ÍNDICE GENERAL

CAPÍTULO 2

CAPA DE RED

2.1 Introducción

Si internet es un **conjunto de redes**, entonces la capa de red (el protocolo IP, como parte de esa capa) se encarga de **interconectar** esas redes.

Definición 2.1.1 Conmutación

Mandar tráfico entre nodos de red. Se suele usar en el contexto de paquetes, puesto que son unidades de información que mandamos entre routers, switches y hosts.

Definición 2.1.2 Encaminamiento

Encontrar la mejor ruta para llegar a nuestro destino. Cuando un paquete llega a un router, tiene una dirección destino que desea alcanzar. El router revisa su tabla de direcciones y le dice por dónde tiene que ir para llegar de forma transpartente y eficaz.

2.2 Conmutación

2.2.1. Conmutación de Circuitos

En la comunicación de circuitos tengo que tener un circuito dedicado previo a la comunicación. Es **orientado a conexión**, es decir, no va a empezar a mandar paquetes/información hasta que se cercione de que la conexión ha sido establecida.

Pasos

1. **Establecer conexión**: Conectamos los dos nodos (origen y destino) con un cable. Dicho de otra manera, **reservamos** los **recursos** para la comunicación.
2. **Transmisión**: Mandamos toda la información que queramos a través de la conexión establecida.
3. **Cierre de conexión**: **Liberamos los recursos** que habíamos reservado.

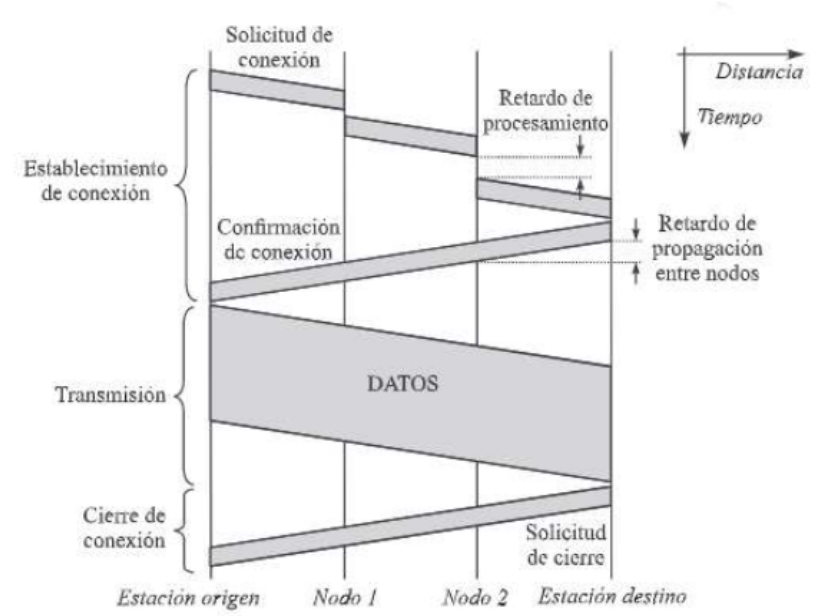

Figura 2.1: Conmutación de circuitos

Características

- **Pros**:
  - Al **no** ser un **recurso compartido** (una conexión por comunicación) tiene **mayor velocidad**.
  - Transmisión en **tiempo real** (porque no tenemos que esperar colas de paquetes).
  - **No** hay **contención** para acceder al medio: Como el recurso está reservado, no tenemos problemas de sobrecarga de recursos de red.
  - **Circuito fijo**, nos ahorramos el encaminamiento salto a salto.
  - **Administrar** nodos intermedios es más **fácil**, puesto que no necesitamos nada entre nodo y nodo más que la conexión.
- **Contras**: Tarda en conectarse y es difícil cambiar la dirección. Además es menos eficiente.

Un ejemplo de esta conmutación serían las llamadas telefónicas antiguas. Había terminales en los que unas operadoras se encargaban de conectar con cables las diferentes salidas y entradas según a dónde quisieras llamar.

2.2.2. Conmutación de Paquetes

> **Definición 2.2.1 Datagrama**
> Unidad de información en la que se divide el mensaje. Tiene cabecera de origen y destino, además de información para las distintas capas del modelo de red.

La conmutación de paquetes **no es orientada a conexión**, así que mandamos la información sin esperar una confirmación de que la ruta está completa. El destinatario podría estar o no disponible, pero nosotros los mandamos igual. Los paquetes pueden llegar **desordenados**. El **camino** puede

