

# TEMA 3. La capa de red. Protocolo IP



Profesora: Guadalupe Miñana Ropero

Transparencias basadas en las de la asignatura de Redes de los grados de Informática de FDI

# La capa de red

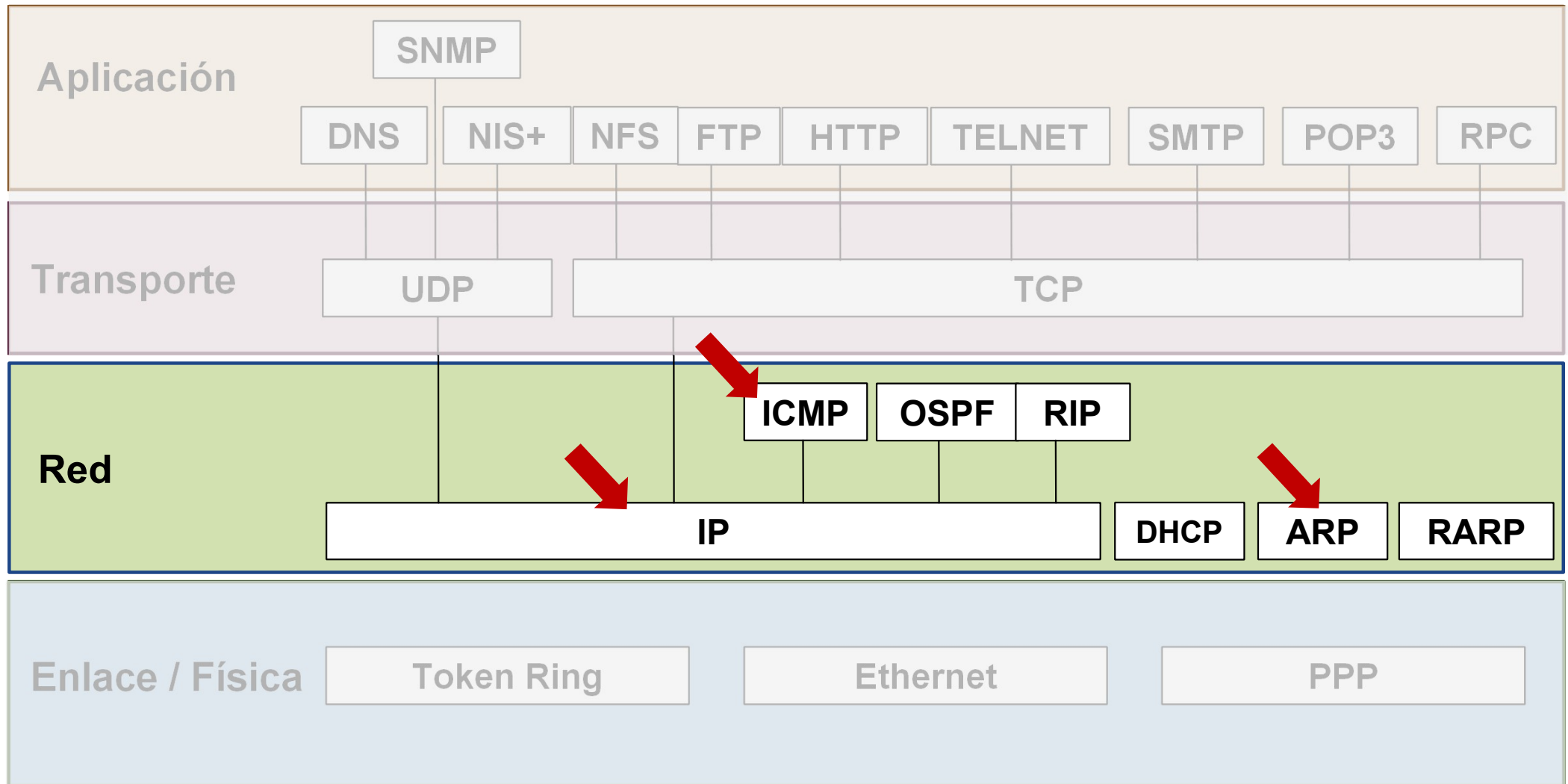
## Protocolo de red de Internet (IP)

- Proporciona un servicio básico de entrega de paquetes
- Protocolo **no orientado a conexión y no fiable**
  - No realiza detección ni recuperación de paquetes perdidos o erróneos
  - No garantiza que los paquetes lleguen en orden
  - No garantiza la detección de paquetes duplicados

## Funciones básicas del protocolo IP

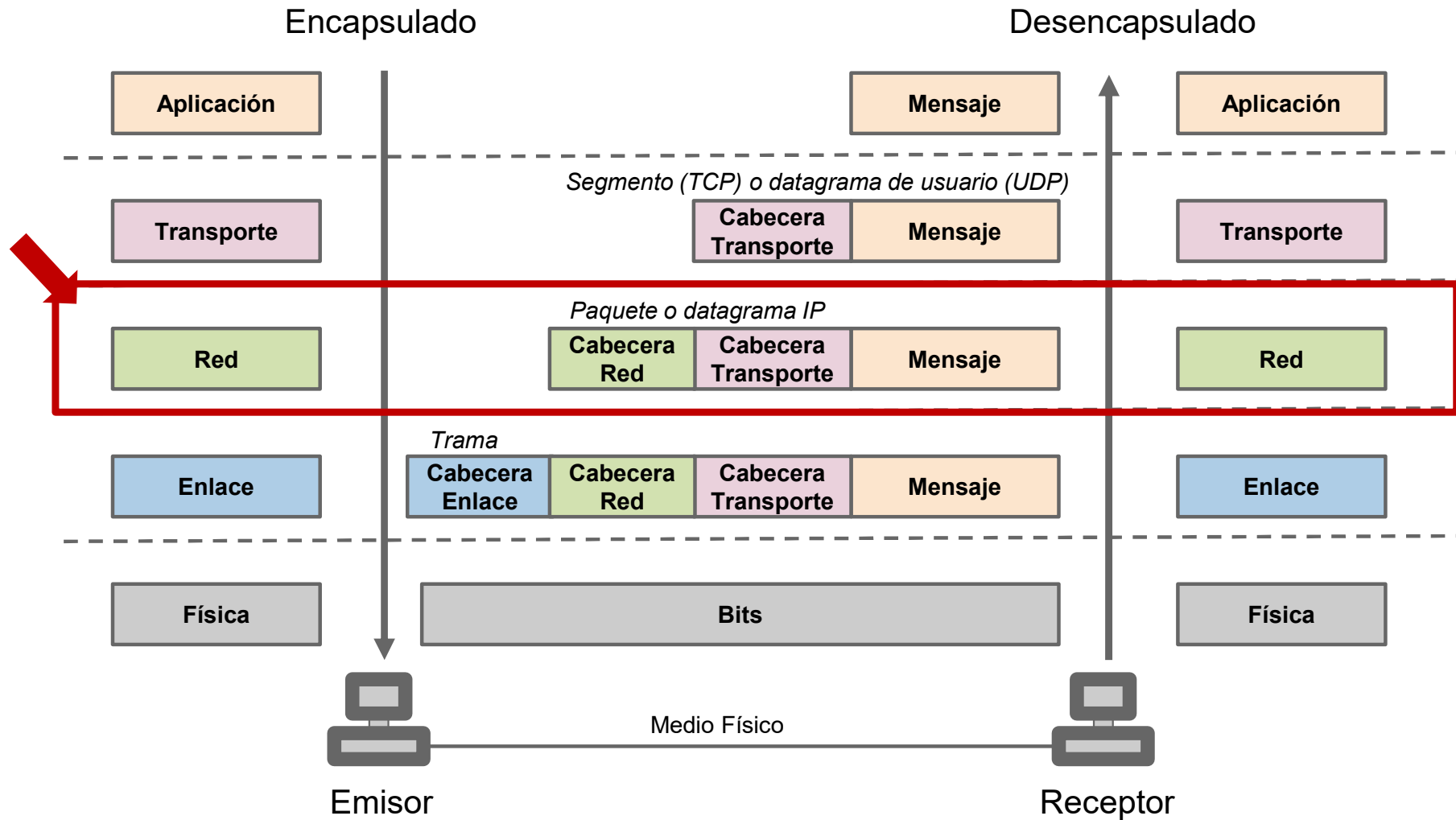
- **Direccionamiento**
  - Esquema global de direccionamiento
- **Fragmentación y reensamblaje** de paquetes
  - División del paquete en fragmentos de un tamaño aceptable por la red
- **Encaminamiento** de paquetes
  - Encaminado de paquetes atendiendo a información de tabla de rutas
  - La construcción de tablas de rutas puede ser
    - Manual (**encaminamiento estático**)
    - Mediante algún protocolo de **encaminamiento dinámico**: RIP, OSPF, BGP, etc.

# Introducción: La capa de red

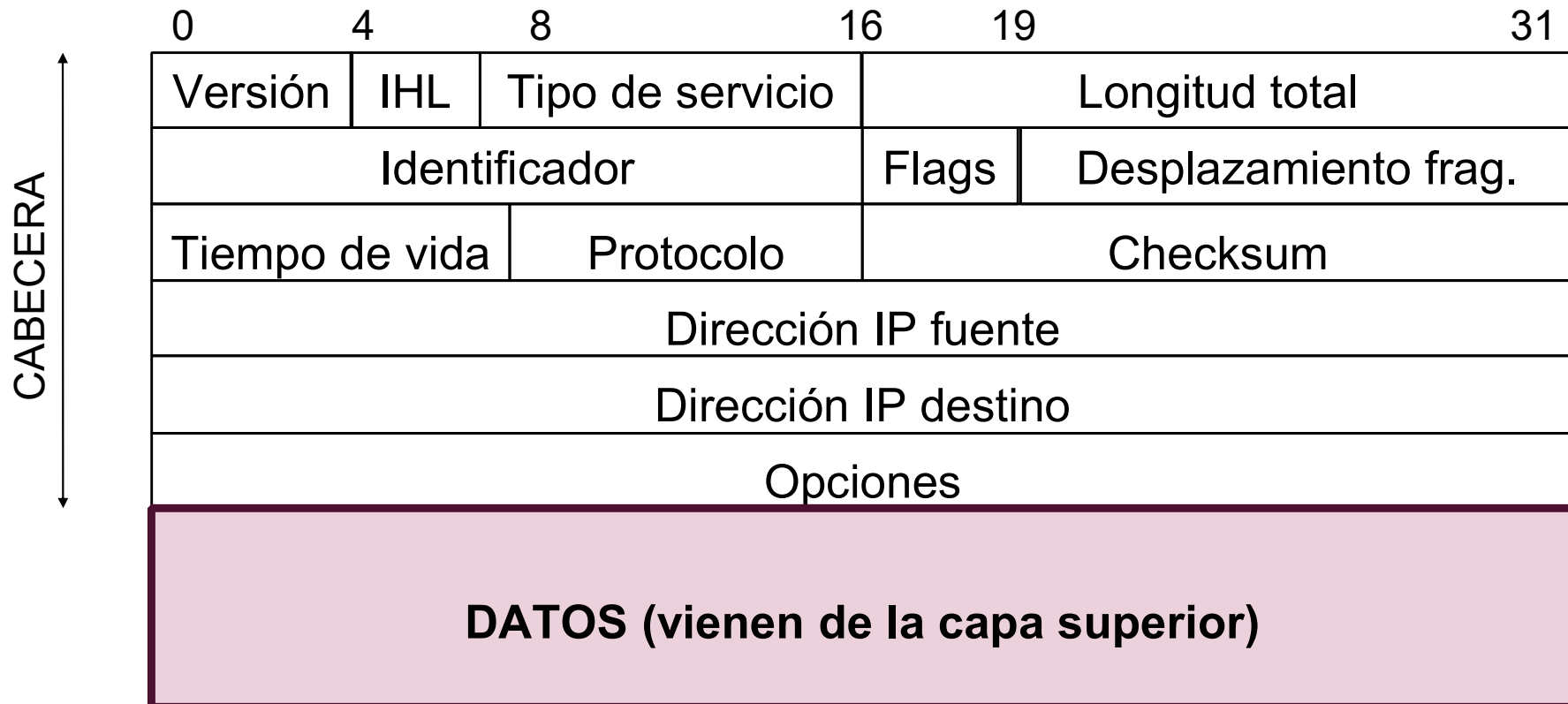
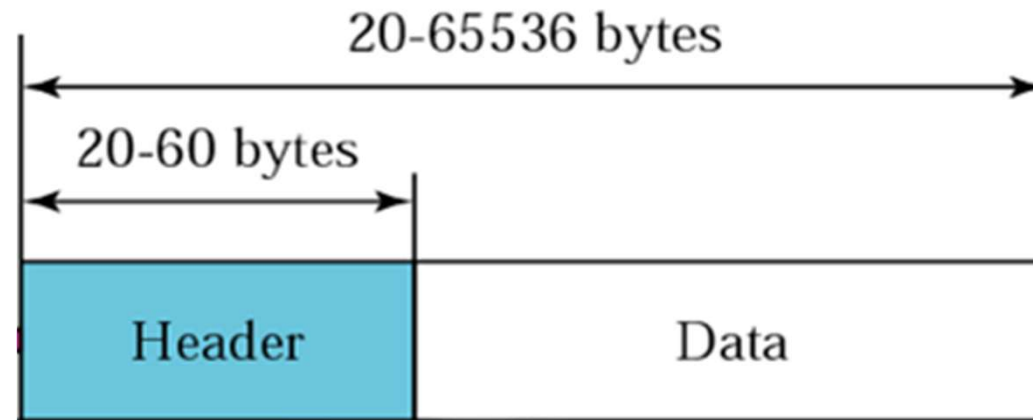


# **Protocolo IP. Formato del Datagrama**

# Formato del datagrama



# Formato del datagrama

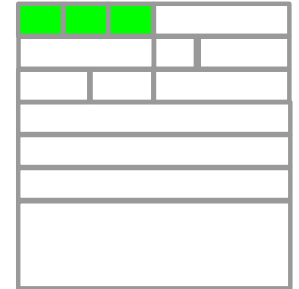


# Formato del datagrama

- **Versión:** valor=4 (IPv4)
- **IHL**
  - Longitud de la cabecera, en palabras de 32 bits
  - Campo IHL ocupa 4 bits
  - Tamaño máximo de la cabecera = 15 palabras (60 bytes)
- **Tipo de servicio**

0	1	2	3	4	5	6	7
Prioridad			Calidad de servicio (QoS)			Reserv.	

