
CAPÍTULO 1

Conceptos básicos de TCP/IP

Contenidos

- » Arquitectura TCP/IP, modelo Cliente/Servidor y servicios de red.
- » Capa de red en TCP/IP - Protocolo IP.
- » Capa de transporte en TCP/IP - Protocolos TCP y UDP.
- » Traducción de direcciones de red - NAT.
- » Uso de las máquinas virtuales.

Objetivos

- » Reconocer la estructura de las redes identificando sus elementos y principios de funcionamiento.
- » Realizar tareas de administración de la red.
- » Configurar equipos y redes locales con software de virtualización.

Este capítulo presenta de forma sintetizada los conocimientos previos necesarios para afrontar el estudio de los servicios de red en la arquitectura TCP/IP.

1.1. Introducción

En este capítulo inicial se pretende ofrecer, necesariamente de una forma rápida y esquemática, un resumen de los conocimientos previos mínimos imprescindibles para abordar el resto del libro con garantías. No es por tanto una exposición completa ni exhaustiva y debe tomarse como simple referencia de los conocimientos más importantes que se darán por supuesto de ahora en adelante.

Dado que se trata de un libro orientado fundamentalmente a cubrir las enseñanzas del módulo “**Servicios de Red e Internet**” perteneciente al segundo curso del Ciclo Formativo de Grado Superior “**Administración de Sistemas Informáticos y Redes**”¹, se presupone que el lector/alumno conoce en menor o mayor medida diversos temas relacionados con la estructura y funcionamiento interno de la arquitectura de comunicaciones TCP/IP, modelo cliente/servidor y en general, posee conocimientos suficientes para configurar y administrar cuanto menos una pequeña red de área local.

Todo ello debería ser posible para el lector/alumno si ha cursado con anterioridad el módulo “**Planificación y Administración de Redes**” del primer curso del Ciclo Formativo mencionado anteriormente. Si esto es así, este capítulo simplemente le servirá de recordatorio de lo ya aprendido y le concretará aquellos elementos que debe tener claros para el resto de los capítulos.

Si no se parte de esta base inicial, el lector deberá estudiar pausada y atentamente todo lo que en este capítulo se sintetiza con objeto de alcanzar las destrezas mínimas necesarias para entender lo que sigue.

Es importante destacar que, asimismo, en este capítulo se realizará la configuración inicial de la red virtual local basada en máquinas virtuales la cual se utilizará a lo largo de los siguientes capítulos para poner en práctica los conocimientos teóricos que vayan adquiriéndose.

1.2. La arquitectura TCP/IP, el modelo Cliente/Servidor y los servicios de red

Desde un principio la arquitectura TCP/IP ha estado orientada a funcionar en un entorno cliente/servidor, lo que ha facilitado enormemente la implantación de diversos servicios de red tanto en redes locales como en Internet.

1.2.1. La arquitectura TCP/IP y el modelo OSI

Durante muchos años se pensó que la **arquitectura OSI** (*Open Systems Interconnection*), más estructurada que la **arquitectura TCP/IP**, acabaría imponiéndose como estándar de hecho en el campo de las arquitecturas de comunicaciones. Esto nunca llegó a suceder motivado fundamentalmente por el hecho incuestionable de que, aunque OSI ofrecía un modelo muy bien diseñado, este modelo era completamente teórico. Mientras que TCP/IP, que era un modelo anterior a OSI, ya se encontraba en pleno funcionamiento cuando OSI surgió. La realidad es que las buenas intenciones de OSI nunca se vieron reflejadas en un modelo comercial y su utilidad actual se limita a servir de referencia como modelo de lo que debe ser toda arquitectura de comunicaciones.

¹Real Decreto 1629 / 2009 B.O.E.

La arquitectura TCP/IP nos proporciona una estructura y una serie de normas de funcionamiento para poder interconectar sistemas. La complejidad de esta tarea ha necesitado una subdivisión del trabajo en niveles o capas coordinadas de manera que, cada capa realiza una labor concreta y así la integración modular y jerárquica de todas ellas hace posible la comunicación.

En cada capa existen una serie de **protocolos** que ofrecen unas normas estrictas a seguir para el diálogo entre los sistemas. Cada protocolo se apoya en los protocolos de las capas inferiores para realizar su labor y, a su vez, ofrece sus servicios a las capas superiores. Esta es una de las características fundamentales de la arquitectura TCP/IP. Así, por ejemplo, tenemos el protocolo IP en la capa de red, TCP y UDP en la capa de transporte, FTP y SNMP en la capa de aplicación, etc. En la Figura 1.1 se ilustra la equivalencia entre los niveles OSI y TCP/IP.

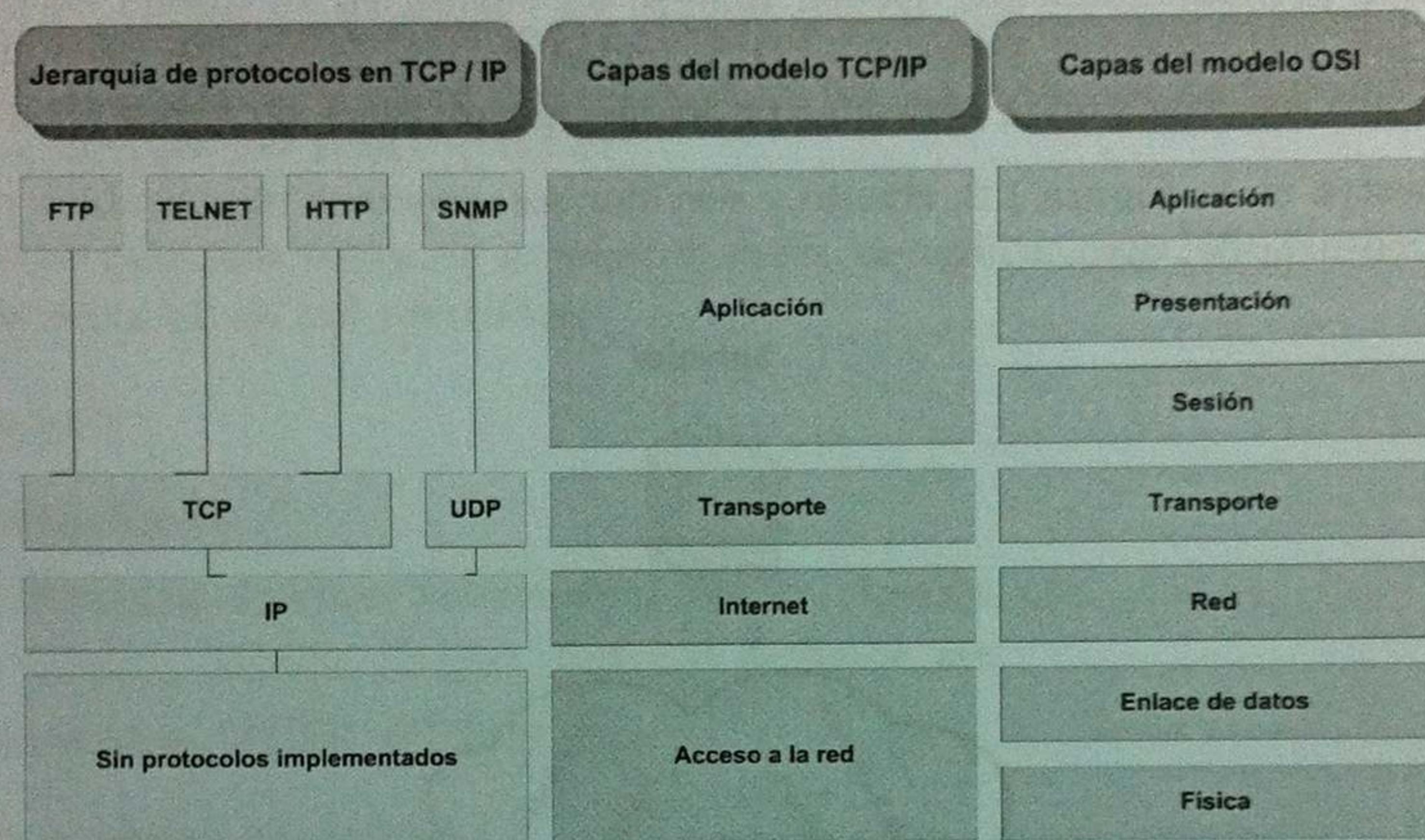


Figura 1.1: Comparativa de arquitecturas TCP/IP y OSI

1.2.2. El modelo Cliente/Servidor

Para la comunicación de aplicaciones a través de una red se emplean fundamentalmente tres paradigmas:

- El modelo **cliente/servidor** es el más extendido y utilizado. En él se distingue entre un proceso cliente (que generalmente solicita servicios) y un proceso servidor (que presta el servicio al cliente).
- El modelo **entre pares** o **P2P** (*Point to Point*) de difusión creciente y donde todos los nodos de la red son responsables por igual en la comunicación de las aplicaciones y no existe un elemento que centralice la comunicación.
- El modelo **híbrido** que resulta de la combinación de los dos anteriores y donde el servidor no presta el servicio como tal, sino que generalmente pone en contacto a los clientes para que estos se comuniquen entre sí.

El modelo **cliente/servidor** es el más extendido y el que se aplica en la mayoría de los diferentes servicios incluidos en esta obra, y será por tanto, en el que vamos a centrar preferentemente nuestra atención.

El modelo **cliente/servidor** define tanto la estructura de las aplicaciones que se comunican entre sí como su sincronización. Está formado por dos procesos que interactúan entre sí, el proceso **cliente** y el proceso **servidor**:

- El **cliente** es el proceso que habitualmente inicia la comunicación por lo que su papel suele ser activo en la misma, envía una petición a un proceso servidor y a continuación queda a la espera de la respuesta.
- El **servidor** suele ser un proceso que inicialmente permanece a la espera escuchando las posibles conexiones de los potenciales clientes. En este punto, su papel es pasivo en la comunicación. Aún así, son sistemas complejos, difíciles de diseñar y programar, y suelen requerir privilegios del sistema. Deben ser robustos, pues ofrecen servicios que es posible que deban estar activos permanentemente. Deben tenerse en cuenta en su diseño cuestiones como la autenticación, autorización, seguridad y privacidad de la información. También es habitual que el proceso servidor pueda gestionar peticiones simultáneas de varios clientes.

Como se muestra en la Figura 1.2, cliente y servidor son procesos inseparables y uno carece de sentido sin la presencia del otro.

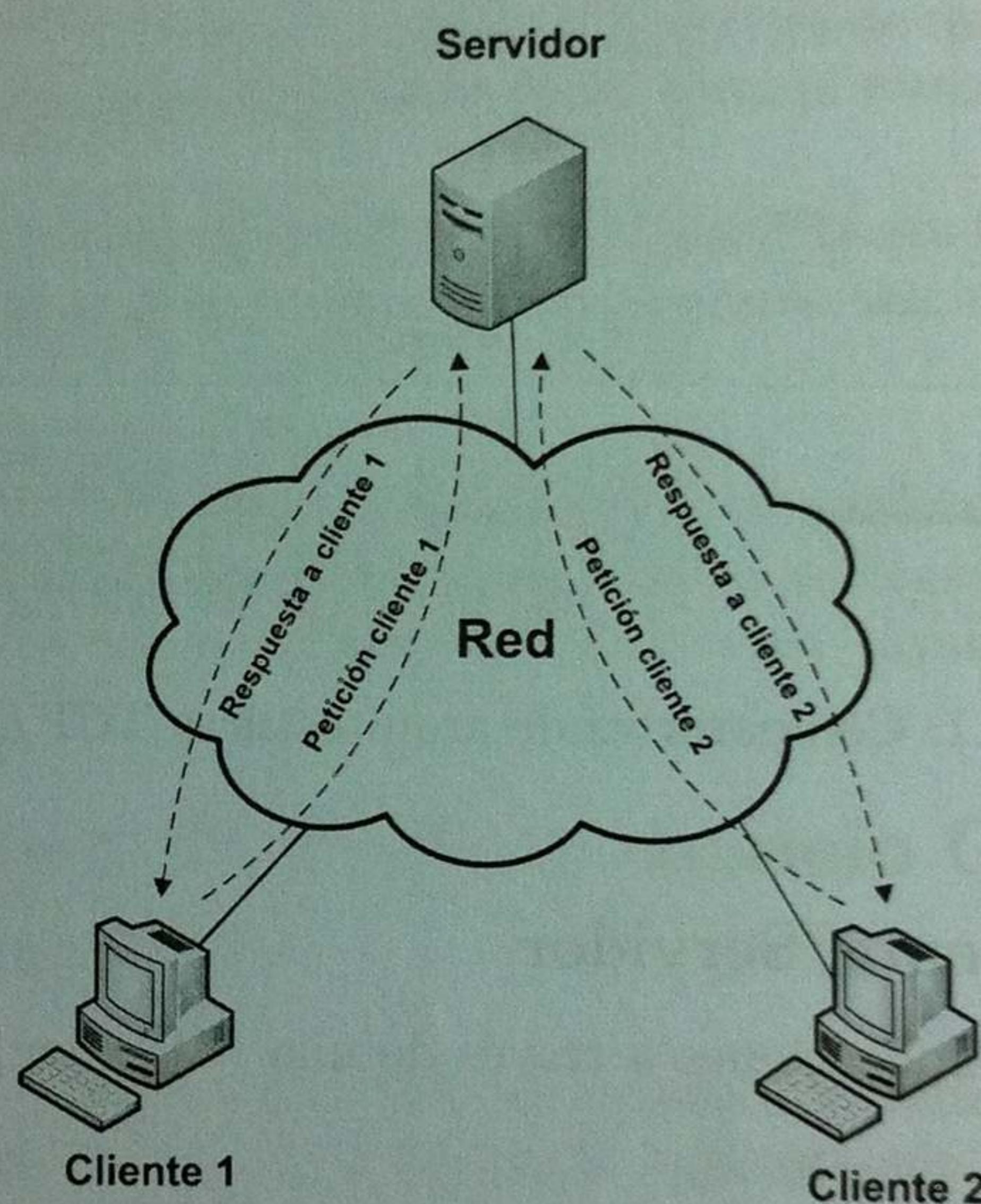


Figura 1.2: Modelo Cliente / Servidor

¿Sabías que ... ? La arquitectura TCP/IP permite que los procesos servidores puedan atender miles de conexiones de clientes de forma simultánea, aunque generalmente esta cantidad suele limitarse por parte del administrador del servidor para evitar la ralentización del sistema o incluso los ataques al mismo.

