CAPÍTULO 4

La capa de red: el plano de datos

Hemos estudiado en el capítulo anterior que la capa de transporte proporciona varias formas de comunicación proceso a proceso basándose en el servicio de comunicación host a host de la capa de red. También hemos visto que la capa de transporte lleva a cabo esta tarea sin saber cómo la capa de red implementa realmente este servicio. Así que es posible que se esté preguntando, ¿cuál es el mecanismo subyacente al servicio de comunicación de host a host que lo hace funcionar?

En este capítulo y en el siguiente, vamos a ver exactamente cómo la capa de red puede proporciona su servicio de comunicación host a host. Veremos que, a diferencia de las capas de transporte y de aplicación, *existe un componente de la capa de red en todos y cada uno de los hosts y routers de la red*. Por esta razón, los protocolos de la capa de red se encuentran entre los más desafiantes (¡y, por tanto, entre los más interesantes!) de la pila de protocolos.

La capa de red probablemente sea también la capa más compleja de la pila de protocolos y, por tanto, serán muchas las cuestiones que vamos a tener que abordar. De hecho, son tantos los temas, que dedicaremos dos capítulos a la capa de red. Veremos que esta capa puede descomponerse en dos partes que interaccionan mutuamente, el **plano de datos** y el **plano de control**. En el Capítulo 4, hablaremos primero de las funciones del plano de datos de la capa de red —las funciones *implementadas en cada router* de la capa de red que determinan cómo reenviar a uno de los enlaces de salida de ese router los datagramas (es decir, los paquetes de la capa de red) que llegan a través de alguno de los enlaces de entrada del router. Hablaremos tanto del reenvío IP tradicional (en el que el reenvío se basa en la dirección de destino del datagrama), como del reenvío generalizado (en el que pueden llevarse a cabo el reenvío y otra funciones, dependiendo de los valores contenidos en diversos campos de la cabecera del datagrama). Estudiaremos en detalle los protocolos y el direccionamiento IPv4 e IPv6. En el Capítulo 5 hablaremos de las funciones del plano de control de la capa de red —la lógica *global de la red* que controla el modo en que se enruta un datagrama a lo largo de una serie de routers que componen un trayecto extremo a extremo, desde el host de origen hasta el host de destino. Veremos los algoritmos de enrutamiento, así como los protocolos de enrutamiento de uso más extendido en la Internet actual, como OSPF y BGP. Tradicionalmente, estos protocolos de enrutamiento del plano de control y las funciones de reenvío del plano de datos se han implementado de forma conjunta, monolítica, dentro de un router. La técnica de redes definidas por software (SDN, *Software-Defined Networking*) separa explícitamente el plano de datos y el plano de control, implementando las

funciones del plano de control como un servicio separado, situado típicamente en un "controlador" remoto. También hablaremos de los controladores SDN en el Capítulo 5.

Esta distinción entre las funciones del plano de datos y del plano de control, dentro de la capa de red, es un concepto importante, que debemos tener presente mientras estudiemos la capa de red —nos ayudará a estructurar nuestros conceptos sobre la capa de red y refleja también la visión moderna de cuál es el papel que la capa de red juega en las redes de computadoras.

4.1 Introducción a la capa de red

La Figura 4.1 muestra una red simple formada por dos hosts, H1 y H2, y varios routers en la ruta que va de H1 a H2. Supongamos que H1 está enviando información a H2 y veamos el papel de la capa de red en estos hosts y en los routers intervinientes. La capa de red en H1 toma segmentos de la capa de transporte en H1, encapsula cada segmento en un datagrama y, a continuación, envía los datagramas al router más próximo, R1. En el host de recepción, H2, la capa de red recibe los datagramas de su router más próximo R2, extrae los segmentos de la capa de transporte y los entrega a la capa de transporte de H2. La función principal del plano de datos de cada router consiste en reenviar los datagramas desde sus enlaces de entrada a sus enlaces de salida; la función principal del plano de control de la red consiste en coordinar estas acciones de reenvío locales de cada router individual, de modo que los datagramas terminen transfiriéndose de extremo a extremo, a lo largo de series de routers comprendidos entre los hosts de origen y de destino. Observe que los routers de la Figura 4.1 se ilustran con una pila de protocolos truncada, es decir, sin capas por encima de la capa de red, porque los routers no ejecutan protocolos de la capa de transporte ni de la capa de aplicación como los que hemos examinado en los Capítulos 2 y 3.

4.1.1 Reenvío y enrutamiento: los planos de datos y de control

La función principal de la capa de red es engañosamente simple: transporta paquetes desde un host emisor a un host receptor. En la realización de esta tarea podemos identificar dos importantes funciones de la capa de red:

- *Reenvío (forwarding).* Cuando un paquete llega al enlace de entrada de un router, este tiene que pasar el paquete al enlace de salida apropiado. Por ejemplo, un paquete que llega procedente de H1 al router R1 de la Figura 4.1, debe ser reenviado al siguiente router de alguna ruta que conduzca a H2. Como tendremos oportunidad de ver, el reenvío es solo una de las funciones (¡aunque la más importante y común!) implementadas en el plano de datos. En el caso más general, que veremos en la Sección 4.4, también puede prohibirse a un paquete que salga de un router (por ejemplo, si el paquete tiene su origen en un host emisor del que se sabe que es malicioso, o si el paquete está dirigido a un host de destino prohibido), o bien el paquete puede duplicarse y enviarse a través de múltiples enlaces de salida.
- *Enrutamiento (routing).* La capa de red tiene que determinar la ruta o camino que deben seguir los paquetes a medida que fluyen de un emisor a un receptor. Los algoritmos que calculan estas rutas se conocen como **algoritmos de enrutamiento**. Un algoritmo de enrutamiento determinaría, por ejemplo, la ruta por la que fluirán los paquetes para ir de H1 a H2 en la Figura 4.1. El enrutamiento se implementa en el plano de control de la capa de red.

A menudo, los autores que hablan acerca de la capa de red emplean de forma indistinta los términos *reenvío* y *enrutamiento*. Sin embargo, en este libro utilizaremos estos términos de manera mucho más precisa. El **reenvío** hace referencia a la acción local que realiza un router al transferir un paquete desde una interfaz de un enlace de entrada a una interfaz del enlace de salida apropiado.

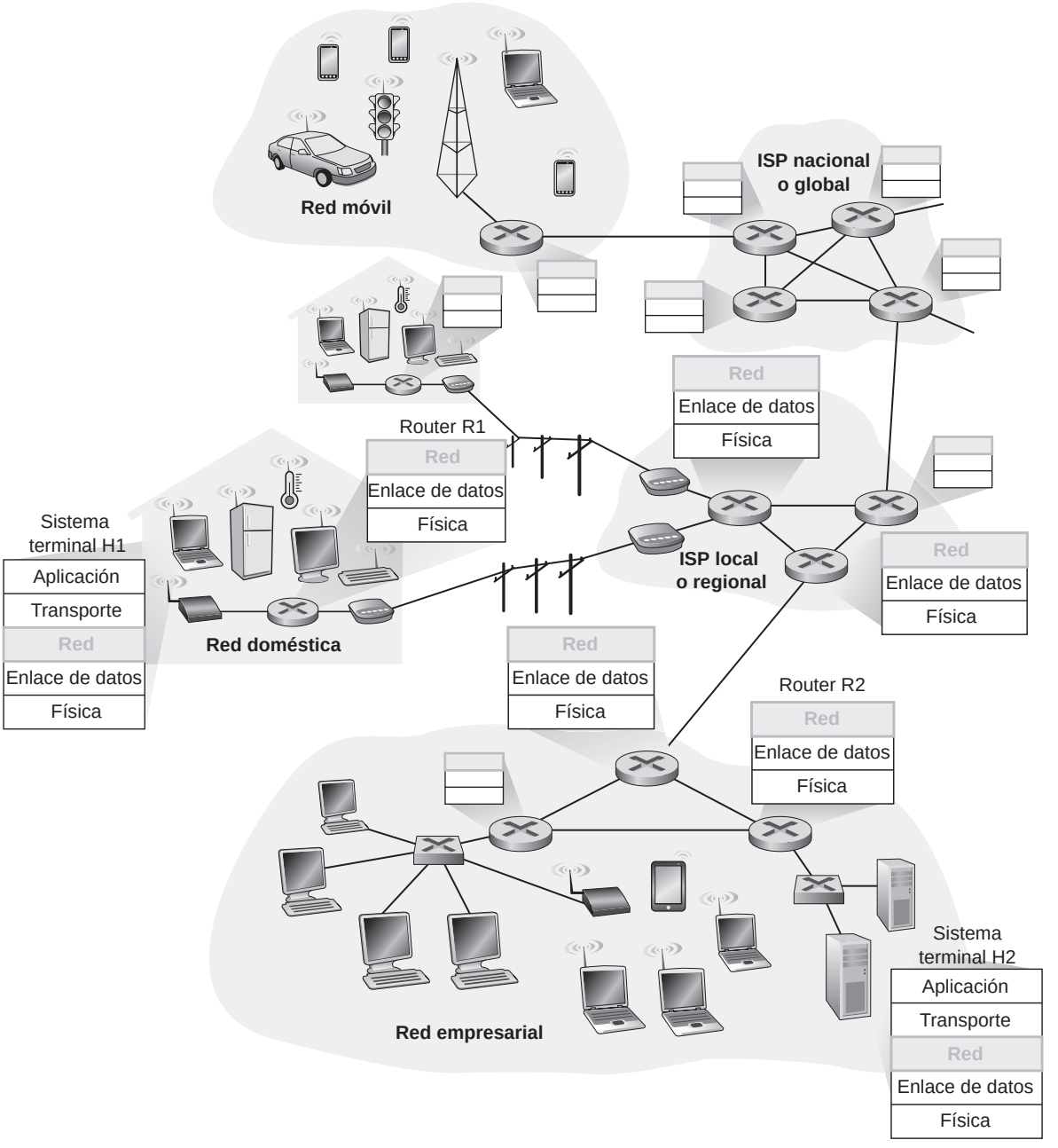


Figura 4.1 ♦ La capa de red.

El reenvío tiene lugar en escalas de tiempo muy cortas (típicamente de unos pocos nanosegundos) y se implementa normalmente en hardware. El **enrutamiento** hace referencia al proceso que realiza la red en conjunto para determinar las rutas extremo a extremo que los paquetes siguen desde el origen al destino. El enrutamiento tiene lugar con escalas de tiempo mucho más largas (normalmente de segundos) y, como veremos, suele implementarse en software. Utilizando nuestra analogía sobre conducir un vehículo, recuerde el trayecto desde Pensilvania a Florida que realizó nuestro viajero de la Sección 1.3.1. Durante ese viaje, nuestro conductor atraviesa muchas intersecciones en su camino a Florida. Podemos pensar en el reenvío como en el proceso de atravesar una intersección: un coche entra en una intersección viniendo por una carretera y determina qué otra carretera tomar para salir de la intersección. Podemos pensar en el enrutamiento como en el proceso de planificación del viaje desde Pensilvania hasta Florida: antes de iniciar el viaje, el conductor consulta un

mapa y elige uno de los muchos posibles caminos, estando cada uno de ellos definido por una serie de tramos de carretera que se conectan en las intersecciones.

Un elemento crucial en todo router de una red es su **tabla de reenvío**. Un router reenvía un paquete examinando el valor de uno o más campos de la cabecera del paquete entrante y utilizando después esos valores de la cabecera para realizar una indexación dentro de su tabla de reenvío. El valor almacenado en la entrada de la tabla de reenvío correspondiente a esos valores indica cuál es la interfaz del enlace de salida del router a la que hay que reenviar el paquete. Por ejemplo, en la Figura 4.2 un paquete con un valor 0110 en el campo de cabecera llega a un router. El router busca en su tabla de reenvío y determina que la interfaz del enlace de salida para este paquete es la interfaz 2. Entonces, el router reenvía internamente el paquete a la interfaz 2. En la Sección 4.2 examinaremos el interior de un router y analizaremos la función de reenvío con mucho más detalle. El reenvío es la funcionalidad clave del plano de datos de la capa de red.

Plano de control: el enfoque tradicional

El lector se estará ahora preguntando cómo se configuran, para empezar, las tablas de reenvío de un router. Esta cuestión es crucial, ya que expone la importante relación existente entre el reenvío (en el plano de datos) y el enrutamiento (en el plano de control). Como se muestra en la Figura 4.2, el algoritmo de enrutamiento determina el contenido de las tablas de reenvío de los routers. En este ejemplo, se ejecuta un algoritmo de enrutamiento en todos y cada uno de los routers, y cada router contiene funciones tanto de reenvío como de enrutamiento. Como veremos en las secciones 5.3 y 5.4, la función encargada del algoritmo de enrutamiento en un router se comunica con la función correspondiente en otros routers para calcular los valores con los que rellenar su tabla de reenvío. ¿Cómo se lleva a cabo esta comunicación? ¡Intercambiando mensajes de enrutamiento, que contienen información de enrutamiento, de acuerdo con lo dispuesto por un protocolo de enrutamiento! Hablaremos de los algoritmos y protocolos de enrutamiento en las secciones 5.2 a 5.4.

La diferencia en los propósitos de las funciones de reenvío y de enrutamiento puede ilustrarse aún más claramente considerando el caso hipotético (y nada realista, pero técnicamente factible) de una red en la que todas las tablas de reenvío fueran configuradas directamente por operadores de red humanos que estuvieran físicamente presentes en los routers. ¡En este caso, *no* se necesitarían

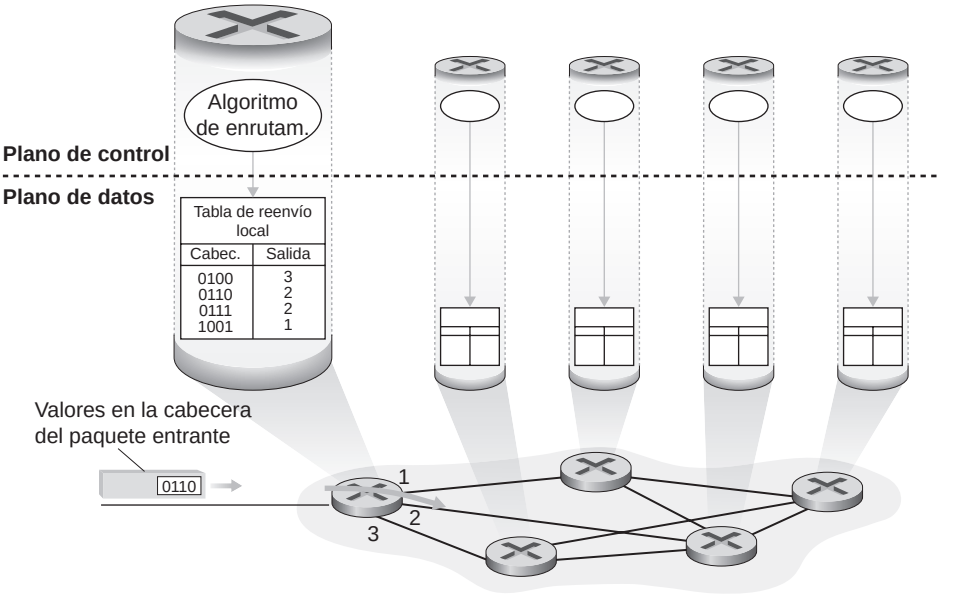


Figura 4.2 ♦ Los algoritmos de enrutamiento determinan los valores almacenados en las tablas de reenvío.

protocolos de enrutamiento! Por supuesto, los operadores humanos tendrían que interactuar entre sí para garantizar que las tablas de reenvío estuvieran configuradas de tal forma que los paquetes llegaran a sus destinos. Probablemente también, si esta configuración la hicieran personas sería más propensa a errores y mucho más lenta en responder a los cambios en la topología de la red que un protocolo de enrutamiento. Por tanto, ¡tenemos suerte de que todas las redes dispongan *tanto* de la función de reenvío *como* de la de enrutamiento!

Plano de control: la técnica SDN

La técnica de implementación de la funcionalidad de enrutamiento mostrada en la Figura 4.2 (en la que cada router tiene un componente de enrutamiento que se comunica con los correspondientes componentes de enrutamiento de otros routers) ha sido la solución tradicional adoptada por los fabricantes de productos de enrutamiento, al menos hasta hace poco. La observación que hacíamos, de que los seres humanos podrían configurar manualmente las tablas de enrutamiento, sugiere, sin embargo, que puede haber otras formas de que las funciones del plano de control determinen el contenido de las tablas de reenvío del plano de datos.

La Figura 4.3 muestra un enfoque alternativo, en el que un controlador remoto, físicamente separado (de los routers), calcula y distribuye las tablas de reenvío que hay que usar en cada router. Observe que los componentes del plano de datos de las Figuras 4.2 y 4.3 son idénticos. En la Figura 4.3, sin embargo, la funcionalidad de enrutamiento del plano de control está separada del router físico —el dispositivo de enrutamiento solo se encarga del reenvío, mientras que el controlador remoto calcula y distribuye las tablas de reenvío—. El controlador remoto puede implementarse en un centro de datos remoto de alta fiabilidad y redundancia, y puede ser gestionado por el ISP o por algún otro proveedor. ¿Cómo pueden comunicarse los routers y el

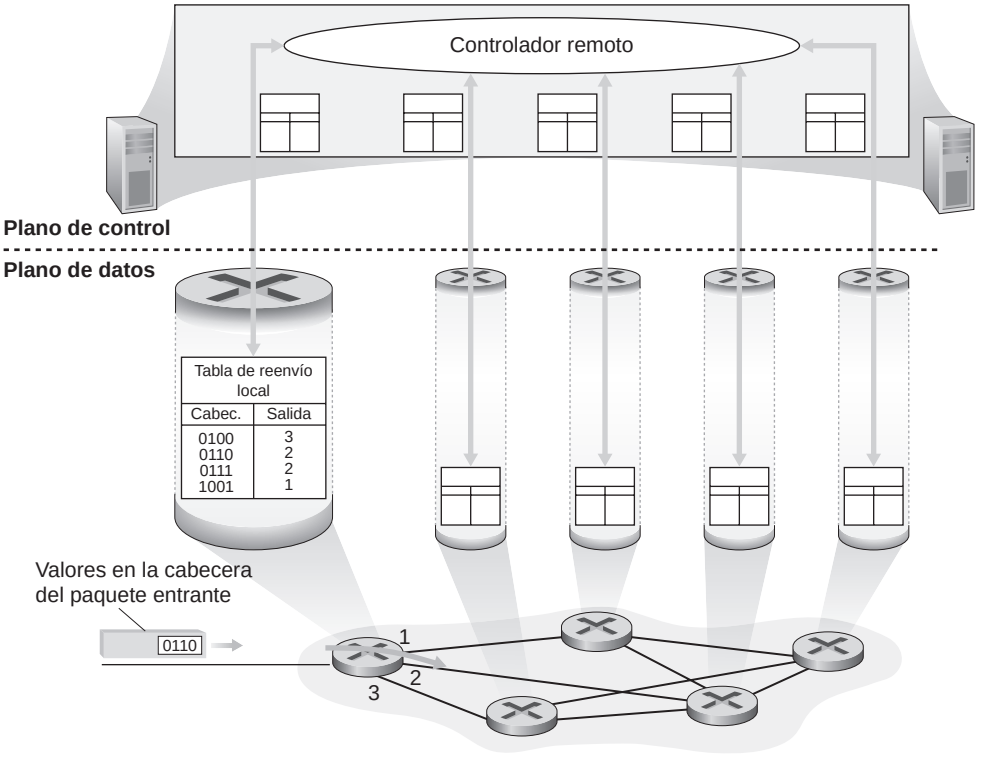


Figura 4.3 ♦ Un controlador remoto determina y distribuye los valores que se almacenan en las tablas de reenvío.

controlador remoto? Intercambiando mensajes que contienen tablas de reenvío y otros tipos de información de enrutamiento. La técnica de implementación del plano de control mostrada en la Figura 4.3 constituye la base de las **redes definidas por software (SDN, *Software-Defined Networking*)**, donde la red está "definida por software" porque el controlador que calcula las tablas de reenvío y que interacciona con los routers está implementado en software. Cada vez más, estas implementaciones software son también abiertas, es decir, de forma similar al código del sistema operativo Linux, el código de esas implementaciones software está públicamente disponible, permitiendo a los ISP (¡y a los investigadores y estudiantes del campo de las redes!) innovar y proponer cambios del software que controla la funcionalidad de la capa de red. Hablaremos del plano de control SDN en la Sección 5.5.

4.1.2 Modelo de servicio de red

Antes de profundizar en la capa de red, terminemos nuestra introducción considerando desde un punto de vista más amplio los distintos tipos de servicio que puede ofrecer esta capa. Cuando la capa de transporte de un host emisor transmite un paquete a la red (es decir, lo pasa a la capa de red del host emisor), ¿puede la capa de transporte confiar en que la capa de red entregue el paquete al destino? Cuando se envían varios paquetes, ¿se entregan a la capa de transporte del host receptor en el orden en que fueron enviados? ¿El intervalo de tiempo entre el envío de transmisiones de dos paquetes secuenciales será el mismo que el intervalo entre sus respectivas recepciones? ¿Realimentará la red información acerca de la congestión de la misma? Las respuestas a estas y otras preguntas están determinadas por el modelo de servicio proporcionado por la capa de red. El **modelo de servicio de red** define las características del transporte de paquetes extremo a extremo entre los hosts emisor y receptor.

Consideremos ahora algunos de los posibles servicios que podría proporcionar la capa de red. Estos servicios podrían incluir:

- *Entrega garantizada*. Este servicio garantiza que un paquete enviado por un host de origen terminará por llegar al host de destino.
- *Entrega garantizada con retardo limitado.* Este servicio no solo garantiza la entrega del paquete, sino que dicha entrega tendrá un límite de retardo especificado de host a host (por ejemplo, de 100 milisegundos).
- *Entrega de los paquetes en orden.* Este servicio garantiza que los paquetes llegan al destino en el orden en que fueron enviados.
- *Ancho de banda mínimo garantizado.* Este servicio de la capa de red emula el comportamiento de un enlace de transmisión con una velocidad de bit específica (por ejemplo, de 1 Mbps) entre los hosts emisor y receptor. Mientras que el host emisor transmita los bits (como parte de los paquetes) a una velocidad inferior a la velocidad de bit especificada, todos los paquetes terminarán por entregarse al host de destino.
- *Seguridad*. La capa de red podría cifrar todos los datagramas en el origen y descifrarlos en el destino, proporcionando así confidencialidad a todos los segmentos de la capa de transporte.

Esta solo es una lista parcial de los servicios que una capa de red podría proporcionar —existen incontables posibles variaciones—.

La capa de red de Internet proporciona un único servicio, conocido con el nombre de **servicio de mejor esfuerzo (*best-effort service*)**. Con un servicio de mejor esfuerzo, no está garantizado que los paquetes se reciban en el orden que fueron emitidos y ni siquiera se garantiza la entrega de los paquetes transmitidos. No hay ninguna garantía sobre el retardo extremo a extremo, ni tampoco se garantiza un ancho de banda mínimo. Podría parecer que lo de *servicio de mejor esfuerzo* es un eufemismo para no decir que no proporciona ningún servicio *en absoluto* —¡una red que no entregara *ningún* paquete a su destino satisfaría la definición de servicio con entrega de mejor esfuerzo!—. Otras arquitecturas de red han definido e implementado modelos de servicio que van

más allá del servicio de mejor esfuerzo de Internet. Por ejemplo, la arquitectura de red ATM [MFA Forum 2016, Black 1995] proporciona un retardo secuencial garantizado, un retardo máximo y un ancho de banda mínimo garantizado. También se han propuesto extensiones al modelo de servicio de la arquitectura Internet; por ejemplo, la arquitectura Intserv [RFC 1633] trata de proporcionar garantías de retardo extremo a extremo y una comunicación libre de congestión. Resulta interesante observar que, a pesar de estas alternativas bien desarrolladas, el modelo básico de servicio de mejor esfuerzo de Internet, combinado con una provisión adecuada de ancho de banda, ha resultado ser más que "suficientemente bueno" como para hacer posible un asombroso rango de aplicaciones, incluyendo servicios de flujos de vídeo como Netflix y aplicaciones de voz y vídeo sobre IP para conferencias en tiempo real, como Skype y Facetime.

Una panorámica del Capítulo 4

Habiendo hecho una somera presentación de la capa de red, en las siguientes secciones de este capítulo hablaremos del plano de datos de dicha capa. En la Sección 4.2 profundizaremos en el funcionamiento del hardware interno de un router, incluyendo el procesamiento de los paquetes entrantes y salientes, el mecanismo interno de conmutación de un router y el tema de las colas y la planificación de paquetes. En la Sección 4.3 examinaremos el reenvío IP tradicional, en el que los paquetes se reenvían a los puertos de salida basándose en sus direcciones IP de destino. En esa sección hablaremos del direccionamiento IP, los famosos protocolos IPv4 e IPv6 y muchas cosas más. En la Sección 4.4 veremos mecanismos de reenvío más generalizados, en los que los paquetes pueden ser reenviados a los puertos de salida basándose en un gran número de valores de la cabecera (es decir, no solo basándose en la dirección IP de destino). Los paquetes pueden ser bloqueados o duplicados en el router, o puede que ciertos campos de su cabecera sean reescritos (todo ello bajo control software). Esta forma más generalizada de reenvío de los paquetes es un componente clave del plano de datos de una red moderna, incluyendo el plano de datos de las redes definidas por software (SDN).

Mencionaremos aquí de pasada que los términos *reenvío* y *conmutación* se utilizan a menudo de forma intercambiable por parte de los investigadores y profesionales de las redes de computadoras; nosotros también usaremos ambos términos de forma intercambiable en este libro. Y continuando con las cuestiones terminológicas, también merece la pena comentar que hay otros dos términos que se emplean a menudo indistintamente, pero que nosotros emplearemos con más cuidado. Reservaremos el término *conmutador de paquetes* para referirnos a un dispositivo de conmutación de paquetes general, que transfiere un paquete desde la interfaz del enlace de entrada a la interfaz del enlace de salida, de acuerdo con los valores almacenados en los campos de la cabecera del paquete. Algunos conmutadores de paquetes, denominados **conmutadores de la capa de enlace** o switches (que se examinan en el Capítulo 6), basan su decisión de reenvío en valores almacenados en los campos de la trama de la capa de enlace; por eso se dice que los switches son dispositivos de la capa de enlace (capa 2). Otros dispositivos de conmutación de paquetes, conocidos como **routers**, basan su decisión de reenvío en los valores almacenados en los campos de la cabecera del datagrama de la capa de red. Los routers son, por tanto, dispositivos de la capa de red (capa 3). (Con el fin de apreciar esta importante distinción, le invitamos a que repase la Sección 1.5.2, en la que se abordaron los datagramas de la capa de red y las tramas de la capa de enlace, junto con sus relaciones). Puesto que este capítulo está dedicado a la capa de red, utilizaremos fundamentalmente el término *router* en lugar de *conmutador de paquetes*.

4.2 El interior de un router

Ahora que ya hemos visto una introducción a los planos de datos y de control de la capa de red, ahora que hemos señalado la importante diferencia entre reenvío y enrutamiento y ahora que hemos presentado los servicios y funciones de la capa de red, vamos a volver nuestra atención a su

función de reenvío: la transferencia real de paquetes desde los enlaces de entrada de un router a los apropiados enlaces de salida.

