N. Freed; **N. Borenstein** (1996, novembre). *RFC 2046 - Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types.*

K. Moore (1996, novembre). *RFC 2047 - Multipurpose Internet Mail Extensions (MIME) Part Three: Message Header Extensions for Non-ASCII Text.*

N. Freed; **J. Klensin**; **J. Postel** (1996, novembre). *RFC 2048 - Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures.*

N. Freed; **N. Borenstein** (1996, novembre). *RFC 2049 - Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples.*

La norma defineix cinc tipus de contingut (`tipus-discret`), amb els subtipus corresponents. Aquests tipus corresponen a dades monomèdia:

- `text`: per a informació de text.
- `image`: per a imatges.
- `audio`: per a so.

ANOTACIONS

- `video`: per a vídeo.
- `application`: per a qualsevol altre tipus de dades, per norma general en format específic d'alguna aplicació.

La norma també defineix dos tipus de dades compostes (*`tipus-compost`*):

- `multipart`: per a dades formades de moltes parts o per a dades independents dins d'un mateix missatge.

> **Nota**
>
> En els missatges de tipus `multipart`, el paràmetre *boundary* inclou una cadena, precedida per dos caràcters de guionet, "--", que s'utilitza com a separador entre les diferents parts del missatge.
>
> Per a indicar que s'han acabat totes les parts, s'utilitza la cadena del paràmetre *boundary*, precedida i seguida de dos caràcters de guionet, "--".

- `message`: per a encapsular un altre missatge dins del missatge.

La taula següent presenta els valors de tipus, subtipus i paràmetres per als `Content-Type` definits en la norma:

Taula 13.

Valors definits per a `Content-Type`		
Tipus	**Subtipus**	**Paràmetres**
`text`	`plain`	`charset = ISO-8859-(1 | ... | 9) | us-ASCII`
	`enriched`	`charset = ISO-8859-(1 | ... | 9) | us-ASCII`
`image`	`gif`	–
	`jpeg`	–
`audio`	`basic`	–
`video`	`mpeg`	`quicktime`
`application`	`octet-stream`	`type =` *`cadena`*`;` `padding =` *`sencer`*
	`postscript`	–
`multipart`	`mixed`	`boundary =` *`cadena`*
	`alternative`	`boundary =` *`cadena`*
	`parallel`	`boundary =` *`cadena`*
	`digest`	`boundary =` *`cadena`*

Valors definits per a `Content-Type`		
Tipus	**Subtipus**	**Paràmetres**
`message`	`rfc 822`	–

	`partial`	`id = cadena;`
		`number = sencer`
		`[;total = sencer ]`

	`external-body`	`access-type = ftp | anon-ftp | tftp |`
		`afs | local-file | mail-server`
		`[;expiration = date-time ]`
		`[;size = sencer ]`
		`[;permission=read | read-write]`
		`[;name = cadena ]`
		`[;site = cadena ]`
		`[;dir = cadena ]`
		`[;mode = netascii | octet | mail |`
		`ascii | ebcdic | image | localn ]`
		`[;server = cadena ]`
		`[;subject = cadena ]`

Especificador del tipus de codificació (`Content-Transfer-Encoding`)

En alguns protocols, com l'SMTP, la informació que s'envia ha de ser en ASCII de 7 bits. A vegades, les dades originals no tindran aquest format i, llavors, els serà necessari aplicar algun tipus de codificació abans d'enviar-les.

El camp de capçalera `Content-Transfer-Encoding` serveix per a especificar el tipus de codificació que s'ha aplicat al contingut del missatge perquè el receptor pugui descodificar-lo, si és necessari. La norma defineix diferents tipus de codificació:

```
Content-Transfer-Encoding: mecanisme
mecanisme = 7bit | 8bit | binary | quoted-printable | base64
```

a) Codificació `7bit | 8bit | binary`

Els mecanismes de codificació `7bit`, `8bit` i `binary` només serveixen per a indicar de quin tipus són les dades transmeses. En aquests casos, les dades no es codifiquen i, per norma general, s'utilitzen per a aplicacions que no restringeixen la representació de dades en 8 bits.

ANOTACIONS

b) Codificació `quoted-printable`

El mecanisme de codificació `quoted-printable` s'utilitza per a la representació i codificació de dades en ASCII de 7 bits, la majoria de les quals ja és de bytes representables en aquest format. És a dir, aquest mecanisme s'aplica quan la informació és majoritàriament de caràcters de text. Les normes bàsiques de codificació són les següents:

- Qualsevol byte es pot representar amb el caràcter "=" seguit per la notació hexadecimal en dos dígits (i lletres en majúscula) del valor del byte.

- Els bytes amb valors decimals entre 33-60 i 62-126, inclosos els quatre, es poden representar amb el caràcter ASCII corresponent.

c) Codificació `Base64`

El mecanisme de codificació `Base64` ofereix una codificació d'informació binària a ASCII de 7 bits que no hagi de ser llegible. Els algorismes de codificació i descodificació són molt simples.

El procés de codificació es duu a terme prenent grups de 3 bytes (24 bits). Mantenint l'ordre de bits original, aquests 24 bits es reagrupen en 4 blocs de 6 bits (6 bits = 64 combinacions).

El fitxer codificat s'obté prenent cada un d'aquests blocs de 6 bits i codificant-lo com un caràcter alfanumèric a partir del valor binari dels 6 bits, segons la taula següent:

Nota

La codificació de dades en `Base64` utilitza 4 caràcters per a cada 3 bytes. Per aquest motiu, `Base64` augmenta la mida de la informació un 33% (4/3).

Taula 14.

Taula de codificació `Base64`

Valor	Caràcter	Valor	Caràcter	Valor	Caràcter	Valor	Caràcter
0	'A'	26	'a'	52	'0'	62	'+'
1	'B'					63	'/'
...		...		...		pad	'='
25	'Z'	51	'z'	61	'9'		

Nota

Si s'ha de codificar un nombre de bytes que no sigui múltiple de 3 (la codificació, per tant, no seria múltiple de 4 bytes), es codifiquen tots els bytes i, al final de la cadena de caràcters ja codificada, s'afegeixen tants caràcters "=" (caràcter per a omplir, *pad*) com calgui (màxim dos) fins a arribar a un nombre de caràcters múltiple de 4.

En el procés de descodificació, s'agafen els caràcters alfanumèrics rebuts i es reconverteixen al seu valor binari corresponent (6 bits) segons la taula. Cada 4 caràcters rebuts donen lloc a 24 bits, i només cal reagrupar-los en 3 bytes per a generar el fitxer descodificat. Abans de descodificar tot el missatge, cal eliminar tots els caràcters "=" que es trobin al final.

Identificador del contingut del missatge (`Content-ID`)

El camp de capçalera `Content-ID` s'utilitza per a proporcionar un identificador únic al contingut del missatge. Amb aquest identificador, es fa referència al contingut de manera no ambigua.

```
Content-ID: id-msg
```

Informador descriptiu del contingut (`Content-Description`)

El camp `Content-Description` proporciona informació descriptiva del contingut en forma de text.

```
Content-Description: text
```

19.5.2. Extensions per a text no ASCII en les capçaleres

El MIME també permet codificar text no ASCII en les capçaleres dels missatges RFC 822 de manera no ambigua. Aquest mètode permet codificar tota la capçalera o només una part. El text (que pot ser de més d'un caràcter) ja codificat, precedit pel conjunt de caràcters i un

ANOTACIONS

identificador del tipus de codificació que s'ha aplicat, s'inclou al lloc corresponent de la capçalera entre els caràcters "=?" i "?=":

```
=? charset ? codificació ? text-codificat ?= charset =
ISO-8859- (1 |... | 9) | us-ASCII codificació = Q | B
```

Nota

a) **Conjunt de caràcters** (*charset*): és vàlid qualsevol dels caràcters permesos en el `Content-Type text/plain`.

b) **Codificacions**: el format MIME permet dos tipus de codificació similars a les codificacions que es poden aplicar en el cos del missatge:

- `Q`: és semblant al `Content-Transfer-Encoding quoted-printable`. Es recomana quan la majoria dels caràcters que s'han de codificar són ASCII.
- `B`: és idèntica al `Content-Transfer-Encoding Base64`. És adequada per a la resta de casos.

19.5.3. Missatges multipart

En els missatges multipart, cada una de les parts sol estar formada per una petita capçalera i el contingut. A la capçalera es poden trobar els camps `Content-Type`, `Content-Trans-fer-Encoding`, `Content-ID` i `Content-Description`. Tots es refereixen al contingut de la part en qüestió.

19.5.4. Exemple

En aquest exemple es pot llegir un missatge RFC 822 amb extensions MIME versió 1.0. És un missatge de dues parts: la primera és de text codificat amb `quoted-printable` i la segona és una imatge en format .gif codificada en `Base64`. Així mateix, es pot veure un camp de capçalera amb una lletra *í* codificada amb codificació `Q`.

ANOTACIONS

Nota

La *í* de *Ramon Martí* s'ha codificat amb codificació Q.

```
Date: 27 Jun 2003 0932 PDT
From: Jordi Inyigo <jinyigo@uoc.edu>
To: Ramon =?iso-8859-1?Q?Mart=ED?=
  <rmarti@uoc.edu>
MIME-Version: 1.0
Content-Type: multipart/mixed;
  boundary="=_250699_"

--=_250699_
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable

Això és un missatge RFC 822 que conté MIME.

--=_250699_
Content-Type: image/gif; name="Readme.gif"
Content-Transfer-Encoding: base64

R0lGODlhIAAgAIAAAAAAAP///yH5BAEAAAAALAAAAAAgACAAA
AJzjI+pywoQXoSywoaontzeRhnXKJYc82GOuaoL2brJlx7pg+
eSzc9yHAHqhpmi8ddLspCT3Y2pXNZ+UxpUpGNNZVsYdBsMf4l
BsMocuqLFyCHO5I6/MWY2sy7PL4F3+nU/kxcHePTl91dnR+WS
yAcimGVQAAA7

--=_250699_--
```

20. Servei de notícies: l'NNTP

El servei de notícies (en anglès, *news*) permet la tramesa de missatges, com el servei de correu electrònic, però amb la diferència que l'originador no especifica el destinatari o destinataris, sinó que qualsevol usuari amb accés al servei els pot llegir. Aquesta funcionalitat, doncs, es pot comparar a la d'un tauler d'anuncis, en el qual tothom pot llegir els missatges que hi ha penjats.

El servei de notícies a Internet es coneix amb el nom *Usenet*, que prové del de la xarxa en la qual es va desenvolupar originàriament. Actualment, la distribució de notícies s'ha estès per tot Internet, de manera que els anuncis enviats es poden llegir a qualsevol part del món. A la pràctica, tanmateix, és possible restringir l'àmbit en el qual es vol distribuir els missatges, ja que alguns només seran interessants, per exemple, en una zona geogràfica determinada.

20.1. El model NNTP

En el servei de notícies, la distribució dels missatges o anuncis, que a Usenet es denominen ***articles***, s'efectua de manera descentralitzada. Tenint en compte l'enorme volum de trànsit que genera aquest servei, no seria viable tenir un servidor central en què tothom deixés els seus articles i anés a llegir els dels altres. Per aquest motiu, s'utilitza un mecanisme de propagació en el qual l'autor d'un article l'envia a un servidor de notícies, que s'encarregarà de reenviar-lo a una sèrie de servidors pròxims o amb els quals estigui connectat directament, que al seu torn el reenviaran a altres servidors, i així successivament.

Amb aquest mecanisme de propagació, s'aconsegueix que un article estigui disponible per a ser llegit, idealment, a tots els servidors de

ANOTACIONS

la xarxa. Els usuaris que vulguin llegir els articles ho podran fer, doncs, connectant-se a qualsevol servidor, preferiblement al que tinguin més pròxim.

D'altra banda, els servidors només emmagatzemen cada article durant cert temps. Hi ha articles que tenen una data de caducitat predeterminada, i el servidor els esborra automàticament. D'altres s'esborren, per exemple, quan el servidor decideix que els usuaris que hi puguin estar interessats ja han tingut bastant temps per a llegir-los.

Lectura complementària

Si voleu més informació sobre l'NNTP, consulteu l'obra següent:

B. Kantor; P. Lapsley (1986, febrer). *RFC 977 - Network News Transfer Protocol.*

El mètode utilitzat per a propagar els articles té certes limitacions; per exemple, no es pot garantir que un article arribi a tots els servidors en un termini determinat (o, a vegades, que arribi a un servidor concret), és necessari controlar el camí per on passa cada article per a evitar bucles, etc. Tot i amb això, aquest mètode és més pràctic i eficient que la solució d'un sol servidor central.

