Prácticas de laboratorio de Telemática II

Práctica 3

Departamento de Ingeniería Telemática

(ENTEL)

Mónica Aguilar
Juanjo Alins
Oscar Esparza
Jose L. Muñóz
Marcos Postigo
Antoni X. Valverde

La composición de este manual de prácticas ha sido realizada con el programa LaTeX 2ε. Agradecer a DONALD KNUTH la idea y el programa TeX y a LESLIE LAMPORT sus magníficas macros que han derivado en LaTeX.

Barcelona a 5 de Febrero de 2003

• Revision 1.8 ip.lyx

Edición electrónica de este documento. Este manual de prácticas se puede conseguir en http://labtel2.upc.es

Índice de las prácticas

Índice de figuras

PRÁCTICA 3

Internet Protocol (IP)

3.1. Introducción

Los protocolos de red se diseñan en forma de capas o niveles, donde cada nivel es responsable de una faceta diferente de las comunicaciones. Una torre de protocolos es la combinación de diversos protocolos de varios niveles. TCP/IP se considera un sistema de 4 niveles (o 4 capas) tal y como se muestra en la figura 3.1.

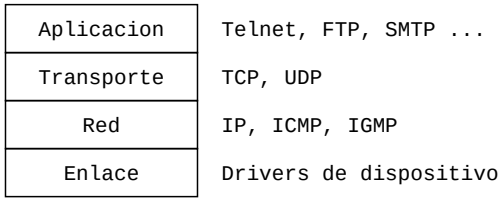


Figura 3.1: Las 4 capas de TCP/IP

TCP e IP son los dos protocolos que dan nombre a una arquitectura de red. Sin embargo, una red TCP/IP necesita otros protocolos para tener todas las funcionalidades. La figura 3.2 muestra los protocolos que forman la familia TCP/IP.

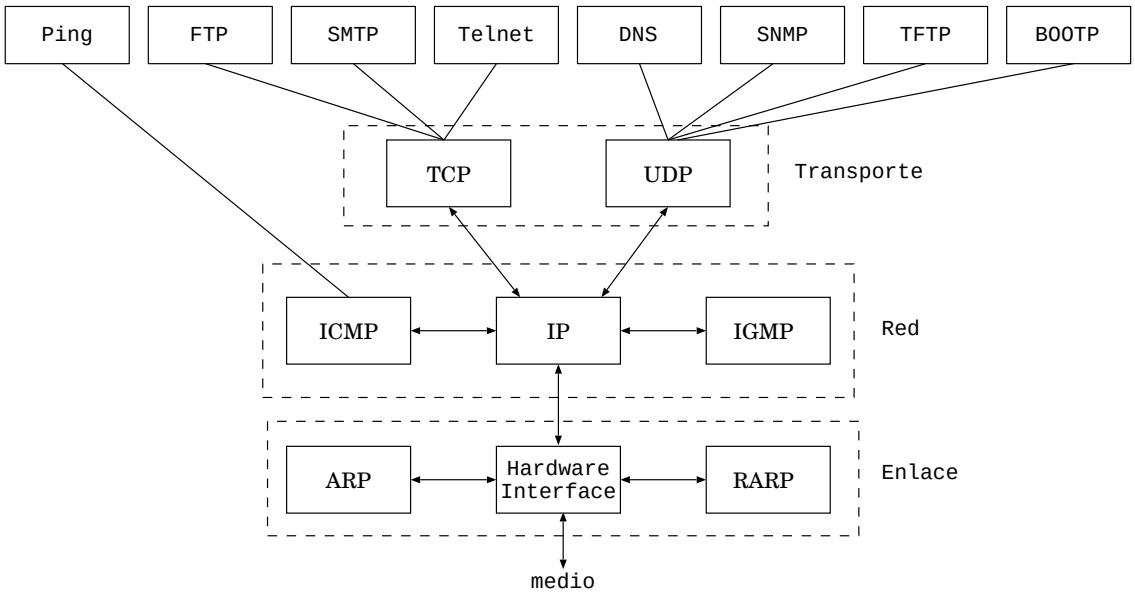


Figura 3.2: Protocolos de la familia TCP/IP

El objetivo del protocolo IP es convertir redes físicamente diferentes (como pueden ser *Ethernet*, *Token Ring*, X.25, *Frame Relay*, ATM...) en una red aparentemente homogénea, lo que se conoce como interconexión de redes. De esta forma IP oculta las redes físicas subyacentes creando por encima de éstas una única red "virtual" o red de redes, donde:

- Hay un esquema de identificación (o direccionamiento) de todos los sistemas, uniforme y universal. Este esquema de direccionamiento ha de ser independiente del *hardware*. Esto se consigue asignando a cada nodo un número único de 32 bits (en IPv4, *Internet Protocol* versión 4) llamado dirección IP.
- Los datos viajan por la red virtual IP agrupados en paquetes denominados datagramas. La manipulación de estos datagramas por parte de la red sigue un método uniforme llamado "encaminamiento de datagramas", independiente de la red en particular donde se encuentren estos datagramas.
- La entrega de datagramas de un extremo a otro por la red IP es:
  - No fiable (no hay control de errores, ni control de flujo).
  - No orientado a conexión.
  - La red se esfuerza al máximo para entregar los paquetes, pero no asegura la entrega (*best effort*).

Por tanto los datagramas se pueden perder, desordenar o incluso duplicar, e IP no tendrá en cuenta estas situaciones (lo controlarán protocolos superiores). El protocolo IP asume pocas cosas de las capas inferiores, sólo que los datagramas "probablemente" serán transportados al *host* de destino.

3.2. Las direcciones de Internet

Las direcciones de Internet pueden ser simbólicas o numéricas:

- Las direcciones simbólicas son las que usualmente utilizamos los humanos porque son más fáciles de recordar que las numéricas. Estas direcciones son cadenas de caracteres separadas por puntos. Por ejemplo: **www.upc.es**.
- Las direcciones numéricas son las que usualmente utiliza el *software* IP. La forma numérica es una secuencia de 32 bits. Para que sea más inteligible, habitualmente se expresa en forma de números decimales separados por puntos (*dotted-decimal* en inglés), donde cada decimal representa 8 bits (rango 0-255). Por ejemplo, la dirección binaria **10010011. 01010011. 00010100. 00000010** se puede expresar en notación decimal como **147.83.20.2** (que a su vez se corresponde con la dirección simbólica **www.upc.es**).

Las funciones de mapeo entre las direcciones simbólicas y las direcciones numéricas las realiza el DNS (*Domain Name System*), tal como se vio en anteriores prácticas.

Una dirección no identifica una máquina en Internet, sino a un determinado interfaz de red de la máquina. Dicho de otra forma, la dirección IP identifica a una determinada máquina en una determinada red física. Cuando la máquina está conectada a más de una red se la denomina "*multi-homed*" y tiene una dirección por cada interfaz de red.

En realidad, para interpretar correctamente una determinada dirección IP se necesita una segunda secuencia de 32 bits denominada "máscara" (*netmask*). La máscara se utiliza para dividir la dirección en dos partes:

- La parte de red, denominada formalmente "número de red" (RFC 1166), aunque en ocasiones se usan los términos dirección de red y netID. Los números de red están administrados centralmente por el INTERNIC (*INTErnet Network Information Center*) y son únicos en toda Internet. Cuando se solicita al INTERNIC una dirección IP, no se asigna una dirección a cada máquina individual, sino que se da un número de red, permitiendo asignar según las necesidades todas las direcciones IP válidas dentro de este rango.
- La parte de *host*, denominada formalmente "número de *host*", aunque ocasionalmente se usan los términos dirección de *host* y hostID.

Para dividir una dirección IP en número de red y número de *host* se utiliza la máscara, de manera que aquellos bits de la dirección donde la máscara tiene un "1" forman parte del número de red, mientras que las posiciones donde la máscara tiene "0" forman parte del número de *host*. Por ejemplo si tenemos la dirección 10010011. 01010011. 00010100.

00000010 con una máscara 11111111. 11111111. 00000000. 00000000 tenemos un número de red 10010011. 01010011. y un número de *host* 00010100. 00000010.

Por convenio el número de red se encuentra en los bits más significativos (los bits de la izquierda de la dirección) mientras que el número de *host* se suele encontrar en los bits menos significativos (los bits de la derecha de la dirección). Además los números de red y *host* suelen estar formados por bits consecutivos de la dirección IP (es muy extraño ver máscaras con ceros y unos no consecutivos).

La máscara de red también se suele representar en formato decimal, de manera que para el ejemplo anterior tenemos la dirección 147.83.20.2 y la máscara 255.255.0.0. Otra manera de representar una dirección IP es poniendo la dirección en formato decimal y la máscara como un número decimal que indica cuántos unos consecutivos forman el número de red. Así para nuestro ejemplo, la dirección 147.83.20.2 con máscara 255.255.0.0 se puede escribir como 147.83.20.2/16.

El tamaño de la parte dedicada al *host* depende del tamaño de la red. Para satisfacer múltiples necesidades se han definido varias clases de redes, fijando diferentes puntos donde dividir la dirección IP. De esta manera se dispone las siguientes clases:

Clase A

La clase A comprende redes desde 1.0.0.0 hasta 127.0.0.0. El identificador de red está contenido en el primer octeto, quedando para el *host* los 24 bits restantes y permitiendo aproximadamente 1.6 millones de máquinas por red. Por lo tanto, la máscara de red resultante es 255.0.0.0 (11111111. 00000000. 00000000. 00000000 en formato binario). Dentro de esta clase, el rango 10.0.0.0 hasta 10.255.255.255 es un rango reservado para uso privado. Una dirección de *host* reservada posible sería 10.0.0.1/8.

Clase B

La clase B comprende las redes desde 128.0.0.0 hasta 191.255.0.0, donde el *netid* está en los dos primeros octetos. Esta clase permite 16320 redes con 65024 lugares cada una. La máscara de esta clase es 255.255.0.0 (11111111. 11111111. 00000000. 00000000 en formato binario). En esta clase el rango 172.16.0.0 hasta 172.31.0.0 se reserva para uso privado. Una dirección de *host* posible reservada sería 172.31.0.1/16.

Clase C

Las redes de clase C van desde 192.0.0.0 hasta 223.255.255.0, donde el *netid* está en los tres primeros octetos. Esta clase permite cerca de 2 millones de redes con 254 lugares. La máscara de esta clase es de la forma 255.255.255.0 (11111111. 11111111. 11111111. 00000000 en formato binario). En esta clase se reserva para uso privado el rango 192.168.0.0 hasta 192.168.255.0. Una posible dirección de *host* privada puede ser 192.168.128.1/24.

Clase D

Son direcciones dentro del rango 224.0.0.0 hasta 239.255.255.255 y se las llama direcciones *multicast* o de multidifusión.

Clase E

Las direcciones que están en el rango 240.0.0.0 hasta 247.255.255.255 son experimentales o están reservadas para futuras aplicaciones.

Direcciones Especiales

Cabe destacar que en la parte reservada a *host*, los valores 0 y 255 se reservan porque tienen un significado especial:

- Una dirección IP donde todos los bits de la parte de *host* son cero se refiere a la dirección de la red misma.
- Una dirección donde todos los bits de la parte de *host* son uno se denomina dirección de difusión (o *broadcast*), ya que hace referencia a todas las máquinas de la red específica. Así 147.83.20.255 no es un *hostid* válido, pero se refiere a todos los *hostid* de la red 147.83.20.0.

