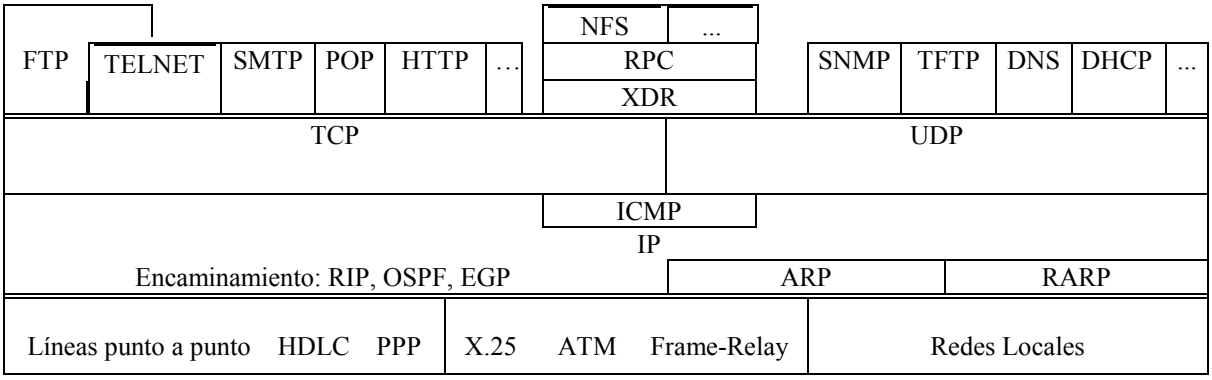
TEMA 6: EL PROTOCOLO TCP/IP

INDICE TEMA 6

1. UNA FAMILIA DE PROTOCOLOS

La denominación TCP/IP recoge la descripción de una serie de protocolos, la topología y la arquitectura que sirven de base para una red de área extensa (WAN) como es el caso de Internet. Entre los protocolos descritos bajo esa denominación se encuentran el IP (Internet Protocol) y el TCP (Transmision Control Protocol) junto con varios más. Todos ellos sirven de soporte a un conjunto de aplicaciones y servicios de aplicación, muy conocidos por su utilización en la red Internet. La descripción de todos los elementos que forman parte de la arquitectura TCP/IP y la mayor parte de las aplicaciones que hacen uso de ella, se encuentran recogidos como estándares "de facto" en los RFCs (Request For Comments). Se trata de documentos manejados por la comunidad de Internet donde se incluyen los protocolos y estándares de la red Internet, las propuestas de estándar, documentos puramente informativos, etc.



2. DIRECCIONES IP Y ENCAMINAMIENTO MEDIANTE ROUTERS

Para hacer un sistema de comunicación universal, se necesita un método de identificar computadoras aceptado globalmente. Cada computadora tendrá su propio identificador, conocido como dirección IP o dirección Internet.

Las direcciones IP se representan como cuatro enteros decimales separados por puntos, donde cada entero da el valor de un octeto de la dirección. Así, por ejemplo, la dirección de 32 bits 10000000 00001010 00000010 00011110 se escribe 128.10.2.30.

Puede pensarse en la red Internet como cualquier otra red física. La diferencia está en que la red Internet es una estructura virtual implementada enteramente en "software". Por tanto, los diseñadores fueron libres de escoger los tamaños y formatos de los paquetes, las direcciones, las técnicas de distribución de paquetes, etcétera. Para las direcciones, se escogió un sistema análogo al direccionamiento en redes físicas, en el cual al "host" (elemento conectado a la red: computadora, estación de trabajo o cualquier otro tipo de dispositivo) se le asigna un número entero de 32 bits como identificador, llamado dirección IP. Estos enteros están cuidadosamente escogidos para hacer el proceso de encaminamiento o "routing" eficiente. Las direcciones IP codifican la identificación de la red a la que el "host" se encuentra conectado, así como la identificación de ese "host" dentro de la red. Por tanto, todas las computadoras conectadas a una misma red tienen en su número de dirección una serie de bits comunes (evidentemente, los bits de identificación de red).

Cada dirección IP es un par de identificadores (redid,hostid), donde *redid* identifica una red y *hostid* identifica a una computadora dentro de esa red. En la práctica, hay tres clases distintas de direcciones (clases A,B y C), como se muestra en las figuras.

Clase A (1.0.0.0 a 126.255.255.255)

0	redid (7 bits)	hostid (24 bits)

Clase B (128.0.0.0 a 191.255.255.255)

1	0	Redid (14 bits)	Hostid (16 bits)

Clase C (192.0.0.0 a 223.255.255.255)

1	1	0	redid (21 bits)	Hostid (8 bits)

Dada una dirección IP, se puede determinar su clase a partir de los tres bits de orden más alto, siendo sólo necesario dos bits para distinguir entre las clases primarias. Las direcciones de clase A se usan para redes que tienen desde 65.364 "hosts" hasta 16.777.214, utilizando 7 bits para *redid* y 24 bits para *hostid*.. Las direcciones de clase B se usan para redes de tamaño intermedio, que tienen entre 254 y 65364 "hosts", localizando 14 bits en *redid* y 16 bits en *hostid*. Finalmente, las redes de clase C, que tienen 254 "hosts" o menos, utilizan 21 bits para *redid* y solamente 8 bits para *hostid*.

En cada red de clase A, B o C el administrador de la misma puede usar los bits del *hostid* para identificar cada una de los "hosts" de su organización o incluso puede hacer uso de parte de los bits correspondiente al *hostid* para a su vez denominar distintas subredes dentro de su organización. En cualquier caso, la dirección de "host" 0 (todos los bits a cero en el *hostid*) no se asigna a ningún "host" y se reserva para la identificación de la red, y la dirección con todos los bits a 1 en el *hostid* es la dirección broadcast de la red.

Existen otras dos clases para usos especiales:

- Clase D: redes multicast, desde 224.0.0.0 hasta 239.255.255.255 (RFC3171)
- Clase E: experimental, desde 240.0.0.0 hasta 254.255.255.255 (RFC 1700)

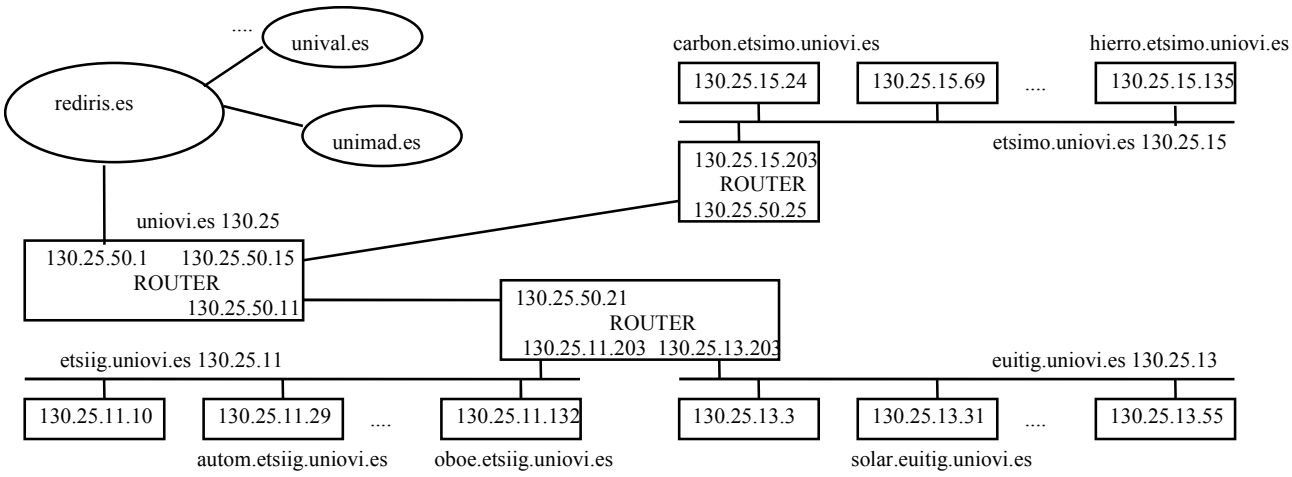
También se reservan ciertas direcciones para determinados usos (RFC 3330), entre las cuales están:

- **Direcciones de bucle:** Todas las direcciones de la red de clase A 127.0.0.0 se utilizan para funciones de bucle de prueba dentro del propio "host" (se interpretan como direcciones del propio "host").
- **Direcciones para redes privadas (intranets):** Estas direcciones no se usan como globales en Internet (RFC 1918). Se reservan para la creación de redes TCP/IP no conectadas a Internet y por lo tanto se pueden utilizar a voluntad. Para conectar estas intranets a Internet es necesario utilizar algún tipo de servicio NAT (Network Address Traslation, RFC 1631) que convierta estás direcciones en otra u otras válidas en Internet.
  - Clase A: direcciones de la 10.0.0.0 a la 10.255.255.255 (una red de clase A, 10.0.0.0/8)
  - Clase B: direcciones de la 172.16.0.0 a la 172.31.255.255 (16 redes de clase B, 172.16.0.0/12)
  - Clase C: direcciones de la 192.168.0.0 a la 192.168.255.255 (256 redes de clase C, 192.168.0.0/16)

Address Block	Present Use	Reference
0.0.0.0/8	"This" Network	[RFC1700]
10.0.0.0/8	Private-Use Networks	[RFC1918]
14.0.0.0/8	Public-Data Networks	[RFC1700]
24.0.0.0/8	Cable Television Networks	--
39.0.0.0/8	Reserved but subject to allocation	[RFC1797]
127.0.0.0/8	Loopback	[RFC1700]
128.0.0.0/16	Reserved but subject to allocation	--
169.254.0.0/16	Link Local	--
172.16.0.0/12	Private-Use Networks	[RFC1918]
191.255.0.0/16	Reserved but subject to allocation	--
192.0.0.0/24	Reserved but subject to allocation	--
192.0.2.0/24	Test-Net	
192.88.99.0/24	6to4 Relay Anycast	[RFC3068]
192.168.0.0/16	Private-Use Networks	[RFC1918]
198.18.0.0/15	Network Interconnect Device Benchmark	[RFC2544]
223.255.255.0/24	Reserved but subject to allocation	--
224.0.0.0/4	Multicast	[RFC3171]
240.0.0.0/4	Reserved for Future Use	[RFC1700]

Los "routers" basan sus decisiones de encaminamiento en el *redid* del “host” de destino, es decir en la red a la que van destinados los datos. Los "routers", por tanto, deciden dónde van a mandar los datos basándose en la red destino y no en la computadora específica a la que van destinados los datos, lo que disminuye las necesidades de memoria de los "routers" a medida que aumenta el número de estaciones conectadas a la red Internet.

Sin embargo, siguiendo el criterio explicado hasta ahora no sería posible, por ejemplo, dar una dirección a un "router" que esté unido a dos redes, puesto que el *redid* de las dos redes no es el mismo. La solución a esto es asignar a este tipo de máquinas conectadas a más de una red, varias direcciones, una por cada red a que estén conectadas.



Para que al usuario le resulte más cómodo recordar la identificación de los distintos “hosts” y dominios, se les dan nombres o alias. En algunos casos un nombre hace referencia a más de una dirección IP (generalmente varias direcciones de dominios) y en otras varios nombres (alias) hacen referencia a una misma dirección IP de “host” o de dominio. Esto obliga a que exista una base de datos en el “host” que relacione nombres lógicos con direcciones IP.

Generalmente esta base de datos es un fichero de tipo texto que solo contiene unos pocos de los nombres existentes dentro de la red Internet. Sería inviable que cada “host” tuviese una base de datos completa y actualizada con todos los nombres y direcciones IP.

Si un "host" no tiene en su propia base de datos la identificación de un "host", consulta a un servidor DNS (*Domain Name Service)*. Se trata de un "host" que mantiene la base de datos para uno o varios dominios y que da servicio de nombres a los ordenadores de ese dominio, que a su vez deben conocer cual o cuales son los servidores DNS que tienen más cercanos. Si el servidor DNS no contiene la identificación del "host" que le han solicitado, consulta a su vez a otros servidores DNS de la red Internet hasta encontrarlo.

Las direcciones IP y los nombres de dominio en Internet son asignadas por los NIC (*Network Information Center*), organizaciones que dependen de la IANA (*Internet Assigned Numbers Authority,* www.iana.org). Estas autoridades sólo asignan la parte de la dirección o nombre de dominio que identifica a la red, delegando la asignación de las direcciones de los "hosts" a la organización que ha formulado la petición de conexión a la red Internet. El registro delegado en España depende del Centro de Comunicaciones CSIC RedIRIS ES-NIC en cuya página www.nic.es se puede encontrar información sobre otras organizaciones involucradas en estas asignaciones como ICANN (*Internet Corporation for Assigned Names and Numbers*, www.icann.net) o RIPE NCC (*Réseaux IP Européens Network Coordination Centre*, www.ripe.net)

2.1 CIDR

Para evitar que Interrnet se quedara sin direcciones IP rápidamente, se ideó CIDR (ClasslessInterDomain Routing). El sistema de clases original desaprovecha gran cantidad de direcciones cuando a una determinada organización se le asigna una clase A o B y utiliza un porcentaje mínimo de las mismas. Sólo un pequeño porcentaje del espacio de direcciones de las clases A y B está asignado a una computadora en Internet.

El nuevo sistema fue propuesto por primera vez en 1992 y se denominó Supernetting. Con este sistema las máscaras de red se extienden de manera que, por ejemplo, una dirección de red y máscara de subred pueden especificar varias subredes de clase C. Si se necesitasen 1000 direcciones aproximadamente se podrían juntar 4 subredes de clase C.

```
192.60.128.0    (11000000.00111100.10000000.00000000)  Dirección de subred de clase C
192.60.129.0    (11000000.00111100.10000001.00000000)  Dirección de subred de clase C
192.60.130.0    (11000000.00111100.10000010.00000000)  Dirección de subred de clase C
192.60.131.0    (11000000.00111100.10000011.00000000)  Dirección de subred de clase C
-------------------------------------------------------
192.60.128.0    (11000000.00111100.10000000.00000000)  Dirección de subred de la superred
255.255.252.0   (11111111.11111111.11111100.00000000)  Máscara de subred
192.60.131.255  (11000000.00111100.10000011.11111111)  Dirección broadcast
```

En este ejemplo la subred 192.60.128.0 incluye todas las direcciones desde 192.60.128.0 hasta 192.60.131.255. La representación binaria de la máscara de la subred muestra que la parte de la dirección que representa a la red tiene 22 bits de longitud (bits a 1 en la máscara) y la parte que representa al host tiene 10 bits de longitud (bits a 0 en la máscara).

Bajo CIDR, la notación de la máscara de la subred se simplifica. Para el ejemplo anterior, en lugar de escribir la dirección y la máscara de la subred así:

192.60.128.0, Máscara de subred 255.255.252.0

la dirección de la subred se escribe simplemente así:

192.60.128.0/22

lo que indica la dirección inicial de la subred y el número de bits a 1 (22) en la máscara de la subred (11111111.11111111.11111100.00000000).

La direcciones del sistema de clases original se pueden representar fácilmente en la notación de CIDR (clase A = /8, clase B = /16, y clase C = /24).