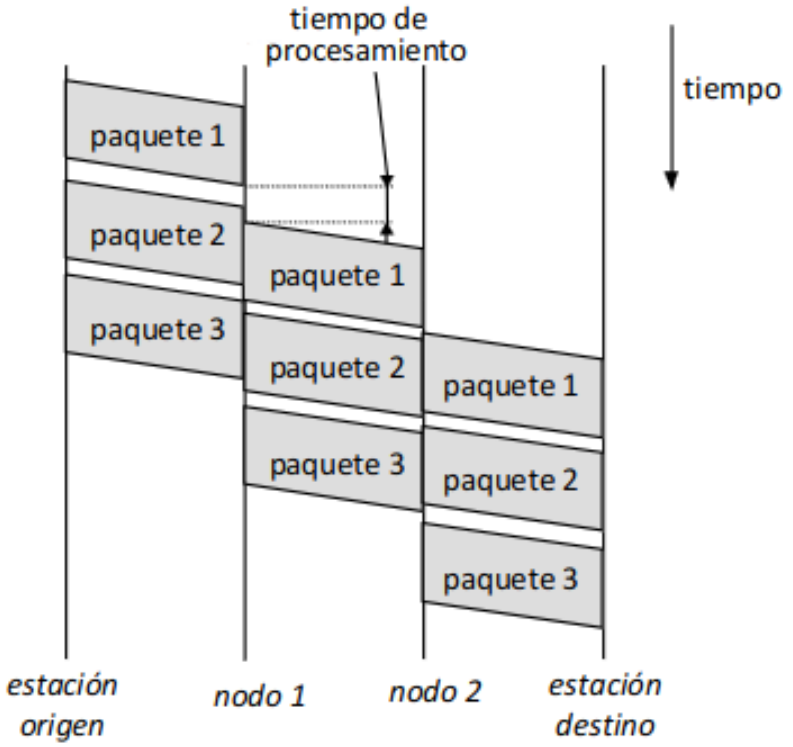

Figura 2.2: Conmutación de circuitos vs paquetes.

ser **diferente** entre paquete y paquete. Como hemos **dividido** el segmento en **paquetes**, hasta que **no llegan** todos **no se transmite**.

Existen **mecanismos de reenvío** de paquetes perdidos, por si alguno se queda atascado en la cola de un router o es rechazado por un camino que ha elegido.

En la conmutación de paquetes, **TCP** se encarga de **reordenar los paquetes** mientras que a **IP**

solo le interesa que **lleguen a su destino**.

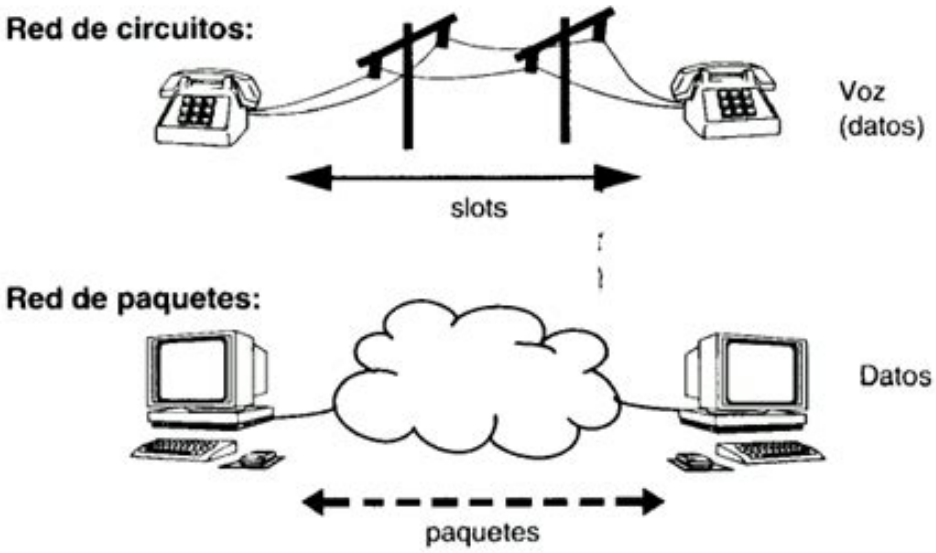

Figura 2.3: Conmutación de circuitos vs paquetes.

2.3 El protocolo IP

2.3.1. Introducción y características

Características

1. **Interconexión** de redes (llamadas subredes).
2. **Encaminamiento**: encontrar la ruta al destino.
3. **Salto a salto**. Involucra *hosts* y *routers*.
4. No orientado a conexión y no fiable
   - No negociación (*handshake*).
   - No control de errores, flujo ni congestión.
5. La unidad de datos se llama **datagrama** = cabecera + datos.
6. Protocolo de **máximo esfuerzo** ("*best effort*"), es decir, los paquetes se pueden perder, duplicar, retrasar o desordenar.
7. **Fragmentación**: Adaptar el tamaño del datagrama a las MTU de las subredes.

Definición 2.3.1 MTU

Las *Maximum Transfer Units* son cantidades máximas de datos que pueden ser transmitidos por una conexión de red sin fragmentar los paquetes. Determinan el tamaño mínimo de cada paquete en un protocolo de Internet (IP).

Cada entidad IP, se identifica con su dirección IP. Es decir, www.google.com sería 172.194.34.209. Las direcciones IPv4 (32 bits) tienen dos partes:

1. **Identificador de subred**.
2. **Identificador de dispositivo** dentro de esa subred.

Definición 2.3.2 Máscara de red

Patrón de 1s que determina qué bits de la dirección IP pertenecen al identificador de subred.