1.2.3. Los servicios de red

Un **servicio de red** es, o puede considerarse, una función o prestación que ofrecen las **aplicaciones** y los **protocolos** a los **usuarios** o a otras **aplicaciones**.

En este sentido, las **aplicaciones** son sistemas software que se comunican e intercambian información con otras aplicaciones, con ayuda de los **protocolos de la arquitectura TCP/IP**, tanto de nivel de aplicación como de niveles inferiores.

Es importante no confundir los **protocolos del nivel de aplicación** con las **aplicaciones** que los utilizan:

- Las **aplicaciones** son los diferentes programas, instalados por el usuario o que son parte del sistema operativo, que se sirven de los protocolos de la arquitectura TCP/IP para comunicarse. Ejemplos de aplicaciones son *Mozilla Thunderbird*, *Google Chrome*, etc.
- Los **protocolos** son normas concretas, descritas formalmente, que detallan cómo se produce la comunicación entre sistemas para ofrecer los servicios de red. Ejemplos de protocolos del nivel de aplicación son IMAP y HTTP.

Vemos algunos ejemplos reales de la relación entre las aplicaciones, los protocolos del nivel de aplicación y los servicios de red prestados.

- **Servicio web:**
 - **Aplicaciones:**
 - Servidor: *Internet Information Server*, *Apache*, etc.
 - Cliente: *Mozilla Firefox*, *Internet Explorer*, etc.
 - **Protocolos:** HTTP, HTTPS, etc.
- **Servicio de correo electrónico:**
 - **Aplicaciones:**
 - Servidor: *Sendmail*, *Postfix*, *Exchange*, etc.
 - Cliente: *Mozilla Thunderbird*, *Outlook*, *Sylepheed*, etc.
 - **Protocolos:** POP, SMTP, IMAP, etc.

Para realizar su función los protocolos del nivel de aplicación emplean los protocolos de niveles TCP/IP inferiores para funcionar. Así, por ejemplo, el protocolo de aplicación *Telnet* emplea el protocolo TCP en la capa de transporte, mientras que el protocolo de aplicación DHCP emplea el protocolo UDP en la capa de transporte.

Para entender claramente el funcionamiento de un servicio de red es especialmente importante prestar atención a los niveles de **red** y **transporte** de la arquitectura TCP/IP.

1.3. Nivel de red en TCP/IP - El protocolo IP

A nivel de red se realiza el **direcciónamiento** de los dispositivos y el **encaminamiento** de la información a través de la red. Todo ello se lleva a cabo con el protocolo IP que es el principal de este nivel en la arquitectura TCP/IP. El esquema de direcciónamiento utilizado en cada nodo de la red y los procesos de encaminamiento, que se ejecutan en los dispositivos que interconectan las redes, son las funciones principales de este protocolo. La comunicación a nivel IP se hace mediante

unidades de datos llamadas datagramas que siguen el formato especificado en el propio protocolo IP.

Actualmente se emplea mayoritariamente la versión 4 del protocolo IP (**IPv4**), pero la versión siguiente **IPv6** está desarrollada desde hace años y empieza a implantarse en un número creciente de entornos. Dado que la versión dominante sigue siendo IPv4, todas las referencias que se hagan a IP se referirán implícitamente a IPv4, mientras que cualquier indicación relativa a IPv6 se hará de forma expresa.

1.3.1. Direcciónamiento IP

El protocolo IP proporciona conectividad extremo a extremo en la comunicación. Esto supone que debe ser capaz de direccionar de forma única todos los dispositivos que tengamos conectados en nuestra red y, por extensión, en todo Internet. Este direccionamiento es abstracto, de forma que es independiente del dispositivo físico al que se asigna y puede ser modificado vía software.

¿Sabías que ... ? Una dirección IP no identifica a un ordenador en la red, sino que identifica a un interfaz de red de un ordenador en la red. Por eso es posible que un mismo equipo pueda tener varias direcciones IP, una por interfaz, y eso hace posible que pueda estar conectado a redes diferentes de manera simultánea. Incluso es posible que un interfaz pueda tener varias direcciones IP denominándose entonces direccionamiento virtual.

1.3.1.1. Formato de direcciones IP

Una dirección IP es un número binario de 32 bits. Esto permite un espacio de direcciones de 2^{32} (4.294.967.296) direcciones diferentes posibles. Habitualmente, la notación empleada para facilitar la legibilidad de las direcciones IP es la notación decimal con puntos. Así, se dividen los 32 bits en 4 grupos de 8 bits, escribiendo cada uno de ellos en base decimal, separando por puntos los cuatro números resultantes. Por tanto, como ilustra la Figura 1.3, una dirección IP estará formada por cuatro números entre 0 y 255 separados por puntos.

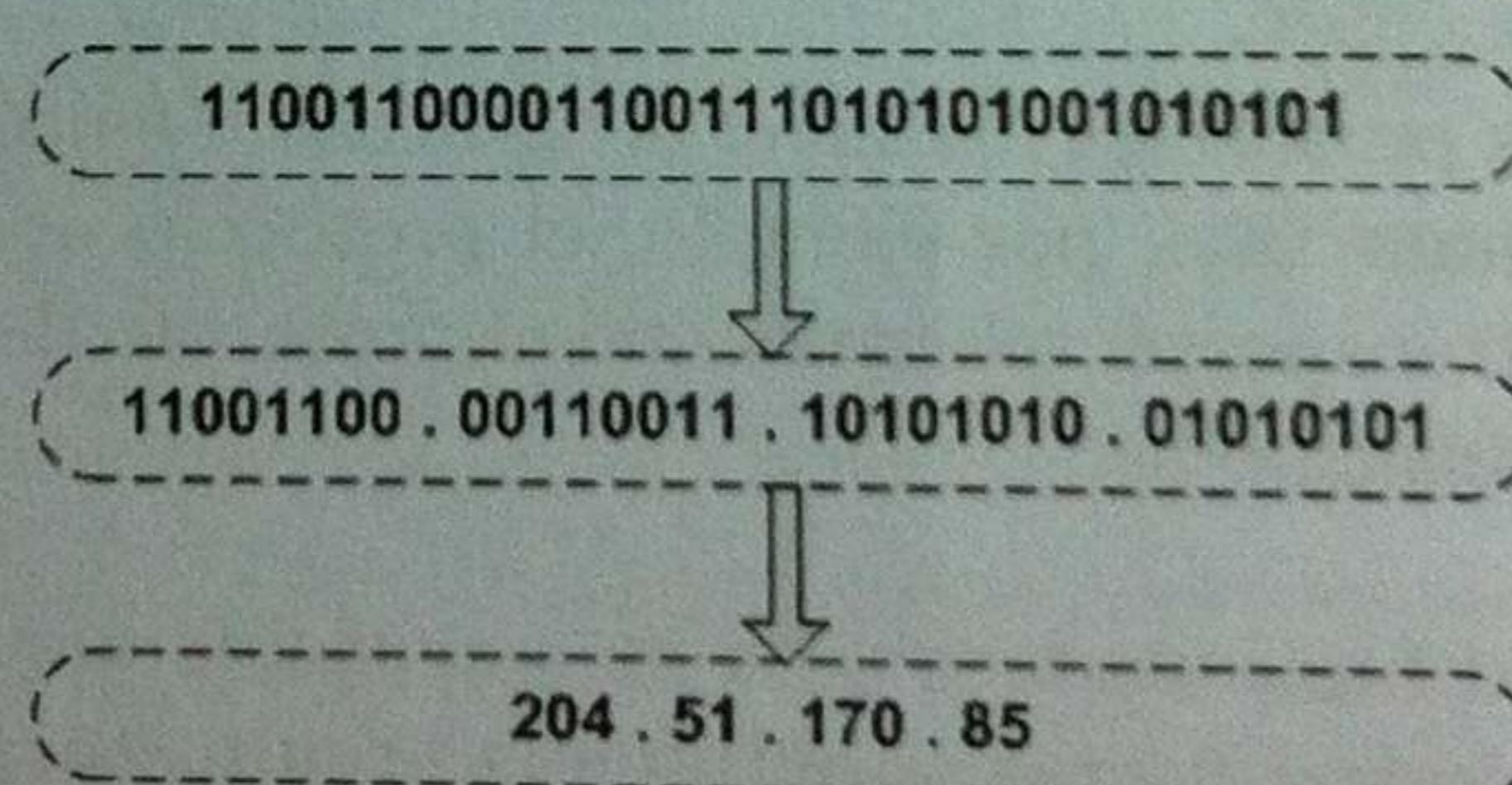


Figura 1.3: Ejemplo de dirección IP

A efectos de direccionamiento y de encaminamiento las direcciones IP constan de dos partes:

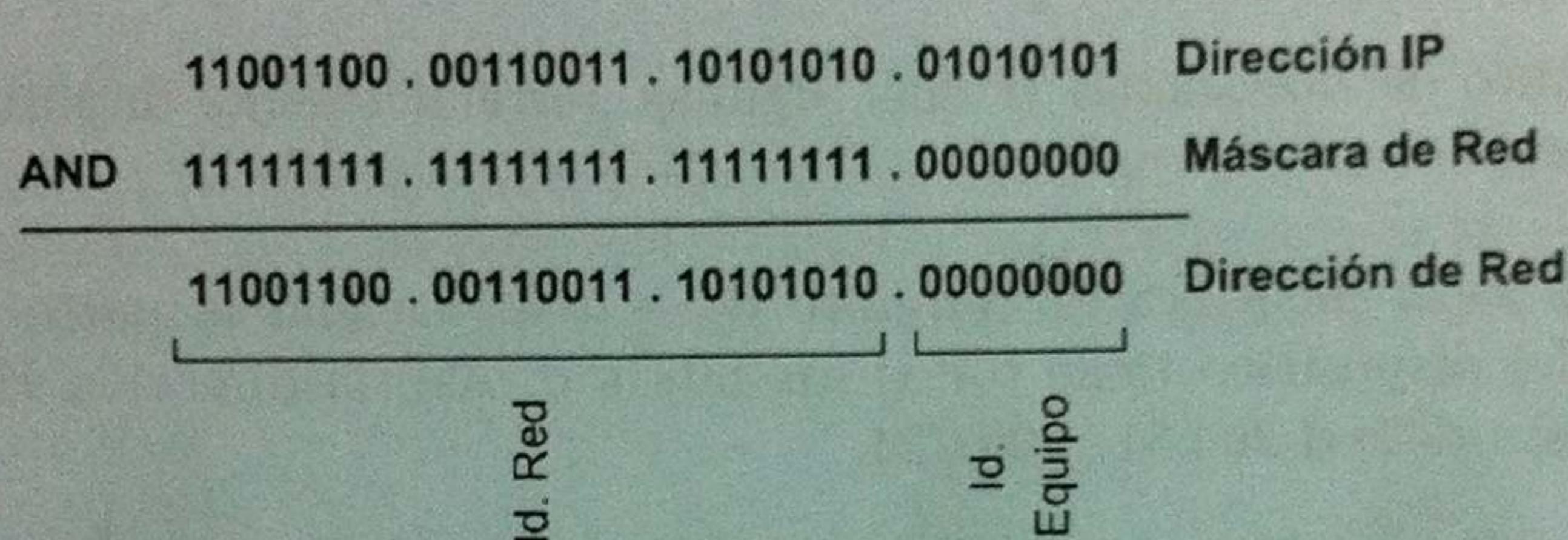
- **Identificador de red**, que determina la red en la que se encuentra el dispositivo.
- **Identificador del host** dentro de la red.

De esta forma todos los *hosts* de una misma red comparten la parte de identificador de red. Esta estructura se asemeja al sistema telefónico de prefijo + número local. Además, el identificador

de red podrá tener el valor que se quiera en función del tamaño de la red. Así, las redes grandes tendrán un identificador de red pequeño y las redes pequeñas tendrán un identificador de red grande.

1.3.1.2. Máscara de red

La máscara de red se emplea para diferenciar el prefijo de la dirección IP correspondiente al identificador de red, de la parte correspondiente al identificador del *host*. La máscara de red es un número de 32 bits que define en las posiciones a “1” el prefijo o identificador de red, y en las posiciones a “0” el sufijo o identificador del *host*. Como se ve en la Figura 1.4, será un número con N 1’s a la izquierda y (32 - N) 0’s a la derecha.



La máscara de red también puede expresarse mediante la notación CIDR (*Classless Inter-Domain Routing*) consistente en situar un sufijo a continuación de la dirección IP que indica cuántos bits de la máscara de red están a 1, Figura 1.5.

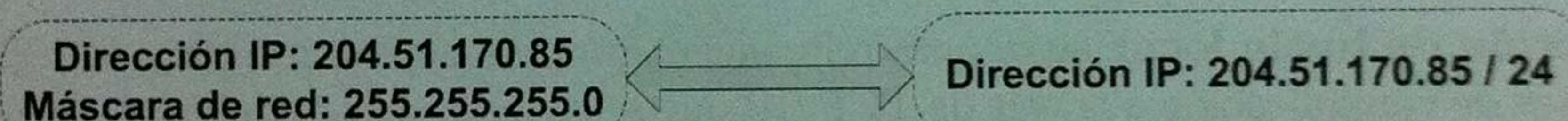


Figura 1.5: Notación CIDR

1.3.1.3. Clases de direcciones IP

Como ya se ha dicho anteriormente, el identificador de red puede tener una longitud en bits variable, aunque en un principio se predeterminaron una serie de máscaras de red concretas para facilitar el proceso de encaminamiento. Surge así el concepto de **clase de direcciones**. Así:

- **Clase A:** un *byte* para el identificador de red y tres *bytes* para el identificador del *host* (redes grandes, ya que deja gran número de bits (24) para direccionar los *host* dentro de la red). La máscara de red equivalente es 255.0.0.0.
- **Clase B:** dos *bytes* para el identificador de red y dos *bytes* para el identificador del *host* (redes medianas). La máscara de red equivalente es 255.255.0.0.
- **Clase C:** tres *bytes* para el identificador de red y un *byte* para el identificador del *host* (redes pequeñas, ya que deja solo 8 bits para direccionar los *host* dentro de la red). La máscara de red equivalente es 255.255.255.0.

- **Clase D:** se emplea para *multicast*, es decir, envío de datagramas a un grupo de equipos de la red. No diferencian entre identificador de red y de *host*.
- **Clase E:** reservadas para uso experimental en proyectos de investigación. Dado que su uso no está especificado, tampoco podemos hablar en este caso de una estructura interna tipo identificador de red, identificador de *host* para estas direcciones.