Las direcciones de red 0.0.0.0 y 127.0.0.0 están reservadas. La primera dirección se llama encaminamiento por defecto (por donde IP encamina los datagramas), y la segunda es la dirección de *loopback*.

La red 127.0.0.0 está reservada para el tráfico local IP de la máquina. Normalmente la dirección 127.0.0.1 se asigna a una interfaz de la máquina (la interfaz de *loopback*) que actúa como un circuito cerrado. Cualquier paquete IP enviado a este interfaz será devuelto como si hubiera vuelto desde alguna red. Esto permite por ejemplo instalar *software* de red o probar tarjetas *ethernet*, aunque no se disponga de una red real.

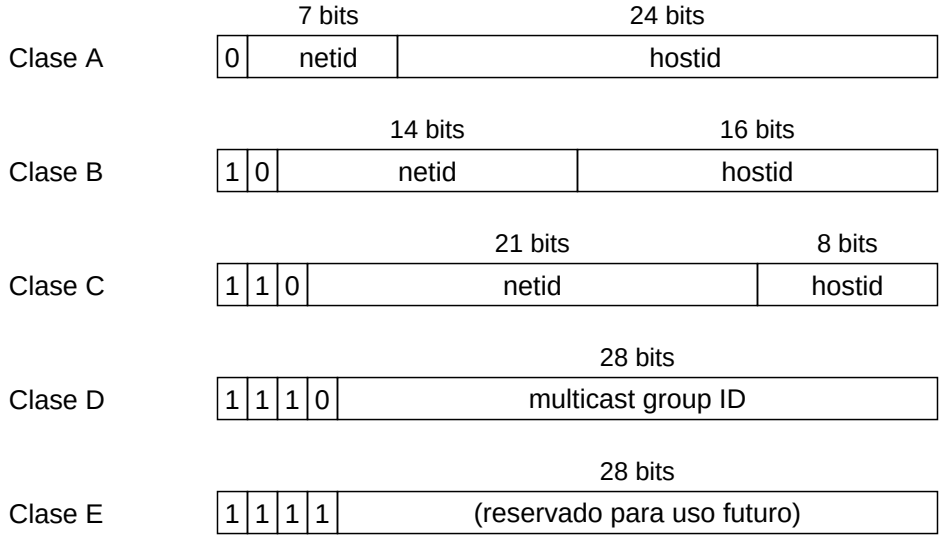


Figura 3.3: Clases de direcciones IP

3.3. El datagrama IP

El datagrama IP es la unidad de transferencia en las redes IP. Básicamente consiste en una cabecera IP y un campo de datos para protocolos superiores. El datagrama IP está encapsulado en la trama de nivel de enlace, que suele tener una longitud máxima (MTU, *Maximum Transfer Unit*), dependiendo del *hardware* de red usado. Para Ethernet, esta es típicamente de 1500 bytes. En vez de limitar el datagrama a un tamaño máximo, IP puede tratar la fragmentación y el reensamblado de sus datagramas. En particular, IP no impone un tamaño máximo, pero establece que todas las redes deberían ser capaces de manejar al menos 576 bytes. Los fragmentos de datagramas tienen todos una cabecera, copiada básicamente del datagrama original, y de los datos que la siguen. Los fragmentos se tratan como datagramas

normales mientras son transportados a su destino. Nótese, sin embargo, que si uno de los fragmentos se pierde, todo el datagrama se considerará perdido, y los restantes fragmentos también se considerarán perdidos.

3.3.1. Formato del datagrama IP

La cabecera del datagrama IP está formada por los campos que se muestran en la figura 3.4.

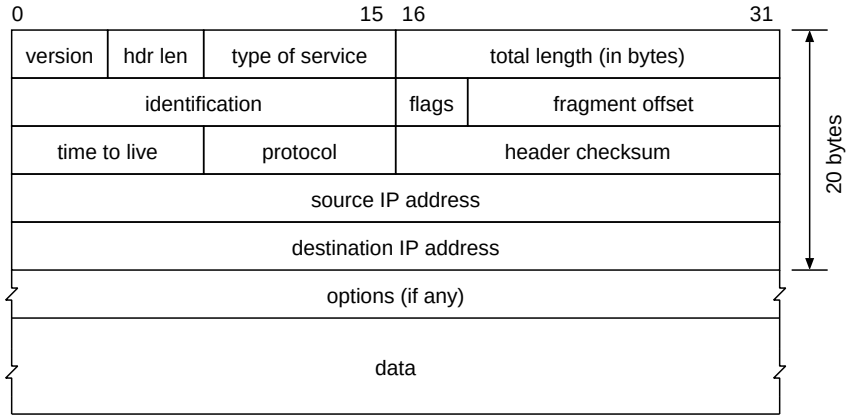


Figura 3.4: Formato del datagrama IP

Donde:

Version es la versión del protocolo IP. La versión actual es la 4. La 5 es experimental y la 6 es IPng.

Hdr Len es la longitud de la cabecera IP contada en cantidades de 32 bits. Esto no incluye el campo de datos.

Type Of Service es el tipo de servicio es una indicación de la calidad del servicio solicitado para este datagrama IP. Una descripción detallada de este campo se puede encontrar en el RFC 1349.

Total Length es la longitud total del datagrama, cabecera y datos, especificada en bytes.

Identification es un número único que asigna el emisor para ayudar a reensamblar un datagrama fragmentado. Los fragmentos de un datagrama tendrán el mismo número de identificación.

Flags son flags para el control de fragmentación.

Donde:

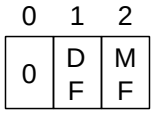


Figura 3.5: Detalle del campo *Flags*

El bit 0 está reservado y debe ser 0.
El bit **DF** significa no fragmentar (***D**o not **F**ragment*). Con 0 se permite fragmentación y con 1 no.
El bit **MF** significa que hay más fragmentos (***M**ore **F**ragments*). Con 0 significa que se trata del último fragmento del datagrama y con 1 que hay más fragmentos.

Fragment Offset (FO) se usa en datagramas fragmentados para ayudar al reensamblado de todo el datagrama. El valor es el número de partes de 64 bits (no se cuentan los bytes de la cabecera) contenidas en fragmentos anteriores. En el primer (o único) fragmento el valor es siempre cero.

Time To Live especifica el tiempo (en segundos) que se le permite viajar a este datagrama. Cada "*router*" por el que pase este datagrama ha de sustraer de este campo el tiempo tardado en procesarlo. En la realidad un "*router*" es capaz de procesar un datagrama en menos de 1 segundo; por ello restará uno de este campo y el TTL se convierte más en una cuenta de saltos que en una métrica del tiempo. Cuando el valor alcanza cero, se asume que este datagrama ha estado viajando en un bucle y se desecha. El valor inicial lo debería fijar el protocolo de alto nivel que crea el datagrama.

Protocol indica el número oficial del protocolo de alto nivel al que IP debería entregar los datos del datagrama. Algunos valores importantes se muestran en la Tabla 3.1.

Protocolo	Número
Reservado	0
ICMP	1
IGMP	2
IP encapsulado	4
TCP	6
UDP	17
OSPF	89

Tabla 3.1: Algunos *Protocol Numbers*

Header Checksum es el *checksum* de la cabecera. Se calcula como el complemento a uno de la suma de los complementos a uno de todas las palabras de 16 bits de la cabecera. Si el *checksum* de la cabecera no se corresponde con los contenidos, el datagrama se desecha, ya que al menos un bit de la cabecera está corrupto, y el datagrama podría haber llegado a un destino equivocado.

Source IP Address es la dirección IP de 32 bits del *host* emisor.

Destination IP Address es la dirección IP de 32 bits del *host* receptor.

Options es un campo de longitud variable. Las opciones se incluyen en principio para pruebas de red o depuración, por tanto no se requiere que toda implementación de IP sea capaz de generar opciones en los datagramas que crea, pero sí que sea capaz de procesar datagramas que contengan opciones.
El campo **Options** tiene longitud variable en función de la opción seleccionada. Algunas opciones tienen una longitud de un solo byte y otras tienen longitudes variables. El primer byte de cualquier opción se denomina *type byte* y su estructura se muestra en la figura 3.6.

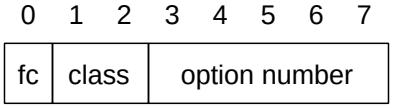


Figura 3.6: Detalle del *type byte*

Donde:

FC es el flag de copia (***F**lag **C**opy*), e indica si el campo de opciones se ha de copiar (1) o no (0) cuando el datagrama está fragmentado.

class es un entero sin signo de 2 bits, donde:

number es un entero sin signo de 5 bits que indica el número de opción dentro de cada clase.

Option class	significado
0	Control de red o datagrama
1	Reservado para uso futuro
2	Depuración y medición
3	Reservado para uso futuro

Tabla 3.2: Option classes

3.3.2. La fragmentación de datagramas IP

Cuando un datagrama IP viaja de un *host* a otro puede cruzar distintas redes físicas. Las redes físicas imponen un tamaño máximo de trama, llamado MTU (*Maximum Transmission Unit*), que limita la longitud de un datagrama.

Por ello, existe un mecanismo para fragmentar los datagramas IP grandes en otros más pequeños, y luego reensamblarlos en el *host* de destino. IP requiere que cada enlace tenga un MTU de al menos 68 bytes, de forma que si cualquier red proporciona un valor inferior, la fragmentación y el reensamblado tendrán que implementarse en la capa de la interfaz de red de forma transparente a IP. 68 es la suma de la mayor cabecera IP, de 60 bytes, y del tamaño mínimo posible de los datos en un fragmento (8 bytes).

Las implementaciones de IP no están obligadas a manejar datagramas sin fragmentar mayores de 576 bytes, pero la mayoría podrá manipular valores más grandes, típicamente ligeramente por encima de 8192 bytes o incluso mayores, y raramente menos de 1500.

Un datagrama sin fragmentar tiene a cero toda la información de fragmentación. Es decir, los flags **FC** y **FO** están a 0.

Cuando se ha de realizar la fragmentación, se ejecutan los siguientes pasos:

- Se chequea el bit de flag **DF** para ver si se permite fragmentación. Si está a uno, el datagrama se desecha y se devuelve un error al emisor usando ICMP [1](*Internet Control Message Protocol*).
- Basándose en el valor MTU, el campo de datos se divide en partes donde cada parte, excepto la última, debe ser múltiplo de 8 bytes.
- Todas las porciones de datos se colocan en datagramas IP.
- Se copian las cabeceras de la cabecera original, con algunas modificaciones:
  - El bit de flag **MF** se pone a uno en todos los fragmentos, excepto en el último.
  - El campo **FO** se pone al valor de la localización de la porción de datos correspondiente.
  - Si se incluyeron opciones en el datagrama original, el bit **FC** del type byte determina si se copiaran o no en todos los fragmentos o sólo en el primero[2].
  - Se inicializa el campo **Hdr Len** (longitud de la cabecera).
  - Se inicializa el campo **Total Length** (longitud total).
  - Se recalcula el **Header Checksum** de la cabecera.