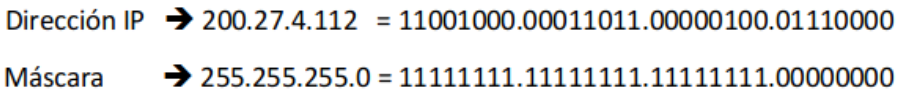

Figura 2.4: IP vs Máscara de subred. El **identificador de subred** sería 200.27.4.0

La máscara se puede representar diciendo el **número de 1** que tiene, el de la imagen sería 200.27.4.112/**24**. Para obtener el identificador de subred de una IP, solo tenemos que hacer una operación lógica AND sobre la IP con su máscara.

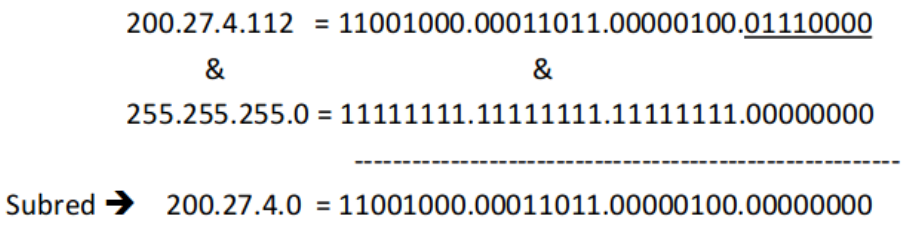

Figura 2.5: Operación AND

Definición 2.3.3 Subred

Rango de direcciones IP, delimitada por una Máscara de subred.

Definición 2.3.4 Switch

Dispositivo digital de interconexión de equipos que opera en capa de enlace. Interconecta dos o más host pasando datos de un segmento a otro según las direcciones MAC de destino.

Definición 2.3.5 Router

Dispositivo que interconecta redes con distinto prefijo en su IP. Establece la mejor ruta que destinará a cada paquete de datos para llegar al destino.

Nota Determinar subredes

Para determinar las subredes, separe cada interfaz de los hosts y routers, creando redes ais-

ladas. Dichas redes aisladas se corresponden con las subredes. Es decir, cada línea saliendo de un router corresponde con una subred distinta y todo lo conectado por un switch es una subred.

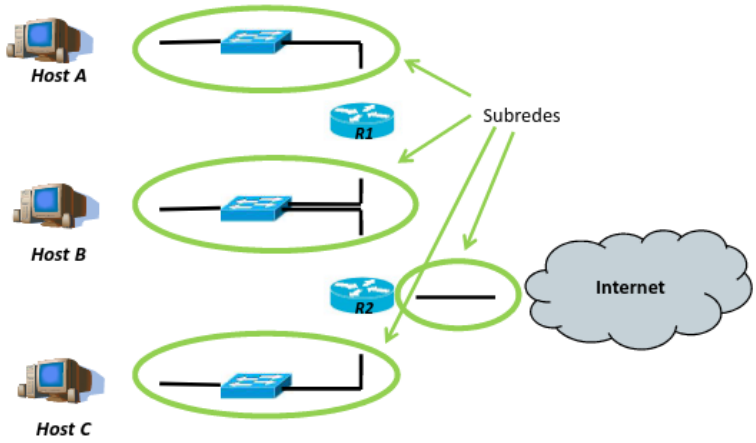

Figura 2.6: Subredes. Lo redondo son routers y lo cuadrado switches.

Fórmula 2.3.1 Máscara de subred

Para elegir la máscara tenemos que tener en cuenta el número de dispositivos de la red para no desaprovechar direcciones.

$$\#dispositivos = 2^{\#ceros} - 2$$

8 ceros en la máscara (/24) permite 254 dispositivos. Tanto la que acaba con todo 0 (**subred**) como la que acaba con todo 1 (**broadcast**) están reservadas y por eso se restan.

2.3.2. Direcciones IP (públicas y privadas)

Tipos

- Las direcciones **públicas** son un identificador **único en internet**. Se asigna una por dispositivo y **centralizadamente** (hay una entidad que lo ofrece). Son las direcciones de los **routers** que acceden a internet.
- Las direcciones **privadas** son un identificador **único en intranet**. Se pueden **repetir en distintas intranet**, las asigna el usuario según desee y solo sirven dentro de la misma intranet. Son las direcciones de **los hosts**.

Clases de IPv4

Las IP tienen 32 bits y son de una de estas clases:

Clases

1. Clase A
   - 128 redes (7 bit) x 2^{24} (24 bit) hosts
   - 10.0.0.0 red privada reservada clase A (24 bits de hosts)
2. Clase B
   - 2^{14} (14 bit) redes x 2^{16} (16 bit) hosts
   - 172.16.0.0 – 172.31.0.0, 16 redes privadas clase B
3. Clase C
   - 2^{21} (21 bit) redes x 256 (8 bit) hosts
   - 192.168.0.0 – 192.168.255.0 256 redes privadas de clase C
4. Clase D: Sirve para multicast. Permite el envío simultáneo de información a varios usuarios de una red desde un punto o nodo. A diferencia del broadcast, los destinatarios son previamente seleccionados por el emisor. Esto significa que el envío está **restringido** y no todos los usuarios de una red reciben los datos.
5. Clase E: sirve para cuando se "gasten" las IP. Tienen los 5 primeros bits reservados y lo demás está por definir.