En la Figura 4.4 se muestra una visión de alto nivel de la arquitectura de un router genérico. Podemos identificar los cuatro componentes siguientes de un router:

- *Puertos de entrada.* Un **puerto de entrada** realiza varias funciones clave. Lleva a cabo la función de la capa física consistente en terminar un enlace físico entrante en un router; esto se muestra mediante el recuadro situado más a la izquierda del puerto de entrada y el recuadro más a la derecha del puerto de salida en la Figura 4.4. Un puerto de entrada también realiza las funciones de la capa de enlace necesarias para interoperar con la capa de enlace en el lado remoto del enlace de entrada; esto se representa mediante los recuadros centrales de los puertos de entrada y de salida. Quizá lo más importante es que también se lleva a cabo una función de búsqueda en el puerto de entrada; esto sucede en el recuadro situado más a la derecha del puerto de entrada. Es aquí donde se consulta la tabla de reenvío, para determinar el puerto de salida del router al que se reenviará un paquete entrante a través del entramado de conmutación. Los paquetes de control (por ejemplo, paquetes que transportan la información del protocolo de enrutamiento) son reenviados desde un puerto de entrada al procesador de enrutamiento. Observe que aquí el término "puerto" —que hace referencia a las interfaces físicas de entrada y salida del router —es claramente distinto de los puertos software asociados con las aplicaciones de red y los sockets, de los que hemos hablado en los Capítulos 2 y 3. En la práctica, el número de puertos soportados por un router puede ir desde un número relativamente pequeño en los routers empresariales, hasta centenares de puertos a 10 Gbps para un router situado en la frontera de un ISP, que es donde el número de líneas entrantes tiende a ser mayor. El router de frontera Juniper MX2020, por ejemplo, soporta hasta 960 puertos Ethernet a 10 Gbps, teniendo el router una capacidad global del sistema de 80 Tbps [Juniper MX 2020 2016].

- *Entramado de conmutación.* El entramado de conmutación conecta los puertos de entrada del router a sus puertos de salida. Este entramado de conmutación está completamente contenido dentro del router: ¡una red dentro de un router de red!
- *Puertos de salida.* Un **puerto de salida** almacena los paquetes recibidos desde el entramado de conmutación y los transmite al enlace de salida, llevando a cabo las funciones necesarias de la capa de enlace y de la capa física. Cuando un enlace es bidireccional (es decir, transporta tráfico en ambas direcciones), el puerto de salida del enlace normalmente estará emparejado con el puerto de entrada de dicho enlace, en la misma tarjeta de línea.
- *Procesador de enrutamiento.* El procesador de enrutamiento lleva a cabo las funciones del plano de control. En los routers tradicionales, ejecuta los protocolos de enrutamiento (que veremos en las secciones 5.3 y 5.4), mantiene las tablas de enrutamiento y la información asociada de estado de los enlaces y calcula la tabla de reenvío para el router. En los routers SDN, el procesador de enrutamiento se encarga de comunicarse con el controlador remoto para (entre otras actividades) recibir entradas de la tabla de reenvío calculadas por el controlador remoto e instalar dichas entradas en los puertos de entrada del router. El procesador de enrutamiento también realiza las funciones de gestión de red que estudiaremos en la Sección 5.7.

Los puertos de entrada, los puertos de salida y el entramado de conmutación de un router se implementan casi siempre en hardware, como se muestra en la Figura 4.4. Para entender por qué hace falta una implementación hardware, piense en que con un enlace de entrada a 10 Gbps y un datagrama IP de 64 bytes, el puerto de entrada solo dispone de 51,2 ns para procesar el datagrama, antes de la posible llegada de otro. Si se combinan N puertos en una tarjeta de línea (como se suele hacer en la práctica), la *pipeline* de procesamiento de datagramas debe funcionar N veces más rápido —demasiada velocidad para una implementación software—. El hardware de reenvío puede implementarse usando los propios diseños hardware del fabricante del router o construirse a partir de chips comerciales de silicio que se adquieran (por ejemplo, como los ofrecidos por empresas como Intel y Broadcom).

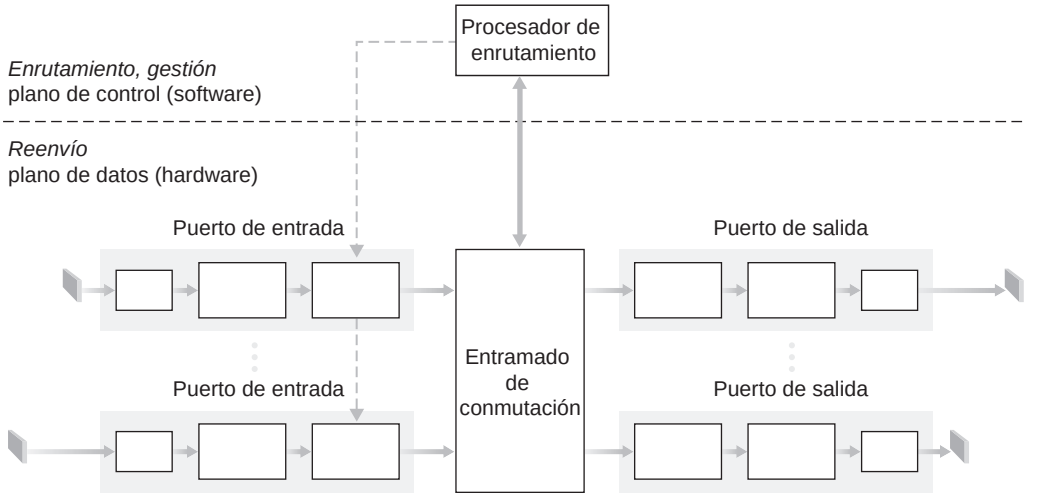


Figura 4.4 ♦ Arquitectura de un router.

Mientras que el plano de datos opera con una escala de tiempo de nanosegundos, las funciones de control de un router (ejecutar los protocolos de enrutamiento, responder a la activación o desactivación de los enlace conectados, comunicarse con el controlador remoto en el caso de SDN y realizar funciones de gestión) operan con una escala de tiempos de milisegundos o incluso de segundos. Por ello, estas funciones del **plano de control** suelen implementarse en software y se ejecutan en el procesador de enrutamiento (normalmente una CPU tradicional).

Antes de entrar en los detalles internos de un router, volvamos a nuestra analogía del principio del capítulo, en la que comparábamos el reenvío de paquetes con los vehículos que entran y salen de una intersección. Supongamos que la intersección es una rotonda y que hace falta un poco de procesamiento cuando un vehículo entra en la rotonda. Veamos qué información hace falta para ese procesamiento:

- *Reenvío basado en el destino (destination-based forwarding)*. Suponga que el vehículo se detiene en una garita a la entrada de la intersección y que indica cuál es su destino final (no en la rotonda, sino el destino último de su viaje). Un operario situado en la garita busca el destino final, determina la salida de la rotonda que conduce a ese destino final y le dice al conductor del vehículo qué salida de la rotonda debe tomar.
- *Reenvío generalizado (generalized forwarding)*. El operario podría determinar también la salida apropiada para el vehículo basándose en otros muchos factores, además del destino. Por ejemplo, la salida podría depender del origen del coche o de la provincia a la que pertenezca su placa de matrícula. Podría ordenarse a los vehículos de una serie de provincias que usaran una determinada salida (que conduce al destino por una carretera más lenta), mientras que a los de otras provincias se les podría indicar que usaran una salida diferente (que conduce al destino a través de una autopista). La misma decisión podría adoptarse basándose en el fabricante, modelo y año de fabricación del vehículo. O podría prohibirse el paso a través de la rotonda a un vehículo que no se considerara apto para circular. En el caso del reenvío generalizado, son muchos los factores que pueden contribuir a la selección de la salida realizada por el operario para un vehículo concreto.

Una vez que el vehículo entra en la rotonda (que puede estar repleta de otros vehículos incorporándose desde otras carreteras de entrada y dirigiéndose hacia otras salidas), terminará por salir a través de la salida prescrita, donde puede encontrarse con otros vehículos que también estén intentando usar esa salida para abandonar la rotonda.

Podemos reconocer fácilmente en esta analogía los cuatro componentes principales de un router mostrados en la Figura 4.4: la carretera de entrada y la garita de entrada se corresponden con el puerto de entrada (con una función de búsqueda para determinar el puerto local de salida); la rotonda se corresponde con el entramado de conmutación y la carretera de salida de la rotonda se corresponde con el puerto de salida. Con esta analogía, resulta instructivo pensar en dónde podrían producirse cuellos de botella. ¿Qué sucede si los vehículos llegan a una velocidad impresionante (¡por ejemplo, porque la rotonda está en Alemania o Italia!), pero el operario de la garita es lento? ¿A qué velocidad debe trabajar el operario para garantizar que no se forme una cola en una carretera de entrada? Incluso con un operario asombrosamente veloz, ¿qué sucede si los vehículos atraviesan la rotonda lentamente? ¿Podrían producirse atascos en ese caso? ¿Y qué sucede si la mayoría de los vehículos que acceden a la rotonda a través de todas las carreteras de entrada quieren tomar la misma salida de la rotonda? ¿Podrían producirse atascos en la salida, o en algún otro lugar? ¿Cómo debería funcionar la rotonda si quisiéramos asignar prioridades a los diferentes vehículos, o si quisiéramos prohibir a ciertos vehículos el acceso a la rotonda? Todas estas preguntas son análogas a ciertas cuestiones críticas a las que se enfrentan los diseñadores de routers y switches.

En las siguientes subsecciones, vamos a examinar con mayor detalle las funciones de un router. [Iyer 2008, Chao 2001; Chuand 2005; Turner 1998; McKeown 1997a; Partridge 1998; Serpanos 2011] proporcionan una explicación de arquitecturas de router específicas. En aras de la concreción y de la simplicidad, asumiremos inicialmente en esta sección que las decisiones de reenvío solo están basadas en la dirección de destino del paquete, en vez de en un conjunto generalizado de campos de la cabecera del paquete. Cubriremos el caso del reenvío más generalizado de paquetes en la Sección 4.4.

4.2.1 Procesamiento en el puerto de entrada y reenvío basado en el destino

En la Figura 4.5 se muestra una visión más detallada del procesamiento de entrada. Como acabamos de ver, la función de terminación de línea y el procesamiento de la capa de enlace en el puerto de entrada implementan las capas física y de enlace de datos para ese enlace de entrada concreto. La búsqueda realizada en el puerto de entrada resulta crucial para el funcionamiento del router —es aquí donde el router usa la tabla de reenvío para determinar el puerto de salida al que será reenviado un paquete entrante a través del entramado de conmutación. La tabla de reenvío es calculada y actualizada por el procesador de enrutamiento (usando un protocolo de enrutamiento para interaccionar con los procesadores de enrutamiento de otros routers de la red) o se recibe desde un controlador SDN remoto. La tabla de reenvío es copiada en las tarjetas de línea desde el procesador de enrutamiento a través de un bus independiente (por ejemplo, un bus PCI), indicado por la línea punteada que va del procesador de enrutamiento a las tarjetas de las líneas de entrada en la Figura 4.4. Estando disponible esa copia de la tabla en cada tarjeta de línea de entrada, las decisiones de reenvío pueden tomarse localmente, en cada puerto de entrada, sin invocar al procesador centralizado de enrutamiento para cada paquete y evitando así la aparición de un cuello de botella en el procesamiento centralizado.

Consideremos ahora el caso "más simple" de que el puerto de salida hacia el que haya que conmutar un paquete entrante esté basado en la dirección de destino del paquete. En el caso de

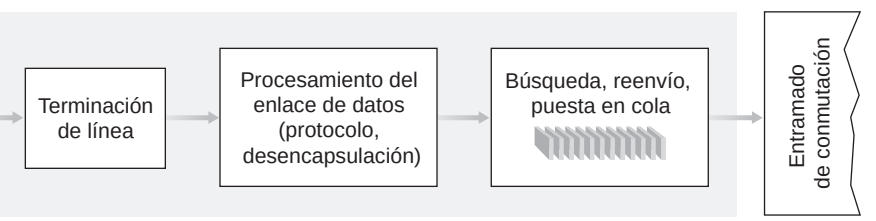


Figura 4.5 ♦ Procesamiento en el puerto de entrada.

direcciones IP de 32 bits, una implementación por fuerza bruta de la tabla de reenvío tendría una entrada para cada una de las posibles direcciones de destino. Dado que existen más de 4.000 millones de posibles direcciones, esta opción está totalmente fuera de lugar.

Como ejemplo de cómo se puede manejar esta cuestión de escala, supongamos que nuestro router tiene cuatro enlaces, numerados de 0 a 3, y que los paquetes deben ser reenviados a las interfaces de enlace como sigue:

Rango de direcciones de destino	Interfaz de enlace
11001000 00010111 00010000 00000000 hasta 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 hasta 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 hasta 11001000 00010111 00011111 11111111	2
En otro caso	3

Evidentemente, para este ejemplo no es necesario disponer de 4.000 millones de entradas en la tabla de reenvío del router. Podríamos, por ejemplo, tener la siguiente tabla de reenvío con solo cuatro entradas:

Prefijo	Interfaz de enlace
11001000 00010111 00010	0
11001000 00010111 00011000	1
11001000 00010111 00011	2
En otro caso	3

Con este tipo de tabla de reenvío, el router busca la coincidencia de un **prefijo** de la dirección de destino del paquete con las entradas de la tabla; si existe una coincidencia, el router reenvía el paquete a un enlace asociado con esa coincidencia. Por ejemplo, suponga que la dirección de destino del paquete es `11001000 00010111 00010110 10100001`; puesto que el prefijo de 21 bits de esta dirección coincide con la primera entrada de la tabla, el router reenvía el paquete a la interfaz de enlace 0. Si un prefijo no se corresponde con ninguna de las tres primeras entradas, entonces el router reenvía el paquete a la interfaz predeterminada, la número 3. Aunque este método puede parecer bastante simple, esconde una importante sutileza. Es posible que se haya dado cuenta de que la dirección de destino puede corresponderse con más de una entrada. Por ejemplo, los primeros 24 bits de la dirección `11001000 00010111 00011000 10101010` coinciden con la segunda entrada de la tabla y los primeros 21 bits con la tercera entrada de la tabla. Cuando existen varias coincidencias, el router aplica la **regla de coincidencia con el prefijo más largo**; es decir, busca la entrada más larga de la tabla con la que exista una coincidencia y reenvía el paquete a la interfaz de enlace asociada con el prefijo más largo. Cuando estudiemos con más detalle el direccionamiento de Internet, en la Sección 4.3, veremos *por qué* exactamente se utiliza esta regla de coincidencia con el prefijo más largo.

Dada la existencia de una tabla de reenvío, la búsqueda resulta conceptualmente simple: la lógica hardware simplemente recorre la tabla de reenvío, buscando la coincidencia con el prefijo más largo. Pero para velocidades de transmisión del orden de los Gigabits, esta búsqueda debe realizarse en nanosegundos (recuerde nuestro ejemplo anterior de un enlace a 10 Gbps y un datagrama IP de 64 bytes). Por tanto, no solo debe realizarse la búsqueda mediante hardware, sino que hacen falta técnicas que vayan más allá de una simple búsqueda lineal en una tabla de gran

tamaño; puede encontrar una revisión de algoritmos rápidos de búsqueda en [Gupta 2001, Ruiz-Sánchez 2001]. También debe prestarse especial atención a los tiempos de acceso a memoria, lo que da como resultado diseños con DRAM integrada en el chip y memorias SRAM más rápidas (usadas como caché DRAM). En la práctica, también se utilizan a menudo para las búsquedas las memorias TCAM (*Ternary Content Addressable Memory*, memoria ternaria direccionable por contenido) [Yu 2004]. Con una TCAM, se presenta una dirección IP de 32 bits a la memoria, que devuelve el contenido de la entrada de la tabla de reenvío correspondiente a esa dirección en un tiempo esencialmente constante. Los routers y switches de las series Cisco Catalyst 6500 y 7600 pueden almacenar más de un millón de entradas TCAM de tabla de reenvío [Cisco TCAM 2014].

Una vez determinado, mediante la búsqueda, el puerto de salida de un paquete, dicho paquete puede ser enviado al entramado de conmutación. En algunos diseños, se puede bloquear temporalmente la entrada del paquete en el entramado de conmutación, si hay paquetes procedentes de otros puertos de entrada que están usando actualmente el entramado. Los paquetes bloqueados serán puestos en cola en el puerto de entrada, planificándose su paso por el entramado para algún instante de tiempo posterior. En breve examinaremos más detalladamente el bloqueo, las colas y la planificación de paquetes (tanto en los puertos de entrada como en los de salida). Aunque la "búsqueda" es, probablemente, la acción más importante dentro del procesamiento en el puerto de entrada, existen muchas otras acciones que hay que llevar a cabo: (1) debe realizarse el procesamiento físico y de la capa de enlace, como hemos comentado anteriormente; (2) hay que comprobar los campos de número de versión, de suma de comprobación y de tiempo de vida del paquete (de todos los cuales hablaremos en la Sección 4.3) y es necesario reescribir estos dos últimos campos y (3) es necesario actualizar los contadores utilizados para la gestión de red (como el del número de datagramas IP recibidos).

Vamos a cerrar nuestra exposición sobre el procesamiento en el puerto de entrada resaltando que los pasos llevados a cabo por el puerto de entrada, en lo referente a buscar una dirección IP de destino ("correspondencia") y luego enviar el paquete a través del entramado de conmutación al puerto de salida especificado ("acción"), constituyen un caso específico de una abstracción "correspondencia más acción" (*match plus action*) más general que se lleva a cabo en muchos dispositivos de red, no solo en los routers. En los switches de la capa de enlace (de los que hablaremos en el Capítulo 6), se buscan las direcciones de destino de la capa de enlace y pueden llevarse a cabo varias acciones, además de enviar la trama hacia el puerto de salida a través del entramado de conmutación. En los cortafuegos (de los que hablaremos en el Capítulo 8 y que son dispositivos que eliminan diversos paquetes entrantes seleccionados), un paquete entrante cuya cabecera se corresponda con un determinado conjuntos de criterios (por ejemplo, una combinación de números de puerto de la capa de transporte y de direcciones IP de origen/destino) puede ser eliminado (acción). En un traductor de direcciones de red (NAT, *Network Address Translator*, de los que hablaremos en la Sección 4.3), un paquete entrante cuyo número de puerto de la capa de transporte se corresponda con un cierto valor, verá su número de puerto reescrito antes del reenvío (acción). De hecho, la abstracción "correspondencia más acción" es potente y prevalente en los dispositivos de red actuales, y resulta clave para la noción de reenvío generalizado de la que hablaremos en la Sección 4.4.

4.2.2 Conmutación

El entramado de conmutación se encuentra en el corazón de todo router, ya que es a través de este entramado donde los paquetes son realmente conmutados (es decir, reenviados) desde un puerto de entrada a un puerto de salida. La conmutación puede llevarse a cabo de varias formas, como se muestra en la Figura 4.6:

- *Conmutación vía memoria.* Los primeros routers, más simples, eran computadoras tradicionales, en las que la conmutación entre los puertos de entrada y de salida se realizaba bajo el control directo de la CPU (procesador de enrutamiento). Los puertos de entrada y de salida funcionaban como dispositivos de E/S tradicionales en un sistema operativo tradicional. El puerto de entrada al que llegaba un paquete enviaba una señal en primer lugar al procesador de enruta-

miento mediante una interrupción. A continuación, el paquete se copiaba desde el puerto entrada en la memoria del procesador. Después, el procesador de enrutamiento extraía la dirección de destino de la cabecera, buscaba en la tabla de reenvío el puerto de salida apropiado y copiaba el paquete en los buffers del puerto de salida. En este escenario, si el ancho de banda de la memoria es tal que pueden escribirse en, o leerse de, la memoria un máximo de *B* paquetes por segundo, entonces la tasa de transferencia global de reenvío (la velocidad total a la que los paquetes se transfieren desde los puertos de entrada a los de salida) tiene que ser menor que *B*/2. Observe también que no pueden reenviarse dos paquetes al mismo tiempo, incluso aunque tengan diferentes puertos de destino, ya que solo puede realizarse una lectura/escritura de memoria cada vez a través del bus compartido del sistema.

Algunos routers modernos también realizan la conmutación vía memoria. Sin embargo, una diferencia importante con los primeros routers es que los procesadores de las tarjetas de línea de entrada se encargan de realizar la búsqueda de la dirección de destino y de almacenar el paquete en la posición de memoria adecuada. De alguna forma, los routers que conmutan vía memoria se parecen mucho a los multiprocesadores de memoria compartida, encargándose los procesadores de las tarjetas de línea de conmutar (escribir) los paquetes en la memoria del puerto de salida apropiado. Los conmutadores de la serie Catalyst 8500 de Cisco [Cisco 8500 2016] conmutan internamente los paquetes mediante una memoria compartida.

- *Conmutación vía bus.* Con esta técnica, el puerto de entrada transfiere directamente un paquete al puerto de salida a través de un bus compartido, sin intervención del procesador de enrutamiento. Esto se suele realizar haciendo que el puerto de entrada añada al paquete como prefijo, antes de transmitir el paquete hacia el bus, una etiqueta interna al conmutador (cabecera), que indica el puerto local de salida al que se está transfiriendo dicho paquete. Todos los puertos de salida reciben el paquete, pero solo el puerto que se corresponda con la etiqueta lo conservará. Después, la etiqueta es eliminada en el puerto de salida, ya que dicha etiqueta solo se usa dentro del conmutador para atravesar el bus. Si llegan múltiples paquetes al router simultáneamente, cada uno a través de un

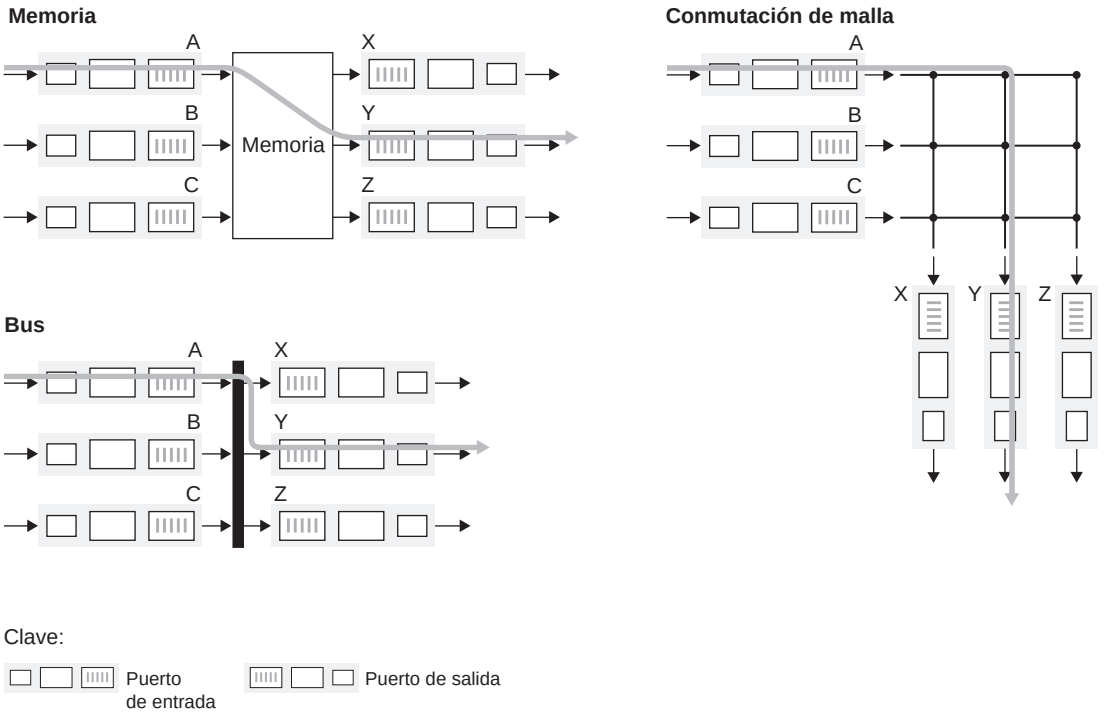


Figura 4.6 ♦ Tres técnicas de conmutación.

puerto de entrada distinto, todos los paquetes menos uno deberán esperar, ya que los paquetes deben atravesar el bus de uno en uno. Puesto que todos los paquetes tienen que atravesar el único bus, la velocidad de conmutación del router está limitada por la velocidad del bus; en nuestra analogía de la rotonda, es como si la rotonda solo pudiera contener un único vehículo cada vez. De todos modos, la conmutación vía bus suele ser suficiente para los routers que operan en redes de área local o redes empresariales de pequeño tamaño. El router Cisco 6500 [Cisco 6500 2016] conmuta internamente los paquetes mediante un bus tipo *backplane* a 32 Gbps.

- *Conmutación vía una red de interconexión.* Una forma de soslayar la limitación del ancho de banda de un único bus compartido, consiste en emplear una red de interconexión más sofisticada, como las que se han empleado en el pasado para interconectar procesadores en una arquitectura de computadora multiprocesador. Un conmutador de malla (*crossbar switch*) es una red de interconexión que consta de 2*N* buses que conectan *N* puertos de entrada a *N* puertos de salida, como se muestra en la Figura 4.6. Cada bus vertical intersecta con cada bus horizontal en un punto de cruce, que puede ser abierto o cerrado en cualquier momento por el controlador del entramado de conmutación (cuya lógica forma parte del propio entramado de conmutación). Cuando un paquete llega a través del puerto A y necesita ser reenviado al puerto Y, el controlador del conmutador cierra el punto de cruce situado en la intersección de los buses A e Y, y el puerto A envía a continuación a través de su bus el paquete, que será transferido (solo) al bus Y. Observe que puede reenviarse al mismo tiempo un paquete del puerto B hacia el puerto X, ya que los paquetes de A a Y y de B a X utilizan diferentes buses de entrada y de salida. Por tanto, a diferencia de las dos técnicas de conmutación anteriores, los conmutadores de malla son capaces de reenviar múltiples paquetes en paralelo. Un conmutador de malla es **no bloqueante**: un paquete que esté siendo reenviado hacia un puerto de salida no se verá bloqueado para alcanzar ese puerto, siempre y cuando no haya ningún otro paquete que esté siendo reenviado actualmente hacia ese puerto de salida. Sin embargo, si dos paquetes de dos puertos de entrada distintos están destinados a un mismo puerto de salida, entonces uno de ellos deberá esperar a la entrada, ya que solo se puede enviar un único paquete en cada momento a través de un bus determinado. Los conmutadores de la familia 12000 de Cisco [Cisco 12000 2016] utilizan una red de conmutación de malla; la familia Cisco 7600 puede configurarse para utilizar un bus o un conmutador de malla [Cisco 7600 2016].

  Otras redes de interconexión más sofisticadas utilizan múltiples etapas de elementos de conmutación para permitir que paquetes procedentes de diferentes puertos de entrada viajen simultáneamente hacia un mismo puerto de salida, a través del entramado de conmutación multietapa. En [Tobagi 1990] podrá encontrar una panorámica de las arquitecturas de conmutación. El sistema Cisco CRS emplea una estrategia de conmutación no bloqueante de tres etapas. La capacidad de conmutación de un router también puede ampliarse utilizando múltiples entramados de conmutación en paralelo. Con esta solución, los puertos de entrada y de salida se conectan a *N* entramados de conmutación que funcionan en paralelo. El puerto de entrada descompone el paquete en *K* segmentos de menor tamaño y envía ("distribuye") los segmentos a través de *K* de esos *N* entramados de conmutación hacia el puerto de salida seleccionado, que vuelve a ensamblar los *K* segmentos para obtener el paquete original.

