La capa de xarxa

René Serral i Gracià
Miquel Font Rosselló
Xavier Vilajosana Guillén
Eduard Lara Ochoa

PID_00171176

Universitat Oberta de Catalunya
www.uoc.edu

Índex

Introducció

La capa de xarxa s'encarrega de proporcionar connectivitat i d'oferir mecanismes per a la selecció del millor camí entre dos punts separats de la xarxa, de manera que permet la interconnexió d'equips que poden estar ubicats en xarxes geogràficament separades entre si, tot garantint la connectivitat d'extrem a extrem, independentment de la tecnologia d'enllaç de dades utilitzada i del camí que segueixi la informació en els punts intermedis.

Els principals avantatges que ens proporciona aquesta capa són, per una banda, independència de la tecnologia de xarxa (envers les capes inferiors), i per l'altra un sistema d'abstracció que permet utilitzar una gran diversitat d'aplicacions i protocols de transport (envers capes superiors), com per exemple TCP o UDP, vistos en el mòdul "La capa de transport de dades".

Bàsicament la capa de xarxa, especialment a Internet, està composta per tres grans blocs. Primer el protocol, que descriu la manera d'enviar informació; segon, el protocol d'encaminament, que decideix per on han d'anar els datagrames per a arribar a la seva destinació. Finalment, la capa de xarxa també identifica el mecanisme per a informar de qualsevol error que s'hagi produït en l'enviament de la informació.

L'objectiu d'aquest mòdul és descriure com implementa la comunicació la capa de xarxa. Així, l'apartat 1 introduirà els fonaments i funcionalitats bàsiques presents en aquesta capa. L'apartat 2 descriurà els diferents serveis que proporciona la capa de xarxa des del punt de vista de l'enviament d'informació entre els diferents nodes. El mòdul continua amb l'apartat 3, que conté la part més important del capítol, la descripció del protocol de xarxa més utilitzat avui dia: l'Internet Protocol; hi descrivim què és i com s'utilitza en un entorn real com Internet. Per completar la comprensió del protocol s'introduiran els mecanismes existents per a fer arribar la informació enviada a qualsevol part de la xarxa. Finalment, el mòdul acaba amb les conclusions de la capa de xarxa i del protocol IP.

1. Funcionalitats bàsiques: encaminament

Una xarxa està composta bàsicament per dos tipus d'entitats: els equips finals[1] i els encaminadors[2].

(1)En anglès, *hosts*. També se'n poden dir *clients*.

(2)En anglès, *routers*.

Els equips finals són els equips de xarxa encarregats de la comunicació, són l'origen i el final de la xarxa. Normalment són servidors d'informació o bé equips d'usuaris finals que accedeixen als servidors. Per la seva part, els encaminadors, tot i que en segons quins casos també poden ser equips finals, es limiten a enviar la informació que reben per una interfície d'entrada a la corresponent de sortida que porti els datagrames cap a la seva destinació. Per a poder saber cap on va la informació els encaminadors s'ajuden del que es coneix com a *taules d'encaminament*.

Si bé deixem per a l'apartat 4 la descripció dels principals algorismes d'encaminament, i per a l'apartat 5 la descripció de l'encaminament dins d'una xarxa com Internet, en aquesta secció detallarem les tasques generals a la capa de xarxa que fan els encaminadors, com fan l'enviament d'informació i els problemes amb què es poden trobar.

La capa de xarxa necessita que tant els encaminadors com els equips finals tinguin un identificador únic. Aquest identificador permet que qualsevol altre equip de la xarxa el pugui localitzar i enviar-li informació. En particular, en una xarxa com Internet aquests identificadors es coneixen com a adreces (adreces IP).

Vegeu també

Vegeu les adreces IP en l'apartat 3 d'aquest mòdul didàctic.

La figura 1 mostra una xarxa amb vuit encaminadors i dos equips finals. A la figura també es pot observar una simplificació de com funciona un encaminador internament. Per simplicitat, en comptes d'indicar les adreces dels diversos equips els hem identificat per una banda amb *R* (de *router*) i un número que identifica els diferents encaminadors, i per l'altra amb *H* (de *host*) i un número per a identificar els diferents equips finals.

Els encaminadors estan compostos per una sèrie d'interfícies d'entrada i sortida, que són les encarregades de rebre els datagrames dels equips veïns; aquestes interfícies estan controlades per unes cues[3], que emmagatzemen els paquets (d'entrada o de sortida) per a poder-los enviar quan sigui possible, o el que és el mateix, quan l'encaminador tingui recursos per a atendre les cues d'entrada, o bé quan la xarxa tingui recursos (amplada de banda disponible) per a les cues de sortida. Internament l'encaminador disposa d'una lògica per a decidir què

(3)En anglès, *buffers*.

cal fer amb els datagrames que arriben. Aquesta decisió normalment implica enviar el datagrama per una altra interfície que el portarà més prop de la seva destinació.

Així el datagrama va saltant pels encaminadors fins a arribar a la destinació. Cada equip de xarxa pel qual passa el datagrama es coneix com a *salt*[4]. Cal notar que els encaminadors treballen a nivell de xarxa, cosa que vol dir que no interpreten els camps presents als nivells superiors, tal com mostra la figura 2.

(4)En anglès, *hop*.

Figura 1. Exemple de xarxa amb encaminadors i equips finals

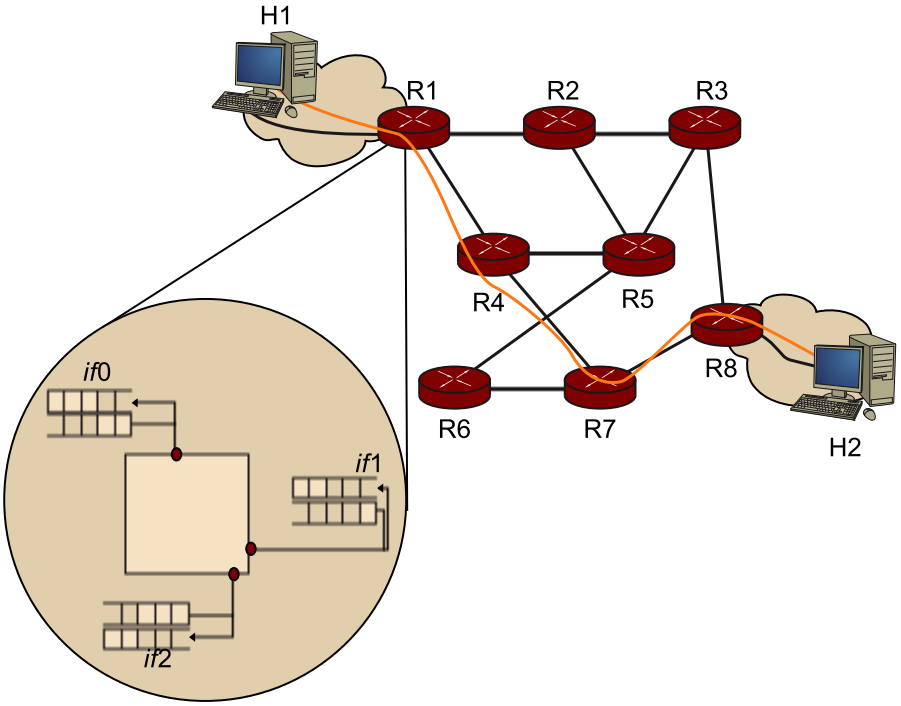
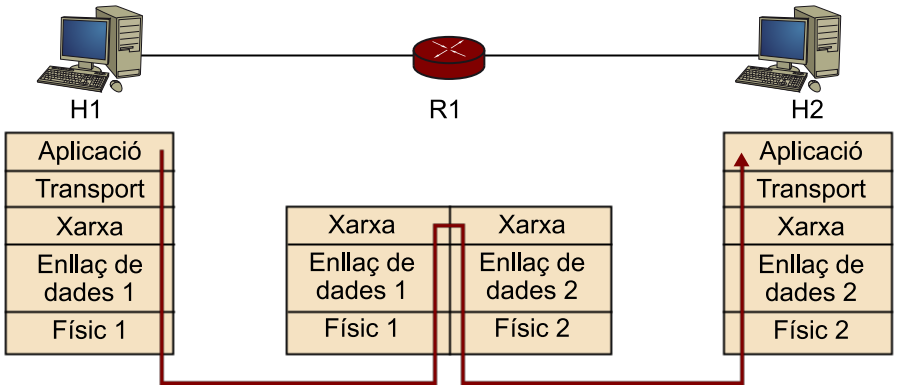


Figura 2. Capes usades per a l'encaminament en protocols de xarxa



Quan un equip envia un datagrama cap a una destinació, aquest datagrama inicialment va dirigit a l'encaminador associat a la xarxa de l'equip. Aquest encaminador mirarà la destinació del datagrama, i l'enviarà per la interfície que el porti cap a la seva destinació depenent d'una taula d'encaminament. L'encaminador següent farà el mateix fins que el datagrama arribi a la seva destinació final. La llista d'encaminadors que segueix un datagrama es coneix com el camí[5] del datagrama. Cal notar que aquest camí serà diferent depenent

(5)En anglès, *path*.

de l'origen i la destinació del datagrama. Com a exemple es pot veure a la figura 1 que el camí que segueixen els datagrames per a anar des de H1 fins a H2 és H1-R1-R4-R7-R8-H2, de manera que fa un total de cinc salts per a arribar a la destinació.

Hem dit que l'equip envia el datagrama a l'encaminador de la seva xarxa, i això implica que s'ha de tenir coneixement *a priori* de com cal arribar a aquest encaminador per tal de poder-li enviar el datagrama. La seqüència específica d'accions que fa l'equip es poden veure a la figura 3, més detalladament:

1) Es crea el datagrama amb les adreces de xarxa origen i destinació apropiades.

2) Es busca l'adreça de xarxa de l'encaminador: pròxim salt[6] (o de l'equip final si està directament connectat a l'encaminador: darrer salt[7]).

[6] En anglès, *next hop*.

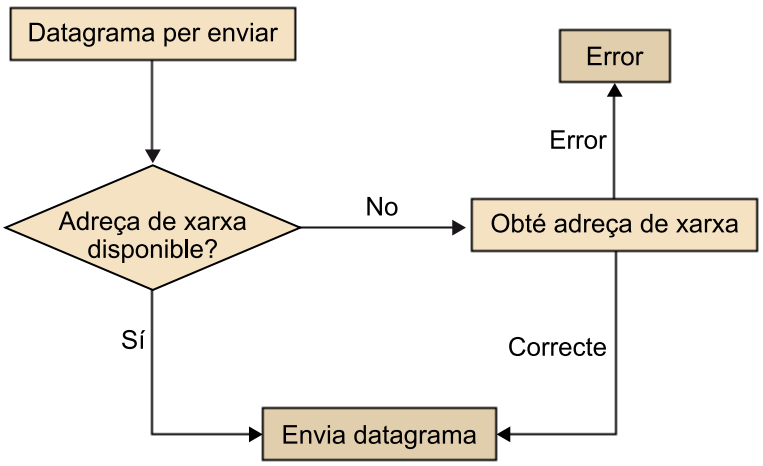
[7] En anglès, *last hop*.

3) Si no es disposa de l'adreça de xarxa sortim amb error, ja que no sabem quin és el pròxim salt per a enviar el datagrama.

4) Si hem pogut aconseguir l'adreça enviem el datagrama (amb les adreces de xarxa origen i destinació originals) al pròxim salt del camí o a l'equip final si estem a l'últim salt.

5) Cal repetir des del pas 2 fins que el datagrama arribi a la seva destinació.

Figura 3. Diagrama de blocs simplificat de l'enviament d'un datagrama a un equip de xarxa



Un punt important per considerar és que el datagrama no segueix qualsevol camí, sinó que els encaminadors disposen d'una taula d'encaminament[8] que indica per quina interfície s'han d'enviar els datagrames depenent de la seva destinació. Per a omplir aquestes taules és necessari utilitzar uns algorismes d'encaminament, que veurem amb detall en l'apartat 4.

[8] En anglès, *forwarding table* o *routing table*.

2. Serveis de xarxa

El servei de xarxa defineix les característiques que ha de tenir el transport punt a punt de les dades a la capa de xarxa. Així es defineixen característiques com la fiabilitat a l'hora d'enviar la informació, l'ordre d'arribada dels paquets, els llindars de retard en fer arribar la informació a la destinació, la informació de congestió a la xarxa, etc., entre els diferents emissors i receptors dins de la xarxa.

Actualment hi ha dos models de serveis de xarxa clarament diferenciats, el model de circuit virtual i el model de datagrama. A continuació es descriuen tots dos, fent èmfasi en el model de datagrama, ja que és l'utilitzat pel nivell de xarxa proposat per a Internet, i per tant el més rellevant actualment.

2.1. Model de xarxa en mode de circuits virtuals

Els circuits virtuals seran explicats en detall en el mòdul 4, ja que en general es consideren del nivell d'enllaç de dades. Aquest subpartat només en fa una breu introducció per a comprendre millor la capa de xarxa.

Vegeu també

Vegeu els circuits virtuals en el mòdul "Nivell d'enllaç i xarxes d'àrea local" d'aquesta assignatura.

> Un circuit virtual és un camí que es preconfigura entre dos punts de la xarxa de manera que els nodes intermedis saben *a priori* l'adreça a la qual s'ha d'enviar la informació pertanyent a cada circuit.

Aquest paradigma permet accelerar enormement l'enviament de paquets entre dos punts, ja que el processament intermedi és mínim; aquesta prereserva, a sobre, permet garantir una sèrie de recursos de xarxa per al trànsit que passa pel circuit. Per això aquest model de xarxa es va pensar per a serveis en temps real (multimèdia).

En qualsevol circuit virtual es poden distingir tres fases clarament separades:

1) Establiment del circuit virtual: aquesta fase s'inicia a la capa de xarxa de l'emissor, utilitzant l'adreça del receptor. L'emissor envia un datagrama de creació de circuit que provoca que cada node intermedi reservi els recursos demanats de manera iterativa fins a arribar a la destinació. Cada un dels nodes intermedis haurà d'actualitzar el seu estat per a acomodar el nou circuit, o denegar-ne la creació en cas que no quedin més recursos disponibles (normalment amplada de banda). Si l'establiment del circuit pot arribar fins al destinatari s'avisa a l'emissor indicant que la connexió ha estat satisfactòria i que es pot començar a enviar informació.

2) Transferència de dades: en el cas que s'hagi pogut establir el circuit virtual es poden començar a enviar dades entre els dos punts.

3) Desconnexió del circuit virtual: aquesta desconnexió pot ser iniciada tant per l'emissor com pel receptor, i s'avisa seqüencialment per mitjà de la capa de xarxa a tots els nodes intermedis fins a arribar a l'altre extrem. Aquesta desconnexió permet alliberar els recursos ocupats pel circuit.

Exercicis

1. Quines diferències creieu que hi pot haver entre l'inici d'un circuit virtual a la capa de xarxa i l'establiment d'una connexió a la capa de transport? (per exemple el *three way handshaking*).

Solució de l'exercici 1

L'establiment de la connexió de la capa de transport involucra únicament dos sistemes finals. Els dos extrems acorden la comunicació i determinen els paràmetres de connexió, mentre que els nodes intermedis de la xarxa no hi intervenen. En contraposició, l'establiment d'un circuit virtual a la capa de xarxa obliga a involucrar tots els nodes intermedis.

2. Indiqueu tres tecnologies de xarxa que utilitzin circuits virtuals.

Solució de l'exercici 2

ATM, la retransmissió de trama i X.25. Molts autors han considerat i consideren ATM i la retransmissió de trama com a tecnologies de nivell d'enllaç; de totes maneres, aquesta consideració es fa pel fet que totes dues poden transportar trànsit IP (el trànsit present a Internet).

El principal inconvenient que té la utilització de circuits virtuals és que els nodes intermedis han de mantenir les reserves de recursos demanades independentment que s'estiguin utilitzant, amb el problema potencial d'infrautilitzar la xarxa.

2.2. Model de xarxa en mode datagrama

Si enviar informació a través d'un circuit virtual implica prèviament establir un camí i reservar recursos, en una xarxa en mode datagrama (també anomenat **commutació de paquets**), el paquet s'envia directament a la xarxa amb una adreça origen i una adreça destinació. Aleshores és feina de la xarxa (per mitjà de les taules d'encaminament de cada encaminador) fer arribar el paquet a la seva destinació.

Com es pot comprovar, en aquest tipus de comunicació no hi ha ni reserva de recursos ni camí preestablert entre els extrems de la comunicació. Per tant, a un encaminador poden arribar datagrames de diferents destinacions a la vegada, i els datagrames poden seguir camins diferents per a arribar a la destinació (de-

penent dels algorismes d'encaminament), fet que provoca l'efecte col·lateral que els paquets poden arribar fora d'ordre (el paquet número 2 arriba abans que el número 1).

Les xarxes en mode datagrama són les més usades actualment principalment a causa que el protocol de xarxa d'Internet (IP) l'utilitza.

Tot i que hem vist que el model de datagrama fa una utilització dels recursos més eficient, això té un cost associat. Amb aquest tipus de xarxes es complica moltíssim la priorització del trànsit, ja que mai no se sap *a priori* quant trànsit es rebrà, i el que és més greu, no se sap quina prioritat s'ha de donar a cada un dels fluxos de dades presents a la xarxa; tant és així que Internet es basa en el paradigma conegut com a ***best effort***, que implica que la Xarxa no ens dóna cap garantia de qualitat i que "ho farà el millor que pugui" per a fer arribar el datagrama a la seva destinació.

Exercicis

3. Quin dels dos models de xarxa vistos consideres que fa un ús dels recursos més eficient?

Solució de l'exercici 3

El fet que un circuit virtual obliga a fer una prereserva de recursos implica que s'ha de saber prèviament el model i el patró de trànsit que segueix l'aplicació; com això molts cops no és possible *a priori*, s'acostuma a fer el que es coneix com a *overprovisioning* (reservar més recursos dels que es consideren necessaris), cosa que inequívocament porta a un sistema menys eficient en termes de recursos.

Per la seva banda, utilitzar el mode datagrama no implica cap prereserva, amb la qual cosa la xarxa sempre enviarà tan ràpid com pugui la informació, sempre que hi hagi recursos disponibles.

2.3. Servei de xarxa orientat i no orientat a la connexió

Anàlogament als protocols de transport que hem vist anteriorment, en el nivell de xarxa també podem tenir protocols que siguin orientats a connexió i d'altres que no ho siguin. La principal diferència entre les dues alternatives és que el servei orientat a connexió desa l'estat de la connexió, o el que és el mateix, té coneixement de totes les connexions establertes, mentre que en el cas del servei no orientat a connexió, no es té constància de les connexions existents. Un exemple clar de servei de xarxa orientat a la connexió és el model de circuits virtuals vist anteriorment.

Cal notar que el disseny d'un protocol de xarxa no orientat a connexió no exclou que en nivells superiors (transport) es pugui definir un protocol orientat a connexió. L'exemple més indicatiu d'això és la pila de protocols TCP/IP[9], en què TCP és orientat a la connexió mentre que IP no ho és. Tant és així que

(9) *TCP* és la sigla de *transmission control protocol*. *IP* és la sigla d'*Internet protocol*.

l'arquitectura actual d'Internet només proporciona el model de servei de datagrama, cosa que no garanteix l'ordre dels paquets, el retard en l'enviament, ni l'arribada del datagrama.

3. Adreçament a Internet. El protocol IP

El protocol de capa de xarxa per excel·lència és el protocol d'Internet (IP). IP és un protocol que basa l'intercanvi d'informació en el model no orientat a connexió. IP és el protocol utilitzat a Internet per a identificar els nodes de la xarxa, i també s'utilitza per a enviar la informació d'una manera estàndard i independent de la tecnologia de xarxa utilitzada (teniu més informació sobre tecnologies de xarxa als capítols següents). Una altra característica molt important és que IP no implementa mecanismes que garanteixin la integritat de les dades que s'envien per la xarxa (això es fa a la capa de transport); només es verifica que no hi hagi errors de transmissió a la capçalera.

Tots els protocols de xarxa requereixen algun mecanisme per tal d'identificar els nodes de la xarxa; aquesta identificació en el protocol IP es fa per mitjà del que es coneix com a adreça IP.

Actualment hi ha dues versions diferents del protocol IP: iPv4 i IPv6. IPv4 és el protocol més utilitzat actualment a Internet, però atès el gran creixement que ha sofert la Xarxa, se n'ha proposat una extensió, IPv6, més actual i que algun dia es preveu que substitueixi IPv4. En els subapartats següents es detalla com funcionen tots dos protocols i quins avantatges i inconvenients tenen.