La idea original de esta clasificación pretendía asignar direcciones de redes IP a los diferentes organismos y/o empresas en función del tamaño de sus redes. Actualmente, el concepto de clase sigue estando vigente, pero a efectos prácticos tanto los *routers* como los *hosts* emplean la máscara de red para diferenciar la parte de la dirección IP que identifica a la red de la que identifica al *host*.

1.3.1.4. Direcciones especiales

Dentro del conjunto de direcciones IP hay algunas particularmente importantes que merecen una explicación aparte:

- **Dirección de red:** identifica al conjunto de la red. En ella la parte correspondiente al identificador del dispositivo tiene todos sus bits a 0. Así, la dirección IP 204.51.170.85/24 tiene la dirección de red 204.51.170.0/24.
- **Dirección de difusión limitada:** se emplea para mandar un mensaje de difusión o *broadcast* al conjunto de dispositivos de la propia red. Es la misma para todas las redes (255.255.255.255).
- **Dirección de difusión dirigida:** se emplea para mandar un mensaje de difusión o *broadcast* al conjunto de dispositivos de una red. Por tanto, no puede asignarse a un interfaz de red en concreto. Viene dado por el identificador de la red en la que queramos hacer la difusión a la izquierda y los bits correspondientes a la dirección del dispositivo todos a 1 a la derecha. Por ejemplo, la dirección IP 204.51.170.255/24 hace difusión dirigida en la red 204.51.170.0/24.
- **Dirección de bucle local:** sirve para referenciar internamente al interfaz, es decir, para los procesos de comunicación a través de TCP/IP que se generan dentro del *host*. Se emplea para ello cualquier dirección de la red 127.0.0.0/8 aunque por comodidad lo habitual es utilizar la dirección 127.0.0.1/8.

◊ **Actividad 1.1:** A partir de los siguientes pares (dirección IP, máscara de red) obtén la dirección de red, y la primera y última dirección posible para un dispositivo concreto en esa red.

- (192.168.14.3, 255.255.0.0) \Rightarrow (192.168.0.0, 192.168.0.1, 192.168.255.254)
- (10.23.31.7, 255.255.255.0)
- (8.45.127.12, 255.0.0.0)
- (8.45.127.12, 255.255.240.0)
- (223.145.90.131, 255.255.255.192)
- (140.30.23.31, 255.224.0.0)

1.3.1.5. Direcciones públicas y privadas

Dentro del espacio de direcciones hay algunas que se han reservado para un uso privado, es decir, son direcciones que no deben tener acceso a Internet. Se distingue así entre:

- **Direcciones públicas:** identifican a un dispositivo conectado a Internet.
- **Direcciones privadas:** son rangos de direcciones reservados para redes privadas o **intranets** y no pueden emplearse en Internet. Son las pertenecientes a las siguientes redes: 10.0.0.0/8, 172.16.0.0/16 y 192.168.0.0/16. Los *routers* conectados a redes públicas (Internet) descartan el tráfico dirigido a direcciones privadas como medida adicional de seguridad.

◊ **Actividad 1.2:** A partir del siguiente diagrama de red, Figura 1.6, completa los espacios con la información adecuada relativa a las direcciones de red y las direcciones de los diversos dispositivos. La solución no es única. Comenta posteriormente la idoneidad de la solución obtenida.

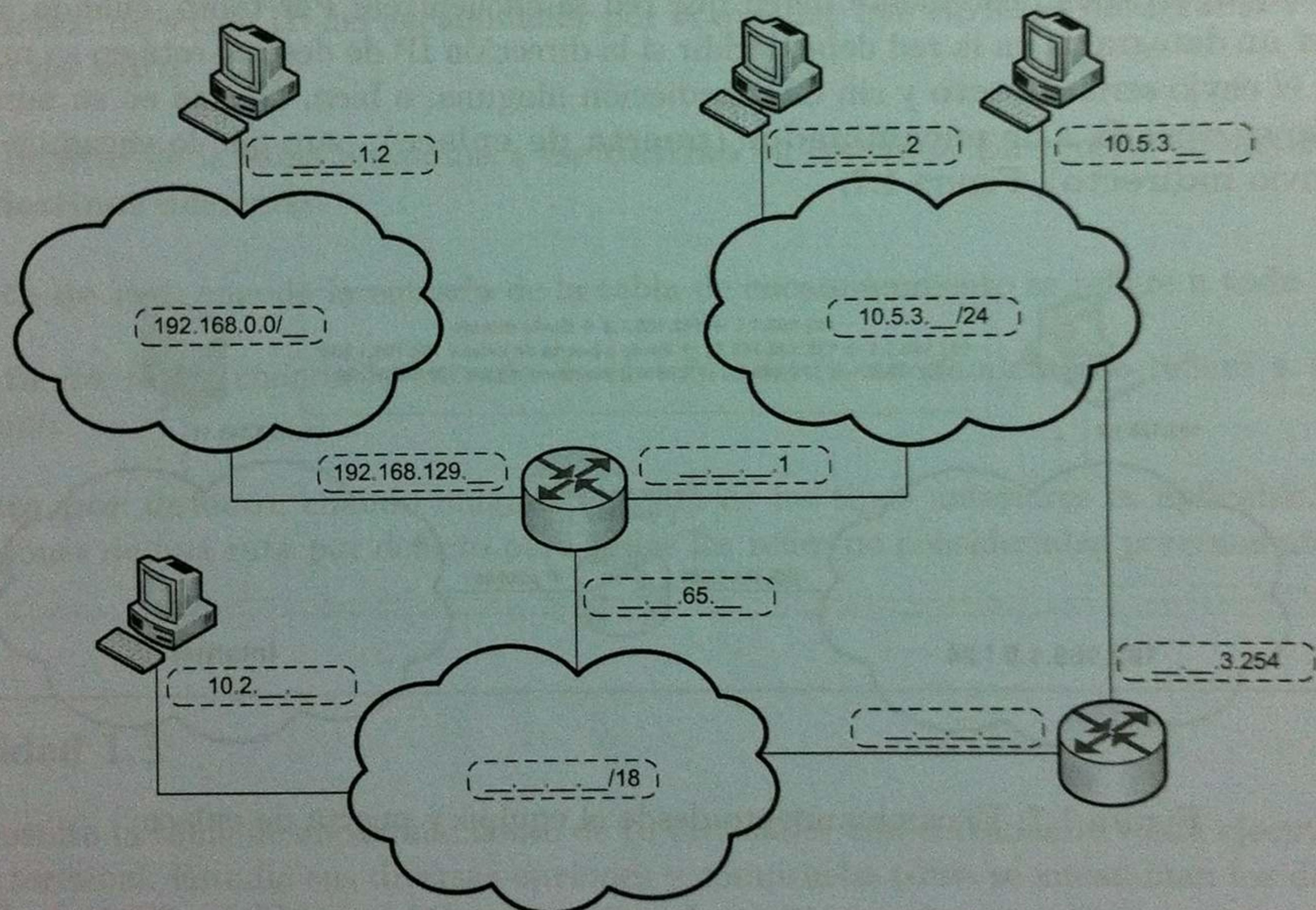


Figura 1.6: Red a direccionar

1.3.1.6. Direcciones de enlace local

Hay un conjunto de direcciones reservado para ser asignado en redes que no dispongan de direccionamiento estático ni dinámico, es decir, son direcciones que se asignan los dispositivos a sí mismos cuando nadie lo hace. Se emplean las direcciones de la red 169.254.0.0/16. Estas direcciones tampoco pueden circular por Internet.

1.3.2. Encaminamiento IP

El **encaminamiento** a nivel IP puede definirse como el proceso de llevar un datagrama desde la máquina origen a la máquina destino, independientemente de si ambas máquinas se encuentran en la misma o en diferentes redes. El protocolo IP es el responsable de este encaminamiento.

1.3.2.1. Encaminadores

Los **encaminadores** o *routers* son dispositivos de nivel 3 que enlazan las diferentes redes que forman parte de una “red de redes” y van a desempeñar un papel crucial en el encaminamiento de los datagramas. Un *router* estará conectado al menos a dos redes y **realizará el encaminamiento de todo el tráfico de datagramas que pase por él**. Para ello se servirá de las tablas de encaminamiento que veremos más adelante.

Además de los *routers*, los propios **equipos** también van a participar en el encaminamiento. De hecho, el encaminamiento se inicia incluso antes de que un equipo coloque un datagrama en la red. Analizamos esta afirmación: es importante apreciar que a nivel de dirección IP y desde el punto de vista de un equipo, el mundo se divide en dos, las direcciones que están en su misma red y luego todas las demás, independientemente de en qué red se encuentren. Por tanto, cuando un equipo desee poner un datagrama en la red debe decidir si la dirección IP de destino está en su misma red, con lo que el envío sería **directo** y sin intermediación ninguna, o bien, no está en su misma red y en este caso se enviaría a un **encaminador (puerta de enlace)**, para que lo encamine hacia su destino (envío **indirecto**), Figura 1.7.

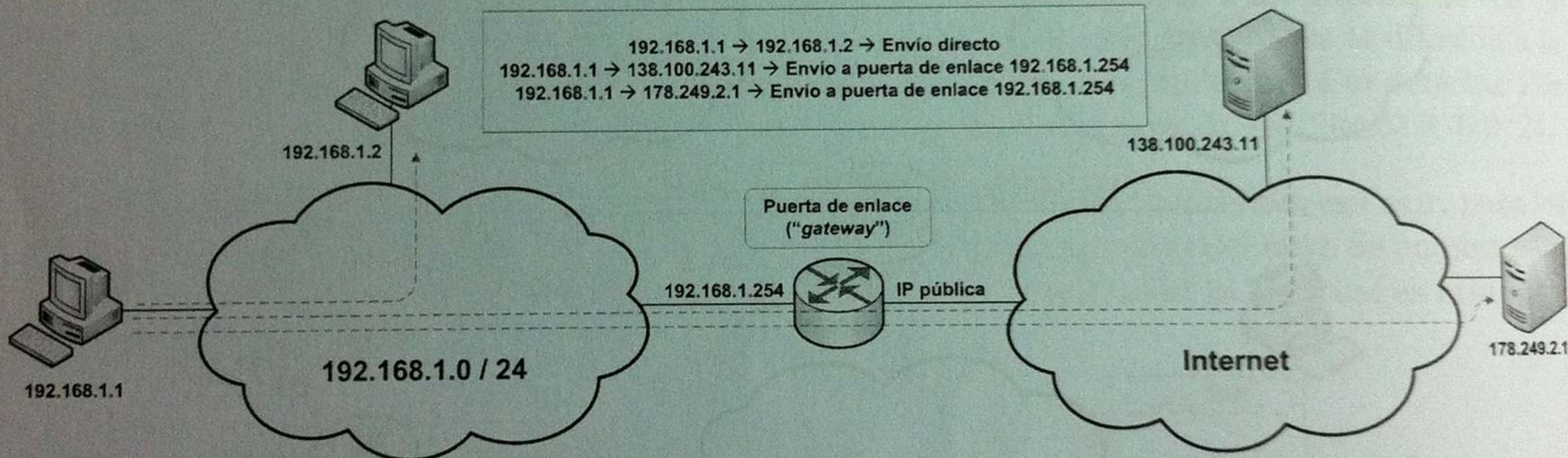


Figura 1.7: Encaminamiento desde el equipo y puerta de enlace

Un matiz importante es que los equipos solamente encaminan el tráfico saliente, a diferencia de los *routers* que encaminan todo el tráfico que pasa por ellos².

El *router*, una vez recibe el datagrama, debe encaminarlo. Pueden darse dos situaciones:

- Que el datagrama vaya dirigido a una dirección que pertenezca a una red conectada directamente al *router*, en cuyo caso la entrega es directa.
- Que el datagrama no vaya dirigido a una red conectada directamente al *router*, en cuyo caso reenviará el datagrama hacia otro encaminador siguiendo lo especificado en su tabla

²Por esta razón un *router* es “algo más” que un ordenador con dos interfaces de red conectados a redes diferentes.

de encaminamiento. Caso de ser necesario este proceso se repetiría hasta alcanzar la red de destino o bien agotar el tiempo de vida (TTL, *Time to live*) del datagrama.

1.3.2.2. Tablas de encaminamiento

Las tablas de encaminamiento, Figura 1.8, almacenan la información necesaria para realizar el encaminamiento de los datagramas y están implementadas tanto en los *routers* como en los *hosts*. La información que contienen depende del protocolo de encaminamiento concreto empleado pero en general, los campos más importantes de los que consta son:

- **Destino (D)**: dirección IP de una *red* o *host*.
- **Máscara de red (MR)**: asociada al **Destino** anterior sirve para determinar exactamente todas las direcciones IP que incluye.
- **Dirección de salto (DS)**: dirección IP a la que se enviará el datagrama si su dirección IP de destino coincide con la especificada por **Destino** y **Máscara de red**.
- **Interfaz**: dirección IP del encaminador por la que hay que enviar el datagrama a la **Dirección de salto**.

Cada registro de la tabla encamina a ese **destino** en concreto. En general se distinguen tres tipos de **destinos** diferentes:

- **Ruta de red**: cuando la entrada de la tabla de encaminamiento se refiere a toda una *red*.
- **Ruta de host**: cuando la entrada de la tabla de encaminamiento se refiere a un *host* o equipo.
- **Ruta por defecto**: cuando ninguna entrada de los tipos anteriores es aplicable, se puede disponer de una ruta por defecto para todas las redes no consideradas previamente.

◊ Actividad 1.3:

- Consulta la tabla de encaminamiento de tu ordenador con el comando *route* ejecutado desde un terminal. Estudia sus diversas opciones y comprueba cómo se encaminan los datagramas hacia la puerta de enlace o bien son entregados directamente a su destino.
- Si te encuentras en *Linux*, puedes ejecutar el comando *route -C* que mostrará la *caché* de encaminamiento con los últimos datagramas encaminados.