Es prácticamente imposible que un individuo o una compañía pueda obtener su propio bloque de direcciones IP sino es a través de un ISP (Internet Service Provider). La razón es el crecimiento de la tabla de encaminamiento de Internet. En 1994 había menos de 5.000 rutas de red en toda Internet. Cinco años más tarde, había 90.000. Utilizando CIDR, los mayores ISPs obtienen grandes bloques del espacio de direcciones (normalmente con una máscara de subred de /19 o incluso menor). Los clientes de los ISPs (frecuentemente otros ISPs más pequeños) obtienen subredes del conjunto que tiene el ISP grande. De esta manera, todos los clientes del ISP grande son accesibles a través de una sola ruta de red (y los clientes de estos y así sucesivamente).

Se espera que CIDR mantenga Internet con suficientes direcciones IP disponibles hasta que llegue IPv6 con direcciones de 128 bits. Los detalles completos de CDIR están documentados en el RFC1519, realizado en septiembre de 1993.

2.2 Direcciones broadcast

Se ha dicho que una de las ventajas de las direcciones IP es que codifican información sobre la red, con lo que se simplifica el encaminamiento. Otra ventaja es que una dirección IP puede referirse tanto a redes como a "hosts". Por convenio, *hostid* 0 no se asigna nunca a un computador, si no que una dirección IP con *hostid* igual a 0, se usa para referirse a la propia red.

Otra ventaja importante del direccionamiento IP es que soporta la dirección *broadcast*, que se refiere a todos los "hosts" conectad os a la red. De acuerdo con el convenio, cualquier *hostid* con todos los bits valiendo 1 se reserva para *broadcast* En muchas tecnologías de red (por ejemplo en la red Ethernet), la transmisión *broadcast*es tan eficiente como una transmisión normal; en otras redes se admiten las direcciones *broadcast*, aunque suponen un retraso considerable; otras redes no las admiten en absoluto. Por tanto, la existencia de direcciones *broadcast* no garantiza la eficacia de este tipo de transmisión.

De igual manera que un campo en la dirección IP con todos los bits 1 significaba "todos", el software IP interpreta un campo con todos los bits valiendo 0 como "éste". Así, una dirección con *hostid* igual a 0, se refiere a "ese" computador, y una dirección con *redid* igual a 0 se refiere a "esa" red. El uso de direcciones con *redid* igual a 0 es especialmente importante en aquellos casos en que un "host" quiere comunicarse sobre una red, pero todavía no sabe su dirección IP. El "host" utiliza temporalmente un *redid* igual a 0, y los otros computadores conectados a la red, interpretan esa dirección significando "esta" red. En la mayoría de los casos, las respuestas tendrán una dirección con el identificador de red especificado, permitiendo al "host" almacenarlo para futuros usos.

2.3 Orden de byte en la red

Estandarizar el orden de byte en la representación de enteros es especialmente importante, pues los paquetes TCP/IP llevan números que especifican información tal como la dirección destino, o la longitud del paquete. Estos valores deben ser perfectamente entendidos tanto por el que envía como por el que recibe. El TCP/IP resuelve el problema definiendo un orden de byte estándar para la red, que debe ser usado por todas las

máquinas en los campos binarios de los paquetes. Cada "host" debe convertir los datos de su propia representación interna al orden de byte de la red antes de enviar un paquete. Lo mismo tiene que hacer el destinatario cuando reciba un paquete. Naturalmente, el campo de datos en el paquete está exento de este estándar; cada usuario es libre de elegir el formato de datos.

El estándar TCP/IP especifica que los enteros son enviados con el byte más significativo delante; es decir, si se consideran sucesivos bytes en un paquete cuando va de una máquina a otra, los enteros tienen el byte alto más cerca del principio del paquete y el byte bajo más cerca del final.

2.4 El servicio DHCP

El DHCP (Dinamic Host Configuration Protocol, RFC 2131) sirve para configurar parámetros de los "hosts" en forma dinámica a través de la red. Estos parámetros suelen ser fundamentalmente los que necesita el sistema para poder ser operativo en una red TCP/IP.

Si no existe un servicio DHCP en la red, esta configuración ha de ser introducida manualmente en el sistema por el administrador de la red o por el usuario del equipo bajo las indicaciones del administrador. Por ejemplo, para una máquina que va a tener la dirección IP 156.35.165.129 la configuración podría ser:

```
Configuración IP básica:
        Dirección IP. . . . . . . . . . . : 156.35.165.129
        Máscara de subred . . . . . . . . : 255.255.255.0
        Puerta de enlace predeterminada   : 156.35.165.210
        Servidores DNS . . . . . . . . . .: 156.35.41.2
                                            156.35.14.2
Opcionalmente:
        Nombre del host . . . . . . . . . : prave
        Sufijo DNS principal  . . . . . . : edv.uniovi.es
```

El nombre para este "host", prave.edv.uniovi.es, sólo es necesario si se necesita que los usuarios puedan recordar con mayor facilidad su identificación, por ejemplo, si va a hacer funciones de servidor en la red.

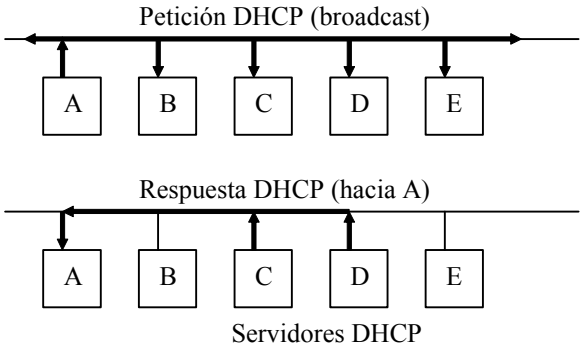
La mascara de subred indica los bits del identificador de red, 24 en este caso, y del identificador de host , 8 bits, dentro de la dirección IP.

La "puerta de enlace" es la denominación que recibe en algunos casos el encaminador o "router" hacia el que el sistema dirigirá aquellos datagramas IP que vayan hacia una dirección IP que no pertenezca a la misma subred (156.35.165.0, conocida por la combinación de la dirección IP del propio "host" y la mascara de subred)

Los servidores DNS, son las máquinas encargadas de proporcionar el servicio de nombres de dominio para esa red y cuya función se describirá más adelante.

Si existe un servidor DHCP, cuando el "host" arranca pide sus parámetros IP. La petición irá en un mensaje broadcast (dirección IP 255.255.255.255) hacia la red, ya que probablemente el "host" tampoco sabe cuál es la dirección del servidor DHCP. El servidor DHCP recibe la petición, consulta su base de datos y responde con los parámetros adecuados (de acuerdo, por ejemplo, a la dirección Ethernet de quien hace la consulta) para una configuración IP válida en esa red. El servidor DHCP puede asignar siempre la misma configuración al mismo equipo o una diferente cada vez que arranca según decida el administrador. Incluso puede denegar la contestación si no reconoce como válida o conocida la identificación (p.e. la dirección física de red) del equipo que hace la petición.

Las peticiones DHCP viajan dentro de mensajes UDP transportados en datagramas IP y estos, a su vez, insertados en el campo de datos de una trama de red (como se verá más adelante). DHCP es por lo tanto un protocolo de nivel de aplicación dentro de la arquitectura TCP/IP.



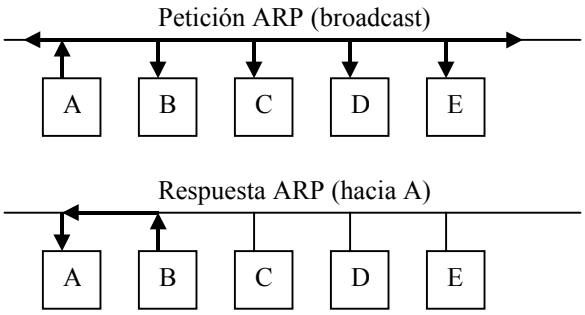
En una misma red local pueden existir varios servidores DHCP, en cuyo caso a una petición pueden llegarle varias respuestas. Es misión del administrador de la red la correcta coordinación del servicio DHCP en uno o varios servidores para que no aparezcan problemas.

El servicio DHCP ha sustituido eficientemente a protocolos anteriores como RARP (RFC 903) o BootP (RFC 951) que tenían misiones similares. En conexiones punto a punto el protocolo PPP proporciona las funciones necesarias para la configuración IP de un equipo cliente que se conecta a la red Internet.

3. EL PROTOCOLO ARP

Se ha dicho anteriormente que una dirección IP era un número de 32 bits que se asignaba a las máquinas conectadas a la red para su identificación, y que esa identificación era suficiente para enviar y recibir paquetes. Sin embargo, las máquinas conectadas a una red local pueden comunicarse sólo si conocen sus respectivas direcciones físicas. Por lo tanto, cuando un "host" quiere comunicarse con otro que está en la misma red local, necesita relacionar la dirección IP del destinatario con su correspondiente dirección física. Si el destinatario no esta el la misma red local, deberá obtener la dirección física del encaminador o "router" que le permita enviar los datagramas IP fuera de la subred en la que se encuentra. De esto se encarga el protocolo ARP.

La idea del ARP (*Address Resolution Protocol*) es sencilla. Cuando un "host" A desea comunicarse con otro B, del que conoce su dirección IP (I_B), pero no su dirección física (F_B), envía un paquete especial con dirección destino *broadcast*, pidiendo al "host" que tiene como dirección IP I_B que responda con su dirección física F_B. Todos los "hosts" conectados a la red reciben el paquete ARP, pero sólo el "host" B, que es al que iba dirigida la pregunta, responde con otro paquete ARP, enviando su dirección física.



Así, una vez completado el intercambio de información mediante el protocolo ARP, el "host" conoce la dirección física del otro con el que quiere comunicarse, de manera que puede enviarle los siguientes paquetes a él directamente. La experiencia demuestra que merece la pena mantener una tabla dinámica en la memoria volátil con las direcciones IP y las correspondientes direcciones físicas de los computadores con los que se ha establecido comunicación más recientemente, ya que, normalmente, la comunicación requiere el envío de varios paquetes. Así, cuando un "host" quiere comunicarse con otro, antes de enviar un paquete ARP, busca en la memoria para ver si tiene su dirección física, con lo que se reducen costes de comunicación. Sin embargo, las entradas de la tabla se eliminan si no son utilizadas durante un cierto tiempo o cuando el computador se apaga. Se evitan así problemas de comunicación si alguno de los ordenadores registrados ha cambiado de dirección física por avería de su interfaz de comunicaciones o cualquier otra eventualidad.

Cuando un paquete ARP viaja por la red de una máquina a otra, lo hace encapsulado en una trama del nivel de enlace, como en el ejemplo de la figura.

Preámbulo	Destino	Origen	Tipo	Mensaje ARP tratado como datos	CRC

Mensaje ARP encapsulado en una trama Ethernet

Para identificar que la trama está llevando un paquete ARP, el computador fuente, asigna un valor especial al campo de tipo en la cabecera del paquete. Cuando la trama llega al destino, el "host" examina el campo tipo para determinar qué contiene esa trama. En el caso de la red Ethernet los paquetes ARP tienen un campo de tipo de valor 0806h (valor en notación hexadecimal).

Al contrario que la mayoría de los protocolos, los datos en paquetes ARP no tienen un formato de encabezamiento fijo. El mensaje está diseñado para ser válido con una variedad de tecnologías de transmisión y de protocolos. El ejemplo de la figura muestra el mensaje ARP de 28 octetos usado para las redes Ethernet (en las que la dirección física tiene una longitud de 48 bits, 6 octetos) y protocolo de red IP (con dirección lógica de 32 bits, 4 octetos).

<table>
<tr><td colspan="2">HARDWARE</td><td>PROTOCOLO</td></tr>
<tr><td>HLON</td><td>PLON</td><td>OPERACIÓN</td></tr>
<tr><td colspan="3">DF ORIGEN (octetos 0-3)</td></tr>
<tr><td colspan="2">DF ORIGEN (octetos 4-5)</td><td>DL ORIGEN (octetos 0-1)</td></tr>
<tr><td colspan="2">DL ORIGEN (octetos 2-3)</td><td>DF DESTINO (octetos 0-1)</td></tr>
<tr><td colspan="3">DF DESTINO (octetos 2-5)</td></tr>
<tr><td colspan="3">DL DESTINO (octetos 0-4)</td></tr>
</table>

Formato del paquete ARP usado para redes Ethernet

A continuación se explica el significado de cada uno de los campos del paquete:

- HARDWARE: Especifica el tipo de interfaz hardware para el que el computador fuente solicita la respuesta; el valor es 1 para la red Ethernet.
- PROTOCOLO: Contiene el número del protocolo al que corresponden las direcciones lógicas.
- HLON y PLON: Especifican, respectivamente, las longitudes de la dirección física y de la dirección de protocolo.

- OPERACION: Especifica el tipo de operación que realiza el mensaje: vale 1 para una petición ARP, 2 para una respuesta ARP, 3 para una petición RARP y 4 para una respuesta RARP. (El protocolo RARP se explicará a continuación).
- DF ORIGEN: Contiene la dirección física del "host" que realiza la petición ARP.
- DL ORIGEN: Contiene la dirección lógica (o de protocolo) del "host" que realiza la petición ARP.
- DF DESTINO: Es un campo vacío en una petición ARP y contiene la dirección física del "host" al que va destinada la petición en la respuesta.
- DL DESTINO: Contiene la dirección lógica del "host" al que va destinado la petición ARP.

4. EL PROTOCOLO IP

El protocolo IP (*Internet Protocol*) define la unidad básica de transmisión de datos, y el formato exacto de todos los datos cuando viajan por una red TCP/IP. Además, el protocolo IP incluye una serie de reglas que especifican como procesar los paquetes y como manejar los errores. El protocolo IP se basa en la idea de que los datos se transmiten con un mecanismo no fiable y sin conexión. Un mecanismo no fiable quiere decir que un paquete puede perderse, duplicarse o enviarse a un destino diferente del deseado. El mecanismo es sin conexión porque cada paquete se trata independientemente de los otros. Paquetes de una secuencia enviados de una máquina a otra pueden ir por distintos caminos, e incluso unos pueden alcanzar su destino mientras que otros no. El protocolo incluye también la idea del encaminamiento (o *routing*) de paquetes.

4.1 El datagrama IP

El datagrama es la unidad básica de transmisión de datos en la red Internet. El datagrama, al igual que las tramas en las redes físicas, se divide en encabezamiento y campo de datos. El encabezamiento contiene las direcciones IP de la fuente y el destino.

La longitud máxima de un datagrama es de 65.536 octetos (64 Kbytes). Sin embargo, para viajar de una máquina a otra, los datagramas lo hacen en el campo de datos de tramas de enlace, por lo tanto los datagramas de longitud excesiva deben ser divididos en fragmentos que quepan en las tramas. Cada fragmento perteneciente a un mismo datagrama tiene el mismo número de identificación que el datagrama original, con lo que es posible su reconstrucción.