3.1. IPv4

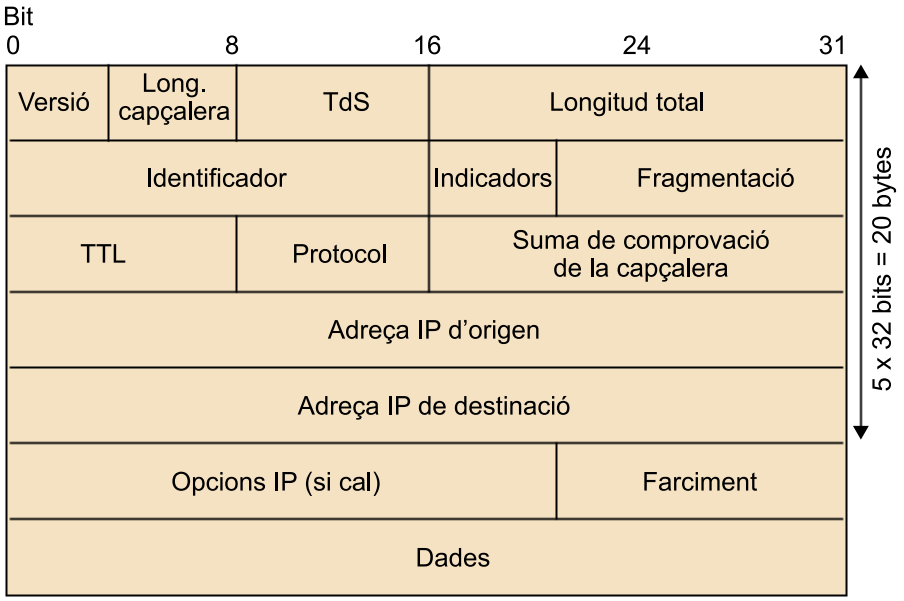
IPv4 va ser proposat el 1981 i actualment encara és el protocol de xarxa per excel·lència. IPv4 defineix el format que s'ha d'utilitzar per a enviar informació entre dos punts distants de la xarxa. El protocol proporciona mecanismes que determinen com es divideix l'adreçament d'una manera escalable en una xarxa tan gran com Internet.

3.1.1. La capçalera IP

IPv4 defineix quina informació de control i quin format han de tenir els paquets que s'envien a la Xarxa. Per això, i igual com ocorre amb els protocols de transport vistos en l'apartat anterior, és necessari definir una capçalera que serveixi per a poder identificar els paquets.

La capçalera d'IPv4 es pot veure a la figura 4.

Figura 4. Capçalera IPv4



Parts de la capçalera IPv4:

- **Versió** [4 bits]: indica quin protocol de xarxa utilitza aquest datagrama. Per a IPv4 està fixat a 0 × 04.

- **Longitud de la capçalera** [4 bits]: la capçalera IP pot tenir una mida variable a causa del camp d'opcions. Aquesta mida indica en quin punt comencen les dades del protocol de transport. En particular, aquest camp indica el valor en funció de la quantitat de paraules de 4 octets[10] que té la capçalera; així un valor de 0 × 05 vol dir una capçalera de 20 octets, el valor usat en la majoria dels casos per ser la mida per defecte quan no hi ha opcions.

- **Tipus de servei (TdS)** [8 bits]: aquest camp permet distingir entre diferents tipus de datagrames IP; inicialment es van definir paràmetres en funció del retard baix, la taxa de transferència alta o la fiabilitat. Així, depenent del tipus de trànsit que contingui el paquet, per exemple trànsit interactiu, podem voler un retard baix, o en el cas que el trànsit sigui de baixa prioritat es pot voler un cost mínim. En la realitat, els encaminadors, normalment ignoren aquest camp i utilitzen la tècnica *best effort* per a encaminar els paquets.

- **Longitud total** [16 bits]: indica la mida total del datagrama en octets, que inclou la capçalera i el camp de dades. Els 16 bits indiquen una mida màxima del datagrama de 65.535 octets. Tot i que en general la mida màxima utilitzada és de 1.500 octets.

[10] En anglès, *bytes*.

- **Identificador** [16 bits], **indicadors** [3 bits] i **fragmentació** [13 bits]: aquests camps fan referència al que es coneix com a *fragmentació IP*. La fragmentació serà explicada més endavant a aquesta mateixa secció.

- **TTL** [8 bits]: inicialment aquest camp feia referència al temps de vida del datagrama en mil·lisegons. Però en la pràctica conté el màxim nombre d'encaminadors que pot travessar el paquet fins que arribi a la destinació. A cada salt, un encaminador decrementa en 1 el valor d'aquest camp, i quan el TTL arriba a 0 el paquet és descartat. Amb aquesta tècnica es permeten descartar datagrames en el cas que hi hagi algun bucle provocat per algun problema amb el sistema d'encaminament, i així evitar tenir paquets a la xarxa més temps del necessari. D'aquest camp es pot derivar que el "diàmetre" màxim possible d'Internet és de 255 salts. Tot i que actualment no acostuma a superar els 30.

- **Protocol** [8 bits]: aquest camp indica el protocol present a la capa de transport, que serà capaç d'interpretar-lo. Normalment aquest camp pot ser 0 × 06 per a TCP o 0 × 11 per a UDP[11]; la llista completa es pot trobar a Internet. Amb aquest enllaç entre la capa de xarxa i la de transport, podem tenir diversos protocols de transport i distingir-los fàcilment, i passar el control al corresponent de manera eficient.

(11) *UDP* és la sigla de *user datagram protocol*.

- **Suma de comprovació de capçalera** [16 bits]: permet detectar algun tipus d'error de transmissió a la capçalera. És important notar que no es comprova la integritat de la capa de transport i superiors. Recordem que IP no garanteix la recepció de les dades. La suma de comprovació[12] es calcula tractant cada dos octets de la capçalera com a enters i sumant-los utilitzant aritmètica de complement a 1, ignorant per a la suma el mateix camp que conté la suma de comprovació. La integritat es comprova comparant la suma amb l'emmagatzemada a la capçalera. En el cas d'error el paquet es descarta. Un petit inconvenient d'aquesta suma de comprovació és que cada encaminador l'ha de recalcular per a cada paquet, atès que el camp TTL (i potser algunes opcions) canvien a cada salt.

(12) En anglès, *checksum*.

- **Adreça d'origen** [32 bits]: indica l'adreça origen del paquet. Es pot trobar més informació sobre l'adreçament més endavant en aquesta secció.

- **Adreça de destinació** [32 bits]: on va dirigit el paquet.

- **Opcions IP**: aquest camp és el que fa que la capçalera IP pugui ser variable en mida. Les opcions, que normalment no s'utilitzen, permeten ampliar les funcionalitats de la capçalera IP. Tot i no fer-se servir gairebé mai, el fet de comprovar-ne l'existència a cada encaminador fa baixar molt el rendiment del protocol IPv4. Per això durant el disseny de la versió 6 del protocol es va canviar la manera d'implementar aquestes opcions.

- ***Padding*** (farciment): per motius d'eficiència les dades han de començar en una posició múltiple de 4 octets; per tant, en el cas que algunes opcions introdueixin una desalineació, el *padding*, que normalment són tot zeros, alinea a la paraula del camp següent.

- ***Data* (*payload*)**: les dades del datagrama que es passaran al nivell de transport, o sigui, la informació que realment es vol transmetre.

Fragmentació IP

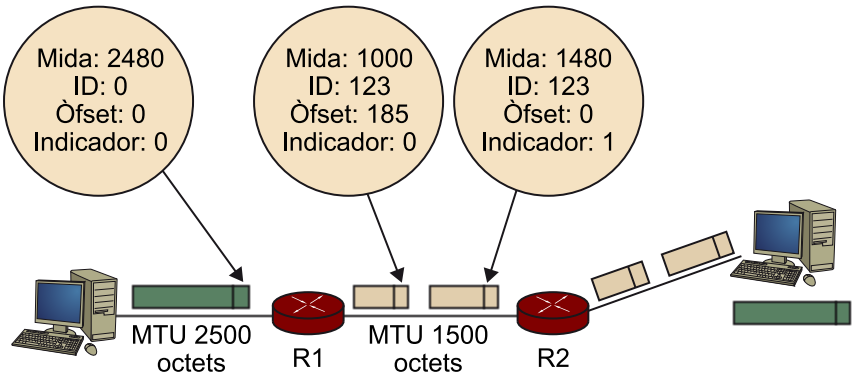
Un dels punts més crítics a l'hora de dissenyar el protocol IP va ser la necessitat d'introduir la fragmentació. La fragmentació IP és necessària perquè, com es veurà en el mòdul següent, no totes les xarxes, ni tots els protocols d'enllaç de dades, poden transportar paquets de mida arbitrària. En general, la mida màxima estarà delimitada depenent de la tecnologia de xarxa utilitzada. Per tant, a causa de la diversitat de tecnologies que coexisteixen a Internet actualment, ens podem trobar casos en què la mida màxima de trama permesa, MTU[13], sigui menor en algun encaminador dins del camí que segueixen els datagrames, i això força IP a dividir la trama en fragments més menuts que puguin ser transmesos. Un exemple pot ser Ethernet, que permet trames de mida màxima de 1.500 octets, mentre que una tecnologia com ATM en general té el màxim a 9.180 octets. Cal notar que quan es fragmenta un datagrama IP, cada fragment ha de ser autocontingut, i ha de poder ser assemblat a la destinació final (fer-ho als encaminadors intermedis representaria una pèrdua de rendiment considerable), i per això només dividir el datagrama no és suficient: és necessari fer-hi algun tipus de procés.

(13)*MTU* és la sigla de *maximum transfer unit*.

Quan s'ha de dividir un datagrama IP, primer es replica la capçalera IP per a cada fragment, i tot seguit s'actualitzen els camps de la capçalera *identification*, *flag*, i *fragmentation offset*. Així, tots els fragments pertanyents al mateix datagrama tindran el mateix identificador, cada fragment contindrà el desplaçament i finalment l'indicador[14], que serà 1 si hi ha més paquets o 0 si és el darrer. Per restriccions en la implementació i per reduir el nombre de bits que es fan servir per a emmagatzemar aquest desplaçament es va decidir fer-ho amb múltiples de 8 octets, i així un desplaçament de 64 octets (o sigui, que el fragment IP conté des de l'octet 65 del datagrama original) es representarà amb un 8 al camp *fragmentation offset* (ja que $8 \times 8 = 64$). La figura 5 en mostra un exemple.

(14)En anglès, *flag*.

Figura 5. Exemple de fragmentació



A la figura es pot veure un cas en què un equip envia un paquet de mida MTU = 2.500 octets. Això vol dir que el paquet tindrà 2.480 octets d'informació útil i 20 de capçalera. A l'hora de fragmentar, es generen dos paquets diferents, un de 1.480 + 20 i un altre de 1.000 + 20. Com es pot veure, la mida útil no canvia, però pel fet de tenir dos paquets diferents estem replicant la capçalera. El valor de l'identificador està determinat per un comptador intern a l'encaminador que fragmenta; el desplaçament[15] per al primer fragment és 0, i l'indicador 1, i això indica que encara hi ha més fragments; per al segon fragment el desplaçament conté un 185, ja que s'especifica en grups de 8 octets, i un 0 a l'indicador indica que es tracta del darrer fragment del datagrama original. Quan el datagrama arribi a la seva destinació final serà reassemblat i passat als nivells superiors de manera transparent.

(15) En anglès, *offset*.

Exercicis

4. Per què creieu que es fa una verificació de la suma de comprovació tant al nivell de xarxa com al nivell de transport?

Solució de l'exercici 4

Cada una de les verificacions té funcionalitats diferents. IP té per objectiu enviar el paquet a una destinació, i per tant el que fa és validar que la capçalera sigui correcta; si hi ha un error a les capes superiors seran les encarregades de fer la verificació. Un altre motiu és l'eficiència: si cada encaminador ha de verificar que tot el *payload* és correcte això implica massa cost computacional. Per altra banda, TCP, com ja hem vist anteriorment, ha de garantir que les dades arriben correctament, i per això fa una verificació de tot el *payload*, directament a la destinació. Això evita sobrecarregar innecessàriament la xarxa.

5. Quines implicacions té el fet de tenir un camp d'opcions a la capçalera IP?

Solució de l'exercici 5

Tenir opcions a la capçalera IP implica que tingui una mida variable (amb funcions opcionals), cosa que força a tenir més camps a la capçalera i més processament als encaminadors intermedis, ja que algunes de les opcions necessiten ser processades per l'encaminador mateix.

6. Sabent la mida de les capçaleres TCP i IP, quina és l'eficiència màxima en la transmissió que podem aconseguir utilitzant aquests protocols?

Solució de l'exercici 6

L'eficiència en la transmissió es pot calcular mirant quants bits útils s'envien sobre el total; així:

$$E_t = 1 - \frac{H_{IP} + H_T}{M}$$

H_{IP} és la mida de la capçalera IP, H_T és la mida de la capçalera de transport i M és la mida total del datagrama. En el cas de TCP/IP i una mida de datagrama de 1.500 octets l'eficiència és:

$$E_t = 1 - \frac{20 + 20}{1.500} = 0{,}973 = 97{,}3\%$$

o el que és el mateix, tenim una penalització del 2,6% en l'enviament d'informació.

7. Un paquet segueix un camí en què les MTU són 5.000 i 1.500. Un equip envia un paquet de 5.000 octets. Com es fragmentarà i quin desplaçament i indicadors contindran cada fragment?

Solució de l'exercici 7

Com la MTU menor és de 1.500 octets el datagrama es dividirà en quatre fragments, que han de contenir un total de 4.980 octets dividits de la manera següent: 1.480, 1.480, 1.480 i 560 octets, respectivament. Així els desplaçaments seran: 0, 1.480/8, (2 × 1.480)/8 i (3 × 1.480)/8, o sigui: 0, 185, 370 i 555. Finalment els indicadors seran tots 1 exceptuant el darrer paquet, que contindrà un 0.

3.1.2. Adreçament IPv4

Com ja s'ha introduït prèviament, els protocols de xarxa necessiten disposar d'una adreça única que permeti identificar tots els nodes de la xarxa. En el cas d'IPv4, tal com es pot deduir de la capçalera IPv4, la màxima quantitat d'adreces disponibles és molt gran 2^{32} (4.294.967.296).

Per simplificar-ne l'escriptura es divideixen els 32 bits en 4 blocs de 8 bits cada un; a més, en comptes d'utilitzar la representació binària, que és poc llegible, en la pràctica una adreça IP s'escriu en notació decimal separada per punts; una adreça estarà formada per 4 blocs de nombres entre 0 i $2^8 - 1$ (255), com per exemple:

Representació binària i decimal d'una adreça IP

```
10001111 00101101 00000001 00010111
   143   .   45   .    1   .   23
```

A més a més, tenir un nombre tan gran d'adreces representa un enorme problema de gestió, i per això es va proposar un sistema d'assignació d'adreces jeràrquic. A Internet les adreces IP estan compostes per dues parts, la part de xarxa i la part de l'equip, i això s'utilitza per a poder estructurar les adreces i organitzar-les per zones administratives.

La part de xarxa està formada pels bits superiors de l'adreça IP i indica a quina xarxa pertanyen un conjunt d'equips, o el que és el mateix, qui és l'encaminador de sortida del conjunt d'equips. Per contra, la part de l'equip són els bits inferiors de l'adreça IP, i identifica l'equip dins de la seva xarxa. Inicialment aquesta divisió en xarxes es va fer per mitjà de classes; concretament, es van definir cinc classes diferents (A, B, C, D i E), tal com mostra la taula següent:

Divisió de xarxes per classes			
Classe	**Bits inicials**	**Bits xarxa**	**Rang xarxa**
A	0	7 (+ 1)	1.0.0.0 – 127.0.0.0
B	1 0	14 (+ 2)	128.0.0.0 – 191.255.0.0
C	1 1 0	21 (+ 3)	192.0.0.0 – 223.255.255.0
D	1 1 1 0	-	224.0.0.0 – 239.0.0.0
E	1 1 1 1 0	-	240.0.0.0 – 255.0.0.0

Les adreces de classe A són les destinades a moltes grans empreses, com per exemple IBM o grans operadores americanes com AT&T WorldNet Services, i proporcionen accés a 2^{24} (16.777.216) equips per xarxa, en què 8 bits estan destinats a identificar la xarxa i la resta fins als 32 s'utilitzen per als equips finals. D'adreces de classe A n'hi ha un total de 2^7 (255). Com veurem més endavant, aquest repartiment de classes A és un dels causants de la forta mancança d'adreces IP avui dia.

Les adreces de classe B són les que es donen a grans entitats, universitats i segons quins proveïdors d'Internet. Permeten repartir 2^{16} (65.536) equips per xarxa, i hi ha un total de 2^{14} (16.384) adreces de classe B.

En el cas de les adreces de classe C, són les destinades a mitjanes empreses amb forta presència a Internet; en aquest cas es disposa de 2^8 (256) adreces, amb un total de 2^{21} (2.097.152) adreces de tipus C per a repartir.

Pel que fa a les adreces de classe D, es consideren un tipus de classe especial, anomenades *classes multidifusió*, que serveixen per a enviar trànsit anomenat *punt multipunt*, aquest tipus de trànsit es veurà en detall més endavant en aquest apartat. Finalment les adreces de classe E estan reservades per a un ús futur.

Adreces de propòsit específic

A part de la divisió en xarxes també es van destinar una sèrie d'adreces de propòsit específic per a casos especials, com són les adreces d'equip final, adreces de xarxa, les adreces de difusió[16], les adreces de *loopback* i les adreces privades.

[16]En anglès, *broadcast*.

- Les adreces d'equip final indiquen un equip dins de la xarxa actual i tenen la forma 0.**host**, en què *host* és la part de l'equip de la xarxa actual, o sigui, que la part de l'adreça de xarxa és tot 0.

- Les adreces de xarxa fan referència a la xarxa però no als equips dins d'aquesta; les adreces de xarxa són de la forma **xarxa**.0 per a adreces de classe C, **xarxa**.0.0 per a la classe B i **xarxa**.0.0.0 per a la classe A, o el que és el mateix, que l'adreça de l'equip de xarxa està tota a 0. Hi ha un cas especial, que és l'adreça 0.0.0.0, que indica "aquest equip final" d'"aquesta xarxa", tot i que no sempre s'implementa en els sistemes operatius actuals.

- Les adreces de difusió indiquen tots els equips d'una xarxa concreta. L'adreça es representa amb **xarxa**.255 per a adreces de classe C. Anàlogament al cas de les adreces de xarxa, les de classe B seran **xarxa**.255.255 i les de classe A **xarxa**.255.255.255, o sigui, que l'adreça de l'equip de xarxa és tot 1. Sempre que es rebi un datagrama a l'adreça de difusió tots els equips hi han de respondre. Les adreces de difusió tenen una adreça especial a la seva vegada, que és la 255.255.255.255, que fa referència a tota la Xarxa (Internet). Aleshores si algú enviés un datagrama a l'adreça 255.255.255.255 tota la Internet hi hauria de respondre. Com això provocaria greus problemes d'escalabilitat i excés de trànsit, no hi ha cap encaminador que reenviï trànsit de difusió per les seves interfícies. El trànsit de difusió sempre es quedarà a la xarxa que l'ha emès.

Exercicis

8. Donada l'adreça IP 120.1.32.54 indiqueu quina és l'adreça de xarxa, l'adreça de l'equip final i l'adreça de difusió de la xarxa.

Solució de l'exercici 8

L'adreça 120.1.32.54 forma part de les adreces de classe A; per tant, l'adreça de xarxa serà la 120.0.0.0, la de l'equip final la 0.1.32.54 i la de difusió seria la 120.255.255.255.

- Les adreces de *loopback* són des de la 127.0.0.0 fins a la 127.0.0.255, i són les que utilitzen internament els equips. Quan un equip arrenca, automàticament crea una interfície virtual (interfície de *loopback*) per a ús intern del sistema operatiu; normalment només s'utilitza la 127.0.0.1.

- Les adreces privades són les utilitzades per xarxes locals internes que no surten a Internet. La llista amb tots els rangs privats es pot trobar a la taula següent. Com es pot observar, es poden configurar internament diversos

rangs d'adreces privades, la seva utilitat és evitar col·lisions en assignacions d'adreces en configuracions internes amb altres nodes d'altres xarxes de l'exterior. Una altra funcionalitat és permetre assignar més adreces a les nostres xarxes que no pas IP públiques assignades pels operadors.

Llista de rangs d'adreces privades		
Classe	**Rang xarxa**	**Nombre de subxarxes**
A	10.0.0.0 - 10.255.255.255	1
B	172.16.0.0 - 172.31.255.255	16
C	192.168.0.0 - 192.168.255.255	255

El problema principal de les adreces privades és que no poden accedir a Internet directament; els encaminadors mai no enviaran a Internet el trànsit originat en adreces privades o amb destinació a aquestes adreces, ja que el pròxim salt no sabria com encaminar-lo. Per tal d'evitar aquesta limitació, i permetre la transferència de dades entre adreces privades i públiques, els encaminadors inclouen una tècnica anomenada *NAT*[17].

[17] *NAT* és la sigla de *network address translation*.