Cada uno de estos datagramas se envía como un datagrama IP normal. IP maneja cada fragmento de forma independiente, es decir, los fragmentos pueden atravesar diversas rutas hacia su destino, y pueden estar sujetos a nuevas fragmentaciones si pasan por redes con MTUs inferiores.

En el *host* de destino, los datos se tienen que reensamblar. Para ello se siguen los siguientes pasos:

- Con el fin de reensamblar los fragmentos, el receptor destina un buffer de almacenamiento en cuanto llega el primer fragmento.

[1] Véase el apartado 3.7, dedicado a este protocolo.
[2] Por ejemplo, las opciones de encaminamiento de la fuente se tendrán que copiar en todos los fragmentos y por tanto tendrán a uno este bit.

- Una vez ha llegado el primer fragmento se inicia un contador temporal[3].
- Cada fragmento se identifica mediante el campo **Identification** que es un número único dentro de los límites impuestos por el uso de un número de 16 bits. Como la fragmentación no altera este campo, los fragmentos que van llegando al destino pueden ser identificados gracias a este identificador y a las direcciones IP fuente y destino del datagrama. Además el campo **Protocol** también se chequea.
- Los fragmentos que van llegando se copian en el *buffer* en la localización indicada por el campo **FO**.
- Cuando han llegado todos los fragmentos, se restaura el datagrama original y se continúa con su proceso.
- Si vence el contador temporal y no se han recibido todos los fragmentos, el datagrama en fase de reensamblado se desecha.

3.3.3. Algunas opciones importantes del datagrama IP

Para la plena comprensión de las opciones que se detallan a continuación es recomendable la lectura previa del apartado 3.6.

*3.3.3.1. Registro de ruta o RR (*Record Route*)*

Esta opción proporciona un medio para almacenar las direcciones IP de los *routers* por los que es encaminado el datagrama. El *host* fuente es el encargado de configurar la opción dejando espacio suficiente para que se puedan almacenar las direcciones IP de los *routers* por los que va pasando el datagrama. Si el espacio para almacenar la ruta se llena antes de que el datagrama llegue a su destino, el datagrama se retransmitirá, pero se dejará de grabar la ruta.

En la figura 3.7 se puede observar el formato de esta opción:

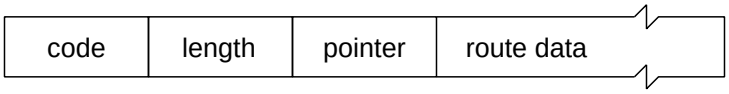


Figura 3.7: Formato de la opción RR

Donde:

Code contiene la clase y el número de opción para RR (**Code** tiene un valor decimal 7 para esta opción).

Lenght indica la longitud total en bytes de la opción.

Pointer se utiliza para apuntar a la siguiente ranura disponible para almacenar una dirección IP. Este campo está en bytes, por lo que cada *router* introduce su dirección IP en la ranura correspondiente e incrementa **Pointer** en 4. Si en un determinado *router* sucede que **Pointer** >**Lenght,** la ruta deja de grabarse desde ese *router*.

IP Address *n* indica la dirección registrada por el *router n.*

*3.3.3.2. Ruta fuente o SR (*source route*)*

Esta opción proporciona al emisor una forma de controlar de forma más fina el proceso de encaminamiento que usualmente se basa en una única dirección destino. Mediante esta opción el emisor puede proporcionar información sobre la ruta que deben seguir los datagramas[4].

[3] En realidad el proceso de reensamblado no es una tarea de la capa IP, este proceso depende de la capa superior (la capa de transporte). Es pues la capa de transporte la que ha de implementar este contador y reconocer la pérdida de fragmentos. Esta capa podría ser TCP para el transporte en un red orientada a conexión o UDP para el caso de redes no orientadas a conexión. El valor de este contador suele ser el campo **TTL**, aunque algunas implementaciones permiten configurarlo.

[4] Con esta opción se pueden encaminar datagramas a través de una determinada red física y de esta forma poder comprobar su estado, realizar tareas de mantenimiento etc.

IP proporciona dos formas de enrutado de fuente:

- Enrutado estricto de fuente o SSR (*Strict Source Routing*).
  La fuente proporciona la ruta "estricta" por la que los datagramas debe encaminarse hasta el destino. Cuando decimos que la ruta es estricta nos referimos a que la ruta entre dos direcciones sucesivas de la lista debe consistir en una sola red física. Si un *router* no puede encaminar directamente el datagrama a la siguiente dirección de la lista se produce un error y se genera un mensaje ICMP indicando destino inalcanzable (ver apartado 3.7).
- Enrutado no estricto de fuente o LSR (*Loose Source Routing*).
  En este caso la lista que especifica el emisor no es estricta, es decir, se permiten múltiples saltos de redes físicas entre direcciones sucesivas de la lista.

El formato es equivalente al de la opción RR (figura 3.7) pero el funcionamiento es un poco diferente:

- Para SSR y LSR el campo **Code** vale 137 y 131 respectivamente.
- El *host* origen pone en el campo de IP Destino la dirección del primer *router* de la ruta.
- Los *n* campos **IP Address** almacenan las direcciones IP de los *routers* de la ruta.
- Cuando un datagrama llega a su siguiente destino y la ruta fuente no está vacía (**pointer** <**lenght**) el receptor:
  - Toma la dirección del campo **IP Address** indicada por **pointer** y la coloca como nueva dirección destino del datagrama.
  - Substituye la dirección del campo **IP Address** indicada por **pointer** por su dirección IP local correspondiente a la red por la que se enviará el datagrama.
  - Incrementa **pointer** en 4 (es decir 32 bits).
  - Transmite el datagrama a la nueva dirección IP de destino.
  - Este procedimiento asegura que la ruta de retorno se graba en orden inverso de modo que el receptor puede usar los datos de la opción para construir la ruta de vuelta.

*3.3.3.3. Sello temporal o IT (*Internet Timestamp*)*

El *timestamp* o sello de tiempo es una opción para forzar a algunos (o a todos) los *routers* en la ruta hacia el destino a poner un sello temporal (*timestamp*) en los datos de la opción. Los *timestamps* se miden en segundos y se pueden usar para depuración. No se pueden emplear para medir el rendimiento de la red por dos razones:

1. No son lo bastante precisos porque la mayoría de los datagramas se envían en menos de un segundo.
2. No son lo bastante precisos porque los *routers* no suelen tener los relojes sincronizados.

3.4. Resolución de direcciones (ARP)

En una sola red física, los *hosts* individuales se conocen en la red a través de sus direcciones físicas o direcciones de nivel 2. Los protocolos de alto nivel encaminan los datagramas al destino mediante direcciones globales o direcciones de nivel 3 (para el caso de redes IP la dirección es una dirección IP) de la red virtual. Los *drivers* del dispositivo de red de la red física (por ejemplo los *drivers* de las tarjetas Ethernet de los PCs del laboratorio) no entienden las direcciones de nivel 3.

Por tanto, es necesario realizar una traducción de direcciones de nivel 3 a direcciones de nivel 2. En el caso de redes IP sobre Ethernet, la traducción se realiza de dirección IP a dirección Ethernet (que es una secuencia de 48 bits). Esta traducción se realiza en el *host* mediante una tabla usualmente denominada "caché ARP"[5]. Cuando la dirección no se encuentra en la caché ARP el *host* utiliza el protocolo ARP (*Address Resolution Protocol*) para averiguarla.

[5]Para consultar y manipular la cache ARP de un Linux se utiliza el comando `arp`. Para más información sobre este comando consulte la página de manual del arp.

ARP es un protocolo de petición/respuesta. La petición se envía en forma de *broadcast* de nivel 2 a la red. Si una de las máquinas de la red reconoce su propia dirección IP en la petición, devuelve una respuesta ARP al *host* que la solicitó. La respuesta contendrá la dirección física correspondiente a la dirección IP solicitada. Esta dirección se almacenará en la caché ARP del *host* solicitante. Todos los posteriores datagramas enviados a esta dirección IP se podrán asociar a la dirección física correspondiente, que será la que utilice el *driver* del dispositivo de red para mandar el datagrama. La figura 3.8 muestra el formato de la trama ARP.

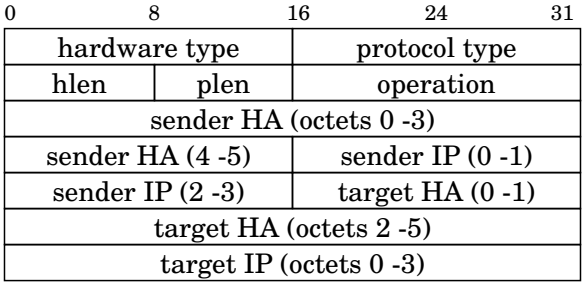


Figura 3.8: Formato de la trama ARP

Si una aplicación desea enviar datos a una determinada dirección IP de destino, el *software* de encaminamiento IP determina en primer lugar la dirección IP del siguiente salto (que puede ser el propio *host* de destino o un *router*) y en segundo lugar el dispositivo de red al que se debería enviar. A continuación se consulta el módulo ARP para hallar la dirección física.

El módulo ARP intenta hallar la dirección en su caché. Si la encuentra, devuelve la correspondiente dirección física. Si no la encuentra, descarta el paquete (se asume que al ser un protocolo de alto nivel volverá a transmitirlo) y genera un mensaje de broadcast de red para una solicitud ARP de dicha dirección física.

Cuando un *host* recibe un paquete ARP (bien un broadcast o una respuesta punto a punto), el dispositivo receptor pasa el paquete al módulo ARP, que lo añadirá a la caché ARP. La próxima vez que un protocolo de nivel superior quiera enviar un paquete a ese *host*, el módulo de ARP encontrará la dirección *hardware*, a la que se enviará el paquete.

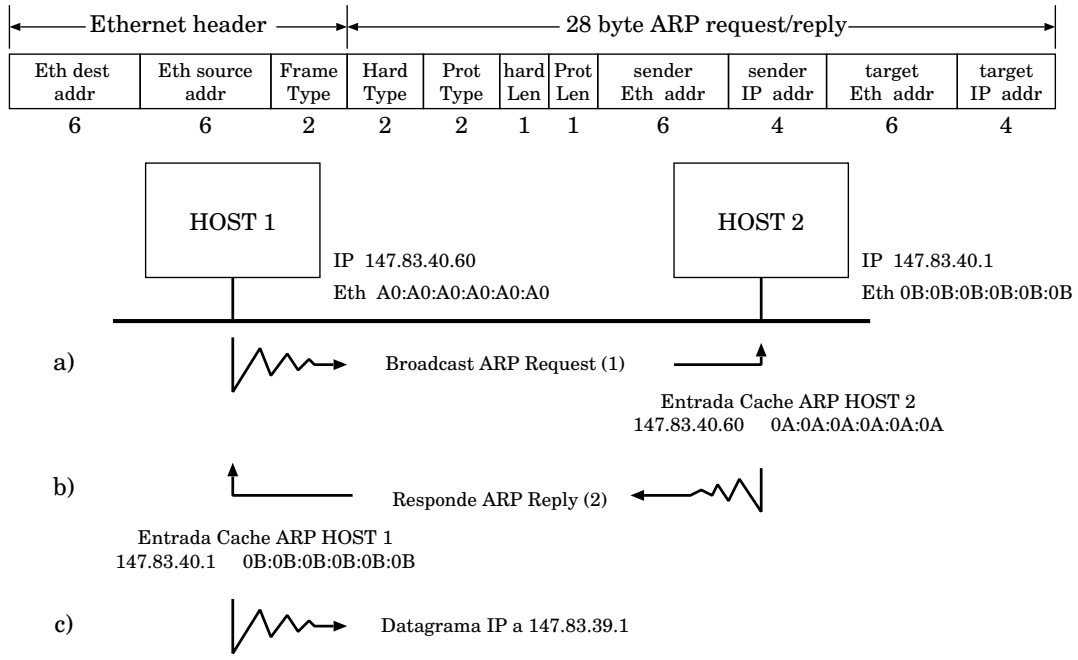
3.5. Las subredes

Es obvio que una dirección de clase A sólo se asignará a redes con un elevado número de *hosts*, y que las direcciones de clase C son adecuadas para redes con pocos *hosts*. Sin embargo, esto significa que las redes de tamaño medio (aquellas con más de 254 *hosts* o en las que se espera que en el futuro haya más de 254 *hosts*) deben usar direcciones de clase B. El número de redes de tamaño pequeño y medio ha ido creciendo muy rápidamente en los últimos años y se temía que, de haber permitido que se mantuviera este crecimiento, todas las direcciones de clase B se habrían usado para mediados de los 90. Esto es lo que se conoce como “el problema del agotamiento de las direcciones IP”.