Existe una IP especial llamada **loopback** que sirve para que el ordenador se mande mensajes a sí mismo. Existe para poder permitir que un router tenga una IP cuando se encienda por primera vez

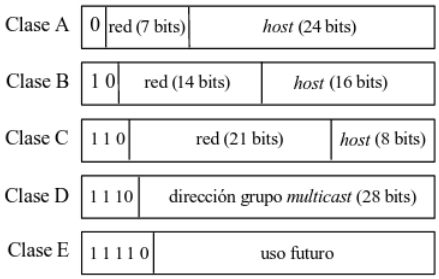

Figura 2.7: Estructura de las clases de redes.

2.3.3. NAT

Definición 2.3.6 NAT

Reasigna direcciones IP (normalmente privadas) a otras (públicas) de los paquetes cuando se transmiten por un router.

Cuando estamos en una intranet, tenemos una dirección, pero cuando pasamos a internet, esas direcciones no nos valen. Por eso tenemos que traducir (Network Addres Translation) nuestra IP a otra que les sirva a los demás routers para encaminar.

Nota **Sobre el NAT**

*Para traducir IP, usa una **tabla de traducciones** que permite hacer y deshacer cambios de IP. **No** se puede **instalar un servidor** con una **dirección privada** porque no sería accesible desde internet. En las aplicaciones web (como whatsapp, en la que hablan dos teléfonos de IP privadas), tienen una **tabla de traducciones en el mismo servidor***

Proceso NAT

1. Cliente intenta conectarse al servidor con su IP privada.
2. NAT añade nueva entrada a la tabla de traducción, cambia IP por pública, el puerto y cambia el chequeo de la integridad.
3. NAT recibe paquete del servidor, traduce la IP pública y su puerto por la privada y se lo da al router.

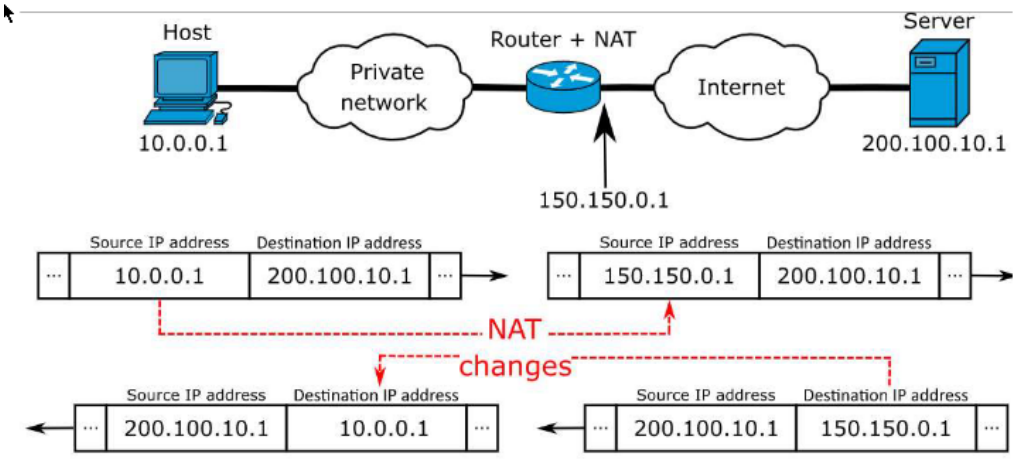

Figura 2.8: Funcionamiento de NAT con tráfico entrante y saliente.

2.3.4. Asignación de direcciones

Queremos ajustar las direcciones para tener el **mínimo desperdicio** (mirar imagen 2.9). Si me dan una red /24 privada puedo ajustar las máscaras para repartir las direcciones con la previsión de ejercicios especificada. Por ejemplo, imaginemos que tenemos 30 dispositivos en 3 subredes. Pues para cada subred necesitaremos 5 ceros ($2^5 - 2 = 30$, estando 11111 y 00000 restringidas). Como necesitamos 5 ceros para cada subred, las máscaras tendrán /27 (27 unos).

En la red pública, por otro lado, necesitamos mínimo 2 IP: una para el router R2 y otra para el router de internet. Teniendo en cuenta que la red de todo 1 y la de todo 0 están reservadas, necesitamos 4 ips. $2^2 = 4$, así que con 2 ceros nos valdría.

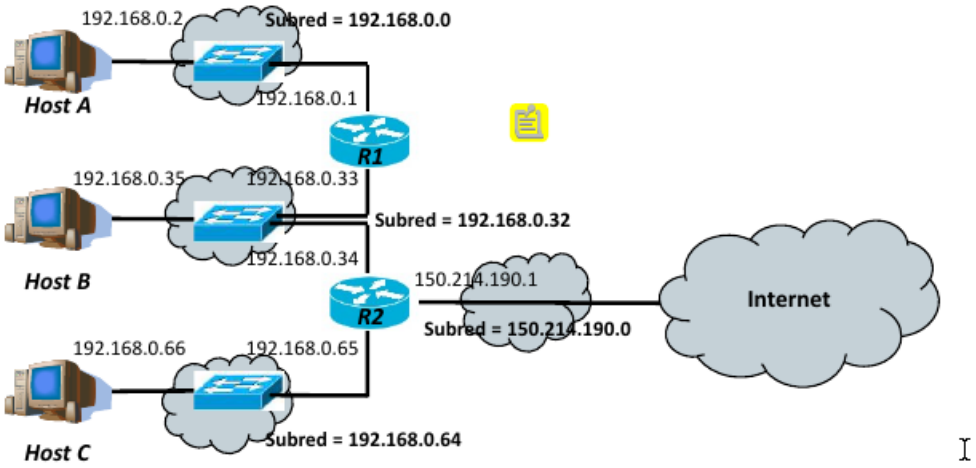

Figura 2.9: Asignamiento de IPs según los dispositivos que tenemos en nuestras subredes.