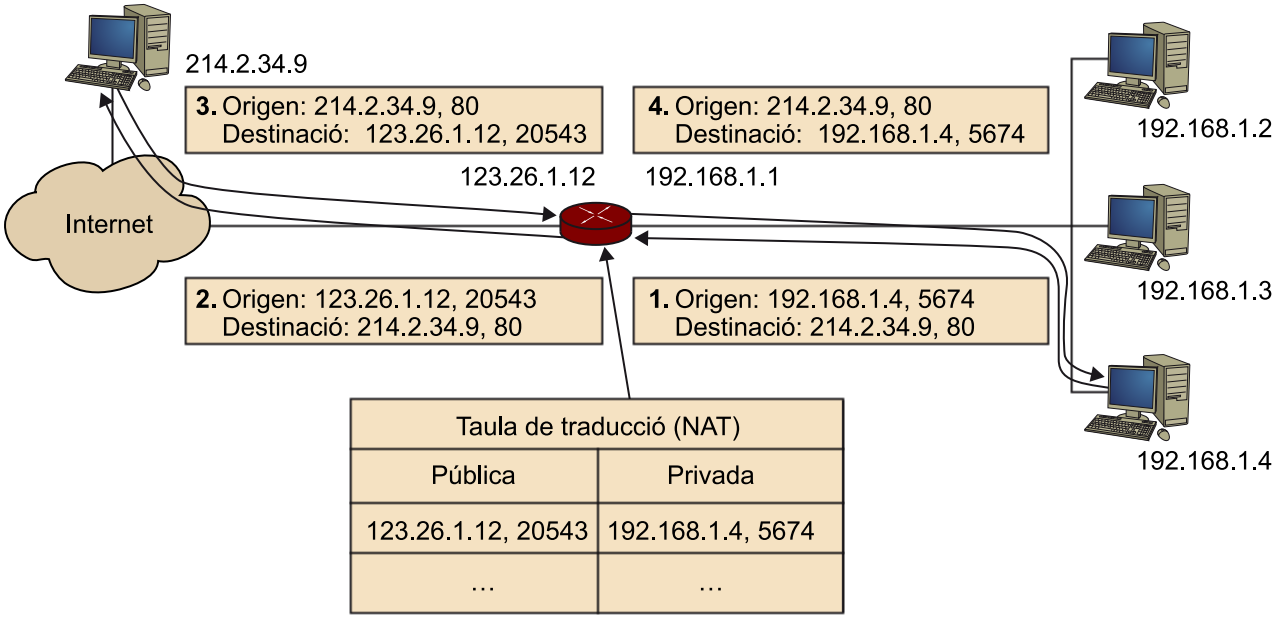
NAT

Pel que es pot deduir del que s'ha vist fins ara, cada equip d'una xarxa IPv4 ha de disposar d'una IP pública per a poder accedir a la Xarxa. Un dels problemes principals que es troben quan es demanen IP a les operadores és que generalment l'usuari (o l'empresa) té més equips que no IP assignades. Un exemple d'això és un usuari amb connexió ADSL, que rep una sola IP per part de la companyia telefònica, mentre que molts cops l'usuari disposa de diversos equips, com ara el PC de sobretaula, el portàtil, la PDA, etc. Per tal de permetre que tots els equips es puguin connectar a la Xarxa al mateix temps hi ha dues opcions: demanar més IP (solució difícil i cara) o bé utilitzar adreces IP privades, i configurar l'encaminador perquè faci la conversió des de l'adreça IP privada a la IP pública disponible. Això es pot aconseguir mitjançant NAT.

Exemple d'utilització de NAT

Si un client amb adreça privada vol establir una connexió amb un equip que té una adreça IP pública (punt 1 de la figura 6), com per exemple un servidor, el client enviarà el paquet cap a l'encaminador de la seva xarxa. Aquest encaminador tindrà configurada una taula de traducció, on transformarà la IP origen del datagrama a una IP pública que tingui reservada a tal efecte. Per a completar la traducció, l'encaminador maparà el port origen (de la capa de transport) a un port nou origen assignat per l'encaminador. El punt 2 de la figura 6 en mostra un exemple, en què l'encaminador transforma la IP origen (192.168.1.4) i el port origen (5674) de l'equip a la IP pública de l'encaminador (123.26.1.12) i un port assignat dinàmicament (20543 en l'exemple). L'estació destinació veu un datagrama com si hagués estat enviat per l'encaminador, al qual respon de la manera habitual usant TCP/IP. Finalment l'encaminador en rebre la resposta mira la taula de traducció i desfà el canvi per a enviar el paquet final a l'estació origen. Si l'entrada no hagués estat a la taula l'encaminador hauria assumit que el paquet anava realment dirigit a ell.

Figura 6. Exemple de xarxa amb NAT



Aquest mecanisme és molt útil per tal d'estalviar l'ús d'adreces públiques, tot i tenir una sèrie d'inconvenients que no el fan usable en segons quins entorns. Primer, hi ha protocols d'aplicació (per exemple, FTP) que incrusten la IP del client dins del datagrama; aquesta IP és usada pel servidor per a establir una nova connexió (per exemple, el cas de l'FTP actiu), i com el client incrusta la IP privada això impedeix que es pugui establir la connexió.

Un altre problema important és que totes les connexions s'han d'iniciar des de l'equip amb IP privada, ja que l'encaminador ha d'establir l'entrada a la taula de traducció abans de poder enviar informació cap a l'equip amb IP privada, fet que implica que normalment no es poden tenir servidors amb IP privades. Cal dir, però, que això es pot solucionar amb una tècnica anomenada *PAT*[18], en què l'encaminador té configurat de manera estàtica un mapatge pel qual, quan arriba un datagrama a un port concret, automàticament reenvia el paquet cap a l'equip amb IP privada que estigui configurat. En segons quins entorns el PAT es coneix també com a DNAT[19] o també com a *port forwarding*, però la idea de fons és la mateixa.

[18] *PAT* és la sigla de *port address translation*.

[19] *DNAT* és la sigla de *destination network address translation*.

3.1.3. CIDR

Un cop definides les diferents classes de xarxes es va veure que aquesta solució era clarament insuficient, ja que igualment forçava les grans i mitjanes operadores (amb classes A i B) a gestionar des d'un sol equip un nombre d'adreces massa gran, i per això es va proposar el CIDR[20].

[20] *CIDR* és la sigla de *classless inter domain routing*.

CIDR proposa un mecanisme més flexible per a poder subdividir les nostres xarxes. És important notar que CIDR no substitueix la divisió per classes, que continuen essent les unitats bàsiques d'assignació d'adreces; per contra, CIDR ens permet dividir les adreces assignades en subxarxes més petites i manejables.

Així amb CIDR la separació entre l'equip i la xarxa s'aconsegueix gràcies a una màscara. Aquesta màscara té la forma d'una adreça IP, que emmascara els bits d'una adreça normal per a poder distingir l'equip i la xarxa de manera senzilla. Així, per exemple, una màscara de 255.255.255.0 permet separar l'adreça de xarxa de la de l'equip final fent AND amb l'adreça IP; així:

```
143 . 45  .  1  . 23
255 . 255 . 255 . 0
143 . 45  .  1  . 0
```

D'aquí es pot extreure l'adreça de la subxarxa (els uns de la màscara – 143.45.1) i l'adreça de l'equip (els zeros de la màscara – 23). En aquest cas la xarxa constarà de 2^8 IP vàlides, com una classe C, de les quals 2^8 – 2 seran assignables a equips; cal recordar que les adreces especials de xarxa i de difusió no són assignables (143.45.1.0 i 143.45.1.255, respectivament). La representació d'aquesta subxarxa es fa amb la nomenclatura següent: 143.45.1.23/255.255.255.0.

Com es pot observar, això ens dóna un nivell més fi de divisió que ens simplificarà molt la gestió interna de xarxes; si una entitat disposa d'una classe B (146.43.0.0), internament l'entitat pot decidir subdividir les 65.536 adreces en diverses subxarxes, per exemple amb 256 subxarxes de 256 IP cada una: des de la 146.43.0.0/255.255.255.0 a la 146.43.255.0/255.255.255.0. Noteu que els valors de 255 i 0 per a l'adreça de xarxa són correctes, i no representen adreces de difusió i de xarxa, respectivament. Això ho podem saber gràcies a la màscara.

Una restricció no escrita però generalment adoptada a l'hora de definir les màscares és que tots els uns de la màscara han de ser consecutius; així, màscares com 255.145.0.0 es consideren invàlides, ja que traduïdes a format binari seria:

```
11111111 10010001 00000000 00000000
```

Mentre que altres com 255.255.128.0 són totalment correctes, ja que en format binari resulta:

```
11111111 11111111 11111110 00000000
```

Aquí tots els uns són consecutius, tot i no estar alineats a l'octet.

Aquesta restricció dels uns consecutius ens permet simplificar la representació de la màscara a un format més compacte; així, una altra manera d'indicar la separació entre la xarxa i l'equip és per mitjà d'un format que indica quants bits representen la xarxa; per exemple, 143.45.1.23/24 indica que la màquina 143.45.1.23 pertany a la xarxa 143.45.1.0/255.255.255.0, o el que és el mateix, que té 24 bits per a l'adreça de xarxa i 8 per a la dels equips.

D'altra banda, gràcies a la classificació per subxarxes els encaminadors tenen la feina més fàcil, ja que per a poder decidir la ruta que ha de prendre qualsevol datagrama és suficient de mirar la xarxa de destinació, i no és necessari comprovar tota l'adreça IP. L'adreça IP sencera, idealment, només la mirarà l'últim encaminador de la cadena, o sigui, el que estigui dins de la mateixa subxarxa a la qual pertanyi aquella destinació. Per a implementar aquest mecanisme els encaminadors basen la decisió d'encaminament en una política anomenada *longest prefix match*, que significa que de totes les rutes possibles sempre s'agafa la que té més bits coincidents amb la destinació del paquet.

Vegeu també

Vegeu més informació sobre la política del *longest prefix match* en l'apartat 5 d'aquest mòdul didàctic.

Exercicis

9. Indiqueu de les subxarxes següents i IP quantes IP assignables pot contenir la xarxa; expliqueu-ne el significat:

Adreça	IP assignables	Explicació
147.83.32.0/24		
1.23.167.23/32		
1.23.167.0/32		
147.83.32.0/16		

Solució de l'exercici 9

Adreça	IP assignables	Explicació
147.83.32.0/24	254	És una adreça de xarxa en què tenim 8 bits per als equips; sabent que l'adreça de difusió i la de xarxa no són assignables acabem amb el total de 2^8 – 2 adreces per assignar.
1.23.167.23/32	1	Adreça amb una subxarxa amb només un equip. No és útil en un cas real però és correcta.
1.23.167.0/32	1	Com no hi ha part d'equip, tot és xarxa, i el fet que l'últim octet sigui 0 no fa que la IP sigui una adreça de xarxa genèrica, sinó una específica, com el cas anterior.
147.83.32.0/16	1	Fa referència a l'equip 32.0 de la xarxa de classe B 147.83.0.0. Cal notar que no és una adreça de xarxa, ja que no tots els bits de fora de la màscara són 0; així, es tracta com una adreça d'equip.

10. De l'adreça de classe B 143.45.0.0/16, indiqueu quines subxarxes /20 es poden crear i quants equips conté cada una.

Solució de l'exercici 10

Xarxa	Subxarxa	Equip		
143.45	SSSS	HHHH.HHHHHHHH	143.45.0.0	Classe B
143.45	0000	HHHH.HHHHHHHH	143.45.0.0/20	$2^{12} - 2$ equips = 4094
143.45	0001	HHHH.HHHHHHHH	143.45.16.0/20	
143.45	0010	HHHH.HHHHHHHH	143.45.32.0/20	
143.45	0011	HHHH.HHHHHHHH	143.45.48.0/20	
…				
143.45	1111	HHHH.HHHHHHHH	143.45.240.0/20	

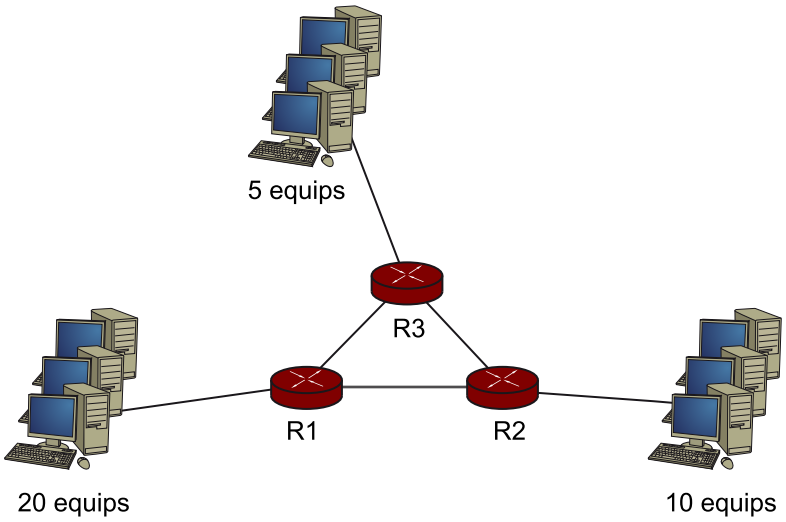
El CIDR generalment s'utilitza en conjunció amb el VLSM[21], tècnica que té per objectiu optimitzar la utilització de les adreces IP per mitjà d'una assignació intel·ligent de les màscares de xarxa. Aquesta assignació ara es farà tenint en compte el nombre de màquines de cada subxarxa, i s'assignaran màscares de mida ajustada a les necessitats particulars de cada una. VLSM es pot veure com la creació de subxarxes de les subxarxes.

[21] *VLSM* és la sigla de *variable length subnet mask*. En català, *màscara de subxarxa de mida variable*.

Exercicis

11. Donada la xarxa de la figura 7, ens proporcionen el rang d'adreces 147.83.85.0/24. Es demana que s'assignin rangs d'adreces a totes les subxarxes i als enllaços entre els encaminadors.

Figura 7



Solució de l'exercici 11

L'assignació d'adreces es pot fer seguint la política següent:

- Els 20 equips més l'encaminador necessiten un total de 5 bits ($2^5 = 32$)
- Els 10 + 1 equips en tenen suficient amb 4 bits ($2^4 = 16$)
- Els 5 + 1 equips en necessiten 3 ($2^3 = 8$), amb la qual cosa aquesta subxarxa no podrà créixer més.
- Finalment els enllaços punt a punt necessitaran 2 bits, ja que necessitem espai per a les adreces de difusió i de xarxa.

En resum, ens faran falta un prefix /27, un /28, un /29 i 3 prefixos /30, respectivament.

D'aquesta manera una possible assignació seria:

Els vint equips poden usar la 147.83.85.0/27, en què el darrer octet seria 000XXXXX, amb adreça de xarxa 147.83.85.0 i adreça de difusió 147.83.85.31.

Els deu equips disposaran de la 147.83.85.32/28, en què el darrer octet serà 0010XXXX, amb adreça de xarxa 147.83.85.32 i adreça de difusió 147.83.85.47.

En el cas dels cinc equips utilitzarem la subxarxa 147.83.85.48/29, en què el darrer octet serà 00110XXX, amb adreça de xarxa 147.83.85.48 i adreça de difusió 147.83.85.55.

Finalment els tres prefixos /30 (dels enllaços entre els encaminadors) es poden dividir amb el darrer octet 001110XX, 001111XX i 010000XX, respectivament. O sigui, 147.83.85.56/30, 147.83.85.60/30 i 147.83.85.64/30, amb adreces de xarxa 147.83.85.56, 147.83.85.60 i 147.83.85.64, i adreces de difusió 147.83.85.59, 147.83.85.63 i 147.83.85.67.

12. Quin problema té l'assignació d'adreces feta a l'exercici 11?

Solució de l'exercici 12

El problema d'aquesta assignació està determinat pel fet que s'han ajustat massa el nombre de bits per a cada subxarxa, i en el cas que creixi el nombre d'equips (especialment a la subxarxa de cinc equips) ens tocaria redimensionar la xarxa de nou, amb el cost que això representa.

3.1.4. Tipus de datagrames IP

IPv4 especifica tres tipus de trànsit clarament diferenciats dins de la xarxa: unidifusió, difusió i multidifusió.

El trànsit d'unidifusió és el més comú; la comunicació està formada per dos interlocutors que s'intercanvien informació, i sovint aquestes connexions són des d'un client cap a un servidor, que a la vegada pot tenir connexions unidifusió cap a altres clients.

El trànsit de difusió es basa a enviar la informació a tots els equips presents en una subxarxa. Com ja hem vist anteriorment, això es pot aconseguir enviant un paquet a una adreça que sigui l'adreça de xarxa i tot 1 a l'adreça de l'equip final; per exemple, per a la xarxa 126.76.31.0/24, l'adreça de difusió seria 126.76.31.255. Per a aconseguir que la difusió sigui realment eficient (que només s'enviï un paquet i el rebin tots els equips de la subxarxa), és necessari tenir suport del protocol d'enllaç de dades, tal com veurem en el mòdul següent.

Cal notar, però, que enviar trànsit de difusió normalment requereix algun privilegi a la xarxa (ser administrador); a més, els encaminadors en general no propaguen aquest tipus de trànsit per tal d'evitar problemes de seguretat, com per exemple atacs del tipus *denial of service* (DoS).

Finalment, el cas del trànsit de multidifusió es basa en el paradigma d'enviar informació des d'un sol origen cap a moltes destinacions a la vegada; la base del trànsit de multidifusió és que l'emissor no ha de tenir necessàriament coneixement de qui seran els seus receptors (en contra de la política d'unidifusió,

(22) *IANA* és la sigla d'*Internet Assigned Numbers Authority*.

que requereix conèixer els interlocutors). Això s'aconsegueix per mitjà del que es coneix com a grups de multidifusió. Com s'ha vist anteriorment, la IANA[22] ha reservat les adreces de tipus D a multidifusió; aquestes són les que van des del rang 224.0.0.0 fins al 239.0.0.0. D'aquest grup d'adreces n'hi ha unes quantes de reservades a grups de multidifusió coneguts com a permanents; la llista completa es pot trobar a Internet.

Així, si una estació concreta està interessada a rebre un contingut multidifusió, el que farà serà subscriure's al servei mitjançant el protocol IGMP[23], que especifica el format del paquet que s'ha de generar per tal de poder registrar-se en un grup i poder rebre'n el contingut. IGMP suporta dos tipus de paquets, els de pregunta i els de resposta. Normalment els de pregunta són uns paquets dirigits a tots els equips en què els que tenen sessions de multidifusió actives responen; així els encaminadors (que han de tenir suport per a multidifusió) poden construir el que es coneix com a *arbre multidifusió*, per a encaminar els paquets cap a les seves destinacions.

(23) *IGMP* és la sigla d'*Internet group management protocol.*

Vegeu també

Vegeu més informació sobre arbres multidifusió en l'apartat 4 d'aquest mòdul didàctic.

L'avantatge principal de la multidifusió és que la informació que s'envia, en comptes de replicar-se des de l'origen un cop per cada destinació, forma un arbre, de manera que es minimitza el nombre de còpies; un exemple d'això es pot veure més endavant en la figura 19b.

Exercicis

13. Un servidor de xat té en un moment donat un total de 80 clients connectats arreu del món. Indiqueu quin nombre i de quin tipus són les connexions que té obertes aquest servidor.

Solució de l'exercici 13

Atès que el xat és un protocol que utilitza TCP/IP, i que els clients, tot i parlar entre ells, passen sempre pel servidor, això és tracta del típic escenari amb 80 connexions unidifusió entre els 80 clients i el servidor.

14. Un administrador de la xarxa 147.83.0.0/16 vol enviar un paquet de difusió a la subxarxa 147.83.20.0/24. Indiqueu quina adreça de destinació tindria el paquet, quants paquets es generarien i a quantes màquines com a màxim podria arribar.

Solució de l'exercici 14

Atès que la subxarxa a la qual es vol enviar la difusió té 8 bits, això implica que es generaria un sol paquet amb adreça destinació 147.83.20.255, i que el rebrien com a màxim 255 – 2 = 253 estacions, ja que l'adreça 147.83.20.0 i la 147.83.20.255 estan reservades per a l'adreça de xarxa i la de difusió, respectivament.

3.1.5. El futur d'IPv4

Quan es va dissenyar IPv4 es creia que el seu gran nombre d'adreces IP (2^{32}) seria suficient per a poder suportar el gran creixement que s'esperava d'una xarxa com Internet. Cal recordar que Internet va entrar en funcionament el 1969 amb el nom d'ARPANet, un projecte subvencionat pel Departament de

Defensa dels Estats Units. Això va provocar que, quan Internet es va desplegar al cap d'uns anys a la xarxa comercial, el repartiment d'adreces no fos equitatiu, ja que les grans empreses americanes es van poder adjudicar una gran quantitat d'adreces de classe A, i van deixar països com la Xina i d'altres que s'han desenvolupament posteriorment amb moltes menys adreces de les necessàries. Com a referència, els EUA tenen uns 1.500 milions d'adreces assignades, mentre que la Xina, amb una població molt més nombrosa, només en disposa aproximadament de 200 milions. Per tenir una idea, Espanya en té assignades actualment entorn de 22 milions.

Amb aquest paradigma, molt aviat es veu que amb l'actual política per al repartiment d'adreces, en molt poc temps no quedaran adreces IPv4 disponibles per assignar, fet que implicarà inevitablement que Internet no podrà créixer més.

Per tal de minimitzar aquest problema es va dissenyar el NAT, com ja hem vist, que permet utilitzar adreces privades per a accedir a la Xarxa amb una sola IP pública. Actualment, països com la Xina o l'Índia estan fent un ús intensiu del NAT per la manca d'adreces disponibles.

Com aquesta solució no és escalable i comporta una sèrie molt important de problemes als proveïdors de serveis, es va arribar a la conclusió que els 32 bits d'adreçament del protocol IPv4 eren insuficients, i per això es va dissenyar el protocol IPv6, com veurem a continuació. Per motius econòmics IPv6 encara no ha estat implantat, i si la demanda d'adreces IPv4 segueix al ritme actual, es preveu que la IANA assignarà el darrer rang d'adreces IPv4 a la meitat del 2011, i que les autoritats regionals esgotaran les que tenen pendents per assignar el 2012, fet que de ben segur forçarà molts països a adoptar prematurament IPv6. En aquest sentit, països com el Japó, la Xina, l'Índia i alguns d'Amèrica del Sud ja han adoptat el protocol i utilitzen algunes tècniques, com veurem més endavant, que permeten la interoperabilitat dels dos protocols.

3.2. IPv6

La mancança d'adreces IPv4 vista anteriorment va incentivar el disseny d'un nou protocol de xarxa, IPv6. Actualment IPv6 està totalment desenvolupat, tot i que encara no és possible utilitzar-lo dins la xarxa comercial, ja que els operadors encara no han preparat els seus equips i tampoc no han fet el repartiment d'adreces als seus usuaris. Això i la dificultat d'implantar progressivament aquesta nova versió en substitució de l'anterior és el que està retardant la seva incorporació a escala comercial.