- **Prioridad**
  - Especifica la prioridad del datagrama (hasta 8 niveles)
  - Un paquete de alta prioridad debe ser reexpedido por un router antes que un paquete de baja prioridad (aunque éste llegase antes)
- **QoS:** Puede tomar los siguientes valores
  - 1000 → Minimizar retardo
  - 0100 → Maximizar rendimiento (velocidad de transmisión)
  - 0010 → Maximizar fiabilidad (seguridad en la entrega)
  - 0001 → Minimizar coste monetario
  - 0000 → Servicio normal



# Formato del datagrama

- **Longitud total (16 bits)**

- Longitud del datagrama (cabecera + datos) medida en bytes
- Campo Longitud Total ocupa 16 bits
  - Longitud máxima del datagrama:  $2^{16}$  bytes = 64 Kbytes



- **Tiempo de vida (TTL, *Time To Live*)**

- N° encaminadores que puede atravesar el paquete
- Cuando TTL=0 el paquete debe ser descartado

- **Campos para FRAGMENTACION:**

- **Identificador:** Número de 16 bits que identifica al datagrama original
- **Flags:**
  - **MF (More Fragments):** si está a 1 indica que no es el último fragmento
  - **DF (Don't Fragment):** si es 1 prohíbe la fragmentación
- **Desplazamiento del fragmento (Offset):**
  - Posición de los datos del fragmento en los datos del datagrama original.
  - En bloques de 8 bytes



# Formato del datagrama

- **Protocollo**

- Protocolo de la capa superior al que deben entregarse los datos
  - 1: Internet Control Message Protocol (ICMP)
  - 2: Internet Group Management Protocol (IGMP)
  - 6: Transmission Control Protocol (TCP)
  - 8: Exterior Gateway Protocol (EGP)
  - 17: User Datagram Protocol (UDP)
  - 41: IP Version 6 (IPv6)
  - 89: Open Shortest Path First (OSPF)

- **Checksum**

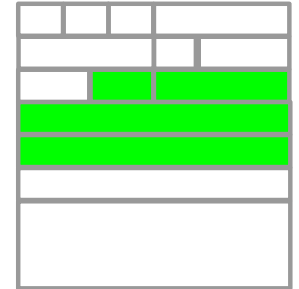
- Suma de control de la cabecera

- **Direcciones IP origen y destino**

- Identifican al host emisor y al receptor del datagrama

- **Opciones**

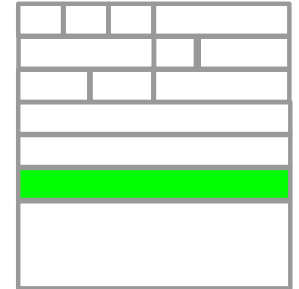
- Campo opcional, con opciones especiales
- Ejemplos: encaminamiento de origen, sello de ruta, sello de tiempo, etc.
- Tamaño máximo del campo opciones: 10 palabras



# Formato del datagrama: Campo Opciones

## Encaminamiento estricto de origen (*Strict Source Routing*)

- Proporciona un medio para que el emisor del paquete pueda especificar la ruta explícita que debe seguir el datagrama



## Registro de ruta (*Record Route*)

- Proporciona un medio para registrar la ruta exacta que ha seguido el datagrama en el camino hacia su destino (direcciones de los routers por los que ha pasado el datagrama)

## Sello de tiempo de Internet (*Internet Timestamp*)

- Proporciona un medio para registrar los instantes temporales en los que el paquete ha pasado por cada router y, adicionalmente, las direcciones de estos routers

# **IMPORTANTE**

## **En los ejemplos y ejercicios**

- **NO SE PONDRÁN TODOS LOS CAMPOS DE LA TRAMA**
- ✓ **SE PONDRÁN LOS CAMPOS DE LA TRAMA QUE INTERESEN PARA DICHO EJERCICIO**

# **Protocolo IP. Fragmentación**

# Fragmentación

## Fragmentación

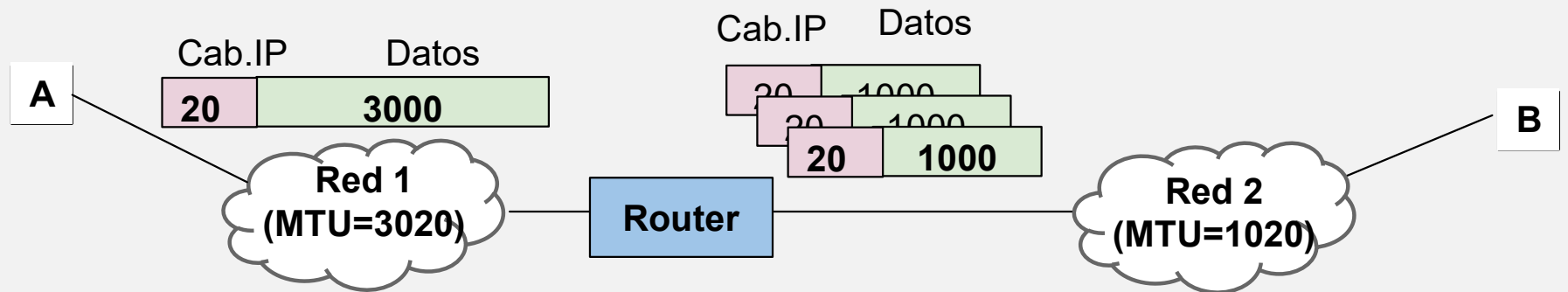
- Necesidad de adaptar el tamaño del datagrama IP a la MTU (maximum transfer unit) del camino
- Opciones:
  - Fragmentación en origen
  - Fragmentación en el camino (routers)
- El datagrama se reensambla en el destino
- La pérdida de un fragmento supone la retransmisión completa del datagrama

## Fragmentación en el camino

- El encaminador divide el datagrama en fragmentos para ajustar su tamaño a la MTU de la red
- Para cada fragmento se fija:
  - flags DF y MF (0 marca el último fragmento)
  - Desplazamiento de los datos (en unidades de 8 bytes): Indica el número de la primera unidad que se envía en ese fragmento

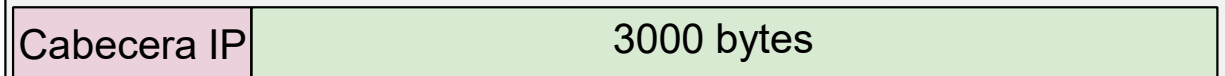
# Fragmentación en el camino

## Ejemplo 1



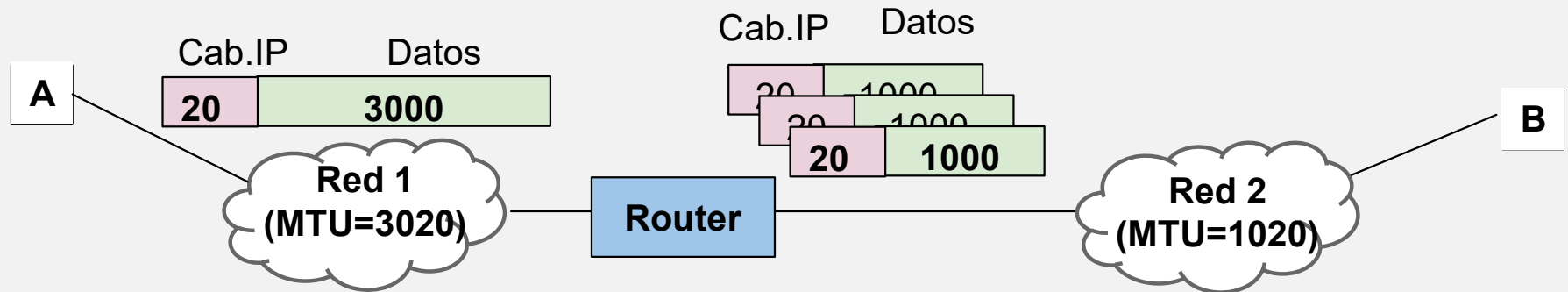
Datagrama original:

**Identificación:** 2667  
**Desplazamiento:** 0  
**MF=0; DF=0**



# Fragmentación en el camino

## Ejemplo 1



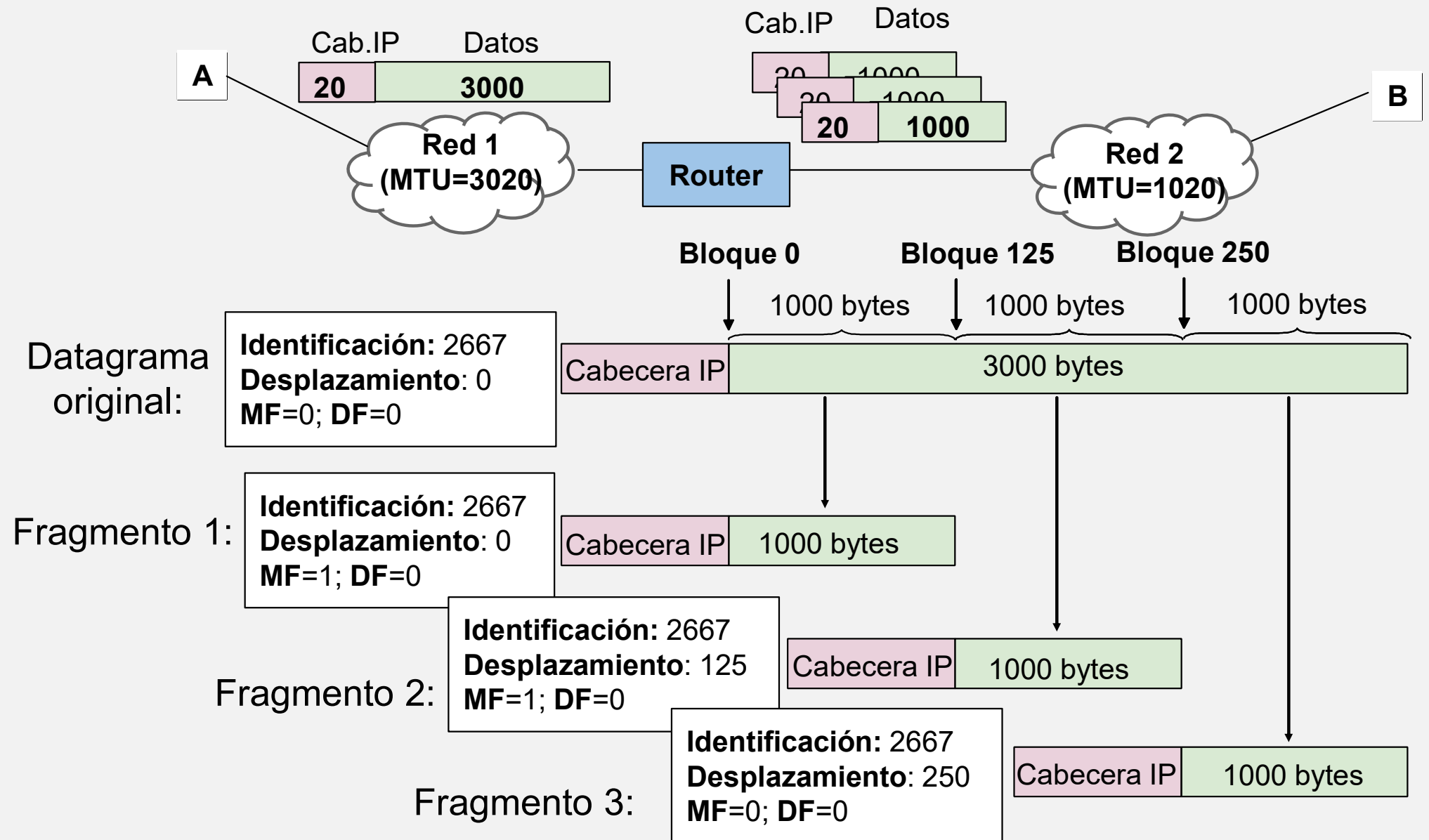
Como la MTU de la siguiente red es más pequeña (MTU=1020) **el router tienen que fragmentar el paquete:**

**Nº Unidades red2 = (MTU red2 – cabecera) / 8 bytes por unidad** =  $1000 / 8 = 125$  unidades => en cada fragmento se envían como máximo 1000 bytes de datos (125 unidades x 8 bytes por unidad)

**Nº fragmentos = Nº bytes de datos a enviar / Max. nº bytes de datos que envía en cada fragmento** =  $3000 / 1000 = 3$  fragmentos

# Fragmentación en el camino

## Ejemplo 1

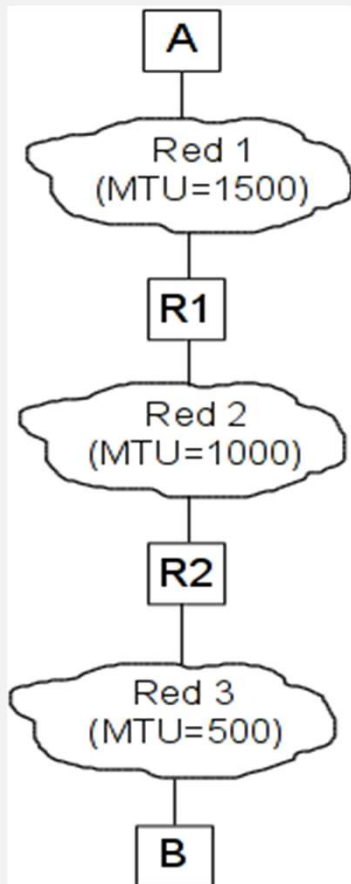




# Fragmentación en el camino

**Ejemplo 2:** Del Host A sale el siguiente **paquete (datagrama IP)** para el Host B. dicho paquete tiene que atravesar redes que tienen distinto MTU. El router 1 se encarga de fragmentar el paquete para que pueda atravesar la Red 2 y el router 2 se encarga de fragmentar el paquete para que pueda llegar al B que está en la Red 3

**¿Qué fragmentos envía cada una de los router?**



**Original**

20	1480
----	------

Fragmento	Identif	DF	MF	Desp.	Tam.
Original	12345	0	0	0	1500

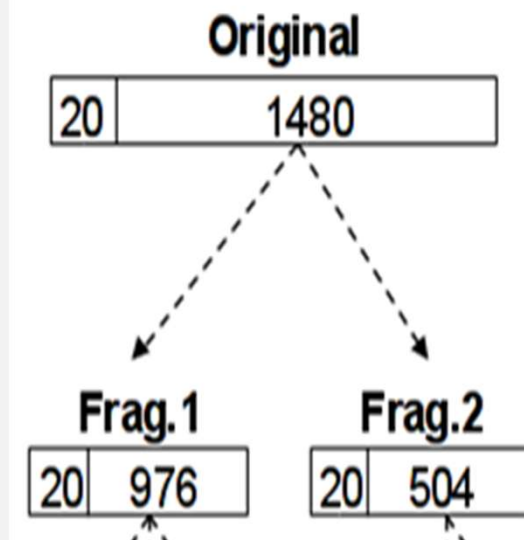
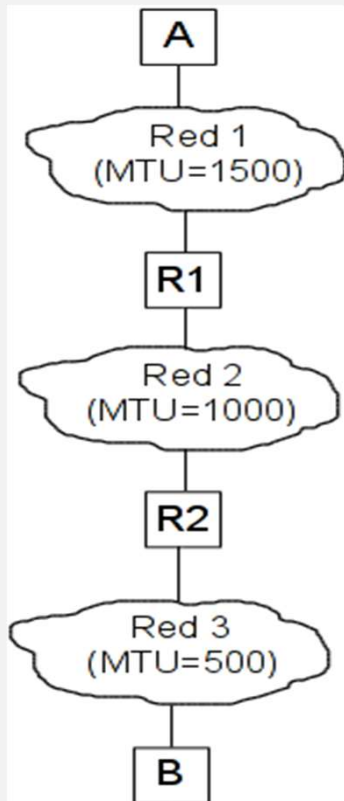
# Fragmentación en el camino

## Ejemplo 2 (Cont.)

**Nº Unidades red2 = (MTU red2 – cabecera) / 8 bytes por unidad** =  $980 / 8 = 122,5 \approx 122$  unidades =>

=> 122 unidades x 8 bytes por unidad = 976 bytes es el máx. nº de bytes de datos que se envían en cada fragmento

**Nº fragmentos = Nº bytes de datos a enviar / Max. nº bytes de datos que envía en cada fragmento** =  $1480 / 976 = 1,51 \Rightarrow 2$  fragmentos



**Datos + Cabecera**

Fragmento	Identif	DF	MF	Desp.	Tam.
Original	12345	0	0	0	1500

Fragmento	Identif	DF	MF	Despl.	Tam.
Frag.1	12345	0	1	0	996
Frag.2	12345	0	0	122	524