Como se comentó en la sección 3.2 cada parte de la dirección IP (número de red y número de *host*) es responsabilidad de entidades diferentes: el número de red es asignado por INTERNIC, mientras que la asignación de los números de *host* es responsabilidad de la autoridad que controla la red. Como veremos a continuación el número de *host* puede dividirse aún más: esta división también es controlada por la autoridad propietaria de la red, y no por INTERNIC.

Debido al crecimiento explosivo de Internet, el uso de direcciones IP asignadas mediante las clases estándar se volvió demasiado rígido para permitir cambiar con facilidad la configuración de redes locales. Estos cambios podían ser necesarios cuando:

- Se instala una nueva red física.
- El crecimiento del número de *hosts* requiere dividir la red local en dos o más redes.



(1) ARP Request

FF:FF:FF:FF:FF:FF-A0:A0:A0:A0:A0:A0-0806-0001-0800-06-04-A0:A0:A0:A0:A0:A0-147.83.40.60-00:00:00:00:00:00-147.83.40.1

(2) ARP Reply

A0:A0:A0:A0:A0:A0-0B:0B:0B:0B:0B:0B-0806-0001-0800-06-04-0B:0B:0B:0B:0B:0B-147.83.40.1-A0:A0:A0:A0:A0:A0-147.83.40.60

Figura 3.9: Ejemplo de ARP

Para evitar tener que solicitar direcciones IP adicionales en estos casos, se introdujo el concepto de subred, donde el número de *host* de la dirección IP se subdivide de nuevo en un número de red y de *host*. Esta segunda red se denomina subred. La red principal consiste ahora en un conjunto de subredes y la dirección IP se interpreta como

<número de red, número de subred, número de *host*>

La combinación del número de subred y del *host* suele denominarse “dirección local” o “parte local”. La creación de subredes se implementa de forma que es transparente a redes remotas. Un *host* dentro de una red con subredes es consciente de la existencia de éstas, pero un *host* de una red distinta no lo es y sigue considerando la parte local de la dirección IP como un número de *host*.

La división de la parte local de la dirección IP en números de subred y de *host* queda a libre elección del administrador local. Cualquier serie de bits de la parte local se puede tomar para la subred requerida. Por ejemplo, el *subnetting* es muy utilizado por los ISPs (*Internet Service Provider*) que se encargan de dar servicio de acceso a la Internet, éstos suelen tener asignado un rango de direcciones públicas de Internet, que dividen en subredes para después asignarlas a sus clientes.

La división se efectúa empleando una máscara de red diferente a las que dividen las direcciones IP en clases estándar. El funcionamiento de estas máscaras es equivalente al que se describe en la sección 3.2 para las clases estándar, es decir, los bits a cero en esta máscara indican posiciones de bits correspondientes al número de *host*, y los que están a uno, indican posiciones de bits correspondientes al número de subred. Las posiciones de la máscara pertenecientes al número de red se ponen a uno pero no se usan.

El tratamiento especial de “todos los bits a cero” y “todos los bits a uno” se aplica a cada una de las tres partes de dirección IP con subredes del mismo modo que a una dirección IP que no las tenga. Notar que al hacer *subnetting*, el número de máquinas sobre las que se hace *broadcast* (difusión) se reduce, optimizando el tráfico útil dentro de la red.

Como ejemplo supongamos que tenemos la red privada de clase C 192.168.1.0/24. En este caso tendremos un total de 254 (es decir $2^8 - 2$, por tener 8 bits de *hostid*) direcciones posibles de *host*, de la 192.168.1.1/24 hasta la 192.168.1.254/24. Observe que las direcciones 192.168.1.0 y 192.168.1.255 se reservan, la primera para identificar la red y la segunda como dirección de *broadcast*. Supongamos que hemos de repartir este direccionamiento en dos subredes de la misma dimensión, ya que tenemos dos oficinas en poblaciones diferentes y montaremos dos LANs unidas por la WAN (*Wide Area Network* o red de transporte que alquilaremos a algún operador). Así pues, tenemos que la parte de *hostid* ha de ceder terreno para la parte de subred. Como necesitamos hacer dos subredes, con un bit tendremos suficiente para identificar la subred en la que nos encontramos, quedándonos dos subredes: la 192.168.1.0/25 y la 192.168.1.128/25, con 126 direcciones posibles de *host* en cada subred (es decir $2^7 - 2$, con 7 bits de *hostid*). A continuación resumimos las direcciones de red, de máscara y de *hosts* implicadas en el *subnetting* de este ejemplo.

Red básica original: 11000000. 10101000. 00000001. 00000000 = 192.168.1.0/24

Máscara: 11111111. 11111111. 11111111. 00000000 = 255.255.255.0 = /24

Hay 254 direcciones posibles para asignar a la red:

host # 0 11000000. 10101000. 00000001. 00000001 = 192.168.1.1/24 (*netmask* 255.255.255.0)

host # 1 11000000. 10101000. 00000001. 00000010 = 192.168.1.2/24 (*netmask* 255.255.255.0)

host # 253 11000000. 10101000. 00000001. 11111110 = 192.168.1.254/24 (*netmask* 255.255.255.0)

Si hacemos *subnetting* nos quedarán estas dos subredes:

Subnet # 0 11000000. 10101000. 00000001. 00000000 = 192.168.1.0/25 (*netmask* 255.255.255.128)

Subnet # 1 11000000. 10101000. 00000001. 10000000 = 192.168.1.128/25 (*netmask* 255.255.255.128)

Con las siguientes direcciones de *host* disponibles en cada oficina:

Subnet # 0 de una oficina:

host # 0 11000000. 10101000. 00000001. 00000001 = 192.168.1.1/25 (*netmask* 255.255.255.128)

host # 1 11000000. 10101000. 00000001. 00000010 = 192.168.1.2/25 (*netmask* 255.255.255.128)

host # 125 11000000. 10101000. 00000001. 01111110 = 192.168.1.126 / 25 (*netmask* 255.255.255.128)

Subnet # 1 de la otra oficina:

host # 0 11000000. 10101000. 00000001. 10000001 = 192.168.1.129/25 (*netmask* 255.255.255.128)

host # 1 11000000. 10101000. 00000001. 10000010 = 192.168.1.130/25 (*netmask* 255.255.255.128)

host # 125 11000000. 10101000. 00000001. 11111110 = 192.168.1.254/25 (*netmask* 255.255.255.128)

3.6. Encaminamiento IP

3.6.1. Introducción

La función más importante de la capa IP es el encaminamiento de datagramas extremo a extremo a través de la red virtual. Por tanto esta función proporciona los mecanismos necesarios para interconectar distintas redes físicas. La formación de la red virtual que conecta múltiples redes se consigue por medio de unos *hosts* especiales denominados "*routers*".

Es importante distinguir entre un *hub*, un puente, un *router* y una pasarela.

El *hub* interconecta segmentos de LAN a nivel de interfaz de red y envía tramas entre ellos. El *hub* sirve como prolongación del cable físico que conecta las máquinas de la LAN y su única función es difundir la señal que llega por un cierto puerto (entrada) al resto de puertos. Los hubs pueden ser pasivos (si no amplifican las señales recibidas por sus puertos) o activos (si las amplifican).

El conmutador (*switch*) es un dispositivo parecido al *hub* pero en el que se realiza conmutación entre sus diferentes puertos, es decir, conmuta los paquetes observando sus direcciones físicas origen/destino.

Un puente (*bridge*) interconecta segmentos de LAN a nivel de interfaz de red y envía tramas entre ellos. Un puente realiza la función de retransmisión MAC (*Medium Access Control*) y es independiente de cualquier capa superior (incluyendo LLC). Proporciona, si se necesita, conversión de protocolos a nivel MAC. Un puente es transparente para IP. Es decir, cuando un *host* envía un datagrama a otro *host* en una red con el que se conecta a través de un puente, envía el datagrama al *host* y el datagrama cruza el puente sin que el emisor se dé cuenta. El puente es capaz de aprender las direcciones *hardware* de las máquinas que tiene en cada puerto y aislar el tráfico y las colisiones de cada tramo LAN.

Un *router* interconecta redes físicas diferentes a nivel de la capa de red y encamina paquetes entre ellas. El *router* debe comprender la estructura de direccionamiento asociada con los protocolos que soporta (IP en nuestro caso) y debe elegir las mejores rutas de transmisión así como tamaños óptimos para los datagramas realizando fragmentación si lo considera oportuno.

La pasarela (*gateway*) interconecta redes a niveles superiores que los puentes y los *routers*. Una pasarela suele soportar el mapeado de direcciones de una red a otra, así como la transformación de datos entre distintos entornos para conseguir conectividad entre los extremos de la comunicación. Las pasarelas típicamente proporcionan conectividad de dos redes para un subconjunto de protocolos de aplicación soportados en cada una de ellas. Una pasarela es opaca para IP. Es decir, un *host* no puede enviar un datagrama IP a través de una pasarela: sólo puede enviarlo a la pasarela. La pasarela se ocupa de transmitirlo a la otra red con la información de los protocolos de alto nivel que vaya en él.
Estrechamente ligado al concepto de pasarela, está el de **cortafuegos** (*firewall*), que se usa para restringir la circulación de datagramas entre redes por motivos de seguridad.

3.6.2. Entrega directa e indirecta

El encaminamiento IP se puede dividir en dos partes: entrega directa y entrega indirecta:

- La entrega directa es la transmisión de un datagrama desde el *host* origen hasta el *host* destino a través de una sola red física, dicho de otra forma, dos *hosts* sólo pueden comunicarse mediante entrega directa si ambos están conectados directamente a la misma red física (por ejemplo, una sola red Ethernet).
  Básicamente en la entrega directa el emisor encapsula el datagrama dentro de una trama física, transforma la dirección IP destino en una dirección física y envía la trama resultante al destino a través del *driver* del dispositivo *hardware* correspondiente.