Aquest subapartat descriu breument aquest protocol, se'n remarquen les diferències amb la versió anterior, i les novetats que incorpora. Per acabar es descriuen els principals problemes que hi ha per a la migració d'IPv4 a IPv6.

3.2.1. Motivació

Inicialment, a l'hora de dissenyar el protocol, es va pensar que no era necessari crear un protocol sencer, i només fent una adaptació d'IPv4 hauria de ser suficient. Però ben aviat es va veure que per a poder gaudir de bones optimitzacions, comparat amb la versió anterior, caldrien bastants més canvis. Així, es va optar per un disseny que té poc en comú amb la versió anterior.

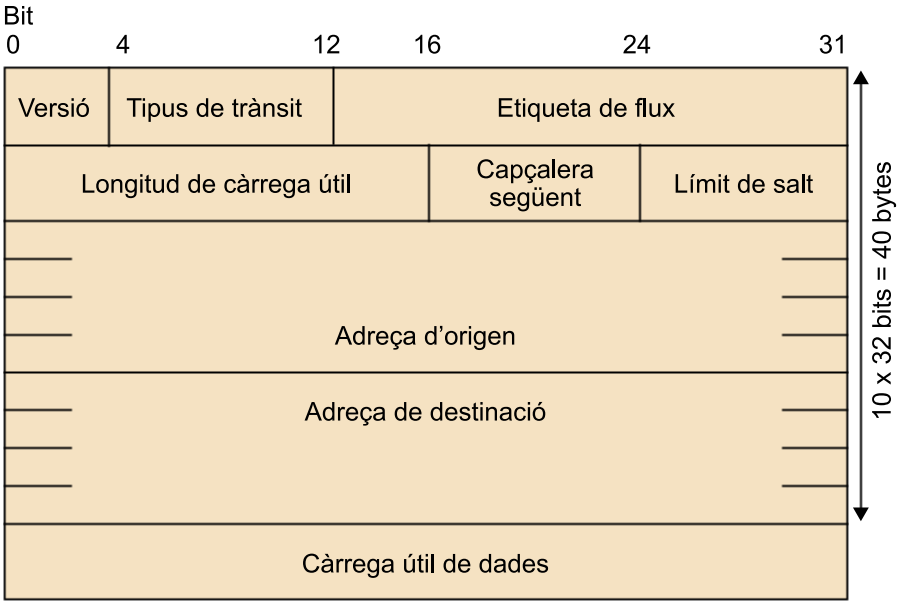
El motiu principal que va portar a plantejar-se una nova versió del protocol era el limitat rang d'adreces que permet IPv4, que encara que pugui semblar molt elevat, es va veure que seria clarament insuficient per a la demanda del mercat en un futur. IPv6 soluciona aquest problema proposant un camp d'adreces de 128 bits.

Sobretot, l'aparició els darrers anys d'una gran quantitat de dispositius mòbils que volen formar part de la gran xarxa que és Internet, ha provocat ràpidament que els 32 bits d'adreçament IPv4 siguin insuficients; tant és així, que si tots aquests dispositius es volguessin connectar de manera simultània a la Xarxa probablement els operadors tindrien problemes per la manca d'adreces IPv4. Per veure aquest problema només cal pensar quants telèfons mòbils hi ha actualment només a l'Estat espanyol, uns 44 milions, respecte al nombre d'IP que hi ha assignades actualment al país, uns 22 milions. Això sense considerar els usuaris que es connecten des de les seves llars. Es podria pensar que el problema es podria minimitzar amb la utilització de NAT, però a la llarga això pot representar un greu problema de rendiment als encaminadors, per haver de mantenir les taules de traducció d'adreces de milions de connexions a la vegada. A sobre, com més va hi ha més petites i mitjanes empreses que volen oferir als seus clients una sèrie de serveis que precisen una connexió permanent a la Xarxa, amb la conseqüent despesa d'adreces i la impossibilitat d'utilitzar el NAT massivament.

3.2.2. Capçalera IPv6

La capçalera IPv6 té una longitud fixa de 40 octets (vegeu la figura 8), i consta dels camps següents:

Figura 8. Capçalera IPv6



- ***Version*** (4 bits): indica la versió del protocol que conté el paquet. Aquest camp té el mateix significat que el de la versió IPv4, però ara amb el valor 0 × 06.

- ***Traffic class*** (8 bits): aquest camp classifica un paquet dins d'un tipus de trànsit determinat; conceptualment és l'equivalent al TOS d'IPv4.

- ***Flow label*** (20 bits): serveix per a etiquetar un conjunt de paquets que tinguin les mateixes característiques; servirà per a poder oferir qualitat de servei.

- ***Payload length*** (16 bits): longitud del *payload* del paquet, o sigui, el paquet sense la capçalera IP. La mida està representada en octets.

- ***Next header*** (8 bits): aquest camp és una gran innovació d'IPv6 respecte a IPv4, ja que permet tenir una capçalera bàsica de mida fixa. Aquest camp indica la posició a la qual es pot trobar la capçalera següent, i així estalvia temps de procés als encaminadors intermedis en no haver-hi opcions.

- ***Hop limit*** (8 bits): aquest camp és equivalent al TTL d'IPv4 però directament compta salts i no temps.

- ***Source address*** (128 bits): adreça de l'equip final que ha originat el paquet.

- ***Destination address*** (128 bits): adreça de l'equip final al qual està destinat el paquet.

De les dades de la capçalera, es pot observar que la diferència més directa que hi ha entre tots dos protocols és la longitud de les adreces IP; on IPv4 té 32 bits, IPv6 passa a tenir-ne 128. Aquest augment en l'espai d'adreçament permet que

el rang d'adreces de la Xarxa passi de 2^{32} a 2^{128} adreces possibles. Com que ara tenim molts més bits per a l'adreça, la forma d'especificar adreces IPv6 es fa amb "la notació dels dos punts". Una adreça IPv6 es representa amb blocs de 16 bits representats en hexadecimal i separats pel símbol ":". Per exemple: 2001:0DB8:0000:0000:0319:8A2E:0370:7348.

Una simplificació d'aquesta notació es pot aplicar en el cas que una adreça tingui molts 0 consecutius; la manera abreviada de representar-la és utilitzant "::". Així, la forma compacta de representar l'adreça anterior seria 2001:0DB8::0319:8A2E:0370:7348.

Una altra diferència notable entre IPv4 i IPv6 és la jerarquització de les adreces. L'assignació d'adreces IPv4 es va fer, al seu temps, d'una manera molt anàrquica, ja que no s'esperava que el creixement d'Internet fos tan espectacular. Actualment cada corporació o cada operadora de telefonia té rangs d'adreces molt dispersos i mal dimensionats, cosa que fa extremadament difícil la gestió de les adreces disponibles, l'assignació de les noves i l'encaminament global. Per això IPv6 el que ha fet ha estat jerarquitzar d'una manera més intel·ligent el repartiment de les seves adreces, de manera que cada país, operador o ISP disposa d'un rang concret amb un nombre d'adreces proporcional a la seva possible utilització de la Xarxa. Independentment de la millora d'aquesta jerarquia pel que fa a la localització geogràfica, el fet de separar d'aquesta manera les adreces permet assignar-ne de noves d'una manera molt més senzilla que fins ara.

De manera semblant a IPv4, es pot identificar quin tipus d'adreça és només amb el prefix de l'adreça IPv6, com indica la taula següent:

Assignació d'adreces	
Prefix	**Espai d'assignació**
0000::/8	Reservat. Les adreces de *loopback* i les adreces amb integració d'IPv4 surten d'aquest prefix.
0100::/8	Reservat.
0200::/7	Reservat.
0400::/6	Reservat.
0800::/5	Reservat.
1000::/4	Reservat.
2000::/3	Adreça unidifusió global. D'aquí surt el rang d'adreces que es repartiran als usuaris. Hi ha 2^{125} adreces disponibles.
4000::/3	Reservat.
6000::/3	Reservat.
8000::/3	Reservat.

Assignació d'adreces	
Prefix	**Espai d'assignació**
A000::/3	Reservat.
C000::/3	Reservat.
E000::/4	Reservat.
F000::/5	Reservat.
F800::/6	Reservat.
FC00::/7	Adreça unidifusió local única.
FE00::/9	Reservat.
FE80::/10	Adreça d'enllaç local unidifusió.
FEC0::/10	Reservat.
FF00::/8	Adreces multidifusió.

Un fet molt interessant que es va considerar per a fer aquesta assignació d'adreces és que dóna la possibilitat de representar adreces de diverses tecnologies incrustades dins la nova versió del protocol. D'aquesta manera es poden representar adreces IPv4, i fins i tot adreces de maquinari de l'enllaç de dades (com per exemple Ethernet).

El gran avantatge d'inserir altres tipus d'adreces directament a la IPv6 és que tenen un prefix assignat; així, per exemple, per a tenir una adreça Ethernet d'un equip dins d'una IPv6 el prefix LAN és FE80::/10. Per tant, si l'adreça de la targeta Ethernet és 00:90:F5:0C:0F:ED, aleshores l'adreça IPv6 queda fE80::0090:F50C:0FED. Com es pot observar, el procés també es pot fer a la inversa: quan arriba un paquet amb el prefix de xarxa FE80 ja es pot suposar que es tracta d'una adreça local[24] i se'n pot extreure l'adreça de maquinari fàcilment. A més a més, amb aquest mecanisme qualsevol interfície de xarxa es pot configurar de manera automàtica i autònoma.

[24] En anglès, *link-local*.

També cal remarcar que IPv6, a part de tenir adreces unidifusió, difusió i multidifusió com IPv4, afegeix suport per a un quart tipus, que són les adreces de servei.

Les adreces de servei són una gran innovació d'IPv6, sobretot perquè aprofiten les adreces unidifusió ja existents. Així, una adreça unidifusió esdevé de servei des del moment que una mateixa IPv6 s'assigna a més d'una interfície (incloent-hi equips diferents). La idea al darrere d'aquesta implementació és que respongui les peticions a un servei concret l'estació més propera. Així, per exemple, imaginem dos servidors web amb la mateixa IPv6, per exemple, 2001:0DB8::0319:8A2E:0370:7348; quan la xarxa rebi un paquet dirigit cap a

aquesta IPv6, l'enviarà a totes dues estacions, i la primera que respongui serà la que estigui més prop de l'equip que fa la petició. Actualment, per complexitats amb la implementació d'aquest tipus d'adreces, només s'utilitzen per a encaminadors. Així, una subxarxa pot tenir més d'un encaminador per a sortir a Internet usant el mateix prefix, i cada equip fa servir el que està més proper a l'estació, i aconsegueix de manera senzilla un sistema de balanceig de càrrega.

Una altra innovació rellevant que incorpora IPv6 és la utilització molt més intensiva del trànsit de multidifusió dins de les xarxes locals; tant és així que els equips, per defecte, escolten adreces multidifusió amb el prefix FF02::1:FF00:0000/104, com veurem en seccions posteriors, per tal d'evitar la generació de trànsit de difusió que afecta tots els equips de la subxarxa, i no sempre és desitjable.

Altres millores menors que introdueix aquest protocol són:

- **Mecanisme d'opcions ampliat**: les opcions formen part d'una capçalera col·locada entre la capçalera IP pròpiament dita i la capçalera de la capa de transport. Aquesta manera de posar les opcions permet una gestió més simple de les capçaleres per part dels dispositius que han de tractar el paquet fins que no arriba a la seva destinació, i permet un sistema més simple i flexible.

- **Adreces d'autoconfiguració**: l'assignació dinàmica d'adreces ha estat substancialment millorada respecte al seu predecessor. Un dels motius principals d'això és el fet que es pot afegir l'adreça de maquinari dins l'adreça IPv6, i així només amb un prefix donat pel dispositiu d'encaminament més proper i amb l'adreça de maquinari es garanteix una adreça única a escala mundial, sempre que s'utilitzi el prefix assignat per l'operador a l'hora de generar l'adreça.

- **Facilitat per a l'assignació de recursos**: el que amb IPv4 era el Type of Service ara s'anomena *traffic class*, tot i que ara es té la possibilitat de marcar fluxos individuals, cosa que dóna molta més flexibilitat a l'hora de marcar trànsit prioritari.

- **Capacitats de seguretat**: com que la seguretat, avui dia, és un tema molt important IPv6 inclou característiques d'autenticació i privacitat. Per defecte IPv6 inclou funcionalitats natives per a la creació de xarxes privades virtuals (VPN) per mitjà d'IPsec, protocol de xifratge de les dades en temps real, que amb IPv4 era opcional.

Exercicis

15. Un PC té una targeta Ethernet amb MAC 34:27:A4:6F:AE:53. L'operador li proporciona el prefix 2001:0A54:0039::/48. Indiqueu l'adreça local i l'adreça d'autoconfiguració d'aquest equip.

Solució de l'exercici 15

L'adreça local estarà determinada pel prefix *link-local*, i així l'adreça serà fE80::3427:A46F:AE53. L'adreça d'autoconfiguració surt del prefix i de la MAC; per tant: 2001:0A54:0039::3427:A46F:AE53.

3.2.3. Problemes de la migració a IPv6

Un dels motius principals pels quals encara és treballa amb IPv4 és la dificultat que comporta la migració al nou protocol. La incompatibilitat de les adreces i de les capçaleres de tots dos protocols fan que l'actualització a la nova versió no sigui fàcil. També cal tenir en compte que les aplicacions existents només suporten el sistema d'adreces d'IPv4, i per a acceptar les noves adreces s'ha de canviar el codi de l'aplicació, i també totes les crides al sistema d'accés a la xarxa.

A part del nivell d'aplicació, hi ha un altre problema molt greu. Atesa la gran diversitat de xarxes que formen Internet, hi ha equips molt diversos en funcionament, i no tots aquests equips de comunicacions tenen suport per al nou protocol; per tant, s'ha d'actualitzar el sistema operatiu dels encaminadors de la xarxa, amb la conseqüent despesa econòmica i de temps que això representa, cosa que moltes de les empreses no estan disposades a invertir (especialment les grans corporacions americanes, que són les que tenen adreces suficients).

Finalment, a causa de la gran utilització que té actualment Internet, és un gran problema haver de parar totes les xarxes per a fer la migració. Així, el problema que es troba és l'enorme despesa econòmica per a les empreses que controlen totes les seves transaccions per mitjà de la xarxa, cosa que força a fer la migració de manera progressiva, transparent per als usuaris i sense deixar d'oferir els serveis disponibles en cap moment.

3.2.4. Mecanismes per a assistir la transició

Una migració entre dos protocols quan un s'està utilitzant massivament és extremadament complexa. Com hem vist, les raons normalment són econòmiques, ja que la gran majoria de les aplicacions actualment només suporten IPv4, i migrar-les a la nova versió no sempre és senzill (per exemple, aplicacions bancàries). D'altra banda, tot l'equipament de maquinari que forma la columna vertebral[25] de la Xarxa, si bé està preparat per a suportar IPv6, no sempre té la configuració correcta, ni l'assignació d'adreces feta. Amb tot, s'espera que hi hagi una fase de coexistència dels dos protocols. De totes maneres, tot i la coexistència hi ha escenaris que obliguen a dissenyar un pla de migració controlat.

[25] En anglès, *backbone*.

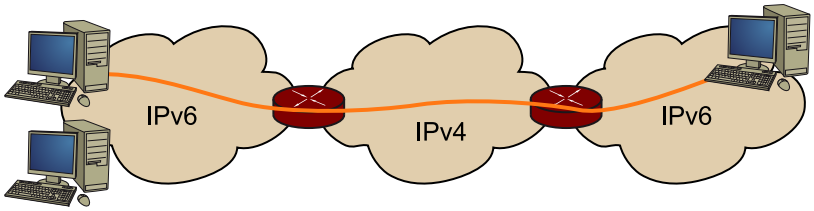
Així, els diferents mecanismes de transició es poden dividir en dos grans grups: mecanismes bàsics i mecanismes per a la interconnexió d'illes.

De mecanismes bàsics se'n poden distingir dos: el conegut com a *dual-stack*, en què els equips utilitzen simultàniament els dos protocols i es connecten amb el que més s'ajusti a les necessitats del moment, i el de *tunneling*, en què dos equips amb *dual-stack*, creen un túnel IPv4 entre ells, i per dins del túnel es comuniquen amb IPv6.

Cal notar que un túnel és aquell mecanisme pel qual s'encapsulen dos protocols de xarxa dins d'un mateix datagrama, i així hi ha dues capçaleres de xarxa consecutives del nivell de xarxa. IPv4 suporta el mecanisme de túnel per mitjà d'un valor especial al camp *Protocol* que es troba a la capçalera.

Pel que fa als mecanismes per a connectar illes, pretenen resoldre el cas en què diverses màquines interconnectades per mitjà d'IPv6 (illa IPv6) es volen connectar amb una altra illa IPv6, però pel camí hi ha una illa IPv4, tal com mostra la figura 9.

Figura 9. Xarxes IPv4 i xarxes IPv6



Aquests mecanismes també estan basats principalment en túnels. Se'n poden distingir els tipus següents:

1) Túnels configurats: els extrems dels túnels entre les illes es configuren de manera manual entre les dues xarxes. Els extrems han de ser *dual-stack*.

2) Túnels automàtics: s'utilitzen adreces IPv4 mapades dins d'IPv6 amb el prefix reservat, que és el ::/96; per exemple, ::195.123.57.93, que l'encaminador converteix a l'adreça IPv4, i a l'altre extrem es torna a convertir a la IPv6. Els extrems no s'adonen del canvi i es poden comunicar amb IPv6 sense problemes.

3) Túnel Broker: s'utilitza un gestor[26], que indica al client un guió[27] per tal de fer el túnel de manera automàtica. El client ha de ser *dual-stack*, ja que la petició al gestor va amb IPv4, que contesta amb un guió que permet connectar a un *tunnel-server* (que també és *dual-stack*), que permet connectar-se a la xarxa IPv6.

[26] En anglès, *broker*.

[27] En anglès, *script*.

4) 6to4: s'assigna una adreça IPv4 compatible amb el prefix IPv6 als encaminadors, que fan un túnel. Per exemple, per a la xarxa 2001:d002:0507::/48 l'encaminador tindria l'adreça 208.2.5.7 (que surt de d002:0507). A l'altre extrem es faria l'operació anàloga i s'establiria el túnel.

3.3. Protocols de suport a IP

Tant IPv4 com IPv6 són protocols de xarxa, però tots dos necessiten suport d'altres protocols de la mateixa capa per a poder dur a terme certes funcionalitats que seria complex aconseguir d'una altra manera. Tot i que, ateses les diferències estructurals entre els dos protocols, molts dels serveis són diferents, s'engloben tots en aquesta secció per simplificar-ne la comprensió, ja que si bé els protocols divergeixen, la seva funcionalitat sovint és molt similar.

3.3.1. ICMP

IP és un protocol no orientat a connexió que té per objectiu l'enviament d'informació independentment de la tecnologia de nivells inferiors utilitzada. Això proporciona un entorn ideal per a poder comprovar l'estat de la xarxa, o enviar informació de control en cas que hi hagi problemes a la xarxa. Per exemple, quan un datagrama no pot arribar a la seva destinació, o bé si hi ha congestió a un enllaç, o bé si a un paquet se li ha expirat el TTL. Totes aquestes situacions requereixen algun protocol que, treballant a la mateixa capa que IP, permeti avisar de manera automàtica de qualsevol d'aquests esdeveniments. Per això es va dissenyar el protocol ICMP.

El protocol ICMP[28] és l'encarregat d'enviar missatges de control (i d'error) entre els diferents equips que formen la xarxa.

[28] *ICMP* és la sigla d'*Internet Control Message Protocol.*

La taula següent mostra tots els tipus de missatges existents amb ICMP.

Els missatges ICMP tenen diverses utilitats: des d'informar d'errors fins a depurar l'estat de la xarxa. Una de les eines més utilitzades per a poder veure si un equip està connectat a la xarxa és *ping*. Una altra funcionalitat és possible gràcies al camp TTL de la capçalera IP, que ajuda a fer una altra tasca de depuració mitjançant una eina anomenada **traceroute**; aquesta eina ens permet descobrir els encaminadors intermedis entre l'origen i la destinació dels datagrames. Per a aconseguir saber quins encaminadors travessa un datagrama, el *traceroute* envia paquets IP consecutius amb un TTL d'1, un altre de 2, un altre de 3, i així successivament fins a arribar a la destinació. L'efecte d'això és que el primer encaminador, en rebre un paquet amb TTL = 1, el decrementa, i en ser 0, el descarta i envia de tornada un paquet ICMP de *TTL expired*. Ara l'aplicació només cal que miri qui ha enviat aquest paquet (IP origen) per a

saber de quin encaminador es tracta. Per descomptat, amb TTL = 2 succeirà el mateix amb el segon encaminador, i després amb el tercer i així successivament fins a la destinació.