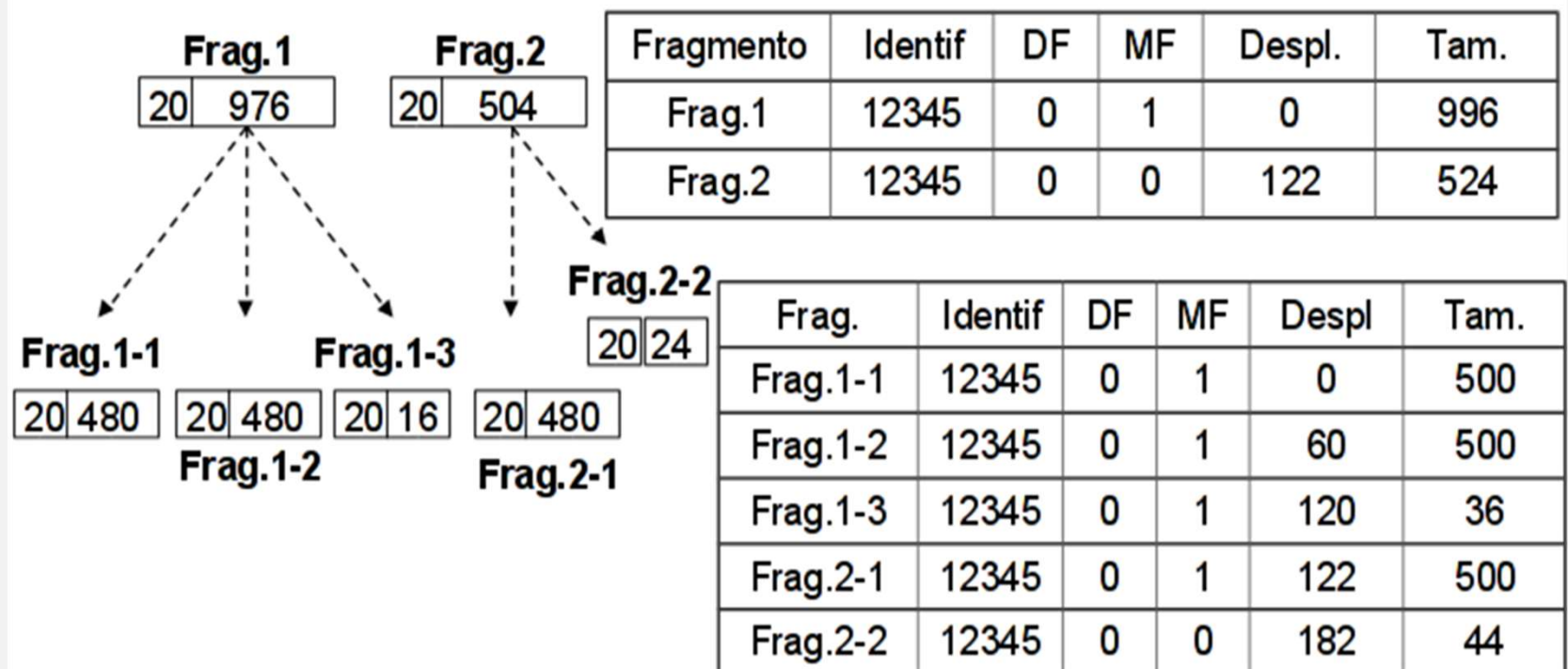
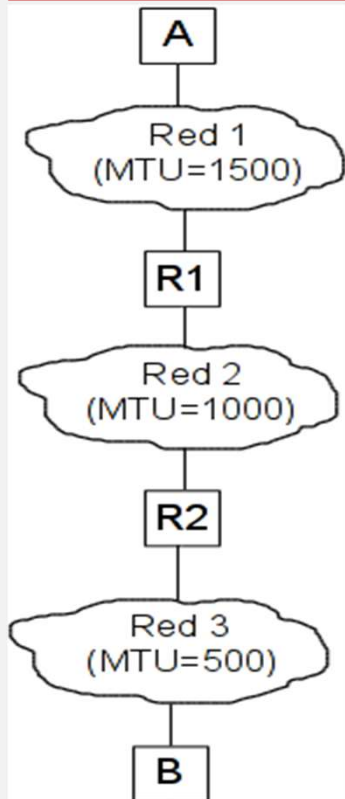
# Fragmentación en el camino

## Ejemplo 2 (Cont.)

**Nº Unidades red3 = (MTU red3 – cabecera) / 8 bytes por unidad** =  $480 / 8 = 60$  unidades  $\Rightarrow$  480 bytes es el máx. nº de bytes de datos que se envían en cada fragmento

Para el Frag.1  $\rightarrow$  **Nº fragmentos = Nº bytes de datos a enviar / Max. nº bytes de datos que envía en cada fragmento** =  $976 / 480 = 2,03 \Rightarrow 3$  fragmentos

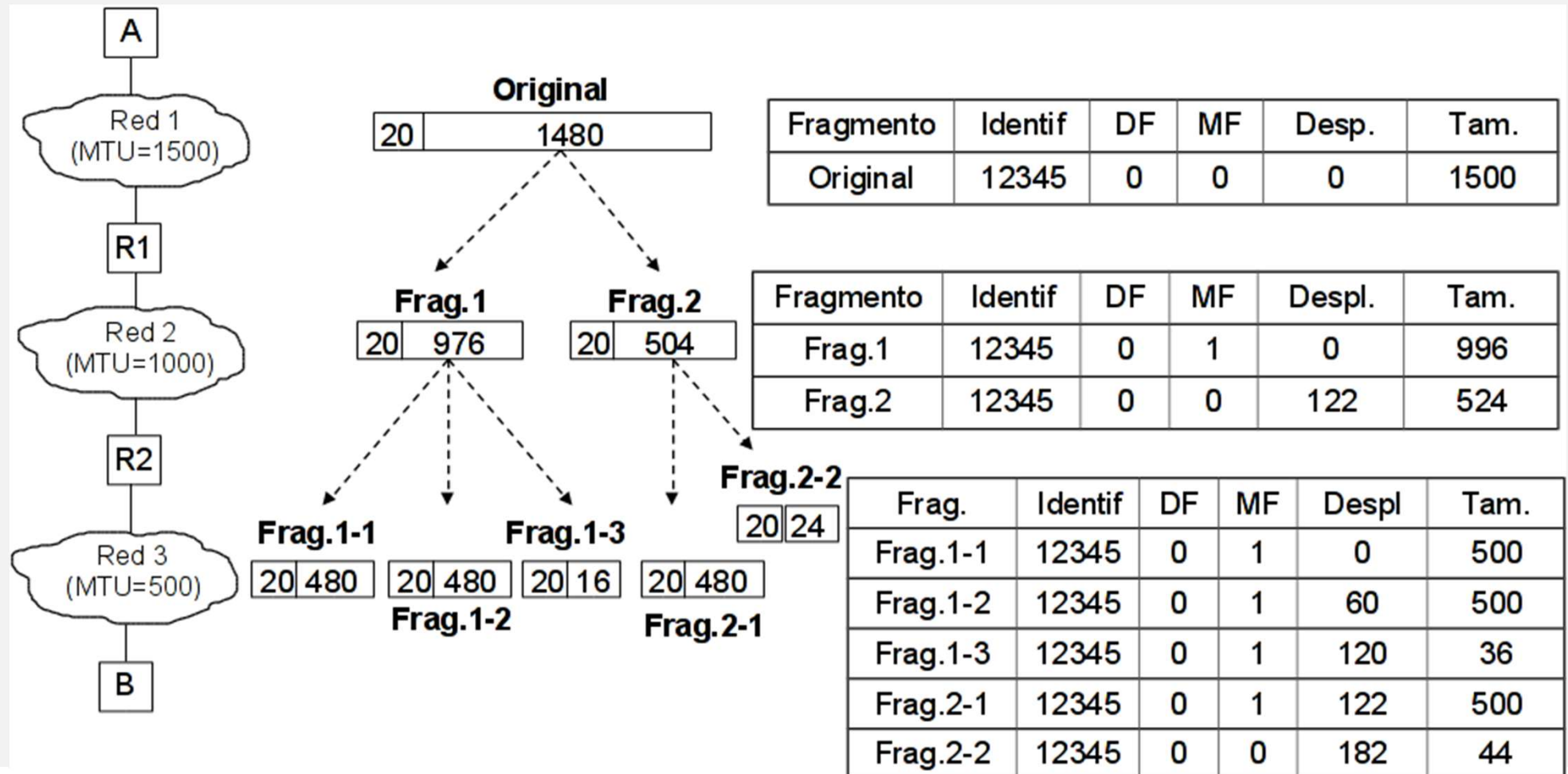
Para el Frag.2  $\rightarrow$  **Nº fragmentos = Nº bytes de datos a enviar / Max. nº bytes de datos que envía en cada fragmento** =  $504 / 480 = 1,05 \Rightarrow 2$  fragmentos



# Fragmentación en el camino

**Ejemplo 2 (completo):** Del Host A sale el siguiente **paquete (datagrama IP)** para el Host B. dicho paquete tiene que atravesar redes que tienen distinto MTU. El router 1 se encarga de fragmentar el paquete para que pueda atravesar la Red 2 y el router 2 se encarga de fragmentar el paquete para que pueda llegar al B que está en la Red 3

**¿Qué fragmentos envía cada una de los router?**



# Fragmentación en origen

Primero el host que envia el paquete necesita descubrir el MTU mínimo del camino entre el origen y destino (*path MTU discovery*), para ello tiene que realizar los siguientes pasos:

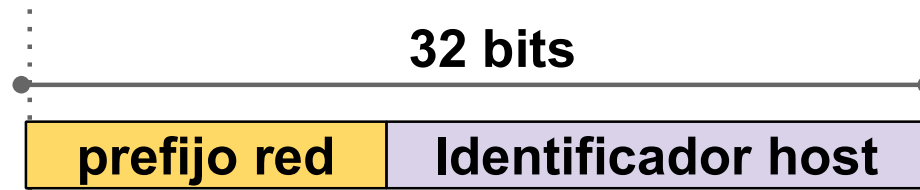
- El host que quiere enviar el paquete (también llamado datagrama)
- El host origen envía el paquete con el bit DF=1 y el tamaño ajustado al MTU de la red local
- Si el paquete encuentra una red con MTU menor:
  - El router no realiza la fragmentación porque DF es 1
  - El router envías un mensaje de error al host origen indicandole la MTU de la red
- El host origen divide el paquete en trozo mas pequeños ajustados a la MTU de la red que no podía atravesar y mantiene el bit DF=1 y los vuelve a enviar
- Este proceso se repite hasta que el paquete llega a su destino

# **Protocolo IP. Direccionamiento**

# Notación y Tipos de Direccionamiento

## Estructura y Notación

- Las direcciones IP constan de 4 bytes (32 bits)
- Para expresarlas se utiliza la “notación de punto” (Ej. 10.0.0.1)



## Tipos de Direccionamiento

- Unicast: La dirección identifica unívocamente un único host (*uno-a-uno*)
- Broadcast: La dirección identifica a todos los hosts posibles (*uno-a-todos*)
- Multicast: La dirección representa a un grupo de hosts (*uno-a-varios*)

# El protocolo IP: Direcciones

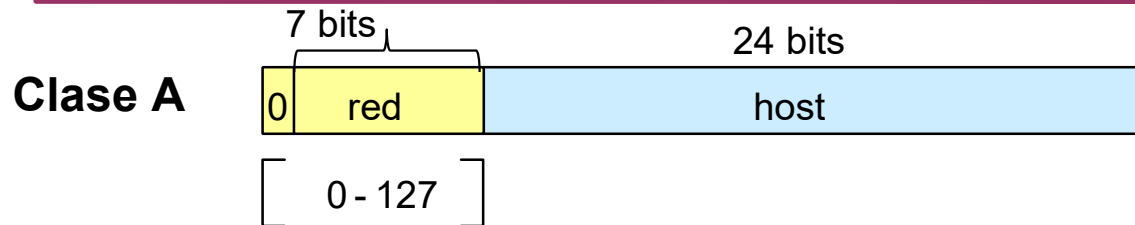
## Administración y registro de direcciones

- **Entidades regionales de registro de Internet:** La parte que identifica a la red es fija para cada red y es necesario solicitarla a una de las entidades regionales de registro de Internet (**RIR, Regional Internet Registries**)
- Cada RIR es responsable de administrar y registrar las direcciones IP de su región geográfica:
  - **ARIN** (*American Registry for Internet Numbers*): Norteamérica
  - **RIPE** (*Reseaux IP Europeens*): Europa y Oriente Medio
  - **APNIC** (*Asia Pacific Network Information Center*): Asia-Pacífico
  - **LACNIC** (*Latin American and Caribbean Network Information Center*): Sudamérica y países caribeños
  - **AfriNIC** (*African Network Information Center*): África



# Direccionamiento basado en clases

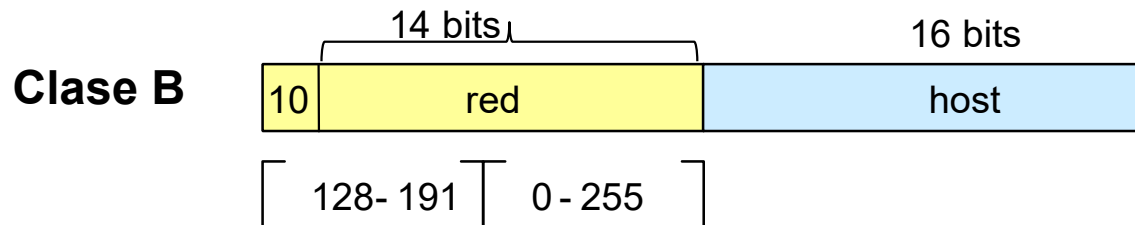
## Direcciones IPv4 basadas en clase (classful)



$2^7 = 128$  redes

$2^{24} = 16.777.216$  direcciones

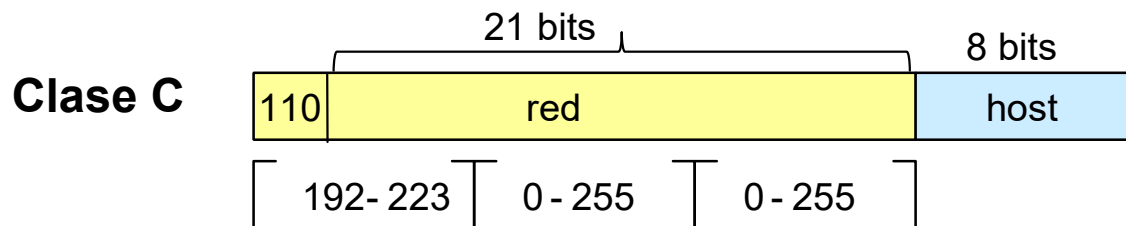
Ejemplo: **26**.56.120.9



$2^{14} = 16.384$  redes

$2^{16} = 65.536$  direcciones

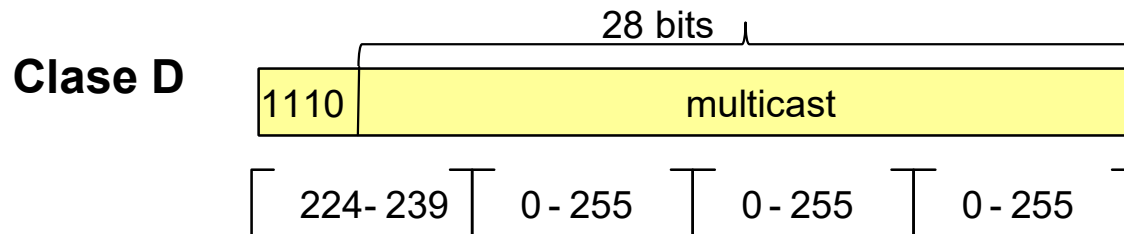
Ejemplo: **147.96**.50.110



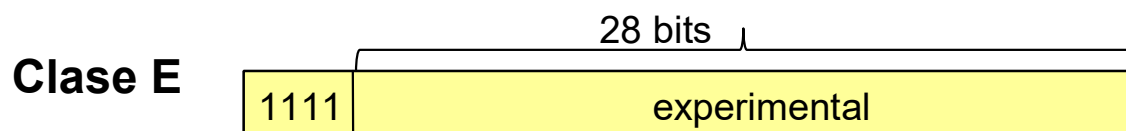
$2^{21} = 2.097.152$  redes

$2^8 = 256$  direcciones

Ejemplo: **217.6.95**.44



Ejemplo: **224.0.0.1**



# Direccionamiento basado en clases

## Problema de las direcciones con clases

Ejemplo:

- **Una empresa/institución necesita aprox. 15.000 direcciones IP**

Con el direccionamiento con clase:

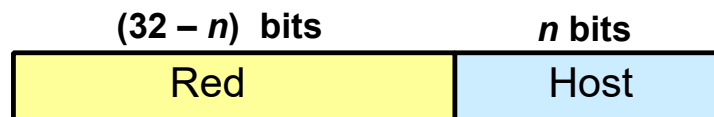
- Clase C tiene 256 direcciones luego no es suficiente
- Tiene que contratar una red clase B completa (65.536 direcciones)
  - El coste es muy elevado (más direcciones de las necesarias)
  - Se desaprovechan la mayoría de las direcciones (más de 50.000)
- Usando direcciones sin clases
  - Se puede contratar un conjunto más ajustado (en potencias de 2)
    - En este caso, se pueden contratar  $2^{14} = 16.384$  direcciones

# Direccionamiento sin clases

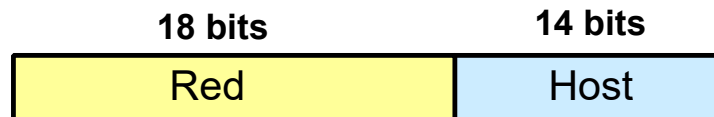
## Direcciones IPv4 sin clase (classless)

### Formato de las direcciones sin clase

- Los campos de red y host no están limitados a un número entero de bytes



**Ejemplo:** En el ejemplo anterior con  $n=14$



# Direcciones especiales

## Direcciones reservadas para redes privadas

- Existe un conjunto de direcciones reservadas para uso privado
- No son válidas para su uso en Internet
  - Se pueden asignar a redes aisladas de Internet
  - Se pueden asignar a redes conectadas a través de un router que hace traducción de direcciones de red (NAT)
- Los rangos de direcciones IP privadas son los siguientes:
  - ✓ 10.0.0.0 – 10.255.255.255      ~ 1 red privada de clase A
  - ✓ 172.16.0.0 – 172.31.255.255      ~ 16 redes privadas de clase B
  - ✓ 192.168.0.0 – 192.168.255.255      ~ 256 redes privadas de clase C