2.3.5. Encaminamiento

Ahora que ya tenemos asignadas las IP, vamos a hablar del **encaminamiento**. Cómo se asignan las direcciones afecta al encaminamiento. Se realiza **paquete a paquete**, es decir cada paquete se encamina independientemente. Se hace en base a dos inputs:

- La **IP destino**, que va en la cabecera de cada paquete.
- La **tabla de encaminamiento**, que asocia destinos con rutas.

Cabe añadir que cada host y router tendrá **una IP por subred de la que sea parte**. En el móvil si tenemos una wifi, si tenemos un operador móvil, tendremos dos IP.

Definición 2.3.7 Encaminar

Encaminar = elegir el salto siguiente en un router o un host.

Definición 2.3.8 Store and forward

Cada paquete se almacena en los recursos locales de cada router, se inspecciona, se le saca la cabecera de IP destino y según la tabla de encaminamiento se enrutan hacia otro nodo. Se realiza salto a salto, paquete a paquete.

Modos de encaminamiento

- **Encaminamiento/Routing directo**: No necesito un router directo para acceder a ellos (no hay intermediario) Ej. H1 y H2
- **Encaminamiento/Routing no directo**: Rutas o destino sque impliquen la intermediación de una entidad IP colaboradora que típicamente es un router. Ej: H1 y H3 (Imagen de abajo, número 2.10)

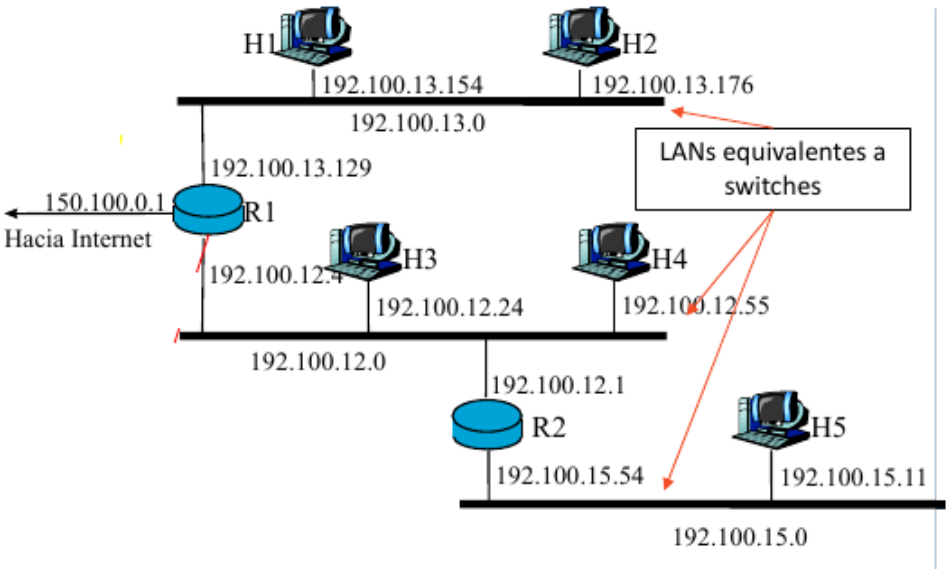

Figura 2.10: Ejemplos para routing/encaminamiento.

Elementos de tabla de encaminamiento, fig 2.11

1. **Destino**: (posibles destinos para alcanzar) puede tener IP explícitas de 32 bits o prefijos de red.
   - 127.*.*.* es el **loopback**, red clase A reservada. Para que el SO de router pueda arrancar teniendo una IP aunque sea ficticia.
   - Redes **directamente conectadas también aparecen** en la tabla (R1 tiene 3)
   - **Default** es un mecanismo para simpificar las tablas de encaminamiento. 0.0.0.0 indica cualquier otra IP no explícitamente indicada en entradas previas.
2. **Gateaway**/ Salto siguiente: Para cada destino por donde tengo que ir. Loopback y las redes directas tienen * porque no hay intermediario, no hay nada a donde saltar.
3. **Máscara**: Cada fila tiene asociada una máscara para saber qué bits pertenecen a la red y qué bits pertenecen al host. Siempre se elige la entrada más explícita, el default es la última que se elige.
4. **Flags**: no son tan relevantes de cara al examen.
5. **Interfaz**: ethernets o wifis o lo que sea por los que se conecta la red