Els servidors de notícies es comuniquen entre si per a intercanviar-se articles per mitjà de l'**NNTP** (*network news transfer protocol*), especificat en el document RFC 977.

Nota

Antigament s'utilitzaven altres mètodes de transmissió d'articles entre servidors, com l'UUCP (*Unix to Unix copy protocol*).

Nota

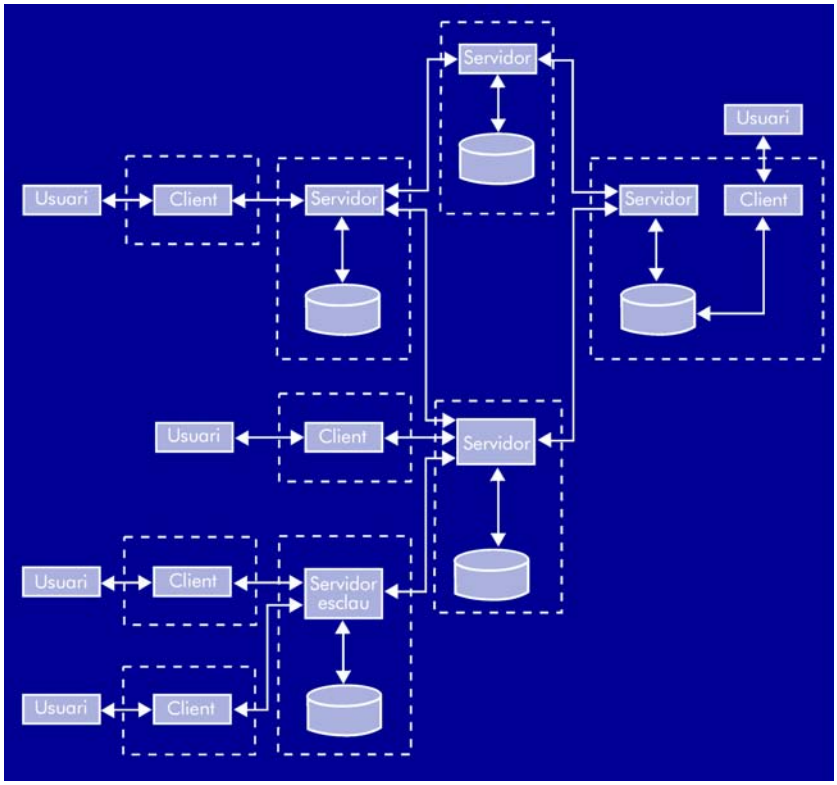
Un usuari que vulgui accedir al servei, tant per a enviar articles com per a llegir-los, pot ser que tingui accés directe a algun dels servidors de notícies. En aquest cas, la comunicació amb el servidor és un assumpte local. El cas més general, tanmateix, és que l'usuari vulgui accedir al servei des d'un altre sistema, que actuarà com a client. Llavors, la comunicació entre el client i el servidor es duu a terme també per mitjà de l'NNTP.

Tant si es tracta d'un client local, com d'un client remot, per norma general hi haurà un altre procés encarregat de la interfície amb l'usuari. Aquest tipus de programa se sol anomenar ***lector de notícies***.

ANOTACIONS

Així mateix, hi ha la possibilitat que un grup de clients accedeixi a un servidor central d'una organització per mitjà d'un servidor esclau:

Figura 88.



Nota

El servidor esclau pot millorar l'eficiència en l'accés, per exemple, emmagatzemant localment còpies dels últims articles llegits, per si els sol·liciten altres clients.

Els articles disponibles en un servidor s'han d'organitzar en grups, segons el seu tema, per a facilitar l'accés als usuaris que els vulguin llegir. Aquesta organització segueix un model jeràrquic: en el nivell més alt dels grups es troben els temes generals que, en un segon nivell, estan dividits en subtemes, que, al seu torn, poden estar subdividits en nivells inferiors.

Nota

És possible que un mateix article pertanyi a més d'un grup. Quan un usuari envia un article al sistema de

ANOTACIONS

notícies, a més de poder especificar en quin àmbit s'ha de distribuir, és necessari que indiqui obligatòriament a quin grup o grups el vol enviar.

La nomenclatura que s'utilitza per a designar els grups consisteix a concatenar els noms de cada nivell, de més alt a més baix, separant-los amb "." (per exemple, `news.announce` o `comp.os.linux.hardware`). En el nivell més alt de la jerarquia, els grups originals de Usenet eren els següents:

- `comp`: temes relacionats amb ordinadors i informàtica.
- `news`: articles sobre el sistema de notícies mateix.
- `rec`: activitats recreatives, *hobbies*, etc.
- `soc`: temes socials, culturals, humanístics, etc.
- `sci`: temes científics.
- `talk`: debats, discussions, opinions, etc.
- `misc`: altres temes no classificables en els apartats anteriors.
- `alt`: la jerarquia alternativa, que alguns aprofiten per a saltar-se les normes establertes en els grups "oficials".

Nota

Actualment, hi ha altres grups del nivell superior, com els que només contenen articles destinats a una regió geogràfica. Per exemple `eu` per a Europa, `es` per a Espanya, `fr` per a França, etc.

En la comunitat Usenet s'han establert una sèrie de regles, basades en un sistema de votacions, per a decidir la creació de grups nous o l'eliminació d'algun dels ja existents. Aquest caràcter de democràcia assembleària, i altres aspectes com la possibilitat de publicar escrits (fotografies, vídeos, etc.) o donar a conèixer les idees d'una persona de manera global i gairebé instantània (un somni que, fins a l'arribada d'Internet, només era a l'abast dels grans mitjans de comunicació), van fer de Usenet el gran fenomen sociològic d'Internet durant uns quants anys.

Avui dia el servei Usenet està eclipsat per l'enorme popularitat del servei WWW, que té una orientació molt més comercial, però no és tan participatiu.

ANOTACIONS

20.2. Conceptes bàsics de l'NNTP

Per norma general, l'NNTP utilitza el protocol de transport TCP. El número de port assignat al servei de notícies és el 119.

En l'NNTP, s'utilitza un esquema de peticions i respostes com el de l'SMTP. Cada **petició** constitueix una línia acabada amb <CRLF> que conté una comanda, possiblement amb paràmetres.

Nota

L'especificació NNTP estableix que les línies de comandes no han de tenir més de 512 caràcters. D'altra banda, les comandes i els paràmetres es poden escriure indistintament en majúscules o minúscules.

Les respostes també segueixen l'estructura general utilitzada en l'FTP o l'SMTP. Cada **resposta** es representa amb una línia que comença amb un codi numèric de tres dígits. A continuació, segons el codi de resposta, hi pot haver una sèrie de paràmetres seguits d'un text arbitrari opcional, fins al final de la línia, que acaba amb <CRLF>.

Els significats del primer dígit del codi de resposta són similars als de l'FTP i l'SMTP; els del segon són els següents:

Nota

Vegeu l'annex 4.

- **x0x**: resposta referent a la connexió, inicialització, etc.
- **x1x**: selecció d'un grup de notícies.
- **x2x**: selecció d'un article.
- **x3x**: distribució d'articles.
- **x4x**: tramesa d'articles.
- **x8x**: extensions no estàndard.
- **x9x**: missatges informatius de prova (*debugging*).

Algunes respostes van seguides d'un text format per una seqüència de línies. En aquest cas, cada línia acaba amb <CRLF>, i el final de la seqüència s'indica amb una línia que només conté el caràcter "." abans de <CRLF>. Si alguna de les línies de la resposta ha de començar amb ".", l'emissor hi insereix un altre "." al principi. És a dir, el receptor ha d'eliminar tots els "." inicials que trobi abans del final de la resposta.

Quan el client estableix la connexió, el servidor respon amb un codi 200 o 201 per a indicar que permet que el client enviï articles o que no ho permet, respectivament.

ANOTACIONS

20.3. Format dels articles

Lectura complementària

Si voleu més informació sobre el format dels articles Usenet, consulteu l'obra següent:

M.R. Horton; R. Adams (1987, desembre). *RFC 1036 - Standard for interchange of USENET messages.*

El format dels articles Usenet està definit en l'especificació RFC 1036, i es basa en el dels missatges de correu electrònic (definit en l'especificació 822), encara que amb algunes restriccions addicionals. Per tant, cada article consta d'una capçalera formada per una sèrie de camps, i d'un cos separat d'aquesta per una línia en blanc.

Hi ha sis camps obligatoris en la capçalera, que són els que s'exposen a continuació:

- `From`: adreça electrònica de l'originador de l'article (i, opcionalment, també el seu nom).
- `Date`: dia i hora en què s'ha originat l'article.
- `Newsgroups`: llista dels grups als quals s'envia l'article, separats per comes si n'hi ha més d'un.
- `Subject`: assumpte de què tracta l'article, que s'utilitzarà com a títol.
- `Message-ID`: identificador únic de l'article. A part d'aquest identificador únic, per a facilitar l'accés als articles, cada servidor els assigna un identificador local consistent en el nom del grup i un número correlatiu dins del grup. Per tant, si un article s'ha enviat a diferents grups, tindrà més d'un identificador. Aquests identificadors només són significatius per a un servidor, ja que un altre servidor probablement assignarà identificadors locals diferents dels mateixos articles.
- `Path`: llista de servidors pels quals ha passat l'article, separats amb símbols de puntuació que no siguin "`.`" (per norma general, s'utilitza el caràcter "`!`"). Cada servidor ha d'afegir el seu nom al principi de la llista. El valor d'aquest camp permet que un servidor sàpiga si l'article ja ha passat per un altre servidor i, per tant, no és necessari tornar-l'hi a enviar.

La capçalera d'un article també pot incloure els camps opcionals següents:

- `Reply-To`: adreça a la qual s'han d'enviar els missatges de correu en resposta a l'article (per defecte, la mateixa del camp `From`).

ANOTACIONS

- `Sender`: identificació de l'usuari que ha enviat l'article al sistema de notícies, si no coincideix amb el del camp `From`.
- `Followup-To`: llista de grups de notícies, separats per comes, als quals s'han d'enviar els articles de resposta (per defecte, els mateixos del camp `Newsgroups`).
- `Expires`: data i hora en què l'article es pot considerar caducat i, per tant, pot ser esborrat pels servidors (en absència d'aquest camp, el servidor decideix quan l'esborrarà).
- `References`: llista d'identificadors d'altres articles als quals es fa referència en l'article.
- `Control`: serveix per a indicar que l'article conté un missatge de control del sistema de notícies. El missatge s'especifica en el valor d'aquest camp i només és interessant per als servidors (els usuaris no necessiten llegir-los). Els missatges de control serveixen per a cancel·lar articles, crear i esborrar grups, informar de quins articles té un servidor i quins demana un altre, etc.
- `Distribution`: llista de distribucions a la qual s'ha d'enviar l'article. Cada distribució és un conjunt de servidors que, per norma general, pertanyen a una mateixa àrea geogràfica, a una mateixa organització, etc.
- `Organization`: nom de l'organització (empresa, institució, etc.) a la qual pertany l'originador de l'article.
- `Keywords`: llista de paraules clau relacionades amb el contingut de l'article.
- `Summary`: breu resum del contingut de l'article.
- `Approved`: alguns grups de notícies són moderats, la qual cosa significa que els usuaris no poden enviar els articles directament. En aquest cas, els articles s'han d'enviar per correu electrònic a una persona, anomenada *moderador del grup*, que és l'única que està autoritzada per a posar articles en el grup. De la totalitat de missatges que rep, el moderador decideix quins envia al grup i quins no. Els missatges que finalment s'envien han de portar en el camp `Approved` l'adreça de correu del moderador.
- `Lines`: nombre de línies del cos de l'article.

ANOTACIONS

- `Xref`: aquest camp el genera cada servidor i no s'ha de transmetre d'un servidor a l'altre. Conté el nom del servidor i una llista d'identificadors locals que té el mateix article en altres grups als quals s'hagi enviat. Una vegada que l'usuari ha llegit l'article, aquest camp permet als lectors de notícies marcar-lo com a llegit en tots els grups en els quals aparegui.

L'especificació RFC 1036 permet que la capçalera inclogui altres camps no estàndard.