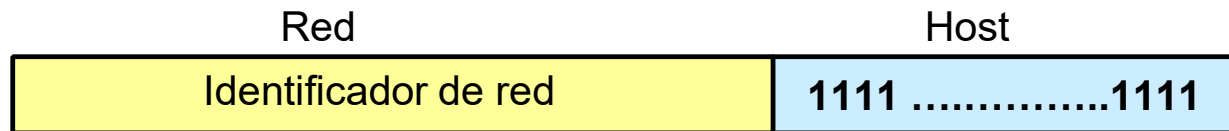
## Direcciones de loopback (127.x.y.z)

- Direcciones de bucle interno (loopback) referidas al mismo host.
- Los datagramas IP con dirección de loopback nunca se envían por la red.
- Formato 127.x.y.z (típicamente 127.0.0.1)

# Direcciones especiales

## Direcciones broadcast (terminadas en 11...111)

- Se utilizan para enviar un paquete a todas las máquinas de la red local
- Formato de las direcciones broadcast
  - Todos los bits de identificador de host se ponen a valor 1
  - Último valor del rango de direcciones de la red



# Direcciones especiales

## Ejemplo: Direcciones de broadcast

- Red de clase A: 

Red (8)	Host (24)
00011011	.11111111.11111111.11111111

 = 27.255.255.255
- Red de clase B: 

Red (16)	Host (16)
10001110.01011000	.11111111.11111111

 = 142.88.255.255
- Red de clase C: 

Red (24)	Host (8)
11000111.01000011.11101111	.11111111

 = 199.67.239.255
- Red sin clase (n=14): 

Red (18)	Host (14)
01011010.00100000.10	11111111.11111111

 = 90.32.191.255
- Red sin clase (n=5): 

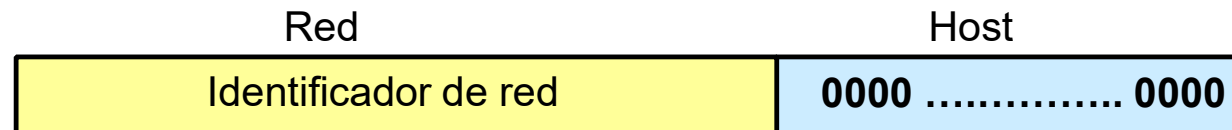
Red (27)	Host (5)
10001111.00011010.00000111.011	11111

 = 143.26.7.127
- Dir. broadcast universal: 255.255.255.255

# Direcciones especiales

## Direcciones de red (terminadas en 00...000)

- Se utilizan para representar a una red completa en las tablas de encaminamiento
- **Nunca se utilizan como dirección destino ni se asignan a un host concreto**
- Formato de las direcciones de red
  - Todos los bits de identificador de host se ponen a valor 0
  - Se corresponde con el primer valor del rango de direcciones de la red



# Direcciones especiales

## Ejemplo: Direcciones de red

- Red de clase A: 

Red (8)	00011011	Host (24)	00000000.00000000.00000000
---------	----------	-----------	----------------------------

 = 27.0.0.0
- Red de clase B: 

Red (16)	10001110.01011000	Host (16)	00000000.00000000
----------	-------------------	-----------	-------------------

 = 142.88.0.0
- Red de clase C: 

Red (24)	11000111.01000011.11101111	Host (8)	00000000
----------	----------------------------	----------	----------

 = 199.67.239.0
- Red sin clase (n=14): 

Red (18)	01011010.00100000.10	Host (14)	00000000.00000000
----------	----------------------	-----------	-------------------

 = 90.32.128.0
- Red sin clase (n=5): 

Red (27)	10001111.00011010.00000111.011	Host (5)	00000
----------	--------------------------------	----------	-------

 = 143.26.7.96



# Máscaras de red

La máscara de red sirve para indicar cuantos bits de la dirección IP identifican a la red y cuantos identifican al host dentro de la red

- Formato de las direcciones de red
  - Bits de red tiene el valor 1
  - Bits del host tiene valor 0

## Ejemplo

- Dirección de clase C

	Red	Host	
Dir. IP de un host	11011101. 01100010. 00010110.	00000010	221.98.22.2
Máscara	11111111. 11111111. 11111111.	00000000	255.255.255.0

## Notación alternativa: 221.98.22.2/24

- El valor /24 indica la longitud de la parte de red (nº de unos de la máscara)
- Esta notación se denomina **CIDR** (*Classless Interdomain Routing*)

# Ejercicio 1

**Dadas las siguientes direcciones de IP calcular:** Máscara de red, Notación CIDR, Dirección de red, Dir. de difusión, Número máximo de hosts, Rango de direcciones útiles

Clase A: Dirección IP: 27.7.130.3	→	<div>red (8)host (24)</div> <div>00011011.00000111.10000010.00000011</div>
Clase B: Dirección IP: 142.88.12.4	→	<div>red (16)host (16)</div> <div>10001110.01011000.00001100.00000100</div>
Clase C: Dirección IP: 199.67.239.6	→	<div>red (24)host (8)</div> <div>11000111.01000011.11101111.00000110</div>
Sin clase: Dirección IP: 90.32.131.5	→	<div>red (18)host (14)</div> <div>01011010.00100000.10000011.00000101</div>
Sin clase: Dirección IP: 143.26.7.99	→	<div>red (27)host (5)</div> <div>10001111.00011010.00000111.01100011</div>

# **Protocolo IP. Encaminamiento**

# El protocolo IP: Encaminamiento

El encaminamiento consiste en **encontrar un camino, desde el origen al destino**, a través de **nodos de conmutación o encaminadores (routers)** intermedios

- **Caminos alternativos**
  - Es necesario decidir cuál es el mejor camino posible (*camino más corto*)
  - El *camino más corto* minimiza una métrica de encaminamiento

## Métricas de encaminamiento

- **Número de saltos:** tiene en cuenta el número de encaminadores y/o redes intermedias que tiene que atravesar el paquete para alcanzar el destino
- **Distancia geográfica:** tiene en cuenta la distancia (en Km) que tiene que recorrer el paquete para alcanzar el destino
- **Retardo promedio:** tiene en cuenta el retardo de las líneas. Dado que éste es proporcional a la distancia, esta métrica es similar a la anterior
- **Ancho de banda:** tiene en cuenta la velocidad de transmisión de las líneas por las que tiene que circular el paquete
- **Nivel de tráfico:** tiene en cuenta el nivel de uso de las líneas, para intentar utilizar aquellas líneas con menor nivel de saturación
- Combinación lineal de varias métricas

# El protocolo IP: Encaminamiento

## Encaminamiento estático

- Las decisiones de encaminamiento **consideran la topología de la red**
- Las tablas de encaminamiento **se construyen manualmente**
- **NO se adaptan a los cambios de la red**

## Encaminamiento dinámico

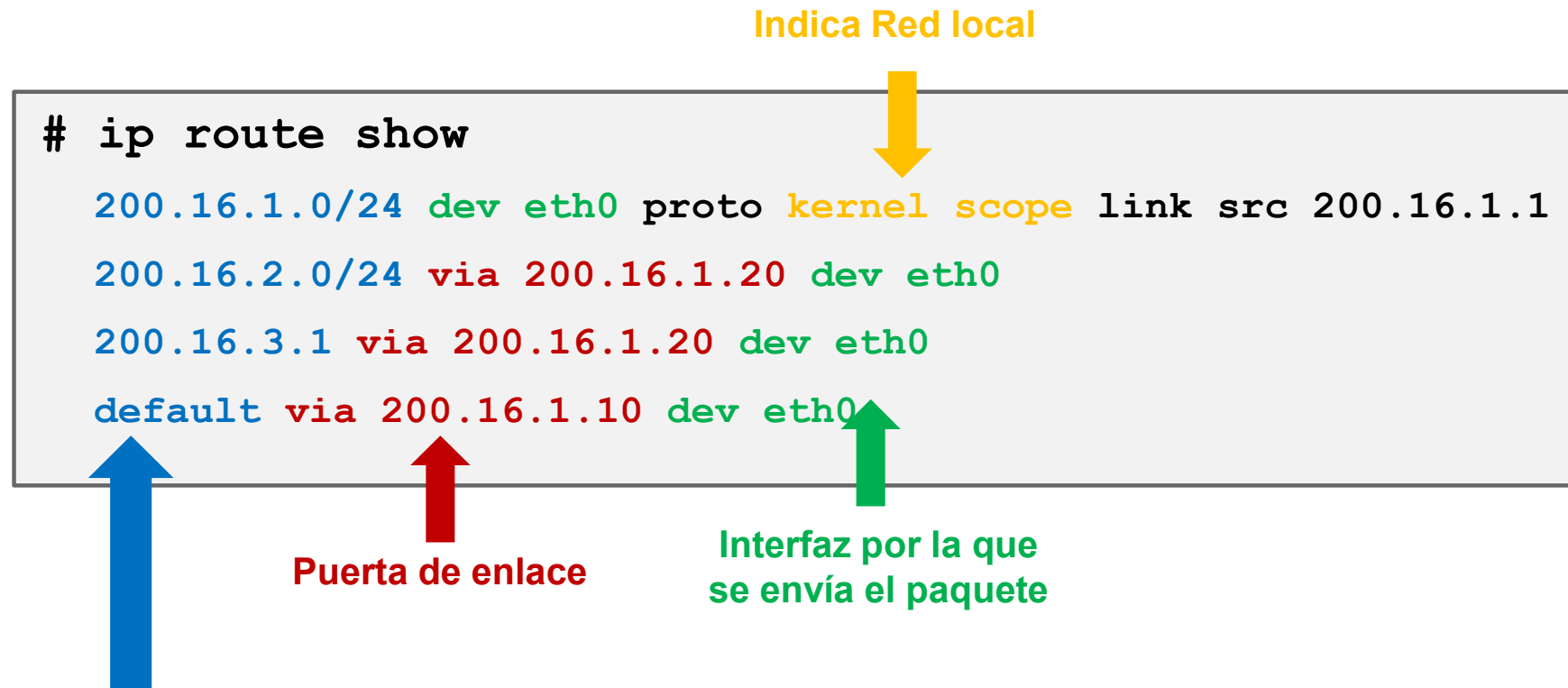
- Las tablas de encaminamiento **se construyen de forma automática** mediante el **intercambio periódico de información** entre los encaminadores
- **Se adaptan automáticamente a los cambios en la topología de la red**
- Las técnicas más comunes son:
  - Encaminamiento por vector de distancias (ej. RIP)
  - Encaminamiento por estado de los enlaces (ej. OSPF)
  - Encaminamiento por vector de rutas (ej. BGP)

# El protocolo IP: Tabla de encaminamiento

Cuando un encaminador (router) recibe un paquete lo retransmite (*forward*) por el enlace adecuado para alcanzar el destino

- La elección del enlace adecuado se realiza atendiendo a la tabla de encaminamiento

Estructura de tabla de encaminamiento en Linux



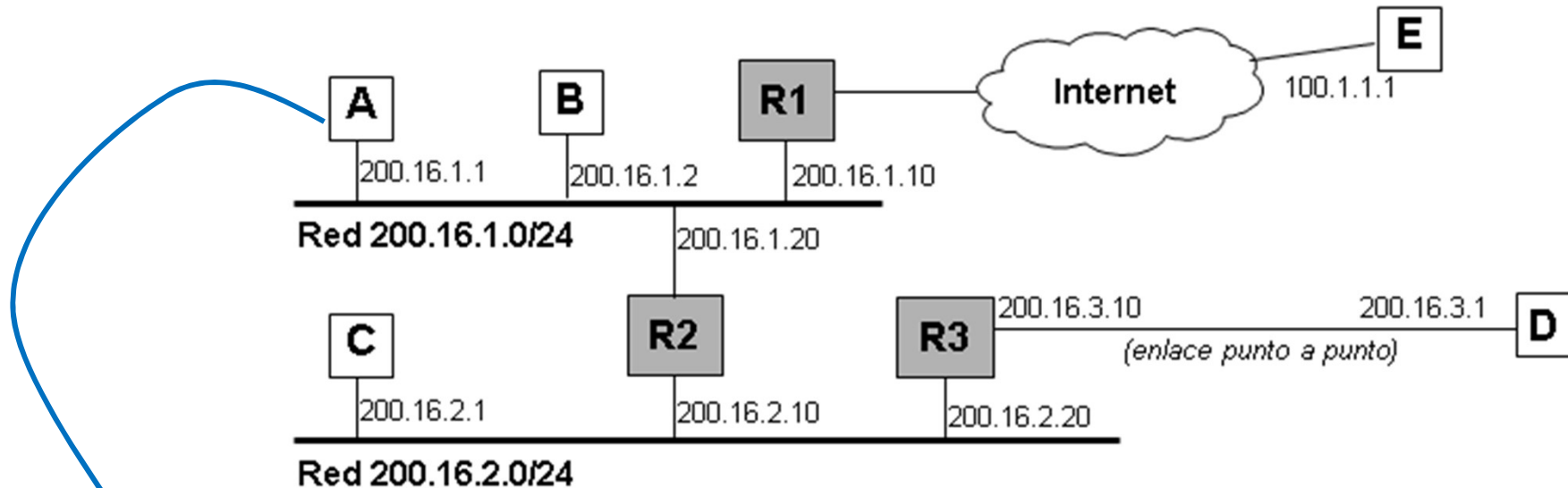
En las entradas de la tabla la **información del destino** puede ser:

- **Host específico** (no es viable para el encaminamiento en internet)
- **Una Red.** Cuando se usan redes sin clase hay que añadir la máscara
- **Default.** Camino para los paquetes que no encajen en ninguna entrada

# El protocolo IP: Tabla de encaminamiento

## Estructura de la tabla de encaminamiento en Linux

- Supongamos la siguiente red



- La tabla de encaminamiento de la **estación A**, podría ser la siguiente:

```
# ip route show  
1 ---> 200.16.1.0/24 dev eth0 proto kernel scope link src 200.16.1.1  
2 ---> 200.16.2.0/24 via 200.16.1.20 dev eth0  
3 ---> 200.16.3.1 via 200.16.1.20 dev eth0  
4 ---> default via 200.16.1.10 dev eth0
```

# El protocolo IP: Tabla de encaminamiento

- Qué significa cada entrada de la tabla de encaminamiento

**Entrada 1** (el destino es la propia red local)

- **200.16.1.0/24**: indica la red/máscara destino (en este caso el destino es la propia red local de la máquina A)
- **dev eth0**: interfaz de red por la que se envían los datagramas para alcanzar ese destino.
- **proto kernel**: indica que la entrada fue añadida automáticamente por el kernel durante la configuración de la interfaz de red (no es una entrada manual o añadida por algún protocolo de routing)
- **scope link**: indica que la entrada hace referencia a una red local.
- **src 192.168.0.1**: indica la dirección IP fuente de los datagramas expedidos por la interfaz de red asociada a esta entrada.

## ¿Cómo añadir esta entrada a la tabla de rutas de forma manual?

- Esta entrada NO es necesario añadirla a la tabla de rutas
- **Se crea de forma automática cuando se configura la IP de la máquina A** mediante el comando:

```
# ip addr add 200.16.1.1/24 broadcast + dev eth0
```



# El protocolo IP: Tabla de encaminamiento

- Qué significa cada entrada de la tabla de encaminamiento

## Entrada 2 (el destino es otra red)

- **200.16.1.0/24**: indica la red/máscara destino
- **via 200.16.1.20**: indica que el destino es alcanzable de forma indirecta a través de este encaminador (puerta de enlace)
- **dev eth0**: interfaz de red por la que se envían los datagramas para alcanzar este destino.

## ¿Cómo añadir esta entrada a la tabla de rutas de forma manual?

```
# ip route add 200.16.1.0/24 via 200.16.1.20 dev eth0
```

# El protocolo IP: Tabla de encaminamiento

- Qué significa cada entrada de la tabla de encaminamiento

## **Entrada 3** (el destino es un host)

- **200.16.3.1**: indica el host destino (al no especificar la máscara, el destino es un host)
- **via 200.16.1.20**: indica que el destino es alcanzable de forma indirecta a través de este encaminador (puerta de enlace)
- **dev eth0**: interfaz de red por la que se envían los datagramas para alcanzar ese destino.

## ¿Cómo añadir esta entrada a la tabla de rutas de forma manual?

```
# ip route add 200.16.3.1/32 via 200.16.1.20 dev eth0
```

# El protocolo IP: Tabla de encaminamiento

- Qué significa cada entrada de la tabla de encaminamiento

**Entrada 4** (para definir el encaminador predeterminado)

- **default:** indica que el destino puede ser cualquiera (cuando la red o host destino no coincide con ninguna de las entradas anteriores de la tabla de rutas)
- **via 200.16.1.10:** indica que el destino es alcanzable de forma indirecta a través de este encaminador (puerta de enlace)
- **dev eth0:** interfaz de red por la que se envían los datagramas para alcanzar ese destino.