<table>
<tr><td colspan="8">0 4 8 16 19 24 31</td></tr>
<tr><td colspan="2">VER</td><td colspan="2">LON</td><td colspan="2">TIPO SERVICIO</td><td colspan="2">LONGITUD TOTAL</td></tr>
<tr><td colspan="4">IDENTIFICACION</td><td>FLAGS</td><td colspan="3">DESPLAZAMIENTO</td></tr>
<tr><td colspan="2">TIEMPO</td><td colspan="2">PROTOCOLO</td><td colspan="4">CHECKSUM ENCABEZAMIENTO</td></tr>
<tr><td colspan="8">DIRECCION IP FUENTE</td></tr>
<tr><td colspan="8">DIRECCION IP DESTINO</td></tr>
<tr><td colspan="6">OPCIONES</td><td colspan="2">RELLENO</td></tr>
<tr><td colspan="8">DATOS</td></tr>
<tr><td colspan="8">....</td></tr>
</table>

Preámbulo	Destino	Origen	Tipo	Datagrama IP (o fragmento) tratado como datos	CRC

Datagrama IP encapsulado en una trama Ethernet

A continuación se describen los campos del datagrama IP:

- VER: Este campo de 4 bits se usa para especificar la versión del protocolo IP, y se usa para que destino, fuente y "routers" entre ellos, están de acuerdo en el tipo de datagrama. Si los estándares cambian, las máquinas rechazarán datagramas con versión de protocolo distinta de las de ellas.
- LON:Este campo de 4 bits especifica la longitud del encabezamiento del datagrama medido en palabras de 32 bits. El encabezamiento más común, sin incluir opciones, tiene este campo con valor igual a 5.
- TIPO DE SERVICIO: Este campo de 8 bits especifica como debe ser manejado el datagrama, y se divide en los 5 campos que muestra la figura:

PRIORIDAD (3 bits)	D(1 bit)	T(1 bit)	R(1 bit)	NO USADO

PRIORIDAD: Estos tres bits especifican la prioridad del datagrama, con valores en el rango de 0 (prioridad normal) a 7 (datagramas de control de la red).

Bits D,T y R especifican el tipo de servicio que el datagrama solicita. Cuando están a 1, el bit D solicita bajo retraso, el bit T alta velocidad de transmisión de la información y el bit R solicita alta fiabilidad. Por supuesto, la red no puede garantizar el tipo de servicio requerido si, por ejemplo, el camino al destino no tiene esas propiedades. Estos bits se usan por los "routers" cuando pueden escoger entre varios posibles caminos para un datagrama; eligiendo el que mejor se adapte a los servicios solicitados.

- LONGITUD TOTAL: Este campo da la longitud total del fragmento del datagrama, incluido el encabezamiento, medida en octetos. El tamaño del campo de datos puede calcularse a partir de este campo y del campo LON.
- IDENTIFICACIÓN: Este campo, junto con los de FLAGS y DESPLAZAMIENTO se utilizan para el control de fragmentación. Su propósito es permitir al "host" destino unir todos los fragmentos que pertenecen a un mismo datagrama, y que le pueden llegar fuera de orden. A medida que los fragmentos llegan, el "host" destino utiliza el campo IDENTIFICACION, junto con la dirección fuente para identificar a qué datagrama pertenece ese fragmento. Los computadores normalmente generan un campo IDENTIFICACION único, incrementando un contador cada vez que forman un datagrama.
- FLAGS: Los dos bits más bajos de los tres bits del campo FLAGS controlan la fragmentación. El primero especifica si el datagrama puede ser fragmentado (si está a 1, no puede serlo). El bit más bajo de FLAGS indica, si está a 1, que este fragmento no es el último del datagrama. Este bit es necesario pues el campo LONGITUD TOTAL del encabezamiento

se refiere a la longitud del fragmento, y no a la longitud total del datagrama.

- DESPLAZAMIENTO: Especifica el desplazamiento del fragmento en el datagrama original, medido en unidades de 8 octetos, empezando con desplazamiento 0.

  Para ensamblar un datagrama, el "host" destino debe obtener todos los fragmentos; desde el de desplazamiento cero hasta el de desplazamiento más alto. Si uno o más fragmentos se pierden, el datagrama entero debe ser descartado.
- TIEMPO: Especifica el tiempo máximo que se le permite al datagrama permanecer en la red IP; así se evita que datagramas perdidos viajen por la red indefinidamente. Es difícil estimar el tiempo exacto porque los "routers" normalmente no saben el tiempo que requieren para la transmisión las redes físicas. Para simplificar, "host" y "routers" suponen que cada red utiliza una unidad de tiempo en la transmisión; así deben decrementar el valor de este campo en uno cada vez que procesen un encabezamiento de datagrama. Si el valor del campo llega a cero el datagrama se destruye y se devuelve un mensaje de error ICMP.
- PROTOCOLO: Especifica el tipo de protocolo de alto nivel que soporta los datos que lleva el datagrama. Los valores de este campo para distintos protocolos los asigna una autoridad central (el NIC). Algunos de ellos aparecen en la siguiente tabla:

PROTOCOLO	VALOR
ICMP	1
TCP	6
EGP	8
UDP	17

- CHECKSUM DEL ENCABEZAMIENTO: Asegura que el encabezamiento no tiene errores. El "checksum" se forma tratando el encabezamiento como una secuencia de enteros de 16 bits. Se suma con aritmética de complemento a uno, el complemento a uno de todos ellos. A efectos de calcular el "checksum" se supone que el campo CHECKSUM DEL ENCABEZAMIENTO tiene valor cero. Como sólo se chequean errores en el encabezamiento, los protocolos de nivel superior deberán añadir otro tipo de comprobación para detectar errores en los datos.
- DIRECCINES IP FUENTE Y DESTINO: Contienen las direcciones IP de 32 bits de los "hosts" fuente y destino del datagrama respectivamente.
- OPCIONES: No es un campo necesario en todos los datagramas. Se incluyen normalmente para chequear o depurar la red. La longitud de las opciones varía dependiendo de cuáles de éstas se seleccionan. Algunas opciones son de longitud un octeto, mientras que otras son de longitud variable. Cada opción consiste en un octeto de código de opción, un octeto de longitud y una serie de octetos para la opción. El código de opción se divide en tres campos, como aparece en la figura.

COPIA (1 bit)	CLASE DE OPCION(2 bits)	NUMERO DE OPCION(5 bits)

Cuando el bit COPIA está a 1, especifica que la opción sólo debe ser copiada al primer fragmento, y no a los demás.

Los campos CLASE DE OPCION y NUMERO DE OPCION, especifican la clase general de la opción y dan la opción específica dentro de esa clase. En la tabla siguiente se muestra la asignación de las clases.

Clase de opción	Significado
0	Control de datagrama o red
1	Reservado para futuro uso
2	Depuración y medida
3	Reservado para futuro uso

La tabla siguiente muestra las posibles opciones que pueden acompañar a un datagrama y da su clase y número de opción. La mayoría de ellas se usan para propósitos de control.

Clase de opción	Número de opción	Longitud	Descripción
0	0	1	Fin de lista de opción. Usado si las opciones no acaban al final del datagrama.
0	1	1	No operación.
0	2	11	Restricciones de seguridad y manejo.
0	3	variable	Encaminamiento fuente impreciso. Usado para encaminar un datagrama a lo largo de un camino fijado.
0	7	variable	Grabar ruta. Usado para localizar el camino seguido.
0	8	4	Secuencia identificadora. Usado para llevar una secuencia SATNET identificadora.
0	9	variable	Encaminamiento fuente estricto. Usado para encaminar un datagrama por una vía determinada.
2	4	variable	Tiempos Internet. Usado para grabar tiempos durante la ruta.

- RELLENO: Representa octetos conteniendo ceros, que son necesarios para que el encabezamiento del datagrama sea un múltiplo exacto de 32, ya se había visto que el campo de longitud del encabezamiento se especificaba en unidades de palabras de 32 bits.
- DATOS: Es la zona de datos del datagrama.

4.2 Protocolos de encaminamiento IP

Hay que distinguir entre los protocolos de nivel de red, como IP, que son los que definen el esquema de direccionamiento y el formato de las unidades de datos, y los protocolos que establecen como se encaminan dichos paquetes a través de la red. Estos protocolos definen unos procedimientos que regulan el intercambio de la información que comparten los routers, básicamente la contenida en sus tablas de rutas e información sobre el estado del enlace entre ellos. Cada entrada en la tabla de rutas especifica la porción de red de la dirección destino y la dirección del siguiente router a través de la cuál dicha red se puede alcanzar.

4.2.1 Protocolos interiores

Son aquellos diseñados para funcionar en el ámbito interno de una comunidad de usuarios que opera un conjunto de redes de forma autónoma y que por tanto tiene libertad para diseñar su arquitectura de conexión interna. Pueden emplear algoritmos de vector distancia o de estado de enlace:

a) Algoritmo de vector distancia

El encaminamiento basado en algoritmos de vector distancia es muy simple. Mantiene una lista de rutas en una tabla, donde cada entrada identifica una red de destino y da la distancia a esa red medida en saltos (routers intermedios). El protocolo clásico de este tipo es RIP (Routing Information Protocol) ampliamente utilizado en entornos de redes de área local.

RIP distingue entre dispositivos activos que difunden sus tablas de rutas a través de la red y pasivos que se limitan a escuchar y actualizar sus propias tablas a partir de la información que reciben. Típicamente, los dispositivos activos son los routers y los pasivos los servidores de la red. Aunque en el caso de tener RIP configurado como algoritmo estático, un router puede también actuar como dispositivo pasivo y no propagar información.

b) Algoritmo de estado de enlace

El encaminamiento basado en algoritmos de estado de enlace proporciona un mecanismo por el cuál cada router comunica a los demás el estado de todas sus líneas. De este modo todos los routers tienen la misma información. Cuando el estado de una línea cambia, los demás routers son informados automáticamente. El protocolo OSPF (Open Shortest Path First) es de este tipo.

4.2.2 Protocolos exteriores

Para la conexión entre sistemas autónomos se definió el protocolo EGP (Exterior Gateway Protocol), hoy día ya obsoleto. Actualmente el estándar para el intercambio de información entre sistemas autónomos en Internet es el BGP4, que permite definir políticas de encaminamiento entre sistemas autónomos y soporta CIDR (Classless InterDomain Routing), es decir, encaminamiento basado únicamente en prefijos de routing (dirección de red, *redid*), sin tener en cuenta la tradicional distinción en clases A, B y C ya superada.

El uso de CIDR y BGP4 es lo que ha permitido a Internet seguir funcionando, a pesar de su espectacular crecimiento, al ser posible agregar los bloques de redes contiguas asignados a cada proveedor de acceso, resumiendo esta información en la frontera de cada sistema autónomo de cara al exterior. Con lo cuál las tablas de encaminamiento en Internet se reducen considerablemente.

Estos protocolos son complejos y su necesidad sólo se plantea en casos especiales, como por ejemplo, conexiones entre proveedores de servicios Internet.

Lo habitual es que para la conexión entre una organización y su proveedor se emplee encaminamiento estático. Lo mismo se aplica para la conexión directa entre dos organizaciones.

Por último señalar que además de los protocolos mencionados, cuya especificación se ha publicado como RFCs y por lo tanto son estándares de Internet, existen otros protocolos de tipo propietario como IGRP, EIGRP, HSRP y NHRP cuyo uso queda restringido en

aquellos entornos donde todos los equipos utilizados sean de un mismo fabricante o fabricantes que soporten compatibilidad con esos protocolos.

5. LA NUEVA VERSIÓN DEL PROTOCOLO IP: IPV6

En un futuro próximo, la actual versión del protocolo IP (la versión cuatro, IPv4) será sustituida por una nueva versión, la seis, con el denominado IPv6 o IPng (IP new generation). El principal motivo es la ampliación del campo de direcciones IP que pasará ahora de 32 a 128 bits. Con 2^{32} direcciones, es decir, aproximadamente 4000 millones, debería ser suficiente, pero la ineficaz distribución de direcciones en subredes hace que se desaprovechen la mayor parte de ellas.

Las mejoras del IPv6 incluyen los siguientes aspectos:

- Espacio de direcciones ampliado: 128 bits.
- Mecanismo de opciones mejorado: Las opciones van en cabeceras opcionales separadas a partir de la principal. Hay hasta ocho tipos diferentes que sólo se incluyen y/o procesan cuando son necesarias.
- Direcciones de autoconfiguración: Permiten la asignación dinámica de direcciones.
- Mayor flexibilidad en el direccionamiento.
- Facilidad de asignación de recursos al tráfico de alta prioridad: Desaparece el campo de TIPO DE SERVICIO y se incluye uno de PRIORIDAD que clasifica el tráfico en función de sus necesidades especiales como el vídeo en tiempo real.
- Capacidades de seguridad: Incluyen la autentificación y la privacidad de los datos.

La cabecera del IPv6 pasa a tener 40 en lugar de 20 bytes, sin embargo es más simple y el número de campos básicos que incluye es menor.

6. PROTOCOLO ICMP: MENSAJES DE ERROR Y CONTROL

Se ha visto que el protocolo IP proporciona un servicio no fiable y sin conexión; y que los mensajes viajan de "router" en "router" hasta alcanzar el nodo destino. El sistema funciona bien si todas las máquinas trabajan adecuadamente y los "routers" están de acuerdo en los encaminamientos. En caso contrario ocurren errores. Para permitir a las máquinas de una red IP informar sobre errores o circunstancias inesperadas está el protocolo ICMP (*Internet Control Message Protocol*), que es considerado como una parte del protocolo IP.

Los mensajes ICMP viajan en la porción de datos de los datagramas IP, como se muestra en la figura:

Encabezamiento IP	Mensaje ICMP tratado como datos

Los datagramas que llevan mensajes ICMP, se encaminan como los demás, por lo que pueden producirse errores. El protocolo dice que en este caso se produce una

excepción, y especifica que no se deben generar mensajes ICMP sobre errores resultantes de datagramas llevando mensajes ICMP. Los mensajes ICMP facilitan también el control de congestión.

Aunque cada tipo de mensaje ICMP tiene su propio formato, todos empiezan con los mismos tres campos: Un entero de 8 bits indicando TIPO. Un campo de 8 bits, (CODIGO) dando más información sobre el tipo de mensaje y un campo de 16 bits con el "checksum" (se usa el mismo algoritmo que para los datagramas IP, pero incluye sólo el mensaje ICMP). Además, los mensajes ICMP incluyen el encabezamiento del datagrama IP que causó el problema, así como los primeros 64 bits de datos, para ayudar a determinar qué protocolo y qué programa de aplicación causaron el problema. El campo TIPO define el tipo del mensaje ICMP y el formato del resto del paquete. Los tipos son:

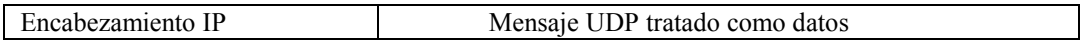
Campo TIPO	Tipo de mensaje ICMP
0	Respuesta de eco
3	Destino inalcanzable
4	Disminución de flujo de la fuente
5	Redireccionar (cambiar la ruta)
8	Petición de eco
11	Tiempo excedido por el datagrama
12	Problema de parámetro en un datagrama
13	Petición de grabar tiempos
14	Respuesta de grabar tiempos
15	Petición de información
16	Respuesta de información
17	Petición de máscara de direcciones
18	Respuesta de máscara de direcciones

7. EL PROTOCOLO UDP

Cuando un datagrama llega a su destino se ha de determinar a cual de las aplicaciones existentes en la máquina se ha de hacer llegar la información contenida en el. Esto no lo hace el protocolo IP directamente, sino que es misión del Protocolo de Transporte. El protocolo UDP proporciona la funcionalidad necesaria para identificar al destinatario final de un datagrama de una manera simple. Como no proporciona ningún mecanismo para el acuse de recibo, secuenciación de mensajes ni control de flujo, proporciona el mismo servicio no fiable que el protocolo IP. Los mensajes UDP pueden perderse o llegar fuera de secuencia, y además, los paquetes pueden llegar más rápido de lo que el receptor es capaz de procesar.