1.3.2.3. Protocolos de encaminamiento

Al arrancar los equipos, tanto *routers* como *hosts*, las tablas de encaminamiento se inicializan con las rutas correspondientes a las redes adyacentes al mismo. A partir de este momento, se pueden seguir dos estrategias para configurar las tablas de encaminamiento:

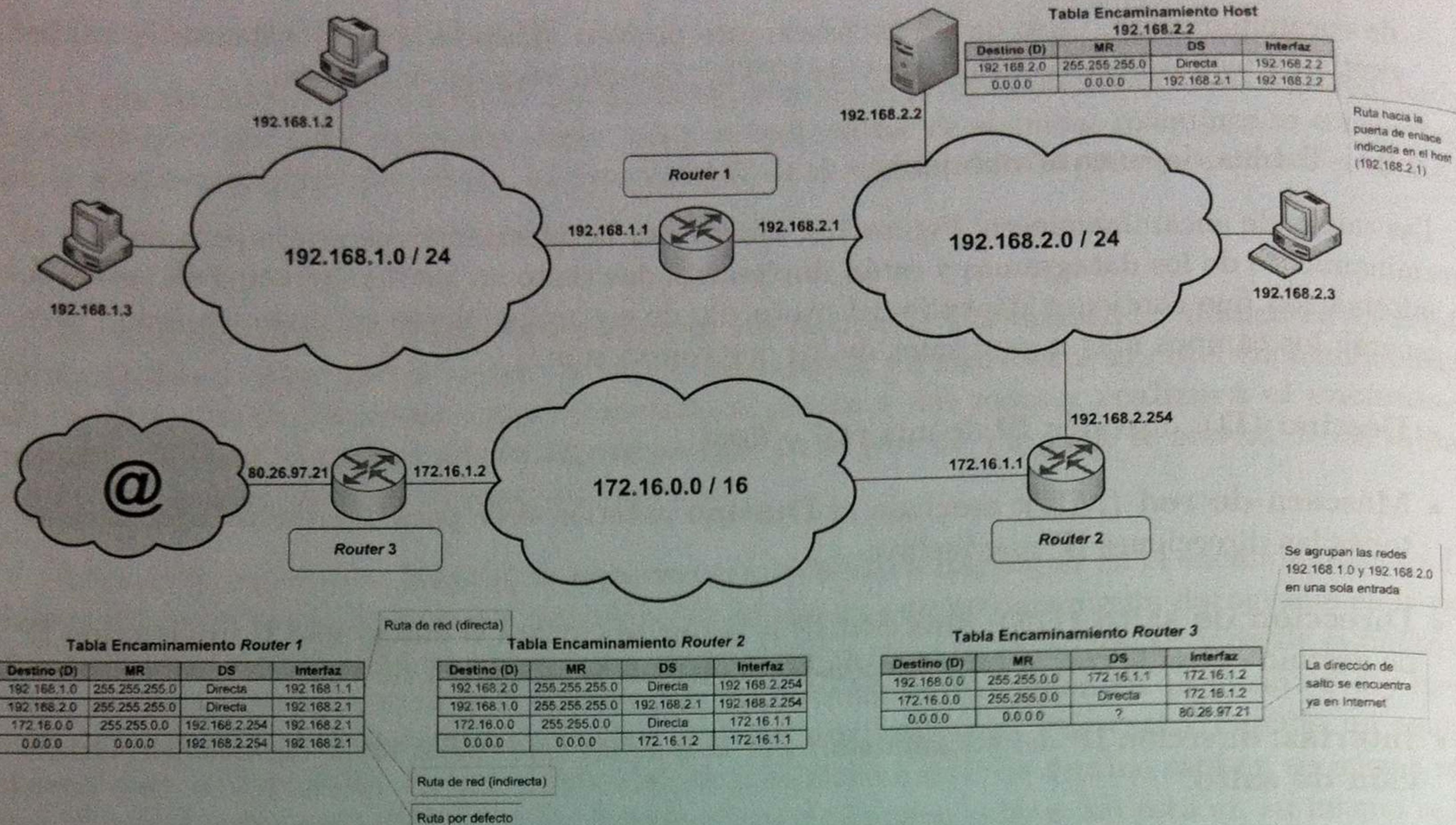


Figura 1.8: Ejemplo de tablas de encaminamiento

- **Encaminamiento estático:** la configuración de las tablas de encaminamiento se hace de forma manual. Es una estrategia no adaptativa, es decir, que cualquier cambio que se produzca en la topología de la red debe ser supervisado por el administrador para evitar rutas imposibles o bucles indeseados. Por esto mismo es muy sensible a fallos y solo recomendable en redes de pequeño tamaño y topología fija.
- **Encaminamiento dinámico:** el propio encaminador actualiza sus tablas gracias a la utilización de protocolos específicos como **RIP** (*Routing Information Protocol*), **OSPF** (*Open Shortest Path First*) y **BGP** (*Border Gateway Protocol*) que permiten que los encaminadores se intercambien información de encaminamiento para mantener sus tablas lo más actualizadas posibles.

En resumen, cada estrategia y cada protocolo de encaminamiento presenta ventajas e inconvenientes.

1.4. Nivel de transporte en TCP/IP - Protocolos TCP y UDP

En los sistemas operativos multitarea y en red actuales es habitual que las comunicaciones impliquen, de forma simultánea, a varios procesos dentro de una misma máquina y a varias máquinas dentro de la red.

Hasta ahora, hemos visto que el protocolo IP nos permite comunicar dos máquinas remotas haciendo que los datagramas puedan ir del origen al destino, pero al disponer únicamente de las direcciones IP de origen y destino como mecanismo de diferenciación en la comunicación, se plantea

el problema de que no nos permite, por ejemplo, mantener varias comunicaciones simultáneas entre los dos mismos equipos, ya que a nivel IP no podríamos diferenciar los datagramas pertenecientes a unas u otras.

El **nivel de transporte** nos provee de elementos para diferenciar y gestionar, de forma simultánea, múltiples orígenes y destinos en una comunicación y múltiples comunicaciones en cada equipo. También permite identificar los extremos finales en la comunicación y nuevos servicios orientados a conexión.

1.4.1. Puertos de comunicaciones

Los protocolos del nivel de transporte implementan el concepto de **puerto de comunicaciones** que nos permite identificar los procesos del nivel de aplicación entre los que está establecida una comunicación.

Así, cada proceso del nivel de aplicación tiene asociado uno o varios puertos a través de los cuales es accesible. Cada puerto se identifica por un número binario de 16 bits que en notación decimal puede variar entre 0 y $2^{16} - 1 = 65535$.

Existen varias clases de puertos en función del uso que se hace de ellos:

- **Puertos conocidos (0 - 1023)**: se conocen como *well known ports* y están reservados para aplicaciones y servicios estándar como HTTP, FTP, etc. Las aplicaciones clientes se conectan a estos puertos para acceder a los servicios.
- **Puertos registrados (1024 - 49151)**: para aplicaciones no estándar instaladas por el usuario que no tienen un puerto *well known* preasignado. Estos puertos pueden asignarse dinámicamente a clientes si ningún servicio está haciendo uso de ellos.
- **Puertos dinámicos (49152 - 65535)**: habitualmente se emplean para iniciar conexiones desde el cliente. No suelen emplearse en procesos servidores.

La correspondencia entre procesos y puertos se hace de dos formas distintas:

- **Asignación estática**: los *well known ports* están reservados para aplicaciones estándar y solo pueden ser empleados por estos procesos.
- **Asignación dinámica**: cuando un proceso necesita un puerto y este no se asigna estéticamente, el sistema operativo le asigna uno que esté disponible (1024-65535).

En el nivel de transporte disponemos de dos protocolos completamente independientes (TCP y UDP) y ambos manejan el concepto de puerto, pero es importante destacar que los puertos TCP y UDP son totalmente independientes, no teniendo por tanto ninguna relación el puerto 53 de TCP con el puerto 53 de UDP.

1.4.2. Protocolo UDP

El protocolo **UDP** (*User Datagram Protocol*) proporciona un servicio no orientado a conexión (al igual que IP) con todo lo que esto supone: sin establecimiento de conexión previo a la transmisión, sin control de flujo (existirá la posibilidad de entrega de segmentos duplicados o desordenados), etc.

Al tratarse de un protocolo muy básico suele emplearse en casos en los que prevalece más la velocidad de la transmisión respecto a la fiabilidad de la misma o bien en aplicaciones con requerimientos sencillos del tipo petición-respuesta como DHCP, DNS, *streaming* y voz IP.

1.4.3. Protocolo TCP

El protocolo **TCP** (*Transmission Control Protocol*) proporciona un servicio orientado a conexión con lo que existen grandes diferencias respecto a UDP. TCP obliga al establecimiento previo de una conexión antes de empezar a transmitir y ofrece control de flujo y de errores con lo que garantiza al nivel de aplicación un **servicio fiable**.

1.4.3.1. Conexiones TCP

La **conexión TCP** es el paso previo imprescindible para iniciar una comunicación. Una vez se ha establecido la conexión cualquiera de los extremos de la misma puede empezar a transmitir y también terminar la conexión en el momento que lo deseé. La conexión TCP se define de forma única por los datos relativos a los puntos extremos de la comunicación, es decir, por estos cuatro elementos: (**Dirección IP origen, Puerto TCP origen**) \Rightarrow (**Dirección IP destino, Puerto TCP destino**). No puede haber dos conexiones TCP que tengan en común estos cuatro elementos. Se aprecia claramente en la Figura 1.9.

◊ **Actividad 1.4:** Emplea el comando *netstat* para averiguar información sobre las conexiones TCP/IP y los puertos de escucha de **tu** ordenador. Utiliza la ayuda con *netstat -h*.

- Abre en un navegador la página www.garceta.es y consulta las conexiones TCP establecidas. ¿Reconoces la conexión a la página anterior? Comenta todo lo que sepas de ella.
- Consulta por un lado los puertos UDP abiertos y por otro todas las conexiones TCP establecidas. ¿Qué diferencias aprecias en la información mostrada referente a UDP y TCP? ¿A qué se debe?
- Muestra las aplicaciones que han creado cada una de las conexiones TCP.

◊ **Actividad 1.5:** Emplea la utilidad *Nmap* para averiguar información sobre las conexiones TCP/IP y los puertos de escucha de **otros** ordenadores. Descarga la utilidad desde <http://nmap.org/download.html>. Hay disponibles versiones de línea de comandos y de interfaz gráfica (*Zenmap*).

- Consulta los puertos TCP y UDP abiertos de tu propio ordenador (*localhost*) y contrástalo con lo obtenido mediante *netstat*.
- Averigua los puertos TCP y UDP abiertos en **la puerta de enlace** de tu red. Comenta la utilidad de todos los puertos conocidos, *well known ports*, que hayas encontrado.
- Averigua los puertos TCP y UDP abiertos en www.google.es y en asterix.fi.upm.es.

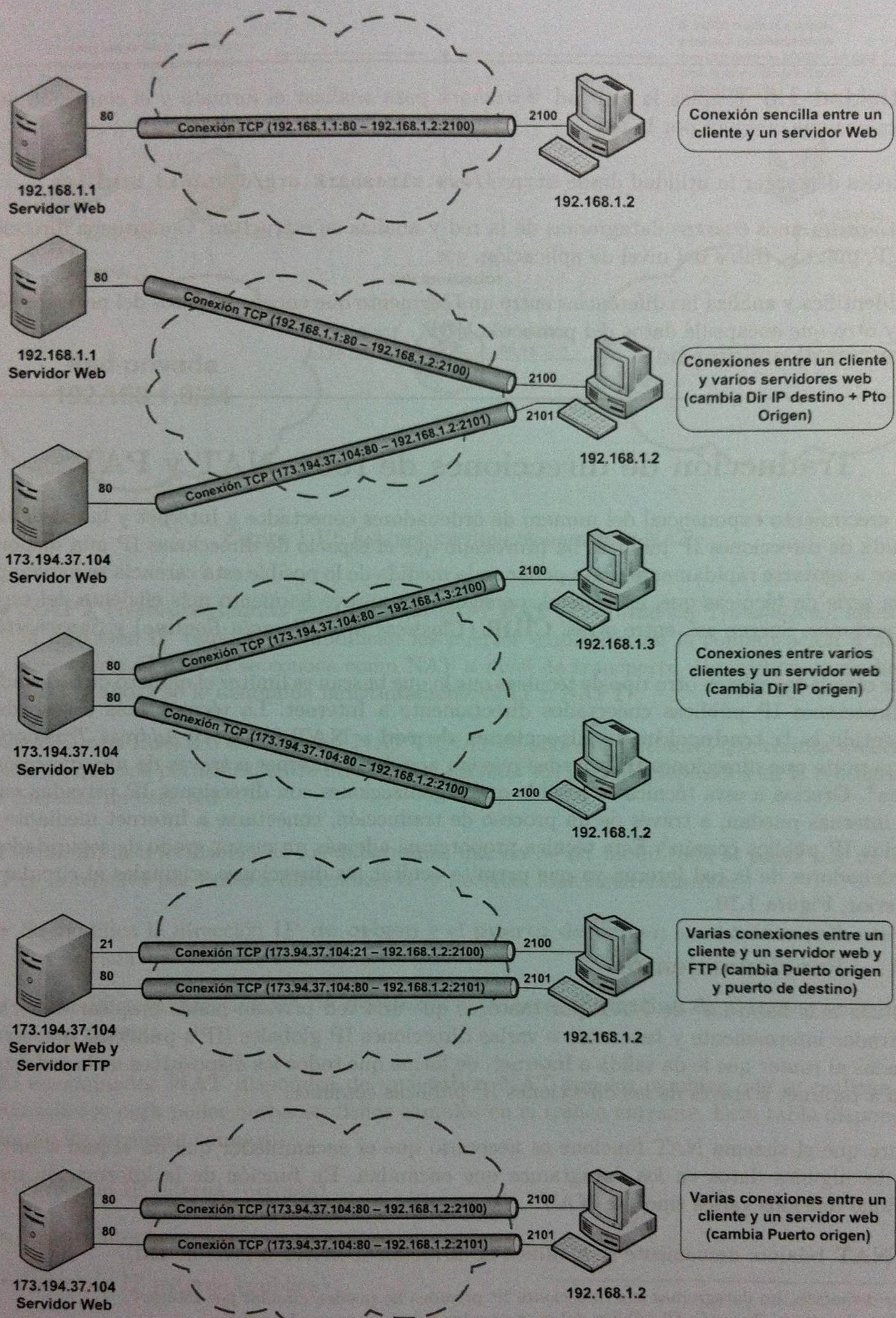


Figura 1.9: Conexiones TCP

-
- ◊ **Actividad 1.6:** Emplea la utilidad *Wireshark* para analizar el formato y el contenido de los datagramas que circulan por la red.

Puedes descargar la utilidad desde <http://www.wireshark.org/download.html>.

- Captura unos cuantos datagramas de la red y analiza su estructura. Comprueba direcciones IP, puertos, datos del nivel de aplicación, etc.
 - Identifica y analiza las diferencias entre una segmento que encapsule datos del protocolo TCP y otro que encapsule datos del protocolo UDP.
-

1.5. Traducción de direcciones de red - NAT y PAT

El crecimiento exponencial del número de ordenadores conectados a Internet y la consiguiente demanda de direcciones IP públicas ha provocado que el espacio de direcciones IP aún disponible empiece a agotarse rápidamente. Para paliar en la medida de lo posible esta carencia se han empleado una serie de técnicas que, en general, persiguen un aprovechamiento más eficiente del espacio de direcciones. Surgen así, entre otras, **CIDR** (*Classless Inter-Domain Routing*) y *Supernetting*.