¿Cómo añadir esta entrada a la tabla de rutas de forma manual?

```
# ip route add default via 200.16.1.20 dev eth0
```

# El protocolo IP: Tabla de encaminamiento

## Cómo se consulta la tabla de encaminamiento:

- ✓ Se realiza la función AND entre la dirección IP del host al que se quiere enviar el paquete y la máscara de red de cada una de las entradas de la tabla

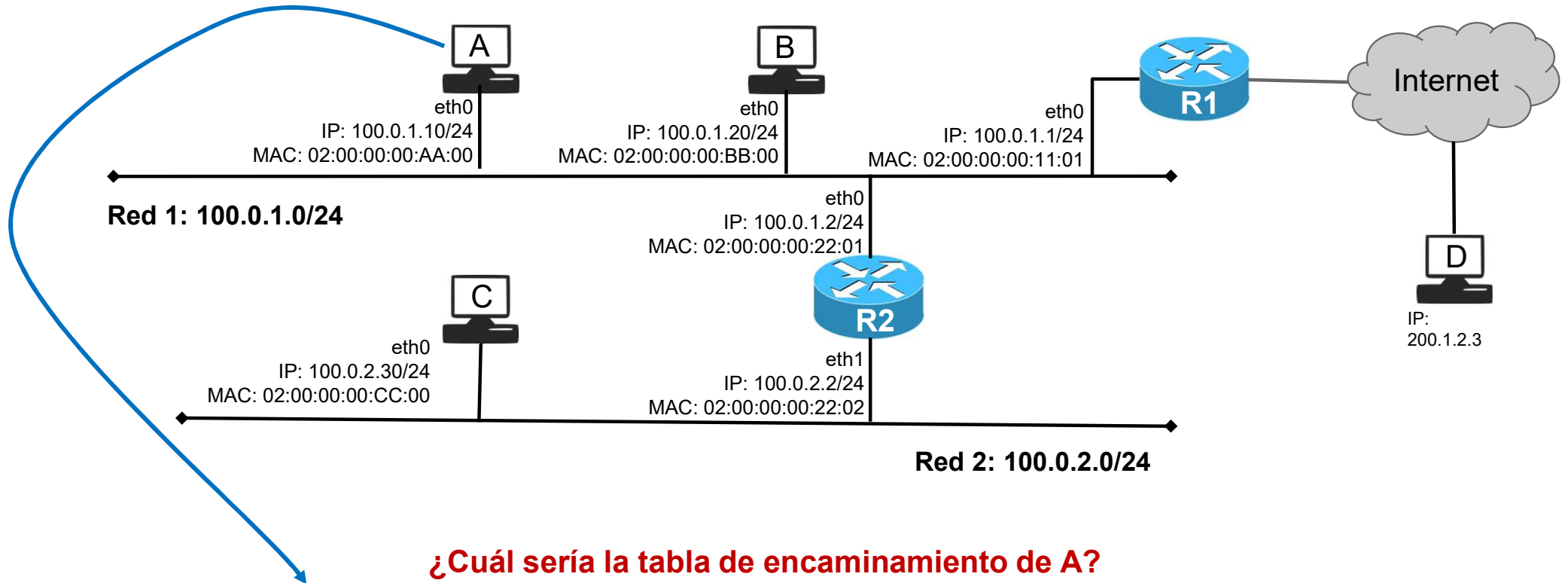


**Esto convierte la dir. del host destino en una dir. de red**

- Si la dir. de red resultante coincide con la dirección destino de la entrada que hace referencia a una red local (**scope link**), significa que el destino está en la misma red que la fuente: el mensaje se manda directamente
- Si la dir. de red resultante coincide con alguna entrada que no es red local, significa que el destino no está en la misma red que la fuente: el mensaje se envía por la puerta de enlace indicada en esa entrada
- Si la dir. de red resultante NO coincide con ninguna entrada de la tabla de ruta, se envía por la puerta de enlace por defecto (**default**)
- Si no hay encaminamiento por defecto (**default**) no se puede enviar el paquete y se envía un mensaje ICMP notificando el error (destination unreachable)

# El protocolo IP: Tabla de encaminamiento

## Ejemplo de funcionamiento del encaminamiento:



```
# ip route show

100.0.1.0/24 dev eth0 proto kernel scope link src 100.0.1.10

100.0.2.0/24 via 100.0.1.2 dev eth0

default via 100.0.1.1 dev eth0
```

# El protocolo IP: Tabla de encaminamiento

Ejemplo (cont.): **Host-A envía un paquete a Host-B** (IP: 100.0.1.20)

**Host-A usa su tabla de encaminamiento para saber si el destino está en su red o fuera de su red** ⇒ Realiza una AND entre la IP destino y la máscara de la primera entrada de la tabla

```
100.0.1.0/24 dev eth0 proto kernel scope link src 100.0.0.10
```

IP destino:	100.0.1.20	=	01100100	.	00000000	.	00000001	.	00010100
Máscara:	255.255.255.0	=	<u>11111111</u>	.	<u>11111111</u>	.	<u>11111111</u>	.	<u>00000000</u>
Red destino:	100.0.1.0	=	01100100	.	00000000	.	00000001	.	00000000

- ✓ El resultado coincide con la red destino de la primera entrada de la tabla ⇒
- ✓ **El destino está en la propia red**

# El protocolo IP: Tabla de encaminamiento

Ejemplo(cont.): **Host-A envía un paquete a Host-C** (IP: 100.0.2.30)

Host-A usa su tabla de encaminamiento para saber si el destino está en su red o fuera de su red ⇒  
Realiza una AND entre la IP destino y la máscara de la primera entrada de la tabla:

```
100.0.1.0/24 dev eth0 proto kernel scope link src 100.0.0.10
```

IP destino:	100.0.2.30	=	01100100	.	00000000	.	00000010	.	00011110
Máscara:	255.255.255.0	=	<u>11111111</u>	.	<u>11111111</u>	.	<u>11111111</u>	.	<u>00000000</u>
Red destino:	100.0.2.0	=	01100100	.	00000000	.	00000010	.	00000000

- ✓ El resultado NO coincide con la red destino de la primera entrada de la tabla ⇒ Luego el **Host-C no está en la misma red que el Host-A**
- ✓ Se repite la AND con la máscara de la segunda entrada que en este caso es la misma. El resultado coincide con la red destino de la segunda entrada de la tabla

```
100.0.2.0/24 via 100.0.1.2 dev eth0
```

- ✓ ⇒ Luego para llegar al Host-C **hay que enviar el paquete por la IP = 100.0.1.2** que corresponde a la interfaz eth0 del Router 2

**El Router 2 es la puerta de enlace asociada a la red destino 100.0.2.0**

# El protocolo IP: Tabla de encaminamiento

Ejemplo(cont.): **Host-A envía un paquete a Host-D** (IP: 200.1.2.3)

Host-A usa su tabla de encaminamiento para saber si el destino está en su red o fuera de su red ⇒  
Realiza una AND entre la IP destino y la máscara de la primera entrada de la tabla:

```
100.0.1.0/24 dev eth0 proto kernel scope link src 100.0.0.1
```

IP destino:	200.1.2.3	=	11001000	.	00000001	.	00000010	.	00000011
Máscara:	255.255.255.0	=	<u>11111111</u>	.	<u>11111111</u>	.	<u>11111111</u>	.	<u>00000000</u>
Red destino:	200.1.2.0	=	11001000	.	00000001	.	00000010	.	00000000

- ✓ El resultado NO coincide con la red destino de la primera entrada de la tabla ⇒ Luego el **Host-D no está en la misma red que el Host-A**
- ✓ Se repite la AND con la máscara de la segunda entrada que en este caso es la misma. El resultado NO coincide con la red destino de la segunda entrada de la tabla
- ✓ AL no haber mas entradas de la tabla con redes destino se usa la entrada por defecto:

```
default via 100.0.1.1 dev eth0
```

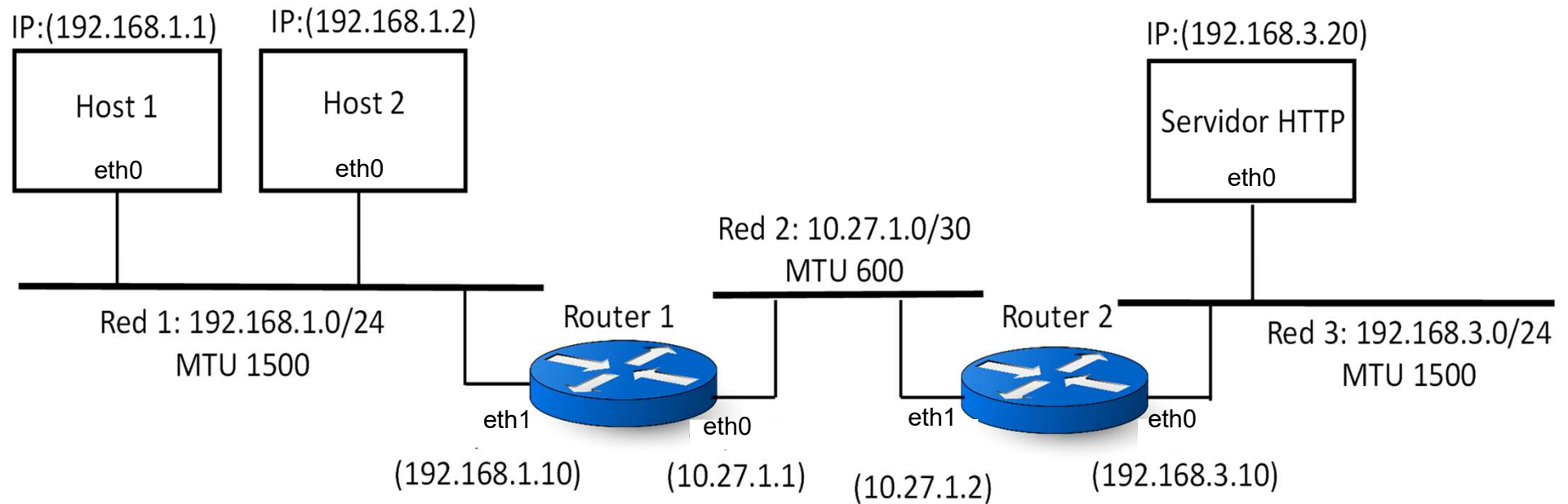
- ✓ Luego para llegar al Host-D hay que enviar el paquete por la **IP = 100.0.1.1** que corresponde a la interfaz eth0 del Router 1

**El Router 1 es la puerta de enlace asociada por defecto**



# Ejercicio 2

Escribir la tabla de encaminamiento que deben tener cada uno de los Host y cada uno de los Routers



# Ejercicio 3

Considerar la ejecución de la siguiente orden en un computador con sistema operativo Linux:

```
# ip route show  
192.168.1.64/26 dev eth0 proto kernel scope link src 192.168.1.67  
192.168.1.128/26 dev eth1 proto kernel scope link src 192.168.1.130  
192.168.1.192/26 via 192.168.1.66 dev eth0
```

Responder a las siguientes cuestiones:

- a) ¿A cuántas redes está conectado directamente? ¿Cuáles son esas redes en notación CIDR?
- b) ¿Existe algún encaminador predeterminado? En caso afirmativo, ¿cuál es su dirección?
- c) ¿Es alcanzable la dirección 192.168.1.150? En caso afirmativo, ¿a través de qué interfaz? ¿Hace falta pasar por algún encaminador intermedio?
- d) ¿Es alcanzable la dirección 192.168.1.250? En caso afirmativo, ¿a través de qué interfaz? ¿Hace falta pasar por algún encaminador intermedio?
- e) ¿Es alcanzable la dirección 192.168.1.50? En caso afirmativo, ¿a través de qué interfaz? ¿Hace falta pasar por algún encaminador intermedio?

# Encaminamiento en Internet

- Internet está organizada en **Sistemas Autónomos** (*Autonomous Systems, AS*)
  - Un AS es una conjunto de redes y encaminadores gestionados y administrados por una misma autoridad
  - Cada AS se identifica mediante un número de AS (*AS Number, ASN*)
  - Hay más de 54.000 ASs
- **Encaminadores internos** del AS
  - Interconectan únicamente redes dentro del propio AS
  - Sólo conocen en detalle la organización del AS local
  - Desconocen el camino a otros ASs y su organización interna
  - Utilizan protocolos de encaminamiento denominados IGP (*Interior Gateway Protocol*)
- **Encaminadores externos** o frontera (*border router*)
  - Interconectan varios ASs
  - Conocen el camino al resto de ASs de la red, pero no conocen la organización interna de los mismos
  - Utilizan protocolos de encaminamiento denominados EGP (*Exterior Gateway Protocol*)
  - El protocolo usado en la actualidad es BGP (vector de rutas)

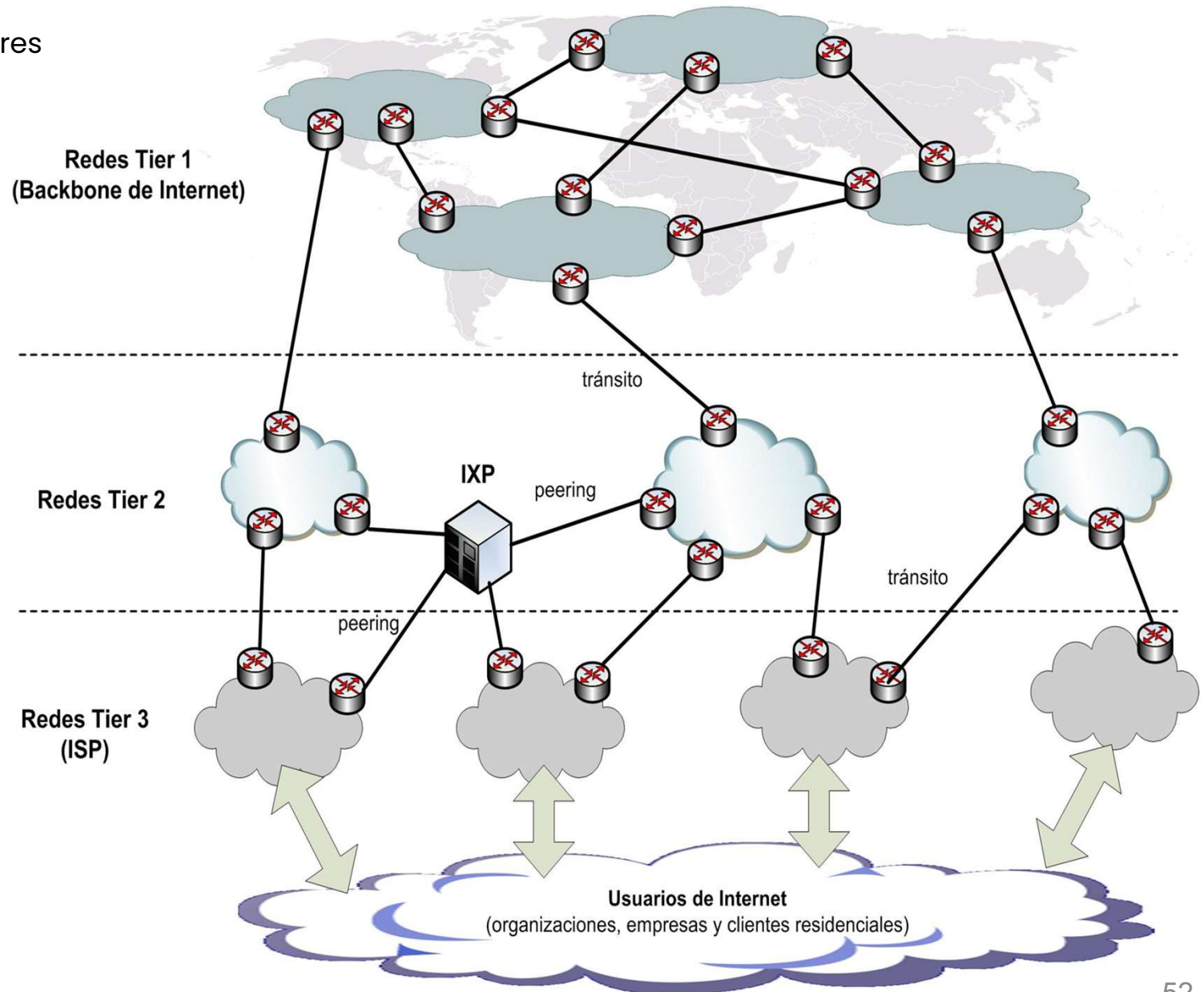
# Encaminamiento en Internet

<https://www.submarinecablemap.com/>

**La estructura actual de Internet** está basada en la interconexión de redes de forma más o menos jerárquica con varios niveles, conocidos como TIER. De forma general existen tres niveles