Descripció dels diferents missatges ICMP	
Missatge	**Descripció**
Destination unreachable	Indica que no es pot arribar a la destinació. Aquest missatge té un camp de codi que indica si és culpa de la xarxa, de l'equip en particular o bé del port. També distingeix si no es pot arribar a la destinació o bé si la destinació és desconeguda.
Echo request / *Echo reply*	Aquests dos missatges són el de petició i el de resposta; quan una estació rep un *echo request*, ha de respondre amb un *echo reply* si està activa. En general és aconsellable que tots els equips responguin aquestes peticions, tot i que per seguretat molts cops es filtren els missatges. L'aplicació per excel·lència que utilitza els *echo request*/*reply* és el *ping*.
Source quench	Aquest paquet serveix per a regular; indica que aquell enllaç està patint congestió. Actualment aquest tipus de paquet no s'utilitza perquè el control es fa majorment a la capa de transport.
Router advertisement	Aquest paquet d'ICMP s'envia a una adreça multidifusió en què tots els encaminadors escolten per defecte. Amb aquest missatge és possible descobrir automàticament l'existència de nous encaminadors a la xarxa.
Router discovery	És un paquet complementari al de *router advertisement*. Però en aquest cas és un encaminador que acaba d'entrar a una xarxa qui pregunta quins altres encaminadors hi ha a la xarxa.
TTL expired	Quan el TTL d'un paquet arriba a 0 aquest es descarta, i l'encaminador que l'ha descartat genera un paquet *TTL expired* a l'origen del paquet descartat.
IP header bad	En el cas que es detecti un error de la suma de comprovació a la capçalera d'un paquet IP, aquest es descarta i s'avisa a l'origen amb aquest paquet ICMP.

3.3.2. ARP

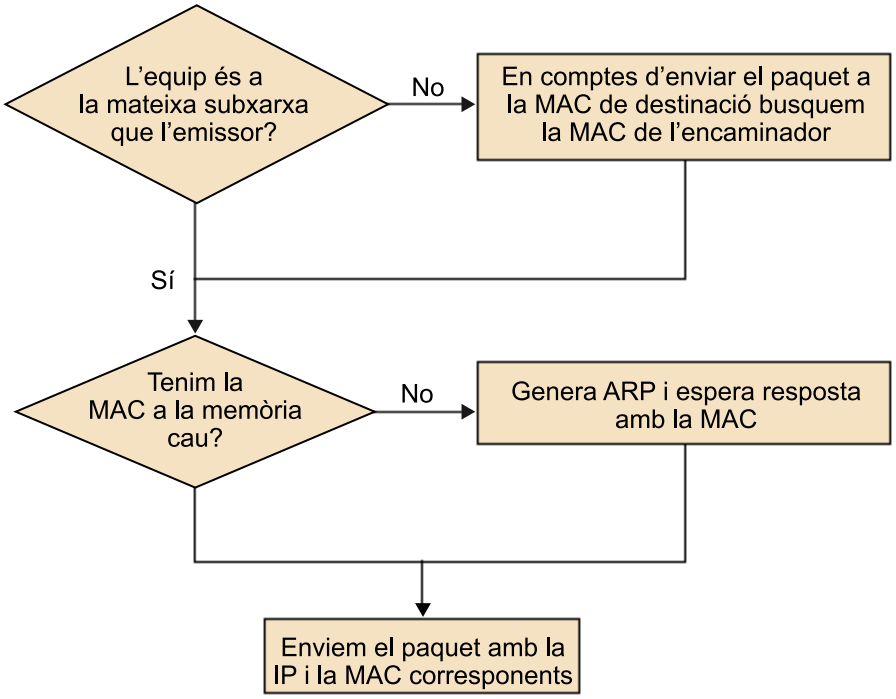
A l'inici del mòdul hem vist que per a enviar un datagrama IP a una estació de la mateixa xarxa que l'emissor era necessari descobrir quina adreça del nivell de l'enllaç de dades té aquesta estació, ja que les tecnologies de capes inferiors no entenen què és una adreça IP.

Així doncs, perquè IP funcioni necessita interactuar amb les capes inferiors i descobrir de manera automàtica quina és l'adreça d'enllaç de dades a la qual respon un equip per a poder-se intercanviar informació, i per això va ser dissenyat el protocol ARP[29] específicament per a IPv4.

[29] *ARP* és la sigla d'*address resolution protocol*.

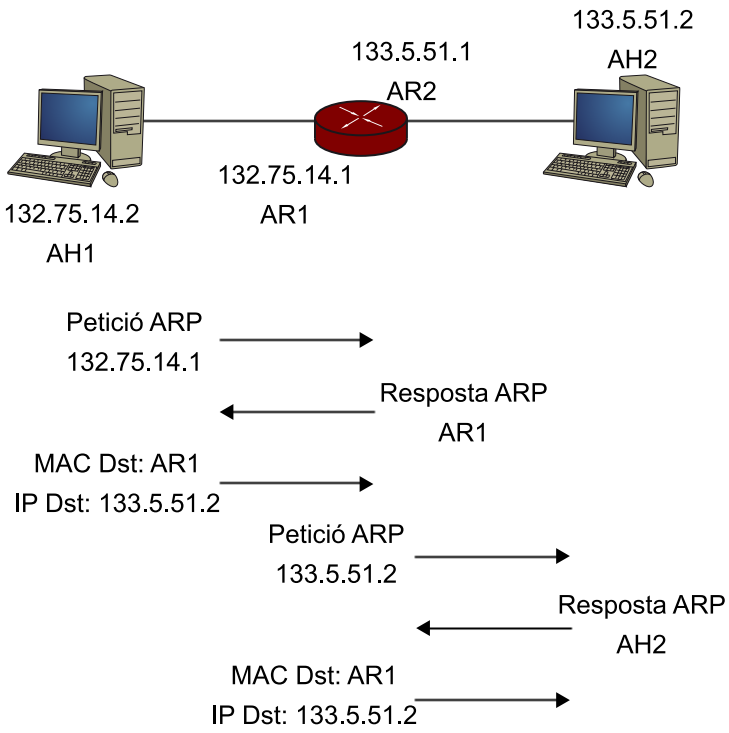
Per a aconseguir el descobriment de l'adreça maquinari d'un equip a partir de la seva IP ARP se serveix de la funcionalitat que ens proporciona la capa d'enllaç de dades per a enviar paquets de difusió. Aleshores el procés per a aconseguir l'adreça maquinari (anomenada *MAC*, com veurem en el mòdul 4) i poder enviar el paquet, és tal com es pot veure al diagrama de la figura 10.

Figura 10. Diagrama de seqüència per a l'enviament d'un paquet IP



El procés d'ARP, il·lustrat en la figura 11 amb un exemple, segueix una sèrie de passos per tal de descobrir l'adreça. Primer el que s'ha d'aconseguir és l'adreça MAC del pròxim salt, i per això s'envia un paquet ARP a l'adreça de difusió de la nostra xarxa local. Aquest ARP buscarà la IP de destinació o bé la de l'encaminador, depenent de si l'estació forma part de la subxarxa o no. Un cop s'obté l'adreça MAC, es construeix un paquet que té com a adreça MAC destinació l'obtinguda per ARP i com a IP destinació l'original (o sigui, en el cas que s'enviï el paquet a l'encaminador, aquesta IP serà la de la destinació final, no la de l'encaminador).

Figura 11. Exemple de petició per ARP



Hem vist que l'ARP ens serveix per a poder descobrir quina adreça MAC correspon a una IP, i també hem vist la importància d'aquesta operació. Per completesa, ARP té una variant anomenada RARP que ens permet fer l'operació inversa, o sigui, des d'una adreça MAC poder esbrinar a quina IP correspon. RARP no és ben bé un protocol de la capa de xarxa, ja que inclou moltes funcionalitats de la capa d'enllaç de dades, però atesa l'estreta relació amb ARP s'acostumen a considerar conjuntament. RARP ja no s'utilitza, ja que hi ha protocols com BOOTP i DHCP que ens ofereixen aquesta i més funcionalitats, com veurem a continuació.

3.3.3. NDP

ARP és un protocol que va ser dissenyat específicament per a IPv4, i amb els avenços de les xarxes fins avui, ha provat que és insuficient per a segons quins serveis. Per això, amb l'aparició d'IPv6, es va decidir que calia un protocol més complet. Així va aparèixer NDP[30].

[30] *NDP* és la sigla de *network discovery protocol.*

NDP és un protocol que permet descobrir els veïns existents en una xarxa local. El mode d'operació és molt similar al que utilitzàvem amb IPv4: s'envien *neighbour solicitations* i es reben *neighbour advertisements*. Però, la gran diferència amb ARP és que s'utilitza trànsit multidifusió en comptes de difusió. I NDP forma part d'un protocol més gran anomenat *ICMPv6*, que és l'extensió a IPv6 del protocol ICMP.

Com ja hem vist, un node IPv6 quan es dóna d'alta es posa a escoltar un conjunt d'adreces multidifusió, i una d'aquestes és la de *solicited-node*. Per a automatitzar aquest procediment, l'adreça multidifusió *solicited-node* d'una estació es construeix de la manera següent: s'agafen els darrers tres octets de l'adreça unidifusió i s'hi afegeix al principi el prefix multidifusió FF02::1:FF00:0000/104. Per exemple, l'adreça multidifusió solicited-node per a 2001:630:1310:FFE1:02C0:4EA5:2161:AB39 seria FF02::1:FF61:AB39. Aleshores l'estació es posa a escoltar el grup multidifusió per a respondre amb l'adreça maquinari de l'equip a qui va dirigida la petició.

Vegeu també

Vegeu l'IPv6 en el subapartat 3.2 d'aquest mòdul didàctic.

Això ens dóna la versatilitat que no tots els nodes reben els anuncis, i així en el cas que no vagin dirigits cap a ells ja ni els arriben a veure, amb la reducció en l'ús de recursos que representa això.

3.3.4. BOOTP

El protocol BOOTP[31] s'utilitza per a obtenir de manera automàtica una adreça IPv4 des d'un servidor que està situat a la mateixa subxarxa que el client. Normalment s'utilitza en equips finals durant el procés d'engegada, cosa que permet que la imatge del nucli per a arrencar el sistema s'obtingui a través de la xarxa.

(31) *BOOTP* és la sigla de *bootstrap protocol*.

El funcionament del protocol BOOTP és molt simple: el client envia un paquet de difusió del tipus *bootrequest*, indicant l'adreça maquinari i, si la sap, la seva adreça IP. El servidor li contesta amb un *bootreply* amb les dades necessàries i un enllaç a la imatge del nucli preconfigurada per a aquell equip.

Avui dia aquest protocol està en relatiu desús, ja que han aparegut alternatives més modernes, com ara el PXE[32], proposat per Intel. El PXE dóna més versatilitat a l'hora de configurar quin sistema operatiu ha d'arrencar.

(32) *PXE* és la sigla de *preboot execution environment*.

3.3.5. DHCP

El protocol següent de xarxa que veurem (DHCP[33]) no és realment un protocol de xarxa, sinó un protocol d'aplicació. De totes maneres, atès que s'utilitza per a configurar la xarxa, s'explica en aquesta secció. El protocol DHCP és el que utilitzen els dispositius per a obtenir informació de la configuració dels paràmetres de xarxa per a un equip IPv4 de manera automàtica.

(33) *DHCP* és la sigla de *dynamic host configuration protocol*.

L'administrador de la xarxa configura un prefix de xarxa, juntament amb un subrang d'adreces destinades a autoconfiguració (aquest rang d'adreces s'anomena *pool*). Quan es rep una petició, el protocol comprova si el client està autoritzat, i si ho està se li assigna una IP preconfigurada, que s'obté d'una base de dades a partir de l'adreça maquinari de l'equip, o bé una a l'atzar del *pool* si l'adreça maquinari no es troba. Aquesta cessió d'IP està controlada per un temporitzador, i quan aquest temporitzador expira i no s'ha rebut cap notícia

del client es torna la IP al *pool* d'adreces lliures. Per a evitar això, el protocol implementa un sistema de *keep-alive*, que va enviant renovacions d'ús de la IP al servidor per a evitar que caduquin. Per a fer totes aquestes tasques DHCP utilitza UDP per a enviar la informació.

Entrant en una mica més de detall, un client quan dóna d'alta una interfície enviarà un *DHCP discovery*, que és un paquet difusió per a descobrir servidors DHCP. El servidor, quan veu el paquet, comprova la validesa del client (base de dades de MAC) i envia un *DHCP offer* amb la seva IP. Això el client ho respon directament amb un *DHCP request*, i finalment el servidor ho accepta amb un *DHCP acknowledgement*, que conté la durada de la IP i la configuració específica que el client hagi demanat en el *DHCP request*. Per exemple: encaminador per defecte, servidor de DNS, etc.

3.3.6. DNS

Una funcionalitat molt important, i molt utilitzada actualment, és el DNS[34]. Aquest servei es va crear per a simplificar la identificació dels diferents equips de la Xarxa. Fins ara hem vist que un node de la Xarxa s'identifica per mitjà de la seva adreça IP, però com es pot observar, un usuari pot tenir dificultats per a recordar adreces IP i associar-les al servei que proporciona.

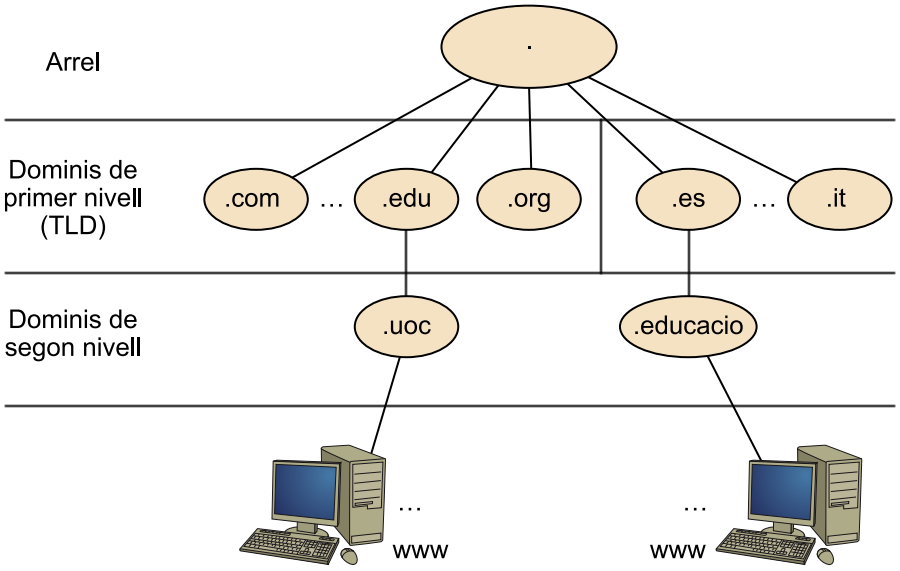
(34) *DNS* és la sigla de *domain name service*. En català, servei de noms de domini.

El DNS ens permet fer un mapatge d'una adreça IP a un nom fàcil de recordar per a un humà. Així, cada equip o encaminador que tingui una IP pot tenir un nom assignat a aquesta IP, i el servei de DNS ens permetrà resoldre la IP que pertanyi a un nom i viceversa.

DNS és una base de dades distribuïda que utilitza TCP o UDP al port 53 per a oferir el servei. Per tal de simplificar-ne la gestió dins d'una xarxa tan gran com Internet, utilitza una aproximació jeràrquica que permet distribuir la càrrega entre diversos nodes. Concretament, el DNS disposa del que es coneix com a node arrel, que és des d'on es deriven tots els noms assignats. Aquest node arrel està compost actualment per 13 servidors i generalment s'identifica amb un ".", i d'aquest pengen els dominis d'alt nivell (TLD[35]). De TLD n'hi ha de dos tipus: els genèrics (.com, .edu...) i els geogràfics (.es, .uk, .it), tal com es mostra a la figura 12. A la seva vegada dels TLD deleguen la resolució de noms als dominis de segon nivell, dels quals a la vegada surten els de tercer nivell, i així successivament fins a arribar als equips finals. A la figura 12 es mostren dos exemples: www.uoc.edu i www.educacion.es.

(35) TLD és la sigla de *top level domains*.

Figura 12. Jerarquia del DNS



Els exemples anteriors ens permeten veure que el TLD .edu (pertanyent a les universitats a escala mundial) té registrada una universitat .uoc, de la qual penja una màquina (www), que per conveni normalment fa referència a un servidor web. La potència d'aquesta solució és que quan hi ha una consulta al DNS aquesta es fa de manera recursiva. Així, per al cas de www.uoc.edu, un dels servidors arrel passarà la pregunta al domini .edu (que està format per diversos servidors), que enviarà la pregunta recursivament al domini uoc. Aquest finalment consultarà a la seva base de dades i retornarà l'adreça IP que correspon a la consulta. Quan el client rebi la resposta ja podrà establir una connexió amb aquesta adreça.

Un tipus de TLD que no s'ha comentat fins ara és el .arpa. Aquest domini és el que s'utilitza per a fer el que es coneix com la resolució inversa, o sigui, obtenir un nom a partir d'una adreça IP. Per a fer això es fa una consulta al DNS invertint la IP de la consulta i posant-li el nom de domini in-addr.arpa. Així, per a demanar el nom associat a 193.45.15.48, es construiria una consulta de la forma 48.15.45.193.in-addr.arpa., que es resoldria de manera anàloga a la resolució directa de noms vista fins ara.

4. Algorismes i mecanismes d'encaminament

Un algorisme d'encaminament és aquell procés que permet l'enviament d'un datagrama generat des d'un node origen de la xarxa a qualsevol altre node d'aquesta xarxa. Aquests dos nodes generalment estaran connectats a encaminadors diferents, per la qual cosa el datagrama haurà d'anar passant a través de diversos nodes de la xarxa fins a arribar a la seva destinació.

Actualment els algorismes d'encaminament es poden classificar principalment en dos tipus, els coneguts com a *estat de l'enllaç* (LS[36]) i els de *vector distància* (DV[37]). Aquesta secció estarà destinada a detallar els diferents algorismes d'encaminament genèrics, que posteriorment veurem que s'utilitzen per als protocols d'encaminament específics a Internet.

(36) *LS* és la sigla de *link state*.

(37) *DV* és la sigla de *distance vector*.

Normalment la topologia d'Internet es modela com un graf; per tant, a l'hora de dissenyar els algorismes d'encaminament s'utilitza la teoria de grafs. Abans de descriure com funcionen els algorismes recordarem una mica què és un graf i quines eines ens proporciona per a poder-lo recórrer.

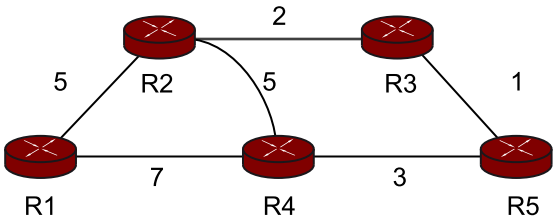
Un graf $G = (N, A)$ és un conjunt de nodes N i una sèrie A d'arestes que uneixen els nodes formant el graf. Cada aresta uneix un parell de nodes de N. Cal notar que dins d'una xarxa els nodes seran els encaminadors i les arestes els enllaços, ja que des del punt de vista de l'encaminament els equips finals no es consideren. En el nostre cas assumirem que tots els enllaços són bidireccionals, i per això el graf resultant de la xarxa sempre serà no dirigit. Dins d'una xarxa, igual que dins d'un graf, es coneix el node $\boldsymbol{x}$ com a *veí* del node $\boldsymbol{y}$ si hi ha una aresta $\boldsymbol{a}$ que els uneix. Mentre que cada aresta (enllaç) tindrà un cost associat per a recórrer-lo, el cost o mètrica d'un enllaç és una funció que defineix "quant costa" enviar un datagrama a través d'aquell enllaç, per la qual cosa generalment ens interessarà minimitzar aquest cost. Es poden definir moltes funcions de cost, com per exemple, el preu que s'ha de pagar per a enviar informació a través d'aquell enllaç, o bé el retard amb què la informació arriba a la destinació. Una altra possible mètrica és l'amplada de banda disponible (que en aquest cas ens interessa maximitzar).

Històricament el cost es mesurava en funció del retard i la càrrega dels enllaços, ja que les diferències de velocitats podien ser molt grans, per exemple de 56 kbps d'un port sèrie a 1,5 Mbps d'una línia T1. Avui dia en general el valor de

cost més utilitzat és el nombre de salts, ja que es considera que com més salts per a arribar a la destinació més lent serà l'enviament; cal notar que al nucli de la Xarxa tots els enllaços tenen amplades de banda i càrregues comparables.

De totes formes, per ara assumirem que la funció de cost de la nostra xarxa és el cost de cada un dels enllaços que la formen. A la figura 13 es mostra un exemple de xarxa amb la llista de costos associada.

Figura 13. Exemple de xarxa amb costos



Així doncs, en una xarxa qualsevol algorisme d'encaminament tindrà per objectiu obtenir la ruta amb menys cost per a arribar a la destinació. Per exemple, en el cas de la figura 13, la ruta de menys cost[38] per a arribar d'R1 a R5 és (R1, R2, R3, R5), tot i ser una ruta que necessita un salt més que la ruta més curta[39] (R1, R4, R5).

(38)En anglès, *least cost path*.

(39)En anglès, *shortest path*.

En l'exemple anterior, la decisió de la ruta de menor cost ha assumit que tots els nodes disposen d'informació de tota la xarxa[40], ja que per a saber que el millor camí per a arribar a R5 passa per R2, només es pot calcular si tenim informació global de la xarxa. Els algorismes d'estat de l'enllaç que veurem en breu assumeixen aquest coneixement global. En contraposició, es pot construir un algorisme que sigui descentralitzat (o sigui, que només disposem d'informació dels veïns immediats[41]). Els algorismes de vector-distància formen part d'aquesta categoria.

(40)En anglès, *global routing*.

(41)En anglès, *decentralized routing*.

Per a completar les diferents categories d'algorismes, una altra possible classificació és si l'algorisme és **estàtic** o **dinàmic**. Els algorismes estàtics són aquells que un cop configurats canvien poc al llarg del temps, i quan ho fan generalment és perquè algú fa manualment el canvi. En canvi, els algorismes dinàmics (que en la pràctica són els més utilitzats a Internet), estan pensats per a ajustar-se als canvis topològics de la xarxa de manera automàtica.