Por otro lado, existen otro tipo de técnicas que lo que buscan es limitar el número de ordenadores con direcciones IP públicas conectados directamente a Internet. La técnica más destacada en este sentido es la **traducción de direcciones de red - NAT** (*Network Address Translation*). NAT permite que direcciones IP privadas puedan acceder a Internet a través de una dirección IP pública³. Gracias a esta técnica es posible que organizaciones con direcciones IP privadas en sus redes internas puedan, a través de un proceso de traducción, conectarse a Internet mediante una dirección IP pública común⁴. Esta técnica proporciona además un mayor grado de seguridad para los ordenadores de la red interna ya que permite ocultar las direcciones originales al circular por el exterior, Figura 1.10.

1.5.1. Funcionamiento

El uso más habitual⁵ de **NAT** por tanto es que una red privada pueda emplear direcciones IP privadas internamente y tener una o varias direcciones IP globales (IP's públicas) que estarán asignadas al *router* que le da salida a Internet, de forma que todos los dispositivos de la red interna salgan a Internet a través de las direcciones IP públicas comunes.

Para que el sistema NAT funcione es necesario que el encaminador que da acceso a Internet reescriba algunos datos en los datagramas que encamina. En función de la información que se modifique tenemos varios tipos de **NAT**:

- **NAT básico:** únicamente se modifica la dirección IP (NAT a nivel de red).

³Por definición, los datagramas con direcciones IP privadas no pueden circular por Internet

⁴Generalmente la dirección IP pública del encaminador que da acceso a Internet

⁵NAT incluye otros posibles mecanismos de funcionamiento como por ejemplo mantener una lista de direcciones IP públicas disponibles y asignárselas a los equipos de la red privada que desean establecer comunicación con el exterior.

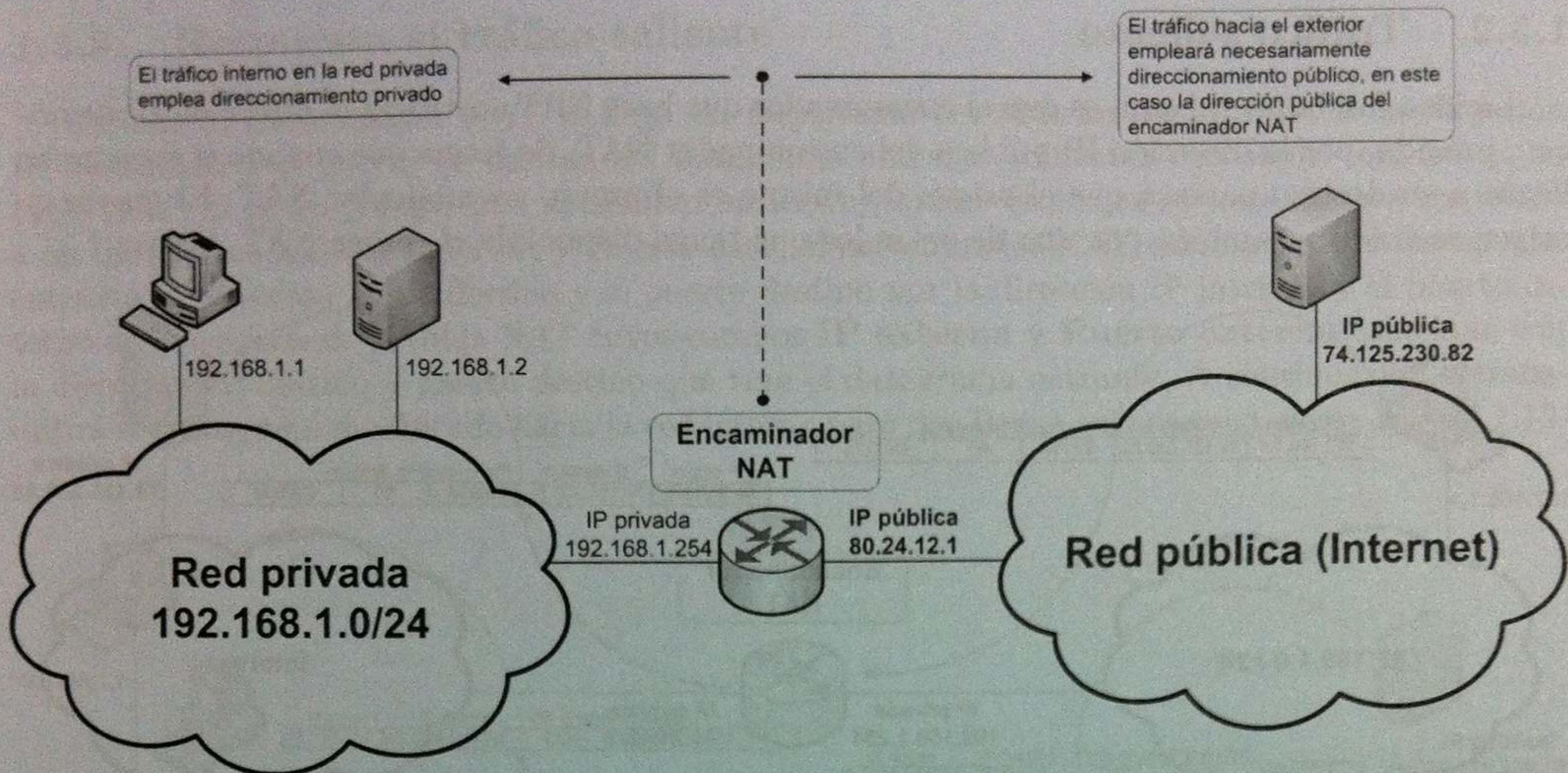


Figura 1.10: Estructura dirección NAT

- **NAPT (Network Address Port Translation) / PAT (Port Address Translation):** además de la dirección IP también se modifican los puertos empleados en la comunicación a nivel de transporte. También se conoce como NAT a nivel de transporte. Ha sustituido de hecho a NAT y ahora se le denomina indistintamente NAT o NAPT.

Centramos nuestro estudio en NAPT (NAT de ahora en adelante) que resulta mucho más versátil ya que nos permitirá, entre otras cosas, cambiar los puertos empleados en la red interna al salir fuera de nuestra red.

Por tanto, la información de los datagramas que ha de ser modificada al pasar por un *router* NAT es la relativa por tanto a direcciones IP y puertos. Más concretamente:

- Se modifica la dirección **IP de origen** y el **puerto de origen** en el tráfico **saliente** de la red privada.
- Se modifica la dirección **IP de destino** y el **puerto de destino** en el tráfico **entrante** en la red privada.

El encaminador **NAT** dispondrá de una **tabla NAT** con los cambios que se realizan en el tráfico saliente para poder deshacer dichos cambios en el tráfico entrante. Esta tabla dispondrá al menos de los siguiente campos:

- Dirección IP interna (privada)
- Puerto interno
- Dirección IP externa (pública)
- Puerto externo
- Protocolo del nivel de transporte (TCP o UDP)

1.5.2. Tráfico saliente

La idea fundamental aquí es que el encaminador que hace NAT sustituya la dirección IP de origen, privada, por la dirección IP pública del encaminador NAT, de forma que cuando el datagrama llegue a su destino parezca que el origen del mismo es el propio encaminador NAT. El puerto de origen se traduce también por uno de entre los que tenga disponible el *router* NAT, Figura 1.11.

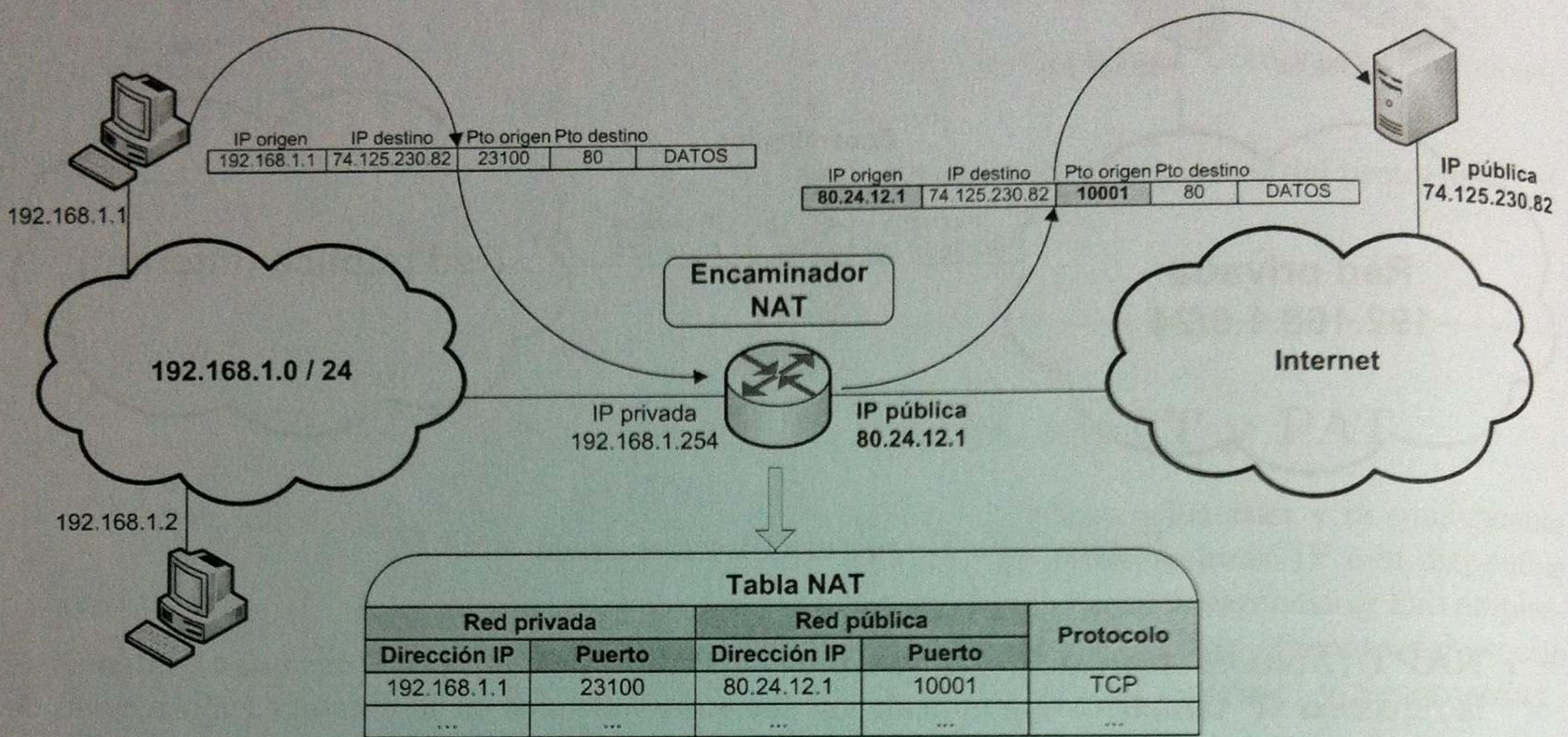


Figura 1.11: Tráfico saliente en NAT - Caso 1

El encaminador NAT empleará los puertos que tenga disponibles para seguir asignándolos al tráfico saliente, Figura 1.12.

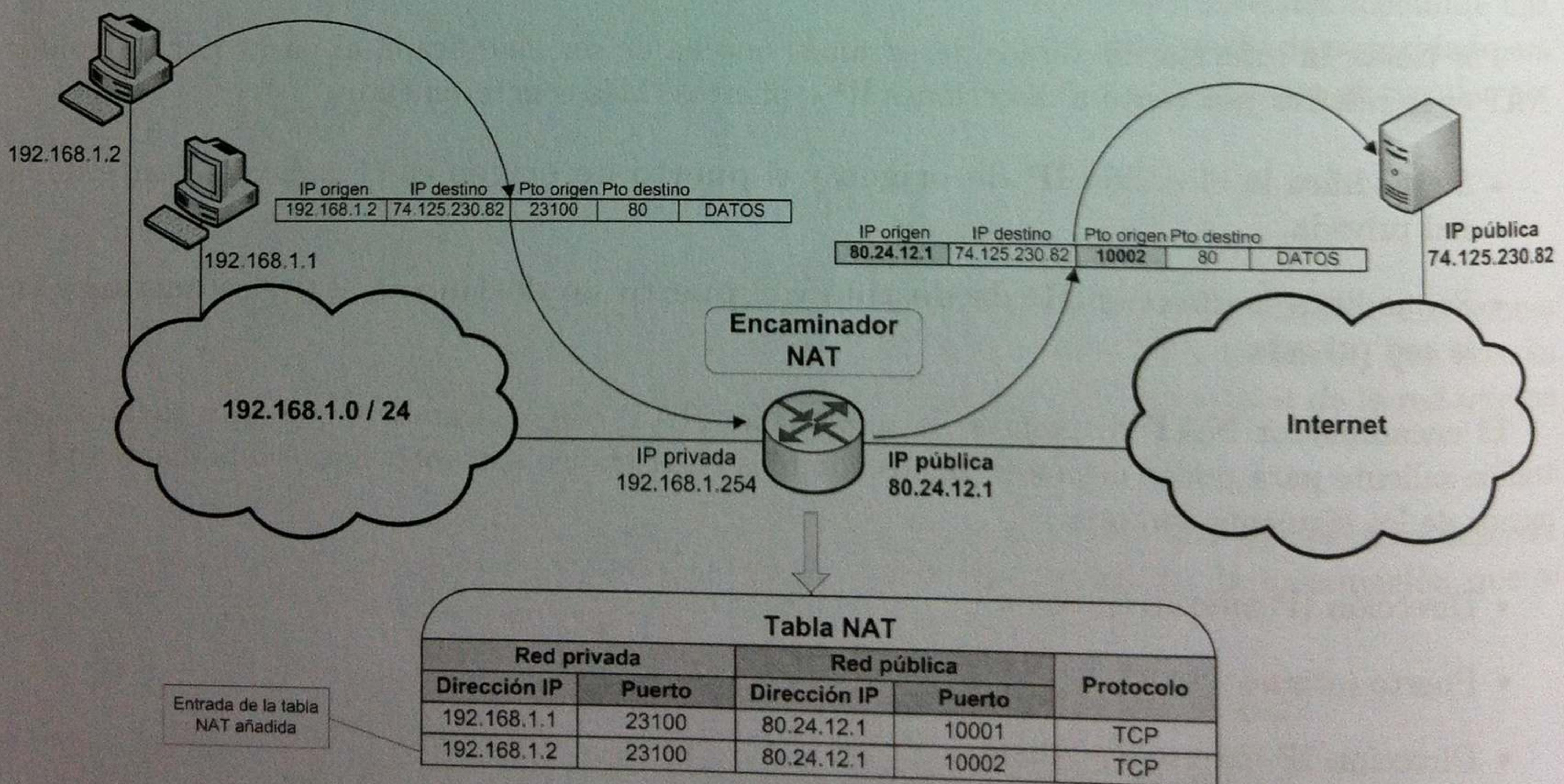


Figura 1.12: Tráfico saliente en NAT - Caso 2

1.5.3. Respuesta al tráfico saliente

Cuando al encaminador NAT le llega un datagrama del exterior, comprueba si la dirección IP de destino y el puerto de destino del datagrama entrante coinciden con algún registro de su tabla NAT. Si es así (**respuesta al tráfico saliente**), quiere decir que se trata de una respuesta a un datagrama saliente anterior, y en este caso, el encaminador NAT cambiará en el datagrama entrante la dirección IP de destino y el puerto destino por la dirección IP interna, y el puerto interno de la entrada de la tabla NAT cuyos campos **IP externa** y **Puerto externo** coincidan con la dirección IP destino y puerto destino que trae el datagrama entrante. A continuación retransmitirá el datagrama modificado hacia la red interna para que llegue a su destino, véase Figura 1.13.