- Redes de los grandes operadores que tienen tendidos de fibra óptica por al menos dos continentes
- Todas las redes TIER 1 tienen que estar conectadas entre sí
- Desde una red TIER 1 se puede acceder a cualquier punto de Internet

- Operadores de ámbito más regional
- Para alcanzar todos los puntos de Internet necesitan conectarse a una red Tier 1
- Pertenecen a operadores que dan servicio de conexión a Internet a los usuarios residenciales y a muchas empresas

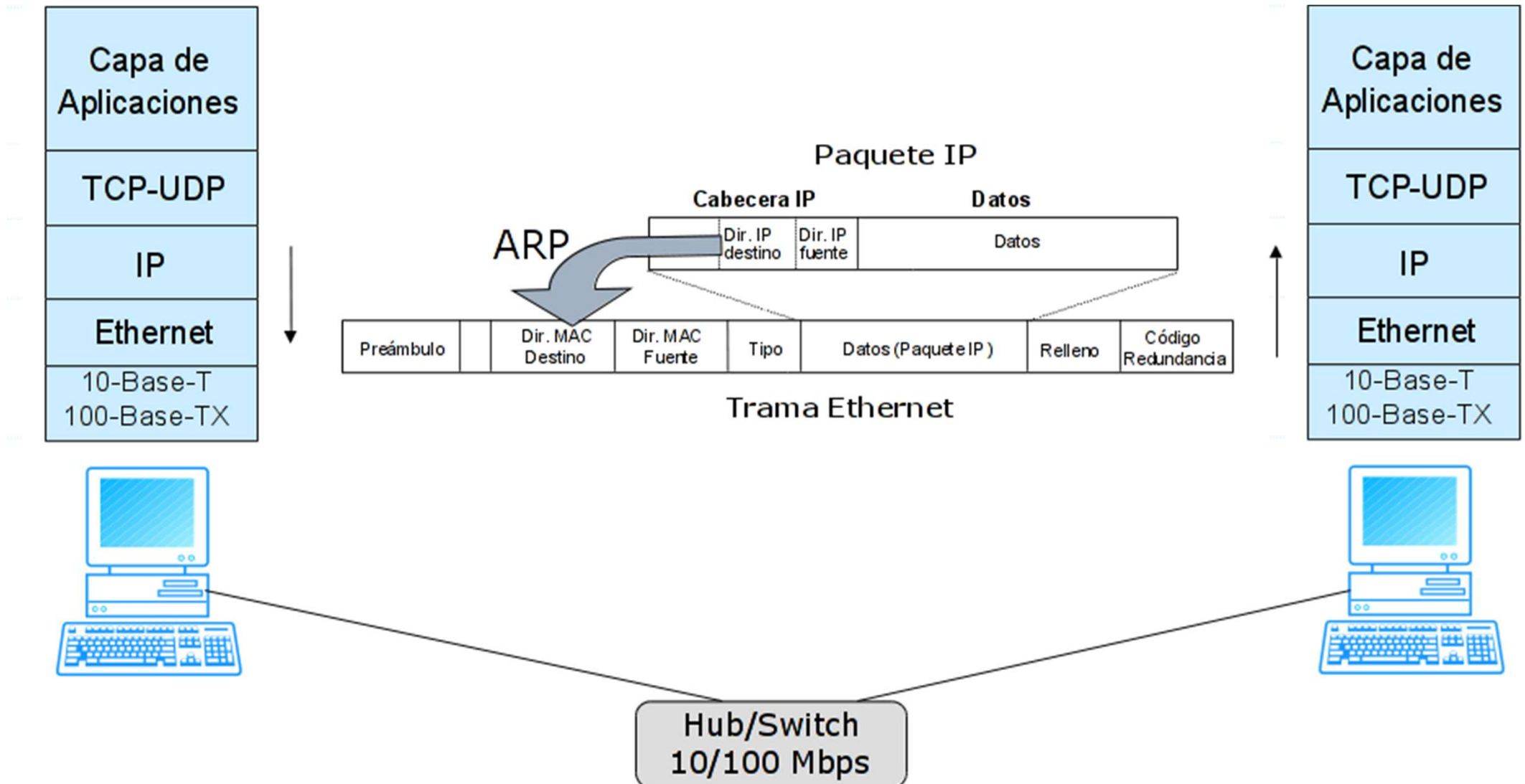


# **Otros Protocolos de Red: ARP**

# Protocolo de traducción de direcciones: ARP

## ARP: Address Resolution Protocol

- Traducción: Dirección IP → Dirección MAC



# Protocolo de traducción de direcciones: ARP

## La tabla ARP

- Mantiene las direcciones IP de las últimas máquinas con las que nos hemos comunicado y las direcciones Ethernet asociadas
- Ejemplo de tabla ARP en Linux

```
# arp -an
? (147.96.80.1) at 0:0:c:9f:f0:50 [ether] on eth0
? (147.96.80.11) at 54:a0:50:7b:f5:8f [ether] on eth0
? (147.96.80.255) at ff:ff:ff:ff:ff:ff [ether] on eth0
? (147.96.80.10) at (incomplete) [ether] on eth0
```

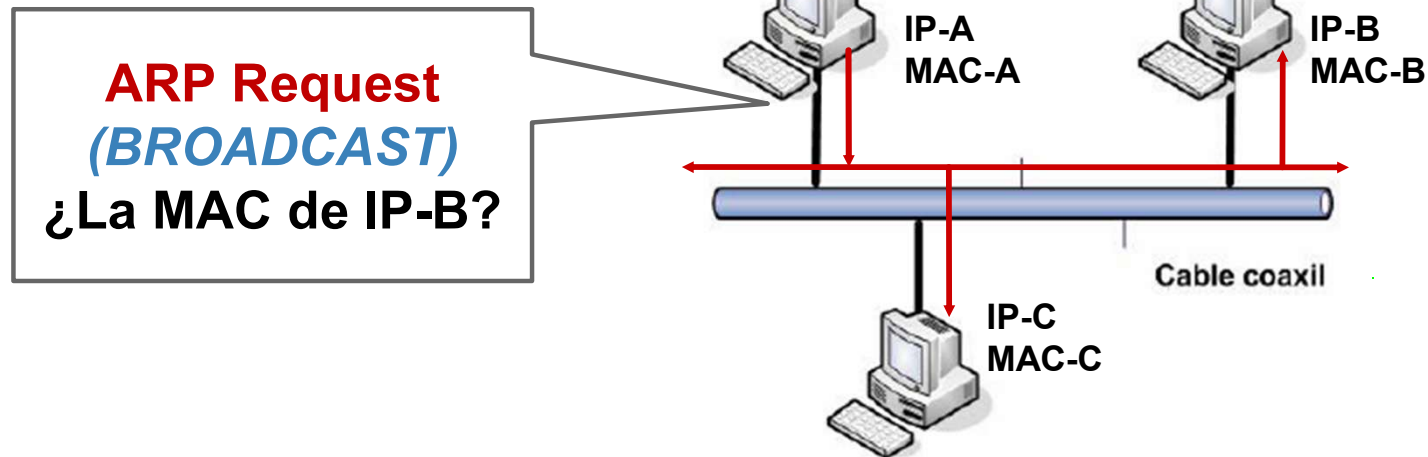
## Funcionamiento de ARP (HostA envía un paquete a HostB)

- HostA consulta su tabla ARP para ver si la dirección MAC de HostB está contenida en dicha tabla
- Si la dirección MAC de HostB no está en la tabla entonces envía un mensaje **broadcast** preguntando por la dirección MAC de HostB → **ARP Request**
- HostB responde a HostA informándole de su dirección MAC → **ARP Response**

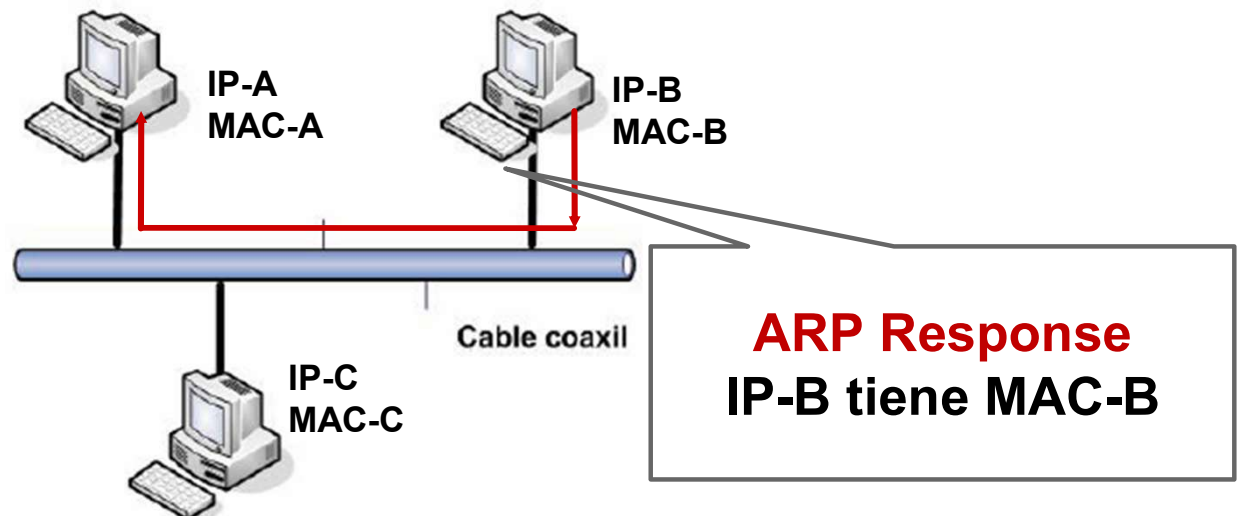
# Protocolo de traducción de direcciones: ARP

Funcionamiento de ARP (HostA envía un paquete al HostB)

- Pregunta ARP



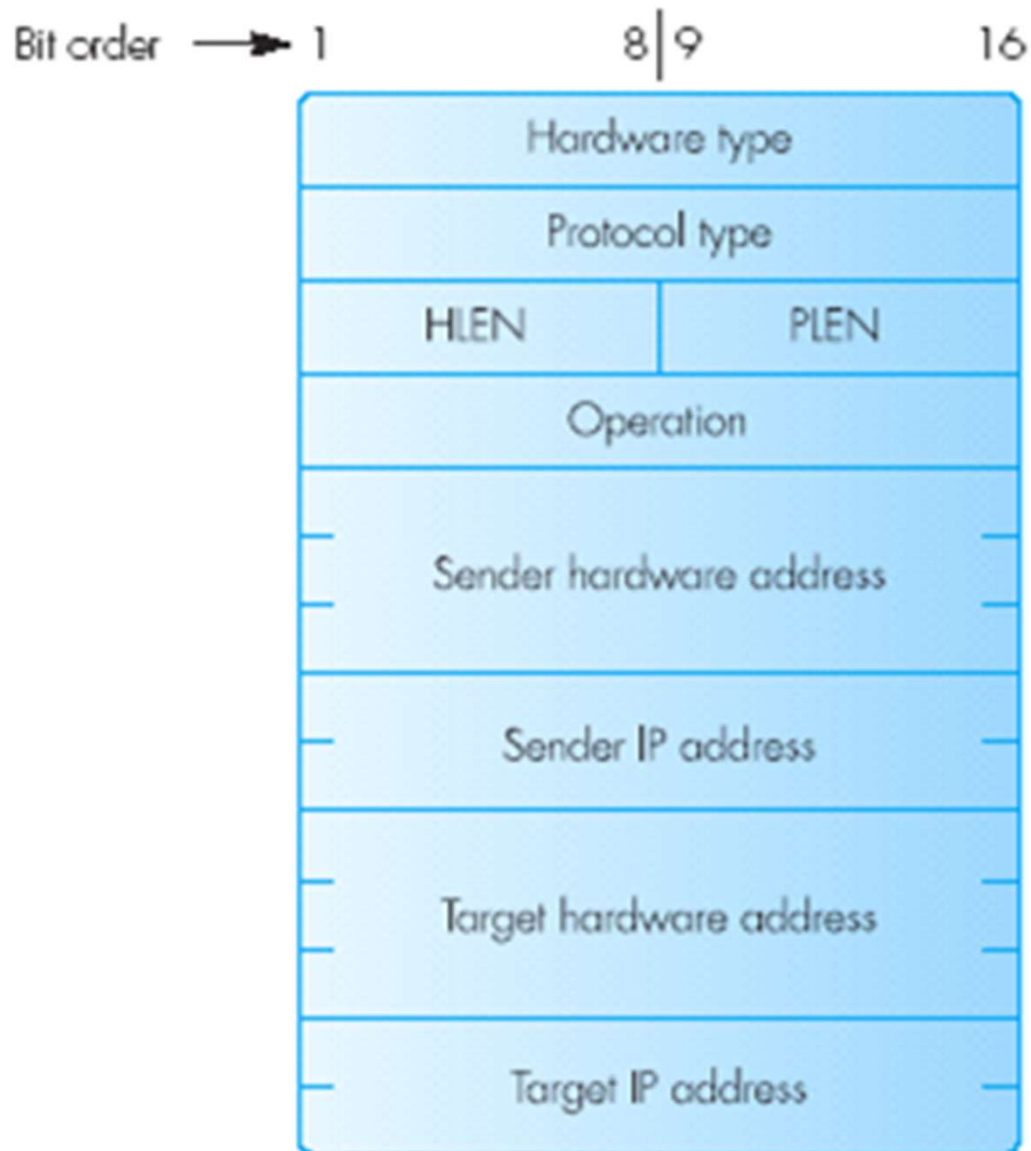
- Respuesta ARP





# Protocolo de traducción de direcciones: ARP

## Formato del paquete ARP



- HLEN: Longitud de la dir. enlace (48 bits, Ethernet)
- PLEN: Longitud de la dir. de red (32 bits, IP)
- Código de Operación:
  - 1 pregunta ARP
  - 2 respuesta ARP
  - 3 pregunta ARP inversa
  - 4 respuesta ARP inversa

# Protocolo de traducción de direcciones: ARP

**ARP Request (BROADCAST)** ¿La MAC de IP-B?

ETHER: ----- Ether Header -----

ETHER: Destination = ff:ff:ff:ff:ff:ff

ETHER: Source = 08:00:20:88:c9:ee

ETHER: Ethertype = 0806 (ARP)

ARP: ----- ARP/RARP Frame -----

ARP: Hardware type = 1

ARP: Protocol type = 0800 (IP)

ARP: Length of hardware address = 6 bytes

ARP: Length of protocol address = 4 bytes

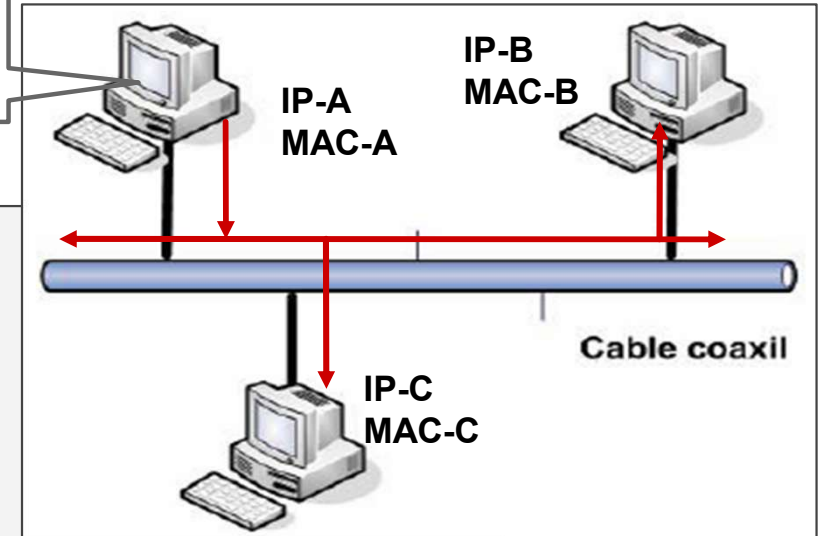
ARP: Opcode 1 (ARP Request)

ARP: Sender's hardware address = 08:00:20:88:c9:ee

ARP: Sender's protocol address = 200.96.21.31

ARP: Target hardware address = 00:00:00:00:00:00 (unknown)