Exemple

Exemple d'article Usenet

```
From: usuari@acme.com (Ernest Udiant)
Path: News.uoc.es!news.rediris.es!
  news.eu.net!newsfeed.omninet.
  org!nntp.acme.com!usuari
Newsgroups: soc.culture.espanol
Subject: Nou llibre de receptes tradi-
  cionals
Message-ID: <KvjsSAam9Ly@acme.com>
Date: Thu, 12 Dec 2002 09:12:30 GMT
Expires: Sat, 19 Dec 2002 00:00:00 GMT
Organization: ACME Inc.
Lines: 2

La setmana que ve es publicarà un llibre
nou de receptes de cuina tradicional.
Continuarem informant...
```

20.4. Ordres de l'NNTP

Nota

Consulteu la taula dels codis de resposta al final del subapartat.

A continuació, es detallen les comandes definides en l'especificació RFC 977 de l'NNTP:

1. **Fer una llista de grups (`LIST`)**

Aquesta comanda serveix per a obtenir una llista dels grups de notícies disponibles. El servidor respon amb un codi 215 i, a continuació,

envia una seqüència de línies (acabada amb "."), una per a cada grup de notícies disponible, cada una amb el format següent:

```
grup últim primer permís
```

Nota

- `grup`: nom del grup.
- `últim`: número correlatiu (identificador local) de l'últim article que hi ha en aquest grup.
- `primer`: número del primer article.
- `permís`: pot ser `i` o `n` per a indicar si es poden enviar articles a aquest grup o no, respectivament.

```
LIST
```

2. Seleccionar un grup (GROUP)

Aquesta comanda permet seleccionar un grup concret. El servidor envia una resposta 211 amb els paràmetres següents (per aquest ordre): nombre estimat d'articles del grup, número del primer article, número de l'últim i nom del grup. Si el grup no existeix, la resposta és 411.

Nota

El nombre d'articles que hi ha en el grup no té per què coincidir amb la diferència entre el número de l'últim article i l'anterior al primer, ja que pot ser que s'hagin esborrat articles intermedis.

```
GROUP grup
```

3. Llegir l'article (ARTICLE)

El paràmetre d'aquesta comanda pot ser un identificador únic d'article o un número d'article dins del grup actualment seleccionat (l'article llegit passa a ser considerat com l'article actual). Sense paràmetre, aquesta comanda llegeix l'article actual. El servidor respon amb un codi 220 seguit de dos paràmetres: el número de l'article i el seu identificador únic. A continuació, envia el contingut de l'article (capçalera i cos), acabat en

ANOTACIONS

"." . Si hi ha algun error, el servidor respon amb els codis 412, 420, 423 o 430, segons el cas.

```
ARTICLE [ < identificador > |
  número]
```

4. Llegir la capçalera (HEAD)

Aquesta comanda és idèntica a ARTICLE; però el servidor només torna la capçalera de l'article. El codi de resposta serà el 221 en lloc del 220, amb els mateixos paràmetres.

```
HEAD [ < identificador > |
  número>]
```

5. Llegir el cos (BODY)

Aquesta comanda és idèntica a ARTICLE; però el servidor només torna el cos de l'article. El codi de resposta serà el 222 en lloc del 220, amb els mateixos paràmetres.

```
BODY [ < identificador > |
  número]
```

6. Obtenir l'estatus (STAT)

Aquesta comanda és idèntica a ARTICLE; però el servidor no torna cap text, només la línia de resposta. El codi serà el 223 en lloc del 220, amb els mateixos paràmetres.

```
STAT [ < identificador > |
  número]
```

7. Seleccionar l'article següent (NEXT)

Aquesta comanda selecciona l'article següent del grup i fa que passi a considerar-se l'article actual (és a dir, el que es llegirà si s'envia la comanda ARTICLE sense paràmetre). El codi de resposta normal serà el 223, com en l'ordre STAT. Un codi d'error específic és el 421.

```
NEXT
```

ANOTACIONS

8. Seleccionar l'article anterior (LAST)

Aquesta comanda selecciona l'article anterior a l'actual. Els codis de resposta són com els de la comanda NEXT, però canviant el 421 pel 422.

```
LAST
```

9. Fer una llista de grups nous (NEWGROUPS)

NEWGROUPS disposa de les mateixes funcions que la comanda LIST, excepte que el codi de resposta és 231 (en lloc de 215) i la llista de grups està restringida als que s'han creat després de la data indicada pels paràmetres (el primer són sis dígits que representen any, mes i dia, i el segon, sis més que representen hora, minuts i segons). Opcionalment, es pot restringir encara més la llista especificant una o més jerarquies d'alt nivell separades per comes (per exemple, news, rec, etc.).

```
NEWGROUPS aammdd hhmmss
[GMT][< jerarquies >]
```

10. Fer una llista d'articles nous (NEWNEWS)

El servidor respon a aquesta comanda amb un codi 230 i, a continuació, envia una seqüència de línies (acabada amb "."), una per a cada article dels grups especificats en el primer paràmetre que s'hagi enviat o rebut després de la data indicada pel segon i el tercer. Les línies contenen els identificadors únics dels articles.

```
NEWNEWS grups aammdd hhmmss
```

El primer paràmetre és un nom de grup o una llista de noms separats per comes. En un nom, pot aparèixer el caràcter especial "*". En aquest cas, es considera que equival a tots els noms de grups que, en lloc del "*", tinguin una seqüència qualsevol de caràcters, que pot incloure el separador "." (per exemple, alt.biz* pot equivaler a alt.biz.misc, alt.bizarre, etc.). D'altra banda, si un nom té el prefix "!", el grup o grups corresponents s'exclouran de la llista (per exemple, comp.lang.*,

ANOTACIONS

`!comp.lang.c.*` equival als grups que comencin per `comp.lang.`, però no per `comp.lang.c.`).

Opcionalment, es pot restringir la llista d'articles a un conjunt d'una o més jerarquies si s'especifiquen en l'últim paràmetre separades per comes. La llista només contindrà articles que pertanyin almenys a un grup de les jerarquies.

11. Enviar un article (POST)

Aquesta comanda s'utilitza per a enviar un article. El servidor respon amb un codi 440 per a indicar que no es permet l'operació, o amb un 340 per a sol·licitar l'article. En el segon cas, el client ha d'enviar el contingut de l'article (capçalera i cos) amb el format definit en l'especificació RFC 1036 i acabat amb una línia en què només hi hagi un ".". Llavors, el servidor enviarà la resposta definitiva, que serà 240 o, si hi ha hagut algun error, 441.

```
POST
```

12. Oferir un article (IHAVE)

Quan un servidor es connecta a un altre, pot utilitzar la comanda IHAVE per a fer saber de quins articles disposa.

```
IHAVE < identificador >
```

A cada article, el servidor receptor enviarà en la resposta un codi 335 si en vol rebre una còpia o 435 si no li interessa. Si cal passar l'article, es fa mitjançant la comanda POST, i el servidor receptor enviarà en la resposta un codi 235 o, en cas d'error, un 436. Així mateix, pot succeir que al receptor no li interessi l'article després d'examinar-ne el contingut; en aquest cas, la resposta serà 437.

Nota

Un servidor no hauria d'oferir a cap altre servidor articles que ja li hagi enviat abans o que ja hagin passat per aquest últim (el camp `Path` de la capçalera ho indica). Aquesta situació es coneix com *doble oferiment*.

ANOTACIONS

13. Connexió des d'un servidor esclau (SLAVE)

Aquesta comanda serveix per a informar el servidor que la connexió prové d'un servidor esclau. El servidor pot decidir, per exemple, do-

nar més prioritat a aquesta connexió que a les que provenen directament de clients perquè se suposa que atén més usuaris. El codi de resposta és el 202.

```
SLAVE
```

14. Missatge d'ajuda (HELP)

Aquesta comanda s'utilitza per a obtenir un missatge d'ajuda. El servidor respon amb un codi 100 i, a continuació, un text informatiu de les comandes que accepta, acabat amb una línia amb un caràcter ".".

```
HELP
```

15. Tancar la connexió (QUIT)

Aquesta comanda s'utilitza per a tancar la connexió. El servidor respon enviant un codi 205 i tancant la connexió.

```
QUIT
```

L'especificació RFC 977 permet que les implementacions afegeixin altres comandes als estàndards; tanmateix, recomana que el nom d'aquestes noves comandes comenci amb X per a evitar possibles conflictes amb posteriors extensions oficials.

Alguns exemples de comandes implementades per molts servidors són els següents:

- XGTITLE: dóna un títol descriptiu d'un o més grups.
- XHDR i XOVER: donen, respectivament, el valor d'un camp i un resum de la capçalera d'un o més articles.
- XPAT: proporciona els valors d'un camp de la capçalera que concorden amb un patró, juntament amb els noms dels articles corresponents.

ANOTACIONS

La taula següent resumeix els codis de resposta del protocol NNTP:

Taula 15.

Codis de resposta	
Codi	**Significat**
100	Missatge d'ajuda.
200	Servidor preparat; es permet enviar articles.
201	Servidor preparat; no es permet enviar articles.
202	Connexió des de servidor esclau.
205	El servidor tanca la connexió.
211	Grup seleccionat.
215	Llista de grups.
220	Contingut de l'article.
221	Capçalera de l'article.
222	Cos de l'article.
223	Identificador de l'article.
230	Llista d'articles nous.
231	Llista de grups nous.
235	Article rebut.
240	Article enviat.
335	Preparat per a rebre un article d'un altre servidor.
340	Preparat per a rebre un article del client.
400	Servei no disponible.
411	No existeix el grup.
412	No hi ha cap grup seleccionat.
420	No hi ha cap article seleccionat.
421	No hi ha article següent.
422	No hi ha article anterior.
423	No hi ha cap article amb aquest número.
430	No hi ha cap article amb aquest identificador.
435	No es vol rebre l'article.
436	Error a la recepció de l'article.
437	Article rebutjat.
440	No es permet enviar article.
441	No s'ha pogut enviar l'article.
500	Comanda desconeguda.

ANOTACIONS

Codis de resposta	
Codi	**Significat**
501	Error de sintaxi en la comanda.
502	Permís denegat.
503	Error intern.

Activitat

Establiu una connexió amb un servidor NNTP (per exemple, amb `telnet` *`servidor`* 119). Efectueu algunes operacions com ara fer una llista dels grups que hi hagi (atenció: en alguns servidors la llista pot tenir més de 10.000 línies!), entrar en un grup, veure la capçalera d'algun article, llegir-ne algun de sencer, etc., i observeu els codis numèrics de resposta que envia el servidor.

ANOTACIONS

21. Servei hipermèdia: WWW

21.1. Documents hipermèdia

El **servei WWW** (World Wide Web) ofereix accés a informació multimèdia, que pot incloure continguts de diferents tipus (text, imatges, àudio, vídeo, etc.) i referències a altres elements d'informació, segons el model dels sistemes hipertext.

Un **sistema hipertext** permet recórrer un document de manera no necessàriament lineal o seqüencial, sinó seguint les referències o enllaços que l'usuari seleccioni i saltant a la part referenciada. És a dir, en lloc de seguir un únic camí lineal, es pot fer un recorregut pel document passant pels arcs d'un graf. Aquesta manera d'accedir a la informació se sol anomenar familiarment *navegar*.

Quan es generalitza la funcionalitat hipertextual perquè permeti navegar per elements d'informació multimèdia i no solament per text, se sol parlar de **sistemes hipermèdia**.

En la terminologia WWW, cada un dels elements d'informació als quals es pot accedir s'anomena *recurs*. Els **documents hipermèdia** són un cas particular de recurs que conté informació estructurada, possiblement amb diferents tipus de contingut i enllaços a altres recursos.

En l'estructura d'un document, en general, es poden distingir els elements següents:

1. **L'estructura lògica**; és a dir, la divisió del contingut en parts que es troben en diferents nivells, com ara capítols, seccions, apartats, subapartats, títols, paràgrafs, figures, taules, encapçalaments, notes, etc.

ANOTACIONS

2. **L'estructura física**; és a dir, la disposició de les parts del document en el medi de presentació (per norma general, paper o pantalla): la divisió del document en pàgines i grups de pàgines, la distribució de l'àrea de cada pàgina en zones per a capçaleres i peus, els marges, els espais reservats a figures, taules, etc.