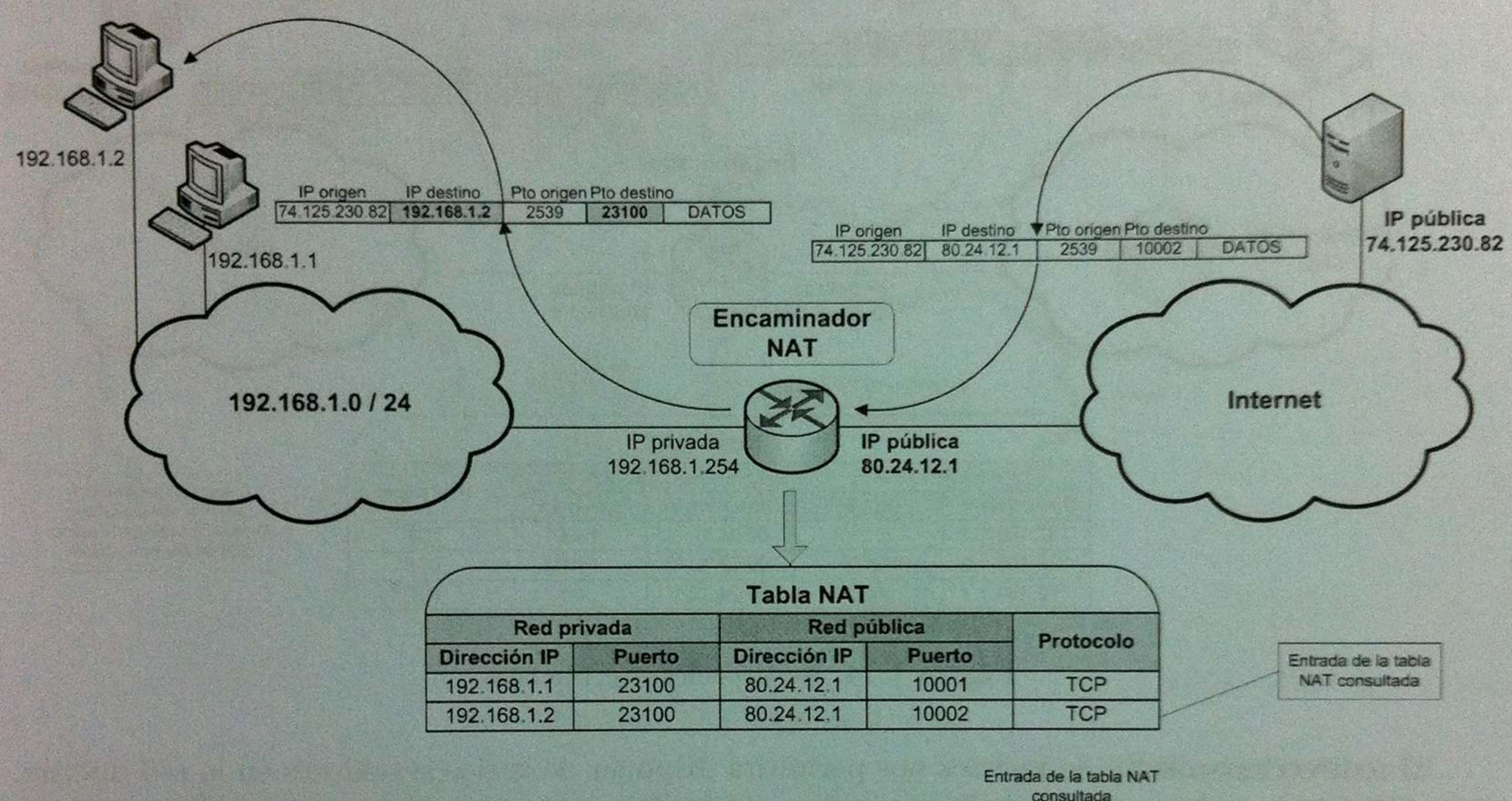


Figura 1.13: Respuesta al tráfico saliente

Si en cambio, el datagrama que le llega al encaminador NAT no se corresponde con un datagrama saliente previo (**tráfico entrante nuevo**), el encaminador NAT no sabrá a qué dirección IP interna y puerto redirigirlo y, en general, **descartará** el datagrama. Existen excepciones a este caso como por ejemplo, que el *router* NAT tenga un servidor escuchando en un puerto, en cuyo caso sí que aceptará los datagramas entrantes nuevos que vayan dirigidos a ese puerto de escucha. Por ejemplo, un servidor web escuchando en el puerto 80 del *router* NAT.

1.5.4. Soluciones al tráfico entrante nuevo - Redirección de puertos

El hecho de descartar por defecto las conexiones entrantes puede suponer una gran limitación en nuestra red interna, ya que si en ella disponemos de algún tipo de servidor nos va a resultar imposible permitir que clientes de la red externa se conecten a nosotros. Para subsanar esta carencia y a pesar del peligro potencial que supone dejar que se inicien conexiones desde la red exterior, es posible permitir las conexiones entrantes nuevas mediante el mecanismo de **redirección de puertos**.

La redirección de puertos (*Port Forwarding*) consiste en indicar al encaminador NAT una dirección IP de la red interna a la que redirigir todo el tráfico entrante nuevo. Para ello, deberemos añadir manualmente nuevas entradas en la tabla NAT que redirijan el tráfico entrante nuevo en función del puerto al que vaya dirigido. De esta forma todo el tráfico entrante dirigido, por ejemplo, al puerto 80 será reenviado a una dirección IP privada (y hacia el puerto que queramos) de nuestra red interna. Vemos un ejemplo en la Figura 1.14.

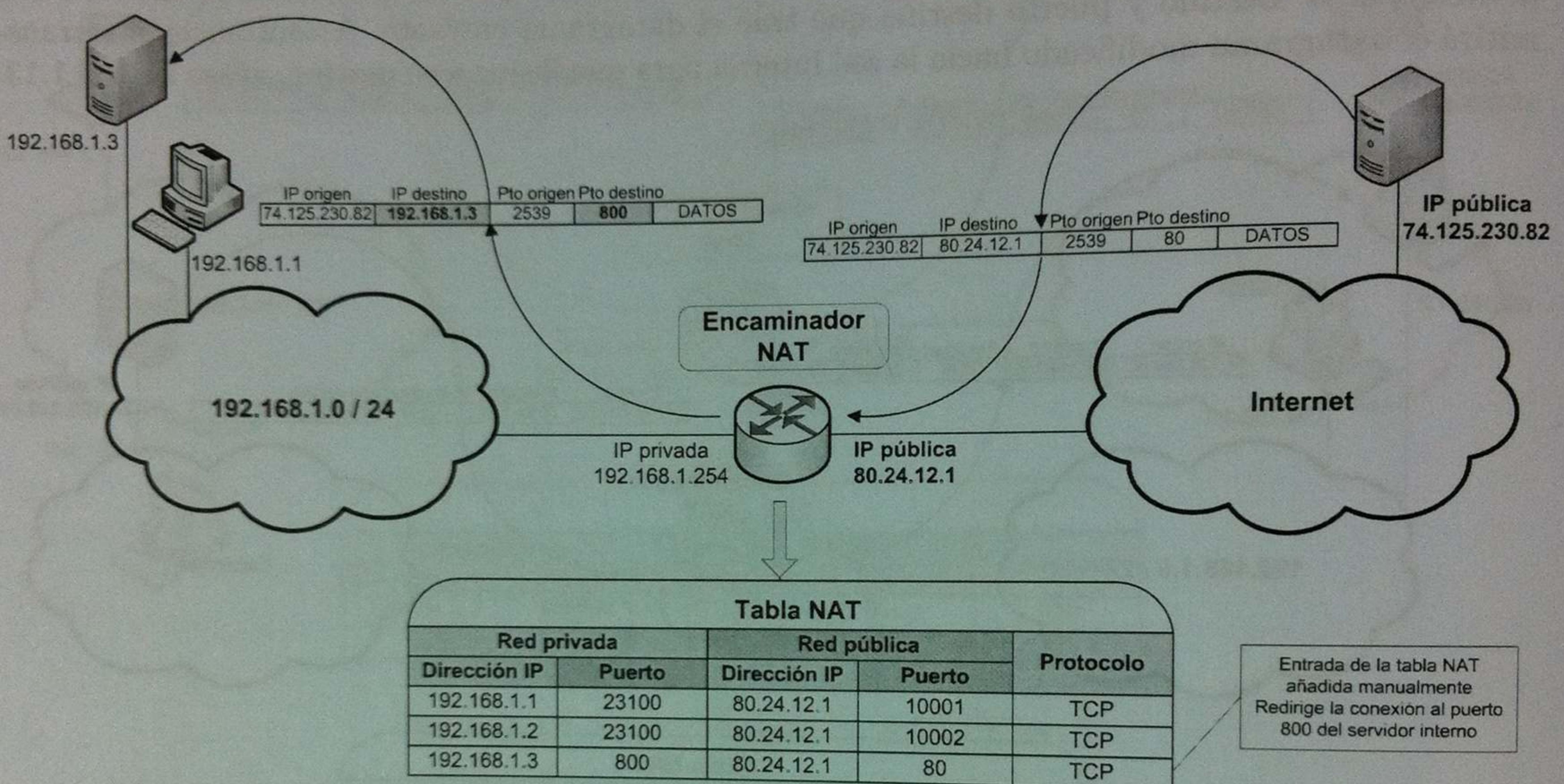


Figura 1.14: Redirección de puertos

El redireccionamiento de puertos nos permitirá disponer de varios servidores en la red interna, accesibles todos ellos a través de la dirección pública de nuestra red. Únicamente obligará a que, para cada servicio, se acceda a un puerto diferente.

1.5.5. Limitaciones de NAT

Fundamentalmente nos interesa el hecho de que hay protocolos de nivel de aplicación que incluyen las direcciones IP o los puertos empleados en la comunicación en el campo de datos del datagrama. Como el encaminador NAT solo actúa sobre las cabeceras de los protocolos IP, TCP y UDP pueden darse incongruencias en los datagramas que, por un lado, indiquen una dirección IP o un puerto en la cabecera y que luego esta no se corresponda con lo indicado en el campo de datos. Si esto sucede y la máquina de destino utiliza los valores que viajan en la parte de datos, no se identificará correctamente al origen.

Para subsanar esto los encaminadores NAT deberían conocer el protocolo concreto de nivel de aplicación encapsulado en cada datagrama para poder modificar también el campo de datos del mismo.

Por fortuna, las últimas implementaciones de NAT van incorporando poco a poco los protocolos más habituales que encapsulan direcciones IP y puertos en el campo de datos, poniendo remedio

a esta limitación aún a costa de añadir un retardo mayor al procesamiento de cada datagrama.

Por otro lado, el uso de NAT es escalable, es decir, podemos añadir tantas etapas de enruteadores NAT y redes privadas intermedias como queramos antes de alcanzar la red pública. Esto permitirá un mayor ahorro de direcciones IP públicas, Figura 1.15.