El protocolo UDP permite a varios programas de aplicación en una misma máquina comunicarse simultáneamente y demultiplexa el tráfico que se recibe entre las distintas aplicaciones. El protocolo UDP incorpora los denominados puertos, que identifican el último destino dentro de una máquina, el programa de aplicación que está haciendo uso de ese puerto. Cada puerto tiene asociado un entero para identificarlo. Dado que el número de puerto es único dentro de una máquina, el destino último queda perfectamente determinado por la dirección IP del "host" y el número de puerto UDP en ese "host".

Los mensajes UDP se denominan *datagramas de usuario* y viajan en la porción de datos de los datagramas IP, como se muestra en la figura:



El protocolo UDP entrega al servicio IP el segmento que contiene los datos, que es el que se transmite realmente, y una *pseudo-cabecera* que no se trasmite, pero que permite al protocolo IP completar los datos del o los datagramas que va a generar. El formato del segmento y de la pseudo-cabecera son los que se muestran en la figura siguiente.

0 16 31

PUERTO FUENTE	PUERTO DESTINO
LONGITUD	CHECKSUM
DATOS	
....	

DIRECCION IP FUENTE		
DIRECCION IP DESTINO		
CERO	PROTOCOLO	LONGITUD UDP

A continuación se describe el significado de los distintos campos del segmento.

- PUERTOS FUENTE Y DESTINO: Estos dos campos contienen los números de los puertos UDP que identifican los programas de aplicación en las máquinas fuente y destino
- LONGITUD: Es el número total de octetos que forman el datagrama UDP incluida la cabecera.
- CHECKSUM: Para calcular el "checksum", la máquina fuente antepone la pseudo-cabecera al datagrama y añade al final de los datos bytes conteniendo ceros hasta conseguir una longitud del segmento múltiplo de 16 bits. El "checksum" se calcula una vez hechos los cambios según el siguiente algoritmo: se considera el segmento formado por enteros de 16 bits y se suman todos los complementos a uno de esos enteros utilizando la aritmética de complemento a uno. A efectos de hacer los cálculos se supone que ese campo tiene valor cero. Los ceros añadidos para rellenar, así como la pseudo-cabecera no se cuentan en la longitud del datagrama y no son transmitidos.

  El “checksum” es opcional. Si no se utiliza, su contenido es cero.

  La razón de usar la pseudo-cabecera es permitir a la máquina destino comprobar que el datagrama ha alcanzado su destino correcto, ya que incluye la dirección IP del "host" destino, así como el número de puerto UDP de la conexión. La máquina destino puede conseguir la información usada en la pseudo-cabecera a partir del datagrama IP que transporta al datagrama UDP.
- DATOS: Representa los datos del datagrama UDP.

El significado de los campos de la pseudo-cabecera es el siguiente:

- DIRECCION IP FUENTE: Es la dirección IP del "host" que envía el segmento.
- DIRECCION IP DESTINO: Es la dirección IP del "host" al que va dirigido el segmento.
- CERO: Campo que contiene el valor cero.
- PROTOCOLO: Especifica protocolo UDP (es decir, 17).
- LONGITUD UDP: Este campo especifica la longitud total del datagrama UDP.

El protocolo UDP combina una adjudicación de números de puertos UDP dinámica y estática, usando una serie de asignaciones de puertos conocida para un conjunto de programas que se utilizan comúnmente (por ejemplo, correo electrónico). Sin embargo, la mayoría de los números de puerto están disponibles para que el sistema operativo los utilice a medida que los programas de aplicación los necesiten. (Ver apéndices)

8. EL PROTOCOLO TCP

A bajo nivel, el protocolo IP proporciona un servicio de distribución no fiable de paquetes. Cuando la red física falla, los paquetes pueden perderse, llegar con errores, duplicados o fuera de secuencia.

Al nivel más alto, los programas de aplicación necesitan, frecuentemente, enviar grandes volúmenes de datos. Utilizar el sistema de distribución no fiable anteriormente mencionado requiere que los programadores construyan algoritmos de detección y recuperación de errores en los programas de aplicación. Estos algoritmos son muy complejos, y por tanto pocos programadores tienen los conocimientos necesarios para diseñarlos. El protocolo UDP no mejora la falta de fiabilidad del protocolo IP, lo que obligaría al programador a implementar los algoritmos necesarios para la fiabilidad que precisa la aplicación. Como alternativa se puede utilizar, el protocolo de transporte TCP, que permite la transmisión fiable de datos mediante el establecimiento y liberación de conexiones, de manera que los datos que llegan a los programas de aplicación lo hagan sin errores y ordenados, pudiendo ser utilizados directamente, sin necesidad de escribir algoritmos de detección y recuperación de errores.

El protocolo TCP está encima del IP en el esquema de capas de la red Internet. TCP permite a varios programas de aplicación en una misma máquina comunicarse simultáneamente y demultiplexar el tráfico TCP que se recibe entre las distintas aplicaciones. El protocolo TCP incorpora los denominados puertos, que identifican el último destino dentro de una máquina, el programa de aplicación que está haciendo uso de ese puerto. Cada puerto tiene asociado un entero para identificarlo. Dado que el número de puerto es único dentro de una máquina, el destino último para el tráfico TCP queda perfectamente determinado por la dirección IP del "host" y el número de puerto TCP en ese "host".

El protocolo TCP combina una adjudicación de números de puertos TCP dinámica y estática, usando una serie de asignaciones de puertos conocida para un conjunto de programas que se utilizan comúnmente (por ejemplo, correo electrónico). Sin embargo, la mayoría de los números de puerto están disponibles para que el sistema operativo los utilice a medida que los programas de aplicación los necesiten. (Ver apéndices)

8.1 Formato del segmento TCP

El segmento TCP es la unidad de transferencia de datos de este protocolo. Los segmentos se intercambian para establecer y liberar conexiones, transferir datos, enviar acuses de recibo e informar sobre tamaños de ventana. Un acuse de recibo que vaya de una máquina A a otra B puede viajar en el mismo segmento que lleve datos de la máquina A a la máquina B (*piggybacking*).

Los segmentos TCP viajan en la porción de datos de los datagramas IP, como se muestra en la figura:

Encabezamiento IP	Segmento TCP tratado como datos

El protocolo TCP entrega al servicio IP el segmento que contiene los datos, que es el que se transmite realmente, y una *pseudo-cabecera* que no se trasmite, pero que permite al protocolo IP completar los datos del o los datagramas que va a generar. El formato del segmento y de la pseudo-cabecera son los que se muestran en la figura siguiente.

0 16 31

PUERTO FUENTE			PUERTO DESTINO
NUMERO DE SECUENCIA			
NUMERO DE ACUSE DE RECIBO			
DESP.	RES.	CODIGO	VENTANA
CHECKSUM			PUNTERO URGENTE
OPCIONES			RELLENO
DATOS			
....			

DIRECCION IP FUENTE		
DIRECCION IP DESTINO		
CERO	PROTOCOLO	LONGITUD TCP

A continuación se describe el significado de los distintos campos del segmento.

- PUERTOS FUENTE Y DESTINO: Estos dos campos contienen los números de los puertos TCP que identifican los programas de aplicación en las máquinas fuente y destino
- NÚMERO DE SECUENCIA: Este campo identifica la posición, en la secuencia de bytes de la máquina fuente, del primer byte de datos del segmento.
- NÚMERO DE ACUSE DE RECIBO: Identifica la posición del byte más alto que la máquina fuente ha recibido, referido al número de secuencia de byte de la máquina a la que va destinado el segmento.
- DESPLAZAMIENTO (DESP.): Este campo de 4 bits contiene un entero que especifica donde comienza el campo de datos del segmento. Es decir, indica la longitud de la cabecera en unidades de 32 bits. Se necesita porque la longitud del campo OPCIONES de la cabecera varía en longitud según las opciones elegidas.
- RESERVADO (RES.): Este campo (6 bits) está reservado para su uso en el futuro.
- CÓDIGO: Es un campo de 6 bits que determina el propósito y contenido del segmento. Los seis bits explican cómo hay que interpretar otros campos en el encabezamiento de acuerdo con la tabla que se muestra a continuación.

Bit (de izquierda a derecha)	Significado
URG	El campo puntero urgente es válido
ACK	El campo acuse de recibo es válido
PSH	El segmento requiere un "push"
RST	"Resetear" la conexión

SYN	Sincronizar números de secuencia
FIN	La fuente ha alcanzado el fin de su secuencia de bytes

- VENTANA: Este campo contiene el "informe de ventana", es decir, el número de bytes que la máquina está dispuesta a aceptar. Este campo se incluye tanto en los segmentos que llevan datos como en los que sólo llevan un acuse de recibo.
- CHECKSUM: Este campo contiene un entero de 16 bits usado para verificar la integridad del encabezamiento y los datos del segmento. Para calcular el "checksum", la máquina fuente antepone la pseudo-cabecera al segmento, y añade al final de los datos bytes conteniendo ceros hasta conseguir una longitud del segmento múltiplo de 16 bits. El "checksum" se calcula una vez hechos los cambios según el siguiente algoritmo: se considera el segmento formado por enteros de 16 bits y se suman todos los complementos a uno de esos enteros utilizando la aritmética de complemento a uno. A efectos de hacer los cálculos se supone que ese campo tiene valor cero. Los ceros añadidos para rellenar, así como la pseudo-cabecera no se cuentan en la longitud del segmento y no son transmitidos.

  La razón de usar la pseudo-encabecera es permitir a la máquina destino comprobar que el segmento ha alcanzado su destino correcto, ya que incluye la dirección IP del "host" destino, así como el número de puerto TCP de la conexión. La máquina destino puede conseguir la información usada en la pseudo-cabecera a partir del datagrama IP que lleva el segmento.
- PUNTERO URGENTE: Cuando el bit URG está a 1, el campo PUNTERO URGENTE especifica la posición en la secuencia de bytes en la que los datos urgentes acaban. Los datos urgentes deben ser distribuidos lo más rápidamente posible. El protocolo especifica que cuando se encuentran datos urgentes, el software TCP que los recibe debe informar al programa de aplicación asociado a la conexión para que se ponga en modo "urgente". Los detalles de cómo el software TCP debe informar al programa de aplicación dependen del sistema operativo que se esté usando.

  Los datos urgentes contienen mensajes en vez de datos normales; por ejemplo, señales de interrupción provocadas por pulsaciones en un teclado.
- OPCIONES: El software TCP usa este campo para comunicarse con el software TCP al otro lado de la conexión. En particular, una máquina puede comunicar a otra el máximo tamaño de segmento que está dispuesta a aceptar. Esto es muy importante cuando un computador pequeño va a recibir datos de uno grande. Escoger un tamaño adecuado de segmento es difícil, puesto que el rendimiento de la transmisión empeora para segmentos excesivamente grandes (pues hay que fragmentarlos en varios datagramas) o excesivamente pequeños (puesto que la proporción de bytes de encabezado es muy alta frente a la de datos).

- RELLENO: Representa octetos conteniendo ceros, que son necesarios para que el encabezamiento sea un múltiplo exacto de 32, ya se había visto que el campo de DESPLAZAMIENTO indica la longitud del encabezamiento en unidades de 32 bits.
- DATOS: Representa los datos del segmento.

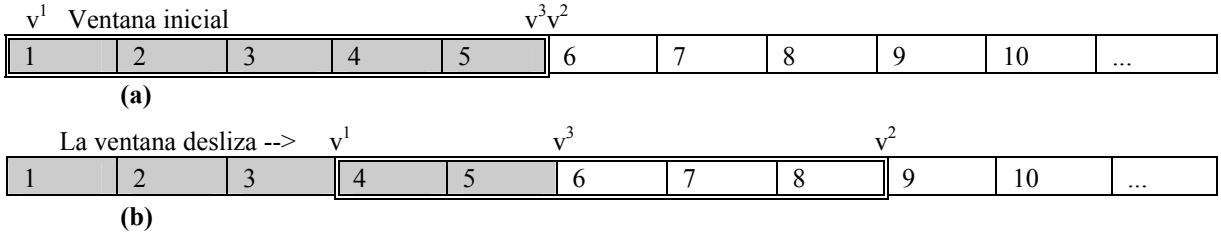
El significado de los campos de la pseudo-cabecera es el siguiente:

- DIRECCION IP FUENTE: Es la dirección IP del "host" que envía el segmento.
- DIRECCION IP DESTINO: Es la dirección IP del "host" al que va dirigido el segmento.
- CERO: Campo que contiene el valor cero.
- PROTOCOLO: Especifica protocolo TCP (es decir, 6).
- LONGITUD TCP: Este campo especifica la longitud total del segmento TCP.

8.2 La ventana deslizante del protocolo TCP

El protocolo TCP utiliza un mecanismo de ventana deslizante de tamaño variable para resolver dos problemas importantes: transmisión eficiente y control de flujo extremo a extremo, permitiendo a la máquina destino ordenar a la fuente restringir el envío de datos hasta que tenga suficiente espacio para tratar más datos. Sin embargo, el protocolo TCP ve a la corriente de datos como una secuencia de octetos o bytes que divide en *segmentos* para la transmisión. Generalmente, cada segmento viaja a través de la red Internet en un sólo datagrama IP.

Así, el mecanismo de ventana deslizante TCP opera a nivel de byte, no al de paquete. Todos los bytes en la secuencia de datos se numeran y el "host" que los envía mantiene tres punteros asociados a cada conexión que define una ventana deslizante. El primer puntero apunta al comienzo de la ventana, separando los bytes que han sido enviados y confirmados de los bytes aún no confirmados o no enviados. El segundo apunta al final de la ventana, definiendo el byte más alto en la secuencia que puede ser enviado sin que llegue un acuse de recibo. El tercer puntero, dentro de la ventana, separa los bytes que han sido enviados de los bytes que no lo han sido. En la siguiente figura se muestra un ejemplo.



En la figura (a) se muestra la técnica de la ventana deslizante con 5 bytes en la ventana ya enviados y no confirmados. En la figura (b), los bytes hasta el 3 han sido enviados y confirmados; los bytes 4 y 5 han sido enviados pero no confirmados; los bytes del 6 al 8 no han sido enviados, pero lo serán sin esperar a recibir acuse de recibo alguno, y los bytes 9 y más altos no se enviarán hasta que la ventana no se mueva.

Los bytes no confirmados son los que han sido enviados pero no han tenido acuse de recibo. El número de bytes no confirmados en un momento dado está limitado al tamaño de la ventana.

La fuente de los datos mantiene una ventana para los datos enviados y el destino mantiene un control de la ventana similar sobre los datos recibidos. Como la comunicación TCP es "full-duplex", se producen dos transferencias de datos simultáneamente en cada conexión, una en cada sentido. Las dos transferencias son independientes, por lo tanto, el software TCP mantiene dos ventanas para cada conexión en cada máquina: una desliza sobre la secuencia de bytes a enviar y la otra sobre los bytes que se van recibiendo.

8.3 Control de flujo

El método de ventana deslizante del protocolo TCP permite que el tamaño de la ventana varíe. Cada acuse de recibo, que especifica cuantos bytes han sido recibidos, contiene un "informe de ventana" que especifica cuantos bytes adicionales de datos está dispuesto a aceptar el "host" que envía el acuse de recibo. Si este informe indica un tamaño de la ventana mayor que el actual tamaño de la ventana, la máquina fuente lo incrementa y comienza a mandar más bytes. Si el informe de ventana especifica un tamaño menor, la fuente disminuye el tamaño de su ventana y no enviará datos más allá del límite de la misma. El software TCP no contradice informes de ventana previos y nunca reduce la ventana a posiciones anteriores a las aceptadas anteriormente.