Pasos encaminamiento

1. Se **extrae** la **Ip destino** del datagrama
2. Se va sacando la IP con las **máscaras** de la tabla.
3. Si **coincide y** es routing **directo**, lo manda por la interfaz necesaria. Si **coincide y no es directo**, lo manda al siguiente salto.
4. Si hay varias coincidencias se elige la **máscara más larga**.
5. Si **no hay coincidencia** se manda mensaje de **error**.

i	Destino (D_i)	Salto siguiente (S_i)	Máscara (M_i)	Flags	Interfaz(I_i)
1	127.0.0.1	*	255.255.255.255	H	lo
2	192.100.12.0	*	255.255.255.0	-	eth0
.	192.100.13.0	*	255.255.255.0	-	eth1
	150.100.0.0	*	255.255.255.252	-	eth2
.	192.100.15.0	192.100.12.1	255.255.255.0	G	eth0
N	Default	150.100.0.2	0.0.0.0	G	eth2

Figura 2.11: Tabla encaminamiento R1

Pasos para diseñar tabla de encaminamiento

1. Redes directamente conectadas (sin siguiente salto, con interfaz y la máscara de la subred)
2. Entrada por defecto (gateaway por defecto normalmente suele ser el router de la subred a la altura de un host).
3. Añadir todas las entradas adicionales (subredes no directamente conectadas o a las que no se accede por el default) necesarias.

Criterios para asignar direcciones IP

- Reducir el número de direcciones IP desperdiciadas (ajustando las máscaras, como lo que hicimos con /27)
- Reducir el número de filas de las tablas de encaminamieto (Haciendo a las subredes agregables).

2.3.6. Sistemas autónomos

Definición 2.3.9 Sistemas Autónomos

Es un **conjunto de redes y routers administrados por una única autoridad** que define cómo es el intercambio de tablas (routing interno) dentro del SA. En cada SA existe un router, denominado **router exterior**, responsable de informar a los otros SAs sobre las redes accesibles a través del SA. Son un **nivel de jerarquía** en la red. Por ejemplo, todas las universidades de españa son un sistema autónomo.

2.3.7. Intercambio automático de tablas: RIP y OSPF

Niveles intercambio de tablas

- **Protocolos IGP**: El administrador elige el protocolo. Intercambio de tablas entre routers **dentro** del SA. RIP y OSPF son de esta categoría
- **Protocolos EGP**: es protocolo externo y es una norma única y obligatoria. Intercambio de información entre routers externos de sistemas autónomos. BGP es de esta categoría.

RIP Routing Information Protocol

Características

- Protocolo de la capa de aplicación (opera sobre UDP puerto 520).
- Reserva una dirección para poder enviar mensajes a todos los routers del sistema autónomo (224.0.0.9)
- Todos los routers necesitan saber cómo llegar a todos los destinos del sistema automáticamente. Para eso, se usa el *vector distancia*.
  - Se usa como métrica el **número de saltos** para llegar al destino.
  - Busca de todos los posibles orígenes a todos los posibles destinos con el menor número de saltos.
- Por los bucles (explicado abajo) existen algunas soluciones que los arreglan:
  - **Split horizon**: Ningun router acepte vectores distancia que vengan por el camino que usariamos para llegar a ese destino. R2 no acepte anuncios por el mismo camino que yo uso para ir al destino.
  - **Hold down**: Cuando detecto un fallo ignoro los vectores distancia durante un tiempo.
  - Poison reverse.

Nota **Ejemplo de vector distancia**

Imaginemos que un router tiene 4 vecinos en su sistema autónomo.

Periódifcamente (30 segundos) hay un intercambio RIP. Los vecinos informan sobre todos los posibles vectores distancia, es decir (destino, coste asociado [en número de saltos]).

*Así, cada vecino dice que para llegar a x, tarda y saltos. El **R1** tarda 3, el **R2** tarda 2, el **R3** 7 y **R4** tarda 5. Elegimos el **R2** porque es el mas chiquito como gateaway. Aprendo que para ir a x voy a tener un coste 2 +1 salto más a través del **R2**. Eso periódicamente cada 30 segundos.*

*Si **R2** deja de funcionar, la red deja de estar accesible. RIP tiene ese tipo de problemas. Vienen bien para aprender las buenas noticias, pero se tarda mucho en propagar las malas. En algunas topologías aparecen **bucles**.*

|red 1| - R1 - R2 - R3

***R1** diría que para llegar a la red 1 tengo una ruta de coste 0, **R2** de coste 1 y **R3** de coste 2. Si el enlace entre red1 y **R1** se corta, aparece un bucle porque **R1** dice que ya no puede*

*acceder a la red 1, pero a los 30 segundos le dirá **R2** que para llegar a red 1 tiene una ruta de coste 1. Así que creerá que a partir de **R2** podrá llegar a la red 1.*

OSPF Open Short-Path First

Características

- Usa como métrica la **función de coste inversa a la velocidad de transmisión**. Aquí es α 1/Vt. Si el coste es 1Mbps por arriba y 3 Mbps por abajo, el inverso de 3 es menor que de 1, así que elijo el de abajo.
- Divide el sistema autónomo en otro sistema de jerarquía con **áreas** (grupitos dentro del sistema). Tiene un área central (**área cero**) de la que ramifican todas las demás (topología en estrella).
- Utiliza el algoritmo de **estado del enlace**. Cada router informa a todos los routers del area sobre el coste hacia sus vecinos. Cuando tengamos el grafo, aplicamos **Dijkstra**,