En el model general de processament dels documents estructurats, l'autor crea l'estructura lògica, amb el contingut corresponent, i hi pot incloure certes directrius per a controlar l'aspecte visual que haurà de tenir el document. Moltes vegades, es denomina *estil* un conjunt determinat de directrius d'aquest tipus. A partir d'aquesta informació, un procés denominat ***de formatació*** s'encarrega de generar l'estructura física.

Hi ha diferents **representacions normalitzades dels documents estructurats**, les quals van destinades a objectius diferents:

a) Representacions que se centren en l'especificació de l'estructura lògica, com l'**SGML** (*standard generalized markup language*, publicat en l'estàndard internacional ISO 8879).

b) Representacions que se centren en l'especificació de l'estructura física, com l'**SPDL** (*standard page description language*, publicat en l'estàndard ISO/IEC 10180).

c) Representacions que comprenen tant l'estructura lògica com la física, per exemple l'**estàndard ODA** (*open document architecture*, publicat en l'estàndard ISO/IEC 8613 i en la sèrie de recomanacions ITU-T T.410).

Nota

SPDL es basa en un altre llenguatge molt popular que serveix per a representar documents formatats: el llenguatge PostScript.

21.2. Marcatge: l'SGML

Nota

DSSSL és la sigla de *document style semantics and specification language*, publicat en l'estàndard ISO/IEC 10179

L'SGML conté l'especificació d'un llenguatge que només permet representar l'estructura d'un document des del punt de vista lògic. En principi, un usuari pot aplicar els estils que vulgui per a visualitzar el document. Tanmateix, el més habitual és que les aplicacions que utilitzen aquest llenguatge proporcionin una sèrie de regles per a interpretar l'estructura del document i, en particular, per a formatar-lo. D'altra banda, s'ha publicat un altre estàndard, anomenat **DSSSL**,

que permet definir estils de presentació amb les transformacions necessàries per a convertir un document SGML en una altra notació, com l'SPDL.

Un exemple d'aplicació basada en SGML és l'**HTML** (*hypertext markup language*) dissenyat per a especificar documents hipermèdia.

Nota

Actualment, l'especificació oficial de l'HTML està sota el control d'un grup d'empreses i organitzacions conegut com a World Wide Web Consortium (W3C). Anteriorment, la responsabilitat era d'un grup de treball de la Internet Engineering Task Force (IETF), que va cessar les seves activitats el 1996 després de publicar el document RFC 1866 amb l'especificació HTML 2.0.

L'SGML es va publicar el 1986 amb l'objectiu de facilitar l'especificació de documents estructurats. El llenguatge definit en aquest estàndard, és prou general per a permetre la representació de molts altres tipus d'informació estructurada (bases de dades, etc.).

Lectura complementària

Podeu ampliar la informació sobre l'SGML en l'obra següent:

C. Goldfarb (1990). *The SGML Handbook*. Oxford: Oxford University Press.

21.3. Transferència d'hipermèdia: l'HTTP

L'HTTP (*hipertext transfer protocol*) és la base del servei WWW. Els seus orígens van ser els treballs d'un grup d'investigadors del CERN (Centre Europeu d'Investigació Nuclear) que, al final dels anys vuitanta i començament dels noranta, van definir un protocol per accedir de manera senzilla i còmoda a la informació distribuïda a les diferents seus del centre. Quan el conjunt de sistemes accessibles amb aquest mecanisme es va estendre a altres organitzacions i països del món, va néixer el que avui coneixem com a WWW (World Wide Web).

De la mateixa manera que l'HTML, l'HTTP també evoluciona constantment a mesura que els implementadors hi incorporen funcionalitats noves per a adaptar-lo a diferents tipus d'aplicacions particulars. El 1996, es va publicar el document RFC 1945, en el qual es defineix la versió HTTP/1.0 (aquest document també conté les especificacions de

ANOTACIONS

la denominada *versió 0.9* per a facilitar la interoperabilitat amb implementacions antigues). Tanmateix, l'any següent ja es va publicar l'especificació HTTP/1.1 en el document *RFC 2068*.

A continuació, veurem les característiques principals de la funcionalitat que ofereix l'HTTP. Tanmateix, abans d'entrar en els detalls del protocol, estudiarem el mètode general utilitzat en el servei WWW per a identificar la informació a què es vol accedir, els identificadors uniformes de recurs (URI).

21.3.1. Identificadors uniformes de recurs (URI)

Nota

La definició d'URI constitueix un altre exemple d'especificació en evolució constant, almenys fins que es va publicar el document *RFC 2396*, el 1998.

Des d'un document es poden referenciar recursos especificant-ne les adreces, que es representen per mitjà del que en la terminologia WWW es denomina *identificador uniforme de recurs* o *URI*.

La forma general d'un URI es pot expressar de la manera següent:

```
esquema: identificador
```

Aquesta és la forma corresponent a un **URI absolut**. La paraula que es troba abans del separador "`:`" és l'esquema, que determina la sintaxi de l'identificador. Aquesta sintaxi també pot permetre l'especificació d'**URI relatius**, en què s'omet l'esquema i el separador "`:`".

Un URI pot ser un **localitzador** (**URL**), si especifica com s'accedeix al recurs, i/o un **nom** (**URN**) si identifica el recurs per mitjà d'un conjunt d'atributs:

```
ftp://[usuari[:contrasenya]@]servidor/camí
```

Segons l'esquema, aquestes són algunes sintaxis possibles dels URL:

a) `ftp`

Aquest tipus d'URL identifica un recurs accessible per mitjà del protocol FTP: `servidor` és el nom de l'ordinador servidor i `camí` és el

nom complet d'un fitxer o directori. Si no hi ha nom d'usuari s'utilitza `anonymous` i, com a contrasenya, qualsevol cadena de caràcters.

b) **news**

Aquest URL identifica un conjunt d'articles si `identificador` és el nom d'un grup de notícies Usenet; en el cas que només sigui un identificador de missatge (camp `Message-ID`), llavors només identificarà un article determinat:

```
News: identificador
```

c) **mailto**

Aquest identificador URL representa una adreça electrònica:

```
mailto: adreça
```

d) **telnet**

Aquest URL representa un servei accessible per mitjà del protocol Telnet en el servidor i el port especificats:

```
telnet://[usuari[:contrasenya]@]servidor[:port][/]
```

e) **http**

Quan s'hagi d'accedir al recurs identificat per mitjà de l'HTTP, s'utilitzarà un URL d'acord amb la sintaxi següent:

```
URL-HTTP = URL-HTTP-absolut | URL-HTTP-relatiu
URL-HTTP-absolut = http://servidor[:port]
[camí-absolut]
URL-HTTP-relatiu = camí-absolut | camí-relatiu
camí-absolut = /camí-relatiu
camí-relatiu = [camí] *(;paràmetre)[? consulta]
[# fragment]
camí = segment *(/segment)
```

ANOTACIONS

En aquesta sintaxi, `servidor` és el nom de l'ordinador amb què cal establir la connexió HTTP per a accedir al recurs, i `port` és el número de port en què s'ha d'efectuar la connexió que, per defecte, és el número assignat al servei WWW, és a dir, el 80.

Nota

`segment`, `paràmetre`, `consulta` o `fragment` poden ser una cadena buida.

Cada un dels segments que formen el `camí` s'associa, per norma general, a un nom de directori o de fitxer. Aquests noms no poden incloure els caràcters "/", ";", "?", "\", "#", "", "<", ">" o espai. Podem trobar el caràcter "/" en un paràmetre, i tant "/" com ";" i "?" podem trobar-los en els components `consulta` i `fragment`.

Si al `camí` apareix el caràcter "%", ha d'anar seguit de dos dígits hexadecimals, i aquesta seqüència de tres caràcters equival al caràcter que tingui el codi indicat pels dígits. Per tant, es pot utilitzar una seqüència "%HH" per a representar qualsevol dels caràcters no permesos en un URL, incloent-hi el mateix caràcter "%" (que es representaria amb "%25").

Nota

A continuació, presentem diferents exemples d'URL amb esquemes diferents (el dos últims són URL HTTP relatius):

```
ftp://ftp.uoc.es/pub/doc/README
news:comp.infosystems.www.misc
mailto:Ernest.Udiant@campus.uoc.edu
http://www.uoc.es/
http://www.acme.com/%7Eadmin/
http://www.acme.com/buscador/busca.cgi?nom=Internet
http://www.acme.com/doc/ayuda.html#buscador
/doc/ayuda.html#buscador
ayuda.html#buscador
```

Els URL relatius es poden utilitzar en documents HTML, en concret en els atributs que representen adreces d'altres recursos (HREF, SRC, ACTION). Aquests URL es consideren relatius a una adreça base determinada, que és la indicada per l'element BASE del document o, si no hi és aquest element, l'adreça que s'ha utilitzat per a accedir al mateix document HTML.

ANOTACIONS

Per a obtenir l'URL absolut corresponent, és necessari aplicar les regles següents:

1. Si l'URL relatiu conté un camí absolut, se substitueix el camí de l'adreça base.

2. Si l'URL relatiu conté un camí relatiu (i el camí no és buit), se substitueix l'últim segment de l'adreça base (la part del camí que hi hagi després de l'últim "/") per aquest. Com a cas especial, cada segment amb nom "`..`" que hi hagi a l'inici del camí relatiu s'ha de suprimir juntament amb l'últim segment que quedi al final del camí de l'adreça base.

3. Si el camí de l'URL relatiu és buit, però hi ha algun dels altres components (paràmetres, consulta, fragment), es pren el camí de l'adreça base i se substitueixen els components corresponents pels d'aquell.

Nota

Si l'URL relatiu és completament buit, es pren directament tota l'adreça base, incloent-hi el fragment, si n'hi ha (en els altres casos, el fragment de l'adreça base no es considera).

Nota

Interpretació de l'URL relatiu

Donada l'adreça base `http://www.uoc.edu/extern/ct/home/home.html`, els URL relatius següents s'haurien d'interpretar d'aquesta manera:

Taula 16.

URL relatiu	URL absolut
/accesset98/	http://www.uoc.edu/accesset98/
altres/pagina.html	http://www.uoc.edu/extern/ct/home/altres/pagina.html
../../logo.jpeg	http://www.uoc.edu/extern/logo.jpeg
#final	http://www.uoc.edu/extern/ct/home/home.html#final

21.3.2. Conceptes bàsics de l'HTTP

L'HTTP segueix el model general de peticions i respostes entre un client i un servidor. Es basa en un servei de transport fiable, però és independent del mecanisme de transport concret utilitzat. No obstant això, el més habitual és que client i servidor es comuniquin per mitjà

ANOTACIONS

del TCP. En aquest cas, el port per defecte per a establir les connexions és l'assignat oficialment en el servei WWW; és a dir, el 80.

En l'HTTP/1.0, el client estableix una connexió amb el servidor i li envia un missatge HTTP amb la petició; i, a continuació, el servidor envia al client un altre missatge HTTP amb la resposta i tanca la connexió. Si vol efectuar més peticions, el client ha d'establir una nova connexió per a cada una. En l'HTTP/1.1, en canvi, és possible intercanviar diferents peticions i respostes en una mateixa connexió que es denomina *connexió persistent*. Aquest és el mode de funcionament per defecte en l'HTTP/1.1.

Un **missatge HTTP** consta d'una primera línia, en la qual hi ha informació específica del protocol, seguida d'un missatge amb el mateix format que els missatges de correu electrònic, segons l'especificació RFC 822. És a dir, després de la primera línia hi ha d'haver una capçalera formada per una sèrie de camps, una línia en blanc i un cos. En casos particulars, la capçalera i/o el cos poden ser buits, però la línia en blanc que els separa sempre hi ha de ser present.

El cos del missatge, juntament amb els camps de la capçalera que proporcionen informació sobre el seu contingut, formen el que en l'HTTP es denomina una *entitat*. Cada entitat correspon a un recurs.

Depenent de si el missatge HTTP és una petició o una resposta, la primera línia rep el nom de *línia de petició* o *línia d'estatus*, respectivament.

La **sintaxi d'una línia de petició** és la següent:

mètode <SP> *URI* <SP> *versió* <CRLF>

En aquesta línia, `mètode` especifica quin tipus d'operació (o, en la terminologia HTTP, quin mètode) sol·licita el client, `URI` identifica el recurs al qual s'ha d'aplicar l'operació i `versió` ha de ser la cadena `HTTP/1.0` o `HTTP/1.1`, segons la versió del protocol. L'URI ha de ser un URL HTTP relatiu que contingui un camí absolut; és a dir, que comenci per "/" (excepte quan el servidor sigui un *proxy*, com veurem més endavant).