- La entrega indirecta es necesaria cuando el *host* destino no está conectado directamente a la red del origen, lo que implica que el datagrama deberá atravesar varias redes físicas, y para ello es necesario atravesar *routers*.
  La entrega indirecta es más compleja ya que el *host* origen ha de identificar al *router* al que debe entregar el datagrama, el primer *router* debe identificar cuál será el siguiente *router* al que debe enviar el datagrama, esto también se denomina identificar el "siguiente salto".
  La comunicación entre dos *routers* consecutivos de la ruta se realiza siempre mediante entrega directa, es decir, un determinado *router* de la ruta y el *router* del siguiente salto deben estar conectados a la misma red física, sino no es posible su comunicación. A su vez, el último *router* de la ruta que sigue el datagrama debe estar conectado a la misma red física que el *host* destino.

El *host* origen averigua si debe realizar entrega directa o no, es decir, si el *host* destino está conectado o no directamente a su red física mediante el prefijo de red. El *host* origen extrae el número de red de la dirección IP del destino y la compara con el número de red de su propia dirección IP. Si ambas se corresponden significa que se puede utilizar entrega directa, sino se ha de utilizar entrega indirecta.

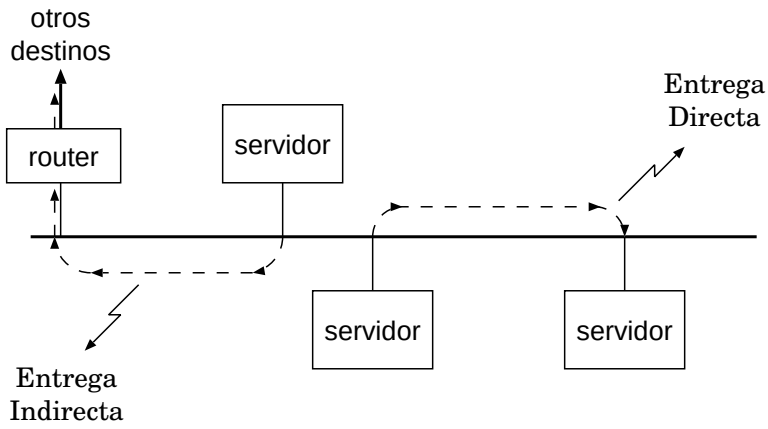


Figura 3.10: Entrega directa e indirecta

3.6.3. El algoritmo de encaminamiento

El método utilizado por un *router* o un *host* para averiguar la siguiente máquina a la que debe enviar un determinado datagrama se denomina genéricamente como el "algoritmo de encaminamiento".

La gran mayoría de algoritmos de encaminamiento utilizan "tablas de encaminamiento". En las tablas de encaminamiento de cada *host* o *router* se almacena información sobre los posibles destinos y sobre cómo alcanzarlos. La información que contienen las tablas de encaminamiento debe ser mínima, ya que si cada tabla de encaminamiento contuviera información sobre cada posible dirección destino sería imposible mantener actualizadas las tablas. Además, las máquinas no tendrían suficiente espacio ni capacidad de proceso para manejarlas.

Se trata de minimizar la información que deben guardar las tablas aplicando un esquema de ocultación de información global, manteniendo sólo la información local mínima necesaria. Afortunadamente el esquema de direccionamiento IP permite realizar esto de forma fácil: como se mencionó en la sección 3.2, la dirección IP se divide en número de red y en número de *host*. Mediante este esquema es posible almacenar números de red en las tablas de encaminamiento en lugar de direcciones IP completas. De esta forma se ocultan los detalles de qué *hosts* y cómo están conectados a las diferentes redes y se minimiza el tamaño de las tablas de encaminamiento.

El contenido de las tablas de encaminamiento suelen ser pares del tipo <N,R>. Donde N es un número de red y R es la dirección IP del *router* en el siguiente salto para alcanzar dicha red (por tanto el *router* debe estar conectado a la misma red física).

Para simplificar más las tablas de encaminamiento aparece el concepto de "ruta por defecto". La ruta por defecto contiene la dirección del *router* del siguiente salto al que se deben enviar los datagramas (también denominado *router* por defecto) si tras recorrer la tabla de encaminamiento no se encontró ninguna ruta específica para el número de red al que va dirigido el datagrama.

Aunque se ha comentado la conveniencia de encaminar en base al número de red destino, la tabla de encaminamiento permite especificar una ruta especial para un *host* en particular.

De esta forma el algoritmo básico de encaminamiento de un datagrama IP es el siguiente:

1. Extraer la dirección IP destino D.
2. Computar el prefijo de red N con la máscara local.
3. Si N se corresponde con alguna red física a la que estamos conectados se realiza entrega directa (realizando ARP).
4. Si no se puede realizar entrega directa, se comprueba si hay ruta específica para D y en caso afirmativo se envía el datagrama al *router* del salto siguiente especificado en la tabla.

5. Si no hay ruta específica, se comprueba si hay una ruta para la red N y en caso afirmativo se envía el datagrama al *router* del salto siguiente especificado en la tabla.
6. Si no hay ruta para N, se envía el datagrama al *router* por defecto especificado en la tabla.
7. Si no hay ruta por defecto y el *software* de encaminamiento IP ha llegado a este punto se produce un error (que se puede reportar mediante ICMP).

Como conclusiones importantes del algoritmo de encaminamiento IP podemos destacar que:

1. En la mayor parte de implementaciones, el tráfico dirigido a una determinada red desde un *host* origen va a seguir el mismo camino aunque existan diversas posibilidades.
2. Sólo el último *router* de la ruta puede determinar si el *host* destino está disponible (estas situaciones se reportan mediante ICMP). Además también necesitamos reportes de los *routers* intermedios si sucede algún problema.
3. Los datagramas que viajen de A a B pueden seguir rutas diferentes a los datagramas que viajen de B a A.

La figura 3.11 muestra 4 redes interconectados a través de 2 *routers* y la tabla de rutas del *router* R_1.

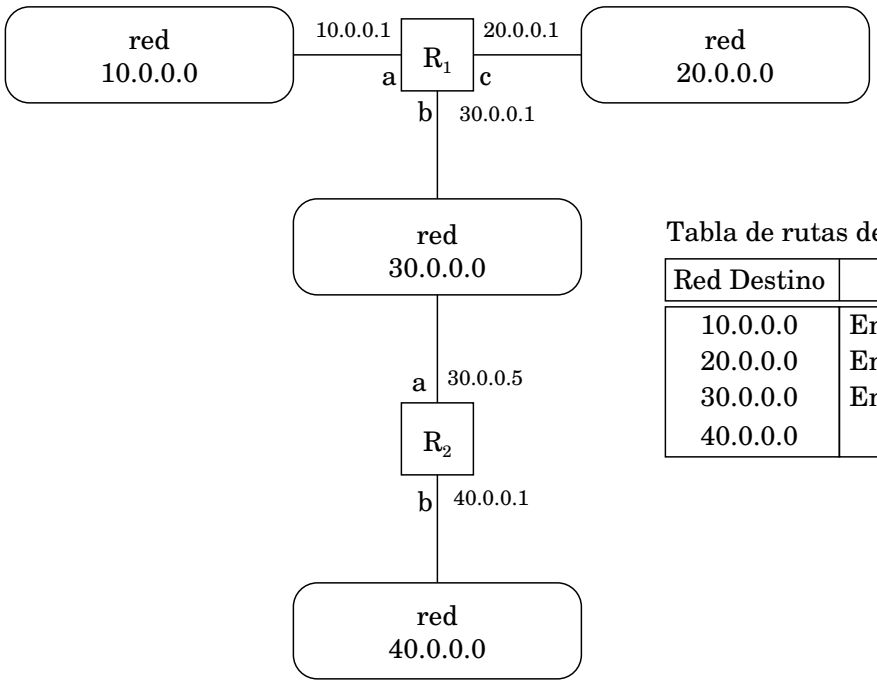


Tabla de rutas de R_1

Red Destino	Enrutar a:	iF
10.0.0.0	Entrega directa	a
20.0.0.0	Entrega directa	c
30.0.0.0	Entrega directa	b
40.0.0.0	30.0.0.5	b

Figura 3.11: Ejemplo de encaminamiento

Es importante entender que a excepción de la disminución del campo TTL el *software* de encaminamiento no modifica la cabecera del datagrama original. En particular, las direcciones IP origen y destino permanecen inalteradas durante toda la ruta[6].

Por lo que respecta a los datagramas entrantes:

- Cuando un datagrama llega a un *host* el *driver* del dispositivo de red lo entrega al *software* IP para su procesamiento. El *software* IP determina si el datagrama es para el propio *host* en cuyo caso lo pasa al *software* del protocolo de nivel alto apropiado. El datagrama se descarta si no es para el propio *host*.
- En el caso de los *routers*, estos deben encaminar el datagrama si no va dirigido hacia ellos.

Decidir si una máquina es o no la destinataria de un datagrama no es una tarea tan trivial como a simple vista pueda parecer. En primer lugar pueden haber muchos interfaces de red cada uno de ellos con su correspondiente dirección IP,

[6]Excepto en la opción SR comentada en la sección 3.3.3.2

type	*code*	*Description*	*Query*	*Error*
0	0	echo reply	•	
3		destination unreachable		
	0	network unreachable		•
	1	host unreachable		•
	2	protocol unreachable		•
	3	port unreachable		•
	4	fragmentation needed but fg not set		•
	5	source route failed		•
	6	destination network unknown		•
	7	destination host unknown		•
	8	source host isolated (obsolete)		•
	9	destination network adimistrativly prohibited		•
	10	destination host administratively prohibited		•
	11	network unreachable for TOS		•
	12	host unreachable for TOS		•
	13	communication administratively prohibited by filtering		•
	14	host precedence violation		•
	15	precedence cut-off in effect		•
4	0	source quench (elementary flow control)		•
5		redirect		•
	0	redirect for network		•
	1	redirect for host		•
	2	redirect for TOS and network		•
	3	redirect for TOS and host		•
8	0	echo request	•	
9	0	router advertisement	•	
10	0	router solicitation	•	
11		time exceeded		
	0	time-to-live equals 0 during transit		•
	1	time-to-live equals 0 during reassembly		•
12		parameter problem		
	0	IP header bad		•
	1	required option missing		•
13	0	timestamp request	•	
14	0	timestamp reply	•	
15	0	information request (obsolete)	•	
16	0	information reply (obsolete)	•	
17	0	address mask request	•	
18	0	address mask reply	•	

Tabla 3.4: Mensajes ICMP

se debe comprobar la correspondiente identificación de subred (si la red está dividida) y además se deben reconocer los mensajes de *broadcast* y los de *multicast*.

3.7. ICMP- Internet Control Message Protocol

ICMP (*Internet Control Message Protocol*), es un protocolo que se utiliza para señalizar errores u otro tipo de condiciones que se pueden producir durante una comunicación. Estos mensajes pueden ser generados por cualquier dispositivo

de red de nivel 3 que necesite informar de una condición de error a un dispositivo emisor cuya transmisión ha causado el error. Esta condición implica que los mensajes ICMP deberán ir encapsulados en datagramas IP, para que puedan ser encaminados desde cualquier punto de la red hasta el receptor del mensaje ICMP. Sin embargo, como puede verse en la figura 3.2, ICMP no utiliza ningún protocolo de transporte ya que el mensaje ICMP no va dirigido a la aplicación de comunicaciones que generó el error, sino a la torre de comunicaciones (TCP/IP) gestionada por el Sistema Operativo que será quien trate el mensaje ICMP y actúe en consecuencia (por ejemplo indicando la condición de error a la aplicación).