4.1. Algorisme d'encaminament per la ruta més curta

Aquesta és una de les tècniques més utilitzades per a desenvolupar protocols d'encaminament, atesa la seva senzillesa. En general es combina amb altres, com veurem més endavant.

Per a poder entendre bé aquest algorisme, primer s'ha de tenir clar què significa "la ruta més curta". La ruta més curta pot canviar depenent de la nostra mètrica; així, possibles mètriques són:

- Mínim nombre de salts
- Menor distància geogràfica
- Major amplada de banda
- Menor càrrega de l'enllaç

En la pràctica això es tradueix en un graf i uns costos a les seves arestes, com hem vist a la figura 13. Ara només ens cal trobar el camí menys costós (més curt) per a arribar a la destinació; hi ha molts algorismes per a aconseguir camins mínims, però el més utilitzat és l'algorisme de Dijkstra, un algorisme que va ser presentat per Edsger Dijkstra el 1959.

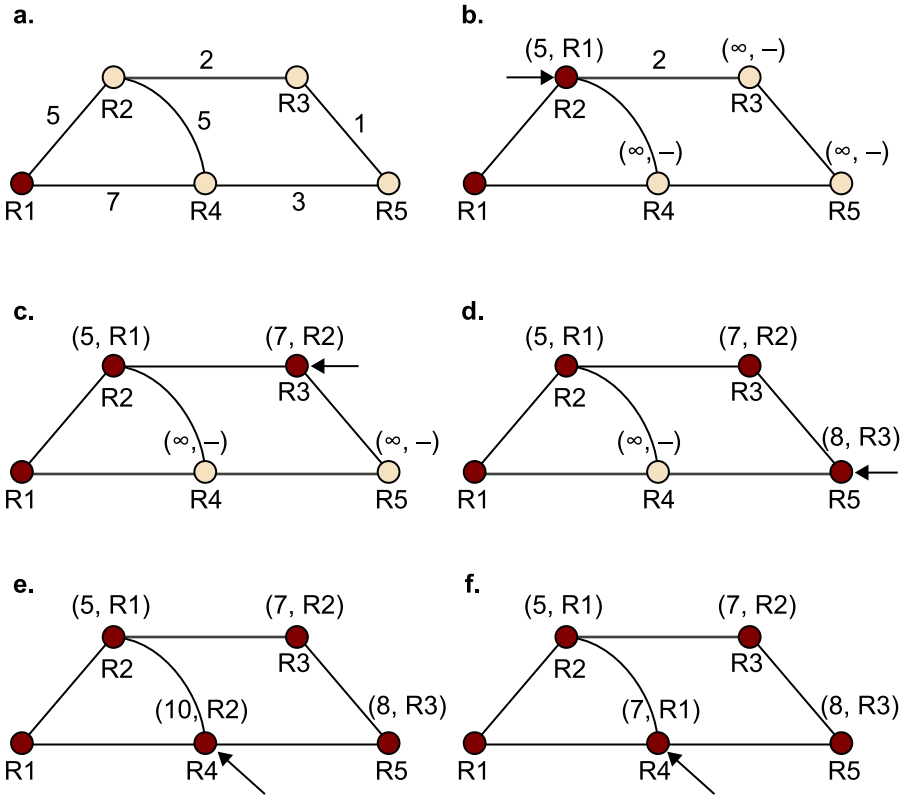
Lectura recomanada

E. W. Dijkstra (1959). "A note on two problems in connexion with graphs". *Numerische Mathematik* (núm. 1, pàg. 269-271).

Bàsicament, l'algorisme de Dijkstra és un algorisme de cerca en grafs que permet solucionar el problema de trobar camins mínims dins d'un graf amb costos positius als nodes. L'algorisme amb *k* iteracions és capaç de trobar els camins mínims cap a *k* destinacions amb els costos menors.

A la figura 14 es mostra un exemple d'execució de l'algorisme de Dijkstra. La figura 14a mostra el graf inicial amb els costos associats, els cercles plens mostren els nodes tractats almenys un cop, i la fletxa indica el node que estem calculant en aquesta iteració. L'objectiu és trobar els camins mínims d'R1 a la resta de nodes de la xarxa.

Figura 14. Exemple d'execució de l'algorisme de Dijkstra



Cada node tractat mostra entre parèntesis el cost total i el node previ per a arribar a R1. El funcionament simplificat de l'algorisme és:

1) Inicialment tots els nodes tenen cost infinit i no tenen predecessor per a arribar al node origen.

2) De totes les arestes del node inicial s'agafa la de menys cost (figura 14b). L'aresta ens porta a un nou node *D*.

3) La resta d'arestes es posen en una llista *L*.

4) S'afegeixen a *L* totes les arestes que surten de *D*, excepte la que ja ha estat tractada.

5) S'agafa l'aresta de cost mínim, que ens porta a un nou node *D*.

6) Si el cost és menor que l'existent s'actualitzen els valors amb els nous i anem al pas 3 de l'algorisme mentre quedin arestes en *L*.

7) En el cas que sigui major s'ignora i es va al punt 3 mentre quedin arestes en *L*.

Finalment ens dóna la ruta més curta. Si això ho repetim per a tots els nodes de la xarxa obtenim totes les rutes més curtes entre tots els nodes. Aquesta versió de l'algorisme és estàtica, i no preveu que hi pugui haver variacions a

la xarxa, així, si hi hagués algun canvi topològic, el sistema no se n'adonaria. Posteriorment veurem alternatives dinàmiques que permeten ajustar-se a canvis topològics.

Un punt important que cal tenir en compte quan es dissenya un algorisme d'encaminament és el cost algorísmic, ja que és una operació que es pot arribar a executar molts cops i pot condicionar fortament el temps de convergència (temps que passa des que es comença el càlcul de les rutes fins que es tenen totes). En el cas de Dijkstra, el cost és el següent:

- Suposem que tenim $n + 1$ nodes, i volem saber el cost de calcular els camins mínims des d'un node cap als n restants.

- Sabem que a cada iteració es va afegint un node a la llista de nodes considerats, i això fa que hàgim de fer $1 + 2 + 3 + ... + (n - 1) + n$ operacions, que fa el total següent:

$$\frac{n(n+1)}{2} = O(n^2)$$

De totes maneres, hi ha mecanismes per a optimitzar l'algorisme que ens permeten reduir el seu cost a $O(n)$.

4.2. Inundació

Un altre mecanisme per a propagar la informació dins d'una xarxa, si bé no és exactament un algorisme d'encaminament, és el que es coneix com a *inundació*.

Aquest algorisme es basa a enviar la informació a totes les interfícies de l'encaminador exceptuant aquella per la qual arriba. D'aquesta manera s'aconsegueix que totes les estacions rebin la informació que els volem transmetre. En aquest cas no es propaguen les rutes sinó directament s'envia la informació per transmetre.

A simple vista es pot veure que aquesta solució té una sèrie d'inconvenients. El primer és la gran quantitat de missatges duplicats que s'arriben a generar i que arriben a les estacions. L'altre és la replicació infinita dels missatges, que es pot evitar de les maneres següents:

- Posant una data de caducitat als paquets: quan un paquet fa un cert temps que ha estat generat se suposa que tothom l'ha rebut, que la seva validesa ha expirat, i llavors deixa de ser difós.

- També es pot limitar el nombre màxim de salts que pot sofrir un paquet, i d'aquesta manera cada encaminador descompta un d'un comptador present a la capçalera del datagrama, que quan arriba a 0 es descarta.

- La darrera forma proposada, i la més utilitzada, és posant un número de seqüència als datagrames. Si una estació rep un datagrama per al qual ja ha fet difusió prèviament, aquest paquet no es reenvia per cap interfície. Aquesta darrera alternativa ha de considerar la possibilitat que els números de seqüència es repeteixin, i per evitar això el que es fa és limitar el nombre de missatges que es poden enviar per unitat de temps, i destinar al comptador suficients bits per a evitar replicacions; per exemple, amb 32 bits es poden generar centenars de milions de números de seqüència sense tenir col·lisions.

A causa de la seva baixa eficiència aquest sistema no s'acostuma a utilitzar tret que sigui en entorns on sigui crític que la informació hagi d'arribar a la destinació, i en topologies molt canviants; per exemple, en operacions militars, en què el risc que els encaminadors siguin destruïts no és negligible i sempre es necessita una ruta alternativa de manera ràpida.

Pel seu disseny, igual que l'algorisme per ruta més curta, l'enviament per inundació es considera un mecanisme estàtic, ja que no es veu afectat pels canvis topològics ni els considera.

4.3. Algorisme d'encaminament d'estat de l'enllaç

Com ja hem introduït anteriorment, els algorismes d'estat de l'enllaç basen el seu funcionament a trobar el camí de cost mínim entre dos nodes i també en el fet que tots els nodes tenen coneixement total de la topologia i dels costos dels enllaços de tota la xarxa. Així, l'algorisme estarà dividit principalment en dues parts: la primera, conèixer la topologia, i la segona, trobar el camí millor per a arribar als altres nodes.

Exercicis

16. Quins problemes creieu que pot representar per a aquest grup d'algorismes el fet d'haver de conèixer tota la topologia en xarxes grans?

Solució de l'exercici 16

Saber tota la topologia implica desar-la en memòria, per la qual cosa si la xarxa està composta per molts nodes (com ara Internet) hi ha un greu problema de memòria per a conservar tot el graf.

Aquest és el primer algorisme dinàmic que veiem. Així, per a considerar aquest dinamisme, els passos que segueix l'algorisme són més elaborats que abans; cada encaminador, quan executa l'algorisme:

1) Esbrina quins són els veïns (ara la llista no és estàtica).

2) Musura el cost de l'enllaç que uneix l'encaminador amb cada un dels veïns.

3) Envia aquesta informació a tots els encaminadors de la xarxa.

4) Calcula la ruta més curta als altres nodes a partir de la informació rebuda (anàloga a la que ell ha enviat), utilitzant algun algorisme de camins mínims, com ara el de Dijkstra.

Per a poder adquirir coneixement de tota la topologia, cada node agafa informació dels seus veïns; això es pot fer enviant un paquet per totes les interfícies de sortida de l'encaminador que pregunti quins encaminadors hi ha a cada subxarxa. Els encaminadors, en rebre un paquet d'aquest tipus, respondran amb l'adreça IP, que serà emmagatzemada a la base de dades de l'encaminador que ha enviat la petició.

Un cop obtinguts tots els veïns, és necessari calcular el cost de l'enllaç, i per això s'envien paquets de prova[42] cap als veïns, de manera que aquests responen (per exemple s'envia un *echo request* i es respon amb un *echo reply*) i es calcula el temps que ha transcorregut per tenir una idea del retard (o sigui, el cost dels enllaços). De vegades, per millorar-ne l'estimació, s'envien diversos paquets i es fa la mitjana del resultat. Si s'utilitzen altres mètriques com ara l'amplada de banda, l'algorisme pot agafar les dades directament de la configuració de l'enllaç, i així s'estalvia els paquets de prova. Aquests datagrames de prova s'envien periòdicament per tal d'anar refrescant l'estat dels enllaços.

[42] En anglès, *probe packets*.

Exercicis

17. Una xarxa amb 10 nodes i 32 enllaços en total té un temps de refresc $t_r = 30$ s. S'envien 3 paquets de prova per ronda de mida 64 octets cada un. Calculeu quin és el sobrecost (*overhead*) que causa el protocol LS per culpa del càlcul del retard.

Solució de l'exercici 17

Cada node enviarà tres paquets cada 30 s per tots els seus enllaços. En la pràctica això vol dir que cada enllaç rebrà 3 × 2 = 6 paquets d'anada i les corresponents 6 respostes cada 30 s. Com hi ha 32 enllaços, això fa un total de (6 + 6) × 32 = 384 paquets cada 30 s. Assumint que es distribueixen uniformement al llarg dels 30 s això fa un total de 384 × 64 = 24.576 octets, que són 196.608 bits cada 30 s, que és equivalent a 196.608/30 = 6.553 ~ 6,5 kbps de sobrecost causat pels paquets de prova.

El pas següent és enviar aquesta informació als altres nodes de la xarxa perquè tots puguin arribar a saber la topologia de tota la xarxa; aquesta informació es pot passar de diverses maneres, com per exemple per mitjà d'inundació, com hem vist anteriorment, o bé utilitzant trànsit de difusió.

El darrer pas de l'algorisme un cop s'ha obtingut la topologia és trobar el camí mínim des de cada encaminador a tots els altres. Els algorismes LS segueixen l'algorisme de Dijkstra vist en el subapartat 4.1 per a trobar els camins mínims. Un cop arribats a aquest punt, es pot dir que l'algorisme ha convergit. O el que és el mateix, que ha arribat a un punt estable on totes les destinacions són conegudes i accessibles.

Vegeu també

Vegeu l'algorisme de Dijkstra en el subapartat 4.1 d'aquest mòdul didàctic.

Amb tot, la feina de l'algorisme no acaba aquí. En una xarxa ideal, un cop l'algorisme ha convergit no caldria fer res més, però en un cas real, hi ha xarxes que deixen d'existir, o bé fallades de maquinari que provoquen que un encaminador caigui, o bé algun problema al cablatge que fa que un enllaç deixi de funcionar, etc. Per això els algorismes LS han de considerar també el cas del càlcul de noves rutes. Per tant, sempre que hi hagi un problema a la xarxa, o apareguin nous encaminadors, un cop detectats s'ha d'informar a tots els nodes de la xarxa de la incidència i s'ha de tornar a executar l'algorisme a cada encaminador. El cost de l'algorisme està determinat pel cost de Dijkstra, i per tant és equivalent al de camins mínims vist anteriorment.

4.4. Algorisme d'encaminament vector-distància

El segon algorisme d'encaminament dinàmic que veurem en aquesta secció és el vector-distància. Com ja hem descrit anteriorment, la diferència més gran entre els algorismes LS i els DV és el fet que els DV no tenen informació topològica de tota la xarxa, només del que aprenen per mitjà dels seus veïns.

Exercicis

18. Quin avantatge creieu que pot aportar el fet de no tenir tota la informació? I quin inconvenient?

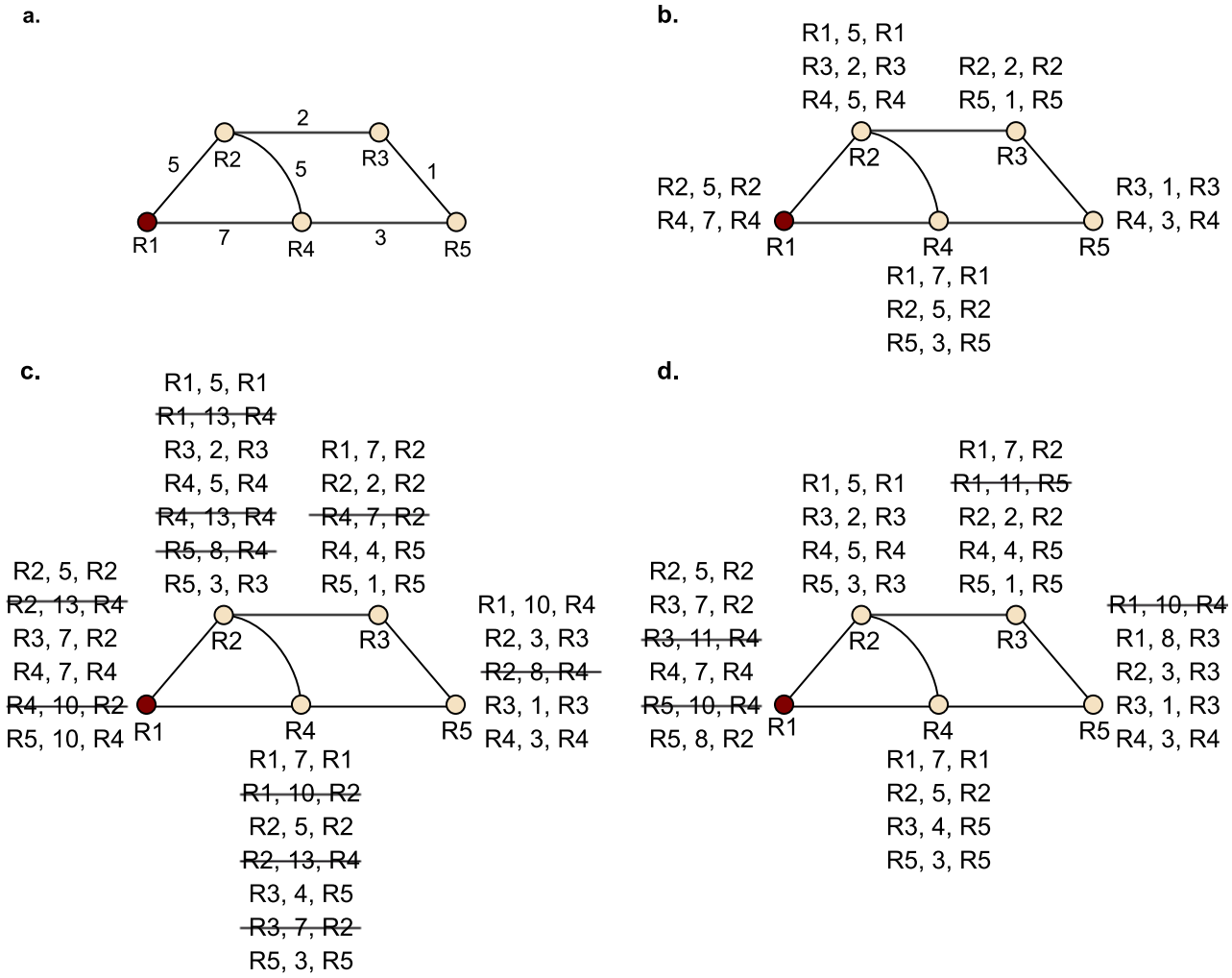
Solució de l'exercici 18

El clar avantatge d'aquesta aproximació és el fet de no haver de reservar en memòria tota la topologia, amb la reducció conseqüent de recursos que implica això en topologies grans. L'inconvenient és que el fet de no tenir tota la topologia no ens permet fer segons quins tipus d'optimitzacions, especialment si hi ha diversos camins per a arribar a les destinacions.

Els algorismes DV es basen a mantenir una taula, que s'acaba tractant com un vector, que informa de la millor distància coneguda cap a cada una de les destinacions. Aquesta informació es va actualitzant dinàmicament amb les dades que es reben dels múltiples veïns de cada node.

Aquesta taula, que és la taula d'encaminament, conté la interfície de sortida, i també quina és la destinació i una mètrica, que en aquest cas acostuma a ser el nombre de salts que són necessaris per a arribar a la destinació especificada. Però utilitzar alguna altra informació, com ara retards, es pot aconseguir preguntant als veïns directes.

Figura 15. Exemple d'execució de l'algorisme DV



Per a descriure els passos que fa l'algorisme utilitzarem la mateixa xarxa que per al cas de camins mínims, tal com mostra la figura 15a. Com els algorismes DV es basen a no tenir informació global del sistema, tots han de treballar conjuntament (de manera distribuïda) per tal d'aconseguir la convergència final.

Els passos que cal seguir són:

1) Cada encaminador mira la mètrica dels enllaços connectats als seus veïns.

2) Propaga les dades d'encaminament adquirides a tots els veïns de manera intel·ligent (no s'envia informació d'un encaminador a si mateix, i tampoc no s'envien les rutes conegudes que no són òptimes).

3) Cada encaminador rep la informació dels nodes veïns i es queda amb les rutes amb cost menor (vegeu la figura 15 per a un exemple complet).

4) Es repeteix des del pas 2 fins que no hi ha més rutes per transmetre.

Com es pot comprovar, aquest algorisme és molt desitjable, ja que ens proporciona els avantatges següents:

- És autoconvergent: o sigui, no ens cal especificar en quin moment s'ha de parar l'enviament de missatges; senzillament, quan se saben totes les rutes es para.

- Treballa de manera asíncrona: si bé a l'exemple de la figura 15 s'ha discretitzat el temps en passos per simplificar-ne l'explicació, l'algorisme funciona independentment de l'ordre en què es reben els missatges i independentment del moment específic en què s'envien.

- És molt eficient en recursos: no ens cal saber tota la topologia, només tenint la taula d'encaminament és suficient.