ANOTACIONS

Nota

Sempre que un usuari vulgui utilitzar un client per a obtenir el recurs donat per l'adreça `http://www.acme.com:8000/doc/principal.html`, el client s'ha de connectar al port 8.000 del servidor `www.acme.com` i enviar-li una petició que comenci amb aquesta línia:

```
GET /doc/principal.html HTTP/1.0
```

En aquest cas, el mètode utilitzat és GET (obtenir recurs). Ara bé, si l'adreça fos `http://www.acme.com:8000/`, la línia que s'hauria d'enviar seria simplement la següent:

```
GET / HTTP/1.0
```

La **sintaxi de les línies d'estatus** és la següent:

```
versió <SP> codi <SP> frase <CRLF>
```

La cadena `versió` és igual que en la línia de petició, el codi és un nombre de tres dígits, els primers del qual indiquen de quin tipus de resposta es tracta, i `frase` és un text explicatiu en format lliure, intel·ligible per als usuaris humans.

Els possibles **codis d'una resposta HTTP/1.0** són els següents:

Taula 17.

Codi	Significat
200	Operació efectuada.
201	S'ha creat un nou recurs.
202	Petició acceptada.
204	La resposta és buida.
301	El recurs sol·licitat ha canviat d'adreça.
302	El recurs sol·licitat ha canviat temporalment d'adreça.
304	El contingut del recurs no ha canviat.

Nota

En l'HTTP, el significat del primer dígit del codi de resposta varia una mica respecte del que es considera comú (que trobareu en l'annex 4) i és el següent:

- 1: Informació.
- 2: Èxit.
- 3: Redireccionament.
- 4: Error en el client.
- 5: Error en el servidor.

Codi	Significat
400	Petició incorrecta.
401	Usuari no autoritzat.
403	Accés prohibit.
404	No s'ha trobat el recurs.
500	Error intern del servidor.
501	Operació no implementada.
502	Error en un altre servidor.
503	Servei no disponible actualment.

Nota

En l'HTTP/1.1 s'han afegit a aquesta llista fins a 22 codis nous de resposta. No obstant això, si un client rep un codi *xyz* que no coneix, l'ha d'interpretar com si fos *x*00. En aquest sentit, el codi 100 significa 'continuar' i el codi 300 vol dir 'recurs accessible en una adreça o més'.

Emmagatzematge local de respostes (memòria cau)

Per a optimitzar el trànsit de la xarxa, és habitual que els clients HTTP desin una còpia de les respostes que reben dels servidors, juntament amb l'URL que han utilitzat en la petició. El magatzem destinat a aquesta finalitat rep el nom de ***memòria cau***. Si l'usuari vol tornar a accedir a un recurs que ja havia sol·licitat prèviament, el client pot utilitzar la còpia de la memòria cau en lloc de tornar a enviar la mateixa petició al servidor.

El client pot decidir, o l'usuari pot configurar, quant de temps es desaran els elements a la memòria cau, la seva dimensió màxima o la política que s'haurà de seguir per a esborrar-ne els elements quan sigui plena. Així mateix, l'HTTP també proporciona directrius per a determinar quins recursos es poden desar en la memòria cau i quins no, o quant temps es pot desar com a màxim un recurs determinat.

No solament els clients poden guardar còpies de les respostes obtingudes dels servidors; també ho poden fer els *proxies*.

ANOTACIONS

Nota

Consulteu els servidors intermediaris en l'apartat 21.3.4 d'aquesta unitat.

Capçaleres dels missatges HTTP

Els camps que hi pot haver en la capçalera d'un missatge HTTP es poden classificar en quatre grups:

1. Camps generals que poden ser presents tant en els missatges de petició, com en els de resposta.
2. Camps referents a l'entitat continguda en el cos del missatge i que també poden ser presents en peticions i respostes.
3. Camps propis de les peticions.
4. Camps propis de les respostes.

L'HTTP proporciona un mecanisme d'extensió que permet incloure camps no estàndard en les capçaleres. Si una implementació no reconeix un camp determinat, el que ha de fer és ignorar-lo (o retransmetre'l tal com es troba, si és un servidor intermediari).

Els camps que pot contenir la capçalera són els següents:

1) **Camps generals definits en HTTP/1.0**:

a) **`Date:` *`data`*.** Indica la data i l'hora en què es va originar el missatge. Els valors dels camps HTTP que representen dates s'han d'expressar d'una d'aquestes tres maneres, sempre referides al temps universal (TU, anteriorment conegut com a **hora del meridià de Greenwich** o GMT):

```
Fri, 06 Dec 2002 09:22:15 GMT (RFC 1123, forma recomanada)
Friday, 06-Dec-2002 09:22:15 GMT (RFC 850)
Fri Dec 6 09:22:15 2002 (format ANSI C)
```

b) **`Pragma:` *`1#directriu`*.** Proporciona una llista de directrius als sistemes que intervenen en la transferència (clients o servidors). La semàntica de les directrius depèn de les implementacions; tanmateix, n'hi ha una de definida pel protocol, que es representa amb la paraula `no-cache`. Si un missatge de petició conté aquesta directriu, significa que s'ha d'enviar la petició al servidor corresponent encara que hi hagi una còpia de la resposta desada en la memòria cau.

ANOTACIONS

Nota

L'especificació HTTP/1.1 afegeix cinc camps generals nous:

- `Cache-Control`: directrius sobre la política de memòria cau.
- `Connection`: opcions de connexió. L'opció **close** indica que l'emissor tancarà la connexió després que s'hagi enviat la resposta.
- `Transfer-Encoding`: codificació aplicada al cos del missatge.
- `Upgrade`: versions del protocol suportades.
- `Via`: intermediaris pels quals ha passat el missatge.

2) **Camps referents a l'entitat definits en l'HTTP/1.0**:

a) **`Content-Length:`** ***`1*dígit`*****`.`** Indica la longitud en bytes del cos de l'entitat. Si aquest camp no és present en una resposta, el client només pot saber quin és el final del cos quan el servidor tanca la connexió. En un missatge de petició que inclogui un cos, el camp `Content-Length` és obligatori, ja que de no ser-hi el servidor no podria enviar la resposta i no es podria tancar la connexió.

Nota

Consulteu els camps de capçaleres nous en el subapartat 19.5.1.

b) **`Content-Type:`** ***`tipus`*****`.`** Indica el tipus de contingut del cos de l'entitat (per exemple, `text/html` si és un document HTML). El valor d'aquest camp es representa segons l'especificació `MIME` i, per tant, pot incloure paràmetres (per exemple, `charset=ISO-8859-1` si els caràcters estan codificats segons l'alfabet llatí número 1). Tots els missatges HTTP amb cos haurien d'incloure el camp `Content-Type`.

c) **`Content-Encoding:`** ***`codificació`*****`.`** Indica si s'ha aplicat alguna transformació al contingut de l'entitat (per exemple: `x-gzip`). El valor d'aquest camp es representa segons l'especificació `MIME`.

d) **`Last-Modified:`** ***`data`*****`.`** Indica la data i l'hora en què el recurs contingut en el cos es va modificar per última vegada.

ANOTACIONS

e) **`Expires:` *`data`*.** Indica la data i l'hora a partir de la qual el contingut del cos es pot considerar obsolet o caducat per al seu emmagatzematge en la memòria cau. La presència d'aquest camp significa que, possiblement, el recurs es modificarà en la data indicada o deixarà d'existir; tanmateix, no implica que els canvis, si es produeixen, s'hagin d'efectuar necessàriament en aquesta data. Ara bé, si la data de caducitat és igual o anterior a l'especificada en el camp Date, l'entitat no s'ha d'emmagatzemar en la memòria cau.

f) **`Allow:` *`1#mètode`*.** Indica els mètodes HTTP que es poden aplicar al recurs sol·licitat.

Nota

L'especificació HTTP/1.1 afegeix sis camps d'entitat addicionals nous:

- `Content-Base`: adreça base per a interpretar URL relatius.
- `Content-Language`: llenguatge que s'ha utilitzat.
- `Content-Location`: URI de l'entitat, en cas que el recurs corresponent en disposi de més d'una.
- `Content-MD5`: seqüència de bits per a comprovar la integritat del contingut.
- `Content-Range`: per si una entitat s'envia en diferents fragments.
- `ETag`: etiqueta associada a l'entitat, per si el recurs en disposa de més d'una.

3. Camps propis de les peticions HTTP/1.0:

a) **`From:` *`adreça`***. Aquest camp conté l'adreça electrònica de l'usuari que sol·licita el recurs.

b) **`User-Agent:` *`1*implementació`*.** Aquest camp permet identificar la implementació del client. S'expressa com una llista de "productes" (per exemple, tipus de navegador, llibreries de suport que utilitza, etc.) amb números de versió opcionals (separats dels noms amb "`/`"), que poden incloure comentaris entre parèntesis.

ANOTACIONS

Exemple

Exemples de llistes de productes:

- `Lynx/2.8rel.2 libwww-FM/2.14.`
- `NCSA_Mosaic/ 2.6 (X11;SunOS 4.1.4 sun4m) libwww/2.12.`
- `Harvest/1.5.17.`

El servidor pot utilitzar aquesta informació per a generar estadístiques, detectar les implementacions que produeixen certs errors, adaptar les respostes al tipus de client, etc.

c) **`Referer: URI.`** Si l'usuari selecciona un enllaç d'un document HTML o d'un altre recurs que té adreça pròpia, el client la pot incloure en el camp `Referer` del missatge de petició. El servidor pot utilitzar aquesta informació per a generar estadístiques, detectar des de quin recurs es referencia una adreça incorrecta o obsoleta, etc.

d) **`If-Modified-Since: data.`** Quan el client ja disposa d'una còpia del recurs sol·licitat en la memòria cau, pot utilitzar aquest camp per a dur a terme una operació GET condicional: si el recurs s'ha modificat posteriorment a la data indicada, s'efectuarà l'operació de manera normal i, si no, el servidor enviarà una resposta sense cos i amb el codi 304. Aquest camp només és aplicable al mètode GET.

Nota

Consulteu la codificació en Base64 en l'apartat 19.5.1.

e) **`Authorization: esquema#paràmetre.`** Amb aquest camp, un usuari pot presentar les seves credencials a un servidor (per norma general, un nom i una contrasenya) perquè li permeti accedir a recursos d'accés restringit. La primera part de la credencial indica l'esquema d'autenticació que s'ha d'utilitzar. L'HTTP/1.0 només defineix un esquema denominat *bàsic*, però se'n pot utilitzar qualsevol mentre sigui conegut pel client i el servidor. Si s'utilitza l'esquema bàsic, el valor d'aquest camp ha de ser la paraula `Basic` seguida de la cadena de caràcters que resulta de codificar en `Base64` el nom d'usuari i la seva contrasenya separats per "`:`".

Exemple

Si el nom d'usuari és `administrador de web` i la seva contrasenya és `Secret`, es codificarà en `Base64` la cadena `webmaster:Secret`. El camp quedaria d'aquesta manera:

```
Authorization:Basic d2VibWFzdGVyOlNlY3JldA==
```

Òbviament, la descodificació d'aquest camp és trivial i, per tant, no hi ha protecció contra tercers que tinguin la possibilitat d'inspeccionar els missatges i vulguin obtenir

ANOTACIONS

un accés no autoritzat. Per a aconseguir que el mètode d'autenticació sigui més segur, es pot utilitzar un esquema més sofisticat o alguna variant del protocol basada en el xifratge de les dades, per exemple l'HTTPS.

La resposta a una petició que inclogui el camp `Authorization` no s'hauria de desar en la memòria cau.

Nota

En l'especificació HTTP/1.1 s'afegeixen els camps de petició següents:

- `Accept`: tipus de contingut que accepta el client.
- `Accept-Charset`.
- `Accept-Encoding`.
- `Accept-Language`.
- `Host`: nom del servidor a qui va dirigida la petició, per si en té més d'un; aquest camp, obligatori en les peticions HTTP/1.1, permet que un ordinador amb diferents noms actuï com si fossin diferents servidors alhora.
- `If-Match`: permet comparar entitats per les seves etiquetes.
- `If-None-Match`.
- `If-Range`.
- `If-Unmodified-Since`.
- `Max-Forwards`.
- `Proxy-Authorization`.
- `Range`: serveix per a sol·licitar un fragment d'una entitat.