Un mensaje ICMP tiene la estructura mostrada en la figura 3.12. Tanto el campo de tipo (*type*) como el campo de

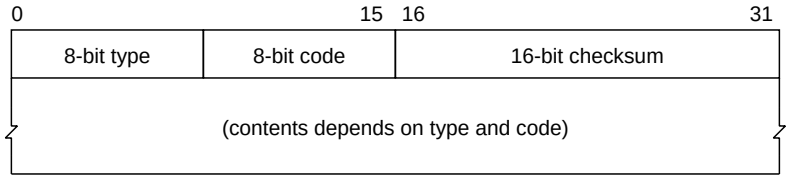


Figura 3.12: Mensaje ICMP

código (*code*), determinan los diferentes mensajes ICMP, así como formato del contenido del mensaje. La tabla 3.4 muestra los diferentes tipos de mensajes ICMP clasificados según el campo *type* y el campo *code*.

Es importante destacar que muchos de los mensajes ICMP son obsoletos, o bien algunos de ellos estan implementados en *hosts* y otros en *routers*, o bien algunos de ellos están específicamente deshabilitados porque han demostrado ser puntos débiles de seguridad.

Las dos últimas columnas de la tabla indican que hay dos tipos de mensajes ICMP, peticiones (*query*) y errores (*error*). Los mensajes ICMP de error son mensajes que se envían al dispositivo cuyo paquete ha generado el error. En este caso, los mensajes ICMP de error incluyen la cabecera IP del datagrama que generó el error (20 bytes) y los primeros 8 bytes del datagrama (incluiría el inicio de la cabecera TCP ó UDP). De esta forma, el dispositivo que generó el error sabe qué aplicación es la responsable del error generado.

Los mensajes ICMP de petición se utilizan para obtener información específica (según el tipo y código del mensaje ICMP) de un dispositivo de red (*hosts*, *routers*, ...). El mensaje de petición corresponde a los etiquetados con *request* en la tabla, mientras que los mensajes de respuesta son los *reply*.

3.7.1. ICMP *echo* y el programa `ping`

Los mensajes ICMP de *echo* han sido utilizados, tradicionalmente, para verificar el estado de las conexiones entre dos dispositivos de red (*hosts*, *routers*, ...).

El formato del mensaje ICMP de *echo* es igual que el mostrado en la figura **??**, donde el campo *type* tiene los valores 0 u 8 (*reply*, *request*) y el campo *code* tiene el valor 0. El *host* que envía un mensaje de petición (*request*) ICMP de *echo*, establece los campos *identifier* y *sequence number* a unos valores que él escoge, así como el contenido de los datos. El *host* que recibe la petición (*request*) responde con un mensaje de respuesta (*reply*) con los mismos valores de los campos *identifier*, *sequence number* y contenido que recibió (realiza un "eco" del mensaje recibido).

El programa `ping` se utiliza para enviar una petición de mensaje ICMP de *echo*. La respuesta (*reply*) se maneja directamente en el *kernel* (núcleo) del sistema y automáticamente envía el mensaje de respuesta (*reply*). En las implementaciones de Unix, el campo *identifier* se establece con el valor del PID del proceso que envía los mensajes, lo que permite a `ping` identificar las respuestas recibidas en el caso de que hayan más procesos `ping` ejecutándose.

El valor del campo *sequence number* se inicializa a 0 y se incrementa en cada mensaje nuevo que se envía, permitiendo detectar si se ha perdido algún paquete.

El programa `ping` calcula el RTT (*round trip time*), tiempo transcurrido entre que se envía una petición de *echo* y se recibe la respuesta correspondiente.

3.8. Estudio Previo

1. Clasifica las siguientes direcciones IP según la clase a la que pertenecen, explicando a qué hacen referencia:
   a) 127.22.50.42
   b) 325.24.13.0
   c) 241.25.3.1
   d) 10.0.1.255
   e) 192.168.10.0
2. Suponiendo que una red utiliza direcciones IP de clase C y que dispone de un único *router* para conectarse a Internet. ¿Cuál es el número máximo de estaciones que podríamos conectar a la red?
3. Asuma que usted tiene asignado el bloque de red 132.45.0.0/16. Usted necesita establecer 8 subredes.
   a) ¿Cuántos dígitos binarios necesita para definir las 8 subredes?
   b) Exprese las subredes en notación decimal con puntos
   c) Liste el rango de direcciones de *hosts* que pueden ser asignadas a la *Subnet* #3 (132.45.96.0/19)
   d) ¿Cuál es la dirección *broadcast* para la *Subnet* #3?

3.9. Ejercicios

Ejercicio 1

En este ejercicio se van a estudiar los paquetes ICMP de *echo*. Para realizar este ejercicio, se utilizará un capturador de tráfico llamado `ethereal`, que funciona en modo gráfico. La configuración del conexionado del laboratorio es la representada en la figura 3.13. Para enviar paquetes ICMP de echo, utilizaremos el programa `ping`, (ejecute el comando `man ping` para obtener la ayuda de funcionamiento del comando). Realice los siguientes pasos para obtener una captura del tráfico ICMP:

1. Abra el programa `ethereal`, tecleando desde una consola `ethereal`. Escoja la opción de menú `capture` ▷ `start`. Se abrirá una nueva ventana (figura 3.14):
   a) En la entrada etiquetada con el nombre de *Interface*, escoja *eth0* (*interface 0*).
   b) En la entrada etiquetada con el nombre de *filter* escriba:
   `host 147.83.40.2x`
   Sustituya la x por el dígito de su máquina. Este filtro indica al motor de captura que sólo lea de la red los datagramas cuya dirección IP origen o destino sea `147.83.40.2x`.
   c) Deshabilite las opciones:
      1) *Enable MAC name resolution*
      2) *Enable network name resolution*
      3) *Enable transport name resolution*
2. Una vez preparado el programa de captura de tráfico, generaremos paquetes ICMP de *echo* con el programa `ping`. Para ello ejecute desde una consola el siguiente comando:
   `telem2-x# ping -c 3 147.83.40.29`
   Con este comando se enviarán 3 paquetes ICMP *echo request*.

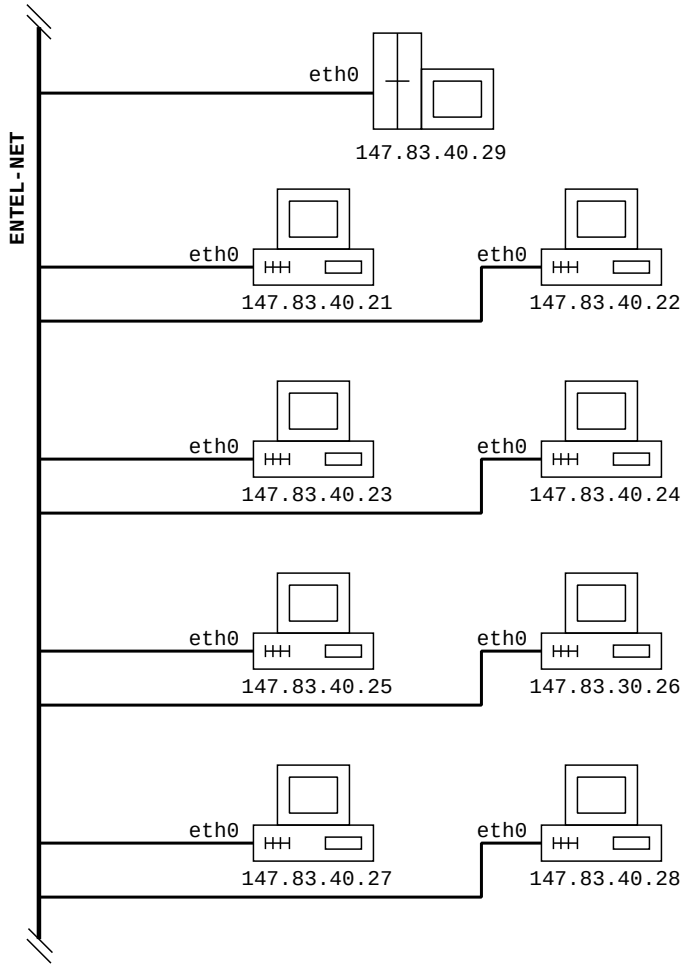


Figura 3.13: Configuración de laboratorio para el ejercicio 1

3. Una vez ha acabado el programa `ping`, pulse el botón de *stop* de `ethereal`, para detener la captura de tráfico.

Analizando los paquetes capturados, responda a las siguientes preguntas:

1. ¿Se identifican tramas ARP en la captura?
2. ¿Si efectivamente hay tramas ARP en la captura, para qué son necesarias?. En caso de que no hubiesen tramas ARP, ¿por qué no hacía falta enviarlas?
3. ¿Pertenece la dirección `147.83.40.29` a su red local? Indique si se ha realizado una entrega directa o indirecta de los mensajes ICMP.
4. Analice la dirección *hardware* destino de uno de los mensajes ICMP *request*. ¿A qué máquina corresponde?
5. Analice, ahora, la dirección *hardware* fuente de uno de los mensajes ICMP *reply*. ¿A qué máquina corresponde?

Realice ahora los siguientes pasos:

1. Escoja la opción de menú `capture` ▷ `start`.
2. Desde una consola ejecute el siguiente comando:
   ```
   telem2-x# ping -c 3 147.83.39.1
   ```
3. Una vez ha acabado el programa `ping`, pulse el botón de *stop* de `ethereal`, para detener la captura de tráfico.

Analizando los paquetes capturados, responda a las siguientes preguntas:

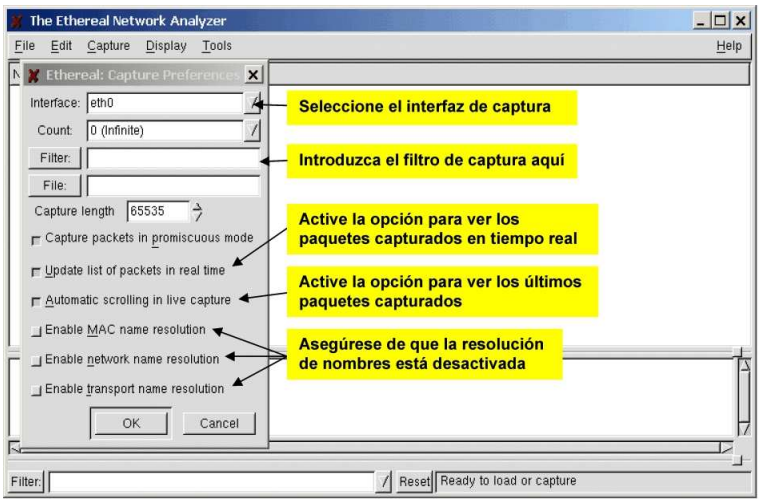


Figura 3.14: Captura de paquetes con *ethereal*.