4.2.3 Procesamiento en el puerto de salida

El procesamiento en los puertos de salida, como se muestra en la Figura 4.7, toma los paquetes que hayan sido almacenados en la memoria del puerto de salida y los transmite a través del enlace de salida. Esto incluye seleccionar los paquetes y extraerlos de la cola para su transmisión y realizar las funciones de transmisión necesarias de las capas de enlace y física.

4.2.4 ¿Dónde se crean colas?

Si nos fijamos en la funcionalidad de los puertos de entrada y de salida, y en las configuraciones mostradas en la Figura 4.6, es evidente que pueden formarse colas de paquetes en los puertos de

entrada *y* en los puertos de salida, de la misma forma que habíamos identificado casos en los que los vehículos podían tener que esperar en las entradas y en las salidas de la intersección de tráfico, en nuestra analogía de la rotonda. La ubicación y la longitud de la colas (en los puertos de entrada o en los de salida) dependerá de la carga de tráfico, de la velocidad relativa del entramado de conmutación y de la velocidad de la línea. Consideremos ahora estas colas con un poco más de detalle, ya que a medida que estas colas crecen, la memoria del router podría terminar agotándose, con lo que se producirá una **pérdida de paquetes** cuando no quede memoria disponible para almacenar los paquetes que lleguen. Recuerde que en nuestras explicaciones anteriores hemos dicho que los paquetes se "perdían dentro de la red" o eran "descartados por un router". *Es aquí, en estas colas de los routers, donde los paquetes realmente se descartan y se pierden*.

Suponga que las velocidades de línea de entrada y de salida (velocidades de transmisión) son todas ellas idénticas e iguales a R_{line} paquetes por segundo, y que existen N puertos de entrada y N puertos de salida. Para simplificar aún más las explicaciones, supongamos que todos los paquetes tienen la misma longitud fija y que los paquetes llegan a los puertos de entrada sincronizadamente. Es decir, suponemos que el tiempo necesario para enviar un paquete a través de cualquier enlace es igual al tiempo necesario para recibir un paquete a través de cualquier enlace, y que durante ese intervalo de tiempo pueden llegar cero o un paquete a través de cada enlace de entrada. Definimos la velocidad de transferencia del entramado de conmutación R_{switch} como la velocidad a la que dicho entramado puede mover los paquetes desde los puertos de entrada hasta los puertos de salida. Si R_{switch} es N veces mayor que R_{line}, entonces las colas que se produzcan en los puertos de entrada serán despreciables. Esto es así porque, incluso en el caso peor, en el que las N líneas de entrada estén recibiendo paquetes y todos esos paquetes deban reenviarse al mismo puerto de salida, el conmutador podrá transferir cada lote de N paquetes (uno por cada puerto de entrada) antes de que llegue el siguiente lote.

Colas de entrada

¿Pero qué sucede si el entramado de conmutación no es lo suficiente rápido (respecto a las velocidades de línea de entrada) como para transferir *todos* los paquetes que le llegan sin producir retardos? En este caso, también pueden crearse colas de paquetes en los puertos de entrada, ya que los paquetes se irán añadiendo a las colas de los puertos de entrada con el fin de esperar su turno para ser transferidos hacia el puerto de salida a través del entramado de conmutación. Para ilustrar una importante consecuencia de estas colas, considere un entramado de conmutación de malla y suponga que (1) todas las velocidades de línea son idénticas, (2) que un paquete puede ser transferido desde cualquier puerto de entrada a un puerto de salida concreto en el mismo intervalo de tiempo que tarda un paquete en ser recibido a través de un enlace de entrada y (3) que los paquetes pasan de una cola de entrada concreta a la cola de salida deseada siguiendo una planificación FCFS. Múltiples paquetes pueden ser transferidos en paralelo, siempre y cuando sus puertos de salida sean diferentes. Sin embargo, si dos paquetes situados en primer lugar de dos colas de entrada están ambos destinados a la misma cola de salida, entonces uno de los paquetes quedará bloqueado y tendrá que esperar en la cola de entrada (el entramado de conmutación solo puede transferir un paquete a un determinado puerto de salida cada vez).

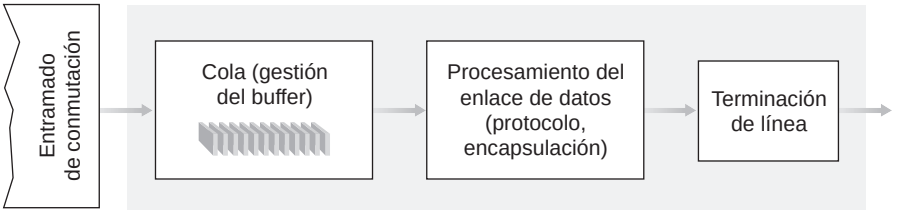


Figura 4.7 ♦ Procesamiento en el puerto de salida.

La Figura 4.8 muestra un ejemplo en el que dos paquetes (los más oscuros) se encuentran al frente de sus respectivas colas de entrada y ambos están destinados al mismo puerto de salida (el de más arriba). Suponga que el entramado de conmutación elige transferir el paquete de la cabecera de la cola superior izquierda. En este caso, el paquete más oscuro de la cola inferior izquierda tendrá que esperar. Pero no solo este paquete más oscuro tiene que esperar, sino también el paquete sombreado más claro que está en la cola detrás del paquete inicial de la cola inferior izquierda, incluso aunque no exista contención para el puerto de salida intermedio (que es el destino del paquete más claro). Este fenómeno se conoce como **bloqueo de cabeza** o **bloqueo HOL** (*Head-of-the-line*, cabeza de línea) en un conmutador con colas de entrada (un paquete que se encuentra en una cola de entrada tiene que esperar a ser transferido a través del entramado de conmutación, aunque su puerto de salida esté libre, porque está bloqueado por otro paquete que se encuentra en la cabeza de la línea). [Karol 1987] demuestra que, a causa del bloqueo HOL, la cola de entrada crecerá de manera ilimitada (informalmente, esto es equivalente a decir que se producirá una pérdida de paquetes significativa) bajo ciertas suposiciones, en cuanto la tasa de llegada de paquetes a través de los enlaces de entrada alcance el 58 por ciento de su capacidad. En [McKeown 1997] se exponen una serie de soluciones para el bloqueo HOL.

Colas de salida

Consideremos a continuación si pueden producirse colas en los puertos de salida de un conmutador. Suponga de nuevo que R_{switch} es N veces mayor que R_{line} y que los paquetes que llegan a cada uno de los N puertos de entrada están destinados al mismo puerto de salida. En este caso, en el tiempo que se tarda en enviar un único paquete a través del enlace saliente, llegarán N nuevos paquetes a este puerto de salida (uno desde cada uno de los N puertos de entrada). Puesto que el puerto de

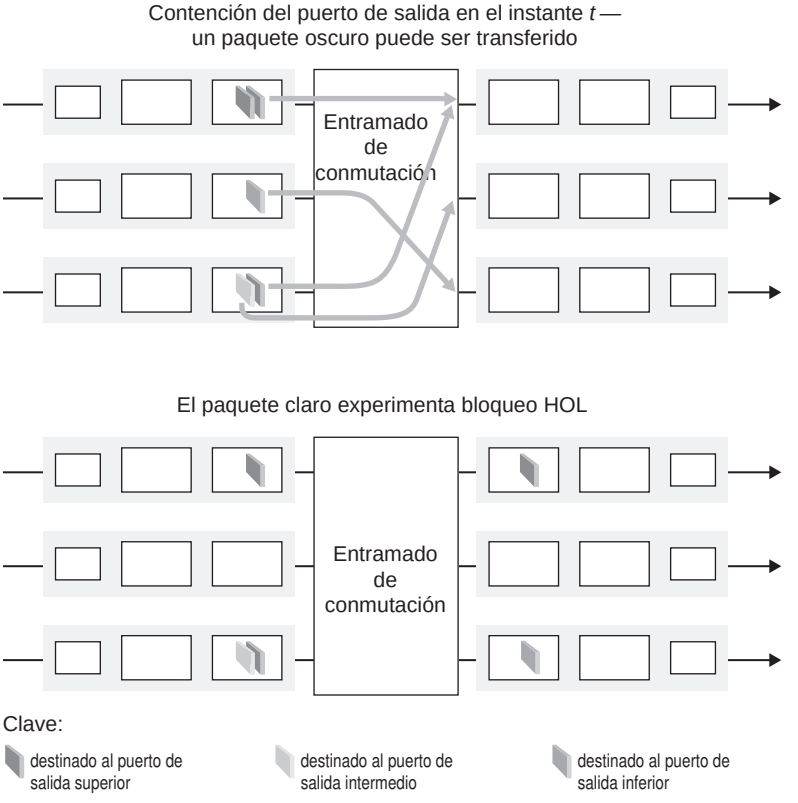


Figura 4.8 ♦ Bloqueo HOL en una cola de entrada de un conmutador.

salida solo puede transmitir un único paquete en cada unidad de tiempo (el tiempo de transmisión del paquete), los *N* paquetes entrantes tendrán que ponerse a la cola (esperar) para poder ser transmitidos a través del enlace saliente. Entonces, *N* paquetes adicionales podrían llegar en el tiempo que se tarda en transmitir solo uno de los *N* paquetes que acababan de ser introducidos en la cola. Y así sucesivamente. Por tanto, pueden formarse colas de paquetes en los puertos de salida incluso aunque el entramado de conmutación sea *N* veces más rápido que la velocidad de línea de los puertos. Podría así llegar a darse el caso de que de que el número de paquetes en cola creciera hasta ser lo suficientemente grande como para agotar la memoria disponible en el puerto de salida.

Cuando no hay suficiente memoria como para introducir en el buffer un paquete entrante, hay que tomar la decisión de eliminar ese paquete entrante (una política conocida con el nombre de **eliminación del último**, o en inglés *drop-tail*) o, por el contrario, eliminar uno o más de los paquetes que ya se encuentran en la cola, para hacer sitio para el paquete recién llegado. En algunos casos, puede resultar conveniente eliminar (o marcar la cabecera de) un paquete antes de que el buffer se llene, para así proporcionar al emisor una indicación de congestión. Se han propuesto y analizado diversas políticas proactivas de eliminación y marcado de paquetes (a las que colectivamente se las ha dado en denominar algoritmos **AQM** —***Active Queue Management,*** gestión activa de colas—) [Labrador 1999, Hollot 2002]. Uno de los algoritmos AQM más ampliamente estudiados e implementados es el algoritmo **RED** (***Random Early Detection,*** **detección aleatoria temprana)** [Christiansen 2001; Floyd 2016].

El fenómeno de las colas en el puerto de salida se ilustra en la Figura 4.9. En el instante *t*, ha llegado un paquete a cada uno de los puertos de entrada, y todos ellos están destinados al puerto de salida situado más arriba. Suponiendo velocidades de línea idénticas y un conmutador operando a tres veces la velocidad de línea, una unidad de tiempo después (es decir, en el tiempo que se necesita para recibir o enviar un paquete) los tres paquetes originales habrán sido transferidos al puerto de salida y estarán en cola esperando a ser transmitidos. En la siguiente unidad de tiempo, uno de estos tres paquetes habrá sido transmitido a través del enlace de salida. En nuestro ejemplo, llegan *dos nuevos* paquetes a la entrada del dispositivo de conmutación y uno de estos paquetes también tiene como destino el puerto de salida de más arriba. Una consecuencia de la aparición de esa cola es que un

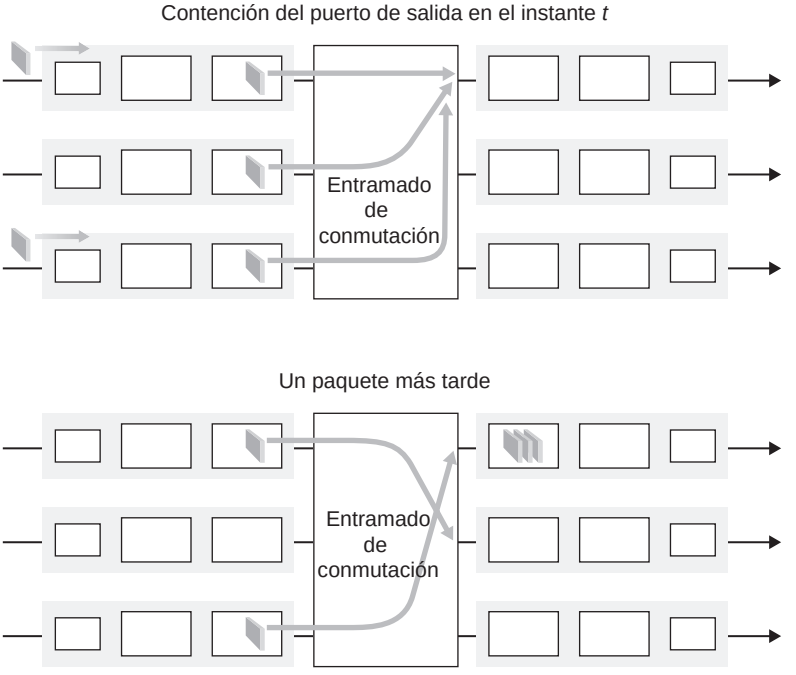


Figura 4.9 ♦ Puesta en cola en el puerto de salida.

planificador de paquetes en el puerto de salida deberá elegir un paquete, de entre todos los de la cola, para su transmisión - un tema del que hablaremos en la próxima sección.

Dado que hacen falta buffers en los routers para absorber las fluctuaciones de la carga de tráfico, es natural plantearse *cuánto* espacio de buffer será necesario. Durante muchos años, la regla heurística [RFC 3439] que se empleaba para determinar el tamaño del buffer era que la cantidad de espacio de buffer (B) debía ser igual al valor promedio del tiempo de ida y vuelta (RTT, digamos 250 milisegundos) multiplicado por la capacidad del enlace (C). Este resultado está basado en el análisis de la dinámica de colas de un número relativamente pequeño de flujos TCP [Villamizar 1994]. Por tanto, un enlace a 10 Gbps con un RTT de 250 milisegundos necesitaría un espacio de buffer igual a $B = RTT \cdot C =$ 2,5 Gbits. Sin embargo, recientes esfuerzos teóricos y experimentales [Appenzeller 2004] sugieren que cuando existe un número grande de flujos TCP (N) atravesando un enlace, la cantidad de espacio en buffer necesaria es $B = RTT \cdot C / \sqrt{N}$. Con una gran cantidad de flujos atravesando normalmente los enlaces de router troncales (véase por ejemplo [Fraleigh 2003]), el valor de N puede ser grande, con lo que la reducción del tamaño de buffer necesario se hace bastante significativa. [Appenzeller 2004; Wischik 2005; Beheshti 2008] proporcionan una serie de estudios comprensibles acerca del problema del tamaño del buffer desde los puntos de vista teórico, de implementación y operacional.

4.2.5 Planificación de paquetes

Volvamos a la cuestión de determinar el orden en el que se transmiten a través de un enlace saliente los paquetes existentes en la cola. Dado que el lector habrá tenido, sin lugar a dudas, que esperar en una larga cola en numerosas ocasiones y habrá observado cómo se presta servicio a los clientes que están esperando, estará familiarizado con muchas de las disciplinas de colas utilizadas en los routers. Existe, por ejemplo, la política de "primero que llega, primero al que se sirve", conocida por las siglas FCFS (*First-Come-First-Served*) o FIFO (*First-In-First-Out*, primero en entrar, primero en salir). Los británicos son famosos por hacer paciente y ordenadamente colas FCFS en las paradas de autobús y en el mercado ("¿Ah, está usted en la cola?"). Otros países utilizan las colas con prioridad, en las que se da prioridad de servicio a una clase de clientes con respecto a otros clientes que están esperando. Existen también las colas de tipo *round-robin* o por turnos, en las que de nuevo se divide a los clientes en clases (como en las colas con prioridad), pero se presta servicio a cada clase de cliente por turno.

FIFO

La Figura 4.10 muestra la abstracción del modelo de cola para la disciplina FIFO de planificación de un enlace. Los paquetes que llegan a la cola de salida del enlace esperan para su transmisión en caso de que el enlace esté actualmente ocupado transmitiendo otro paquete. Si no hay suficiente espacio de buffer para almacenar el paquete entrante, la política de descarte de paquetes de la cola determinará si el paquete es eliminado (perdido) o si se eliminarán otros paquetes de la cola para hacer sitio al paquete entrante, como hemos explicado anteriormente. En las explicaciones que siguen, vamos a ignorar el descarte de paquetes. Cuando se termina de transmitir completamente un

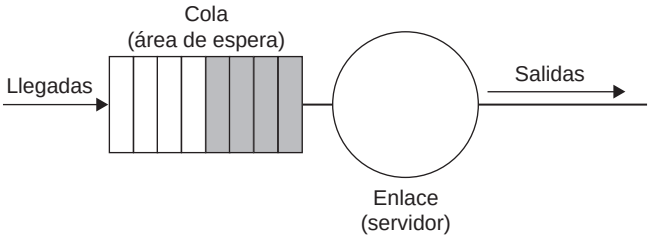


Figura 4.10 ♦ Abstracción de cola FIFO.

paquete a través del enlace saliente (es decir, en cuanto el paquete termina de recibir servicio), se lo elimina de la cola.

La disciplina de planificación FIFO (también denominada FCFS) selecciona los paquetes para su transmisión a través del enlace en el mismo orden en el que llegaron a la cola de salida del enlace. Todos estamos familiarizados con las colas FIFO gracias a nuestra experiencia en cualquier tipo de centros de servicio, donde los clientes que llegan se ponen al final de la única cola existente, permanecen en orden y reciben servicio cuando alcanzan la posición inicial de la línea. La Figura 4.11 ilustra el funcionamiento de la cola FIFO. Las llegadas de paquetes se indican mediante flechas numeradas situadas encima de la línea temporal superior, con el número indicando el orden de llegada del paquete. Las salidas de paquetes individuales se muestran debajo de la línea temporal inferior. El tiempo que un paquete invierte en ser servido (en ser transmitido) se indica mediante el correspondiente rectángulo sombreado situado entre las dos líneas temporales. En estos ejemplos, vamos a asumir que cada paquete tarda tres unidades de tiempo en transmitirse. Bajo la disciplina FIFO, los paquetes salen en el mismo orden en el que entraron. Observe que después de la salida del paquete 4, el enlace permanece inactivo (ya que los paquetes 1 a 4 han sido transmitidos y eliminados de la cola) hasta la llegada del paquete 5.

Colas con prioridad

En el caso de las colas con prioridad, los paquetes que llegan al enlace de salida se clasifican en clases de prioridad al llegar a la cola, como se muestra en la Figura 4.12. En la práctica, un operador de red puede configurar una cola de modo que los paquetes que transporten información de gestión de la red (lo cual se puede saber, por ejemplo, por el número de puerto TCP/UDP de origen o de destino) tengan prioridad sobre el tráfico de usuario; asimismo, los paquetes de voz sobre IP en tiempo real podrían tener prioridad sobre el tráfico que no sea de tiempo real, como los paquetes de correo electrónico SMTP o IMAP. Cada clase de prioridad suele tener su propia cola. Al elegir un paquete para transmitirlo, la disciplina de cola con prioridad transmitirá un paquete de la clase de prioridad más alta que tenga una cola no vacía (es decir, para la que haya paquetes esperando a ser transmitidos). La elección de los paquetes dentro de la misma clase de prioridad suele hacerse por el sistema FIFO.

La Figura 4.13 ilustra el funcionamiento de una cola con prioridad que consta de dos clases de prioridad distintas. Los paquetes 1, 3 y 4 pertenecen a la clase de alta prioridad, mientras que los paquetes 2 y 5 pertenecen a la clase de baja prioridad. El paquete 1 llega y, al encontrar el enlace inactivo, comienza a ser transmitido. Durante la transmisión del paquete 1 llegan los paquetes 2 y 3, que se introducen en las colas de baja prioridad y de alta prioridad, respectivamente. Después de la transmisión del paquete 1, se selecciona para la transmisión el paquete 3 (un paquete de alta prioridad) con preferencia al paquete 2 (el cual, aunque llegó antes, es un paquete de baja prioridad). Al finalizar la transmisión del paquete 3, se comienza a transmitir el paquete 2. El paquete 4

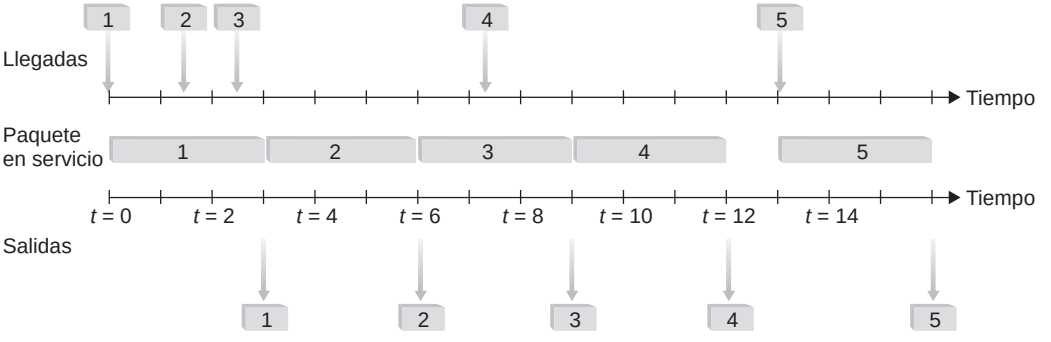


Figura 4.11 ♦ Funcionamiento de la cola FIFO.

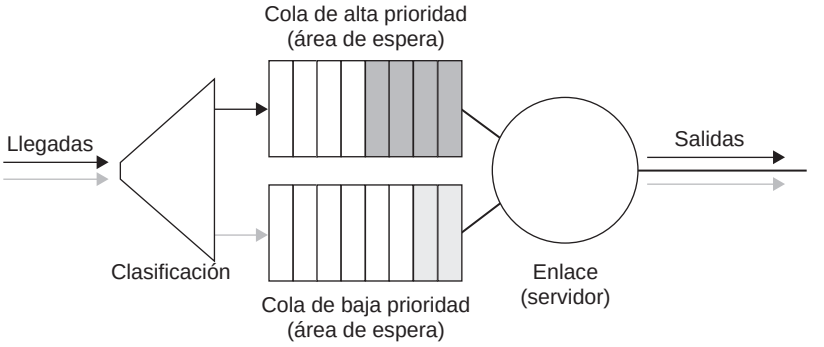


Figura 4.12 ♦ Modelo de cola con prioridad.

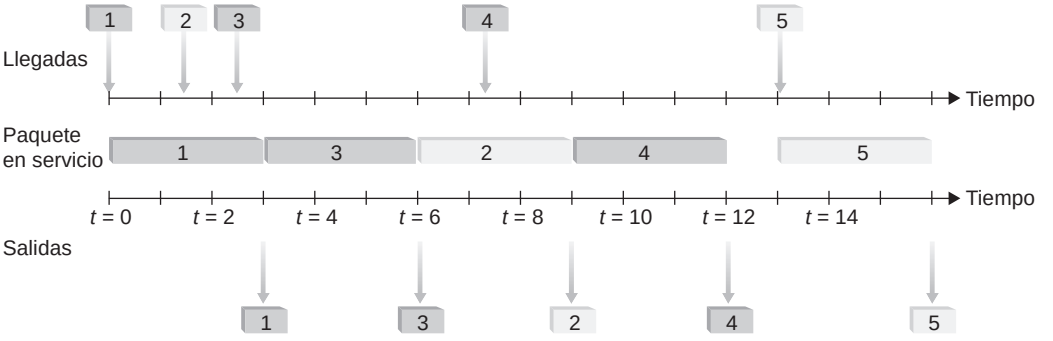


Figura 4.13 ♦ Funcionamiento de la cola con prioridad.

(un paquete de alta prioridad) llega durante la transmisión del paquete 2 (un paquete de baja prioridad). Con la disciplina de **cola con prioridad sin desalojo**, la transmisión de un paquete no se interrumpe una vez que se ha iniciado. En este caso, el paquete 4 se pone en cola de transmisión y comienza a ser transmitido en cuanto se completa la transmisión del paquete 2.

Round robin y colas equitativas ponderadas (WFQ)

Con la disciplina de cola *round robin* (por turnos), los paquetes se distribuyen en clases, igual que en las colas con prioridad. Sin embargo, en lugar de haber una estricta prioridad de servicio entre clases, un planificador *round robin* va alternando el servicio entre las distintas clases. En la forma más simple de planificación *round robin* , se transmite un paquete de clase 1, seguido por un paquete de clase 2, seguido por un paquete de clase 1, seguido de un paquete de clase 2 y así sucesivamente. La denominada disciplina de **cola con conservación del trabajo** no permitirá nunca que el enlace esté inactivo mientras haya paquetes (de cualquier clase) esperando en cola para su transmisión. Una disciplina *round robin* con conservación del trabajo que busque un paquete de una cierta clase y encuentre que no hay ninguno, comprobará inmediatamente la siguiente clase de la secuencia de turnos.

La Figura 4.14 ilustra el funcionamiento de una cola *round robin* con dos clases. En este ejemplo, los paquetes 1, 2 y 4 pertenecen a la clase 1, mientras que los paquetes 3 y 5 pertenecen a la segunda clase. El paquete 1 comienza a ser transmitido inmediatamente después de su llegada a la cola de salida. Los paquetes 2 y 3 llegan durante la transmisión del paquete 1 y, por tanto, se introducen en sus respectivas colas de transmisión. Después de la transmisión del paquete 1, el planificador del enlace busca un paquete de la clase 2 y transmite, por tanto, el paquete 3. Después de la transmisión del paquete 3, el planificador busca un paquete de la clase 1 y transmite, por tanto, el paquete 2.

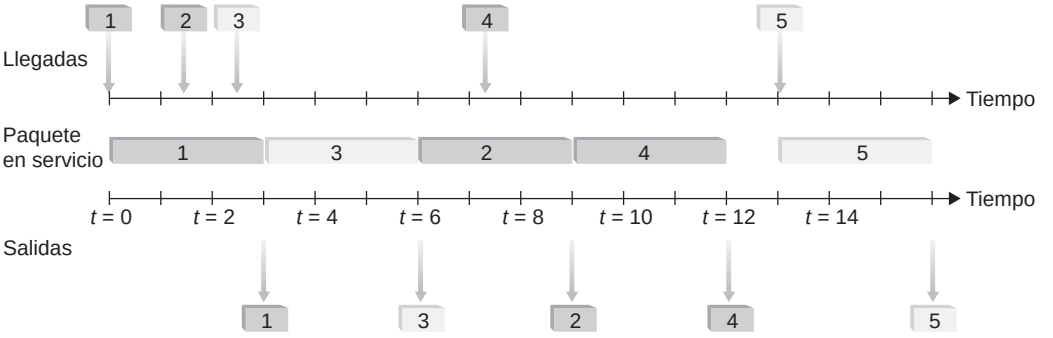


Figura 4.14 ♦ Funcionamiento de una cola round robin con dos clases.

Después de la transmisión del paquete 2, el único paquete que hay en la cola es el paquete 4; por ello, se lo transmite inmediatamente después del paquete 2.