La ventaja de tener un tamaño de ventana variable es que proporciona control de flujo así como transferencia fiable. Si la capacidad de aceptar datos de la máquina destino disminuye, envía un informe de ventana menor. En el caso extremo, el destino puede informar con un tamaño de ventana cero para detener la transmisión. Más tarde, cuando haya espacio disponible, el destino envía un informe de ventana mayor que cero para comenzar a recibir datos de nuevo.

Con el método de las ventanas deslizantes de tamaño variable, el protocolo TCP soluciona el problema de transferencia fiable y control de flujo extremo a extremo; sin embargo, no soluciona el control de congestión en la red Internet, pues ésta tiene conectadas máquinas intermedias de distintas velocidades y tamaños que se comunican a través de redes de distintas características. La misión de ese control corresponde lógicamente al Protocolo de Red IP, que para solucionarlo utiliza el mecanismo menos preciso de los mensajes ICMP de disminución de flujo de la fuente.

8.4 Acuses de recibo y retransmisiones

Puesto que el software TCP envía los datos en segmentos de longitud variable, los acuses de recibo se refieren a la posición en la secuencia de bytes y no a paquetes o segmentos. Cada acuse de recibo especifica la posición siguiente superior al último byte recibido.

Cada vez que envía un segmento, el software TCP inicia un temporizador y espera por un acuse de recibo. Si éste no llega antes de que el temporizador expire, se supone que el segmento se perdió y, en consecuencia, se retransmite. Esto lo hacen la mayoría de los protocolos. La diferencia del protocolo TCP es que está pensado para usarse en la red Internet, en la que el camino entre dos máquinas puede ser desde una simple red de alta velocidad, hasta un camino a través de muchas redes y nodos intermedios. Por tanto, es

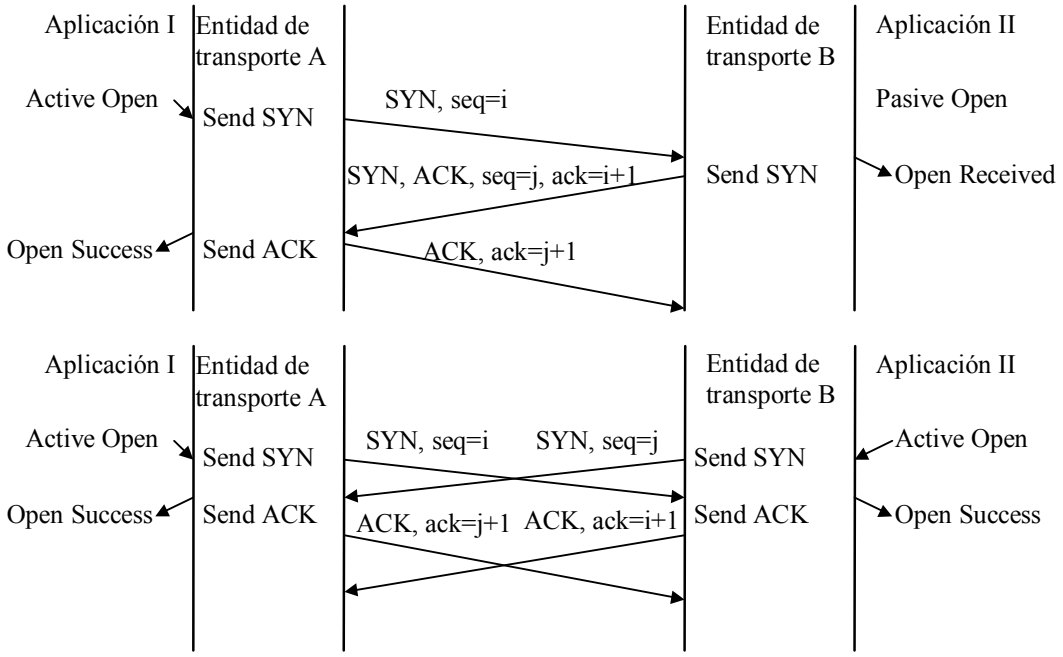
imposible conocer *a priori* lo rápido que van a llegar los acuses de recibo. Además, el retraso depende del tráfico, por lo que éste varía mucho de unos instantes a otros. El protocolo TCP se acomoda a los cambios en los retrasos de la red mediante un algoritmo adaptativo. Para conocer los datos necesarios para el algoritmo adaptativo, el software TCP graba la hora a la que envía el segmento y la hora a la que recibe el acuse de recibo para los datos del segmento. Con estos dos datos el computador calcula el llamado tiempo de retardo. Cada vez que el computador calcula un nuevo tiempo de retardo modifica su noción de tiempo medio de retardo para la conexión. Para calcular este tiempo medio se usa una media ponderada entre el último tiempo de retardo calculado y el tiempo medio anterior; y así el tiempo medio de retardo varía lentamente. Por ejemplo, una técnica de ponderación es usar un factor de peso, *q*, que varía entre cero y uno, para ponderar el tiempo medio antiguo frente al nuevo tiempo de retardo:

$$T_{\text{medio actual}}=(q*T_{\text{medio anterior}})+((1-q)*\text{Nuevo\_Tiempo\_Retardo})$$

Escogiendo un valor de *q* próximo a 1 hace que la media sea inmune a cambios que duren poco tiempo (por ejemplo, un segmento que sufre un retardo muy grande). Cogiendo un valor de *q* próximo a 0, el tiempo medio de retraso responde a los cambios muy rápidamente.

8.5 Establecimiento y liberación de una conexión TCP

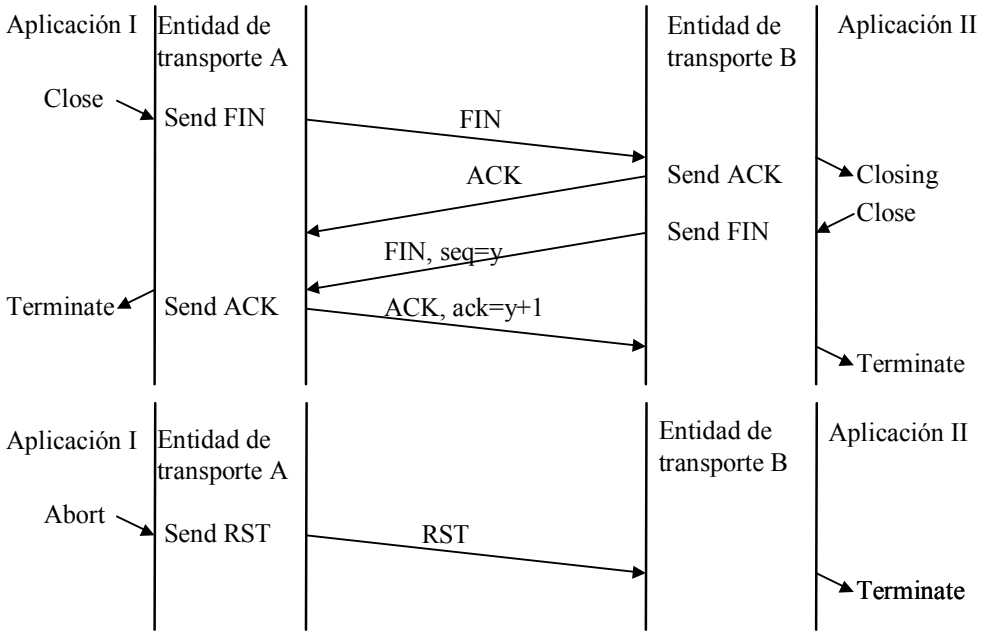
Para establecer una conexión, el protocolo TCP utiliza el *three-way handshake*. El primer segmento transmitido se puede identificar puesto que tiene el bit SYN a 1 en el campo CODIGO. El segundo segmento, respuesta al anterior, tiene los bits SYN y ACK a 1, indicando que reconoce el primer SYN y continúa el proceso de conexión. El último caso es simplemente un acuse de recibo para informar al destino de que ambas máquinas están de acuerdo en que la conexión ha sido establecida. Normalmente, el TCP en una máquina espera pasivamente por una conexión y la otra la inicia; sin embargo, el *three-way handshake* está diseñado para que la conexión se abra incluso si las dos partes la intentan iniciar simultáneamente. Una vez que la conexión está abierta, los datos pueden ir en ambas direcciones.



Al trabajar un protocolo de red no fiable y el software TCP puede tener que usar retransmisiones en las peticiones de conexión. El problema surge cuando se reciben peticiones retransmitidas de una conexión anterior cuando esta ya ha sido abierta, usada y terminada y una nueva está estableciéndose. El *three-way handshake*, junto con la regla de que TCP ignora las peticiones de conexión una vez que la conexión está establecida, soluciona el problema.

El *three-way handshake* sirve además para que los dos lados de la conexión se pongan de acuerdo en los números de secuencia iniciales, permitiendo así una correcta sincronización entre los dos lados de la conexión.. Estos números de secuencia son transmitidos y confirmados durante el establecimiento de la conexión. Los números de secuencia no tienen que empezar en 1 obligatoriamente. De hecho no deberían. Al empezar, una máquina (por ejemplo la A) pasa su número de secuencia inicial, *X*, en el primer paquete, que tiene el bit SYN a 1. La segunda máquina, B, recibe el SYN, graba el número de secuencia inicial de A y responde indicando su número de secuencia inicial, *Y*, así como un número de acuse de recibo que especifica que B está esperando el byte *X+1*. En el tercer paquete, la máquina A reconoce el número de secuencia inicial de B enviando como número acuse de recibo *Y+1*.

Para entender el proceso de libreración de una conexión hay que tener en cuenta que la comunicación es "full-dúplex", y que se transfieren dos secuencias de bytes independientes, una en cada dirección. Cuando un programa de aplicación indica al software TCP que no tiene más datos que transmitir, éste libera la conexión en esa dirección. Para liberar su mitad de conexión, el software TCP envía el último paquete de datos con el bit FIN a 1. En el otro extremo, el software TCP reconoce el FIN e indica al programa de aplicación que no se van a recibir más datos (por ejemplo, usando el mecanismo de final de fichero del sistema operativo). El bit FIN, al igual que se había visto con el SYN, ocupa una posición en la secuencia de bytes de la cabecera.



Una vez que la conexión se ha liberado en una dirección, no se aceptan más datos en esa dirección. Mientras tanto, los datos pueden seguir transmitiéndose en la dirección opuesta hasta que ésta también se libere. Cuando ambas direcciones han sido liberadas, la conexión se borra y los recursos que ésta utiliza (buffers, etc.) se liberan.

Un programa de aplicación libera una conexión cuando deja de usarla. Por tanto, liberar la conexión es algo normal. Sin embargo, a veces, aparecen condiciones anormales que obligan a una aplicación a abandonar la conexión. El protocolo TCP proporciona un mecanismo de "reset" para estas desconexiones anormales. Un lado de la conexión inicia el "reset" enviando un segmento con el bit RST a 1. El otro lado de la conexión responde abortando la misma. También informa al programa de aplicación que el "reset" ha ocurrido. El "reseteo" de una conexión significa que la transferencia en ambas direcciones cesa inmediatamente, y los recursos de la misma (buffers,etc.) se liberan.

8.6 Envío forzado de datos

El protocolo TCP permite dividir la secuencia de bytes a transmitir en segmentos. La ventaja es la eficiencia. Se pueden acumular suficientes bytes en un "buffer" para crear segmentos razonablemente largos, con lo que se reduce el alto porcentaje de encabezamientos de los segmentos cortos.

Aunque el uso de "buffers" incrementa la densidad de transmisión de la red, puede no ser conveniente para algunos tipos de aplicación; por ejemplo, en el caso de una conexión TCP utilizada para enviar caracteres desde un terminal interactivo a una máquina remota. El usuario espera una respuesta instantánea a cada pulsación de una tecla. Si el software TCP "bufferea" los datos, la respuesta se puede retrasar.

Para satisfacer a usuarios interactivos, el TCP proporciona una operación de *push* que un programa de aplicación puede utilizar para obligar al software TCP a transmitir los bytes de la secuencia sin que se llene el "buffer". Además, esta operación hace que el bit PSH del campo CODIGO esté a 1, con lo que los datos serán entregados al programa de aplicación en la máquina destino inmediatamente. Por lo tanto, cuando se envían datos desde una terminal interactiva, se usa la función de *push* después de cada pulsación de tecla.

Además de la función de *push*, el protocolo TCP proporciona una utilidad de *puntero urgente* que permite a la fuente informar al destino que los datos de la secuencia, hasta los que señala este puntero, son urgentes, y, por tanto, deben ser procesados lo más rápidamente posible. Por ejemplo, en una conexión TCP usada para intercambiar datos entre un terminal interactivo y un computador, los caracteres que paran y arrancan la salida por pantalla (por ejemplo control-s y control-q) deberían ser considerados urgentes, ya que el destino debe procesarlos inmediatamente.

9. EL SERVICIO DNS

El servicio de nombres de dominio, DNS (Domain Name Service), es un servicio de directorios basado en una base de datos distribuida. La función básica del servicio de nombres de dominio es la de facilitar el acceso a los recursos y servicios de Internet a través de nombres fácilmente recordables, en lugar de direcciones numéricas.

Las compañías telefónicas proporcionan a los usuarios extensísimas guías impresas o servicios de información por teléfono para que los usuarios puedan encontrar el número de teléfono de otro usuario del cual sólo conocen su nombre. De la misma manera, para conocer la dirección IP de un servidor del cual sólo se conoce su identificador (por ejemplo, www.uniovi.es), el sistema ha de tener una lista interna que relacione nombres y

direcciones IP o conocer un número al que consultar. Este último es el método más apropiado.

El sistema ha de conocer la dirección IP de uno o varios servidores DNS a los cuales consultar para encontrar la dirección IP que corresponde a un determinado nombre. El servidor DNS es un "host" que mantiene la base de datos para uno o varios dominios y que da servicio de nombres a los "hosts" de ese dominio. Si el servidor DNS no contiene la identificación del "host" que le han solicitado, consulta a su vez a otros servidores DNS de la red Internet hasta encontrarlo.

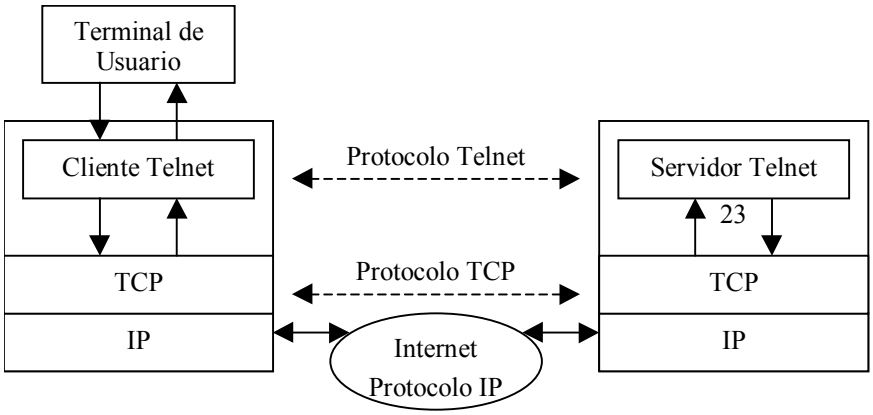
La organización jerarquizada y correcta de todos los servidores DNS en la red es fundamental para el correcto funcionamiento de la red Internet.

10. EL PROTOCOLO TELNET

TELNET permite a un usuario establecer una conexión TCP a un servidor de sesiones de usuario. Aunque TELNET no es tan sofisticado como otros protocolos de terminal remota, está disponible a lo largo de casi toda la red Internet.