1. ¿Se identifican tramas ARP en la captura?
2. ¿Si efectivamente hay tramas ARP en la captura, para qué son necesarias? En caso de que no hubiesen tramas ARP, ¿por qué no hacía falta enviarlas?
3. ¿Pertenece la dirección `147.83.39.1` a su red local? Indique si se ha realizado una entrega directa o indirecta de los mensajes ICMP. ¿Si la entrega ha sido indirecta, a través de que *router* se habrá realizado?
4. Analice la dirección *hardware* destino de uno de los mensajes ICMP *request*. ¿A qué máquina corresponde?
5. Analice, ahora, la dirección *hardware* fuente de uno de los mensajes ICMP *reply*. ¿A qué máquina corresponde?

Ejercicio 2

El objetivo de este ejercicio es observar la fragmentación de los datagramas IP. Para ello se utilizará el programa `ping` para generar mensajes ICMP *echo* de petición (*request*), con un tamaño suficientemente grande y el programa `ethereal` para capturar el tráfico generado y poder analizarlo. La configuración del laboratorio que se debe utilizar en este ejercicio es la misma que en el ejercicio 1.

1. Averigüe la opción que permite establecer el tamaño del mensaje ICMP usando el manual del comando `ping`.
2. Envíe un solo mensaje ICMP de tamaño 1000 bytes a la dirección `147.83.40.29`, realizando la captura del tráfico (refiérase al ejercicio 1 para la sintaxis del filtro de captura).
   a) Analizando el tráfico capturado determine el tamaño total de la trama enviada, la longitud de la cabecera *Ethernet*, longitud de la cabecera IP, longitud de la cabecera ICMP y la longitud de los datos enviados.
3. Envíe un solo mensaje ICMP de tamaño 10000 bytes a la dirección `147.83.40.29`, realizando la captura del tráfico correspondiente. Analizando el tráfico capturado responda a las siguientes preguntas:
   a) ¿Cuantas tramas *Ethernet* han sido enviadas a la dirección `147.83.40.29` para completar la transmisión de un solo mensaje ICMP?
   b) Analice qué cabeceras de protocolos existen en cada uno de las tramas *Ethernet* del mensaje ICMP.
   c) Determine cuantos bytes de información viajan en cada una de las tramas (los datos enviados deben sumar 10000 bytes)
   d) Obtenga una fórmula matemática que permita determinar el número de tramas enviadas para completar la transmisión de un mensaje ICMP de una longitud de L bytes.

4. Obtenga el MTU (*Maximum Transfer Unit*) de la interfaz de red por la que se han enviado los mensajes ICMP. ¿Tiene alguna relación el MTU con la fragmentación observada en el punto anterior?

Ejercicio 3

La red privada que disponemos en el laboratorio es la 192.168.2.0, quedando ya definidas la dirección de *broadcast* 192.168.2.255 y la máscara 255.255.255.0. Con dicha red disponemos de 254 direcciones para *host*, más las direcciones de *broadcast* y de red. Vamos a subdividirla en subredes. Para ello, aquí veremos un ejemplo en que la subdividimos en dos subredes. Queda para el alumno subdividirla en más subredes al realizar el ejercicio propuesto a continuación.

Con la máscara 255.255.255.128 (11111111.11111111.11111111.10000000) nos quedan estas dos subredes:

`Subred #0: 192.168.2.0;Broadcast:192.168.2.127;Direcciones para hosts:128-2=126.`

`Subred #1: 192.168.2.128;Broadcast:192.168.2.255;Direcciones para hosts:128-2=126.`

Con el programa `ifconfig` podemos modificar la IP de una máquina, la máscara y la dirección broadcast con este comando:

`ifconfig interfaz` *`IP_Host`* `netmask` *`IP_Netmask`* `broadcast` *`IP_Brd`*

Las 8 máquinas de los alumnos en el laboratorio deben tener las siguientes direcciones IP asignadas:

Tabla de IP para los *hosts* del laboratorio

telem2-1	192.168.2.16
telem2-2	192.168.2.48
telem2-3	192.168.2.80
telem2-4	192.168.2.112
telem2-5	192.168.2.144
telem2-6	192.168.2.176
telem2-7	192.168.2.208
telem2-8	192.168.2.240

1. Divida la subred privada del laboratorio 192.168.2.0 en dos subredes con la mitad de direcciones IP en cada una. ¿Qué máscara ha utilizado? ¿Cuáles son las direcciones de red, de *broadcast* y las asignadas para los *hosts* en cada una de las subredes? ¿Cuántas direcciones IP hay para los *hosts* en cada subred?
2. Agrupe las máquinas del laboratorio (telem2-1...telem2-8) en estas dos subredes que ha creado en el apartado anterior. ¿En qué subred está cada máquina?
3. Pruebe el comando `ping` con cada una de las direcciones. ¿Qué direcciones responden? Justifique la respuesta
4. Divida el espacio de direcciones original 192.168.2.0 en 4 subredes de igual tamaño. Indique cuál es la máscara necesaria. Indique la dirección de subred y la de *broadcast* que tiene cada subred. ¿Cuántas direcciones IP hay por subred?
5. Pruebe el comando `ping` con cada una de las direcciones. ¿Qué direcciones responden? Justifique la respuesta
6. Divida el espacio de direcciones original 192.168.2.0 en 8 subredes de igual tamaño. Indique cuál es la máscara necesaria ¿Cuántas direcciones IP hay por subred?
7. Pruebe el comando `ping` con cada una de las direcciones. ¿Qué direcciones responden? Justifique la respuesta
8. ¿Cuál es el número máximo de subredes posible que se pueden crear en el rango 192.168.2.0? Justifique la respuesta.

Ejercicio 4

Para probar las consecuencias de un error en la máscara de subred configure los siguientes parámetros para cada *host* del laboratorio

```
  Host            Dirección IP            Máscara
telem2-1         192.168.2.16          255.255.255.0
telem2-2         192.168.2.48          255.255.255.128
telem2-3         192.168.2.80          255.255.255.0
telem2-4         192.168.2.112         255.255.255.128
telem2-5         192.168.2.144         255.255.255.0
telem2-6         192.168.2.176         255.255.255.128
telem2-7         192.168.2.208         255.255.255.0
telem2-8         192.168.2.240         255.255.255.128
```

1. Pruebe el comando `ping` con cada una de las direcciones. ¿Qué direcciones responden? Consulte los resultados con los grupos de las estaciones de trabajo contiguas.
2. Junto con el grupo de trabajo de la estación contigua, justifiquen a qué se debe el comportamiento de los *hosts* con una determinada máscara.

Ejercicio 5

En este ejercicio se pretende simular una posible configuración de red para la empresa Acme SA. La empresa Acme está formada por dos direcciones: la dirección técnica y la dirección de marketing. A su vez marketing se divide en dos departamentos: compras y ventas. Por su parte la dirección técnica se divide en los departamentos de producción y sistemas (ver figura 3.15).

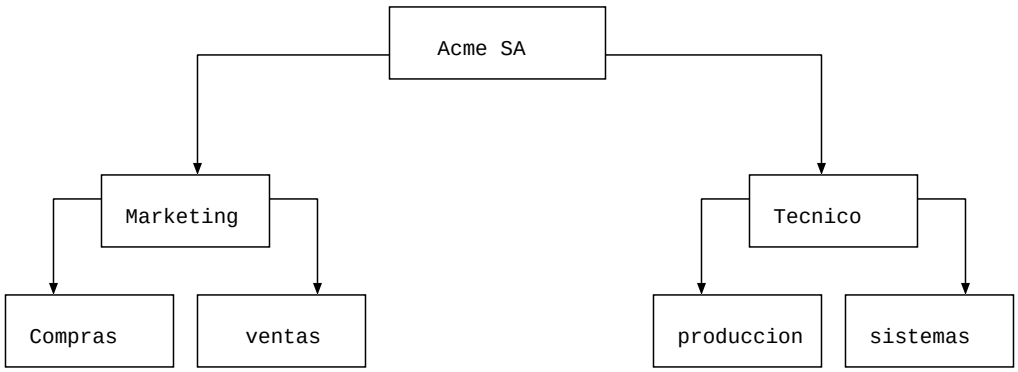


Figura 3.15: Estructura de la empresa Acme SA.

En nuestro laboratorio vamos a asociar un par de máquinas a cada departamento, donde una de ellas hará de *host* y la otra de *router* que se configurarán con los datos de la siguiente tabla 3.5.

Máquina	Departamento	Dirección IP	tipo
telem2-1	compras	192.168.0.1	*router*
telem2-2	compras	192.168.0.254	*host*
telem2-3	ventas	192.168.1.1	*router*
telem2-4	ventas	192.168.1.254	*host*
telem2-5	producción	192.168.2.1	*router*
telem2-6	producción	192.168.2.254	*host*
telem2-7	sistemas	192.168.3.1	*router*
telem2-8	sistemas	192.168.3.254	*host*

Tabla 3.5: Datos de configuración para el ejercicio 5

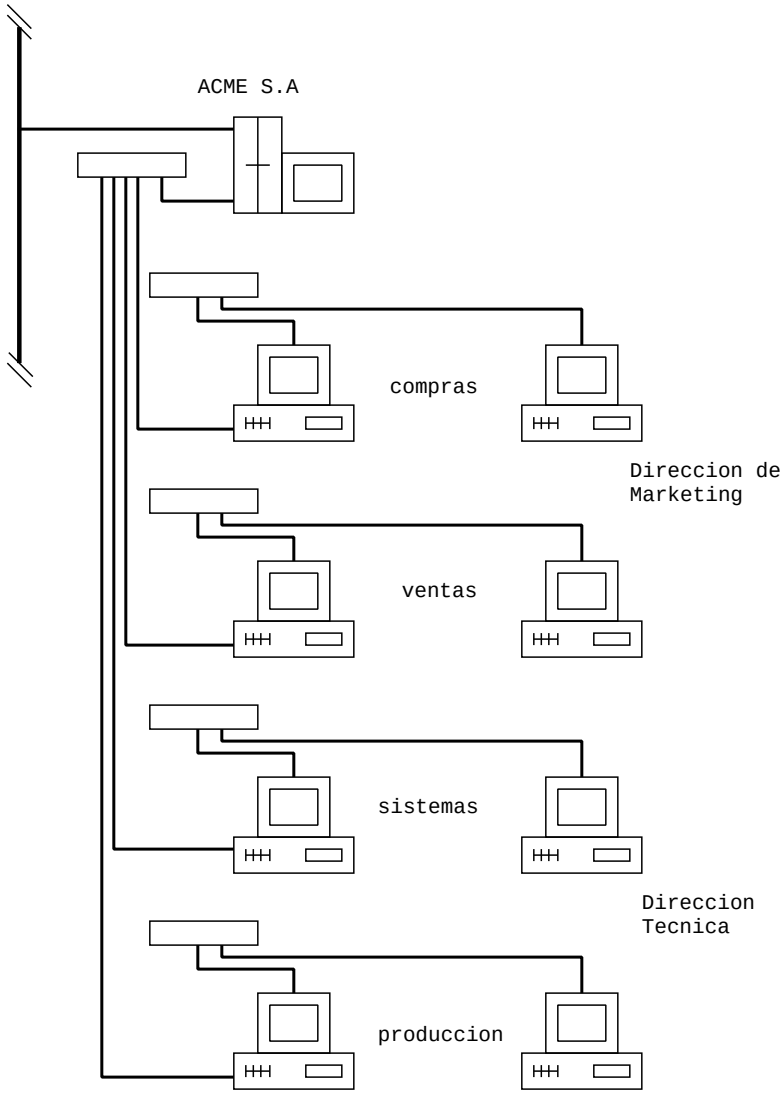


Figura 3.16: Conexionado para el ejercicio 5

La estructura que se va a utilizar se puede observar en la figura 3.16, donde todos los *routers* están conectados mediante sus interfaces *eth0* a través del *hub* situado en el puesto del profesor. El número de red para la red de *routers* sera el 10.0.0.0/24.