Una forma generalizada de cola *round robin* que se ha implementado ampliamente en routers es la denominada disciplina de **cola equitativa ponderada (WFQ, *Weighted Fair Queueing*)** [Demers 1990; Parekh 1993; Cisco QoS 2016]. La disciplina WFQ se ilustra en la Figura 4.15. Aquí, los paquetes entrantes se clasifican y se ponen en cola en el área de espera específica de la clase apropiada. Como en la planificación *round robin*, un planificador WFQ prestará servicio a las clases de forma circular: sirviendo primero a la clase 1, luego a la clase 2, luego a la clase 3 y luego (suponiendo que solo existan tres clases) repitiendo la secuencia de servicio. WFQ es también una disciplina de cola con conservación del trabajo y, por tanto, se moverá inmediatamente a la siguiente clase de la secuencia de servicio cuando se encuentre con una cola de clase vacía.

WFQ difiere de la disciplina *round robin* en que cada clase puede recibir una cantidad diferente de servicio en cualquier intervalo de tiempo determinado. Específicamente, a cada clase i se le asigna un peso w_i. Con WFQ, durante cualquier intervalo de tiempo en el que haya paquetes de la clase i para enviar, a la clase i se le garantiza que recibirá una fracción del servicio igual a $w_i/(\Sigma\, w_j)$, donde la suma del denominador se realiza para todas las clases que también dispongan de paquetes en cola para su transmisión. En el caso peor, que es cuando todas las clases dispongan de paquetes en cola, a la clase i se le seguirá garantizando que recibirá una fracción $w_i/(\Sigma\, w_j)$ del ancho de banda, donde la suma del denominador en el caso peor se realiza para todas las clases. Por tanto, para un enlace con una velocidad de transmisión R, la clase i siempre conseguirá una tasa de transferencia de al menos $R \cdot w_i/(\Sigma\, w_j)$. Nuestra descripción de WFQ está idealizada, ya que no hemos tenido en cuenta el hecho de que los paquetes son discretos y de que la transmisión de un paquete no se interrumpirá para comenzar la transmisión de otro paquete; [Demers 1990; Parekh 1993] explican esta cuestión del tamaño discreto de los paquetes.

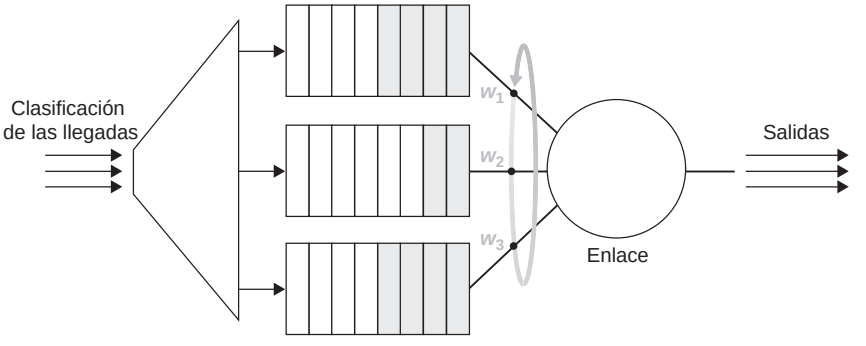


Figura 4.15 ♦ Cola equitativa con prioridad.

4.3 Protocolo de Internet (IP): IPv4, direccionamiento, IPv6 y más

En la exposición que hemos realizado hasta el momento de la capa de red en el Capítulo 4 (la noción de que la capa de red tiene dos componentes, que son el plano de datos y el plano de control; la distinción entre reenvío y enrutamiento; la identificación de diversos modelos de servicio de la red y el examen que hemos hecho del interior de un router), no hemos hecho apenas referencia a ninguna arquitectura de red de computadoras ni protocolo específicos. En esta sección, vamos a fijar nuestra atención en aspectos clave de la capa de red de la actual Internet y en el famoso Protocolo de Internet (IP).

Actualmente hay dos versiones en uso de IP. En primer lugar, examinaremos en la Sección 4.3.1 el muy implantado protocolo IP versión 4, que habitualmente se denomina simplemente IPv4 [RFC 791]. En la Sección 4.3.5, abordaremos la versión 6 de IP [RFC 2460; RFC 4291], protocolo propuesto para sustituir a IPv4. Entre medias, hablaremos principalmente del direccionamiento Internet, un tema que puede parecer árido y demasiado detallado pero que, como veremos, resulta crucial para comprender cómo funciona la capa de red de Internet. ¡Dominar el direccionamiento IP es dominar la propia capa de red de Internet!

4.3.1 Formato de los datagramas IPv4

Recuerde que los paquetes de la capa de red de Internet se denominan *datagramas*. Iniciamos nuestro estudio del protocolo IP con una introducción a la sintaxis y la semántica del datagrama IPv4. Es posible que esté pensando que no puede haber nada más árido que la sintaxis y la semántica de los bits de un paquete. Sin embargo, los datagramas desempeñan un papel central en Internet: todos los estudiantes y profesionales de las redes necesitan comprenderlos y dominarlos. (Y simplemente para ver que puede resultar divertido estudiar las cabeceras de los protocolos, consulte [Pomeranz 2010].) El formato de los datagramas de IPv4 se muestra en la Figura 4.16. Los campos clave de los datagramas de IPv4 son los siguientes:

- *Número de versión.* Estos 4 bits especifican la versión del protocolo IP del datagrama. A partir del número de versión, el router puede determinar cómo interpretar el resto del datagrama IP. Las distintas versiones de IP utilizan distintos formatos de datagrama. El formato de datagrama

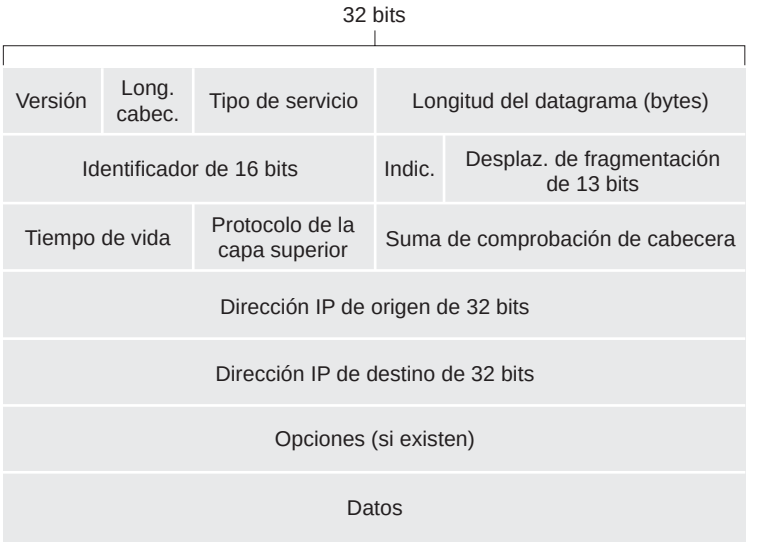


Figura 4.16 ♦ Formato del datagrama IPv4.

para IPv4 es el mostrado en la Figura 4.16. En la Sección 4.3.5 veremos el formato de datagrama correspondiente a la nueva versión de IP (IPv6).

- *Longitud de la cabecera.* Puesto que un datagrama IPv4 puede contener un número variable de opciones (las cuales se incluyen en la cabecera del datagrama IPv4), estos 4 bits son necesarios para determinar dónde comienzan realmente la carga útil (por ejemplo, el segmento de la capa de transporte encapsulado en este datagrama) del datagrama IP. La mayoría de los datagramas IP no contienen opciones, por lo que el datagrama IP típico tiene una cabecera de 20 bytes.
- *Tipo de servicio.* Los bits del tipo de servicio (TOS, *Type Of Service*) se incluyeron en la cabecera de IPv4 con el fin de poder diferenciar entre los distintos tipos de datagramas IP. Por ejemplo, puede resultar útil diferenciar datagramas en tiempo real (como los utilizados en aplicaciones de telefonía IP) del tráfico que no es en tiempo real (como por ejemplo el tráfico FTP). El nivel específico de servicio que se proporcione es una política que determinará y configurará el administrador de red de cada router concreto. También vimos en la Sección 3.7.2 que dos de los bits TOS se utilizan para la notificación explícita de congestión (ECN, *Explicit Congestion Notification*).
- *Longitud del datagrama.* Es la longitud total del datagrama IP (la cabecera más los datos) en bytes. Puesto que este campo tiene una longitud de 16 bits, el tamaño máximo teórico del datagrama IP es de 65.535 bytes. Sin embargo, los datagramas rara vez tienen una longitud mayor de 1.500 bytes, lo que permite que los datagramas IP quepan en el campo de carga útil de una trama Ethernet de tamaño máximo.
- *Identificador, indicadores, desplazamiento de fragmentación.* Estos tres campos tienen que ver con lo que se denomina fragmentación IP, un tema del que hablaremos enseguida. Conviene resaltar que la nueva versión de IP, IPv6, no permite la fragmentación.
- *Tiempo de vida.* El campo Tiempo de vida (TTL, *Time-To-Live*) se incluye con el fin de garantizar que los datagramas no estarán eternamente en circulación a través de la red (debido, por ejemplo, a un bucle de enrutamiento de larga duración). Este campo se decrementa en una unidad cada vez que un router procesa un datagrama. Si el campo TTL alcanza el valor 0, el datagrama tiene que ser descartado por el router.
- *Protocolo.* Este campo solo se suele emplear cuando un datagrama IP alcanza su destino final. El valor de este campo indica el protocolo específico de la capa de transporte al que se pasarán los datos contenidos en ese datagrama IP. Por ejemplo, un valor de 6 indica que los datos se pasan a TCP, mientras que un valor igual a 17 indica que los datos se pasan a UDP. Puede obtener una lista de todos los valores posibles en [IANA Protocol Numbers 2016]. Observe que el número de protocolo especificado en el datagrama IP desempeña un papel análogo al del campo que almacena el número de puerto en un segmento de la capa de transporte. El número de protocolo es el elemento que enlaza las capas de red y de transporte, mientras que el número de puerto es el componente que enlaza las capas de transporte y de aplicación. En el Capítulo 6 veremos que la trama de la capa de enlace también contiene un campo especial que enlaza la capa de enlace con la capa de red.
- *Suma de comprobación de cabecera.* La suma de comprobación de cabecera ayuda a los routers a detectar errores de bit en un datagrama IP recibido. Esta suma de comprobación se calcula tratando cada pareja de 2 bytes de la cabecera como un número y sumando dichos números utilizando aritmética de complemento a 1. Como se ha visto en la Sección 3.3, el complemento a 1 de esta suma, conocido como suma de comprobación Internet, se almacena en el campo Suma de comprobación. Un router calcula la suma de comprobación de cabecera para cada datagrama IP recibido y detecta una condición de error si la suma de comprobación incluida en la cabecera del datagrama no coincide con la suma de comprobación calculada. Normalmente, los routers descartan los datagramas en los que se ha detectado que existe un error. Observe que la suma de comprobación tiene que volver a calcularse y almacenarse en cada router, ya que el campo TTL, y posiblemente también el campo de opciones, pueden cambiar. Una interesante

exposición acerca de algoritmos rápidos para el cálculo de la suma de comprobación Internet puede verse en [RFC 1071]. Una cuestión que suele plantearse en este punto es ¿por qué TCP/IP lleva a cabo una comprobación de errores tanto en la capa de transporte como en la capa de red? Existen varias razones para esta redundancia. En primer lugar, fíjese en que en la capa IP solo se calcula la suma de comprobación para la cabecera IP, mientras que la suma de comprobación TCP/UDP se calcula sobre el segmento TCP/UDP completo. En segundo lugar, TCP/UDP e IP no necesariamente tienen que pertenecer a la misma pila de protocolos. En principio, TCP puede ejecutarse sobre un protocolo diferente de la capa de red (por ejemplo, ATM [Black 1995]) e IP puede transportar datos que no se pasarán a TCP/UDP.

- *Direcciones IP de origen y de destino.* Cuando un origen crea un datagrama, inserta su dirección IP en el campo de dirección IP de origen e inserta la dirección del destino final en el campo de dirección IP de destino. A menudo, el host de origen determina la dirección de destino mediante una búsqueda DNS, como se ha explicado en el Capítulo 2. En la Sección 4.3.3 hablaremos en detalle del direccionamiento IP.
- *Opciones.* El campo de opciones permite ampliar una cabecera IP. La idea original era que las opciones de cabecera rara vez se emplearan: de ahí la decisión de ahorrar recursos no incluyendo la información de los campos opcionales en la cabecera de todos los datagramas. Sin embargo, la mera existencia de opciones complica las cosas, ya que las cabeceras de datagrama pueden tener una longitud variable, por lo que no puede determinarse a priori dónde comenzará el campo de datos. Además, dado que algunos datagramas pueden requerir el procesamiento de opciones y otros no, la cantidad de tiempo necesario para procesar un datagrama IP en un router puede variar enormemente. Estas consideraciones cobran una particular importancia en el procesamiento IP realizado en los hosts y routers de altas prestaciones. Por estas razones y otras, las opciones IP fueron eliminadas en la cabecera de IPv6, como veremos en la Sección 4.3.5.
- *Datos (carga útil).* Finalmente, llegamos al último campo y el más importante: *¡la razón de ser del datagrama!* En la mayoría de las circunstancias, el campo de datos del datagrama IP contiene el segmento de la capa de transporte (TCP o UDP) que va a entregarse al destino. Sin embargo, el campo de datos puede transportar otros tipos de datos, como por ejemplo mensajes ICMP (que veremos en la Sección 5.6).

Observe que un datagrama IP tiene un total de 20 bytes de cabecera (suponiendo que no contiene opciones). Si el datagrama transporta un segmento TCP, entonces cada datagrama (no fragmentado) transporta un total de 40 bytes de cabecera (20 bytes de la cabecera IP más 20 bytes de la cabecera TCP) junto con el mensaje de la capa de aplicación.

4.3.2 Fragmentación del datagrama IPv4

En el Capítulo 6 veremos que no todos los protocolos de la capa de enlace pueden transportar paquetes de la capa de red del mismo tamaño. Algunos protocolos pueden transportar datagramas grandes, mientras que otros solo transportan datagramas pequeños. Por ejemplo, las tramas Ethernet pueden transportar hasta 1.500 bytes de datos, mientras que las tramas para algunos enlaces de área extensa no pueden transportar más de 576 bytes. La cantidad máxima de datos que una trama de la capa de enlace puede transportar se conoce como **unidad máxima de transmisión** (**MTU, *Maximum Transmission Unit***). Puesto que cada datagrama IP se encapsula dentro de una trama de la capa de enlace para ir de un router al siguiente, la MTU del protocolo de la capa de enlace impone un límite estricto a la longitud de un datagrama IP. Esta limitación del tamaño de un datagrama IP no supone un problema importante. Lo que realmente es un problema es que cada uno de los enlaces existentes a lo largo de la ruta entre el emisor y el destino pueden utilizar diferentes protocolos de la capa de enlace y cada uno de estos protocolos puede emplear una MTU diferente.

Con el fin de comprender mejor la cuestión del reenvío, imagine que *usted* es un router que interconecta varios enlaces, que ejecutan distintos protocolos de la capa de enlace con MTU diferentes. Suponga que recibe un datagrama IP procedente de un enlace. Comprueba su tabla de reenvío para

determinar el enlace de salida y ese enlace de salida tiene una MTU que es menor que la longitud del datagrama IP. Lo que nos lleva a plantearnos la pregunta de cómo meter ese datagrama IP sobredimensionado en el campo de carga útil de la trama de la capa de enlace. La solución consiste en fragmentar la carga útil del datagrama IP en dos o más datagramas IP más pequeños, encapsular cada uno de los datagramas IP más pequeños en una trama de la capa de enlace distinta y enviar dichas tramas a través del enlace de salida. Cada uno de estos datagramas más pequeños se conocen como **fragmentos**.

Los fragmentos tienen que ser reensamblados antes de llegar a la capa de transporte del destino. De hecho, tanto TCP como UDP están esperando recibir de la capa de red segmentos completos, no fragmentos. Los diseñadores de IPv4 pensaron que reensamblar los datagramas en los routers añadiría una complejidad significativa al protocolo y reduciría el rendimiento de los routers. (Si usted fuera un router, ¿querría reensamblar fragmentos, además de todo lo que ya tiene que hacer?) Siguiendo el principio de mantener el núcleo de la red simple, los diseñadores de IPv4 decidieron asignar el trabajo de reensamblar los datagramas a los sistemas terminales, en lugar de a los routers de red.

Cuando un host de destino recibe una serie de datagramas procedentes del mismo origen, tiene que determinar si algunos de esos datagramas son fragmentos de algún otro datagrama original más grande. Si algunos datagramas son fragmentos, tiene que determinar además cuándo ha recibido el último fragmento y cómo debe ensamblar los fragmentos que ha recibido para formar el datagrama original. Para que el host de destino pueda llevar a cabo estas tareas de reensamblado, los diseñadores de IP (versión 4) incluyeron los campos *identificación*, *indicadores* y *desplazamiento de fragmentación* en la cabecera del datagrama IP. Cuando se crea un datagrama, el host emisor marca el datagrama con un número de identificación, así como con las direcciones de origen y de destino. Normalmente, el host emisor incrementa el número de identificación para cada datagrama que envía. Cuando un router necesita fragmentar un datagrama, cada datagrama resultante (es decir, cada fragmento) se marca con la dirección de origen, la dirección de destino y el número de identificación del datagrama original. Cuando el destino recibe una serie de datagramas procedentes del mismo host emisor, puede examinar los números de identificación de los datagramas para determinar cuáles de ellos son fragmentos de un mismo datagrama más largo. Puesto que IP es un servicio no fiable, es posible que uno o más de los fragmentos nunca lleguen a su destino. Por esta razón, con el fin de que el host de destino esté absolutamente seguro de que ha recibido el último fragmento del datagrama original, ese último fragmento tiene un bit indicador puesto a 0, mientras que los demás fragmentos tienen el bit indicador puesto a 1. Además, para que el host de destino determine si falta un fragmento (y también para que pueda reensamblar los fragmentos en el orden apropiado), se utiliza el campo desplazamiento para especificar en qué posición dentro del datagrama IP original encaja el fragmento.

La Figura 4.17 proporciona un ejemplo como ilustración. Un datagrama de 4.000 bytes (20 bytes de cabecera IP, más 3.980 bytes de carga útil IP) llega a un router y tiene que ser reenviado a un enlace con una MTU de 1.500 bytes. Esto implica que los 3.980 bytes de datos del datagrama original tienen que ser alojados en tres fragmentos distintos (cada uno de los cuales es también un datagrama IP).

El material en línea del libro y los problemas del final de este capítulo le permitirán explorar con más detalle la cuestión de la fragmentación. Asimismo, en el sitio web del libro proporcionamos un applet de Java que genera fragmentos. No tiene más que proporcionar el tamaño del datagrama de entrada, la MTU y la identificación del datagrama de entrada, y el applet generará automáticamente los fragmentos. Visite el sitio http://www.pearsonhighered.com/cs-resources/.

4.3.3 Direccionamiento IPv4

Ahora vamos a ocuparnos del direccionamiento IPv4. Aunque puede que piense que el direccionamiento es un tema sencillo, al terminar esta sección probablemente se haya convencido de que el direccionamiento en Internet no solo es un tema interesante, profundo y sutil, sino que también tiene

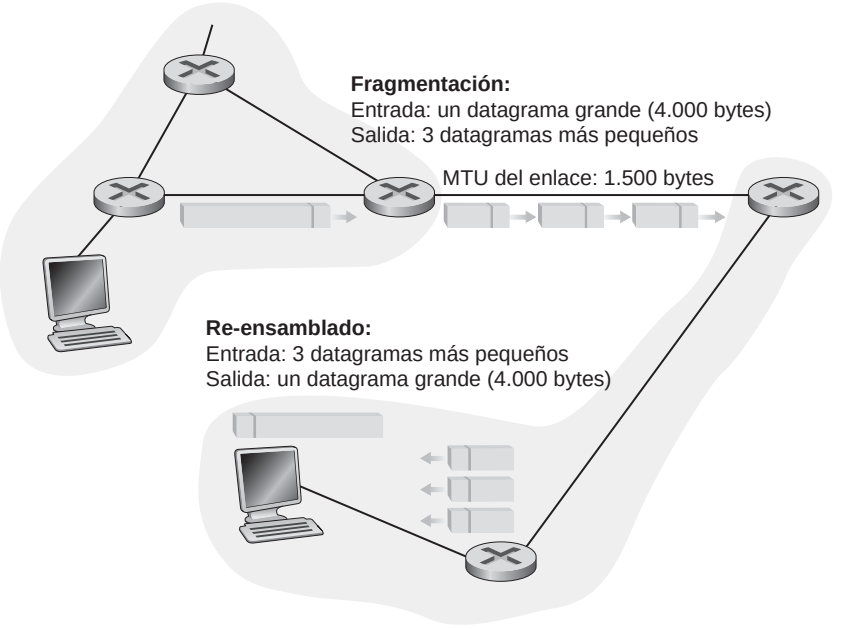


Figura 4.17 ♦ Fragmentación y reensamblado IP.

una importancia crucial para Internet. El primer capítulo de [Stewart 1999] proporciona un excelente tratamiento del direccionamiento IPv4.

Sin embargo, antes de abordar el direccionamiento IP, necesitamos dedicar unas pocas palabras a cómo se conectan los hosts y los routers a la red. Normalmente, un host dispone de un único enlace hacia la red; cuando el protocolo IP del host desea enviar un datagrama, lo hace a través de este enlace. El límite entre el host y el enlace físico se denomina **interfaz**. Consideremos a continuación un router y sus interfaces. Puesto que la tarea de un router consiste en recibir un datagrama por un enlace y reenviarlo a algún otro enlace, un router necesariamente está conectado a dos o más enlaces. El límite entre el router y cualquiera de sus enlaces también se conoce como interfaz. Por tanto, un router tiene varias interfaces, una para cada uno de los enlaces. Puesto que todos los hosts y todos los routers son capaces de enviar y recibir datagramas IP, el protocolo IP requiere que cada interfaz de host y de router tenga su propia dirección IP. *Por tanto, técnicamente, una dirección IP está asociada con una interfaz, en lugar de con el host o con el router que contiene dicha interfaz.*

Las direcciones IP tienen una longitud de 32 bits (lo que equivale a 4 bytes), por lo que existen un total de 2^{32} (unos 4.000 millones) direcciones IP posibles. Estas direcciones normalmente se expresan utilizando la denominada **notación decimal con puntos**, en la que cada byte de la dirección se escribe en formato decimal y se separa mediante un punto del resto de los bytes de la dirección. Por ejemplo, considere la dirección IP 193.32.216.9. El 193 es el número decimal equivalente a los 8 primeros bits de la dirección; el 32 es el equivalente decimal de los segundos 8 bits de la dirección, y así sucesivamente. Por tanto, la dirección 193.32.216.9 en notación binaria se expresa como sigue:

11000001 00100000 11011000 00001001

Cada una de las interfaces de un host o de un router de Internet tiene que tener asociada una dirección IP que sea globalmente única (excepto en el caso de las interfaces utilizadas para NAT, que veremos en la Sección 4.3.4). No obstante, estas direcciones no se pueden elegir a tontas y a locas. Una parte de la dirección IP de una interfaz estará determinada por la subred a la que la interfaz esté conectada.

La Figura 4.18 proporciona un ejemplo de las interfaces y del direccionamiento IP. En esta figura, se utiliza un router (con tres interfaces) para interconectar siete hosts. Echemos un vistazo a

las direcciones IP asignadas a las interfaces de los hosts y del router; hay varios puntos que merece la pena destacar. Los tres hosts de la parte superior izquierda de la Figura 4.18 y la interfaz del router a la que están conectados, tienen todos ellos una dirección IP con el formato 223.1.1.xxx. Es decir, los 24 bits más a la izquierda de la dirección IP de todos ellos son iguales. Además, las cuatro interfaces están interconectadas mediante una red *que no contiene routers*. Esta red podría estar interconectada mediante una LAN Ethernet, en cuyo caso las interfaces se interconectarían mediante un switch Ethernet (como veremos en el Capítulo 6) o mediante un punto de acceso inalámbrico (como veremos en el Capítulo 7). Por el momento, representaremos mediante una nube esa red sin routers que interconecta a esos hosts, y en los Capítulos 6 y 7 profundizaremos en las interioridades de ese tipo de redes.

En términos de IP, esta red que interconecta tres interfaces de host y una interfaz de router forma una **subred** [RFC 950]. (Una subred también se conoce como *red IP* o simplemente *red* en la literatura dedicada a Internet.) El direccionamiento IP asigna una dirección a esta subred: 223.1.1.0/24, donde la notación /24, que en ocasiones se denomina **máscara de subred**, indica que los 24 bits más a la izquierda de ese número de 32 bits definen la dirección de subred. Por tanto, la subred 223.1.1.0/24 consta de tres interfaces de host (223.1.1.1, 223.1.1.2 y 223.1.1.3) y de una interfaz del router (223.1.1.4). Cualquier host adicional conectado a la subred 223.1.1.0/24 *requeriría* una dirección de la forma 223.1.1.xxx. En la Figura 4.18 se muestran otras dos subredes adicionales: la subred 223.1.2.0/24 y la subred 223.1.3.0/24. La Figura 4.19 ilustra las tres subredes IP presentes en la Figura 4.18.

La definición IP de una subred no está restringida a los segmentos Ethernet que conectan varios hosts a una interfaz de un router. Para profundizar un poco más en esta cuestión, considere la Figura 4.20, que muestra tres routers interconectados entre sí mediante enlaces punto a punto. Cada router tiene tres interfaces, una para cada enlace punto a punto y una para el enlace de difusión que conecta directamente el router a una pareja de hosts. ¿Qué subredes hay presentes aquí? Tres subredes, 223.1.1.0/24, 223.1.2.0/24 y 223.1.3.0/24, son similares a las subredes de la Figura 4.18. Pero fíjese en que, en este ejemplo, también existen tres subredes adicionales: una subred, 223.1.9.0/24, para las interfaces que conectan los routers R1 y R2; otra subred, 223.1.8.0/24, para las interfaces que conectan los routers R2 y R3; y una tercera subred, 223.1.7.0/24, para las interfaces que conectan los routers R3 y R1. En un sistema interconectado general de routers y hosts, podemos utilizar la siguiente receta para definir las subredes existentes en el sistema:

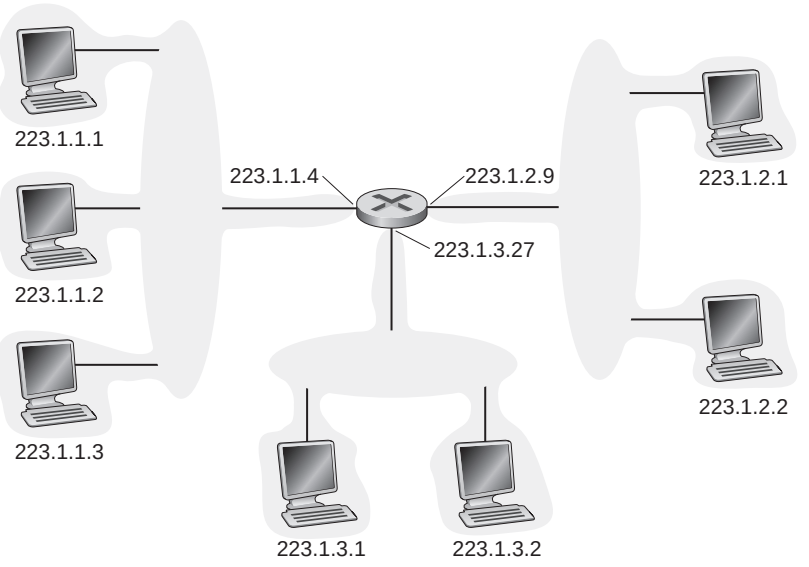


Figura 4.18 ♦ Subredes y direcciones de las interfaces.