2.3.8. Formato de los datagramas IP

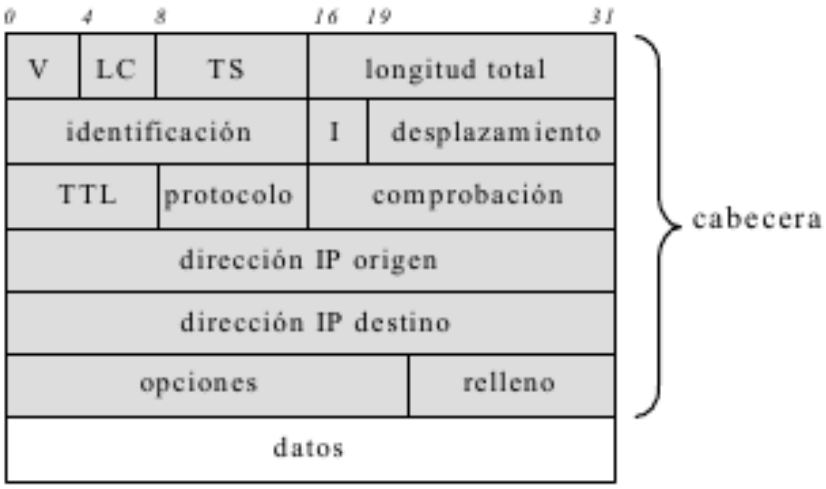

Figura 2.12: Componentes de la cabecera IP

Tiene cabecera + datos. Todas las cabeceras IP tienen 20 Bytes. Si hay opciones serán un poquito más.

Partes

- **IP origen /destino**: según estas dos van a tomar una decisión u otra los routers y no cambian . Lo que cambian son las MAC salto a salto. La IP si no hay NAT no cambia.
- **TTL**: Para evitar que los paquetes se queden infinitamente en un router, por cada salto restamos uno al TTL. Si llega a 0 los routers dicen que el paquete es huérfano y lo descartan. Por cada segundo esperando en la cola también se decrementa el TTL.
- **El campo protocol**: Nos dice cuando lleguemos al destino a quién se lo damos (TCP, UDP, HTTP...) Nos dice lo que hay dentro del paquete. En base al puerto se lo dará al proceso adecuado.
- **Checksum**: coge la cabecera y la pone en palabras de 16 bits. Luego, hace una función lógica (exclusive OR). El resultado permite detectar errores. Si hay errores en una de las palabras y algo se convierte en 0 o 1, habrá un **mismatch**. Compara la operación y si no coincide el resultado lo descarta. Detecta cualquier número impar de errores, un número par no lo detecta. Es una estrategia muy pobre de detección de errores.
- Dejamos **desplazamiento** e **indicadores** para la siguiente sección.

2.3.9. Fragmentación

Las cabeceras me dicen la **longitud total del datagrama**. El paquete más grande de internet es de 64KB. Cuantos más datos pueda meterle mejor, con la misma cabecera transmito más datos. Hay que recordar que en cada salto puede haber una tencología diferente (red móvil, cable de fibra, satélite...) y cada una tiene un MTU (maximum transfer unit). EL host hace los paquetes con el tamaño máximo del primer salto. Puede ocurrir que en un salto implica una MTU menor y no cabe. IP tiene la capacidad de resolver este problema **fragmentándolo**. Se utilizan los siguientes campos:

- **Desplazamiento**: offset respecto del comienzo del paquete.
- **Indicadores** (I): ”Don’t Fragment”, ”More Fragments”.

Todos los fragmentos **replican** el campo **identificacion** pero **añaden** un **offset** para poder **ordenarlos** cuando lleguen. Se reordenan por tamaño de offset de menor a mayor y se ensamblan solo en el destino con un bit llamado MF (more fragments, si está a 0 significa que es el final).

La identificación es la misma que el fragmento original si fragmentas un fragmento.

2.4 Asociación con capa de enlace: el protocolo ARP

¿Como sé a partir de una IP la dirección física? Para resolverlo automáticamente usamos ARP (adress resolution protocol). Cuando quiero ir de Host A a Host B, voy cambiando las MAC (direcciones físicas). Tras consultar la tabla de encaminamiento tengo que mandar el **datagrama a la MAC siguiente**. Las MAC son **direcciones ethernet** = direcciones WIfi, identifica a dispositivos en **capa física**. Son 6 Bytes y se representan separados de byte a byte en hexadecimal. Son **únicas en el mundo**. Los 3 primeros bytes identifican al fabricante y cada uno le pone una MAC distinta a cada tarjeta. Vienen en la eeprom. Hace falta la traducción IP->MAC. ARP es un protocolo que va en trama física. Funciona así:

Funcionamiento de ARP

B quiere hablar con D, hace un mensaje en broadcast (todo a 1 FF-FF-FF-FF-FF-FF) Cuando una trama la ponemos en la red con esa MAC destino todos los equipos son destinatarios de esta trama.