1. Sabiendo que la máquina del profesor tiene la dirección 10.0.0.1 para la red que une los *routers* elija una dirección para cada uno de los *routers* consensuándolo con sus compañeros.
2. Elija una máscara de red adecuada con sus compañeros de departamento .
3. Configure el interfaz de red *eth1* (@IP+*netmask*+*broadcast*) de su máquina, con los datos que se especifican en la Tabla 3.5 y la máscara elegida en 2. El comando que debe utilizarse es:
   `ifconfig <Interface><IPaddress>netmask <Netmask>broadcast <Bcast_address>`
4. Con la configuración actual, ¿a que máquinas puede acceder?¿cómo lo ha comprobado?
5. A continuación junto con los miembros de su departamento, debe configurar el interfaz *eth0* (@IP+*netmask*+*broadcast*) del *router* que tienen asignado.
6. Con la configuración actual, ¿a que máquinas puede acceder su *router*?¿cómo lo ha comprobado?

A continuación se va a realizar la configuración necesaria para posibilitar la comunicación entre los departamentos de la misma dirección. Verifique que el *router* de su departamento puede enrutar paquetes. Para ello edite el fichero `/proc/sys/net/ipv4/ip_forward` para ver si contiene un 1.

1. Configure el router de su departamento para que pueda enrutar paquetes sólo hacia el otro departamento de su dirección. Para ello es necesario añadir una ruta utilizando el comando *route.*

   a) Liste la tabla de rutas existente con el comando `route -n`.

   b) Elimine las rutas que no sean necesarias con el comando:
   `route del [-net|-host] <Target>netmask <Netmask>`

   Observe que `-net` indica que `<Target>` es una dirección de red y `-host` que es una dirección de *host*.

   c) Añada las rutas necesarias para enrutar los datagramas al otro departamento de su dirección:
   `route add [-net|-host] <Target>[netmask <Nm>] [gw <Gw>] [[dev] <If>]`

2. Con ayuda de los compañeros del otro departamento de su dirección y mediante *Ethereal* compruebe que le llegan datagramas del otro departamento. ¿Por que cree usted que no le llegan datagramas de vuelta?

3. Hablando con sus compañeros de dirección intenten arreglar el problema anterior.¿Como lo han hecho?

El presidente de Acme SA se jubila y los directores de la dirección técnica y de *marketing* se disputan el puesto. En cada *host* de los respectivos departamentos hay información que compromete su candidatura, por lo que cada director encarga a su departamento la tarea de acceder a los *hosts* de la otra dirección en busca de información comprometedora para el otro director.

1. Con que opción del protocolo IP se puede conseguir lo que pretende su director.

2. Desde una de las máquinas que actuan como host, intente que algún *host* de la otra dirección le responda a un *ping* sin cambiar las tablas de rutas.

3. Con ayuda de los compañeros de la otra dirección monitorice el camino que sigue el datagrama.

Ejercicio 6

Con este ejercicio se pretende analizar cómo afecta el encaminamiento de paquetes en una red a la carga que se produce en los enlaces. Para ello, analizaremos dos topologías de red, de las cuales la primera es claramente ineficiente. El alumno debe diseñar la segunda topología con el objetivo de minimizar el número de nodos intermedios por los que pasa un paquete.

Diseño previo de la topología de red (*ejemplo para telem2-1, telem2-2, telem2-3 y telem2-4*)

La red a diseñar se compone de 4 subredes (192.168.1.0/24 - 192.168.2.0/24 - 192.168.3.0/24 - 192.168.4.0/24). Cada subred estará compuesta de 2 *routers* que se encargarán de encaminar el tráfico que circule por ellos de forma adecuada. En la figura 3.17 se muestra la topología de red a realizar.

En el laboratorio no se disponen de cables cruzados para conectar directamente las distintas tarjetas de red, por lo que se utilizarán dos *hubs* interconectados para permitir el acceso de todos los interfaces de red a todos los demás. Esto implica que físicamente tendremos la red de la figura 3.18. Obsérvese que la topología de red de la figura 3.17 se realizará mediante mecanismos de encaminamiento.

1. Realice el conexionado de la figura 3.18.

2. Configure los interfaces de red, tal y como se especifica en la figura 3.17, mediante el comando:
   `ifconfig <Interface><IPaddress>netmask <Netmask>broadcast <Bcast_address>`

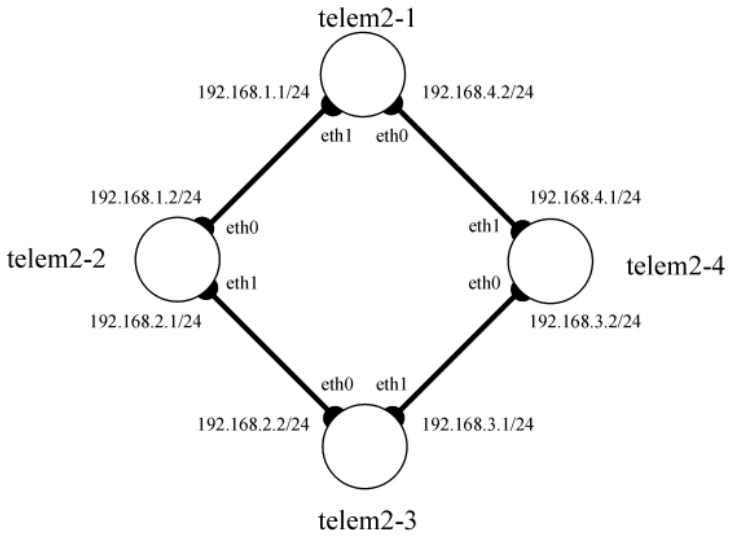


Figura 3.17: Topología de red.

Encaminamiento en anillo

Crearemos a continuación un encaminamiento en anillo, de tal forma que los paquetes circularán sólo en un sentido (telem2-1 - telem2-2 - telem2-3 - telem2-4 - telem2-1...). Para ello, el encaminamiento se realizará siguiendo la tabla 3.6, en la que se indica el *router* al que se envía la información que circula de un *host* a una red remota.

	192.168.1.0/24	**192.168.2.0/24**	**192.168.3.0/24**	**192.168.4.0/24**
telem2-1	X	telem2-2	telem2-2	telem2-2
telem2-2	telem2-3	X	telem2-3	telem2-3
telem2-3	telem2-4	telem2-4	X	telem2-4
telem2-4	telem2-1	telem2-1	telem2-1	X

Tabla 3.6: Tabla de encaminamientos.

Como ejemplo, se muestra la tabla de rutas de telem2-1:

```
Destination     Gateway         Genmask         Flags Metric Ref Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0   0   eth1
0.0.0.0         192.168.1.2     0.0.0.0         UG    0      0   0   eth1
```

Observe que la dirección 0.0.0.0 indica “cualquier otro destino”.

1. Configure correctamente la tabla de rutas de su máquina. Para ello, realice los siguientes pasos:

   a) Liste la tabla de rutas existente con el comando `route -n`.

   b) Elimine las rutas que no sean necesarias con el comando:
   `route del [-net|-host] <Target>netmask <Netmask>`

   Observe que `-net` indica que `<Target>` es una dirección de red y `-host` que es una dirección de *host*.

   c) Añada las rutas necesarias para configurar el encaminamiento en anillo con el comando:
   `route add [-net|-host] <Target>[netmask <Nm>] [gw <Gw>] [[dev] <If>]`

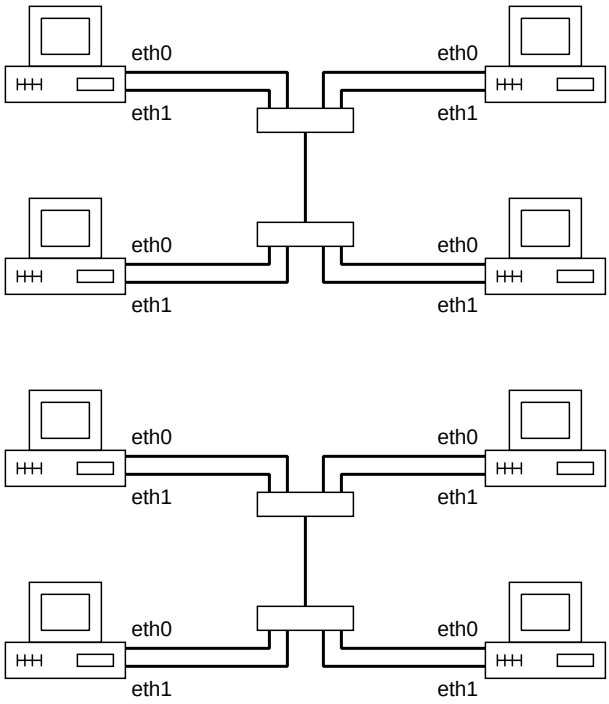


Figura 3.18: Conexión de la red.

2. Ahora es necesario permitir a la máquina que enrute los paquetes que entran por un interfaz hacia otro. Para ello, asegúrese que el archivo `/proc/sys/net/ipv4/ip_forward` contiene un uno (permitir).

3. Póngase de acuerdo con el resto de puestos de trabajo de su anillo y envíen un único mensaje echo (`-c 1`) a cada uno de los interfaces de las otras máquinas para verificar que responden.

4. Ejecute dos analizadores de paquetes `ethereal` para analizar el tráfico por cada interfaz de red. Averigüe el número de protocolo de ICMP en el archivo `/etc/protocols` y utilícelo para capturar sólo paquetes ICMP (no queremos ver ARP u otros protocolos).
   El filtro de captura será el siguiente:
   `ip proto <NumProtoICMP>`

5. Identifique todas las direcciones Ethernet de todos los interfaces de la red.

6. Dispónganse las cuatro máquinas de la red capturando tráfico por todos los interfaces. Envíe un paquete de echo a uno de los interfaces y observe los paquetes que han circulado por la red. Compruebe que se ha realizado correctamente el encaminamiento (la dirección origen Ethernet de los paquetes que se reciben es de un nodo contiguo). ¿Qué conclusiones saca (recuerde que los enlaces son bidireccionales) en cuanto a la utilización de los enlaces?

Encaminamiento óptimo

Cree a continuación una red en la que el tráfico se encamine por el camino más corto (mínimo número de saltos) y esté balanceado por toda la red (no hayan enlaces más cargados que otros). Observe que al ser la topología simétrica, las rutas también lo serán.

1. Rellene la siguiente tabla utilizando el criterio de mínima distancia:

	192.168.1.0/24	**192.168.2.0/24**	**192.168.3.0/24**	**192.168.4.0/24**
telem2-1	X			
telem2-2		X		
telem2-3			X	
telem2-4				X

Tabla 3.7: Tabla de encaminamientos.

2. Escriba y configure la tabla de rutas de su máquina.
3. Repita los pasos 4 y 6 del encaminamiento en anillo.