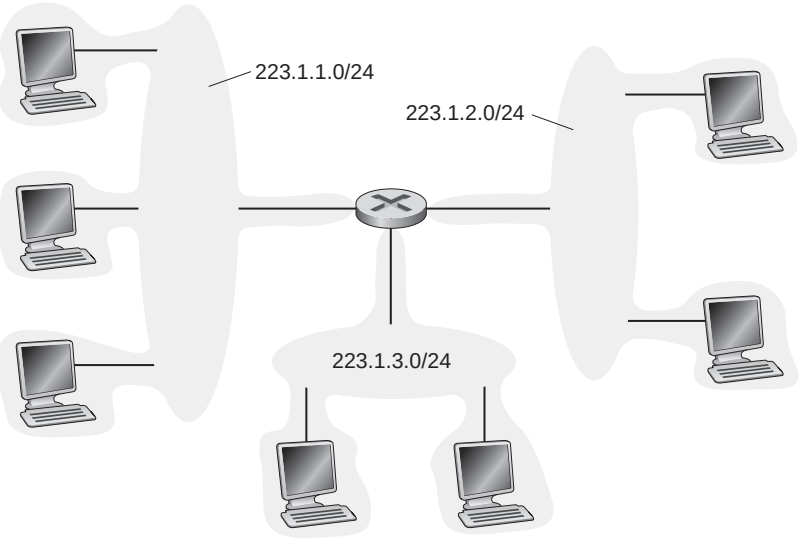


Figura 4.19 ♦ Direcciones de subred.

Para determinar las subredes, desconecte cada interfaz de su host o router, creando islas de redes aisladas, en las que las interfaces actúan como puntos terminales de las redes aisladas. Cada una de estas redes aisladas se dice que es una subred*.*

Si aplicamos este procedimiento al sistema interconectado de la Figura 4.20, obtenemos seis islas o subredes.

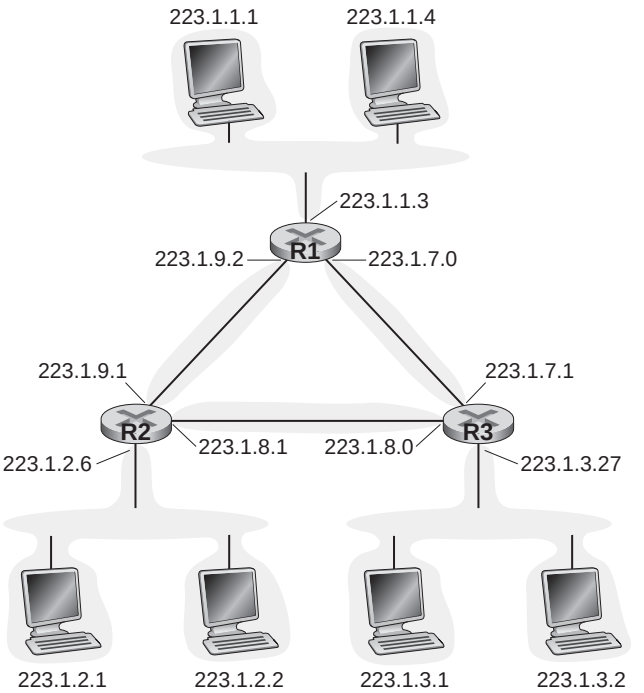


Figura 4.20 ♦ Tres routers que interconectan seis subredes.

A partir de la exposición anterior, está claro que una organización (como por ejemplo una empresa o una institución académica) con múltiples segmentos Ethernet y enlaces punto a punto tendrá varias subredes, teniendo todos los dispositivos de una subred dada la misma dirección de subred. En principio, las distintas subredes podrían tener direcciones de subred bastante diferentes. Sin embargo, en la práctica, sus direcciones de subred a menudo tienen mucho en común. Para entender por qué, veamos cómo se gestiona el direccionamiento en la Internet global.

La estrategia de asignación de direcciones en Internet se conoce como **enrutamiento entre dominios sin clase** (**CIDR, *Classless Interdomain Routing*)** [RFC 4632]. CIDR generaliza la noción de direccionamiento de subred. Al igual que sucede con el direccionamiento de subredes, la dirección IP de 32 bits se divide en dos partes y de nuevo se expresa en notación decimal con puntos como *a.b.c.d/x*, donde *x* indica el número de bits de la primera parte de la dirección.

Los *x* bits más significativos de una dirección en el formato *a.b.c.d/x* constituyen la parte de red de la dirección IP y a menudo se los denomina **prefijo** (o *prefijo de red*) de la dirección. Normalmente, una organización tiene asignado un bloque de direcciones contiguas; es decir, un rango de direcciones con un prefijo común (véase el recuadro En la práctica adjunto). En este caso, las direcciones IP de todos los dispositivos de la organización compartirán el mismo prefijo. Cuando estudiemos el protocolo de enrutamiento BGP de Internet en la Sección 5.4, veremos que los routers externos a la red de la organización solo tienen en cuenta estos *x* primeros bits de prefijo. Es decir, cuando un router externo a la organización reenvía un datagrama cuya dirección de destino está dentro de la organización, únicamente necesita tener en cuenta los primeros *x* bits de la dirección. Esto reduce considerablemente el tamaño de la tabla de reenvío de los routers, ya que una única entrada con el formato *a.b.c.d/x* bastará para reenviar paquetes a cualquier destino dentro de la organización.

Los 32-*x* bits restantes de una dirección pueden emplearse para diferenciar los dispositivos *dentro* de la organización, todos los cuales tienen el mismo prefijo de red. Estos son los bits que habrá que considerar para reenviar paquetes en los routers dentro de la organización. Estos bits de menor peso pueden tener (o no) una estructura en subred adicional, como la que hemos visto anteriormente. Por ejemplo, suponga que los 21 primeros bits de la dirección CIDR a.b.c.d/21 especifican el prefijo de red de la organización y son comunes a las direcciones IP de todos los dispositivos de dicha organización. Los restantes 11 bits identifican entonces a los hosts específicos de la organización. La estructura interna de la organización puede ser tal que estos 11 bits de más a la derecha se empleen para dividir en subredes la organización, como hemos visto anteriormente. Por ejemplo, a.b.c.d/24 podría hacer referencia a una subred específica dentro de la organización.

Antes de que se adoptara el enrutamiento CIDR, la parte de red de una dirección IP estaba restringida a longitudes de 8, 16 o 24 bits, un esquema de direccionamiento conocido como **direccionamiento con clases**, ya que las subredes con direcciones de 8, 16 y 24 bits se conocían, respectivamente, como redes de clase A, B y C. El requisito de que la parte de subred de una dirección IP tuviera exactamente una longitud de 1, 2 o 3 bytes se volvió problemático a la hora de dar soporte al rápido crecimiento del número de organizaciones con subredes de tamaño medio y pequeño. Una subred de clase C (/24) solo puede acomodar hasta $2^8 - 2 = 254$ hosts (dos de las $2^8 = 256$ direcciones están reservadas para usos especiales), que son muy pocos hosts para muchas organizaciones. Sin embargo, una subred de clase B (/16), que puede dar soporte a 65.634 hosts, era demasiado grande. Con el direccionamiento con clases, a una organización con, por ejemplo, 2.000 hosts, se le asignaba normalmente una dirección de subred de clase B (/16). Esto llevó a un rápido agotamiento del espacio de direcciones de clase B y a una poco eficiente utilización del espacio de direcciones asignado. Por ejemplo, la organización que empleaba una dirección de clase B para sus 2.000 hosts tenía asignado espacio suficiente para hasta 65.534 interfaces, dejando bloqueadas más de 63.000 direcciones, que no podían ser utilizadas por otras organizaciones.

Seríamos negligentes si no mencionáramos que existe otro tipo de dirección IP, la dirección IP de difusión 255.255.255.255. Cuando un host envía un datagrama cuya dirección de destino es 255.255.255.255, el mensaje se entrega a todos los hosts existentes en la misma subred. Opcionalmente, los routers también reenvían el mensaje a las subredes vecinas (aunque habitualmente no lo hacen).

EN LA PRÁCTICA

Este ejemplo de un ISP que conecta a ocho organizaciones a Internet ilustra de forma conveniente cómo la asignación cuidadosa de direcciones CIDR facilita el enrutamiento. Suponga, como se muestra en la Figura 4.21, que el ISP (al que llamaremos ISP A) anuncia al mundo exterior que se le debe enviar cualquier datagrama cuyos primeros 20 bits de dirección se correspondan con 200.23.16.0/20. El resto del mundo no necesita saber que dentro del bloque de direcciones 200.23.16.0/20 existen en realidad otras ocho organizaciones, cada una con sus propias subredes. Esta capacidad de emplear un mismo prefijo para anunciar múltiples redes suele denominarse **agregación de direcciones** (o **agregación de rutas**, o también **resumen de rutas**).

La técnica de agregación de direcciones funciona extraordinariamente bien cuando las direcciones se asignan en bloques a los ISP y éstos las asignan en bloques a las organizaciones cliente. Pero, ¿qué ocurre si las direcciones no están asignadas de esa forma jerárquica? ¿Qué ocurriría, por ejemplo, si el ISP A adquiere el ISP B y luego hace que la Organización 1 se conecte a Internet a través del ISP B subsidiario? Como se muestra en la Figura 4.21, el ISP B subsidiario posee el bloque de direcciones 199.31.0.0/16, pero las direcciones IP de la Organización 1 lamentablemente no pertenecen a este bloque de direcciones. ¿Qué habría que hacer en este caso? Por supuesto, la Organización 1 podría renumerar todos sus routers y hosts para disponer de direcciones contenidas en el bloque de direcciones del ISP B. Pero ésta es una solución costosa y la Organización 1 podría ser reasignada en el futuro a otro ISP subsidiario. La solución que normalmente se adoptará es que la Organización 1 mantenga sus direcciones IP en 200.23.18.0/23. En este caso, como se muestra en la Figura 4.22, el ISP A continúa anunciando el bloque de direcciones 200.23.16.0/20 y el ISP B continúa anunciando el bloque 199.31.0.0/16. Sin embargo, el ISP B *también* anuncia ahora el bloque de direcciones de la Organización 1, 200.23.18.0/23. Cuando otros routers de Internet vean los bloques de direcciones 200.23.16.0/20 (del ISP A) y 200.23.18.0/23 (del ISP B) y deseen enrutar hacia una dirección contenida en el bloque 200.23.18.0/23, utilizarán la regla de *coincidencia con el prefijo más largo* (véase la Sección 4.2.1) y llevarán a cabo el enrutamiento hacia el ISP B, ya que anuncia el prefijo de dirección más largo (es decir, más específico) que se corresponde con la dirección de destino.

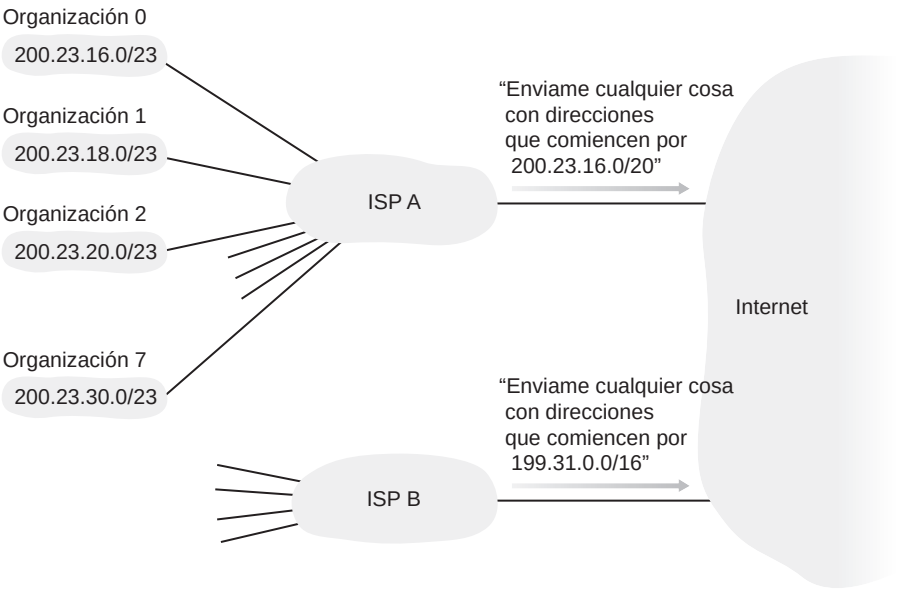


Figura 4.21 ♦ Agregación de rutas y direccionamiento jerárquicos.

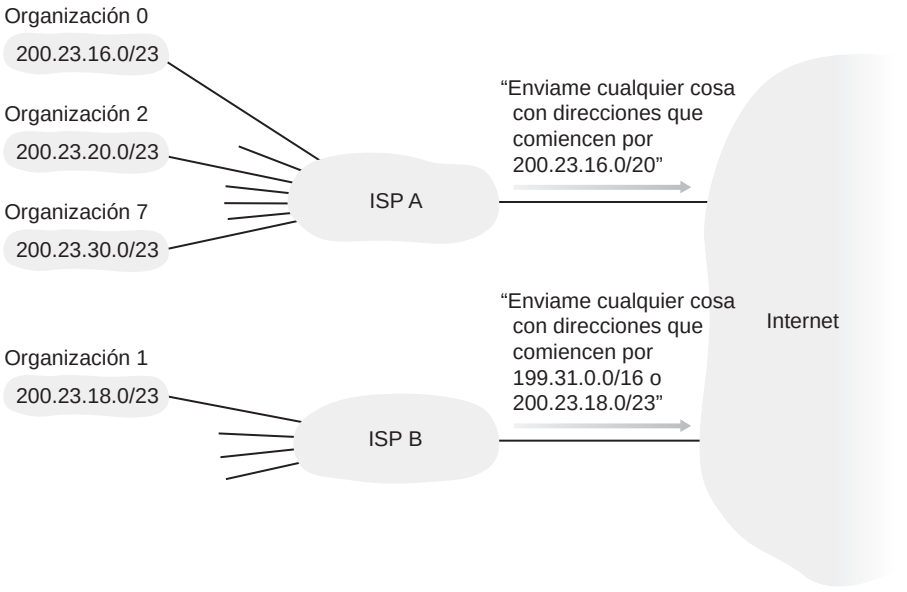


Figura 4.22 ♦ El ISP B tiene una ruta más específica hacia la Organización 1.

Ahora que hemos estudiado en detalle el direccionamiento IP, necesitamos saber cómo los hosts y las subredes obtienen sus direcciones en primer lugar. Comenzaremos viendo cómo una organización obtiene un bloque de direcciones para sus dispositivos, y luego veremos cómo se asigna una dirección del bloque de direcciones de la organización a un dispositivo (por ejemplo, a un host).

Cómo obtener un bloque de direcciones

Para obtener un bloque de direcciones IP que pueda ser utilizado dentro de la subred de una organización, el administrador de red tiene que contactar en primer lugar con su ISP, el cual le proporcionará direcciones extraídas de un bloque de direcciones mayor que ya habrá sido asignado al ISP. Por ejemplo, al ISP pueden haberle asignado el bloque de direcciones 200.23.16.0/20. A su vez, el ISP podría dividir su bloque de direcciones en ocho bloques de direcciones contiguas del mismo tamaño y asignar cada uno de estos bloques de direcciones a hasta ocho organizaciones a las que puede prestar servicio, como se muestra a continuación (hemos subrayado la parte de subred de estas direcciones, para facilitar su identificación).

Bloque del ISP:	200.23.16.0/20	11001000 00010111 0001**0000** 00000000
Organización 0	200.23.16.0/23	11001000 00010111 0001000**0** 00000000
Organización 1	200.23.18.0/23	11001000 00010111 0001001**0** 00000000
Organización 2	200.23.20.0/23	11001000 00010111 0001010**0** 00000000
…	…	…
Organización 7	200.23.30.0/23	11001000 00010111 0001111**0** 00000000

Obtener un conjunto de direcciones de un ISP es una forma de conseguir un bloque de direcciones, pero no es la única forma. Evidentemente, también debe existir una forma para el propio ISP de obtener un bloque de direcciones. ¿Existe una autoridad global cuya responsabilidad última sea gestionar el espacio de direcciones IP y asignar bloques de direcciones a los ISP y otras organizaciones? ¡Por supuesto que existe! Las direcciones IP son gestionadas por la ICANN

(*Internet Corporation for Assigned Names and Numbers*, Corporación de Internet para los números y nombres asignados) [ICANN 2016], basándose en las directrices establecidas en [RFC 7020]. El papel de la organización sin ánimo de lucro ICANN [NTIA 1998] no es solo el de asignar direcciones IP, sino también gestionar los servidores raíz DNS. También tiene el polémico trabajo de asignar nombres de dominio y de resolver las disputas por dichos nombres. La organización ICANN asigna direcciones a los registros regionales de Internet (por ejemplo, ARIN, RIPE, APNIC y LACNIC, que forman la Organización de Soporte de Direcciones de ICANN [ASO-ICANN 2016]), los cuales gestionan la asignación/administración de direcciones dentro de sus regiones.

Cómo obtener una dirección de host: Protocolo de configuración dinámica de host

Una vez que una organización ha obtenido un bloque de direcciones, puede asignar direcciones IP individuales a las interfaces de sus hosts y routers. Normalmente, un administrador de sistemas configura manualmente las direcciones IP de un router (a menudo de forma remota, mediante una herramienta de gestión de red). Las direcciones de host también se pueden configurar manualmente, pero frecuentemente esta tarea se lleva cabo utilizando el **Protocolo de configuración dinámica de host** (**DHCP, *Dynamic Host Configuration Protocol***) [RFC 2131]. DHCP permite a un host obtener (permite que se le asigne) automáticamente una dirección IP. Un administrador de red puede configurar DHCP de modo que un host dado reciba la misma dirección IP cada vez que se conecte a la red, o bien a un host puede asignársele una **dirección IP temporal** que será diferente cada vez que el host se conecte a la red. Además de la asignación de direcciones IP de host, DHCP también permite que un host obtenga información adicional, como por ejemplo su máscara de subred, la dirección de su router del primer salto [a menudo denominado pasarela (*gateway*) predeterminada] y la dirección de su servidor DNS local.

Gracias a la capacidad de DHCP de automatizar el proceso de conexión de un host a una red, a menudo se dice que es un protocolo **plug-and-play** o de **configuración cero (*zeroconf*)** ¡Esta capacidad le hace muy atractivo para el administrador de la red, que en otro caso tendría que realizar estas tareas manualmente! DHCP también disfruta de un amplio uso en las redes residenciales de acceso a Internet, las redes empresariales y las redes LAN inalámbricas, en las que los hosts se unen a la red y salen de ella frecuentemente. Considere, por ejemplo, un estudiante que traslada una computadora portátil desde su casa a la biblioteca y luego a clase. Probablemente, en cada localización el estudiante se conectará a una subred y, por tanto, necesitará una nueva dirección IP en cada lugar. DHCP está idealmente adaptado para estas situaciones, ya que existen muchos usuarios que van y vienen, y que necesitan direcciones solo durante un periodo de tiempo limitado. La ventaja de la capacidad plug-and-play de DHCP está claro, ya que resulta inimaginable que un administrador de sistemas pueda reconfigurar las computadores portátiles en cada posible ubicación y pocos estudiantes (¡excepto los que estén siguiendo un curso de redes de computadoras!) tienen los conocimientos necesarios para configurar sus portátiles manualmente.

DHCP es un protocolo cliente-servidor. Normalmente, el cliente es un host recién llegado que desea obtener información de configuración de la red, incluyendo una dirección IP para sí mismo. En el caso más simple, cada subred (en el sentido de direccionamiento mostrado en la Figura 4.20) tendrá un servidor DHCP. Si en la subred no hay ningún servidor, es necesario un agente de retransmisión DHCP (normalmente un router) que conozca la dirección de un servidor DHCP para dicha red. La Figura 4.23 muestra un servidor DHCP conectado a la subred 223.1.2/24, con el router actuando como agente de retransmisión para los clientes recién llegados que se conectan a las subredes 223.1.1/24 y 223.1.3/24. En la siguiente exposición, supondremos que hay disponible un servidor DHCP en la subred.

Para un host recién llegado a una red, el protocolo DHCP es un proceso de cuatro pasos, como se muestra en la Figura 4.24 para la configuración de red mostrada en la Figura 4.23. En esta figura, `sudirI` (“su dirección Internet”) indica la dirección que se asigna al cliente que acaba de llegar. Los cuatro pasos son los siguientes:

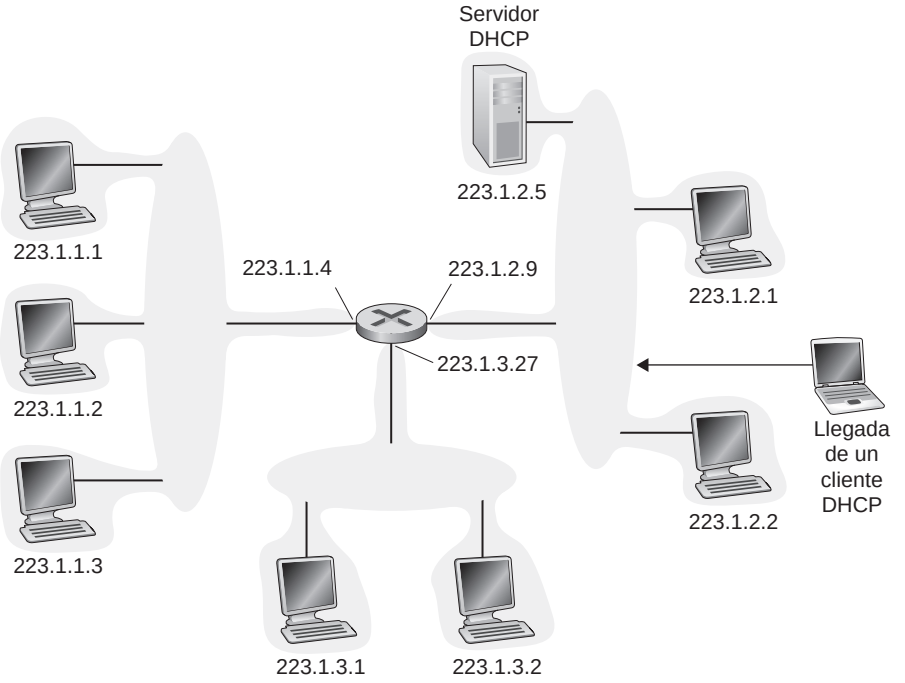


Figura 4.23 ♦ Servidor y clientes DHCP.

- *Descubrimiento del servidor DHCP.* La primera tarea de un host recién llegado es encontrar un servidor DHCP con el que interactuar. Esto se hace mediante un **mensaje de descubrimiento DHCP**, que envía un cliente dentro de un paquete UDP al puerto 67. El paquete UDP se encapsula en un datagrama IP. Pero, ¿a quién debería enviarse este datagrama? El host ni siquiera conoce la dirección IP de la red a la que se está conectando, y mucho menos la dirección de un servidor DHCP de esa red. En esta situación, el cliente DHCP crea un datagrama IP que contiene su mensaje de descubrimiento DHCP junto con la dirección IP de difusión 255.255.255.255 y una dirección IP de origen de "este host" igual a 0.0.0.0. El cliente DHCP pasa el datagrama IP a la capa de enlace, la cual difunde esta trama a todos los nodos conectados a la subred (en la Sección 6.4 estudiaremos en detalle el proceso de difusión de la capa de enlace).
- *Oferta(s) del servidor DHCP.* Un servidor DHCP que recibe un mensaje de descubrimiento DHCP responde al cliente con un **mensaje de oferta DHCP**, que se difunde a todos los nodos de la subred utilizando de nuevo la dirección IP de difusión 255.255. 255.255 (trate de pensar en por qué la respuesta de este servidor también debe difundirse a todos los nodos). Puesto que en la subred pueden existir varios servidores DHCP, el cliente puede encontrarse en la envidiable situación de poder elegir entre varias ofertas. Cada mensaje de oferta de un servidor contiene el ID de transacción del mensaje de descubrimiento recibido, la dirección IP propuesta para el cliente, la máscara de red y el **tiempo de arrendamiento de la dirección** IP (el tiempo durante el que la dirección IP será válida). Es habitual que el servidor defina un tiempo de arrendamiento de varias horas o días [Droms 2002].
- *Solicitud DHCP.* El cliente recién llegado seleccionará de entre las ofertas de servidor y responderá a la oferta seleccionada con un **mensaje de solicitud DHCP**, devolviendo los parámetros de configuración.
- *ACK DHCP.* El servidor contesta al mensaje de solicitud DHCP con un **mensaje ACK DHCP**, que confirma los parámetros solicitados.

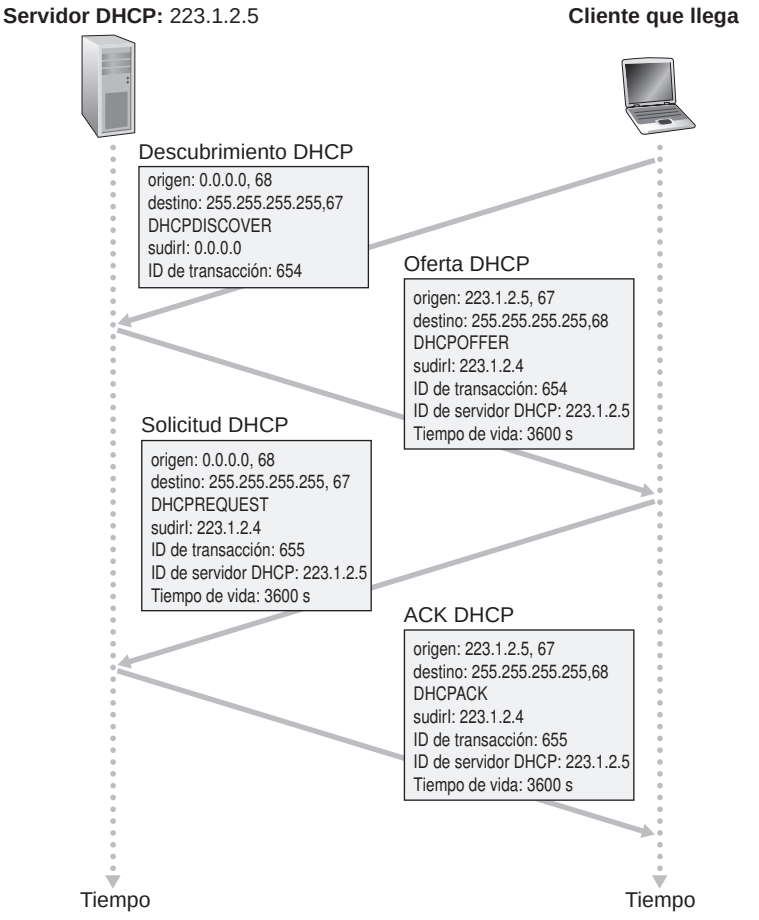


Figure 4.24 ♦ Interacción cliente-servidor DHCP.