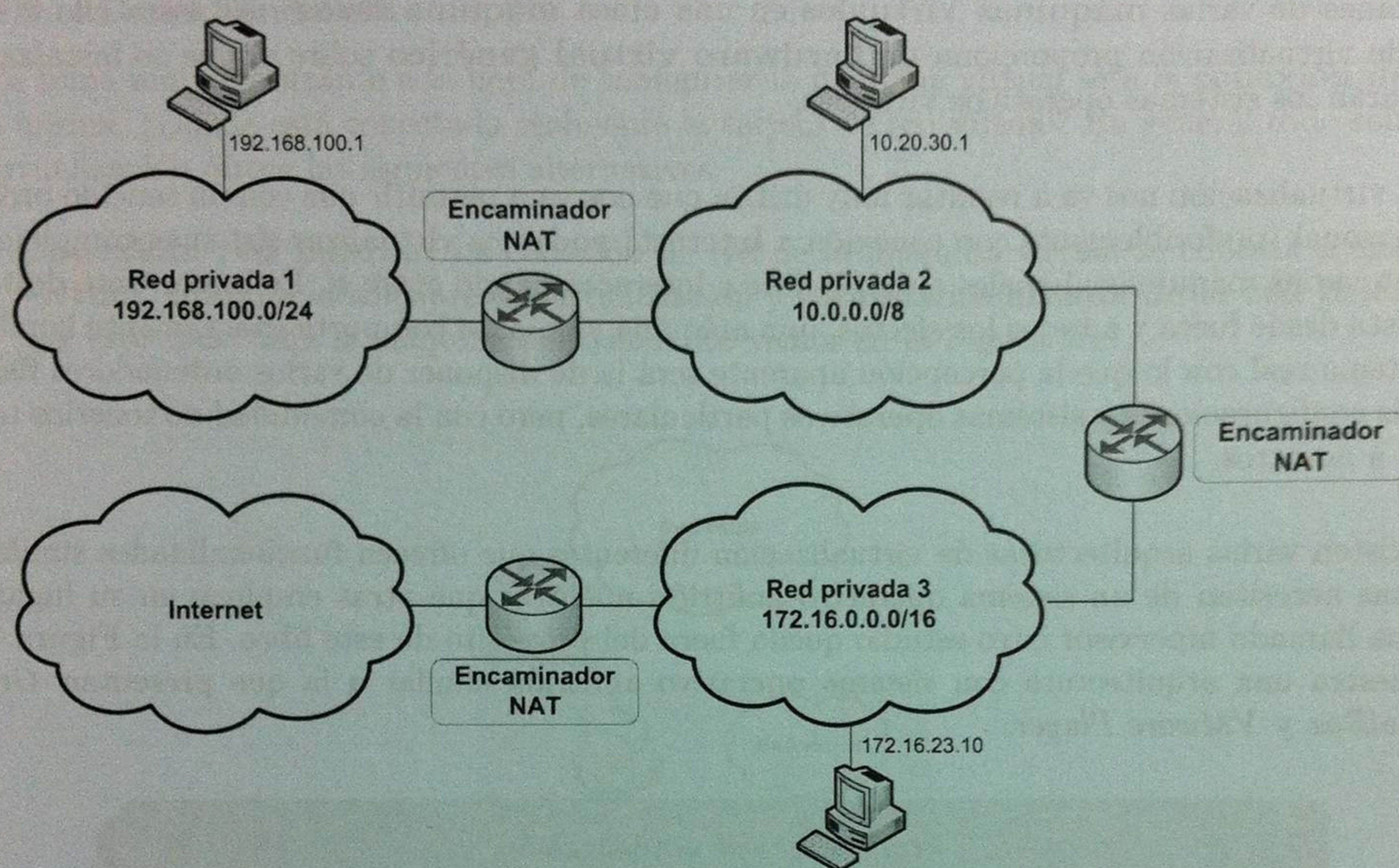


Figura 1.15: NAT en cadena

¿Sabías que ... ? Los proveedores de acceso a Internet (ISP, *Internet Service Provider*) emplean NAT para aumentar el número de direcciones IP disponibles para sus clientes. En la mayoría de los países, y siempre según la legislación vigente en materia de protección de datos, el proceso de NAT realizado por los ISP debe limitarse al transporte de datos entre la red externa e interna y, si no existe el consentimiento expreso del usuario final, no autoriza a realizar registros de uso o a implementar filtros de contenido. En cualquier caso es un tema controvertido a día de hoy por el conflicto de intereses existente, por un lado, entre los ISP y servidores web que desean un registro de sus usuarios, y por otro, el derecho a la privacidad de aquellos que utilizan estos servicios.

1.6. Software de virtualización

A lo largo del libro se plantearán muchas prácticas en las que el software de virtualización es casi imprescindible y por ello se van a dar una serie de nociones de cara a su utilización.

1.6.1. ¿Qué es la virtualización?

El origen de la virtualización se remonta a los años 60 donde se planteó como un recurso para optimizar el rendimiento de sistemas *mainframes* muy costosos, pero no es hasta finales de los años 90 cuando se introduce en el sector de los ordenadores personales con el mismo objetivo. En síntesis, el software de virtualización lo que permite es la ejecución simultánea de **varios sistemas operativos** virtuales en un solo hardware físico **anfitrión**. O lo que es lo mismo, la ejecución simultánea de varias **máquinas virtuales** en una única **máquina física** real. Para ello el software de virtualización proporciona un **hardware virtual genérico** sobre el que se instalarán y ejecutarán los sistemas operativos virtuales.

La virtualización nos va a resultar muy útil ya que nos va a permitir que con un sencillo ordenador personal (preferiblemente con conexión a Internet) podamos virtualizar sistemas complejos en red con varias máquinas virtuales ejecutándose e interaccionando entre sí. Es importante destacar que vista desde fuera y a todos los efectos, una máquina virtual se comporta exactamente igual que un sistema real con lo que la percepción aparente será la de disponer de varios ordenadores físicos, con sus configuraciones y sistemas operativos particulares, pero con la comodidad de tenerlos todos frente a nosotros.

Existen varias arquitecturas de virtualización diferentes que ofrecen funcionalidades similares. Algunas necesitan de un sistema operativo anfitrión mientras que otras emplean en su lugar un sistema llamado hipervisor cuyo estudio queda fuera del propósito de este libro. En la Figura 1.16 se muestra una arquitectura con sistema operativo anfitrión similar a la que presentan *Oracle VirtualBox* y *VMware Player*.

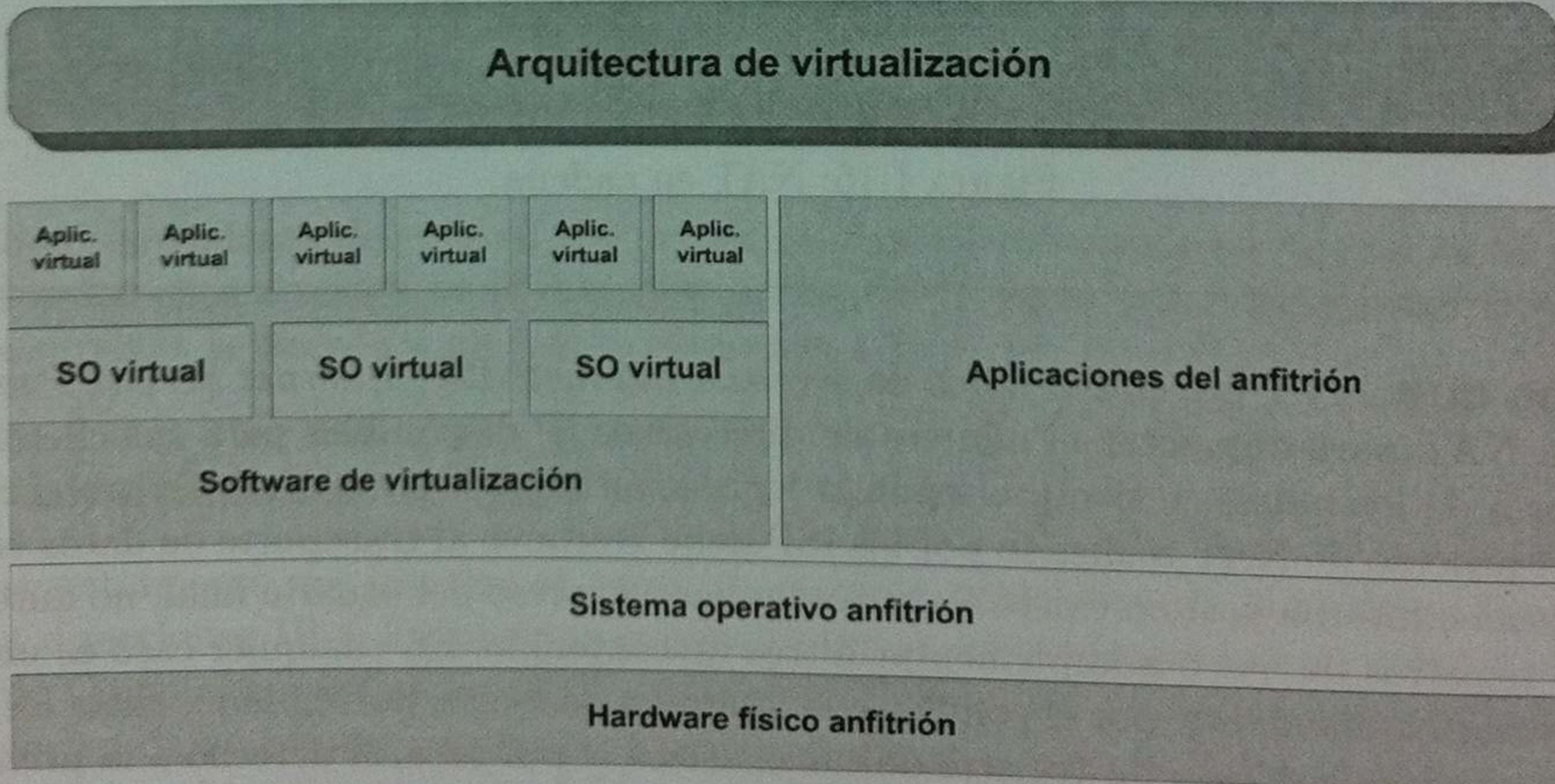


Figura 1.16: Arquitectura de virtualización

1.6.2. Configuración de las máquinas virtuales

Cuando se crea una máquina virtual debemos prestar especial atención a varias cuestiones importantes relativas al hardware virtual, especialmente la memoria RAM empleada por la máquina virtual para funcionar y las conexiones de red de las tarjetas virtuales.

1.6.2.1. Memoria RAM

La memoria RAM suele ser un bien escaso en los ordenadores, por eso si vamos a plantearnos muchas máquinas virtuales ejecutándose al tiempo debemos economizar su uso todo lo posible porque de lo contrario podríamos ralentizar demasiado el funcionamiento del sistema operativo anfitrión.

1.6.2.2. Conexión de red

Un tema muy importante a la hora de configurar la máquina virtual será la **conexión de red** de la misma. ¿Dónde está conectada realmente la tarjeta de red virtual? En general todo software de virtualización ofrece las siguientes alternativas:

- **Conexión: red interna.** La conexión de red de la máquina virtual se conecta a una **red virtual, interna** al software de virtualización y **totalmente aislada de la red física** a la que esté conectada la máquina anfitriona. Lo vemos en la Figura 1.17.

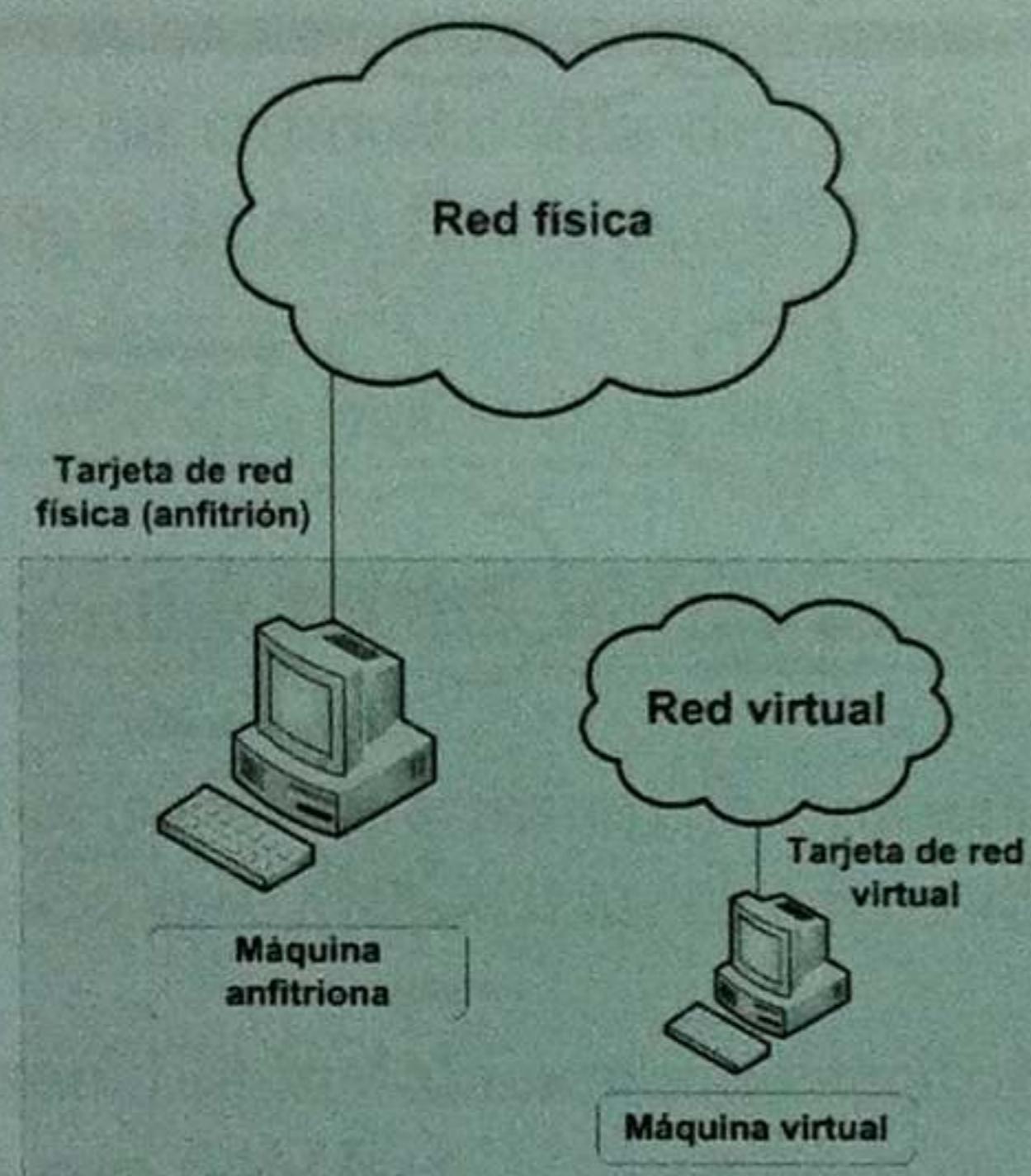


Figura 1.17: Conexión de Red - Red interna

- **Conexión: adaptador puente (*bridge*).** La conexión de red de la máquina virtual se conecta a la **misma red física** a la que esté conectada la máquina anfitriona real pero con **total independencia de la misma** en cuanto a configuración IP. De esta forma cualquier datagrama enviado o recibido por la tarjeta de red de la máquina virtual pasará por la tarjeta de red física de la máquina anfitriona, pero con una dirección IP diferente, aunque perteneciente a la misma red a la que pertenezca la tarjeta de red física. Lo vemos en la Figura 1.18.
- **Conexión: NAT.** La máquina virtual accede a la **misma red física** a la que está conectado el sistema anfitrión a través de la **propia dirección IP** del sistema anfitrión mediante NAT. Un **encaminador NAT virtual** gestionado por el software de virtualización se encarga de ello. Véase la Figura 1.19.