TELNET ofrece tres servicios básicos. En primer lugar, define un terminal virtual que proporciona una interfaz con sistemas remotos. En segundo, incluye un mecanismo que permite al cliente y servidor negociar opciones y proporciona una serie de opciones estándar. Por último, TELNET trata a ambos lados de la conexión simétricamente. Así, en vez de forzar a un lado a conectarse a un terminal de usuario físico, permite a ambos lados de la conexión ser un programa.

Cuando un usuario invoca TELNET, un programa de aplicación en la máquina se convierte en cliente y contacta con un servidor en uno de los puertos TCP reservados, estableciendo una conexión sobre la que se comunicarán. El cliente acepta pulsaciones de tecla del terminal del usuario y las envía al servidor, a la vez que acepta caracteres que envía el servidor y los imprime en la pantalla del terminal.



11. TRANSFERENCIA DE FICHEROS

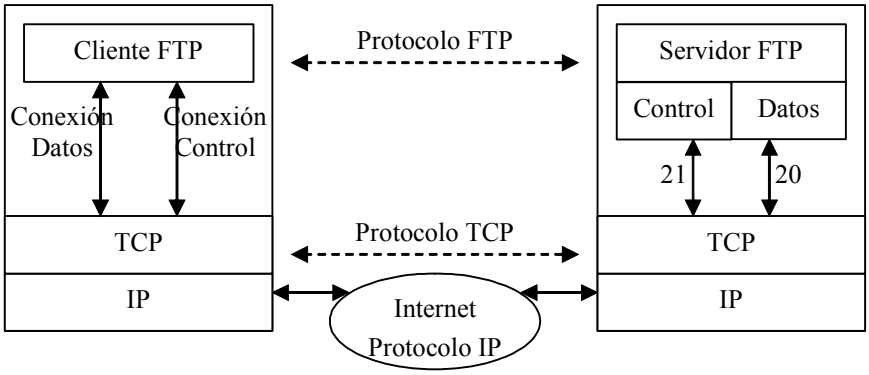
La transferencia de ficheros es una de las operaciones más usadas en la red. Principalmente se utilizan dos protocolos de transferencia: el FTP y el TFTP.

11.1 El protocolo FTP

El protocolo FTP (*File Transfer Protocol*) permite, a usuarios autorizados, entrar en un sistema remoto, identificarse y listar directorios remotos, copiar ficheros desde ó a la máquina remota y ejecutar algunos comandos remotos. Además, FTP maneja varios formatos de ficheros y puede hacer conversiones entre las representaciones más utilizadas (por ejemplo, entre EBCDIC y ASCII). FTP puede ser usado por usuarios interactivos así como por programas.

El protocolo FTP permite al usuario acceder a varias máquinas en una misma sesión. Mantiene dos conexiones TCP independientes para control y transferencia de datos. Usa el protocolo TELNET para el control de la conexión.

La implementación del protocolo FTP depende del sistema operativo usado, aunque casi todas siguen el mismo patrón. En el lado del servidor, un proceso de aplicación, S, corre esperando por una conexión en el puerto asignado para TCP. Cuando un cliente abre una conexión con ese puerto, el proceso S lanza un nuevo proceso de control, N, para manejar la conexión, y vuelve a esperar por otro cliente. El proceso N se comunica con el cliente por medio de la llamada "conexión de control" y cuando recibe una petición de transferencia inicia otro proceso adicional, D, que abre otra conexión con el cliente que solamente se usa para la transmisión de datos. Una vez que la transferencia de datos ha concluido, el proceso D cierra la conexión y termina. El cliente vuelve a su interactividad con el proceso N y puede solicitar otra transferencia de datos.



11.2 El protocolo TFTP

El protocolo TFTP (*Trivial File Transfer Protocol*) proporciona un servicio barato y poco sofisticado de transferencia de ficheros.

Al contrario que FTP, el protocolo TFTP no utiliza un servicio fiable de transmisión. No utiliza el protocolo TCP, sino que se basa en el protocolo UDP (*User Data Protocol*). TFTP utiliza temporización y retransmisión para asegurar que los datos llagan a su destino. La aplicación en la máquina fuente transmite un fichero en bloques de tamaño fijo (512 bytes) y espera por un acuse de recibo para cada bloque antes de enviar el siguiente. Por su parte, la aplicación en la máquina destino devuelve un acuse de recibo por cada bloque que le llega.

12. CORREO ELECTRÓNICO

Es una de las aplicaciones más comúnmente usadas en la red Internet ya que ofrece una forma sencilla de transmitir información. El correo electrónico es distinto a las otras aplicaciones de la red, pues no es necesario esperar a que la máquina remota reciba el mensaje para continuar trabajando. Cada vez que se quiere enviar un mensaje, el sistema crea una copia del mismo junto con la identificación del destino y se realiza una transferencia en modo "background". Posteriormente, el proceso de envío de paquetes tratará de entregar el mismo contactando con el servidor de mensajes en la máquina destino.

Las ventajas de usar el correo electrónico en la red Internet es que ésta proporciona un servicio universal y además fiable, ya que, al usar una comunicación extremo a extremo, se garantiza que el mensaje permanece en la máquina fuente hasta que ha sido copiado satisfactoriamente en la destino.

El estándar que se utiliza para el envío de mensajes es el SMTP (*Simple Mail Transfer Protocol*). Este protocolo se centra en cómo el sistema de distribución de mensajes pasa los datos a través de una unión de una máquina con la otra. Inicialmente, el cliente establece una conexión TCP con el servidor y se intercambian una serie de comandos para establecer la unión. Una vez la conexión está establecida, el cliente puede enviar uno o más mensajes, terminar la conexión o pedir al servidor intercambiar los papeles de emisor y receptor para que los mensajes puedan fluir en la dirección contraria. El receptor debe reconocer cada mensaje y puede abortar la conexión entera o la transferencia del mensaje actual.

El Protocolo de Oficina Postal (POP, actualmente POP3) define el diálogo entre un servidor de correo POP y la aplicación de correo electrónico en el computador del usuario. Esto es necesario cuando el usuario no lee el correo en la propia máquina donde se encuentra el buzón de su correo. Al recibir los mensajes el servidor de correo los almacena en buzones privados para cada usuario. POP permite que un Agente de Usuario (UA) acceda al buzón, descargue todos los mensajes pendientes y después los borra del servidor. De forma similar funciona IMAP (Internet Message Access Protocol), sólo que este protocolo permite al usuario mantener sus mensajes en el servidor donde están los buzones y clasificarlos en carpetas, sin necesidad de hacer copias en su computador local.

13. EL PROTOCOLO HTTP

El Protocolo de Transferencia de HyperTexto (HTTP) es la base de la existencia del World Wide Web (WWW). El servicio HTTP en un “host” se conoce como “servidor web” y permite que usuarios a distancia puedan acceder a los documentos que almacena si conocen su dirección exacta. El protocolo HTTP define un sistema de direcciones basado en Localizadores Uniformes de Recursos (URL). El URL de un recurso indica el protocolo o servicio (http, ftp, …) que se emplea para ser accedido, la dirección del host donde se encuentra el recurso (p.e. www.uniovi.es), y la ubicación del recurso dentro del host (p.e. Estudiante.html): http://www.uniovi.es/Estudiante.html.

La información hypertexto se almacena en formato HTML (o XTML, etc.) en documentos que se denominan "páginas". El navegador o "browser" es el programa de usuario que conecta con el servidor mediante el protocolo HTTP e interpreta la página en formato HTML antes de mostrarla al usuario.

14. BIBLIOGRAFÍA

Bibliografía consultada para la realización de este capítulo:

[STALLINGS 97]
Stallings, W. (1997).
Comunicaciones y redes de computadores, 5ª ed.
Prentice Hall Iberia.

[COMER 88]
Comer, D.E. (1988).
Internetworking With TCP/IP. Principles, Protocols, and Architecture.
Prentice-Hall.

[COMER 91]
Comer, D.E. (1991).
Internetworking With TCP/IP. Vol II: Design, Implementation and Internals.
Prentice-Hall.

Enlaces de interés:

www.nic.es	**Centro de Comunicaciones CSIC RedIRIS ES-NIC**
www.iana.org	**IANA:** ***Internet Assigned Numbers Authority***
www.icann.net	**ICANN:** ***Internet Corporation for Assigned Names and Numbers***
www.ripe.net	**RIPE NCC:** ***Réseaux IP Européens Network Coordination Centre***
www.etsimo.uniovi.es/ftp/pub/docs/rfc/	RFCs en la Universidad de Oviedo
www.freesoft.org/CIE/search.htm	Búsquedas en RFCs
www.ralphb.net/IPSubnet/cidr.html	Descripción de CIDR

15. APENDICES

Puertos estándar en TCP/IP

Port	Keyword	Description
0		Reserved
1	tcpmux	TCP Port Service Multiplexer
2	compressnet	Management Utility
3	compressnet	Compression Process
5	rje	Remote Job Entry
7	echo	Echo
9	discard	Discard
11	systat	Active Users
13	daytime	Daytime (RFC 867)
17	qotd	Quote of the Day
18	msp	Message Send Protocol
19	chargen	Character Generator
20	ftp-data	File Transfer [Default Data]
21	ftp	File Transfer [Control]
22	ssh	SSH Remote Login Protocol
23	telnet	Telnet
24		any private mail system
25	smtp	Simple Mail Transfer
29	msg-icp	MSG ICP
31	msg-auth	MSG Authentication
33	dsp	Display Support Protocol
35		any private printer server
37	time	Time
38	rap	Route Access Protocol
39	rlp	Resource Location Protocol
41	graphics	Graphics
42	name nameserver	Host Name Server
43	nicname	Who Is
44	mpm-flags	MPM FLAGS Protocol
45	mpm	Message Processing Module [recv]
46	mpm-snd	MPM [default send]
47	ni-ftp	NI FTP
49	tacacs	Login Host Protocol (TACACS)
50	re-mail-ck	Remote Mail Checking Protocol
51	la-maint	IMP Logical Address Maintenance
52	xns-time	XNS Time Protocol
53	domain	Domain Name Server
54	xns-ch	XNS Clearinghouse
55	isi-gl	ISI Graphics Language
56	xns-auth	XNS Authentication
57		any private terminal access
58	xns-mail	XNS Mail
59		any private file service
61	ni-mail	NI MAIL
63	whois++	whois++
64	covia	Communications Integrator (CI)
65	tacacs-ds	TACACS-Database Service
66	sql*net	Oracle SQL*NET
67	bootps	Bootstrap Protocol Server
68	bootpc	Bootstrap Protocol Client
69	tftp	Trivial File Transfer
70	gopher	Gopher
71	netrjs-1	Remote Job Service
72	netrjs-2	Remote Job Service
73	netrjs-3	Remote Job Service
74	netrjs-4	Remote Job Service
75		any private dial out service
76	deos	Distributed External Object Store
77		any private RJE service
78	vettcp	vettcp
79	finger	Finger
80	http www www-http	World Wide Web HTTP
81	hosts2-ns	HOSTS2 Name Server
82	xfer	XFER Utility
84	ctf	Common Trace Facility
87		any private terminal link
89	su-mit-tg	SU/MIT Telnet Gateway
91	mit-dov	MIT Dover Spooler
92	npp	Network Printing Protocol
93	dcp	Device Control Protocol
95	supdup	SUPDUP
98	tacnews	TAC News
101	hostname	NIC Host Name Server
102	iso-tsap	ISO-TSAP Class 0
105	cso csnet-ns	CCSO name server protocol Mailbox Name Nameserver
106	3com-tsmux	3COM-TSMUX
107	rtelnet	Remote Telnet Service
108	snagas	SNA Gateway Access Server
109	pop2	Post Office Protocol - Version 2
110	pop3	Post Office Protocol - Version 3
111	sunrpc	SUN Remote Procedure Call
113	ident auth	Authentication Service
114	audionews	Audio News Multicast
115	sftp	Simple File Transfer Protocol
117	uucp-path	UUCP Path Service
118	sqlserv	SQL Services
119	nntp	Network News Transfer Protocol
123	ntp	Network Time Protocol
129	pwdgen	Password Generator Protocol
130	cisco-fna	cisco FNATIVE
131	cisco-tna	cisco TNATIVE
132	cisco-sys	cisco SYSMAINT
133	statsrv	Statistics Service
136	profile	PROFILE Naming System
137	netbios-ns	NETBIOS Name Service
138	netbios-dgm	NETBIOS Datagram Service
139	netbios-ssn	NETBIOS Session Service
140	emfis-data	EMFIS Data Service
141	emfis-cntl	EMFIS Control Service
143	imap	Internet Message Access Protocol
144	uma	Universal Management Architecture
145	uaac	UAAC Protocol
146	iso-tp0	ISO-IP0
147	iso-ip	ISO-IP
150	sql-net	SQL-NET
152	bftp	Background File Transfer Program
153	sgmp	SGMP
156	sqlsrv	SQL Service
159	nss-routing	NSS-Routing
160	sgmp-traps	SGMP-TRAPS
161	snmp	SNMP
162	snmptrap	SNMPTRAP
163	cmip-man	CMIP/TCP Manager
164	cmip-agent	CMIP/TCP Agent
168	rsvd	RSVD
169	send	SEND
174	mailq	MAILQ
175	vmne	VMNET
177	xdmcp	X Display Manager Control Protocol
179	bgp	Border Gateway Protocol
187	aci	Application Communication Interface
189	qft	Queued File Transport
190	gacp	Gateway Access Control Protocol
191	prospero	Prospero Directory Service
192	osu-nms	OSU Network Monitoring System
193	srmp	Spider Remote Monitoring Protocol
194	irc	Internet Relay Chat Protocol
197	dls	Directory Location Service
198	dls-mon	Directory Location Service Monitor
199	smux	SMUX
200	src	IBM System Resource Controller
201	at-rtmp	AppleTalk Routing Maintenance
202	at-nbp	AppleTalk Name Binding
203	at-3	AppleTalk Unused