ANOTACIONS

4. Camps propis de les respostes HTTP/1.0:

a) **`Server: 1*implementació.`** Aquest camp és similar a User-Agent, però referit al servidor.

b) **`Location: URI.`** Indica que el recurs sol·licitat s'ha d'obtenir en una altra adreça. Aquest serà el cas quan el codi de resposta sigui 301 o 302. Per norma general, quan el client rebi una resposta que inclogui aquest camp, generarà de manera automàtica una nova petició amb l'adreça indicada.

c) **`WWW-Authenticate: 1#(esquema realm = "regne" *(, paràmetre )).`** Aquest camp és obligatori quan el codi de les respostes és 401. Indica que és necessari presentar una credencial per a accedir al recurs sol·licitat, o que la que s'ha presentat no és vàlida.

Un servidor pot agrupar els seus recursos d'accés restringit en regnes, cada un amb el seu esquema d'autenticació i conjunt d'usuaris autoritzats (amb una mateixa credencial s'hauria de poder accedir a tots els recursos d'un regne). El valor d'aquest camp és una llista amb els noms dels regnes aplicables al recurs sol·licitat, juntament amb els seus esquemes i paràmetres opcionals per a poder dur a terme l'autenticació amb el camp `Authorization` de la petició.

Nota

En l'especificació HTTP/1.1 s'afegeixen els camps de resposta següents:

- `Age`.
- `Proxy-Authenticate`.
- `Public`: llista de mètodes suportats pel servidor.
- `Retry-After`.
- `Vary`: llista de camps de la petició que s'han utilitzat per a seleccionar una entitat, quan el recurs en disposa de més d'una.
- `Warning`: informació addicional sobre l'estatus de la resposta.

ANOTACIONS

21.3.3. Mètodes del servei HTTP

El protocol HTTP/1.0 defineix els tres mètodes següents:

1. **Mètode `GET`.** Aquest mètode serveix per a obtenir l'entitat corresponent a l'URI especificat en la línia de petició. Per norma general, el servidor traduirà el camí de l'URI a un nom de fitxer o de programa:

- En el primer cas, el cos de l'entitat serà el contingut del fitxer.
- En el segon cas, el servidor executarà el programa i l'entitat serà el resultat que generi.

  Els components `paràmetre` i/o `consulta` de l'URI es poden utilitzar com a arguments del programa.

2. **Mètode `HEAD`.** Aquest mètode és igual que el GET, excepte que en la resposta el cos serà buit i, per tant, només tindrà capçalera (que haurà de ser idèntica a la que s'hauria enviat si el mètode fos el GET). Per norma general, s'utilitza el mètode HEAD, per exemple, per a comprovar si un URL és vàlid o per a obtenir informació sobre un recurs sense necessitat de transferir-ne el contingut.

3. **Mètode `POST`.** Aquest mètode serveix per a enviar una entitat que el servidor ha d'incorporar en el recurs identificat per l'URI de la línia de petició. La semàntica d'aquest mètode depèn del tipus de recurs. Per exemple, es pot utilitzar per a afegir contingut a un recurs existent, per a enviar un missatge a un tauler d'anuncis o a un grup de notícies, per a crear un registre nou en una base de dades, per a passar dades a un programa que s'ha d'executar al servidor, etc. Un cas típic d'aquest últim exemple són les dades d'un formulari HTML.

   El codi de resposta a una operació POST per norma general serà 201, si com a resultat s'ha creat un recurs nou (en aquest cas, el cos de la resposta hauria de contenir una referència a aquest recurs), o bé 200 o 204 si no se n'ha creat cap (el cos d'una resposta 200 contindrà una descripció del resultat obtingut, i el d'una resposta 204 simplement serà buit).

   Una propietat del mètode POST és que, si s'envien dues peticions iguals, el resultat de la segona no ha de ser necessàriament el

Nota

Quan el camí d'un URI identifica un programa, la manera com se li passen els valors dels paràmetres o les consultes és un assumpte local del servidor. Per exemple, un mecanisme utilitzat habitualment és el denominat *CGI* (interfície comuna de passarel·la o *common gateway interface*).

ANOTACIONS

mateix que el de la primera. Per tant, la resposta a una operació POST no s'hauria de desar en la memòria cau.

Nota

En l'HTTP/1.1 s'han afegit una sèrie de mètodes nous, entre els quals hi ha els següents:

- PUT: per a crear un recurs amb l'URI especificat en la petició.
- DELETE: per a esborrar un recurs.
- OPTIONS: per a obtenir informació sobre les opcions de transferència.
- TRACE: per a obtenir una còpia del missatge com ha arribat a la seva destinació final.

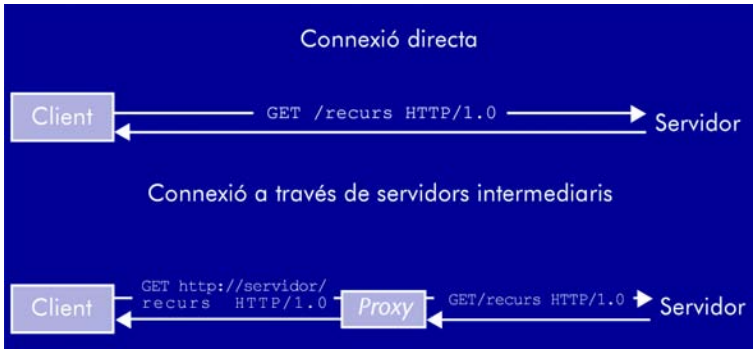
Activitat

Establiu una connexió amb un servidor HTTP amb `telnet` *`servidor`* `80` i esbrineu la data en què ha estat modificada per última vegada la pàgina principal, o alguna altra informació referida al servidor (amb un missatge format per la línia `HEAD / HTTP/1.0`, una capçalera buida, una línia en blanc i un cos buit).

21.3.4. Intermediaris: *proxies* i passarel·les

El model general de l'HTTP no solament considera la connexió directa d'un client amb el servidor, sinó també la connexió per mitjà de servidors intermediaris, com és el cas dels *proxies*:

Figura 88.



En la configuració amb *proxy*, el client estableix la connexió amb el servidor *proxy* que, al seu torn, n'estableix una altra, com a client, amb el servidor final. Quan el *proxy* rep el missatge de petició del client, pot generar una resposta pròpia o retransmetre-la al servidor final. En el segon cas, el *proxy* pot introduir modificacions en la petició, segons l'aplicació per a la qual estigui dissenyat, i, quan rebi la resposta del servidor final, la retransmetrà al client, també amb la possibilitat d'efectuar canvis.

Així mateix, és possible que hi hagi més d'un *proxy* a la cadena; és a dir, que el primer *proxy* no es connecti directament al servidor final, sinó per mitjà d'un altre *proxy*, i així successivament.

En les peticions que un client (o un altre *proxy*) envia a un *proxy*, hi ha una variació respecte al cas de la connexió directa de client a servidor: l'URI de la línia de petició no ha de ser un URL HTTP relatiu, sinó que ha de ser un URI absolut. Altrament, el *proxy* no sabria quin és el servidor final al qual va destinada la petició.

L'ús d'un servidor *proxy* té aplicacions diferents. Les principals són les següents:

a) Actuar com a tallafoc que aïlli una xarxa local de la resta de les xarxes. En aquesta configuració, els clients no tenen accés directe a l'exterior de la seva xarxa i tota comunicació amb els servidors remots té lloc per mitjà del *proxy*. Això permet minimitzar el risc que usuaris externs comprometin la seguretat dels sistemes locals (accessos no autoritzats, sabotatges, etc.).

b) Tenir una memòria cau compartida entre els usuaris de la xarxa local. Si diferents clients sol·liciten directament un mateix recurs, per norma general desaran la mateixa còpia de la resposta en les seves memòries cau respectives. Si el sol·liciten per mitjà d'un *proxy*, la primera petició necessitarà un accés al servidor remot i les següents potser podran aprofitar la còpia ja desada en la memòria cau, encara que provinguin de clients diferents.

c) Construir una jerarquia de memòries cau de *proxies*: en el nivell més baix es troben els *proxies* als quals accedeixen directament els clients, en un segon nivell hi ha els *proxies* als quals accedeixen els del primer nivell, etc. Fins i tot hi pot haver *proxies* a

ANOTACIONS

escala de tot un país. Hi ha un protocol denominat **ICP** (*Internet cache protocol*) que permet coordinar els diferents servidors *proxy* d'una jerarquia.

Nota

A més dels *proxies*, hi ha un altre tipus de servidor intermediari anomenat *passarel·la*. Les passarel·les actuen com si fossin el servidor final, amb la diferència respecte al *proxy* que el client no sap que no es connecta directament al servidor final. Una passarel·la es pot utilitzar com a tallafoc a la xarxa del servidor o per a traduir les peticions HTTP de manera que es pugui accedir a recursos emmagatzemats en sistemes no HTTP.

ANOTACIONS

22. Missatgeria instantània

No hi ha dubte que Internet ha canviat la manera de comunicar-nos. Per a molts de nosaltres, el correu electrònic ha reemplaçat gairebé per complet el correu tradicional i ha fet disminuir considerablement les trucades que fem. Diàriament s'envien milions de correus electrònics.

Malgrat la rapidesa del sistema de correu, encara no és prou ràpid. En alguns servidors, es poden acumular retards d'alguns minuts. A més, per ràpid que sigui, es tracta d'un sistema asíncron, que no permet mantenir una conversa fluida. Cada missatge enviat o rebut implica una sèrie de passos que s'han de seguir, botons que s'han de polsar i finestres que s'han d'obrir. Per tot això, s'han popularitzat els sistemes de missatgeria instantània, un clàssic del model igual a igual.

Les aplicacions de missatgeria instantània permeten mantenir una llista de persones (les llistes de contactes o *buddy lists*) amb qui un es vol mantenir en contacte mitjançant missatges, sempre que estiguin connectats. Aquest detall no ens ha de preocupar, perquè és l'aplicació mateixa la que s'encarrega de comprovar-ho.

La manera d'enviar missatges és molt simple. L'aplicació ens presenta una finestra en què teclegem el contingut del missatge i en què es veurà el que ens contesti l'interlocutor. D'aquesta manera, es pot mantenir una conversa en temps real.

A més de la conversa bàsica, la majoria de programes de missatgeria instantània ofereixen altres prestacions, com les següents:

- Xat que permet establir converses en grup.
- Pissarra que permet veure dibuixos realitzats per l'interlocutor, en el moment o gravats en arxius.
- Arxius que permeten compartir arxius.

Nota

Alguns sistemes permeten l'ús de càmeres de vídeo, micròfons i altaveus perquè la conversa no se circumscrigui només a text escrit.

ANOTACIONS

Des que el 1983 va aparèixer el primer sistema de missatgeria instantània, han sortit al mercat molts altres productes, alguns usant protocols propietaris, d'altres intentant buscar un estàndard. De fet, l'IETF ha recopilat moltes d'aquestes contribucions i desenvolupa l'*instant messaging and presence protocol*, pel qual s'espera que passi el futur de la missatgeria instantània.

Nota

En general, l'honor d'haver creat el primer sistema de missatgeria instantània s'atribueix a Mirabilis, la companyia formada per quatre estudiants israelians que el 1996 va llançar ICQ. Però no hem d'oblidar que en els sistemes Unix hi ha des de 1983 una aplicació anomenada `talk` que permet la conversa entre dos usuaris connectats a sengles sistemes Unix via `telnet` o `rlogin`

Segurament, la proliferació dels ordinadors domèstics, Internet i la web de la darreria dels noranta, majoritàriament en entorn Windows, ha eclipsat una mica tot el que tenia a veure amb l'entorn Unix.

22.1. Programes de missatgeria instantània

Sense pretendre crear una llista exhaustiva, repassarem els programes principals que proporcionen la funcionalitat de la missatgeria instantània en un entorn igual a igual. Alguns dels programes presentats són de programari lliure, però d'altres no.

22.1.1. ICQ

ICQ va ser el primer programa que va aparèixer (el 1996) i encara és dels més populars. Les sigles responen a la frase "I seek you" ('t'estic buscant'). ICQ usa un control centralitzat per mitjà de servidors, i la comunicació entre clients i servidors, i entre clients es realitza mitjançant un protocol propietari (ICQ, v.5).

Nota

L'adreça del web d'ICQ és la següent:
http://www.icq.com.