Básicamente, consiste en que la tarjeta de red de la máquina virtual se conecta a un segmento de red virtual al que también está conectado otro dispositivo virtual gestionado por el software de virtualización y que actúa como puerta de enlace hacia la red de la máquina anfitriona. Este dispositivo virtual actúa como encaminador NAT y también puede incluir

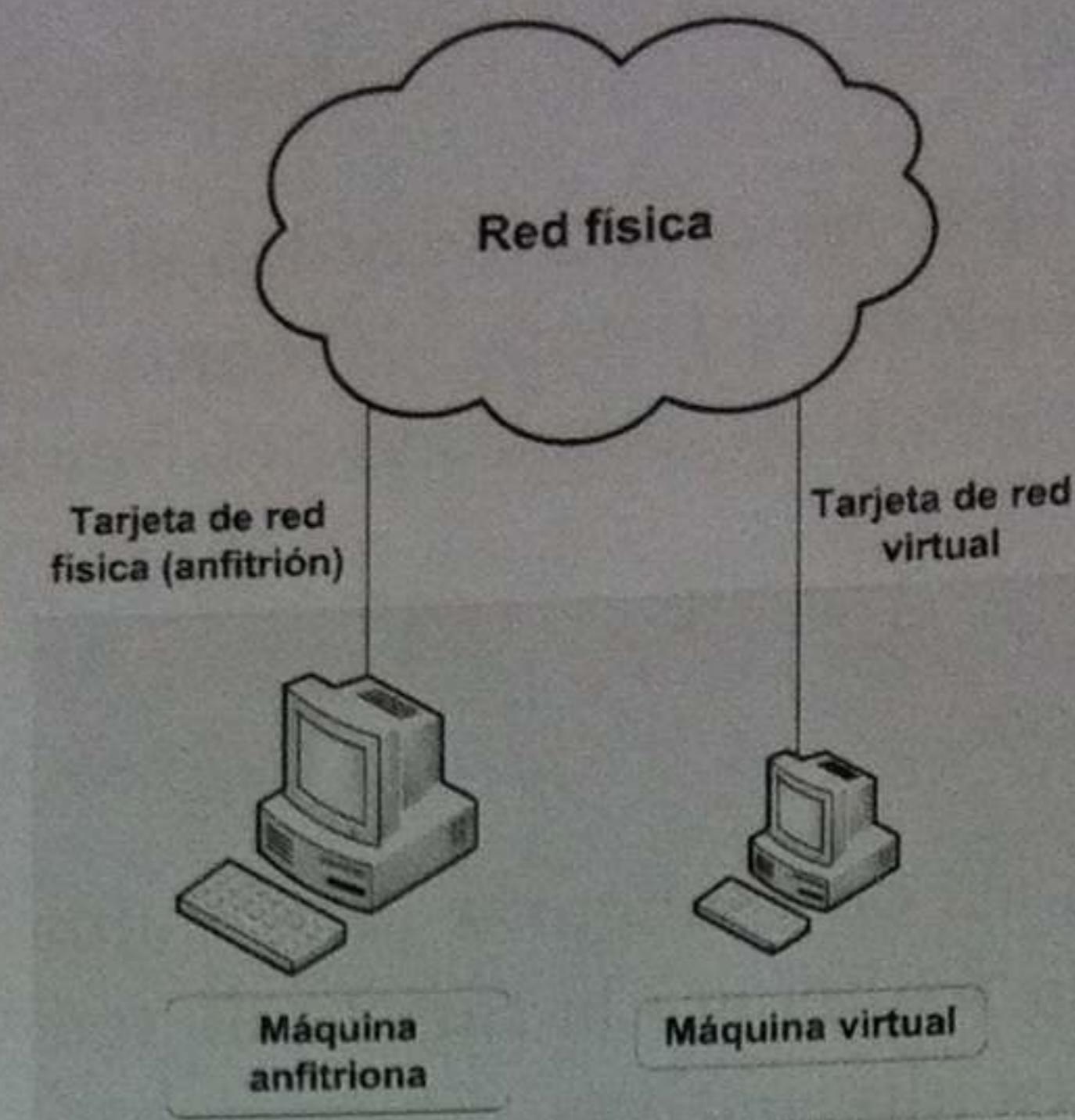


Figura 1.18: Conexión de Red - Adaptador puente

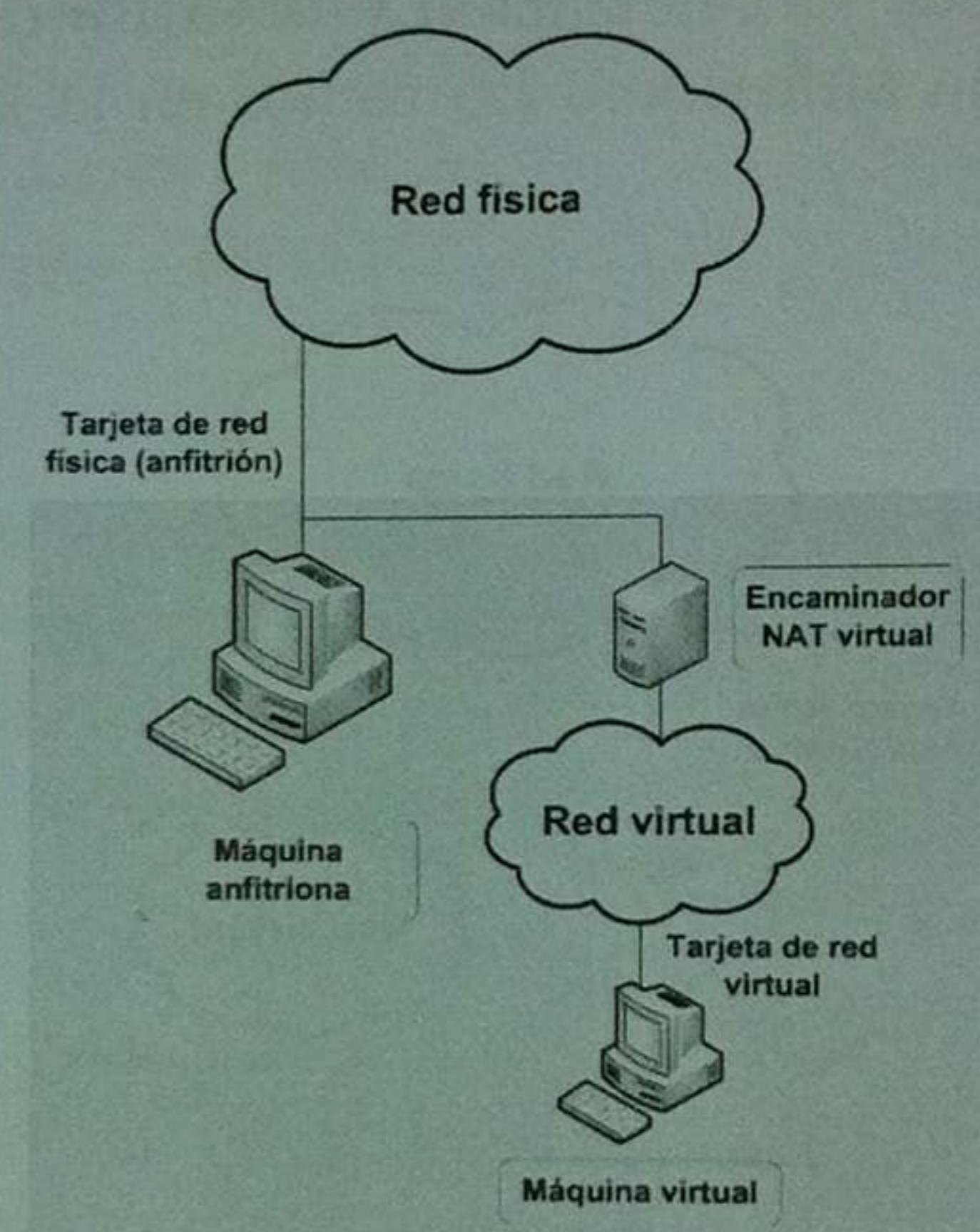


Figura 1.19: Conexión de Red - NAT

un servidor DNS y DHCP para facilitar la configuración de red de la máquina virtual.

Es importante señalar que, dependiendo del software de virtualización, la conexión de red virtual puede modificarse de una configuración a otra incluso en caliente, es decir, con la máquina virtual en ejecución.

1.6.3. Elección del software de virtualización

Hay diversas aplicaciones para virtualización disponibles en el mercado. Cada cual escogerá la que considere más adecuada para su caso concreto. Nosotros hemos trabajado y recomendamos las siguientes:

- **(Oracle) Virtual Box:** con licencia GNU GPL (*GNU General Public License*) que permite su libre uso para cualquier propósito sea comercial o no, así como su libre distribución y modificación (www.virtualbox.org).
- **VMWare Player y VMware Workstation:** ofrecen versiones gratuitas para uso personal no comercial (www.vmware.com).

1.7. Prácticas resueltas

En las primeras prácticas vamos a configurar las redes virtuales locales que se emplearán para el resto del temario. El software de virtualización empleado es indiferente aunque recomendamos *Oracle VM Virtual Box* o *VMware Workstation*.

Emplearemos 4 máquinas virtuales con sistemas operativos: *Linux Debian*, *Linux Ubuntu*, *Windows 7* y *Windows 2008 Server*). Estas máquinas formarán parte de la red local virtual con dirección de red **10.33.XX.0/24**, donde XX indica el puesto informático físico del aula. Así, en un aula de 15 ordenadores las redes locales virtuales irán de la 10.33.1.0/24 a la 10.33.15.0/24. Cada red virtual se comunicará con la red del aula a través de un enrutador NAT, también virtual.

En las prácticas sucesivas que se desarrollarán en este y otros capítulos del libro, supondremos siempre que nos encontramos en la primera red virtual del aula, 10.33.1.0/24.

En la configuración propuesta se ha supuesto una dirección de red para el aula, 192.168.0.0/16, que habrá que adaptar en cada caso a la red física existente.

La estructura de la red virtual que queremos conseguir es la mostrada en la Figura 1.20:

Recomendaciones para la configuración de las máquinas virtuales

Para las máquinas virtuales que se utilizarán en las prácticas, estas son las **necesidades mínimas** que hemos comprobado que garantizan el correcto funcionamiento de las mismas:

- *Linux Debian (6.0)*: una sencilla instalación sin escritorio gráfico como la que se plantea funciona adecuadamente con 64MB
- *Linux Ubuntu (10.04 Lucid Lynx)*: 256MB
- *Windows 7*: 512 MB
- *Windows 2008 Server*: 512 MB

Por tanto, si queremos tener funcionando simultáneamente las cuatro máquinas de la red virtual y también la máquina virtual correspondiente al *router* hacia el exterior, será imprescindible un ordenador anfitrión de al menos 4GB de memoria RAM para poder operar con las máquinas virtuales con cierta agilidad. En caso de no disponer de este hardware las prácticas planteadas son igualmente viables, solo que habrá que tener en cuenta la limitación en el número de máquinas virtuales en ejecución **simultánea**. En este sentido y teniendo en cuenta la función didáctica de las prácticas planteadas, puede resultar también recomendable desactivar las **actualizaciones automáticas** en los sistemas *Microsoft Windows* ya que sobrecargan innecesariamente el procesador y la red.

Evidentemente, también es posible emplear otros sistemas operativos diferentes a los aquí indicados. Los escogidos simplemente pretenden ser una muestra, creemos que significativa, de los existentes en el mercado.

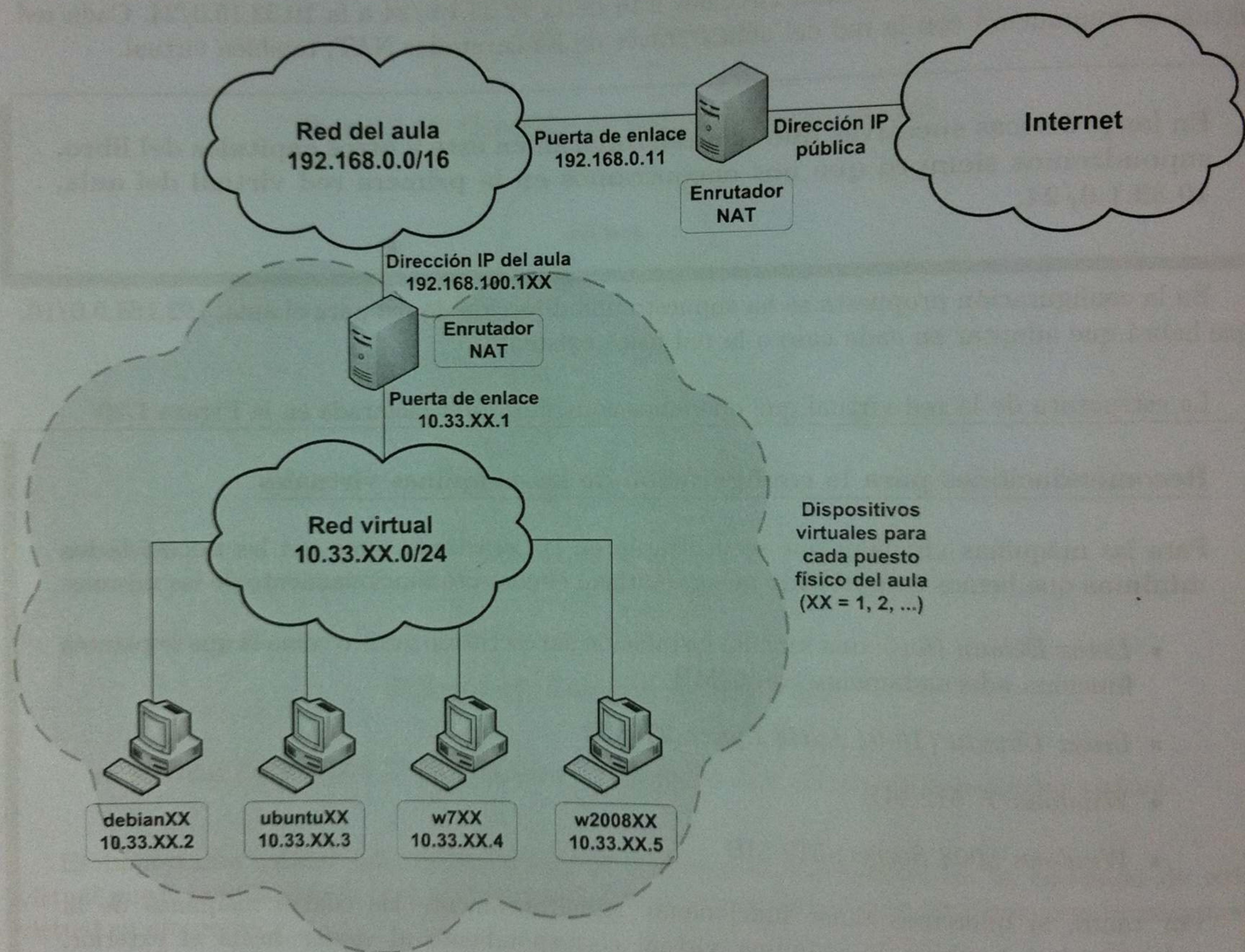


Figura 1.20: Red virtual