Ese mensaje es una pregunta: ¿quién sabe la MAC de D?(a)

Cuando le llega a D, él responde con un mensaje ARP automático con la MAC que B necesita. Tanto la consulta como la respuesta son mensajes ARP.(b)

Ya de paso, a todos los informa con su MAC B. Así se van gestionando dinámicamente. (a) y (c) serían el funcionamiento de RARP.

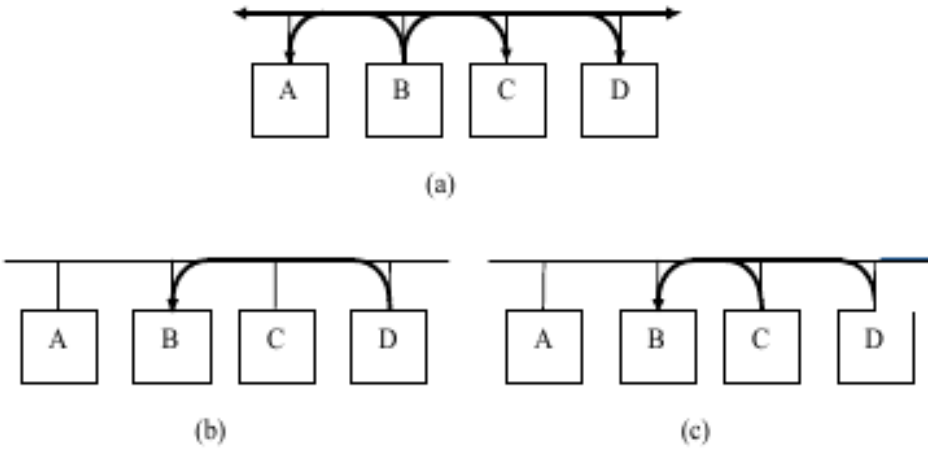

Figura 2.13: Imagen suplementaria al funcionamiento de ARP

Formato ARP

- Se pone la IP y en la respuesta viene la MAC.
- Tiene la dirección del emisor y la IP (H de Hardware, P de IP).
- En la imagen tenemos un request de saber la MAC de una IP y aprovecho para informar sobre mi IP y mi MAC. Todo el mundo aprende la asociación de IP con MAC.

2.5 El protocolo ICMP

Sirve para el **control y notificación de errores** del Protocolo de Internet. Como tal, se usa para **enviar mensajes de error**, indicando por ejemplo que un servicio determinado no está disponible o

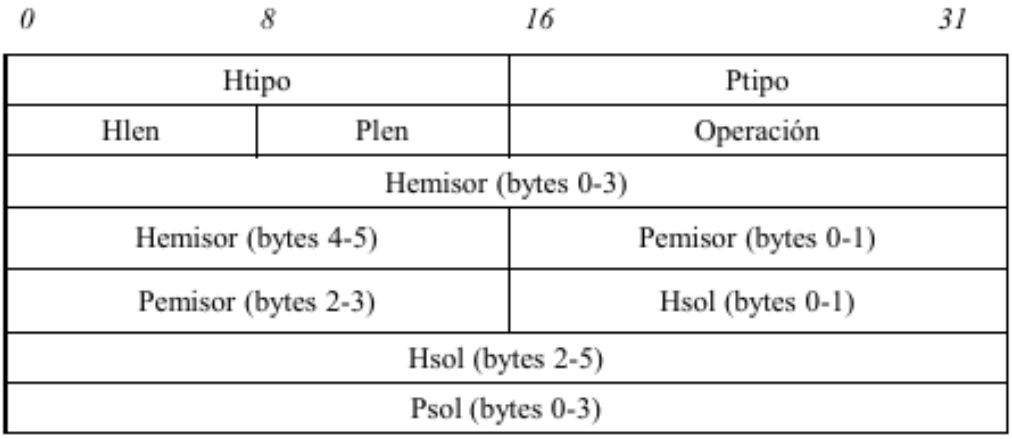

Figura 2.14: Formato de mensaje ARP

que un router o host no puede ser localizado. Si le quitas el default a tu router llegan mensajes de destino inalcanzado, por ejemplo. Se encapsula en IP.

- **Tipo**: Tipo de error.
- **Código**: Subtipo de error.
- **Checksum**: visto anteriormente, operación para detectar errores de bit impares.

2.6 Autoconfiguración de la capa de red (DHCP)

Nadie quiere hacer todo a mano. DHCP es el mecanismo en el que se puede c**onfigurar un nuevo dispositivo automáticamente**.

DHCP es un protocolo sobre UDP que tiene el **puerto 67 reservado**.

Pasos DHCP

1. **DHCP DISCOVER**: descubrir quien es el servidor, le llega a todo el mundo, es un *broadcast*.
2. Le contesta el servidor DHCP poniendo su ip origen sin saber el destino, hace un broadcast **OFFER**. Tiene una propuesta de IP y se la deja durante una hora.
3. El host solicita la IP, con un mensaje **DHCP REQUEST**.
4. El server le otorga la dirección con un mensaje **DHCP ack** (acknowledgement).