204	at-echo	AppleTalk Echo
205	at-5	AppleTalk Unused
206	at-zis	AppleTalk Zone Information
207	at-7	AppleTalk Unused
208	at-8	AppleTalk Unused
213	ipx	IPX
220	imap3	Interactive Mail Access Protocol v3
221	fln-spx	Berkeley rlogind with SPX auth
222	rsh-spx	Berkeley rshd with SPX auth
256	rap	RAP
260	openport	Openport
261	nsiiops	IIOP Name Service over TLS/SSL
264	bgmp	BGMP
280	http-mgmt	http-mgmt
349	mftp	mftp
353	ndsauth	NDSAUTH
385	ibm-app	IBM Application
389	ldap	Lightweight Directory Access Protocol
397	mptn	Multi Protocol Trans. Net.
399	iso-tsap-c2	ISO Transport Class 2 Non-Control over TCP
413	smsp	SMSP
414	infoseek	InfoSeek
420	smpte	SMPTE
433	nnsp	NNSP
443	https	http protocol over TLS/SSL
444	snpp	Simple Network Paging Protocol
444	snpp	Simple Network Paging Protocol (RFC1568)
445	microsoft-ds	Microsoft-DS
469	rcp	Radio Control Protocol
482	bgs-nsi	bgs-nsi
488	gss-http	gss-http
499	iso-ill	ISO ILL Protocol
500	isakmp	isakmp
508	xvttp	xvttp
512	exec	remote process execution;
513	login	remote login a la telnet;
514	shell	cmd
515	printer	spooler
516	videotex	videotex
518	ntalk	
519	utime	unixtime
520	efs	extended file name server
521	ripng	ripng
524	ncp	NCP
525	timed	timeserver
526	tempo	newdate
529	irc-serv	IRC-SERV
530	courier	rpc
531	conference	chat
532	netnews	readnews
533	netwall	for emergency broadcasts
537	nmsp	Networked Media Streaming Protocol
538	gdomap	gdomap
540	uucp	uucpd
541	uucp-rlogin	uucp-rlogin
543	klogin	
544	kshell	krcmd
546	dhcpv6-client	DHCPv6 Client
547	dhcpv6-server	DHCPv6 Server
550	new-rwho	new-who
554	rtsp	Real Time Stream Control Protocol
556	remotefs	rfs server
560	rmonitor	rmonitord
561	monitor	
562	chshell	chcmd
563	nntps	nntp protocol over TLS/SSL (was snntp)
565	whoami	whoami
566	streettalk	streettalk
567	banyan-rpc	banyan-rpc
573	banyan-vip	banyan-vip
580	sntp-heartbeat	SNTP HEARTBEAT
584	keyserver	Key Server
993	imap4-ssl 585	IMAP4+SSL (use instead)
586	password-chg	Password Change
587	submission	Submission
596	smsd	SMSD
600	ipcserver	Sun IPC server
608	sift-uft	Sender-Initiated/Unsolicited File Transfer
609	npmp-trap	npmp-trap
610	npmp-local	npmp-local
611	npmp-gui	npmp-gui
614	sshell	SSLshell
636	ldaps	ldap protocol over TLS/SSL (was sldap)
646	ldp	LDP
647	dhcp-failover	DHCP Failover
648	rrp	Registry Registrar Protocol (RRP)
652	udlr-dtcp	UDLR_DTCP
662	pftp	PFTP
666	doom	doom Id Software
675	dctp	DCTP
689	nmap	NMAP
691	msexch-routing	MS Exchange Routing
711	cisco-tdp	Cisco TDP
750	rfile	
750	loadav	
751	pump	
752	qrh	
753	rrh	
754	tell	send
758	nlogin	
759	con	
760	ns	
761	rxe	
762	quotad	
763	cycleserv	
764	omserv	
765	webster	
767	phonebook	phone
769	vid	
770	cadlock	
771	rtip	
772	cycleserv2	
773	submit	
774	rpasswd	
775	entomb	
776	wpages	
777	multiling-http	Multiling HTTP
780	wpgs	
800	mdbs_daemon	
801	device	
810	fcp-udp	FCP
873	rsync	rsync
989	ftps-data	ftp protocol, data, over TLS/SSL
990	ftps	ftp protocol, control, over TLS/SSL
992	telnets	telnet protocol over TLS/SSL
993	imaps	imap4 protocol over TLS/SSL
994	ircs	irc protocol over TLS/SSL
995	pop3s	pop3 protocol over TLS/SSL (was spop3)
996	vsinet	vsinet
997	maitrd	
998	busboy	
999	garcon	
999	puprouter	
1010	surf	surf
1011-1023		Reserved

Conexión de Supra con Autom: F:> telnet autom
==========

Máquinas que intervienen:

Nombre lógico Dirección IP Dirección Ethernet
supra.etsiig.uniovi.es 156.35.101.11 48:4C:00:01:25:E3
autom.etsiig.uniovi.es 156.35.101.6 AA:00:04:00:35:F2
trono.etsiig.uniovi.es 156.35.41.20 --:--:--:--:--:-- servidor de nombres
rvie00.etsiig.uniovi.es 156.35.101.201 AA:00:04:00:75:1A router

Cada bloque de líneas representa una trama Ethernet enviada, el contenido del datagrama IP que porta y el segmento TCP o datagrama UDP que va encapsulado en este. Las Líneas alineadas a la izquierda son enviadas por Supra y las sangradas hacia la derecha las recibidas.

#Consulta al servidor de nombres 156.35.41.20 (trono) para conocer la dir IP de Autom.

Ether 00:00:00:00:00:00->AA:00:04:00:75:1A
IP 156.35.101.11 ->156.35.41.20 len 68 prot 17
UDP 1055->53 40
DNS Questions autom.etsiig.uniovi.es 0001 0001; size = 40

Ether AA:00:04:00:75:1A->48:4C:00:01:25:E3
IP 156.35.41.20 ->156.35.101.11 len 84 prot 17
UDP 53->1055 56
DNS Answers autom.etsiig.uniovi.es 156.35.101.6; size = 56

state = syn_sent

#Busqueda de la dirección Ethernet del destino con ARP

Ether 48:4C:00:01:25:E3->FF:FF:FF:FF:FF:FF
ARP 1 0001 0800 48:4C:00:01:25:E3:156.35.101.11 ->00:00:00:00:00:00:156.35.101.6

Ether AA:00:04:00:35:F2->48:4C:00:01:25:E3
ARP 2 0001 0800 AA:00:04:00:35:F2:156.35.101.6 ->48:4C:00:01:25:E3:156.35.101.11

#Envio de petición de conexión hacia el destino (autom:156.35.101.6)

Ether 00:00:00:00:00:00->AA:00:04:00:35:F2
IP 156.35.101.11 ->156.35.101.6 len 44 prot 6
TCP 1034->23 seq 00000000 SYN wind 4096 opt 020405B4

Ether AA:00:04:00:35:F2->48:4C:00:01:25:E3
IP 156.35.101.6 ->156.35.101.11 len 44 prot 6
TCP 23->1034 seq 69165800 ack 00000001 SYN ACK wind 3000 opt 020405A8

state = established

Ether 00:00:00:00:00:00->AA:00:04:00:35:F2
IP 156.35.101.11 ->156.35.101.6 len 40 prot 6
TCP 1034->23 seq 00000001 ack 00000001 ACK wind 4096

#Conexión establecida, comienza el intercambio de datos

Ether 00:00:00:00:00:00->AA:00:04:00:35:F2
IP 156.35.101.11 ->156.35.101.6 len 52 prot 6
TCP 1034->23 seq 00000001 ack 00000001 PSH ACK wind 4096 data 12

Ether AA:00:04:00:35:F2->48:4C:00:01:25:E3
IP 156.35.101.6 ->156.35.101.11 len 43 prot 6
TCP 23->1034 seq 00000001 ack 00000001 PSH ACK wind 3000 data 3

.....

Ether AA:00:04:00:35:F2->48:4C:00:01:25:E3
IP 156.35.101.6 ->156.35.101.11 len 48 prot 6
TCP 23->1034 seq 00000099 ack 0000004F PSH ACK wind 3000 data 8

Ether AA:00:04:00:35:F2->48:4C:00:01:25:E3
IP 156.35.101.6 ->156.35.101.11 len 69 prot 6
TCP 23->1034 seq 000000A1 ack 0000004F PSH ACK wind 3000 data 29

 # Finalizado el intercambio de datos. Petición de desconexión

Ether AA:00:04:00:35:F2->48:4C:00:01:25:E3
IP 156.35.101.6 ->156.35.101.11 len 65 prot 6
TCP 23->1034 seq 000000BE ack 0000004F FIN PSH ACK wind 3000 data 25

Ether 00:00:00:00:00:00->AA:00:04:00:35:F2
IP 156.35.101.11 ->156.35.101.6 len 40 prot 6
TCP 1034->23 seq 0000004F ack 000000D8 ACK wind 4034

state = close_wait

Ether 00:00:00:00:00:00->AA:00:04:00:35:F2
IP 156.35.101.11 ->156.35.101.6 len 40 prot 6
TCP 1034->23 seq 0000004F ack 000000D8 FIN ACK wind 4042

state = last_ack

Ether AA:00:04:00:35:F2->48:4C:00:01:25:E3
IP 156.35.101.6 ->156.35.101.11 len 40 prot 6
TCP 23->1034 seq 000000D8 ack 00000050 ACK wind 2999

state = closed

Conexión de Supra con zeus.etsimo.uniovi.es: F:> telnet zeus.etsimo
==========

supra.etsiig.uniovi.es 156.35.101.11 48:4C:00:01:25:E3
zeus.etsimo.uniovi.es 156.35.23.24 --:--:--:--:--:--
trono.etsiig.uniovi.es 156.35.41.20 --:--:--:--:--:-- servidor de nombres
rvie00.etsiig.uniovi.es 156.35.101.201 AA:00:04:00:75:1A router

No hay consulta ARP por que la dirección física del router ya esta en la tabla ARP

Ether 00:00:00:00:00:00->AA:00:04:00:75:1A
IP 156.35.101.11 ->156.35.41.20 len 67 prot 17
UDP 1058->53 39
DNS Questions zeus.etsimo.uniovi.es 0001 0001; size = 39

Ether AA:00:04:00:75:1A->48:4C:00:01:25:E3
IP 156.35.41.20 ->156.35.101.11 len 83 prot 17
UDP 53->1058 55
DNS Answers zeus.etsimo.uniovi.es 156.35.23.24; size = 55

state = syn_sent

Ether 00:00:00:00:00:00->AA:00:04:00:75:1A
IP 156.35.101.11 ->156.35.23.24 len 44 prot 6
TCP 1040->23 seq 00000000 SYN wind 4096 opt 020405B4

Ether AA:00:04:00:75:1A->48:4C:00:01:25:E3
IP 156.35.23.24 ->156.35.101.11 len 44 prot 6
TCP 23->1040 seq 160D8200 ack 00000001 SYN ACK wind 61320 opt 020405B4

state = established

Ether 00:00:00:00:00:00->AA:00:04:00:75:1A
IP 156.35.101.11 ->156.35.23.24 len 40 prot 6
TCP 1040->23 seq 00000001 ack 00000001 ACK wind 4096

 # Conexión establecida. Comienza el intercambio de datos

Ether 00:00:00:00:00:00->AA:00:04:00:75:1A
IP 156.35.101.11 ->156.35.23.24 len 52 prot 6
TCP 1040->23 seq 00000001 ack 00000001 PSH ACK wind 4096 data 12

Ether AA:00:04:00:75:1A->48:4C:00:01:25:E3
IP 156.35.23.24 ->156.35.101.11 len 40 prot 6
TCP 23->1040 seq 00000001 ack 0000000D ACK wind 61308

Ether AA:00:04:00:75:1A->48:4C:00:01:25:E3
IP 156.35.23.24 ->156.35.101.11 len 52 prot 6
TCP 23->1040 seq 00000001 ack 0000000D PSH ACK wind 61308 data 12

Ether 00:00:00:00:00:00->AA:00:04:00:75:1A
IP 156.35.101.11 ->156.35.23.24 len 40 prot 6
TCP 1040->23 seq 0000000D ack 0000000D ACK wind 4084

.....

Ether 00:00:00:00:00:00->AA:00:04:00:75:1A
IP 156.35.101.11 ->156.35.23.24 len 41 prot 6
TCP 1040->23 seq 0000003F ack 000001B4 PSH ACK wind 4096 data 1

Ether AA:00:04:00:75:1A->48:4C:00:01:25:E3
IP 156.35.23.24 ->156.35.101.11 len 44 prot 6
TCP 23->1040 seq 000001B4 ack 00000040 PSH ACK wind 61320 data 4

 # Fin de intercambio de datos. Desconexión

Ether AA:00:04:00:75:1A->48:4C:00:01:25:E3
IP 156.35.23.24 ->156.35.101.11 len 40 prot 6
TCP 23->1040 seq 000001B8 ack 00000040 FIN ACK wind 61320

Ether 00:00:00:00:00:00->AA:00:04:00:75:1A
IP 156.35.101.11 ->156.35.23.24 len 40 prot 6
TCP 1040->23 seq 00000040 ack 000001B9 ACK wind 4092

state = close_wait

Ether 00:00:00:00:00:00->AA:00:04:00:75:1A
IP 156.35.101.11 ->156.35.23.24 len 40 prot 6
TCP 1040->23 seq 00000040 ack 000001B9 FIN ACK wind 4096

state = last_ack

Ether AA:00:04:00:75:1A->48:4C:00:01:25:E3
IP 156.35.23.24 ->156.35.101.11 len 40 prot 6
TCP 23->1040 seq 000001B9 ack 00000041 ACK wind 61319

state = closed

Conexión telnet de Supra a Circe (telnet circe)

supra.etsiig.uniovi.es 156.35.101.11 puerto 1026 (cliente TELNET, telnet)
circe.etsiig.uniovi.es 156.35.101.5 puerto 23 (servidor TELNET, telnetd)

Comandos ejecutados:

Conexión	F:> telnet circe:	username:	
		password:	
Listado de directorio	CIRCE> ls -al	....	
Cierre de conexión	CIRCE> exit	F:>	

Unidades de datos TCP intercambiadas:

Petición de conexión:

state = syn_sent
1026->23 seq 00000000 SYN wind 4096 opt 020405B4
23->1026 seq 00C7B800 ack 00000001 SYN ACK **wind 0** opt 020405B1
state = established
1026->23 seq 00000001 ack 00000001 ACK wind 4096

Conexión establecida, comienza el intercambio de datos:

23->1026 seq 00000001 ack 00000001 ACK **wind 4096**
1026->23 seq 00000001 ack 00000001 PSH ACK wind 4096 data 12
23->1026 seq 00000001 ack 0000000D ACK wind 4084
23->1026 seq 00000001 ack 0000000D PSH ACK wind 4084 data 36
23->1026 seq 00000025 ack 0000000D PSH ACK wind 4087 data 3
23->1026 seq 00000028 ack 0000000D PSH ACK wind 4090 data 3
23->1026 seq 0000002B ack 0000000D PSH ACK wind 4093 data 3
23->1026 seq 0000002E ack 0000000D PSH ACK wind 4096 data 3
1026->23 seq 0000000D ack 00000031 ACK wind 4048
1026->23 seq 0000000D ack 00000031 PSH ACK wind 4096 data 3
23->1026 seq 00000031 ack 00000010 ACK wind 4096
23->1026 seq 00000031 ack 00000010 PSH ACK wind 4096 data 7
1026->23 seq 00000010 ack 00000038 PSH ACK wind 4089 data 3
1026->23 seq 00000013 ack 00000038 ACK wind 4089
23->1026 seq 00000038 ack 00000013 ACK wind 4096
1026->23 seq 00000013 ack 00000038 PSH ACK wind 4096 data 1
23->1026 seq 00000038 ack 00000014 PSH ACK wind 4096 data 1
1026->23 seq 00000014 ack 00000039 ACK wind 4095
1026->23 seq 00000014 ack 00000039 PSH ACK wind 4096 data 1
23->1026 seq 00000039 ack 00000015 PSH ACK wind 4096 data 1
1026->23 seq 00000015 ack 0000003A ACK wind 4095
1026->23 seq 00000015 ack 0000003A PSH ACK wind 4096 data 1
23->1026 seq 0000003A ack 00000016 PSH ACK wind 4096 data 1
1026->23 seq 00000016 ack 0000003B ACK wind 4095
1026->23 seq 00000016 ack 0000003B PSH ACK wind 4096 data 1
23->1026 seq 0000003B ack 00000017 PSH ACK wind 4096 data 1
1026->23 seq 00000017 ack 0000003C ACK wind 4095
1026->23 seq 00000017 ack 0000003C PSH ACK wind 4096 data 1
23->1026 seq 0000003C ack 00000018 PSH ACK wind 4096 data 1
1026->23 seq 00000018 ack 0000003D ACK wind 4095
1026->23 seq 00000018 ack 0000003D PSH ACK wind 4096 data 1
23->1026 seq 0000003D ack 00000019 PSH ACK wind 4096 data 2
1026->23 seq 00000019 ack 0000003F ACK wind 4094
23->1026 seq 0000003F ack 00000019 PSH ACK wind 4096 data 9
1026->23 seq 00000019 ack 00000048 ACK wind 4087
1026->23 seq 00000019 ack 00000048 PSH ACK wind 4096 data 1
23->1026 seq 00000048 ack 0000001A ACK wind 4096
1026->23 seq 0000001A ack 00000048 PSH ACK wind 4096 data 1
23->1026 seq 00000048 ack 0000001B ACK wind 4096
1026->23 seq 0000001B ack 00000048 PSH ACK wind 4096 data 1
23->1026 seq 00000048 ack 0000001C ACK wind 4096
1026->23 seq 0000001C ack 00000048 PSH ACK wind 4096 data 1
23->1026 seq 00000048 ack 0000001D ACK wind 4096
1026->23 seq 0000001D ack 00000048 PSH ACK wind 4096 data 1
23->1026 seq 00000048 ack 0000001E ACK wind 4096
1026->23 seq 0000001E ack 00000048 PSH ACK wind 4096 data 1
23->1026 seq 00000048 ack 0000001F ACK wind 4096
1026->23 seq 0000001F ack 00000048 PSH ACK wind 4096 data 1
23->1026 seq 00000048 ack 00000020 ACK wind 4096
1026->23 seq 00000020 ack 00000048 PSH ACK wind 4096 data 1
23->1026 seq 00000048 ack 00000021 ACK wind 4096
1026->23 seq 00000021 ack 00000048 PSH ACK wind 4096 data 1
23->1026 seq 00000048 ack 00000022 PSH ACK wind 4096 data 2
1026->23 seq 00000022 ack 0000004A ACK wind 4094
23->1026 seq 0000004A ack 00000022 PSH ACK wind 4096 data 2
1026->23 seq 00000022 ack 0000004C ACK wind 4094
23->1026 seq 0000004C ack 00000022 PSH ACK wind 4096 data 100
23->1026 seq 000000B0 ack 00000022 PSH ACK wind 4096 data 7
1026->23 seq 00000022 ack 000000B7 ACK wind 3989
23->1026 seq **000000B7** ack 00000022 PSH ACK wind 4096 data 73
23->1026 seq 00000100 ack 00000022 PSH ACK wind 4096 data 2
23->1026 seq 00000102 ack 00000022 PSH ACK wind 4096 data 48
23->1026 seq 00000132 ack 00000022 PSH ACK wind 4096 data 2
23->1026 seq 00000134 ack 00000022 PSH ACK wind 4096 data 61
23->1026 seq 00000171 ack 00000022 PSH ACK wind 4096 data 60
23->1026 seq 000001AD ack 00000022 PSH ACK wind 4096 data 63
23->1026 seq 000001EC ack 00000022 PSH ACK wind 4096 data 67
23->1026 seq 0000022F ack 00000022 PSH ACK wind 4096 data 65
23->1026 seq 00000270 ack 00000022 PSH ACK wind 4096 data 32
23->1026 seq 00000290 ack 00000022 PSH ACK wind 4096 data 2
23->1026 seq 00000292 ack 00000022 PSH ACK wind 4096 data 29
23->1026 seq 000002AF ack 00000022 PSH ACK wind 4096 data 2
23->1026 seq 000002B1 ack 00000022 PSH ACK wind 4096 data 56
23->1026 seq 000002E9 ack 00000022 PSH ACK wind 4096 data 64
23->1026 seq 00000329 ack 00000022 PSH ACK wind 4096 data 45
23->1026 seq 00000356 ack 00000022 PSH ACK wind 4096 data 49
23->1026 seq 000000B7 ack 00000022 PSH ACK wind 4096 data 927
1026->23 seq 00000022 ack 00000456 ACK wind 3169
1026->23 seq 00000022 ack 00000456 PSH ACK wind 4096 data 1
23->1026 seq **00000456** ack 00000023 PSH ACK wind 4096 data 9
23->1026 seq 0000045F ack 00000023 PSH ACK wind 4096 data 73

23->1026 seq 000004A8 ack 00000023 PSH ACK wind 4096 data 2
23->1026 seq 000004AA ack 00000023 PSH ACK wind 4096 data 2
23->1026 seq 000004AC ack 00000023 PSH ACK wind 4096 data 81
23->1026 seq 000004FD ack 00000023 PSH ACK wind 4096 data 75
23->1026 seq 00000548 ack 00000023 PSH ACK wind 4096 data 75
23->1026 seq 00000593 ack 00000023 PSH ACK wind 4096 data 79
23->1026 seq 000005E2 ack 00000023 PSH ACK wind 4096 data 75
23->1026 seq 0000062D ack 00000023 PSH ACK wind 4096 data 75
23->1026 seq 00000678 ack 00000023 PSH ACK wind 4096 data 81
23->1026 seq 000006C9 ack 00000023 PSH ACK wind 4096 data 2
23->1026 seq 000006CB ack 00000023 PSH ACK wind 4096 data 80
23->1026 seq 0000071B ack 00000023 PSH ACK wind 4096 data 57
23->1026 seq 00000754 ack 00000023 PSH ACK wind 4096 data 2
23->1026 seq 00000756 ack 00000023 PSH ACK wind 4096 data 78
23->1026 seq 000008A5 ack 00000023 PSH ACK wind 4096 data 64
23->1026 seq 000008E5 ack 00000023 PSH ACK wind 4096 data 41
23->1026 seq 0000090E ack 00000023 PSH ACK wind 4096 data 40
23->1026 seq 00000936 ack 00000023 PSH ACK wind 4096 data 59
23->1026 seq 00000971 ack 00000023 PSH ACK wind 4096 data 2
23->1026 seq 00000973 ack 00000023 PSH ACK wind 4096 data 70
23->1026 seq 000009B9 ack 00000023 PSH ACK wind 4096 data 2
23->1026 seq 000009BB ack 00000023 PSH ACK wind 4096 data 2
23->1026 seq 00000456 ack 00000023 PSH ACK wind 4096 data 1383
1026->23 seq 00000023 ack 000009BD ACK wind 2713
23->1026 seq 000009BD ack 00000023 PSH ACK wind 4096 data 100
23->1026 seq 00000A21 ack 00000023 PSH ACK wind 4096 data 7
23->1026 seq 00000A28 ack 00000023 PSH ACK wind 4096 data 22
23->1026 seq 00000A3E ack 00000023 PSH ACK wind 4096 data 6
1026->23 seq 00000023 ack 00000A44 ACK wind 3961
1026->23 seq 00000023 ack 00000A44 PSH ACK wind 4096 data 1
23->1026 seq 00000A44 ack 00000024 PSH ACK wind 4096 data 1
1026->23 seq 00000024 ack 00000A45 ACK wind 4095
1026->23 seq 00000024 ack 00000A45 PSH ACK wind 4096 data 1
23->1026 seq 00000A45 ack 00000025 PSH ACK wind 4096 data 1
1026->23 seq 00000025 ack 00000A46 ACK wind 4095
1026->23 seq 00000025 ack 00000A46 PSH ACK wind 4096 data 1
23->1026 seq 00000A46 ack 00000026 PSH ACK wind 4096 data 1
1026->23 seq 00000026 ack 00000A47 ACK wind 4095
1026->23 seq 00000026 ack 00000A47 PSH ACK wind 4096 data 1
23->1026 seq 00000A47 ack 00000027 PSH ACK wind 4096 data 2
23->1026 seq 00000A49 ack 00000027 PSH ACK wind 4096 data 2
1026->23 seq 00000027 ack 00000A4B ACK wind 4092
23->1026 seq **00000A4B** ack 00000027 PSH ACK wind 4096 data 12
23->1026 seq 00000A57 ack 00000027 PSH ACK wind 4096 data 62
23->1026 seq 00000A95 ack 00000027 PSH ACK wind 4096 data 69
23->1026 seq 00000ADA ack 00000027 PSH ACK wind 4096 data 63
23->1026 seq 00000B19 ack 00000027 PSH ACK wind 4096 data 81
23->1026 seq 00000B6A ack 00000027 PSH ACK wind 4096 data 57
23->1026 seq 00000BA3 ack 00000027 PSH ACK wind 4096 data 68
23->1026 seq 00000BE7 ack 00000027 PSH ACK wind 4096 data 67
23->1026 seq 00000C2A ack 00000027 PSH ACK wind 4096 data 61
23->1026 seq 00000C67 ack 00000027 PSH ACK wind 4096 data 62
23->1026 seq 00000CA5 ack 00000027 PSH ACK wind 4096 data 61
23->1026 seq 00000CE2 ack 00000027 PSH ACK wind 4096 data 68
23->1026 seq 00000D26 ack 00000027 PSH ACK wind 4096 data 65
23->1026 seq 00000D67 ack 00000027 PSH ACK wind 4096 data 61
23->1026 seq 00000DA4 ack 00000027 PSH ACK wind 4096 data 63
23->1026 seq 00000DE3 ack 00000027 PSH ACK wind 4096 data 61
23->1026 seq 00000E20 ack 00000027 PSH ACK wind 4096 data 59
23->1026 seq 00000EDF ack 00000027 PSH ACK wind 4096 data 6
23->1026 seq 00000A4B ack 00000027 PSH ACK wind 4096 data 1178
1026->23 seq 00000027 ack 00000EE5 ACK wind 2918
1026->23 seq 00000027 ack 00000EE5 PSH ACK wind 4096 data 1
23->1026 seq 00000EE5 ack 00000028 PSH ACK wind 4096 data 1
1026->23 seq 00000028 ack 00000EE6 ACK wind 4095
1026->23 seq 00000028 ack 00000EE6 PSH ACK wind 4096 data 1
23->1026 seq 00000EE6 ack 00000029 PSH ACK wind 4096 data 1
1026->23 seq 00000029 ack 00000EE7 ACK wind 4095
1026->23 seq 00000029 ack 00000EE7 PSH ACK wind 4096 data 1
23->1026 seq 00000EE7 ack 0000002A PSH ACK wind 4096 data 1
1026->23 seq 0000002A ack 00000EE8 ACK wind 4095
1026->23 seq 0000002A ack 00000EE8 PSH ACK wind 4096 data 1
23->1026 seq 00000EE8 ack 0000002B PSH ACK wind 4096 data 1
1026->23 seq 0000002B ack 00000EE9 ACK wind 4095
1026->23 seq 0000002B ack 00000EE9 PSH ACK wind 4096 data 1
23->1026 seq 00000EE9 ack 0000002C PSH ACK wind 4096 data 2
23->1026 seq 00000EEB ack 0000002C PSH ACK wind 4096 data 2
23->1026 seq 00000EED ack 0000002C PSH ACK wind 4096 data 8
1026->23 seq 0000002C ack 00000EF5 ACK wind 4084

Circe solicita la desconexión al haber ejecutado el usuario el comando "exit"
23->1026 seq 00000EF5 ack 0000002C FIN ACK wind 4096
1026->23 seq 0000002C ack 00000EF6 ACK wind 4096
state = close_wait

Conexión cerrada en el sentido circe(23)->supra(1026). Falta cerrar el otro sentido de la comunicación ya que es full-duplex.
1026->23 seq 0000002C ack 00000EF6 FIN ACK wind 4096
state = last_ack
23->1026 seq 00000EF6 ack 0000002D ACK wind 4095
state = closed

Conexión cerrada definitivamente en ambos sentidos.

Conexion con ftp.uniovi.es y peticion del directorio raiz: F:> ftp ftp.uniovi.es

==========

supra.etsiig.uniovi.es 156.35.101.11 puerto 1043 (cliente FTP control)
puerto 42561 (cliente FTP datos)
ftp.uniovi.es 156.35.31.31 puerto 21 (servidor FTP control)
puerto 20 (servidor FTP datos)

state = syn_sent
13576.8 1043->21 seq 00000000 SYN wind 4096 opt 020405B4
13577.0 21->1043 seq 7E117001 ack 00000001 SYN ACK wind 4096 opt 020405B4
state = established
13577.0 1043->21 seq 00000001 ack 00000001 ACK wind 4096
13577.3 21->1043 seq 00000001 ack 00000001 PSH ACK wind 4096 data 81
13577.3 1043->21 seq 00000001 ack 00000052 ACK wind 4015
13577.4 1043->21 seq 00000001 ack 00000052 PSH ACK wind 4096 data 14
13577.7 21->1043 seq 00000052 ack 0000000F ACK wind 4096
13577.7 1043->21 seq 0000000F ack 00000052 PSH ACK wind 4096 data 2
13577.8 21->1043 seq 00000052 ack 00000011 PSH ACK wind 4096 data 68
13577.8 1043->21 seq 00000011 ack 00000096 ACK wind 4028
13591.1 1043->21 seq 00000011 ack 00000096 PSH ACK wind 4096 data 27
13591.2 21->1043 seq 00000096 ack 0000002C ACK wind 4096
13591.2 1043->21 seq 0000002C ack 00000096 PSH ACK wind 4096 data 2
13591.4 21->1043 seq 00000096 ack 0000002E ACK wind 4096
13591.5 21->1043 seq 00000096 ack 0000002E PSH ACK wind 4096 data 6
13591.5 1043->21 seq 0000002E ack 0000009C ACK wind 4090
13592.1 21->1043 seq 0000009C ack 0000002E PSH ACK wind 4096 data 1207
13592.1 1043->21 seq 0000002E ack 00000553 ACK wind 2889
13603.7 1043->21 seq 0000002E ack 00000553 PSH ACK wind 4096 data 25
13603.8 21->1043 seq 00000553 ack 00000047 ACK wind 4096
13603.8 1043->21 seq 00000047 ack 00000553 PSH ACK wind 4096 data 2
13603.9 21->1043 seq 00000553 ack 00000049 PSH ACK wind 4096 data 30
13604.0 1043->21 seq 00000049 ack 00000571 ACK wind 4066
state = listen
13604.4 1043->21 seq 00000049 ack 00000571 PSH ACK wind 4096 data 4
13604.4 21->1043 seq 00000571 ack 0000004D ACK wind 4096
13604.5 1043->21 seq 0000004D ack 00000571 PSH ACK wind 4096 data 2
13604.8 21->1043 seq 00000571 ack 0000004F ACK wind 4096
13604.8 20->42561 seq 7E491A01 SYN wind 4096 opt 020405B4
13604.9 42561->20 seq 00000000 ack 00000001 SYN ACK wind 4096 opt 020405B4
state = syn_received
13605.0 20->42561 seq 00000001 ack 00000001 ACK wind 4096
13605.1 21->1043 seq 00000571 ack 0000004F PSH ACK wind 4096 data 53
13605.2 20->42561 seq 00000001 ack 00000001 PSH ACK wind 4096 data 618
13605.3 20->42561 seq 0000026B ack 00000001 FIN ACK wind 4096
13605.4 1043->21 seq 0000004F ack 000005A6 ACK wind 4043
state = established
13605.5 42561->20 seq 00000001 ack 0000026C ACK wind 3478
state = close_wait
13605.7 21->1043 seq 000005A6 ack 0000004F PSH ACK wind 4096 data 24
13605.7 1043->21 seq 0000004F ack 000005BE ACK wind 4019
13607.4 42561->20 seq 00000001 ack 0000026C FIN ACK wind 4096
state = last_ack
13607.7 20->42561 seq 0000026C ack 00000002 ACK wind 4096
state = closed