Una vez que el cliente recibe el mensaje de confirmación (ACK) DHCP, la interacción se habrá completado y el cliente puede utilizar la dirección IP asignada por DHCP durante todo el tiempo de arrendamiento. Dado que un cliente puede desear utilizar su dirección durante más tiempo del arrendado, DHCP también proporciona un mecanismo que permite a un cliente renovar su arrendamiento de una dirección IP.

En lo que respecta a la movilidad, DHCP presenta una deficiencia importante. Puesto que se obtiene una nueva dirección IP mediante DHCP cada vez que un nodo se conecta a una nueva subred, una conexión TCP con una aplicación remota no podría mantenerse a medida que un nodo móvil se desplaza de una subred a otra. En el Capítulo 6 veremos la infraestructura de IP móvil, una extensión de la infraestructura IP que permite a un nodo móvil utilizar una única dirección permanente según se va desplazado entre subredes. Puede encontrar información adicional sobre DHCP en [Droms 2002] y [dhc 2016]. En Internet Systems Consortium [ISC 2016] hay disponible una implementación de referencia de código fuente abierto para DHCP.

4.3.4 Traducción de direcciones de red (NAT)

Después de haber estudiado las direcciones de Internet y el formato de los datagramas IPv4, somos completamente conscientes de que todo dispositivo IP necesita una dirección IP. Con la proliferación de las subredes de oficina doméstica y pequeña oficina (SOHO, *Small Office, Home Office*), podría parecer que esto implica que, cuando una red SOHO desea instalar una LAN para conectar

varias máquinas, el ISP debería asignar un rango de direcciones para cubrir todos los dispositivos IP de la red SOHO (incluyendo teléfonos, tabletas, dispositivos de juegos, televisiones IP, impresoras y otros). Si la subred creciera, habría que asignar un bloque de direcciones mayor. Pero ¿qué ocurre si el ISP ya ha asignado las porciones adyacentes al rango de direcciones actual de la red SOHO? ¿Y qué persona normal querría (o necesitaría) saber cómo gestionar las direcciones IP de la red de su casa? Afortunadamente, existe una forma más simple de asignar direcciones que ha encontrado un uso cada vez más amplio en escenarios de este tipo: la **traducción de direcciones de red** (**NAT, *Network Address Translation***) [RFC 2663; RFC 3022; Huston 2004; Zhang 2007; Cisco NAT 2016].

La Figura 4.25 muestra el funcionamiento de un router con funcionalidad NAT. Este router, que se encuentra en una vivienda, tiene una interfaz que forma parte de la red doméstica situada en la parte derecha de la Figura 4.25. El direccionamiento dentro de la red doméstica es exactamente como hemos visto anteriormente (las cuatro interfaces de la red tienen la misma dirección de subred 10.0.0.0/24). El espacio de direcciones 10.0.0.0/8 corresponde a una de las tres partes del espacio de direcciones IP que está reservado en [RFC 1918] para una **red privada** o para un **ámbito con direcciones privadas**, como la red doméstica de la Figura 4.25. Un ámbito con direcciones privadas hace referencia a una red cuyas direcciones solo tienen significado para los dispositivos internos de dicha red. Veamos por qué esto es importante. Considere el hecho de que existen cientos de miles de redes domésticas y que muchas utilizan el mismo espacio de direcciones, 10.0.0.0/24. Los dispositivos de una red doméstica dada pueden enviarse paquetes entre sí utilizando el direccionamiento 10.0.0.0/24. Sin embargo, los paquetes reenviados *más allá* de la red doméstica, hacia la Internet global, evidentemente no pueden utilizar estas direcciones (ni como dirección de origen ni como dirección de destino), porque existen cientos de miles de redes que emplean ese mismo bloque de direcciones. Es decir, las direcciones 10.0.0.0/24 solo tienen significado dentro de una red doméstica dada. Pero si las direcciones privadas solo tienen significado dentro de la red, ¿cómo se direccionan los paquetes cuando se envían a Internet o se reciben de Internet, donde necesariamente las direcciones tienen que ser unívocas? Para entender esto hay que comprender cómo funciona NAT.

El router NAT no *parece* un router a ojos del mundo exterior. En su lugar, el router NAT se comporta de cara al exterior como un *único* dispositivo con una dirección IP única. En la Figura 4.25, todo el tráfico que sale del router doméstico hacia Internet tiene una dirección IP de origen igual a 138.76.29.7, y todo el tráfico que entra en él tiene que tener la dirección de destino 138.76.29.7.

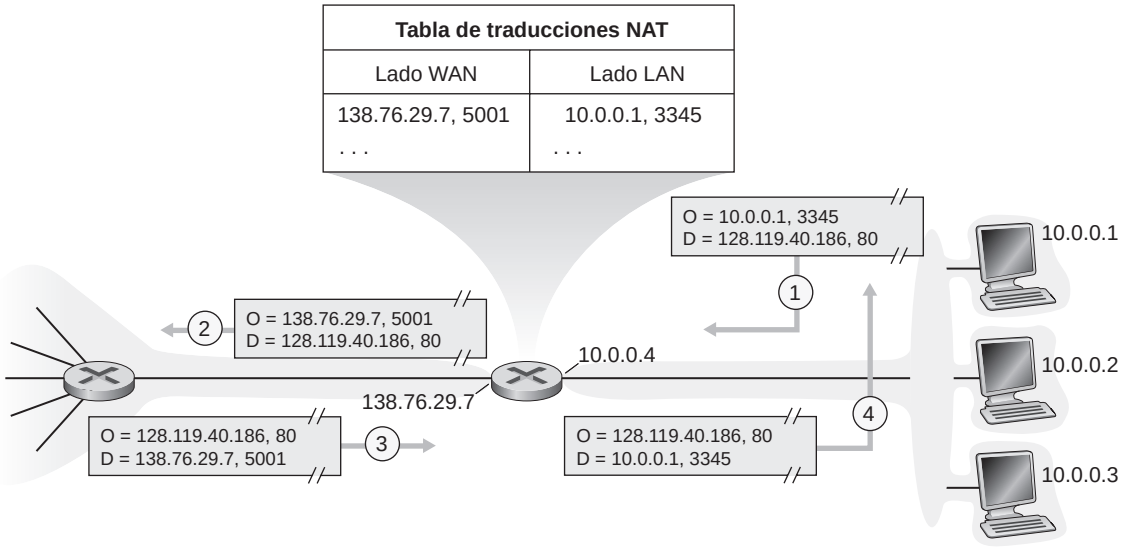


Figura 4.25 ♦ Traducción de direcciones de red.

En resumen, el router NAT oculta los detalles de la red doméstica al mundo exterior. (Como nota al margen, posiblemente se esté preguntando dónde obtienen las computadoras de la red doméstica sus direcciones y dónde obtiene el router su dirección IP unívoca. A menudo, la respuesta a ambas preguntas es la misma: ¡DHCP! El router obtiene su dirección del servidor DHCP del ISP y el router ejecuta un servidor DHCP para proporcionar direcciones a las computadoras, dentro del espacio de direcciones de la red doméstica controlado por el router NAT-DHCP.)

Si todos los datagramas que llegan al router NAT procedentes de la WAN tienen la misma dirección IP de destino (específicamente, la de la interfaz WAN del router NAT), entonces ¿cómo sabe el router a qué host interno debería reenviar un datagrama dado? El truco consiste en utilizar una **tabla de traducciones NAT** almacenada en el router NAT, e incluir números de puerto junto con las direcciones IP en las entradas de la tabla.

Considere el ejemplo de la Figura 4.25. Suponga que un usuario de una red doméstica que utiliza el host con la dirección 10.0.0.1 solicita una página web almacenada en un servidor web (puerto 80) con la dirección IP 128.119.40.186. El host 10.0.0.1 asigna el número de puerto de origen (arbitrario) 3345 y envía el datagrama a la LAN. El router NAT recibe el datagrama, genera un nuevo número de puerto de origen, 5001, para el datagrama, sustituye la dirección IP de origen por su propia dirección IP de la red WAN 138.76.29.7, y sustituye el número de puerto de origen original 3345 por el nuevo número de puerto de origen 5001. Al generar un nuevo número de puerto de origen, el router NAT puede seleccionar cualquier número de puerto de origen que no se encuentre actualmente en la tabla de traducciones NAT. (¡Observe que, puesto que la longitud del campo de número de puerto es de 16 bits, el protocolo NAT puede dar soporte a más de 60.000 conexiones simultáneas utilizando una única dirección IP para el lado WAN del router!) En el router, NAT también añade una entrada a su tabla de traducciones. El servidor web, que afortunadamente no es consciente de que el datagrama entrante que contiene la solicitud HTTP ha sido manipulado por el router NAT, responde con un datagrama cuya dirección de destino es la dirección IP del router NAT y cuyo número de puerto de destino es 5001. Cuando este datagrama llega al router NAT, éste indexa la tabla de traducciones NAT utilizando la dirección IP de destino y el número de puerto de destino para obtener la dirección IP (10.0.0.1) y el número de puerto de destino (3345) apropiados para el navegador de la red doméstica. A continuación, el router reescribe la dirección de destino y el número de puerto de destino del datagrama y lo reenvía a la red doméstica.

NAT ha disfrutado de una gran difusión en los últimos años, aunque no carece de detractores. En primer lugar, se podría argumentar que los números de puerto están pensados para direccionar procesos, no para direccionar hosts. De hecho, esta violación puede causar problemas en los servidores que se estén ejecutando en la red doméstica, ya que, como hemos visto en el Capítulo 2, los procesos de servidor están a la espera de las solicitudes entrantes en números de puerto bien conocidos; y los homólogos en un protocolo P2P necesitan aceptar conexiones entrantes cuando actúan como servidores. Entre las soluciones técnicas a estos problemas están las herramientas de **NAT traversal** [RFC 5389], así como UPnP (*Universal Plug and Play*), un protocolo que permite a un host descubrir y configurar un NAT cercano [UPnP Forum 2016].

Los puristas de la arquitectura han planteado también argumentos más "filosóficos" contra NAT. Lo que les preocupa es que los routers están pensados como dispositivos de capa 3 (es decir, de la capa de red) y solo deberían procesar paquetes hasta la capa de red. NAT viola este principio de que los hosts deberían hablar directamente unos con otros, sin que nodos intermedios modifiquen las direcciones IP, y mucho menos los números de puerto. Pero, guste o no, NAT se ha convertido en un componente importante de Internet, al igual que otros tipos de **dispositivos intermediarios** (*middleboxes*) [Sekar 2011] que operan en la capa de red, pero tienen funciones muy diferentes de las de los routers. Los dispositivos intermediarios no llevan a cabo el reenvío tradicional de datagramas, sino que en su lugar realizan funciones como NAT, equilibrado de carga de los flujos de tráfico, cortafuegos (véase el recuadro Seguridad adjunto) y otras. El paradigma de reenvío generalizado, que en breve estudiaremos en la Sección 4.4, hace posible llevar a cabo varias de estas funciones intermediarias y el reenvío tradicional de los routers, de una forma integrada y común.

SEGURIDAD

INSPECCIÓN DE DATAGRAMAS: CORTAFUEGOS Y SISTEMAS DE DETECCIÓN DE INTRUSIONES

Suponga que le han asignado la tarea de administrar una red doméstica, departamental, universitaria o corporativa. Los atacantes, que saben cuál es el rango de direcciones IP de su red, pueden enviar fácilmente datagramas IP a las direcciones de ese rango. Estos datagramas pueden hacer toda clase de cosas retorcidas, incluyendo confeccionar mapas de la red mediante barridos de ping y escaneo de puertos; dañar los hosts vulnerables con paquetes erróneos; hacer barridos para detectar puertos TCP/UDP abiertos en los servidores de nuestra red e infectar los hosts incluyendo software malicioso en los paquetes. Como administrador de la red, ¿qué hará con todos esos malvados capaces de enviar paquetes maliciosos a su red? Hay disponibles dos mecanismos de defensa muy populares contra los ataques de paquetes maliciosos: los cortafuegos y los sistemas de detección de intrusiones (IDS, *Intrusion Detection System*).

Como administrador de la red, primero puede probar a instalar un cortafuegos entre su red e Internet. (La mayoría de los routers actuales de acceso disponen de cortafuegos.) Los cortafuegos inspeccionan los campos de cabecera de los segmentos y datagramas, denegando el acceso a la red interna a los datagramas sospechosos. Por ejemplo, un cortafuegos puede configurarse para bloquear todos los paquetes de solicitud de eco ICMP (véase la Sección 5.6), impidiendo así que un atacante lleve a cabo un tradicional barrido de puertos a lo largo de su rango de direcciones IP. Los cortafuegos también pueden bloquear paquetes basándose en las direcciones IP y números de puerto de origen y de destino. Además, los cortafuegos se pueden configurar para controlar las conexiones TCP, dejando entrar sólo a los datagramas que pertenezcan a conexiones aprobadas.

Con un sistema IDS puede proporcionarse protección adicional. Un IDS, colocado normalmente en la frontera de la red, realiza una "inspección profunda de los paquetes", examinando no sólo los campos de cabecera sino también las cargas útiles de los datagramas (incluyendo los datos de la capa de aplicación). Un IDS dispone de una base de datos de signaturas de paquetes que se sabe que forman parte de ataques. Esta base de datos se actualiza automáticamente a medida que se descubren nuevos tipos de ataque. A medida que los paquetes atraviesan el sistema IDS, éste intenta encontrar una coincidencia entre los campos de cabecera o las cargas útiles y las signaturas que almacena en su base de datos. Si encuentra una coincidencia, genera una alerta. Un sistema de prevención de intrusiones (IPS, *Intrusion Prevention System*) es similar a un sistema IDS, salvo porque además de generar una alerta, bloquea los paquetes. En el Capítulo 8 estudiaremos los cortafuegos y los sistemas IDS con más detalle.

¿Pueden los cortafuegos y los sistemas IDS proteger completamente a la red frente a todos los ataques? Evidentemente, la respuesta es no, ya que los atacantes encuentran continuamente nuevos ataques para los que las signaturas todavía no están disponibles. No obstante, los cortafuegos y los IDS tradicionales basados en signaturas son útiles para proteger a las redes de los ataques conocidos.

4.3.5 IPv6

A principios de la década de 1990, el Internet Engineering Task Force comenzó a desarrollar un sucesor para el protocolo IPv4. La principal motivación de esta iniciativa fue que se dieron cuenta de que el espacio de direcciones IP de 32 bits estaba comenzando a agotarse, a causa de la velocidad pasmosa con que estaban conectándose a Internet nuevas subredes y nodos IP (a los que se les asignaban direcciones IP unívocas). Para responder a esta necesidad de un espacio de direcciones IP más grande, se desarrolló un nuevo protocolo IP, el protocolo IPv6. Los diseñadores de IPv6 también aprovecharon la oportunidad para ajustar y expandir otros aspectos de IPv4, basándose en la experiencia acumulada sobre el funcionamiento de dicho protocolo.

El momento en que las direcciones IPv4 se agotarían (y por tanto ninguna nueva subred podría conectarse a Internet) fue objeto de un importante debate. Las estimaciones de los dos

líderes del grupo de trabajo *Address Lifetime Expectations* (Expectativas del tiempo de vida de las direcciones) del IETF fueron que las direcciones se agotarían en 2008 y 2018, respectivamente [Solensky 1996]. En febrero de 2011, la IANA asignó a un registro regional el último grupo de direcciones IPv4 que quedaba sin asignar. Aunque los registros regionales aun disponen de direcciones IPv4 sin asignar dentro de sus respectivos bloques, una vez que se agoten esas direcciones ya no habrá más bloques de direcciones disponibles que puedan asignarse desde ninguna autoridad central [Huston 2011a]. En [Richter 2015] puede encontrar un repaso reciente del tema del agotamiento del espacio de direcciones IPv4, así como de los pasos tomados para prolongar la vida del espacio de direcciones.

Aunque las estimaciones sobre el agotamiento de las direcciones IPv4 realizadas a mediados de la década de 1990 sugerían que quedaba bastante tiempo para que el espacio de direcciones de IPv4 se agotara, se hizo evidente que se necesitaría un tiempo considerable para implantar una nueva tecnología a tan gran escala, y por eso se comenzó a trabajar [RFC 1752] en el desarrollo de la versión 6 de IP (IPv6) [RFC 2460]. (Una pregunta que se plantea a menudo es qué ocurrió con IPv5. Inicialmente se pensó que el protocolo ST-2 se convertiría en IPv5, pero ST-2 fue descartado más tarde.) Una excelente fuente de información sobre IPv6 es [Huitema 1998].

Formato del datagrama IPv6

El formato del datagrama IPv6 se muestra en la Figura 4.26. Los cambios más importantes introducidos en IPv6 son evidentes en el formato de su datagrama:

- *Capacidades ampliadas de direccionamiento.* IPv6 aumenta el tamaño de la dirección IP, de 32 a 128 bits. De esta manera, se asegura que el mundo no se quedará sin direcciones IP. Ahora, cada grano de arena del planeta podría tener asignada una dirección IP. Además de las direcciones de unidifusión y de multidifusión, IPv6 ha introducido un nuevo tipo de dirección, denominado **dirección anycast**, que permite entregar un datagrama a uno cualquiera de un grupo de hosts. (Esta funcionalidad podría utilizarse, por ejemplo, para enviar un mensaje GET HTTP al más próximo de una serie de sitios espejo que contengan un determinado documento.)
- *Una cabecera de 40 bytes simplificada.* Como se explica más adelante, algunos de los campos de IPv4 se han eliminado o se han hecho opcionales. La cabecera resultante, con una longitud fija de 40 bytes, permite un procesamiento más rápido del datagrama IP en los routers. Una nueva codificación de las opciones permite un procesamiento más flexible de las mismas.
- *Etiquetado del flujo.* IPv6 utiliza una definición bastante amplia de **flujo**. El documento RFC 2460 establece que esto permite "etiquetar los paquetes que pertenecen a determinados flujos para los que el emisor solicita un tratamiento especial, como una calidad de servicio no prede-

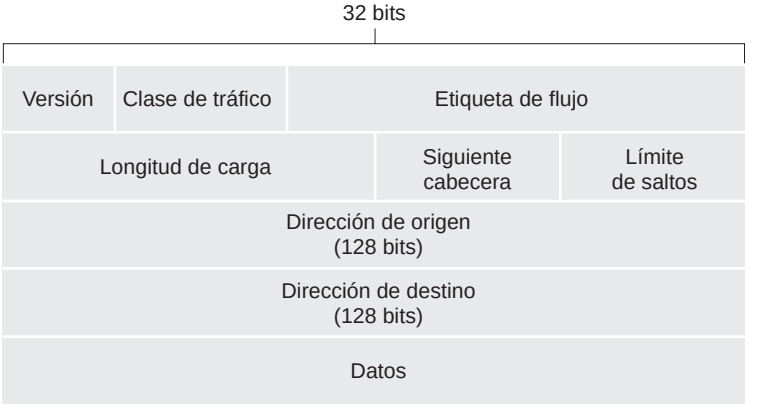


Figura 4.26 ♦ Formato del datagrama IPv6.

terminada o un servicio en tiempo real". Por ejemplo, la transmisión de audio y de vídeo puede posiblemente tratarse como un flujo. Por el contrario, aplicaciones más tradicionales, como la transferencia de archivos y el correo electrónico, podrían no tratarse como flujos. Es posible que el tráfico transportado por un usuario de alta prioridad (por ejemplo, alguien que paga por obtener un mejor servicio para su tráfico) pueda ser tratado también como un flujo. Sin embargo, lo que está claro es que los diseñadores de IPv6 previeron la eventual necesidad de poder diferenciar entre los flujos, incluso aunque el significado exacto de flujo no hubiera sido determinado aún.

Como se ha mencionado anteriormente, una comparación de la Figura 4.26 con la Figura 4.16 revela la estructura más simple y estilizada del datagrama de IPv6. En IPv6 se definen los siguientes campos:

- *Versión.* Este campo de 4 bits identifica el número de versión IP. Nada sorprendentemente, IPv6 transporta un valor de 6 en este campo. Observe que incluir en este campo el valor 4 no implica que se cree un datagrama IPv4 válido. (Si lo hiciera, la vida sería mucho más simple —véase más adelante la exposición sobre la transición de IPv4 a IPv6—).
- *Clase de tráfico.* Al igual que el campo TOS de IPv4, el campo de clase de tráfico, de 8 bits, puede utilizarse para dar prioridad a ciertos datagramas dentro de un flujo, o para dar prioridad a los datagramas de determinadas aplicaciones (por ejemplo, Voz sobre IP) frente a los datagramas de otras aplicaciones (por ejemplo, correo electrónico SMTP).
- *Etiqueta de flujo.* Como hemos mencionado anteriormente, este campo de 20 bits se utiliza para identificar un flujo de datagramas.
- *Longitud de la carga útil.* Este valor de 16 bits se trata como un entero sin signo que proporciona el número de bytes del datagrama IPv6 incluidos a continuación de la cabecera del datagrama, que es de 40 bytes y tiene longitud fija.
- *Siguiente cabecera.* Este campo identifica el protocolo (por ejemplo, TCP o UDP) al que se entregará el contenido (el campo de datos) de este datagrama. El campo utiliza los mismos valores que el campo de protocolo de la cabecera IPv4.
- *Límite de saltos.* Cada router que reenvía un datagrama decrementa el contenido de este campo en una unidad. Si el límite de saltos alcanza el valor cero, el datagrama se descarta.
- *Direcciones de origen y de destino.* Los distintos formatos de la dirección de 128 bits de IPv6 se describen en el documento RFC 4291.
- *Datos.* Esta es la parte de la carga útil del datagrama IPv6. Cuando el datagrama llegue a su destino, la carga útil se extraerá del datagrama IP y se pasará al protocolo especificado en el campo Siguiente cabecera.

Hasta aquí hemos identificado el propósito de los campos incluidos en los datagramas IPv6. Comparando el formato del datagrama IPv6 de la Figura 4.26 con el formato del datagrama IPv4 que hemos visto en la Figura 4.16, podemos observar que varios campos del datagrama IPv4 ya no aparecen en IPv6:

- *Fragmentación/Reensamblado.* IPv6 no permite ni la fragmentación ni el reensamblado en routers intermedios; estas operaciones solo pueden ser realizadas por el origen y el destino. Si un router recibe un datagrama IPv6 y es demasiado largo para ser reenviado por el enlace de salida, el router simplemente lo descarta y envía de vuelta al emisor un mensaje de error ICMP "Paquete demasiado grande" (véase la Sección 5.6). El emisor puede entonces reenviar los datos utilizando un tamaño de datagrama IP más pequeño. La fragmentación y el reensamblado son operaciones que consumen mucho tiempo, por lo que eliminando esta funcionalidad de los routers e incluyéndola directamente en los sistemas terminales se acelera considerablemente el reenvío IP dentro de la red.

- *Suma de comprobación de cabecera.* Puesto que los protocolos de la capa de transporte (por ejemplo, TCP y UDP) y de la capa de enlace de datos (por ejemplo, Ethernet) de Internet utilizan sumas de comprobación, los diseñadores de IP probablemente pensaron que esta funcionalidad ya era suficientemente redundante en la capa de red y podía eliminarse. Una vez más, el procesamiento rápido de los paquetes IP era la preocupación principal. Recuerde de la Sección 4.3.1 que, como la cabecera de IPv4 contiene un campo TTL (similar al campo de límite de saltos de IPv6), la suma de comprobación de la cabecera IPv4 necesitaba ser recalculada en cada router. Al igual que la fragmentación y el reensamblado, esta también era una operación muy costosa en IPv4.
- *Opciones.* La cabecera IP estándar ya no incluye un campo de opciones. Sin embargo, las opciones no han desaparecido. En su lugar, el campo de opciones es una de las posibles siguientes cabeceras a las que se apunta desde dentro de la cabecera IPv6. Es decir, al igual que las cabeceras de los protocolos TCP o UDP pueden ser la siguiente cabecera dentro de un paquete IP, también puede serlo un campo de opciones. La eliminación del campo de opciones dio como resultado una cabecera IP de 40 bytes de longitud fija.

Transición de IPv4 a IPv6

Ahora que hemos visto los detalles técnicos de IPv6, vamos a considerar una cuestión práctica: ¿cómo va a hacer Internet, que está basada en IPv4, la transición a IPv6? El problema está en que, mientras que los nuevos sistemas para IPv6 pueden hacerse compatibles en sentido descendente, es decir, pueden enviar, enrutar y recibir datagramas IPv4, los sistemas para IPv4 ya implantados no son capaces de manejar datagramas IPv6. Existen varias posibles soluciones [Huston 201b, RFC 4213].

Una opción sería declarar un día D: una fecha y hora en la que todas las máquinas conectadas a Internet se apagaran y actualizaran de IPv4 a IPv6. La última transición importante de una tecnología a otra (de NCP a TCP como servicio de transporte fiable) tuvo lugar hace casi 35 años. Incluso entonces [RFC 801], cuando Internet era pequeña y todavía era administrada por un número pequeño de "gurús", se dieron cuenta de que establecer un día D no era posible. Con mayor razón, un día D que implicara a miles de millones de dispositivos sería aún más impensable hoy en día.

La solución más ampliamente adoptada en la práctica para la transición de IPv4 a IPv6 es la de **tunelización** [RFC 4213]. La idea básica en la que descansa el método de tunelización (un concepto clave de las aplicaciones en muchos otros escenarios, más allá de la transición de IPv4 a IPv6, incluyendo su uso generalizado en las redes móviles IP de las que hablaremos en el Capítulo 7) es la siguiente: suponga que dos nodos IPv6 (por ejemplo los nodos B y E de la Figura 4.27) desean interoperar utilizando datagramas IPv6, pero están conectados entre sí a través de routers IPv4. Al conjunto de routers intermedios IPv4 existentes entre dos routers IPv6 lo denominaremos **túnel**, como se ilustra en la Figura 4.27. Mediante la tunelización, el nodo IPv6 del lado emisor del túnel (en este ejemplo, B) toma el datagrama IPv6 *completo* y lo incluye en el campo de datos (carga útil) de un datagrama IPv4. Este datagrama IPv4 se direcciona entonces hacia el nodo IPv6 del lado receptor del túnel (en este ejemplo, E) y se envía al primer nodo del túnel (en este ejemplo, C). Los routers intermedios IPv4 del túnel enrutan este datagrama IPv4 entre ellos mismos, como si fuera cualquier otro datagrama, siendo totalmente inconscientes de que el datagrama IPv4 contiene un datagrama IPv6 completo. El nodo IPv6 del lado de recepción del túnel termina recibiendo el datagrama IPv4 (¡es el destino del datagrama IPv4!), determina que el datagrama IPv4 contiene un datagrama IPv6 (viendo que el campo de número de protocolo del datagrama IPv4 es 41 [RFC 4213], lo que indica que la carga útil IPv4 es un datagrama IPv6), extrae el datagrama IPv6 y, a continuación, lo enruta exactamente igual que si hubiera recibido el datagrama IPv6 desde un vecino IPv6 directamente conectado.