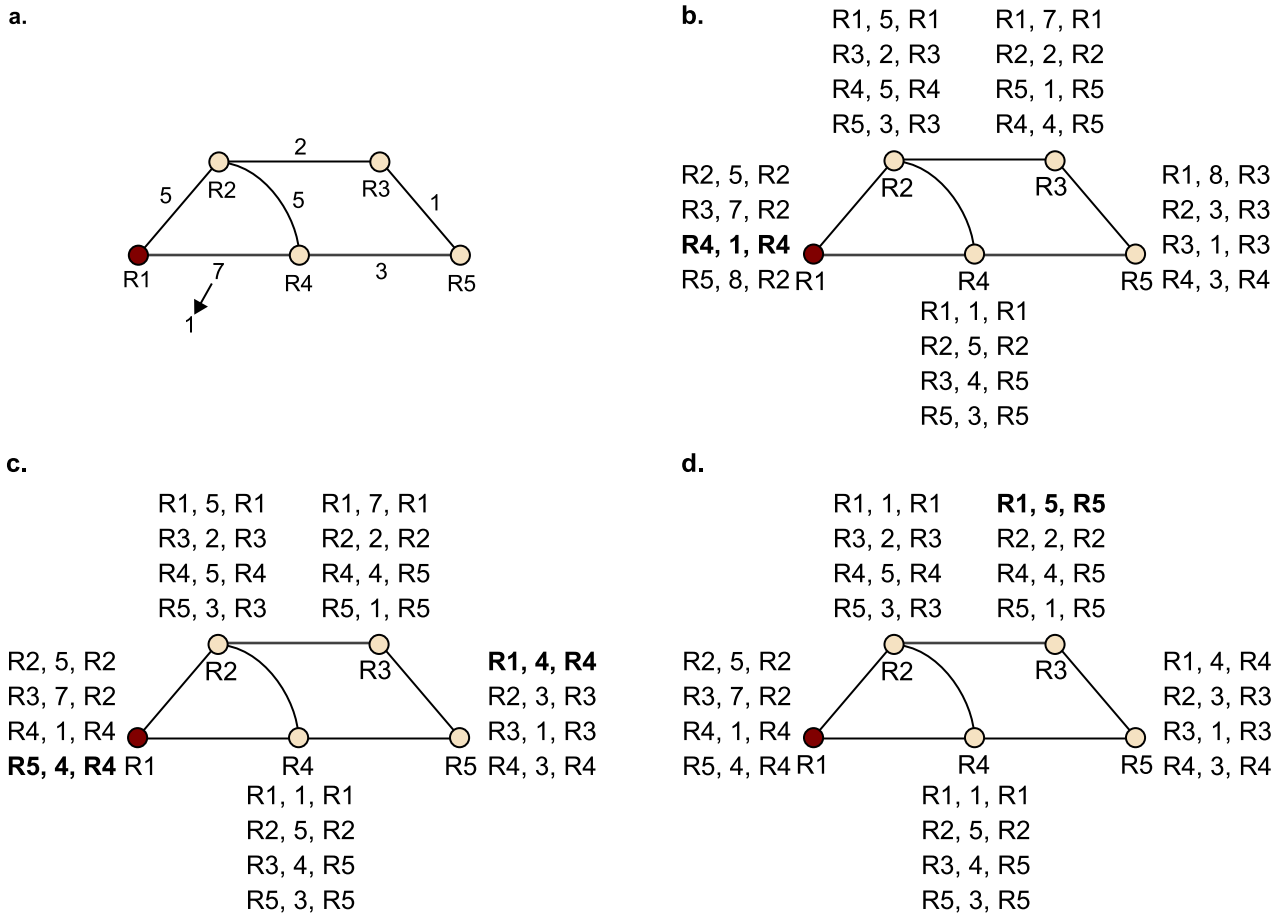
Un cop hem vist com funciona l'algorisme, anem ara a detallar per què funciona de manera òptima. Aquest algorisme va ser inicialment dissenyat per Bellman i Ford el 1957, per la qual cosa es coneix com l'algorisme de Bellman-Ford. Aquest algorisme diu que un camí mínim entre dos punts que involucren diversos segments (salts) està compost per subcamins que a la seva vegada són mínims. En altres paraules, que si el camí òptim per a anar d'R1 a R5 és R1-R4-R3-R5, el camí òptim per a anar d'R4 a R5 serà lògicament R4-R3-R5. L'equació recursiva que modela aquest comportament és: $d_x(y) = \text{mín}_z\{c(x,z) + d_z(y)\}$, en què es vol trobar el mínim per a anar de x a y, i $c(x,z)$ és el cost d'anar del node $\boldsymbol{x}$ al pròxim salt $\boldsymbol{z}$ per a tots els enllaços de x cap a tots altres veïns (z).

Ja s'ha dit que els algorismes DV eren descentralitzats i dinàmics, i això implica que quan hi ha algun canvi topològic s'han de prendre mesures per a mantenir la coherència a les taules d'encaminament de tots els encaminadors. Amb els algorismes LS això era més o menys senzill pel fet que tots els nodes esperen tenir coneixement de tota la topologia, però en el cas dels algorismes DV la cosa es complica una mica més, sobretot en el cas que algun enllaç passa a tenir un cost major.

Quan un node detecta un canvi en un enllaç (bé en el cost, bé perquè ha deixat de funcionar, o bé perquè torna a funcionar) l'algorisme mira si això li provoca algun canvi en el seu camí de menys cost cap a alguna destinació. En el cas que sigui així, informa del canvi als seus veïns, que a la seva vegada tornaran a executar l'algorisme de convergència vist anteriorment per veure si hi ha hagut canvis i propagar-los en cas que sigui necessari.

Analitzem els casos quan un enllaç redueix i incrementa el seu cost. Per a fer-ho utilitzarem la mateixa xarxa exemple que fins ara. Assumirem que l'algorisme ja ha convergit (figura 15d) i aleshores en t_0 el cost de l'enllaç R1-R4 passa de valer 7 a 1. L'evolució de l'algorisme es pot veure a la figura 16: primer s'adonen del canvi R1 i R4, i passaran a propagar aquest nou estat. Com es pot veure a la figura 16b, la ruta des d'R1 a R5 i a R3 no és correcta, però com l'enllaç involucrat no és el que ha canviat, l'algorisme encara no té manera de saber que hi ha una ruta millor. Per simplicitat en aquest cas no es mostren les rutes que es desestimen, només les noves en negreta a cada pas.

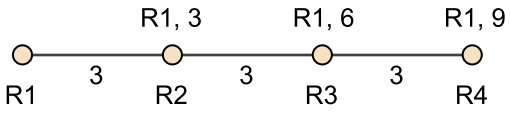
Figura 16. Reducció del cost d'un enllaç amb algorismes DV



En vista dels resultats es podria pensar que en el cas en què un enllaç incrementa el seu cost la convergència també seria ràpida. Bé, anem a veure que això no és així. Per a il·lustrar-ho necessitem una xarxa una mica més senzilla, com la de la figura 17, on es mostren a la part superior els costos per a arribar a R1. Ara suposarem que l'enllaç entre R1 i R2 es desconnecta (té cost infinit). El procés ara seria: R2 avisa R3 dient-li: el cost d'arribar a R1 ja no és 3 sinó que és infinit. I R3 respondria: jo tinc una ruta molt bona que diu que per a arribar a R1 el cost és 6. Penseu que en aquest cas, R2 no pot saber que la ruta que li arriba d'R3 passa per R2. Aleshores R2 actualitzaria la seva taula d'encaminament a R1, 9. En aquest moment R3 veuria que hi ha dues rutes amb el mateix cost (9) per a arribar a R1, i per tant en triaria una a l'atzar i incrementaria el cost a 12, i això s'aniria repetint fins a l'infinit. El problema

aquí és que la informació que es passa és estrictament de cost i no del camí que cal seguir, i per tant els encaminadors no tenen cap manera de saber si ells mateixos formen part del camí anunciat.

Figura 17. Exemple de compte a l'infinit



Per tal de resoldre aquest problema es va proposar el que es coneix com el compte fins a l'infinit. Així, quan un camí assoleix un cost d'"infinit" se suposa que la destinació no és accessible, i aquest infinit tindrà un valor numèric que depèn de la implementació. Cal dir que aquest compte fins a l'infinit només evita un bucle infinit, però no resol el problema del temps de convergència. S'han proposat solucions com ara *poisoned reverse*, però no acaben de resoldre el problema general. En canvi, en seccions posteriors veurem com ho resolen els algorismes utilitzats a Internet.

Per acabar aquesta part farem una petita comparativa dels dos algorismes d'encaminament en els quals es basen els protocols d'encaminament utilitzats a Internet. Així, les principals diferències són:

- Com que LS té coneixement de tota la topologia, no té el problema del compte fins a l'infinit, i per tant sempre convergirà per mitjà de l'algorisme de Dijkstra amb cost $O(n^2)$.

- Els algorismes LS sempre trobaran una ruta igual o millor que els algorismes DV.

- Quan hi ha un canvi a la xarxa, els algorismes LS necessiten tornar a executar tot l'algorisme de convergència, mentre que els algorismes DV només propaguen la ruta en el cas que sigui un nou camí de menys cost, amb una complexitat algorísmica molt més baixa.

- El nombre de missatges que cal enviar també és diferent. En el cas d'LS s'ha d'informar a tots els nodes de la xarxa del canvi, mentre que els algorismes DV només informen als veïns, que propagaran el canvi només si és necessari.

- Els algorismes LS tenen un greu problema d'escalabilitat quan la xarxa és molt gran, i per tant no poden ser utilitzats a tot Internet.

Com es pot comprovar, no hi ha un clar guanyador, i cada alternativa té els seus avantatges. Així doncs, tal com veurem, el que s'acaba fent a Internet és utilitzar totes dues alternatives.

Vegeu també

Vegeu els protocols d'encaminament a Internet en l'apartat 5 d'aquest mòdul didàctic.

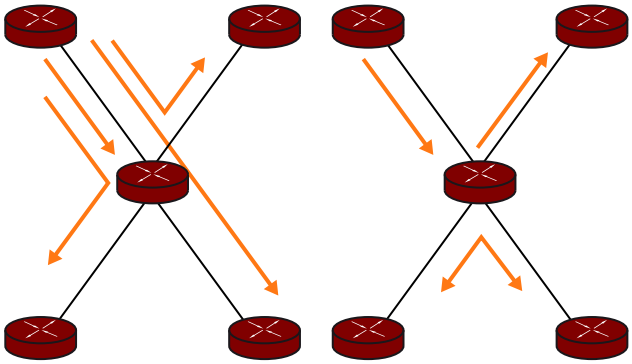
4.5. Encaminament basat en difusió

Com ja hem vist anteriorment, una manera d'enviar informació a una xarxa és per mitjà d'inundació. Aquesta tècnica, tot i ser molt poc eficient, pot portar algun benefici pel fet de ser robusta. L'enviament de dades per inundació no envia informació d'encaminament sinó directament les dades.

L'encaminament basat en difusió, més que un algorisme d'encaminament, és un mecanisme per a transferir informació d'encaminament usant trànsit de difusió. Com ja hem vist, el trànsit de difusió és aquell en què s'envia un datagrama que pot rebre i interpretar tothom de la subxarxa.

Els avantatges d'aquesta solució respecte a la inundació es poden veure clarament a la figura 18.

Figura 18. Comparació d'enviament per unidifusió i per difusió



Com es pot observar, totes dues alternatives generen un datagrama destinat a tots els encaminadors. Però si ens fixem, la diferència és que en el cas d'unidifusió l'encaminador central ha de reencaminar paquets per totes les seves interfícies, mentre que en el cas de difusió (a la dreta a la figura) l'encaminador central es limita a enviar un paquet nou de difusió a tots els seus veïns amb la informació d'encaminament. Aquesta manera d'enviar informació d'encaminament la utilitzen algorismes LS per a passar la informació entre els nodes. De totes maneres, aquest mecanisme es veu afectat pels mateixos problemes que la inundació: cicles al graf i sobrecost per l'excés de missatges a la xarxa, solucionats amb la utilització de números de seqüència, com hem vist anteriorment.

La manera més utilitzada per a evitar que un encaminador rebi la mateixa informació més d'un cop des d'encaminadors diferents és el que es coneix com a arbre d'expansió[43]. L'arbre d'expansió conté tots els nodes del graf, però amb la particularitat que garanteix que no hi ha cicles, i per tant garanteix que no s'enviaran més paquets que els estrictament necessaris.

[43]En anglès, *spanning tree*.

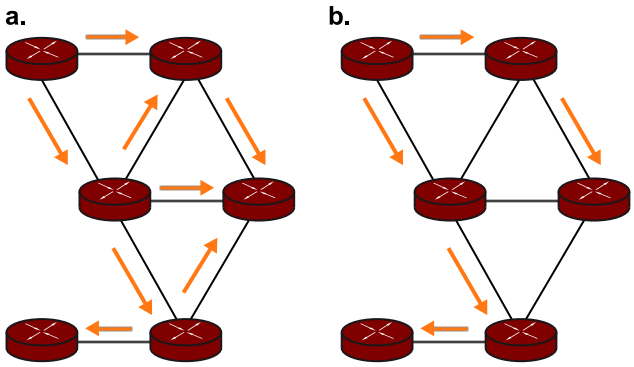
Lectura complementària

Es poden trobar exemples amb diverses tècniques per a construir arbres d'expansió a l'article següent:

F. C. Gartner (2003). "A survey of self-stabilizing spanning tree construction algorithms".

La figura 19 mostra un exemple d'arbre d'expansió; la part de l'esquerra mostra l'intercanvi amb difusió, mentre que a la dreta es mostra un arbre d'expansió i l'intercanvi de missatges en aquest cas. La part més complexa d'aquesta tècnica és la creació i manteniment (quan hi ha canvis topològics) de l'arbre.

Figura 19. Exemple d'arbre d'expansió



El que fa de l'arbre d'expansió una opció interessant és que independentment de quin node enviï la informació mai no s'enviaran més paquets dels estrictament necessaris perquè tothom rebi la informació exactament un cop.

4.6. Encaminament basat en multidifusió

Igual que amb l'encaminament basat en la difusió, l'encaminament basat en la multidifusió no s'utilitza per si mateix, sinó que ens proporciona un mecanisme per a poder encaminar a través d'Internet el trànsit de multidifusió. El principal objectiu d'aquest mecanisme d'encaminament és permetre seleccionar qui rebrà la informació d'encaminament de manera òptima, a diferència de la solució basada en la difusió, en què no es podien fer distincions i tothom rep tota la informació. Evitar la difusió està determinat per motius de seguretat en la transferència de les dades d'encaminament; així, només reben la informació els encaminadors amb suficients privilegis.

En l'encaminament basat en multidifusió cada encaminador d'un grup multidifusió crea un arbre d'expansió des d'ell cap a tots els encaminadors multidifusió de la subxarxa (cal notar que aquests encaminadors han de tenir suport multidifusió, no pot ser qualsevol encaminador). Aquest arbre inclou tots els encaminadors, tant si estan subscrits a algun grup com si no.

Quan es rep un datagrama dirigit a un grup concret, l'encaminador fa una "poda" de l'arbre de manera que només li queden tots els nodes que formen part del grup. La poda d'un arbre d'expansió també és un arbre d'expansió però

amb un conjunt menor de nodes. Un cop feta la poda només cal enviar els datagrames als veïns, els quals a la seva vegada s'encarregaran de processar-los cap als nodes subscrits, i d'enviar-los als seus veïns de l'arbre multidifusió. L'arbre es va ajustant dinàmicament depenent de les noves subscripcions a l'arbre, o baixes en cas que un node deixi de formar-ne part.

El problema principal de la multidifusió, i el motiu pel qual pràcticament no s'utilitza a Internet, és que cada encaminador ha de mantenir en memòria els *n* arbres que representen els *n* grups als quals estan subscrits els seus equips, i això per a xarxes amb molts nodes té un greu problema de memòria i temps de procés a l'encaminador, a part que no tots els encaminadors de la Xarxa tenen el suport multidifusió activat.

5. Protocols d'encaminament a Internet

Abans de començar la descripció dels diferents protocols d'encaminament presents a Internet, és convenient veure una mica amb més de detall com s'estructura la Xarxa.

Fins ara hem vist que Internet se separa per xarxes, que a la seva vegada estan estructurades en subxarxes, però en realitat, per simplificar-ne la gestió, hi ha encara un nivell més d'abstracció: els dominis. Un domini és el conjunt de totes les xarxes que formen part d'una mateixa unitat administrativa; els dominis també es coneixen com a sistemes autònoms (AS[44]). Un sistema autònom indica una única política de gestió dels recursos (adreces IP, separació en subxarxes, etc.) d'una forma interna i transparent a la resta d'Internet. Cada sistema autònom es comunica per mitjà dels encaminadors de vora[45] amb altres sistemes autònoms, tal com mostra la figura 20.

(44) *AS* és la sigla d'*autonomous systems*.

(45) En anglès, *border routers*.

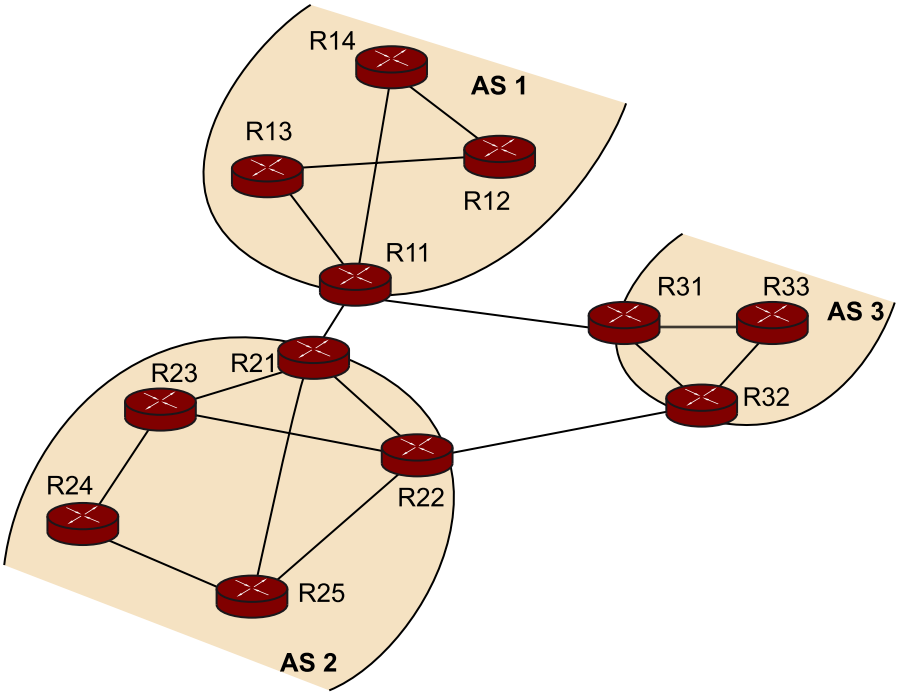
Alguns autors, com Tanenbaum o Kurose, anomenen l'encaminament de nivell d'AS **encaminament jeràrquic**.

Lectures recomanades

A. S. Tanenbaum (2003). *Redes de computadores* (4a. ed.). Pearson.

J. F. Kurose; K. W. Ross (2005). *Computer networking: a top-down approach featuring the Internet*. Addison-Wesley.

Figura 20. Exemple de jerarquia amb AS



A la figura es poden observar tres AS, en què AS1 té un encaminador de vora, i AS2 i AS3 en tenen dos. En aquesta solució jeràrquica, els encaminadors de vora es limiten a anunciar als AS veïns estrictament les rutes que gestiona l'AS; generalment a Internet això es fa per mitjà d'un protocol anomenat *BGP*[46]. Aquest tipus d'encaminament es coneix com a encaminament interdomini, ja

(46) *BGP* és la sigla de *border gateway protocol*.

Vegeu també

Vegeu el protocol BGP en el subapartat 5.3 d'aquest mòdul didàctic.

que involucra connexions entre dominis (AS). En el cas que un AS tingui més d'un encaminador de vora, es dirà que aquest AS té connectivitat *multihoming* a Internet, o sigui, que podrà ser accedit des d'Internet a través de diferents punts.

En paral·lel a l'encaminament interdomini hi ha l'encaminament intradomini, que és el que gestiona internament la connectivitat entre els diversos encaminadors d'un mateix domini; en aquest cas es poden utilitzar protocols d'encaminament com RIP[47], OSPF[48], o bé una variant de BGP coneguda com a IBGP[49].

(47) *RIP* és la sigla de *routing information protocol.*

(48) *OSPF* és la sigla d'*open shortest path first*.

(49) *IBGP* és la sigla d'*intra-domain border gateway protocol.*

Un encaminador, per a poder enviar els datagrames al pròxim salt, basa el seu funcionament en el que es coneix com a *taules d'encaminament*. Una taula d'encaminament indica per a cada prefix destinació quin és l'encaminador següent al camí. Un possible exemple de taula d'encaminament pot ser:

Destination	Gateway	Genmask	Metric	Iface
183.128.13 . 0	0 . 0 .0 .0	255.255.255.0	2	eth0
147.83 .120. 0	183.128.13.2	255.255.255.0	2	eth0
147.83 . 0 . 0	131.10 .4 .2	255.255. 0 .0	2	eth2
0 . 0 . 0 . 0	150.18 .87.1	0 . 0 . 0 .0	2	eth1

Cal dir que el format de la taula pot canviar depenent del fabricant de l'encaminador, però la informació mínima continguda és sempre la mateixa. Detallant per columnes tenim:

- ***Destination***: indica el prefix destinació dels datagrames.
- ***Gateway***: que és la IP del proper salt. Quan és 0.0.0.0 vol dir que ja estem a l'últim salt i s'apliquen les tècniques ARP vistes anteriorment per a l'enviament de datagrames dins de la mateixa subxarxa.
- ***Genmask***: és la màscara de xarxa que indica el conjunt d'IP que corresponen a la destinació.
- ***Metric***: indica el cost d'agafar aquesta ruta.
- ***Iface***: especifica la interfície física per la qual sortiran els datagrames.

De totes maneres, com es pot veure a la taula anterior, hi ha prefixos que se superposen; en concret, el 147.83.120.0/24 està inclòs en el 147.83.0.0/16, que a la seva vegada està inclòs en el 0.0.0.0/0.

Per tal de desfer aquesta ambigüitat, els encaminadors usen la tècnica *longest prefix match*, o el que és el mateix, agafar la ruta per la qual la destinació té més bits en comú (més part de la IP es veu reflectida a la ruta). Per això les rutes en general s'ordenen per màscara, de la màscara més gran a la més petita, de

manera que la primera ruta que correspon a la destinació és la que s'utilitza. Així, si ens arriba un datagrama destinat a l'adreça 147.83.121.3 l'encaminador comprovarà que la 183.128.13.0/24 no correspon, la 147.83.120.0/24 tampoc i la 147.83.0.0/16 sí; per tant, l'encaminador triat per a reenviar el paquet és el 131.10.4.2, que està directament connectat a la interfície *eth2* del nostre encaminador.