ANOTACIONS

22.1.2. AIM

AIM (AOL *instant messenger*), que va aparèixer poc després d'ICQ, és l'evolució de les comunitats que America OnLine ja tenia desenvolupades amb altres tecnologies més antigues, com les BBS (*bulletin board system*). AIM també usa un protocol propietari. Per tant, el client ha de ser específic per a aquest entorn.

Nota

El 1998, AOL va comprar Mirabilis i va integrar ICQ en el seu paquet ofimàtic o *suite* de serveis oferts.

Nota

L'adreça del web d'AIM és la següent:
http://www.aim.com.

22.1.3. MSN Messenger

Microsoft també ha entrat al món dels sistemes de missatgeria instantània amb MSN (Microsoft Network) Messenger. És molt semblant als dos descrits anteriorment.

Nota

L'adreça del web d'MSN Messenger és la següent:
http://messenger.msn.com.

22.1.4. Jabber

Jabber és més que un mer sistema de missatgeria. És un conjunt de protocols i tecnologies que permeten el desenvolupament de múltiples aplicacions igual a igual. Permet desenvolupar clients, servidors, components, biblioteques, etc. Es basa en XML i és la base dels protocols que l'IETF està en procés d'estandarditzar.

Nota

L'adreça del web de Jabber és la següent:
http://www.jabber.org.

Els seus autors presenten entre els avantatges de Jabber els següents: permet entorns descentralitzats, és obert, públic, segur, extensible i flexible.

22.1.5. GAIM

Gaim és un client de missatgeria instantània multiprotocol, per a GNU/Linux, Unix BSD, MacOS X i Windows. És compatible amb AIM, ICQ, MSN Messenger, Yahoo, IRC, Jabber, Gadu-Gadu i Zephyr.

Nota

L'adreça del web de GAIM és la següent:
http://gaim.sourceforge.net.

Una de les seves millors prestacions és que permet estar connectat amb diferents xarxes de missatgeria simultàniament. Així, un usuari de Gaim pot estar en un xat d'AOL mentre conversa amb un amic de Yahoo i dins d'un canal IRC, tot en la mateixa aplicació.

ANOTACIONS

Resum

En aquest curs hem presentat les bases de funcionament de la xarxa Internet.

En la primera part s'han presentat els conceptes bàsics de xarxes de computadors i els diferents sistemes de comunicacions que s'utilitzen avui dia, seguint un fil conductor històric, que permet entendre el perquè de moltes de les limitacions i particularitats que presenten. S'ha exposat, també, l'arquitectura de protocols, un concepte bàsic en xarxes de computadors. Com a paradigma d'aquest concepte, s'ha explicat el model de referència OSI i s'han descrit els set nivells que el formen. Les arquitectures reals, en particular Internet, se solen descriure i explicar-se en relació amb el model OSI.

En la segona part hem fet un repàs, des d'un punt de vista descriptiu, de les xarxes d'àrea local. El principi bàsic de funcionament de les xarxes d'àrea local és la difusió de trames en un medi compartit. Així, les estacions de la xarxa, quan vulguin establir una comunicació amb una altra estació (o d'altres), hauran de generar trames de bytes a partir de les dades que vulguin transferir. I posar-les al medi perquè arribin al seu o als seus destinataris.

S'han presentat les topologies més usuals, destacant les topologies en bus i en anell. En un bus, totes les estacions es connecten a un mateix cable, mentre que en un anell, cada estació està connectada a dos més, creant una estructura circular. Totes dues topologies tenen avantatges i inconvenients, el més important dels quals és la dificultat de manteniment i localització d'avaries que presenten totes dues. Això ha afavorit el desenvolupament del cablatge estructurat, com a mecanisme per a instal·lar xarxes d'àrea local més fiables i seguint estàndards universals.

Associats al cablatge estructurat han aparegut dispositius molt comuns avui en dia com són els concentradors i els commutadors. Els primers simulen un bus i els segons aprofiten el concepte de commu-

ANOTACIONS

tació per a aconseguir xarxes més eficients. Sempre que es pot, les trames no es difonen a totes les estacions presents a la xarxa, sinó solament a les seves destinatàries.

Finalment s'han presentat les polítiques d'accés al medi més habituals, les quals van associades a la topologia usada. Així, hem vist la política de pas de testimoni com la més adequada per a les xarxes en anell, i la CSMA/CD, que s'usa en la majoria de les xarxes en bus, tant si és un bus cablat com per mitjà d'un concentrador.

En la tercera part, s'han descrit els protocols que s'utilitzen en la xarxa Internet. En concret, s'han vist els protocols del nivell d'interconnexió de xarxa, que són IP i els seus associats ARP i ICMP, i els protocols del nivell de transport: TCP i UDP.

S'ha vist el fenomen de l'encapsulació de la informació, com a resultat de tenir diferents protocols involucrats en una mateixa connexió, i com aquesta encapsulació afecta el nivell de xarxa.

Un dels aspectes més rellevants del protocol IP és l'assignació d'adreces. Hem vist com cada interfície connectada a Internet ha de tenir una adreça IP única que la identifiqui. Les adreces IP no solament identifiquen estacions, sinó també la xarxa o subxarxa on hi ha l'estació. D'aquesta manera, és possible encaminar els paquets IP per mitjà de diferents encaminadors i, per tant, per mitjà de diferents xarxes.

El protocol ICMP ens permet resoldre incidències que puguin ocórrer a la xarxa. Hem descrit les peculiaritats d'aquest protocol, i també dues utilitats com són el `ping` i el `traceroute`, d'ús molt habitual.

Hem descrit les xarxes d'accés a Internet més habituals, com són la xarxa Ethernet, l'accés mitjançant una connexió telefònica amb el protocol PPP i l'accés mitjançant el protocol ADSL, que permet usar el cable telefònic però sense haver d'establir una trucada.

Pel que respecta al nivell de transport, hem vist com el seu principal objectiu és lliurar la informació als nivells orientats a l'aplicació en

ANOTACIONS

els extrems de la xarxa. En la jerarquia de protocols TCP/IP es defineixen dos protocols de transport:

1. L'UDP, que és un protocol no orientat a la connexió. Per tant, no efectua cap control d'errors ni de flux. Si un datagrama UDP arriba equivocat (l'UDP utilitza un codi detector d'errors), l'UDP el descarta i no el lliura a l'aplicació. Aquesta última haurà de ser capaç de respondre a aquest tipus de servei o haurà d'assumir la pèrdua de la informació. Aquest tipus de servei pot ser útil en aplicacions en temps real, en les quals és més important que la informació arribi quan li toqui; és a dir, amb un retard limitat, que no se'n perdi una part.

2. El TCP, que és un protocol orientat a la connexió. Hi haurà una fase d'establiment de la connexió (l'anomenat procediment *three-way handshake*), una fase de transmissió de la informació i una fase d'acabament de la connexió. El TCP lliurarà la informació a l'aplicació totalment lliure d'errors. Per a aconseguir-ho, necessita efectuar un control d'errors i de flux. El TCP utilitza un codi detector d'errors juntament amb un protocol de retransmissions per a recuperar la informació errònia. Com que les memòries intermèdies de recepció es poden desbordar, el TCP utilitza un control de flux per finestra lliscant. El TCP ha de dimensionar correctament els temporitzadors de retransmissió.

En la quarta i última part, s'ha presentat el concepte de programació distribuïda i el model client/servidor, que és el més utilitzat. També s'ha presentat el model igual a igual, que no és una alternativa a l'anterior com de vegades s'ha plantejat, perquè no és un model de programació d'aplicacions, sinó que fa referència al tipus de màquines que s'interconnecten.

També s'ha descrit el servei de noms de domini, que proporciona accés a una base de dades distribuïda per tota la xarxa que permet obtenir informació sobre un nom determinat. Les consultes més habituals són per a esbrinar l'adreça IP que correspon a un nom de servidor, però també és possible obtenir el nom del servidor de correu d'un domini donat, el nom d'una màquina a partir de la seva adreça IP, etc.

ANOTACIONS

Pel que respecta a les pròpies aplicacions, s'han presentat els anomenats *serveis bàsics d'Internet*, que són els següents:

- Telnet: protocol general per a establir sessions interactives amb un servidor que permet que el client actuï com un terminal virtual.
- Rlogin: protocol de terminal virtual amb facilitats per a simplificar el procés d'identificació de l'usuari.
- Rsh, rexec: protocols per a l'execució remota de processos al servidor.

S'han presentat també els protocols més usats potser avui en dia, com són la transferència d'arxius, el correu electrònic, el servei de missatges (*news*), el WWW i la missatgeria electrònica, amb les seves múltiples possibilitats.

Sobre l'FTP, s'ha comentat que permet copiar arxius del client al servidor, del servidor al client i també d'un servidor a un altre, i que proporciona operacions perquè el client pugui manipular el sistema d'arxius del servidor.

Sobre el correu electrònic, que és un servei basat en els mateixos conceptes del correu postal, s'ha comentat en concret la filosofia d'emmagatzematge i reenviament.

La norma RFC 822 especifica el format dels missatges, i la norma MIME especifica un conjunt d'extensions que permeten, per exemple, adjuntar arxius i usar diversos idiomes.

Tres protocols proporcionen les diferents funcionalitats necessàries:

- SMTP: transferència de missatges.
- POP3: accés simple a bústies de correu.
- IMAP4rev1: accés complex a bústies de correu.

El servei de notícies (*news*) facilita la publicació de missatges amb un abast que pot anar des de l'àmbit local d'una organització, fins a l'àmbit mundial (tot Internet). Es basa en la propagació dels missatges entre els diferents servidors per mitjà de l'NNTP. Aquest mateix

ANOTACIONS

protocol el poden utilitzar els clients per a accedir als missatges desats en un servidor o per a enviar-ne d'altres de nous.

El servei WWW permet "navegar" per un conjunt d'elements d'informació interconnectats amb referències o enllaços entre si que poden contenir dades amb diferents tipus de representacions (text, imatges, àudio, vídeo). El protocol que proporciona l'accés a aquesta informació és l'HTTP.

El servei de missatgeria instantània està suportat per diferents protocols, alguns propietaris. Hi ha diferents plataformes que el proporcionen i també s'han desenvolupat diferents clients multiprotocol que permeten accedir a diferents xarxes de missatgeria simultàniament.

ANOTACIONS

Bibliografia

Comer, D.E. (2000). "Principles, Protocols and Architecture". *Internetworking with TCP/IP* (vol. 1, 4a. ed.). Upper Saddle River: Prentice Hall.

Pierce, J.R.; Noll, A.M. (1995). *Señales. La ciencia de las telecomunicaciones*. Barcelona: Reverté.

Rose, Marshall T. (1990). *The open book*. Englewood Cliffs: Prentice Hall.

Stallings, W. (2000). *Comunicaciones y redes de computadores* (6a. ed.). Madrid: Prentice-Hall.

Stevens, W.R. (1994). "The Protocols" *TCP/IP Illustrated* (vol. I). Wilmington: Addison-Wesley.

Tanenbaum, A.S. (2003). *Redes de computadoras* (4a. ed.). Mèxic: Pearson Educación.

ANOTACIONS

Annexos

Annex 1

L'algorisme de la suma de verificació (*checksum*)

Els protocols IP i TCP utilitzen, entre d'altres, un senzill *checksum* per a detectar errors. Aquesta és una versió de l'algorisme de la suma de verificació:

```
u_short checksum(u_short * addr,int len)
{
   int aux_len = len;
   u_short *aux_addr = addr;
   u_short res;
   int sum = 0;

   while(aux_len > 1)
   {
   sum+ = *aux_addr++;
   aux_len- = 2;
   }

   if(aux_len == 1)
   sum+ = *(u_char) *aux_addr;

   sum = (sum>>16) + (sum & 0xffff);
   sum+ = (sum>>16);
   res = ~sum;

   return res;
}
```

ANOTACIONS

Annex 2

Aplicacions esmentades en el text

1. Aplicacions estàndard

Nota

En els sistemes Unix podreu trobar més informació utilitzant la comanda man. Per exemple, per a saber més detalls de l'ordre route de Unix podeu executar:

```
$ man route
```

Nota

Per a executar les comandes que es mostren en aquest annex es requereix tenir privilegis de superusuari.

A continuació, resumim les aplicacions estàndard que s'han vist en el curs i que estan disponibles en entorns Unix i GNU/Linux.