Para terminar esta sección, diremos que aunque la adopción de IPv6 inicialmente fue lenta [Lawton 2001; Huston 2008b], recientemente está empezando a tomar impulso. NIST [NIST

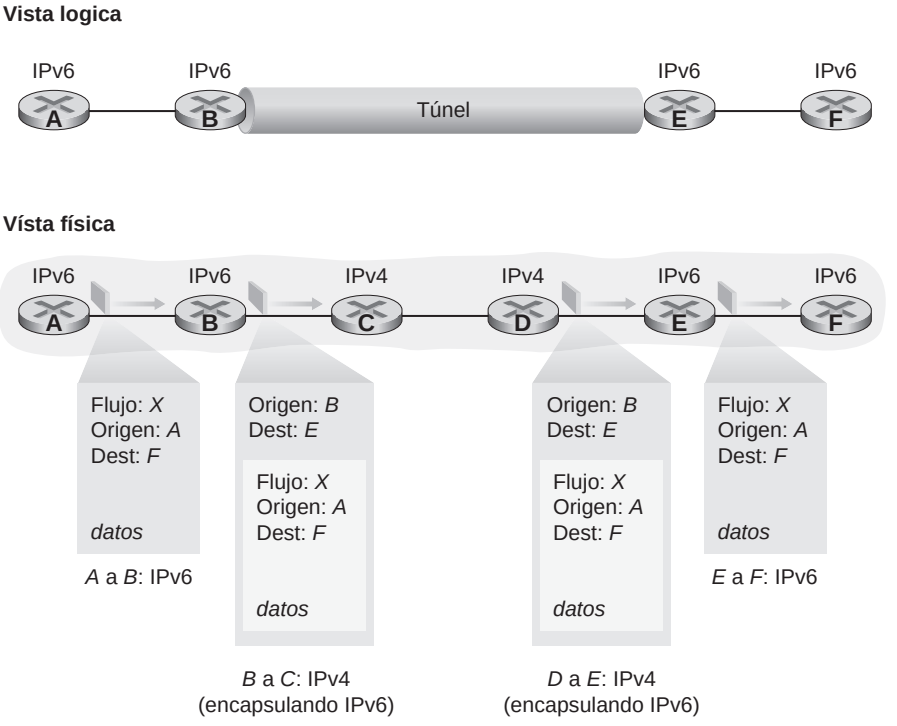


Figura 4.27 ♦ Tunelización.

IPv6 2015] informa de que más de un tercio de los dominios de segundo nivel del gobierno de los EE.UU. son compatibles con IPv6. En el lado de los clientes, Google afirma que solo un 8%, aproximadamente, de los clientes que acceden a los servicios Google lo hacen a través de IPv6 [Google IPv6 2015]. Pero otras medidas recientes [Czyz 2014] indican que la adopción de IPv6 se está acelerando. La proliferación de dispositivos como teléfonos compatibles con IP y otros dispositivos portátiles proporciona un empuje adicional para una más amplia implantación de IPv6. El programa europeo Third Generation Partnership Program [3GPP 2016] ha especificado IPv6 como esquema estándar de direccionamiento para multimedia móvil.

Una lección importante que podemos aprender de la experiencia de IPv6 es que resulta enormemente complicado cambiar los protocolos de la capa de red. Desde principios de la década de 1990, se han anunciado numerosos nuevos protocolos de la capa de red como la siguiente revolución más importante de Internet, pero la mayoría de estos protocolos han tenido una penetración limitada hasta la fecha. Entre estos protocolos se incluyen IPv6, los protocolos de multidifusión y los protocolos de reserva de recursos; en el suplemento en línea de este libro hay una explicación de estos dos últimos protocolos. De hecho, la introducción de nuevos protocolos en la capa de red es como sustituir los cimientos de una casa: resulta difícil de hacer sin tirar abajo toda la casa o al menos reubicar temporalmente a sus habitantes. Por otro lado, Internet ha sido testigo de la rápida implantación de nuevos protocolos en la capa de aplicación. Ejemplos clásicos de esto son, por supuesto, la Web, la mensajería instantánea, los flujos multimedia, los juegos distribuidos y diversos tipos de redes sociales. Introducir nuevos protocolos de la capa de aplicación es como añadir una nueva capa de pintura a las paredes de una casa: es relativamente fácil de hacer y, si se elige un color atractivo, otras personas del vecindario nos copiarán. En resumen, en el futuro podemos esperar ver cambios en la capa de red de Internet, por supuesto que sí, pero probablemente esos cambios se producirán en una escala temporal mucho más lenta que los cambios que tendrán lugar en la capa de aplicación.

4.4 Reenvío generalizado y SDN

En la Sección 4.2.1 hemos comentado que las decisiones de reenvío de un router de Internet han estado basadas tradicionalmente tan solo en la dirección de destino de los paquetes. En la sección anterior, sin embargo, hemos visto también que se ha producido una proliferación de dispositivos intermediarios que realizan muchas funciones de la capa 3. Los dispositivos NAT reescriben las direcciones IP y los números de puerto de las cabeceras de los datagramas; los cortafuegos bloquean el tráfico basándose en los valores de los campos de cabecera o redirigen los paquetes para un procesamiento adicional, como por ejemplo una inspección profunda de paquetes (DPI, *Deep Packet Inspection*). Los dispositivos de equilibrado de carga reenvían los paquetes que solicitan un determinado servicio (por ejemplo, solicitudes HTTP) a uno entre un conjunto de servidores que proporcionan dicho servicio. [RFC 3234] enumera diversas funciones intermediarias comunes.

Esta proliferación de dispositivos intermediarios, switches de capa 2 y routers de capa 3 [Qazi 2013] —cada uno con sus propios hardware, software e interfaces de administración especializados— ha resultado, sin lugar a dudas, en costosos dolores de cabeza para muchos operadores de red. Sin embargo, los recientes avances en redes definidas por software prometían, y están ahora cumpliendo, proporcionar un enfoque unificado para la realización de muchas de estas funciones de la capa de red, así como de ciertas funciones de la capa de enlace, de una manera moderna, elegante e integrada.

Recuerde que en la Sección 4.2.1 caracterizamos el reenvío basado en el destino como un proceso en dos pasos, consistente en buscar una dirección IP de destino ("correspondencia") y luego enviar el paquete hacia el puerto de salida especificado, a través del entramado de conmutación ("acción"). Vamos a considerar ahora un paradigma "correspondencia más acción" bastante más general, en el que la "correspondencia" puede buscarse para múltiples campos de cabecera asociados con diferentes protocolos en diferentes capas de la pila de protocolos. La "acción" puede consistir en reenviar el paquete a uno o más puertos de salida (como en el caso del reenvío basado en el destino), en enviar los paquetes a una u otra de entre múltiples interfaces salientes que conducen a un cierto servicio (como en el equilibrado de carga), en reescribir valores de la cabecera (como en NAT), en bloquear/eliminar un paquete deliberadamente (como en un cortafuegos); en enviar el paquete a un servidor especial para procesamiento y acciones adicionales (como en DPI) u otras acciones.

En el reenvío generalizado, una tabla de correspondencia más acción generaliza la noción de tabla de reenvío basada en el destino que ya hemos visto en la Sección 4.2.1. Puesto que las decisiones de reenvío pueden tomarse basándose en las direcciones de origen y de destino de la capa de red y/o de la capa de enlace, los dispositivos de reenvío mostrados en la Figura 4.28 se podrían describir con más precisión como "conmutadores de paquetes", más que como "routers" de la capa 3 o "switches" de la capa 2. Por ello, en el resto de esta sección, y en la Sección 5.5, denominaremos a estos dispositivos conmutadores de paquetes, adoptando la terminología que está consiguiendo una amplia aceptación en la literatura sobre SDN.

La Figura 4.28 muestra una tabla correspondencia-acción (*match plus action*) en cada conmutador de paquetes, siendo dicha tabla calculada, instalada y actualizada por un controlador remoto. Hay que resaltar que, aunque resulta posible para los componentes de control de cada conmutador de paquetes individual interaccionar entre sí (por ejemplo, de forma similar a la de la Figura 4.2), en la práctica la funcionalidad generalizada correspondencia-acción se implementa mediante un controlador remoto que calcula, instala y actualiza esas tablas. Tómese un momento para comparar las figuras 4.2, 4.3 y 4.28. ¿Qué similitudes y diferencias encuentra entre el reenvío basado en el destino, ilustrado en las Figuras 4.2 y 4.3, y el reenvío generalizado mostrado en la Figura 4.28?

Nuestra siguiente exposición sobre el reenvío generalizado está basada en OpenFlow [McKeown 2008, OpenFlow 2008, Casado 2014, Tourrilhes 2014], un estándar muy conocido y aceptado que ha sido pionero en lo que respecta a la abstracción de reenvío correspondencia-acción y sus controladores, así como en lo que respecta a la revolución SDN en general [Feamster 2013]. Nos centraremos principalmente en OpenFlow 1.0, que introdujo importantes abstracciones y funcionalidad SDN

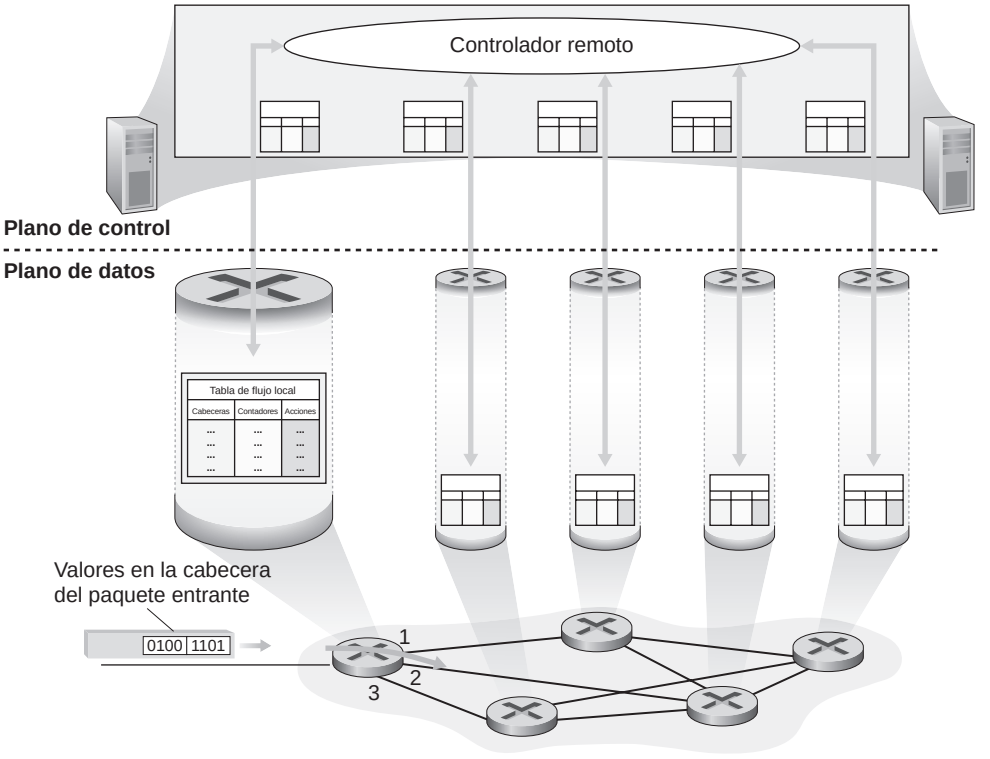


Figura 4.28 ♦ Reenvío generalizado: cada conmutador de paquetes contiene una tabla de correspondencia-acción que es calculada y distribuida por un controlador remoto.

de una forma particularmente clara y concisa. Versiones posteriores de OpenFlow introdujeron capacidades adicionales como resultado de la experiencia ganada con la implementación y el uso; puede encontrar las versiones actual y anteriores del estándar OpenFlow en [ONF 2016].

Cada entrada de la tabla de reenvío correspondencia-acción, que se conoce con el nombre de **tabla de flujo** en OpenFlow, incluye:

- *Un conjunto de valores de campos de cabecera* con los que se buscará una correspondencia en el paquete entrante. Al igual que en el caso del reenvío basado en el destino, la búsqueda de correspondencias basada en hardware puede llevarse a cabo con la máxima velocidad usando memorias TCAM, siendo posible disponer de más de un millón de entradas de dirección de destino [Bosshart 2013]. Un paquete que no se corresponda con ninguna entrada de la tabla de flujo puede ser eliminado o enviado al controlador remoto para un procesamiento adicional. En la práctica, una tabla de flujo puede ser implementada mediante múltiples tablas de flujo, por razones de coste o de prestaciones [Bosshart 2013], pero aquí vamos a centrarnos en la abstracción de una única tabla de flujo.
- *Un conjunto de contadores* que se actualizan a medida que se encuentran correspondencias de paquetes con entradas de la tabla de flujo. Estos contadores podrían incluir el número de paquetes para los que se ha encontrado correspondencia con la entrada de la tabla y el tiempo transcurrido desde que la entrada de la tabla se actualizó por última vez
- *Un conjunto de acciones que hay que tomar* cuando un paquete se corresponde con una entrada de la tabla de flujo. Estas acciones podrían consistir en reenviar el paquete a un puerto de salida determinado, eliminar el paquete, hacer copias del paquete y enviarlas a múltiples puertos de salida y/o reescribir ciertos campos seleccionados de la cabecera.

Exploraremos las correspondencias y las acciones con más detalle en las secciones 4.4.1 y 4.4.2, respectivamente. Después, en la Sección 4.4.3, veremos el modo en que puede emplearse el conjunto de reglas de correspondencia de paquetes existentes en la red para implementar un amplio rango de funciones, incluyendo el enrutamiento, la conmutación de capa 2, los cortafuegos, el equilibrado de carga, las redes virtuales y otras. Para concluir, resaltemos que la tabla de flujo es, esencialmente, una API: la abstracción mediante la cual puede programarse el comportamiento de un conmutador de paquetes individual; veremos en la Sección 4.4.3 que se pueden programar de forma similar comportamientos de la red completa, programando/configurando apropiadamente estas tablas en un conjunto de conmutadores de paquetes de la red [Casado 2014].

4.4.1 Correspondencia

La Figura 4.29 muestra los once campos de cabecera de los paquetes, más la ID del puerto entrante, con los que puede buscarse una correspondencia en una regla correspondencia-acción de OpenFlow 1.0. Recuerde de la Sección 1.5.2 que una trama de la capa de enlace (capa 2) que llegue a un conmutador de paquetes, contendrá como carga útil un datagrama de la capa de red (capa 3), que a su vez contendrá, normalmente, un segmento de la capa de transporte (capa 4). La primera observación que conviene hacer es que la abstracción correspondencia de OpenFlow permite buscar una correspondencia con campos seleccionados de tres capas de cabeceras de protocolo (violando así descaradamente el principio de separación en capas que hemos estudiado en la Sección 1.5). Puesto que todavía no hemos estudiado la capa de enlace, baste decir que las direcciones MAC de origen y de destino mostradas en la Figura 4.29 son las direcciones de la capa de enlace asociadas con las interfaces emisora y receptora de la trama; al poder reenviar basándose en las direcciones Ethernet, en lugar de en las direcciones IP, vemos que un dispositivo compatible con OpenFlow puede comportarse tanto como un router (dispositivo de capa 3) que reenvía datagramas, cuanto como un switch (dispositivo de capa 2) que reenvía tramas. El campo del tipo Ethernet se corresponde con el protocolo de la capa superior (por ejemplo, IP) hacia el que se demultiplexará la carga útil de la trama, mientras que los campos VLAN están relacionados con las denominadas redes de área local virtuales, que estudiaremos en el Capítulo 6. El conjunto de doce valores con los que se pueden buscar correspondencias en la especificación OpenFlow 1.0 ha crecido hasta 41 valores en las especificaciones OpenFlow más recientes [Bosshart 2014].

El puerto de ingreso hace referencia al puerto de entrada del conmutador de paquetes a través del cual se recibe un paquete. De los campos de dirección IP de origen, dirección IP de destino, protocolo IP y tipo de servicio (TOS) IP ya hemos hablado anteriormente, en la Sección 4.3.1. También pueden buscarse correspondencias con los campos de número de puerto de origen y de destino de la capa de transporte.

Las entradas de la tabla de flujo también pueden tener caracteres comodín. Por ejemplo, una dirección IP de 128.119.*.* en una tabla de flujo se corresponderá con todo campo de dirección relevante de cualquier datagrama que tenga como primeros 16 bits de la dirección el valor 128.119. Cada entrada de la tabla de flujo tiene también asociada una prioridad. Si un paquete se corresponde con múltiples entradas de la tabla de flujo, la correspondencia y la acción correspondiente seleccionadas serán las de la entrada de mayor prioridad con la que el paquete se corresponda.

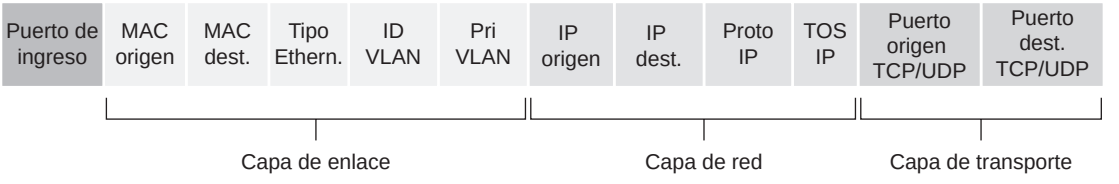


Figura 4.29 ♦ Campos de correspondencia de los paquetes en la tabla de flujo de OpenFlow 1.0.

Por último, observe que no pueden establecerse correspondencias con todos los campos de una cabecera IP. Por ejemplo, OpenFlow no permite buscar correspondencias con el campo TTL ni con el campo de longitud del datagrama. ¿Por qué se permite buscar correspondencias con algunos campos y con otros no? Indudablemente, la respuesta tiene que ver con el compromiso entre funcionalidad y complejidad. El "arte" a la hora de seleccionar una abstracción consiste en proporcionar la suficiente funcionalidad como para llevar a cabo una tarea (en este caso, implementar, configurar y administrar una amplia gama de funciones de la capa de red, que antes se implementaban mediante una variedad de dispositivos de la capa red), pero sin sobrecargar la abstracción con tantos detalles y tanta generalidad que la abstracción se expanda demasiado y resulte inutilizable. Como dice la famosa frase de Butler Lampson [Lampson 1983]:

> *Haz una cosa cada vez y hazla bien. Una interfaz debe capturar los detalles mínimos esenciales de una abstracción. No generalices; las generalizaciones son generalmente erróneas.*

Dado el éxito de OpenFlow, podemos dar por supuesto que sus diseñadores seleccionaron bien su abstracción. Puede encontrar detalles adicionales sobre las correspondencias en OpenFlow en [OpenFlow 2009, ONF 2016].

4.4.2 Acción

Como se muestra en la Figura 4.28, cada entrada de la tabla de flujo tiene una lista de cero o más acciones, que determinan el procesamiento que hay que aplicar a un paquete que se corresponda con dicha entrada. Si hay múltiples acciones, se ejecutan en el orden especificado en la lista.

Las más importantes de las acciones posibles son:

- *Reenvío.* Un paquete entrante puede ser reenviado a un puerto físico de salida concreto; puede ser difundido a través de todos los puertos (excepto aquél por el que haya llegado) o puede ser enviado a través de un conjunto seleccionado de puertos. El paquete también puede ser encapsulado y enviado al controlado remoto correspondiente a este dispositivo. El controlador puede entonces (o no) llevar a cabo alguna acción sobre dicho paquete, incluyendo la instalación de nuevas entradas de la tabla de flujo, y puede devolver el paquete al dispositivo, para que lo reenvíe de acuerdo con el conjunto de reglas actualizado de la tabla de flujo.
- *Eliminación.* Una entrada de la tabla de flujo sin ninguna acción indica que un paquete que se corresponda con ella debe ser eliminado.
- *Modificación de campos.* Los valores de diez campos de cabecera del paquete (todos los campos de las capas 2, 3 y 4 mostrados en la Figura 4.29, con excepción del campo Protocolo IP) pueden ser reescritos antes de reenviar el paquete al puerto de salida elegido.

4.4.3 Ejemplos de correspondencia-acción en OpenFlow

Habiendo repasado los componentes de correspondencia y acción del reenvío generalizado, juntemos estos conceptos en el contexto de la red de ejemplo mostrada en la Figura 4.30. La red tiene 6 hosts (h1, h2, h3, h4, h5 y h6) y tres conmutadores de paquetes (s1, s2 y s3), con cuatro interfaces locales cada uno (numeradas de 1 a 4). Vamos a considerar diversos comportamientos de la red que nos gustaría implementar, así como las entradas de tabla de flujo que s1, s2 y s3 necesitarían para implementar cada uno de esos comportamientos.

Un primer ejemplo: reenvío simple

Como ejemplo sencillo, supongamos que el comportamiento de reenvío deseado consiste en que los paquetes de h5 o h6 destinados a h3 o h4 deben reenviarse de s3 a s1 y luego de s1 a s2 (evitando así completamente el uso del enlace existente entre s3 y s2). La entrada de la tabla de flujo en s1 sería:

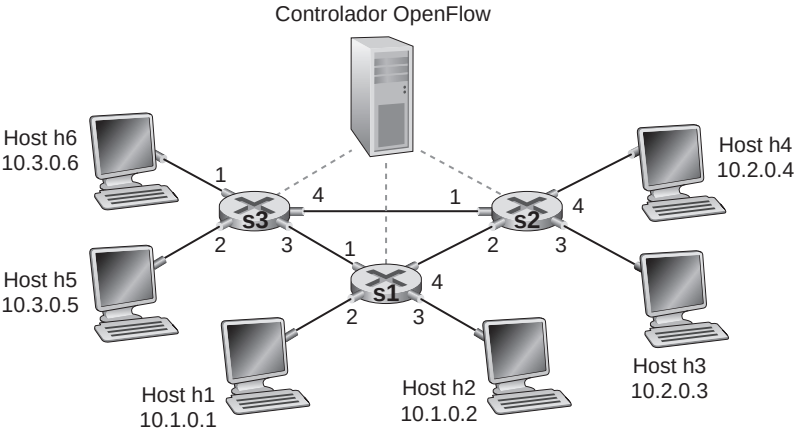


Figura 4.30 ♦ Red correspondencia-acción de OpenFlow con tres conmutadores de paquetes, 6 hosts y un controlador OpenFlow.

Tabla de flujo de s1 (Ejemplo 1)	
Correspondencia	Acción
Puerto de ingreso = 1 ; Origen IP = 10.3.*.* ; Destino IP = 10.2.*.*	Reenvío(4)
…	…

Por supuesto, necesitaremos también una entrada de tabla de flujo en s3, de modo que los datagramas enviados desde h5 o h6 se reenvíen a s1 a través de la interfaz saliente 3:

Tabla de flujo de s3 (Ejemplo 1)	
Correspondencia	Acción
Origen IP = 10.3.*.* ; Destino IP = 10.2.*.*	Reenvío(3)
…	…

Por último, también necesitamos una entrada de tabla de flujo en s2 para completar este primer ejemplo, de modo que los datagramas que lleguen desde s1 se reenvíen a su destino, el host h3 o h4:

Tabla de flujo de s2 (Ejemplo 1)	
Correspondencia	Acción
Puerto de ingreso = 2 ; Destino IP = 10.2.0.3	Reenvío(3)
Puerto de ingreso = 2 ; Destino IP = 10.2.0.4	Reenvío(4)
…	…

Un segundo ejemplo: equilibrado de carga

Como segundo ejemplo, consideremos un escenario de equilibrado de carga, en el que los datagramas de h3 destinados a 10.1.*.* deben reenviarse a través del enlace directo entre s2 y s1, mientras que los datagramas de h4 destinados a 10.1.*.* deben reenviarse a través del enlace entre s2 y s3 (y luego de s3 a s1). Observe que este comportamiento no podría conseguirse mediante el reenvío basado en el destino IP. En este caso, la tabla de flujo en s2 sería:

Tabla de flujo de s2 (Ejemplo 2)	
Correspondencia	Acción
Puerto de ingreso = 3; Destino IP = 10.1.*.*	Reenvío(2)
Puerto de ingreso = 4; Destino IP = 10.1.*.*	Reenvío(1)
…	…

También hacen falta entradas de tabla de flujo en s1 para reenviar a h1 o h2 los datagramas recibidos de s2; y hacen falta entradas de tabla de flujo en s3 para reenviar hacia s1, a través de la interfaz 3, los datagramas recibidos de s2 a través de la interfaz 4. Trate de deducir cuáles deberían ser esas entradas de tabla de flujo en s1 y s3.

Un tercer ejemplo: cortafuegos

Como tercer ejemplo, consideremos un caso de cortafuegos en el que s2 solo quiere recibir (a través de cualquiera de sus interfaces) el tráfico enviado por los hosts conectados a s3.

Tabla de flujo de s2 (Ejemplo 3)	
Correspondencia	Acción
Origen IP = 10.3.*.* Destino IP = 10.2.0.3	Reenvío(3)
Origen IP = 10.3.*.* Destino IP = 10.2.0.4	Reenvío(4)
…	…

Si no hubiera ninguna otra entrada en la tabla de flujo de s2, entonces solo el tráfico procedente de 10.3.*.* sería reenviado a los hosts conectados a s2.

Aunque solo hemos considerado aquí unos pocos escenarios básicos, esperamos que hayan quedado claras la versatilidad y las ventajas del reenvío generalizado. En los problemas del capítulo, exploraremos cómo pueden usarse las tablas de flujo para crear muchos comportamientos lógicos distintos, incluyendo redes virtuales —dos o más redes separadas lógicamente (cada una con su propio comportamiento de reenvío distinto e independiente)— que usen el *mismo* conjunto físico de conmutadores de paquetes y de enlaces. En la Sección 5.5, volveremos sobre las tablas de flujo cuando estudiemos los controladores SDN que las calculan y distribuyen, y el protocolo utilizado para la comunicación entre un conmutador de paquetes y su controlador.

4.5 Resumen

En este capítulo hemos estudiado las funciones del **plano de datos** de la capa de red —las funciones de cada router que determinan cómo se reenvían a uno de los enlaces de salida del router los paquetes recibidos por uno de los enlaces de entrada del mismo. Hemos comenzado examinando detalladamente el funcionamiento interno del router, estudiando la funcionalidad de los puertos de entrada y de salida, el reenvío basado en el destino, el mecanismo interno de conmutación de un router y la gestión de las colas de paquetes, entre otros temas. Hemos analizado tanto del reenvío IP tradicional (en el que el reenvío se basa en la dirección de destino de un datagrama), como del reenvío generalizado (en el que pueden llevarse a cabo el reenvío y otras funciones dependiendo de los valores de varios campos distintos de la cabecera del datagrama), y hemos podido comprobar la versatilidad de esta última técnica. También hemos estudiado en detalle los protocolos IPv4 e IPv6, así como el tema del direccionamiento en Internet, que ha resultado ser mucho más profundo, sutil e interesante de lo que cabría esperar.

Armados con nuestros nuevos conocimientos acerca del plano de datos de la capa de red, estamos ya listos para sumergirnos en el plano de control de dicha capa. ¡Pero eso será en el Capítulo 5!

Problemas y cuestiones de repaso

Capítulo 4 Cuestiones de repaso

SECCIÓN 4.1

R1. Revisemos parte de la terminología utilizada en el libro. Recuerde que el nombre que recibe un paquete de la capa de transporte es *segmento* y que el nombre de un paquete de la capa de enlace es *trama*. ¿Cuál es el nombre de un paquete de la capa de red? Recuerde que tanto los routers como los switches de la capa de enlace se denominan *conmutadores de paquetes*. ¿Cuál es la diferencia fundamental entre un router y un switch de la capa de enlace?