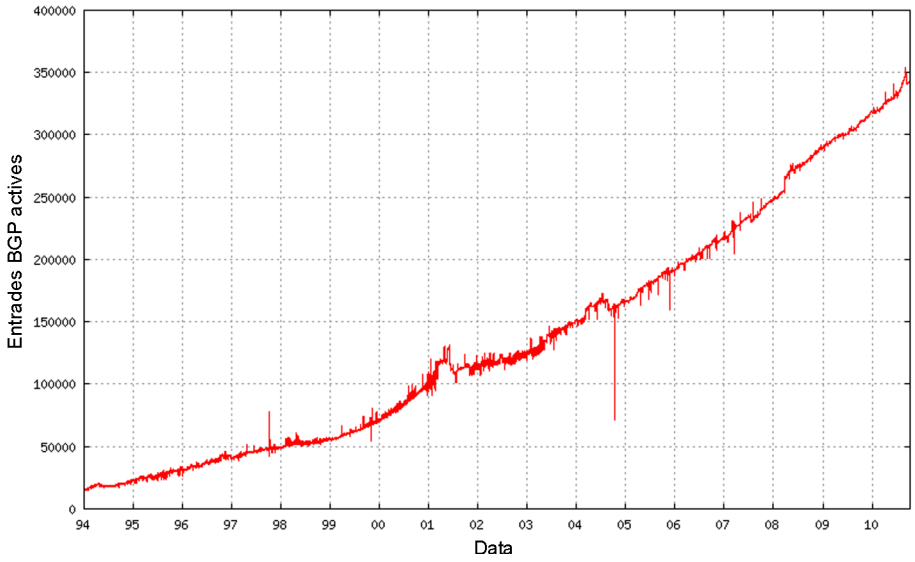
Si la IP destinació hagués estat la 120.134.23.235, s'hauria agafat la darrera ruta. Cal indicar que 0.0.0.0/0 indica la ruta per defecte[50], que és la ruta per on s'envien els datagrames que no tenen cap ruta més específica. Un dels grans problemes d'Internet actualment és que els encaminadors del nucli de la Xarxa[51] tenen el que es coneix com a DFZ[52], de manera que no tenen cap entrada a la taula d'encaminament que sigui per defecte. Això implica que coneixen el camí cap a totes les subxarxes, i per tant les seves taules d'encaminament estan formades per centenars de milers d'entrades. La figura 21 mostra un exemple de l'evolució des del 1994 fins a començaments del 2010 en nombre d'entrades de la mida de la taula d'encaminament a la DFZ.

(50) En anglès, *router of last resort*.

(51) En anglès, *core routers*.

(52) *DFZ* és la sigla de *default free zone*.

Figura 21. Exemple del nombre d'entrades en la taula d'encaminament d'un encaminador del nucli



Font: http://bgp.potaroo.net/as6447/

5.1. RIP

RIP és un dels primers protocols dissenyats per a ARPANet; està basat en els mecanismes d'encaminament del tipus DV i s'aplica, encara avui dia, en entorns intradomini.

Hi ha una primera versió, i una segona compatible amb l'anterior.

La forma de funcionament del protocol és pràcticament la mateixa que la vista per a l'algorisme DV teòric. Les úniques particularitats que té RIP respecte de DV són:

Vegeu també

Vegeu l'algorisme d'encaminament vector-distància en el subapartat 4.4 d'aquest mòdul didàctic.

- Que el cost dels enllaços es compta segons el nombre de salts, cosa que vol dir que cada enllaç té un cost d'1. Es compten tots els salts i la subxarxa destinació; després en veurem un exemple.

- Es posa l'infinit a 15 salts, fet que implica que la xarxa controlada per RIP no pot tenir un diàmetre més gran que 15 salts.

- Els encaminadors RIP es passen informació cada 30 s per a refrescar la informació d'encaminament.

- Si no es reben actualitzacions dels veïns durant 180 s, se suposa que el veí ha caigut i s'actualitzen les rutes i es propaga la informació a la resta de veïns.

Així RIP genera dos tipus de missatges: missatges d'anunci[53] i missatges de resposta[54]. Per la seva banda, els missatges d'anunci s'utilitzen per a anunciar la presència de l'encaminador i per a respondre als veïns que s'ha rebut correctament una resposta.

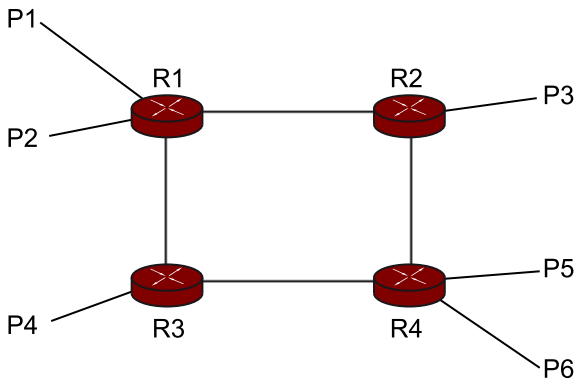
(53)En anglès, *RIP advertisements*.

(54)En anglès, *RIP response messages*.

Els missatges de resposta contenen les rutes conegudes per l'encaminador (fins a 25 per paquet). En concret s'envien els prefixos coneguts i la distància a la qual es troben. El receptor del missatge de resposta afegirà aquesta informació a la seva taula d'encaminament utilitzant la tècnica descrita pels algorismes DV.

Per a veure un exemple de funcionament de RIP, suposem la xarxa de la figura 22. Volem veure l'evolució dels anuncis enviats cap a R1; a la figura es mostren amb R# els encaminadors i amb P# els diferents prefixos que anuncia cada xarxa. Cal notar que RIP, com tots els protocols d'encaminament, utilitza prefixos (IP) per als anuncis, no encaminadors concrets.

Figura 22. Exemple de xarxa RIP



Així, a l'instant 0 la taula d'encaminament d'R1 serà:

Subxarxa destinació	Encaminador següent	Nombre de salts
P1	-	1
P2	-	1

Al cap de 30 s es rebran els anuncis d'R2 i d'R3, que configuraran la xarxa:

Subxarxa destinació	Encaminador següent	Nombre de salts
P1	-	1
P2	-	1
P3	R2	2
P4	R3	2

Finalment al cap de 30 s més (60 s de l'inici), l'algorisme convergirà amb aquesta taula:

Subxarxa destinació	Encaminador següent	Nombre de salts
P1	-	1
P2	-	1
P3	R2	2
P4	R3	2
P5	R2	3
P6	R2	3

En aquest cas es tria R2, però com el cost és el mateix, s'hauria pogut triar R3 com a pròxim salt per a arribar a P5 i P6.

Suposem ara que l'enllaç que uneix R2 i R4 cau; al cap de 180 s, R2 veurà que no es reben actualitzacions d'R4, i per tant avisarà d'això. Farà que la taula d'R1 quedi de nou:

Subxarxa destinació	Encaminador següent	Nombre de salts
P1	-	1
P2	-	1
P3	R2	2
P4	R3	2

Això succeeix perquè les rutes que no s'utilitzen es descarten automàticament, i quan arribi l'actualització següent d'R3, el protocol convergirà per R1 amb el següent:

Subxarxa destinació	Encaminador següent	Nombre de salts
P1	-	1
P2	-	1
P3	R2	2
P4	R3	2
P5	R3	3
P6	R3	3

Lògicament, aquesta informació arribarà a R2 al cap de 30 s afegint les entrades:

.	.	.
.	.	.
P5	R1	4
P6	R1	4

5.2. OSPF

OSPF, per sobre de RIP, és un dels protocols intradomini més utilitzats actualment. Si RIP s'acostuma a utilitzar en xarxes relativament petites, OSPF està pensat per a ser utilitzat internament en AS de mides majors. Per això sorprèn que OSPF sigui un protocol del tipus LS.

La versió 2 d'OSPF té un funcionament bastant intuïtiu: tots els nodes de la xarxa envien informació topològica per mitjà de trànsit de difusió (tot i que hi ha altres extensions que permeten utilitzar trànsit d'unidifusió o fins i tot multidifusió). Un cop la topologia és sabuda per tothom, s'executa l'algorisme de Dijkstra tal com ja hem vist. Per robustesa, cada 30 min l'algorisme intercanvia informació topològica, tot i que no hagi canviat res a la xarxa.

Vegeu també

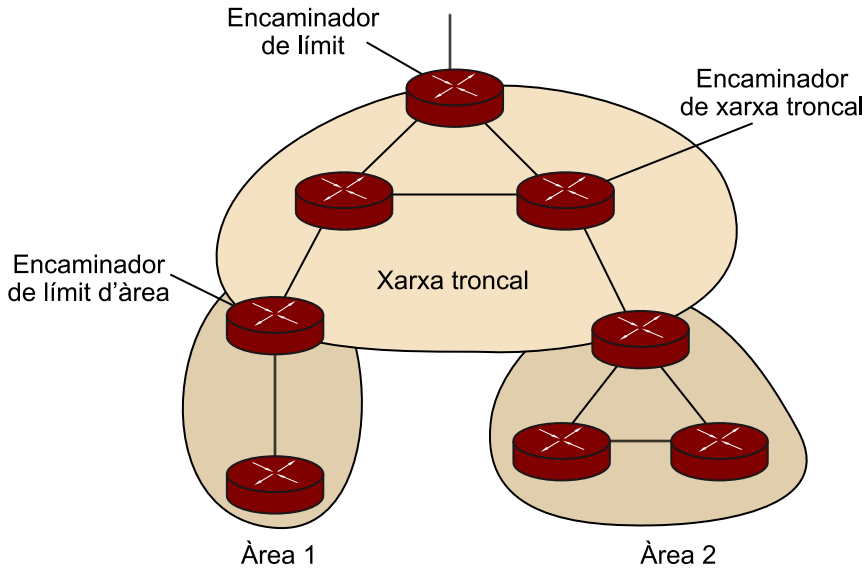
Vegeu l'algorisme de Dijkstra en el subapartat 4.1 d'aquest mòdul didàctic.

Com que OSPF està pensat per a xarxes bastant més grans que RIP, per tal de proporcionar un protocol escalable, OSPF permet la divisió de la xarxa en àrees. Cada àrea és un conjunt d'encaminadors que intercanvien informació OSPF entre si. Dins de cada àrea hi ha un encaminador d'àrea de vora[55] que agrega tota la informació de l'àrea a la resta de la xarxa. Cada encaminador

(55)En anglès, *area border router*.

d'àrea de vora està connectat a la columna vertebral de l'AS, que s'encarregarà, també utilitzant OSPF (o algun altre protocol), d'interconnectar les diferents àrees. La figura 23 en mostra un exemple.

Figura 23. Exemple de divisió per àrees en OSPF



Un punt important que cal considerar és que OSPF deixa a l'administrador de la xarxa la tasca d'especificar quins són els costos dels enllaços; així, es pot triar que siguin tots 1 per la política de menor nombre de salts, però també es poden prioritzar els enllaços amb més amplada de banda si es vol.

Atès que OSPF va ser pensat per a substituir RIP, era obligatori que tingués mecanismes de seguretat, i per això OSPF permet autenticar les peticions OSPF, per tal d'evitar atacs malintencionats d'anuncis de xarxes falses. D'aquesta manera, els encaminadors OSPF de la xarxa utilitzen una clau secreta (decidida per l'administrador) que permet xifrar el *hash* generat a partir de la petició per a garantir que ningú que no tingui la clau pugui generar peticions falses.

El darrer punt d'innovació d'OSPF sobre RIP va ser la introducció del que avui dia es coneix com a *load-balancing*. Així, si hi ha dues rutes amb el mateix cost cap a la mateixa destinació, el sistema permet que el trànsit es vagi balancejant entre els enllaços d'igual cost.

5.3. BGP

Els protocols d'encaminament vistos fins ara comparteixen un problema, que és el fet que a Internet hi ha massa nodes per a poder-los encaminar a partir de l'adreça IP, fet que provoca que els encaminadors del nucli de la Xarxa haurien de tenir bilions d'entrades a la seva taula d'encaminament. Per tal de simplificar això s'utilitzen les adreces per prefixos (CIDR), com ja hem vist anteriorment. Però tot i això, la informació d'encaminament continua essent massa elevada per a enviar-la directament. És per això que apareix el BGP.

De tots els protocols d'encaminament presents a Internet el més complex, però també el més utilitzat, és BGP. Aquest protocol s'utilitza principalment per a intercanviar informació d'encaminament entre dominis diferents (entre sistemes autònoms diferents).

Així doncs, BGP anunciarà prefixos de xarxa entre sistemes autònoms veïns, fet que implica que l'abstracció que es fa de la xarxa és en el nivell del sistema AS, i no pas en el nivell d'encaminador, com havíem vist fins ara. Això redueix enormement la quantitat d'informació per transmetre entre els diferents dominis.

AS

El sistema autònom (AS, per la sigla anglesa d'*autonomous system*) és un conjunt de xarxes i encaminadors IP administrats per una sola entitat (o, a vegades, més d'una) qe comparteixen una política d'encaminament comuna.

Abans d'entrar en una descripció més exhaustiva és important tenir clars alguns conceptes que fan de BGP l'estàndard *de facto* a Internet. Els protocols d'encaminament vistos fins ara desaven l'identificador del pròxim salt (normalment la IP) que ens permetia enviar el paquet cap a la destinació. Per contra, BGP el que desa és tot el camí, però en comptes de desar les adreces IP desa el que es coneix com a AS-Path, que és la llista d'AS que s'han de travessar per a arribar a la destinació. Pel fet de tenir tota la llista de dominis per creuar, BGP pot evitar de manera eficient bucles als camins, i com les connexions interdomini acostumen a ser totes amb amplades de banda molt grans, es pot utilitzar la política de salts mínims per a arribar a la destinació. Cal notar que quan es parla de salts mínims amb BGP ens referim als salts a escala d'AS, i no d'encaminador físic, cosa que ens permet obviar els diferents encaminadors interns que tingui l'AS que rep els nostres paquets.

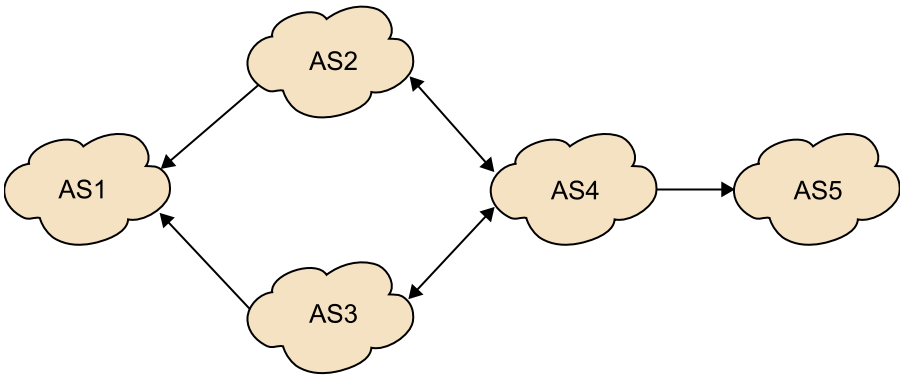
Si tota la feina que hagués de fer BGP fos l'exposada fins ara, el protocol seria força senzill, però el fet que BGP s'utilitza en connexions interdomini (eBGP[56]), força que unitats d'administració diferents (companyies que poden ser competència) hagin d'intercanviar informació interna, com per exemple quines connexions es tenen cap a l'exterior, i ha provocat que BGP, a part d'intercanviar informació d'encaminament, s'hagi de dissenyar, per tal d'evitar donar més informació de la necessària, amb el que es coneix com a *forwarding policies*. Per tant, BGP anunciarà les rutes als seus veïns depenent de les relacions comercials que tinguin, i així es distingeixen tres tipus de relacions entre dominis: clients, proveïdors i parells[57].

(56)*eBGP* és la sigla d'*external BGP*.

(57)En anglès, *peers*.

Un client és un AS que paga a un proveïdor per tal que li ofereixi trànsit de la seva informació a la resta de la xarxa. Un proveïdor és aquell que ofereix els seus serveis (i connexions) als seus clients. Finalment, un parell fa referència a aquella relació existent entre dos dominis, per als quals es dóna trànsit de només un subconjunt de rutes.

Figura 24. Relacions entre diferents AS



Per a entendre millor aquests conceptes, la figura 24 mostra una xarxa d'exemple, en què AS1 és client d'AS2 i AS3. AS2 i AS4 són parells (doble fletxa), igual que AS3 i AS4. Per la seva banda, AS4 és proveïdor d'AS5 (o AS5 és client d'AS4).

En vista d'aquesta xarxa s'han de tenir en compte les consideracions següents: aS1 està pagant tant a AS2 com a AS3 per tenir un servei, cosa que vol dir que AS1 ha d'evitar enviar trànsit d'AS2 cap a AS3 i viceversa, ja que d'aquesta manera AS2 tindria trànsit gratuït (*free-ride*) cap a AS3. Per la seva banda, AS2 i AS4 tenen una relació de parells[58], cosa que vol dir que han arribat a un acord econòmic per a compartir les despeses de l'enllaç que els uneix, i això implica que AS4 voldrà enviar trànsit entre AS2 i AS5 de manera gratuïta però no entre AS2 i AS3, perquè tots dos són parells i no clients (un altre cas de *free-ride*).

(58)En anglès, *peering*.

En resum, les polítiques d'anunci de rutes estan determinades per les relacions comercials entre els veïns amb les normes bàsiques següents:

- Un proveïdor anunciarà als seus clients totes les rutes conegudes per ell.
- Un proveïdor mai no anunciarà les rutes d'un parell (o d'un altre proveïdor) a un altre parell (o proveïdor); només les dels seus clients.
- Un client mai no anunciarà a un proveïdor i a un parell les adreces dels seus proveïdors; només anunciarà els seus clients.

Amb tot, a Internet ens trobem tres nivells de jerarquia: aquells AS que no tenen proveïdors, que s'anomenen *Tier-1*; aquells que tenen clients, proveïdors i potser parells, anomenats *Tier-2* i *Tier-3* segons com estiguin de ben connectats dins de la jerarquia, i finalment els *Stub AS*, que són els que no tenen clients (la majoria) i que no fan trànsit, o sigui, que són origen o final de la comunicació.

Cal notar que les interconnexions lògiques entre AS no sempre corresponen a enllaços físics, i això vol dir que moltes vegades hi haurà diversos enllaços entre els dos AS, i d'altres cops hi haurà encaminadors intermedis que faran trànsit a escala d'IP entre els AS.

Un cop un AS ha après una ruta enviarà la informació als altres encaminadors interns al domini, per tal que sàpiguen com poden arribar a les diferents destinacions; això es fa mitjançant sessions BGP a escala intradomini (iBGP), mentre que al mateix temps els encaminadors de vora, depenent de les polítiques d'encaminament presents, avisaran els AS veïns de les noves destinacions apreses.

Lectures recomanades

A. S. Tanenbaum (2003). *Redes de computadores* (4a. ed.). Pearson.

J. F. Kurose; **K. W. Ross** (2005). *Computer networking: a top-down approach featuring the Internet*. Addison-Wesley.

Resum

Aquest mòdul ha descrit la capa de xarxa. Aquesta capa té per objectiu donar connectivitat d'extrem a extrem entre dos punts qualssevol independentment de la tecnologia utilitzada. Així, en el nivell de xarxa es defineixen dues entitats, els encaminadors i els equips finals, en què els primers tenen la tasca de fer arribar la informació als segons. Això es pot aconseguir mitjançant un protocol. Actualment IP és el protocol per excel·lència a Internet, i permet l'intercanvi d'informació dins de la Xarxa. IP requereix que cada equip tingui una adreça única a escala global; la versió IPv4 del protocol defineix aquesta adreça amb 32 bits, i la nova versió IPv6 ho fa amb 128. Atesa la gran quantitat d'adreces i de nodes existents a la Xarxa, IP necessita simplificar la gestió per mitjà de les xarxes i les subxarxes, que són conjunts d'adreces IP pertanyents a la mateixa unitat administrativa, i així es distribueix la gestió de la Xarxa.

A banda d'identificar els nodes de la Xarxa també és necessari tenir mecanismes per a fer arribar els datagrames entre dos punts distants de la Xarxa; per això IP s'ajuda dels protocols d'encaminament, que defineixen la política que cal seguir per a l'enviament de la informació. A escala interdomini el protocol d'encaminament més utilitzat és BGP, que per mitjà dels sistemes autònoms dóna un nivell suficient d'abstracció de la xarxa que fa viable fer l'encaminament a escala global. A més petita escala, hi ha els protocols basats en l'estat de l'enllaç, com per exemple OSPF, que s'encarreguen de fer l'encaminament a escala intradomini, i completen així els mecanismes d'encaminament i permeten l'intercanvi d'informació dins d'Internet de manera transparent als usuaris finals.

Bibliografia

Dijkstra, E. W. (1959) "A note on two problems in connexion with graphs". *Numerische Mathematik* (núm. 1, pàg. 269-271).

Gartner, F. C. (2003). "A survey of self-stabilizing spanning tree construction algorithms".

Kurose, J. F.; Ross, K. W. (2005). *Computer Networking: a Top-Down Approach Featuring the Internet.* Addison-Wesley.

Tanenbaum, A. S. (2003). *Redes de computadores* (4a. ed.). Pearson.