- **`ping`:** per mitjà de paquets ICMP (del tipus petició d'eco i resposta d'eco) permet saber si una màquina funciona o no i tenir una idea de quin és el retard d'anada i tornada (*round-trip*). Així mateix, permet saber per quines màquines passa el paquet fins que arriba a destinació. Tot i que per a aquesta funció, va millor el programa `traceroute`, a causa de les limitacions inherents en els paquets ICMP.
- **`traceroute`:** permet esbrinar la ruta que se segueix des de l'equip local fins a una destinació qualsevol d'Internet. Marca els retards existents a cada un dels nodes que cal creuar. Es basa en el camp TTL del paquet IP i els missatges ICMP-*time-to-live-exceeded*. Aquesta aplicació no està disponible en algunes distribucions de Unix i distribucions de GNU/Linux, però es pot aconseguir fàcilment.
- **`arp`:** permet consultar i modificar la taula ARP (memòria ARP) d'una màquina.
- **`route`:** serveix per a consultar i modificar la taula d'encaminament IP d'una màquina connectada a Internet.
- **`ifconfig`:** permet consultar l'estat de les targetes de xarxa disponibles en el sistema local.
- **`netstat`:** proporciona estadístiques dels protocols TCP i UDP. Permet fer una llista dels ports que s'escolten. Mostra l'estat en què es troben els *sockets* TCP. Si voleu una llista de tots els *sockets* actius i ports oberts, feu `netstat -a`, i, si us interessen altres variants, consulteu l'ajuda.
- **`telnet`:** ofereix, a part de la principal emulació de terminal, connectar i experimentar els protocols ASCII.

ANOTACIONS

2. Aplicacions no estàndard

- **`netcat`** (llicència GPL-Llenguatge C): aquesta aplicació ens permet utilitzar connexions TCP i UDP des de la línia de comandes (per exemple, transmetre un fitxer) o comprovar quins ports té oberts una màquina determinada, entre altres serveis.

- **`tcpdump`** [Unix (programari lliure-llenguatge C)]: permet analitzar el trànsit de la xarxa per mitjà de connexió (LAN o punt a punt). Al contrari del que el seu nom indica, és capaç d'interpretar els paquets no solament en l'àmbit TCP, sinó també en l'IP, xarxa, i aplicació (per a aplicacions comunes). És una eina imprescindible per a qualsevol administrador de sistemes, encara que no aparegui en les distribucions de les diferents variants de Unix. El codi és de lliure distribució. Els seus autors són Van Jacobson, Craig Leres i Steven McCanne, de la Universitat de Califòrnia, encara que el programa en què es basava l'original, l'Etherfind, era propietat de Sun Microsystems i es distribuïa dins de SunOS.

> ***Nota***
>
> En LAN, el `tcpdump` posa la targeta de xarxa en mode promiscu. Això significa que tot el trànsit de la xarxa és visible, amb la qual cosa qualsevol usuari malintencionat (un *hacker*) en pot fer un mal ús. Per tant, el tcpdump i altres eines similars o derivades del `tcpdump`, coneguts com a detectors (*sniffers*), es poden considerar com una eina potencialment perillosa. De fet, el que és arriscat és no utilitzar almenys un dels tres mecanismes de protecció més importants contra els detectors: xifratge, xifratge i xifratge.

A continuació, s'indiquen els resultats que pot mostrar el programa tcpdump.

3. El programa `tcpdump`

El programa `tcpdump` permet capturar i filtrar paquets que creuen una interfície de xarxa que s'ha activat en *mode promiscu* (tots els paquets que passen pel medi són capturats i són filtrats). Els

ANOTACIONS

paquets són processats per un programari especial que només pot executar el superusuari (*root*) de la màquina.

Per a veure com funcionen els diferents protocols, proporcionem exemples de les traces que mostra el `tcpdump` durant l'establiment o l'acabament de la connexió.

Per a veure com funciona aquest programa, es pot utilitzar la instrucció `man` de Unix (o GNU/Linux). En aquest annex oferim un exemple del significat de la traça del `tcpdump`.

El format general de sortida del `tcpdump` és el següent:

```
Data  src> dst: flag data-sqno ack window urgent options
```

El significat dels components del format són els següents:

- **`data`**: aquest component ens proporciona l'hora en què es va produir l'esdeveniment en format hora:minuts:segons.microsegons (o mil·lisegons, segons la resolució del rellotge).
- **`src`** i **`dst`**: són les adreces IP i els ports TCP/UDP de les connexions de font i de destinació.
- **`flags`**: són una combinació dels possibles indicadors d'un segment/datagrama TCP/UDP: S (SYN), F (FINAL), P (PUSH), R (RST) i '.' (que significa que no hi ha indicadors).
- **`data-sqno`**: descriu el número de seqüència de la porció de dades.
- **`ack`**: és el número de seqüència del byte següent que espera rebre l'altre extrem TCP/UDP.
- **`window`**: és la mida de la finestra que el receptor anuncia al transmissor.
- **`urgent`**: indica que hi ha dades urgents en aquest segment/datagrama.
- **`options`**: són les opcions TCP que solen anar entre claudàtors del tipus < >; per exemple, la mida màxima del segment (per exemple, <mss 1.024>).

ANOTACIONS

Suposem que fem un `rlogin` des de la màquina *argos* (font) fins a la màquina *helios* (destinació). Un exemple d'una línia que retorna l'ordre `tcpdump` seria el següent:

- `12:34:23.165439 argos.1023 > helios.login: S 7838437: 7838437 (0) win 4096 <mss 1024>`
- `12:34:23.165439 helios.login > argos.1023: S 453534: 453534 (0) ack7838438 win 4096 <mss 1024>`

La primera línia indica el següent:

a) L'esdeveniment té lloc a les 12:34:23,165439.

b) El seu origen és la màquina *argos* amb el port 1.024. La seva destinació és la màquina *helios* amb el port `login`.

c) L'indicador S assenyala que és un segment de SYN (per exemple, per un inici de connexió).

d) Anuncia un número de seqüència 7.838.437 i consumeix fins a aquest número. Convé tenir en compte que el `tcpdump` escriu el número inicial i el final del número de seqüència de dades, i, entre parèntesis, la diferència indicant la longitud del camp de dades (en aquest cas 0 bytes, ja que és un segment de petició de connexió). Per tant, indica entre parèntesis la longitud del segment de dades. Aquest número de seqüència és absolut. Les sortides següents del `tcpdump` indicaran els números de seqüència relatius a l'inicial. Per exemple, en lloc d'indicar 7.838.437:7.838.450, (13) en notació absoluta, indicarà 1:13, (13) en notació relativa a l'ISN (es compleix que valor absolut és igual que l'ISN més el valor relatiu). El mateix s'aplica als números de seqüència de l'ACK.

e) Anuncia una finestra de 4.096 bytes.

f) Hi ha una petició de mida màxima de segment de 1.024 bytes.

La segona línia indica el següent:

a) L'origen és la màquina *helios* amb el port `login`. La destinació és la màquina *argos* amb el port 1.023.

ANOTACIONS

b) L'indicador *S* assenyala que és un segment de SYN (per exemple, per a un inici de connexió).

c) Indica el número de seqüència inicial 453.534.

d) Reconeix amb un ACK el byte següent que espera del transmissor, el 7.838.438 (és a dir, 7.838.437 + 1), ja que ha rebut correctament fins al 7.838.437).

e) Anuncia una finestra de 4.096 bytes.

f) Hi ha una petició de mida màxima de segment de 1.024 bytes.

El programa `tcpdump` permet escoltar, de manera senzilla, tot el que succeeix a la xarxa. Admet tota una sèrie d'indicadors per a filtrar només les adreces IP de font o de destinació que poden interessar, o el tipus de protocol que es vol escoltar (TCP, UDP, ARP, etc.). Així mateix, admet indicadors per a obtenir el camp de dades, per a filtrar un nombre fix de segments, etc. Per a veure totes les possibles opcions que permet el programa `tcpdump` s'utilitza la comanda `man`.

ANOTACIONS

Annex 3

Notació

En aquest annex es detalla la notació que s'ha utilitzat per a descriure els camps dels missatges i les ordres suportades pels diferents protocols.

Cadena literal

Les cadenes literals es definiran amb lletra d'espaiat fix.

```
cadena
```

Variable

Les variables es definiran amb lletra inclinada d'espaiat fix o lletra cursiva.

```
variable
```

Alternativa

Per a indicar que en una especificació s'ha de triar un element d'una llista de *n* elements, s'utilitza el caràcter "|" per a separar les diferents alternatives.

```
element1 | element2 | ... | elementn
```

Repetició

Per als elements que es poden repetir entre *n* i *m* vegades, s'utilitza la construcció `n*m` com a prefix. Els valors *n* i *m* es poden suprimir. En aquest cas, es prenen per defecte els valors *n* = 0 i *m* = buit.

```
n*melement
n*element
*melement
*element
```

Opcional

Per a indicar que una cadena o una variable són opcionals, es tanquen entre els caràcters "[" i "]".

```
[element]
```

Repetició específica

Quan es vol indicar que un element ha d'aparèixer amb exactitud un nombre enter *n* vegades, s'ha de posar el nombre *n* com a prefix.

```
nelement
```

Llista

Si es vol especificar una llista d'entre *n* i *m* elements separats per comes, s'utilitza la construcció n#m com a prefix. Els valors *n* i *m* es poden suprimir. En aquest cas es prenen els valors per defecte, que són *n* = 0 i *m* = buit.

```
n#melement
```

Agrupament

Quan es vol que un o més elements siguin tractats com un únic element per als operadors |, * i #, s'indica tancant-los amb els caràcters "(" i ")". Se sol utilitzar en especificacions de repeticions i llistes.

```
(element1 element2... elementn)
```

Caràcters especials

Alguns caràcters especials s'inclouen entre els caràcters "<" i ">".

```
<CR>: retorn de carro (CR, carriage return)
<LF>: salt de línia (LF, line feed)
<CRLF>: retorn de carro + salt de línia
```

ANOTACIONS

Annex 4

Codis de resposta

Les diferents RFC defineixen els codis de resposta possibles per a cada una de les seves comandes , i també el seu significat. Aquests codis estan formats per tres dígits i, per a facilitar-ne la interpretació, porten associat un significat concret per a cada dígit.

Primer dígit

Indica si l'operació s'ha efectuat o no.

Taula 18.

Primer dígit	
1iz	Resposta preliminar: s'ha iniciat l'operació i, quan s'hagi completat, el servidor enviarà una resposta nova.
2iz	Operació completada correctament.
3iz	Resposta intermèdia: l'operació s'ha acceptat; tanmateix, el client ha d'enviar comandes noves amb més informació per a completar-la.
4iz	Resposta negativa temporal: no es pot dur a terme l'operació, però es podrà efectuar si el client ho reintenta.
5yz	Resposta negativa definitiva: no es pot efectuar l'operació i probablement tampoc no es podrà dur a terme si el client ho reintenta.

Nota

Quan es vol llegir un fitxer amb accés temporalment bloquejat perquè un altre procés hi escriu dades, el servidor envia una resposta negativa temporal. En canvi, quan es vol llegir un fitxer que no existeix, el servidor envia una resposta negativa definitiva, encara que sempre hi hagi la possibilitat que mentrestant un altre procés el creï.

Segon dígit

Indica el tipus de resposta.

Taula 19.

Segon dígit	
x0z	Error de sintaxi, comanda no implementada o, en general, resposta no pertanyent a cap de les altres categories.
x1z	Resposta informativa.
x2z	Resposta referent a les connexions.
x3z	Resposta referent al procés d'autenticació (nom d'usuari, contrasenya, etc.).
x5z	Resposta referent a l'aplicació concreta (per exemple, al sistema de fitxers o de correu).

ANOTACIONS

Tercer dígit

El tercer dígit complementa la informació del segon dígit per a especificar de quina resposta concreta es tracta.

ANOTACIONS

GNU Free Documentation License

GNU Free Documentation License
Version 1.2, November 2002

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

ANOTACIONS

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

ANOTACIONS

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent.

An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

ANOTACIONS

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition.

Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

ANOTACIONS

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material.

If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

ANOTACIONS

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

ANOTACIONS

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

ANOTACIONS

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number.

Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

ANOTACIONS

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit.

When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form.

Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of

ANOTACIONS

those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

ANOTACIONS

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

ANOTACIONS


UOC
La universidad virtual
Formació de Postgrau