R2. Ya hemos indicado que la funcionalidad de la capa de red puede dividirse, en términos generales, en funcionalidad del plano de datos y funcionalidad del plano de control. ¿Cuáles son las principales funciones del plano de datos? ¿Y las del plano de control?

R3. Hemos hecho la distinción entre la función de reenvío y la función de enrutamiento realizadas en la capa de red. ¿Cuáles son las diferencias fundamentales entre el enrutamiento y el reenvío?

R4. ¿Cuál es el papel de la tabla de reenvío dentro de un router?

R5. Hemos dicho que el modelo de servicio de una capa de red "define las características del transporte extremo a extremo de paquetes entre los hosts emisor y receptor". ¿Cuál es el modelo de servicio de la capa de red de Internet? ¿Qué garantías proporciona el modelo de servicio de Internet, en lo que respecta a la entrega de datagramas de un host a otro host?

SECCIÓN 4.2

R6. En la Sección 4.2 hemos visto que un router está compuesto, normalmente, por puertos de entrada, puertos de salida, un entramado de conmutación y un procesador de enrutamiento. ¿Cuáles de estos elementos se implementan en hardware y cuáles en software? ¿Por qué? Volviendo a las nociones de plano de datos y plano de control de la capa de red, ¿cuál de ellos se implementa en hardware y cuál en software? ¿Por qué?

R7. Explique por qué cada puerto de entrada de un router de alta velocidad almacena una copia de la tabla de reenvío.

R8. ¿Qué se entiende por reenvío basado en el destino? ¿En qué difiere del reenvío generalizado? Tras leer la Sección 4.4, ¿cuál de esas dos técnicas es la adoptada por la tecnología SDN (redes definidas por software)?

R9. Suponga que un paquete entrante se corresponde con dos o más entradas de la tabla de reenvío de un router. Con el reenvío tradicional basado en el destino, ¿qué norma aplica un router para determinar cuál de las dos reglas hay que invocar a la hora de determinar el puerto de salida hacia el que debe ser conmutado el paquete entrante?

R10. En la Sección 4.3 se han abordado tres tipos de entramados de conmutación. Enumere y describa brevemente cada uno de ellos. ¿Cuál de ellos, si es que hay alguno, permite enviar múltiples paquetes en paralelo a través del entramado?

R11. Describa cómo pueden perderse paquetes en los puertos de entrada. Describa cómo puede eliminarse la pérdida de paquetes en los puertos de entrada (sin utilizar buffers de capacidad infinita).

R12. Describa cómo puede producirse una pérdida de paquetes en los puertos de salida. ¿Puede evitarse esta pérdida incrementando la velocidad del entramado de conmutación?

R13. ¿Qué es el bloqueo HOL? ¿Se produce en los puertos de entrada o en los puertos de salida?

R14. En la Sección 4.2 hemos estudiado las siguientes disciplinas de planificación de paquetes: FIFO, prioridad, *Round Robin* (RR) y cola equitativa ponderada (WFQ, *Weighted Fair*

Queueing). ¿Cuáles de estas disciplinas de colas garantizan que todos los paquetes salgan en el mismo orden en que llegaron?

R15. Proporcione un ejemplo que muestre por qué un operador de red podría querer dar prioridad a una clase de paquetes con respecto a otra clase de paquetes.

R16. ¿Qué diferencia esencial hay entre las planificaciones de paquetes RR y WFQ? ¿Hay algún caso (*Sugerencia*: piense en las ponderaciones WFQ) en el que RR y WFQ se comporten exactamente igual?

SECCIÓN 4.3

R17. Suponga que el host A envía al host B un segmento TCP encapsulado en un datagrama IP. Cuando el host B recibe el datagrama, ¿cómo sabe la capa de red del host B que debería pasar el segmento (es decir, la carga útil del datagrama) a TCP, en lugar de a UDP o a cualquier otro protocolo?

R18. ¿Qué campo de la cabecera IP puede usarse para garantizar que un paquete se reenvíe a través de no más de *N* routers?

R19. Recuerde que, como hemos visto, la suma de comprobación Internet se utiliza tanto en los segmentos de la capa de transporte (en las cabeceras UDP y TCP; Figuras 3.7 y 3.29, respectivamente), como en los datagramas de la capa de red (cabecera IP; Figura 4.16). Considere ahora un segmento de la capa de transporte encapsulado en un datagrama IP. ¿Existen bytes comunes en el datagrama IP sobre los cuales se calculen las sumas de comprobación de la cabecera del segmento y de la cabecera del datagrama? Explique su respuesta.

R20. Cuando se fragmenta un datagrama de gran tamaño en múltiples datagramas más pequeños, ¿dónde se reensamblan estos datagramas más pequeños para obtener un único datagrama de mayor tamaño?

R21. ¿Tienen direcciones IP los routers? En caso afirmativo, ¿cuántas?

R22. ¿Cuál es el equivalente binario de 32 bits de la dirección IP 223.1.3.27?

R23. Visite un host que utilice DHCP para obtener su dirección IP, su máscara de red, su router predeterminado y la dirección IP de su servidor DNS local. Enumere estos valores.

R24. Suponga que hay tres routers entre un host de origen y un host de destino. Ignorando la fragmentación, se envía un datagrama IP desde el host de origen al host de destino. ¿A través de cuántas interfaces pasará? ¿Cuántas tablas de reenvío habrá que consultar para transportar el datagrama desde el origen al destino?

R25. Suponga que una aplicación genera fragmentos de 40 bytes de datos cada 20 milisegundos y que cada fragmento se encapsula en un segmento TCP y luego en un datagrama IP. ¿Qué porcentaje de cada datagrama será información administrativa y qué porcentaje será datos de aplicación?

R26. Suponga que adquiere un router inalámbrico y que lo conecta a su módem por cable. Suponga también que su ISP asigna dinámicamente una dirección IP a su dispositivo conectado (es decir, a su router inalámbrico). Además, suponga que tiene cinco equipos PC en su domicilio que utilizan 802.11 para conectarse de forma inalámbrica a su router inalámbrico. ¿Cómo se asignan las direcciones IP a los cinco PC? ¿Utiliza NAT el router inalámbrico? ¿Por qué o por qué no?

R27. ¿Qué quiere decir el término “agregación de rutas”? ¿Por qué resulta útil para un router realizar la agregación de rutas?

R28. ¿Qué es un protocolo “plug-and-play” o de configuración cero (*zeroconf*)?

R29. ¿Qué es una dirección de red privada? ¿Debe haber datagramas con direcciones de red privadas en la red Internet pública? Explique su respuesta.

R30. Compare y contraste los campos de cabecera de IPv4 e IPv6. ¿Tienen campos en común?

R31. A veces se dice que cuando IPv6 se tuneliza a través de routers IPv4, IPv6 trata los túneles IPv4 como protocolos de la capa de enlace. ¿Está de acuerdo con esta afirmación? ¿Por qué o por qué no?

SECCIÓN 4.4

R32. ¿En qué se diferencia el reenvío generalizado del reenvío basado en el destino?

R33. ¿Cuál es la diferencia entre las tablas de reenvío basado en el destino, con las que nos hemos encontrado en la Sección 4.1, y la tabla de flujo de OpenFlow de las que hemos hablado en la Sección 4.4?

R34. ¿Qué queremos decir al hablar de la operación "correspondencia más acción" de un router o un switch? En el caso de un conmutador de paquetes con reenvío basado en el destino, ¿qué correspondencia se busca y cuál es la acción realizada? En el caso de una SDN, enumere tres campos con los que puedan buscarse correspondencias y tres acciones que puedan tomarse.

R35. Enumere tres campos de la cabecera de un datagrama IP con los que pueda buscarse una correspondencia en el reenvío generalizado compatible con OpenFlow 1.0. Indique, asimismo, tres campos de cabecera de un datagrama IP con los que *no pueda* buscarse una correspondencia en OpenFlow.

Problemas

P1. Utilice la red mostrada en la parte inferior de la página.

a. Especifique la tabla de reenvío del router A, de modo que todo el tráfico destinado al host H3 sea reenviado a través de la interfaz 3.

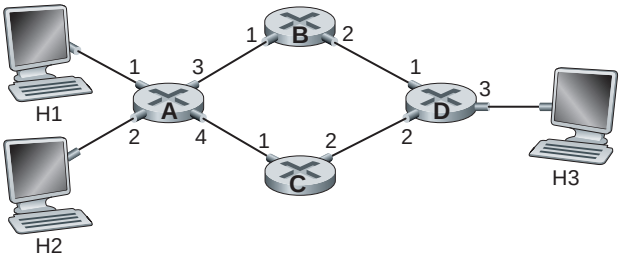
b. .¿Puede escribir una tabla de reenvío para el router A, de manera que todo el tráfico de H1 destinado al host H3 sea reenviado a través de la interfaz 3, mientras todo el tráfico de H2 destinado al host H3 sea reenviado a través de la interfaz 4? (*Sugerencia*: esta pregunta tiene truco.)

P2. Suponga que dos paquetes llegan a dos puertos de entrada distintos de un router exactamente al mismo tiempo. Suponga también que no hay ningún otro paquete en ninguna parte del router.

a. Suponga que los dos paquetes deben ser reenviados a dos puertos de salida diferentes. ¿Es posible reenviar los dos paquetes a través del entramado de conmutación al mismo tiempo, cuando el entramado utiliza un bus compartido?

b. Suponga que los dos paquetes deben ser reenviados a dos puertos de salida diferentes. ¿Es posible reenviar los dos paquetes a través del entramado de conmutación al mismo tiempo, cuando el entramado utiliza la conmutación vía memoria?

c. Suponga que los dos paquetes deben ser reenviados al mismo puerto de salida. ¿Es posible reenviar los dos paquetes a través del entramado de conmutación al mismo tiempo, cuando el entramado utiliza una malla?



P3. En la Sección 4.2 dijimos que el retardo máximo de cola es $(n\text{-}1)D$ si el entramado de conmutación es n veces más rápido que las velocidades de las líneas de entrada. Suponga que todos los paquetes tienen la misma longitud, que n paquetes llegan simultáneamente a los n puertos de entrada y que los n paquetes desean ser reenviados a diferentes puertos de salida. ¿Cuál será el retardo máximo de un paquete para los entramados de conmutación (a) de memoria, (b) de bus y (c) de malla?

P4. Considere el dispositivo de conmutación mostrado en la parte inferior de la página. Suponga que todos los datagramas tienen la misma longitud fija, que el dispositivo opera de forma síncrona y particionada y que en una partición temporal se puede transferir un datagrama desde un puerto de entrada a un puerto de salida. El entramado de conmutación emplea una estructura de malla, por lo que, como máximo, se puede transferir un datagrama a un puerto de salida determinado en una partición de tiempo, pero distintos puertos de salida pueden recibir datagramas procedentes de distintos puertos de entrada en una misma partición de tiempo. ¿Cuál es el número mínimo de particiones de tiempo necesarias para transferir los paquetes mostrados desde los puertos de entrada a sus puertos de salida, suponiendo cualquier orden de planificación de la cola de entrada que desee (es decir, no tiene por qué existir el bloqueo HOL)? ¿Cuál es el número máximo necesario de particiones de tiempo, suponiendo el orden de planificación de caso peor que pueda imaginar y suponiendo que una cola de entrada no vacía nunca está inactiva?

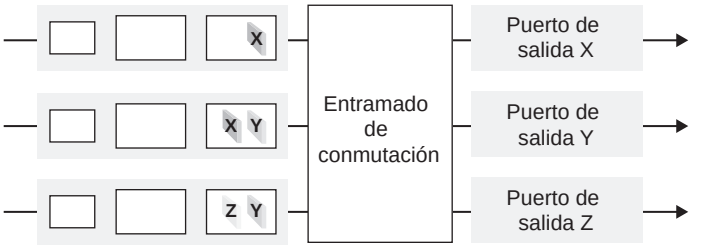
P5. Considere una red de datagramas que utiliza direcciones de host de 32 bits. Suponga que un router tiene cuatro enlaces, numerados de 0 a 3, y que los paquetes deben ser reenviados a las interfaces de los enlaces como sigue:

Rango de direcciones de destino	**Interfaz de enlace**
11100000 00000000 00000000 00000000 hasta 11100000 00111111 11111111 11111111	0
11100000 01000000 00000000 00000000 hasta 11100000 01000000 11111111 11111111	1
11100000 01000001 00000000 00000000 hasta 11100001 01111111 11111111 11111111	2
en otro caso	3

a. Proporcione una tabla de reenvío con cinco entradas, que utilice la regla de coincidencia con el prefijo más largo y que reenvíe los paquetes a las interfaces de enlace correctas.

b. Describa cómo determina su tabla de reenvío la interfaz de enlace apropiada para los datagramas con las siguientes direcciones de destino:

11001000 10010001 01010001 01010101
11100001 01000000 11000011 00111100
11100001 10000000 00010001 01110111



P6. Considere una red de datagramas que utiliza direcciones de host de 8 bits. Suponga un router que utiliza las coincidencias con el prefijo más largo y cuya tabla de reenvío es la siguiente:

Coincidencia de prefijo	Interfaz
00	0
010	1
011	2
10	2
11	3

Para cada una de las cuatro interfaces, proporcione el rango asociado de direcciones del host de destino y el número de direcciones contenidas en el rango.

P7. Considere una red de datagramas que utiliza direcciones de host de 8 bits. Suponga un router que utiliza las coincidencias con el prefijo más largo y cuya tabla de reenvío es la siguiente:

Coincidencia de prefijo	Interfaz
1	0
10	1
111	2
En otro caso	3

Para cada una de las cuatro interfaces, proporcione el rango asociado de direcciones del host de destino y el número de direcciones contenidas en el rango.

P8. Sea un router que interconecta tres subredes: Subred 1, Subred 2 y Subred 3. Suponga que se requiere que todas las interfaces de cada una de estas tres subredes tengan el prefijo 223.1.17/24. Suponga también que se requiere que la Subred 1 admita al menos 6 interfaces, que la Subred 2 tiene que admitir al menos 90 interfaces y que la Subred 3 debe admitir hasta 12 interfaces. Determine tres direcciones de red (de la forma a.b.c.d/x) que satisfagan estas restricciones.

P9. En la Sección 4.2.2 se ha proporcionado una tabla de reenvío de ejemplo (utilizando la coincidencia con el prefijo más largo). Vuelva a escribir esta tabla de reenvío utilizando la notación a.b.c.d/x en lugar de la notación en binario.

P10. En el Problema P5 se le ha pedido que proporcione una tabla de reenvío (utilizando la regla de coincidencia con el prefijo más largo). Escriba de nuevo esta tabla de reenvío utilizando la notación a.b.c.d/x en lugar de la notación en binario.

P11. Considere un subred cuyo prefijo es 128.119.40.128/26. Proporcione un ejemplo de una dirección IP (de la forma xxx.xxx.xxx.xxx) que pueda ser asignada a esta red. Suponga que un ISP posee el bloque de direcciones 128.119.40.64/26. Suponga que desea crear cuatro subredes a partir de este bloque de direcciones, teniendo cada bloque el mismo número de direcciones IP. ¿Cuáles serán los prefijos (expresados en formato a.b.c.d/x) para las cuatro subredes?

P12. Considere la topología mostrada en la Figura 4.20. Denomine a las tres subredes con hosts (comenzando en el sentido horario a partir de las 12:00) como redes A, B y C. Denomine a las subredes que no tienen hosts como redes D, E y F.

a. Asigne direcciones de red a cada una de estas seis subredes, teniendo en cuenta las siguientes restricciones: todas las direcciones tienen que ser asignadas a partir de 214.97.254/23;

la subred A tendrá que disponer de las direcciones suficientes como para dar soporte a 250 interfaces; la subred B tendrá que disponer de las direcciones suficientes como para dar soporte a 120 interfaces y la subred C tendrá que disponer de las direcciones suficientes como para dar soporte a 120 interfaces. Por supuesto, las subredes D, E y F deberían poder dar soporte a dos interfaces. Para cada subred, la asignación debería hacerse empleando el formato a.b.c.d/x o a.b.c.d/x – e.f.g.h/y.

b. Utilizando su respuesta al apartado (a), proporcione las tablas de reenvío (utilizando la regla de coincidencia con el prefijo más largo) para cada uno de los tres routers.

P13. Utilice el servicio whois del ARIN —American Registry for Internet Numbers— (http://www.arin.net/whois) - para determinar los bloques de direcciones IP de tres universidades. ¿Pueden usarse los servicios de whois para determinar con certidumbre la ubicación geográfica de una dirección IP específica? Utilice www.maxmind.com para determinar las ubicaciones de los servidores web de cada una de esas universidades.

P14. Se envía un datagrama de 2.400 bytes por un enlace que tiene una MTU de 700 bytes. Suponga que el datagrama original está marcado con el número de identificación 422. ¿Cuántos fragmentos se generan? ¿Cuáles son los valores de los distintos campos relacionados con la fragmentación, en los datagramas IP generados?

P15. Suponga que el tamaño de los datagramas está limitado a 1.500 bytes (incluyendo la cabecera) entre el host A y el host de destino B. Suponiendo una cabecera IP de 20 bytes, ¿cuántos datagramas se necesitarían para enviar un archivo MP3 de 5 millones de bytes? Explique los cálculos que haya realizado para dar una respuesta.

P16. Considere la red de la Figura 4.25. Suponga que el ISP asigna al router la dirección 24.34.112.235, en lugar de la que se muestra en la figura, y que la dirección de la red doméstica es 192.168.1/24.

a. Asigne direcciones a todas las interfaces de la red doméstica.

b. Suponga que cada host tiene dos conexiones TCP activas, todas ellas con el puerto 80 del host 128.119.40.86. Proporcione las seis entradas correspondientes de la tabla de traducciones NAT.

P17. Suponga que está interesado en detectar el número de hosts que hay detrás de un traductor NAT y que observa que la capa IP marca secuencialmente con un número de identificación cada paquete IP. El número de identificación del primer paquete IP generado por un host es un número aleatorio y los números de identificación de los subsiguientes paquetes IP se asignan de forma secuencial. Suponga que todos los paquetes IP generados por los hosts que hay detrás del traductor NAT se envían al exterior.

a. Basándose en esta observación y suponiendo que puede husmear todos los paquetes enviados por el traductor NAT al exterior, ¿puede esbozar una técnica sencilla para detectar el número de hosts diferentes que hay detrás del traductor NAT? Justifique su respuesta.

b. Si los números de identificación no se asignan secuencialmente, sino aleatoriamente, ¿serviría su técnica? Justifique su respuesta.

P18. En este problema se explora el impacto de NAT sobre las aplicaciones P2P. Suponga que un par cuyo nombre de usuario es Arnold descubre mediante una consulta que otro par con el nombre de usuario Bernard tiene un archivo que desea descargar. Suponga también que Bernard y Arnold están detrás de un traductor NAT. Intente deducir una técnica que permita a Arnold establecer una conexión TCP con Bernard sin realizar una configuración NAT específica de la aplicación. Si tiene dificultades para definir una técnica así, explique por qué.

P19. Considere la red SDN OpenFlow mostrada en la Figura 4.30. Suponga que el comportamiento de reenvío deseado para los datagramas que llegan a s2 es el siguiente:

- Los datagramas que lleguen de los hosts h5 o h6 a través del puerto de entrada 1 y que estén destinados a los hosts h1 o h2, deben reenviarse a través del puerto de salida 2.

- Los datagramas que lleguen de los hosts h1 o h2 a través del puerto de entrada 2 y que estén destinados a los hosts h5 o h6, deben reenviarse a través del puerto de salida 1.
- Los datagramas que lleguen a través de los puertos de entrada 1 o 2 y que estén destinados a los hosts h3 o h4, deben entregarse al host especificado.
- Los hosts h3 y h4 deben poder intercambiarse datagramas.

Especifique las entradas de la tabla de flujo de s2 que permiten implementar este comportamiento de reenvío.

P20. Considere de nuevo la red SDN OpenFlow mostrada en la Figura 4.30. Suponga que el comportamiento de reenvío deseado para los datagramas que llegan a s2 desde los hosts h3 o h4 es el siguiente:

- Los datagramas que lleguen del host h3 y que estén destinados a los hosts h1, h2, h5 o h6, deben reenviarse través de la red en el sentido de las agujas del reloj.
- Los datagramas que lleguen del host h4 y que estén destinados a los hosts h1, h2, h5 o h6, deben reenviarse través de la red en el sentido contrario a las agujas del reloj.

Especifique las entradas de la tabla de flujo de s2 que permiten implementar este comportamiento de reenvío.

P21. Considere de nuevo el escenario del Problema P19. Especifique las entradas de las tablas de flujo de los conmutadores de paquetes s1 y s3, de modo que los datagramas entrantes con una dirección de origen igual a h3 o h4 se enruten hacia los hosts de destino especificados en el campo de dirección de destino del datagrama IP. (*Sugerencia*: Las reglas de sus tablas de reenvío deben incluir tanto el caso de que un datagrama entrante esté destinado a un host directamente conectado, como el caso de que haya que reenviarlo a un router vecino para su eventual entrega allí a un host.)

P22. Considere de nuevo la red SDN OpenFlow mostrada en la Figura 4.30. Suponga que queremos que el conmutador s2 funcione como un cortafuegos. Especifique la tabla de flujo de s2 que implemente los siguientes comportamientos de cortafuegos para la entrega de datagramas destinados a los hosts h3 y h4 (proporcione una tabla de flujo diferente para cada uno de los cuatro comportamientos de cortafuegos indicados). No hace falta que especifique el comportamiento de reenvío de s2 que se encarga de reenviar tráfico hacia otros routers.

- Solo el tráfico que llega de los hosts h1 y h6 debe entregarse a los hosts h3 o h4 (es decir, se bloquea el tráfico que llegue de los hosts h2 y h5).
- Solo se permite entregar a los hosts h3 o h4 el tráfico TCP (es decir, se bloquea el tráfico UDP).
- Solo hay que entregar el tráfico destinado a h3 (es decir, se bloquea todo el tráfico destinado a h4).
- Solo hay que entregar el tráfico UDP de h1 destinado a h3. Todo el tráfico restante se bloquea.

Práctica de laboratorio con Wireshark

En el sitio web del libro, http://www.pearsonhighered.com/cs-recources, encontrará una práctica de laboratorio con Wireshark que examina el funcionamiento del protocolo IP y el formato del datagrama IP en concreto.

UNA ENTREVISTA CON...

Vinton G. Cerf

Vinton G. Cerf es Vicepresidente y Jefe de Estrategia de Internet de Google. Ha trabajado durante más de 16 años en MCI en distintos puestos, acabando allí su carrera profesional como Vicepresidente senior de Estrategia Tecnológica. Es muy conocido por haber sido co-diseñador de los protocolos TCP/IP y de la arquitectura de Internet. Durante el tiempo que trabajó, entre 1976 y 1982, en la Agencia de Proyectos de Investigación Avanzada de Defensa (DARPA, Department of Defense Advanced Research Projects Agency) de Estados Unidos, desempeñó un papel crucial dirigiendo el desarrollo de Internet y de tecnologías de seguridad y de empaquetado de datos relacionadas con Internet. Recibió la Medalla Presidencial de la Libertad en 2005 y la Medalla Nacional de Tecnología en 1997. Es licenciado en Matemáticas por la Universidad de Stanford y doctor en Ciencias de la Computación por UCLA.

¿Qué le hizo especializarse en el campo de las redes?

Estaba trabajando como programador en UCLA a finales de la década de 1960. Mi trabajo estaba financiado por la Agencia de Proyectos de Investigación Avanzada de Defensa de Estados Unidos (que entonces se llamada ARPA y ahora DARPA). Trabajaba en el laboratorio del profesor Leonard Kleinrock en el Centro de Medidas de Red para la recientemente creada red ARPAnet. El primer nodo de ARPAnet fue instalado en UCLA el 1 de septiembre de 1969. Yo era responsable de programar una computadora que se utilizaba para capturar información de rendimiento acerca de ARPAnet y de devolver esa información para compararla con los modelos matemáticos y las predicciones relativas al rendimiento de la red.

A algunos otros estudiantes y a mí mismo nos encargaron trabajar en los denominados protocolos de nivel de host de ARPAnet, es decir, en los procedimientos y formatos que permitirían que muchos tipos distintos de computadoras interactuaran entre sí a través de la red. Fue una investigación fascinante en un mundo nuevo (para mí) de comunicación y computación distribuidas.

Cuando diseñó el protocolo, ¿se imaginaba que IP llegaría a ser tan ubicuo como lo es hoy día?

Cuando Bob Kahn y yo comenzamos a trabajar en el tema en 1973, creo que estábamos muy concentrados en el problema fundamental: cómo hacer que una serie de redes de paquetes heterogéneas interoperaran, partiendo del supuesto de que no podíamos modificar las propias redes. Intentábamos encontrar una forma de poder interconectar un conjunto arbitrario de redes de conmutación de paquetes de manera transparente, de modo que las computadoras host pudieran comunicarse extremo a extremo sin necesidad de realizar ninguna traducción intermedia. Creo que éramos conscientes de que estábamos trabajando con una tecnología potente y ampliable, pero me parece que no teníamos una imagen clara de lo que podía llegar a ser nuestro mundo con centenares de millones de computadoras interconectadas a través de Internet.

¿Cuál cree que será el futuro de las redes y de Internet? ¿Cuáles cree que son los principales desafíos/ obstáculos a los que se enfrentarán?

Creo que la propia Internet y las redes en general continuarán expandiéndose. Ya existen suficientes evidencias para afirmar que pronto habrá miles de millones de dispositivos compatibles con Internet, incluyendo equipos tales como teléfonos móviles, frigoríficos, asistentes digitales personales (PDA), servidores domésticos, televisiones, además de la colección habitual de computadoras portátiles, servidores, etc. Entre los principales desafíos se incluyen el soporte para la movilidad, la capacidad de las baterías, la capacidad de los enlaces de acceso a la red y la capacidad de ampliar el núcleo óptico de la red de forma ilimitada. El diseño de una extensión interplanetaria de Internet es un proyecto en el que estoy profundamente involucrado en el Jet Propulsion Laboratory. Asimismo, necesitaremos hacer la transición desde IPv4 [direcciones de 32 bits] a IPv6 [direcciones de 128 bits]. ¡Hay una larga lista de desafíos!

¿Quién le ha servido de inspiración profesionalmente?

Mi colega Bob Kahn; mi director de tesis, Gerald Estrin; mi mejor amigo, Steve Crocker (¡nos conocimos en la facultad y fue él el que me introdujo en el mundo de las computadoras en 1960!); y los miles de ingenieros que hacen que Internet continúe evolucionando hoy día.

¿Qué consejo le daría a los estudiantes que inician su andadura en el campo de las redes/ Internet?

Que tienen que pensar sin tener en cuenta las limitaciones de los sistemas existentes. Deben tratar de imaginar qué cosas son posibles y a continuación hacer el trabajo duro de intentar averiguar cómo ir hasta allí desde el estado actual de las cosas. Es preciso ser capaz de soñar: media docena de colegas y yo hemos estado trabajando en el Jet Propulsion Laboratory en el diseño de una extensión interplanetaria de la Internet terrestre. Puede que se tarden décadas en implementar esto, misión a misión, pero parafraseando un conocido dicho: “El hombre debe poder llegar más allá de donde su mano alcanza; ¿o para qué son los cielos, si no?”.