ARP: Target protocol address = 200.96.21.120



# Protocolo de traducción de direcciones: ARP

ETHER: ----- Ether Header -----

ETHER: Destination = 08:00:20:88:c9:ee

ETHER: Source = 00:03:ba:0d:e7:0e

ETHER: Ethertype = 0806 (ARP)

ARP: ----- ARP/RARP Frame -----

ARP: Hardware type = 1

ARP: Protocol type = 0800 (IP)

ARP: Length of hardware address = 6 bytes

ARP: Length of protocol address = 4 bytes

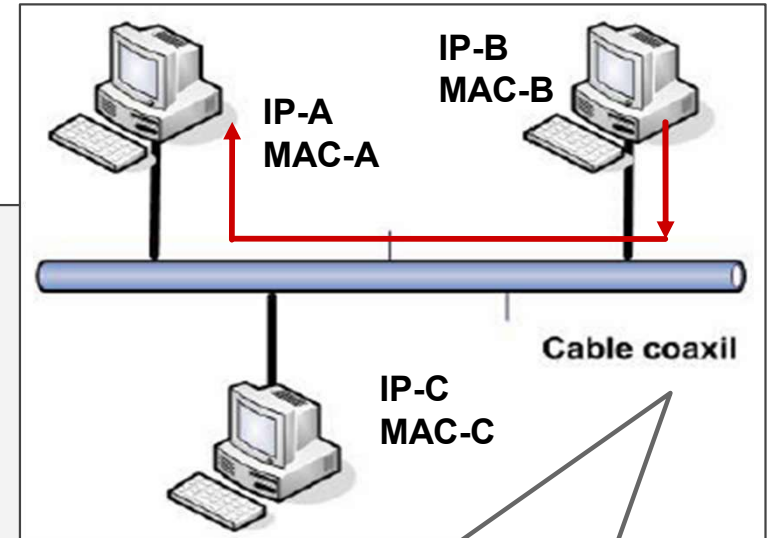
ARP: Opcode 2 (ARP Reply)

ARP: Sender's hardware address = 00:03:ba:0d:e7:0e

ARP: Sender's protocol address = 200.96.21.120

ARP: Target hardware address = 08:00:20:88:c9:ee

ARP: Target protocol address = 200.96.21.31



**ARP Response. IP-B es MAC-B**

# Ejemplo: Realizar el procedimiento de comunicación completo

**NOTA:** Las tablas ARP inicialmente están vacías

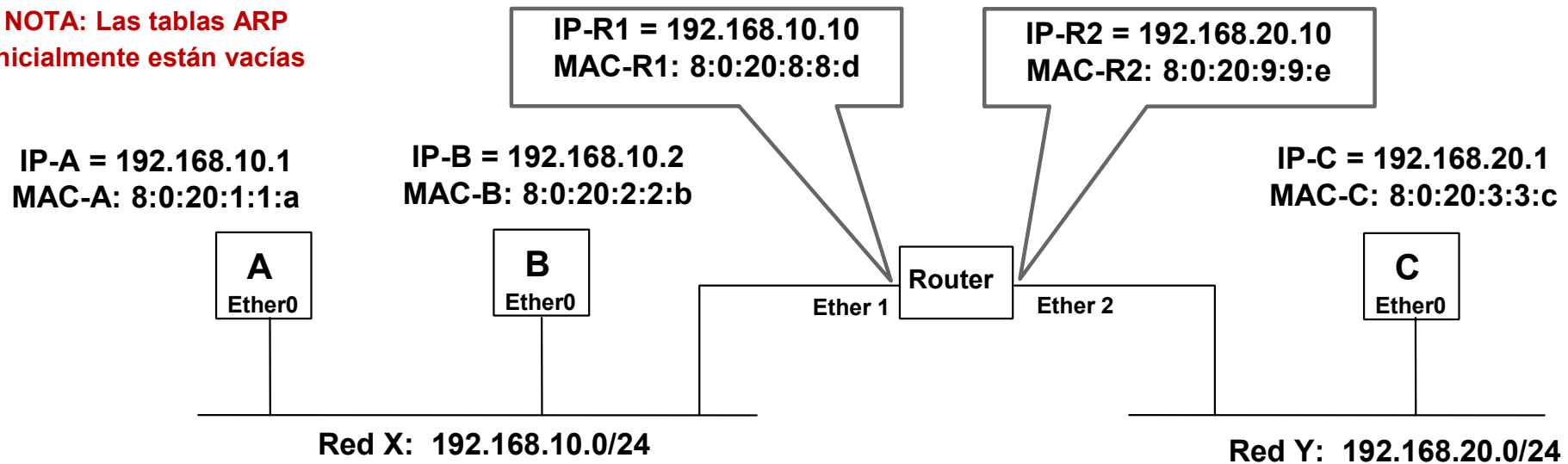


Tabla de encaminamiento de A

```
192.168.10.0/24 dev eth0 proto kernel scope link src 192.168.10.1
192.168.20.0 via 192.168.10.10 dev eth0
```

Tabla de encaminamiento de B

```
192.168.10.0/24 dev eth0 proto kernel scope link src 192.168.10.2
192.168.20.0 via 192.168.10.10 dev eth0
```

Tabla de encaminamiento de C

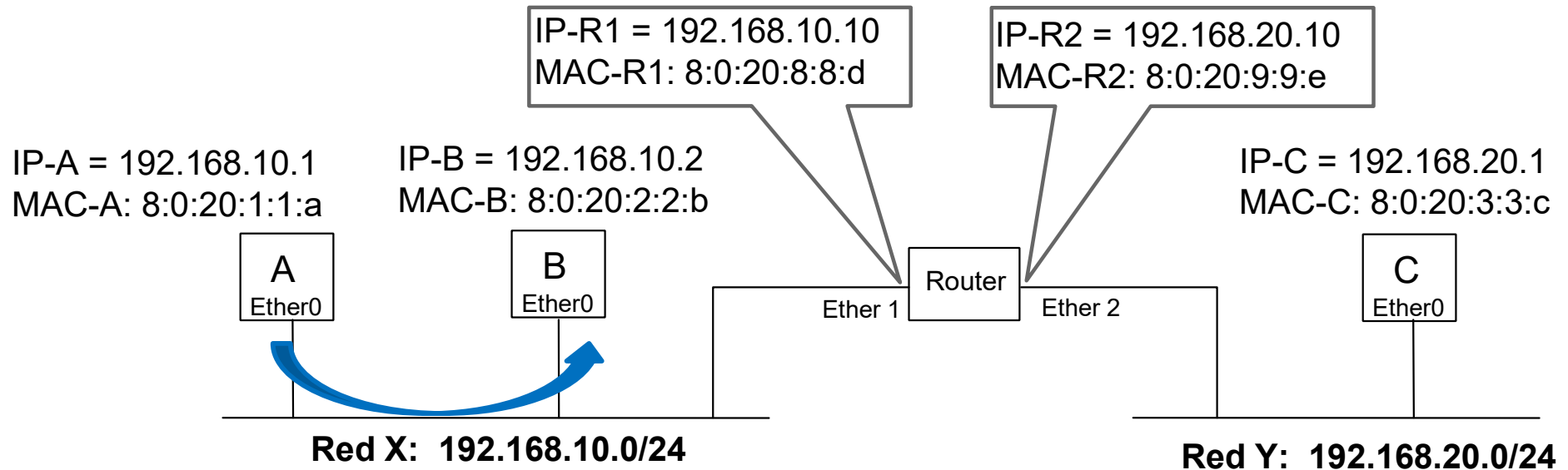
```
192.168.20.0/24 dev eth0 proto kernel scope link src 192.168.20.1
192.168.10.0/24 via 192.168.20.10 dev eth0
```

Tabla de encaminamiento del Router

```
192.168.10.0/24 dev eth1 proto kernel scope link src 192.168.10.10
192.168.20.0/24 dev eth2 proto kernel scope link src 192.168.20.10
```

# Ejemplo (cont.)

## Pasos que realiza el Host A cuando envía un mensaje al Host B



## Pasos que realiza el host A

1. Toma la dirección de destino (IP-B) y busca en su tabla de rutas una coincidencia:
  - Para cada una de las entradas de la tabla de rutas, aplica la máscara de red, y comprueba si coincide con la red destino
  - La IP destino (IP-B) coincide con la entrada 1 de la tabla de rutas de A ⇒ significa que el host destino está en la misma red (Red X) ⇒ entrega directa
2. Consulta su tabla ARP para buscar **la MAC asociada a IP-B (MAC-B)** → No la conoce

Tabla de encaminamiento de A

```
192.168.10.0/24 dev eth0 proto kernel scope link src 192.168.10.1
192.168.20.0 via 192.168.10.10 dev eth0
```

Tabla ARP de A

Dir. IP	Dir. MAC

# Ejemplo (cont.)

## Pasos que realiza el host A (cont.)

3. Utiliza el protocolo ARP para averiguar la MAC-B. A y B actualizan sus tablas ARP

Cabecera Ethernet  
(Capa de enlace)

Mensaje protocolo ARP  
(Capa de red)

MAC Dest.	MAC Src.	Tipo	Oper.	Hw_src.	IP_src.	Hw_dest.	IP_dest.
FF:....:FF	MAC-A	ARP	1	MAC-A	IP-A	00:....:00	IP-B

ARP Request

MAC Dest.	MAC Src.	Tipo	Oper.	Hw_src.	IP_src.	Hw_dest.	IP_dest.
MAC-A	MAC-B	ARP	2	MAC-B	IP-B	MAC-A	IP-A

ARP Responce

Tabla ARP B

Dir. IP	Dir. MAC
IP-A	MAC-A

IP-A = 192.168.10.1  
MAC-A: 8:0:20:1:1:a

IP-B = 192.168.10.2  
MAC-B: 8:0:20:2:2:b

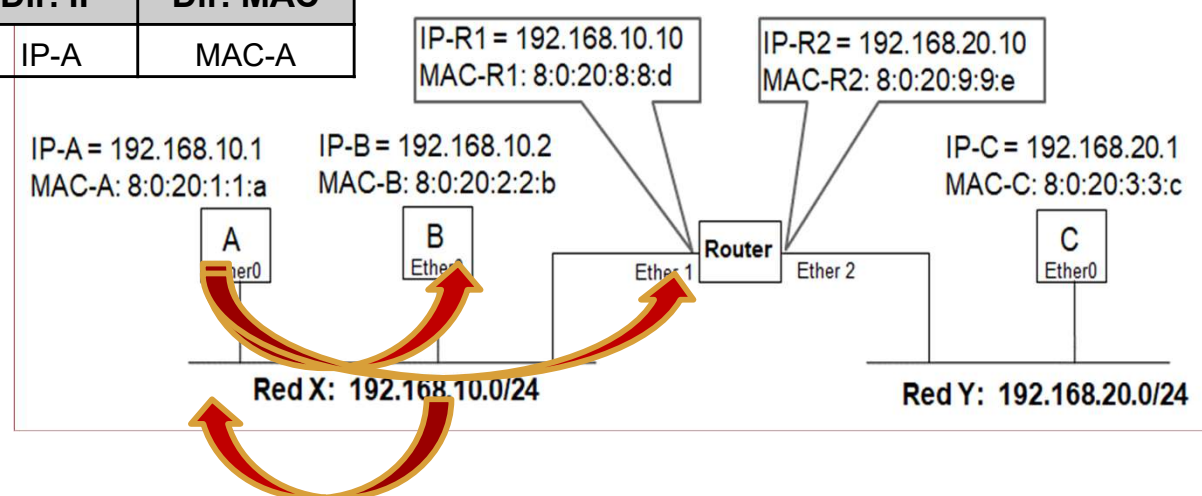
IP-R1 = 192.168.10.10  
MAC-R1: 8:0:20:8:8:d

IP-R2 = 192.168.20.10  
MAC-R2: 8:0:20:9:9:e

IP-C = 192.168.20.1  
MAC-C: 8:0:20:3:3:c

Tabla ARP A

Dir. IP	Dir. MAC
IP-B	MAC-B

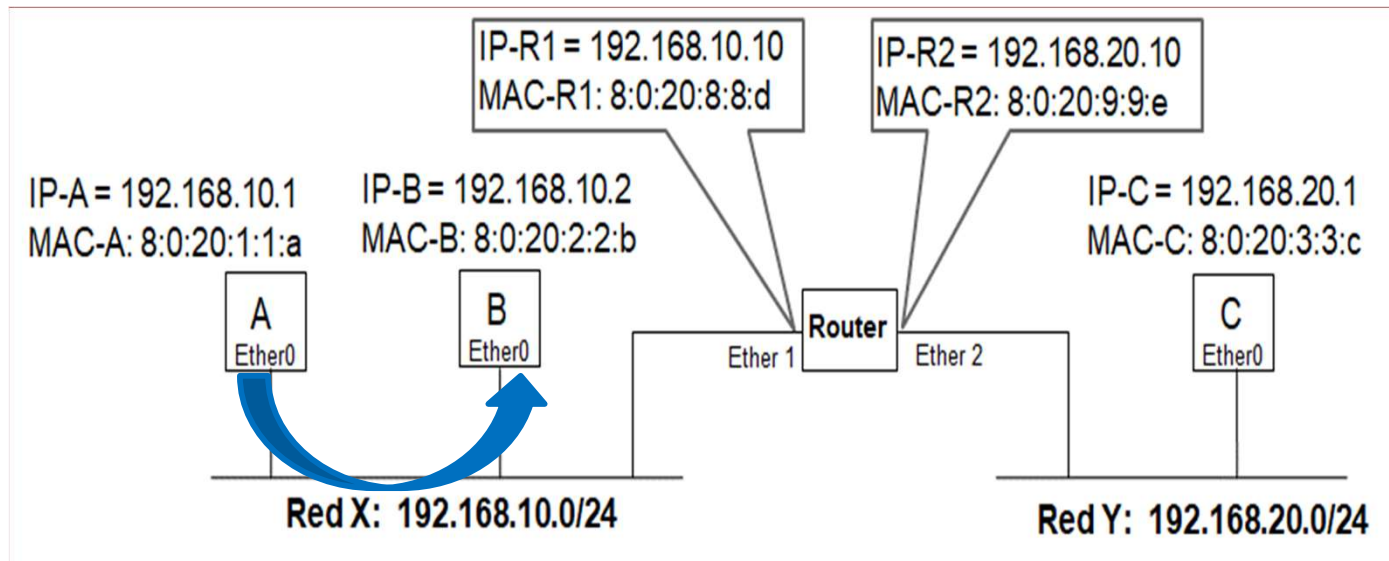


# Ejemplo (cont.)

## Pasos que realiza el host A (cont.)

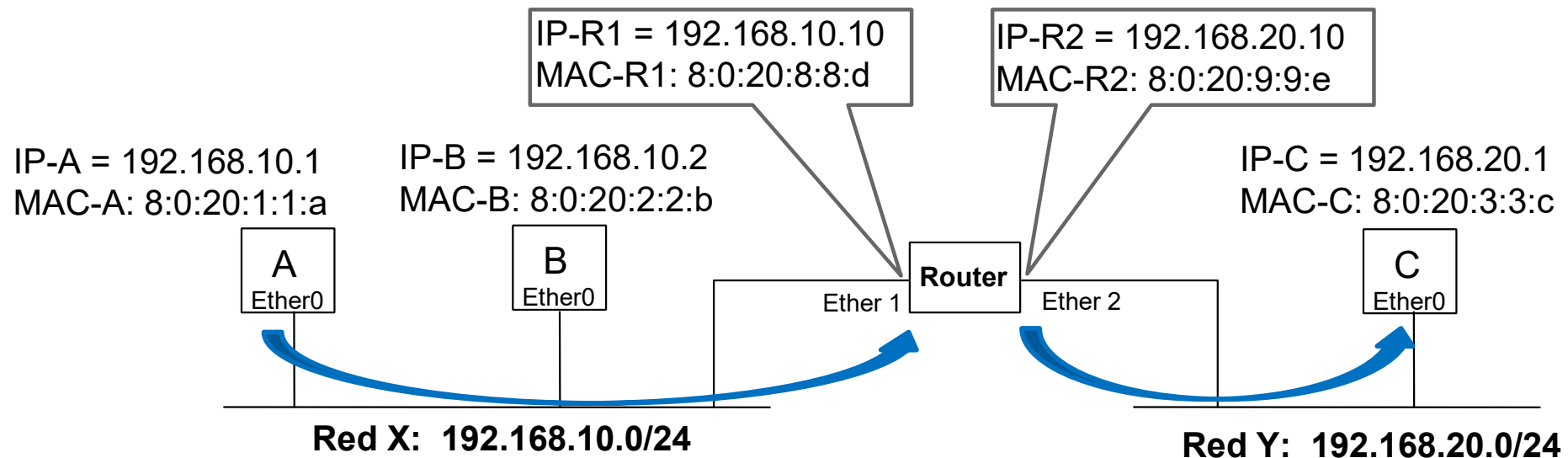
4. Envía el paquete a B: el paquete IP viaja del host A al host B, a través de la Red X, dentro de una trama Ethernet:

Cabecera Ethernet (Capa de enlace)			Cabecera IP (Capa de Red)		
MAC Dest.	MAC Src.	Tipo	IP Scr.	IP Dest.	
MAC-B	MAC-A	IP	IP-A	IP-B	Datos



# Ejemplo (cont.)

## Pasos que realiza el Host A cuando envía un mensaje al Host C



## Pasos que realiza el host A

1. Toma la dirección de destino (IP-C) y busca en su tabla de rutas una coincidencia:
  - Para cada una de las entradas de la tabla de rutas, aplica la máscara de red, y comprueba si coincide con la red destino
  - La IP destino (IP-C) está en otra red (Red Y) hay que usar la puerta de enlace IP-R1
2. Consulta su tabla ARP para buscar **la MAC asociada a IP-R1 (MAC-R1)** → No la conoce

Tabla de encaminamiento de A

```
192.168.10.0/24 dev eth0 proto kernel scope link src 192.168.10.1
192.168.20.0 via 192.168.10.10 dev eth0
```

Tabla ARP de A

Dir. IP	Dir. MAC
IP-B	MAC-B



# Ejemplo (cont.)

## Pasos que realiza el host A (cont.)

- Utiliza el protocolo ARP para averiguar la MAC-R1. A y el router actualizan sus tablas ARP

Cabecera Ethernet (Capa de enlace)				Mensaje protocolo ARP (Capa de red)				ARP Request
MAC Dest.	MAC Src.	Tipo	Oper.	Hw_src.	IP_src.	Hw_dest.	IP_dest.	
FF:....:FF	MAC-A	ARP	1	MAC-A	IP-A	00:....:00	IP-R1	

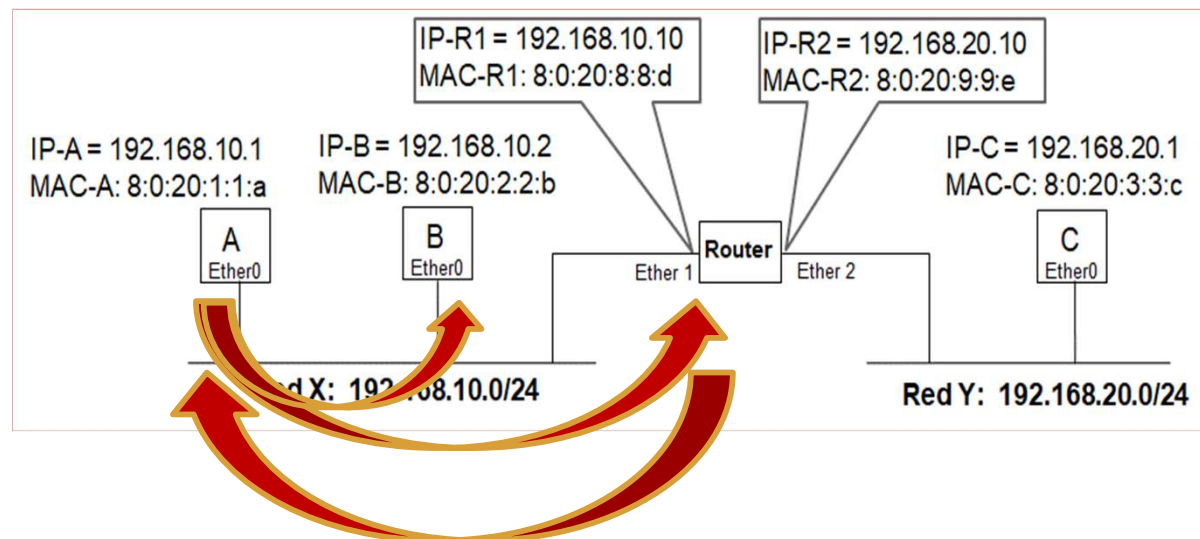
MAC Dest.	MAC Src.	Tipo	Oper.	Hw_src.	IP_src.	Hw_dest.	IP_dest.	ARP Response
MAC-A	MAC-R1	ARP	2	MAC-R1	IP-R1	MAC-A	IP-A	

Tabla ARP del Router

Dir. IP	Dir. MAC
IP-A	MAC-A

Tabla ARP de A

Dir. IP	Dir. MAC
IP-B	MAC-B
IP-R1	MAC-R1

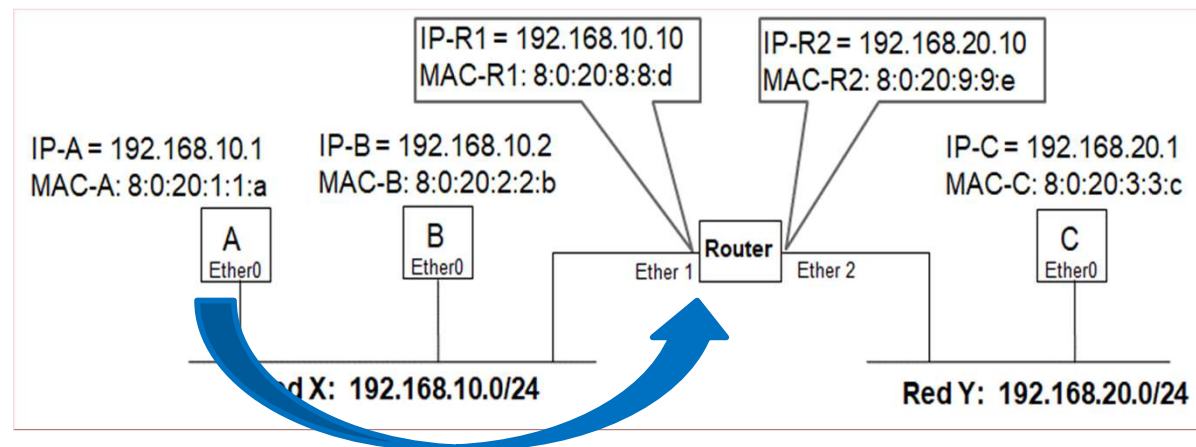


# Ejemplo (cont.)

## Pasos que realiza el host A (cont.)

4. Envía el paquete al router: el paquete IP viaja del host A al router (eth1), a través de la Red X, dentro de una trama Ethernet:

Cabecera Ethernet (Capa de enlace)			Cabecera IP (Capa de Red)		
MAC Dest.	MAC Src.	Tipo	IP Scr.	IP Dest.	
MAC-R1	MAC-A	IP	IP-A	IP-C	Datos



# Ejemplo (cont.)

**Cuando el router recibe el paquete, aplica los siguientes pasos :**

## Pasos que realiza el router

1. El paquete va dirigido a otra máquina (IP-C)  $\Rightarrow$  el router debe reenviar el paquete
2. Toma la dirección de destino (IP-C) y busca en su tabla de rutas una coincidencia:
  - Para cada una de las entradas de la tabla de rutas, aplica la máscara de red, y comprueba si coincide con la red destino
  - La IP destino (IP-C) coincide con la entrada 2 de la tabla de rutas de R  $\Rightarrow$  el host destino está la misma red (Red Y)  $\Rightarrow$  entrega directa
3. **Consulta su tabla ARP para buscar la MAC asociada a IP-C (MAC-C)  $\rightarrow$  No la conoce**

Tabla de encaminamiento del Router

192.168.10.0/24	dev	eth1	proto	kernel	scope	link	src	192.168.10.10
192.168.20.0/24	dev	eth2	proto	kernel	scope	link	src	192.168.20.10

Tabla ARP del Router

Dir. IP	Dir. MAC
IP-A	MAC-A

# Ejemplo (cont.)

## Pasos que realiza el router (cont.)

- Utiliza el protocolo ARP para averiguar la MAC-C. El router y C actualizan sus tablas ARP

Cabecera Ethernet  
(Capa de enlace)

MAC Dest.	MAC Src.	Tipo	Oper.	Hw_src.	IP_src.	Hw_dest.	IP_dest.
FF:....FF	MAC-R2	ARP	1	MAC-R2	IP-R2	00:....:00	IP-C

Mensaje protocolo ARP  
(Capa de red)

MAC Dest.	MAC Src.	Tipo	Oper.	Hw_src.	IP_src.	Hw_dest.	IP_dest.
MAC-R2	MAC-C	ARP	2	MAC-C	IP-C	MAC-R2	IP-R2

ARP Request

ARP Response

Tabla ARP del router

Dir. IP	Dir. MAC
IP-A	MAC-A
IP-C	MAC-C

IP-A = 192.168.10.1  
MAC-A: 8:0:20:1:1:a

IP-B = 192.168.10.2  
MAC-B: 8:0:20:2:2:b

IP-R1 = 192.168.10.10  
MAC-R1: 8:0:20:8:8:d

IP-R2 = 192.168.20.10  
MAC-R2: 8:0:20:9:9:e

IP-C = 192.168.20.1  
MAC-C: 8:0:20:3:3:c

A  
Ether0

B  
Ether0

Ether 1

Router

Ether 2

C  
Ether0

Red X: 192.168.10.0/24

Red Y: 192.168.20.0/24

Tabla ARP de C

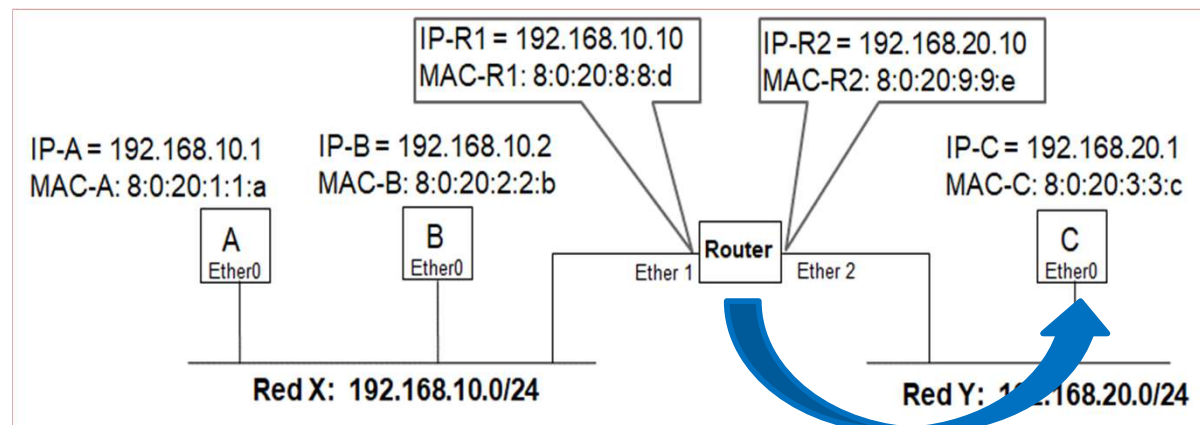
Dir. IP	Dir. MAC
IP-R2	MAC-R2

# Ejemplo (cont.)

## Pasos que realiza el router (cont.)

5. Envía paquete a C: el paquete IP viaja del Router (eth2) al host C, a través de la Red Y, dentro de una trama Ethernet:

Cabecera Ethernet (Capa de enlace)			Cabecera IP (Capa de Red)		
MAC Dest.	MAC Src.	Tipo	IP Scr.	IP Dest.	
MAC-C	MAC-R2	IP	IP-A	IP-C	Datos

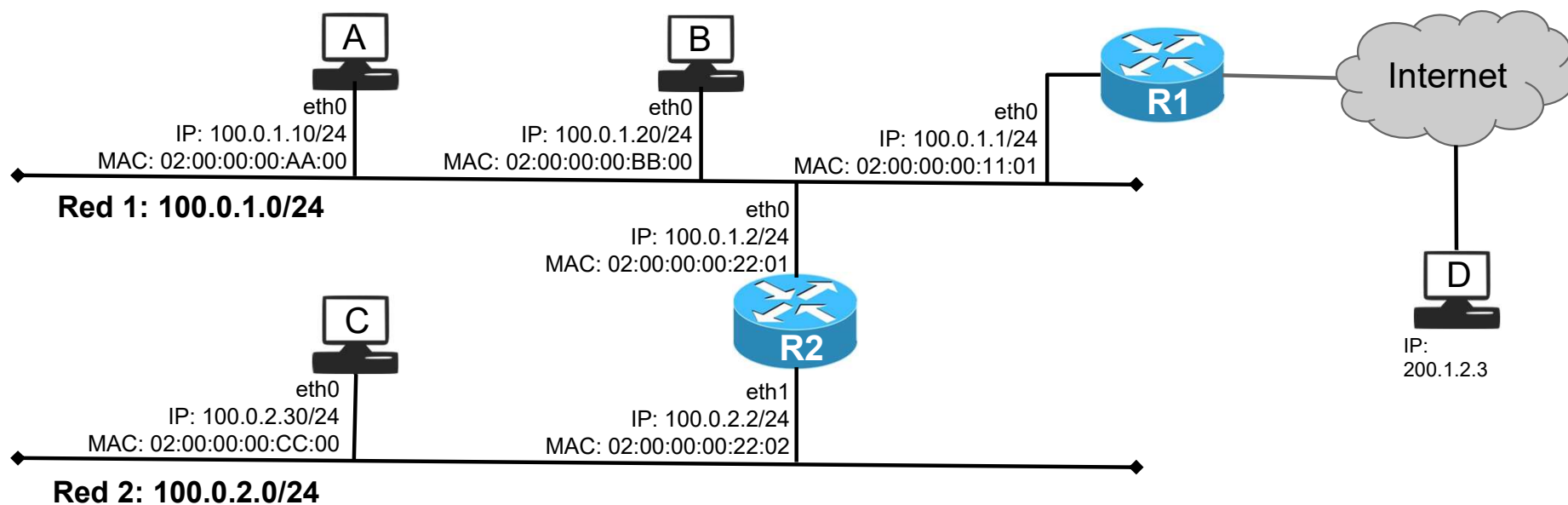


# Ejercicio 4

Dada la siguiente red:

- Escribe las tablas de encaminamiento de cada una de los Host y de los Routers
- Realizar el procedimiento de comunicación completo en cada uno de los siguientes casos (**indicar todos los pasos que se realizan**)

- ✓ **Caso 1: comunicación A → B**
- ✓ **Caso 2: comunicación A → C**
- ✓ **Caso 3: comunicación A → D**



**NOTA: Las tablas ARP inicialmente están vacías**

# Otros Protocolos de Red: ICMP

# Protocolo ICMP

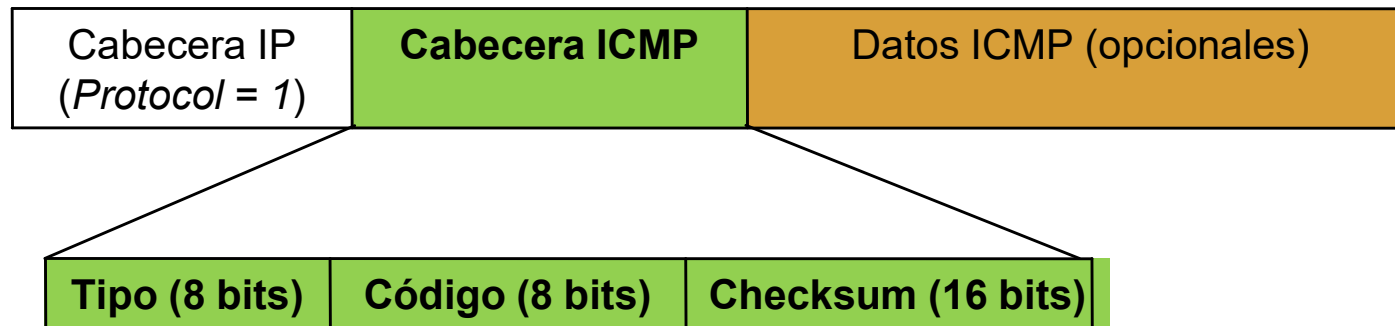
**ICMP (Internet Control Message Protocol):** Protocolo de mensajes de control de Internet

- Es un protocolo para el intercambio de mensajes de control en la red
- Los mensajes ICMP se pueden clasificar en dos tipos:
  - **Mensajes de error**
    - Permiten informar de situaciones de error en la red
    - Ejemplos: destino inalcanzable, tiempo excedido, problema de parámetro, etc.
  - **Mensajes informativos**
    - Permiten intercambiar información sobre la presencia o el estado de un determinado sistema
    - Ejemplos: mensajes de eco, anuncio o solicitud de router, redirecciones, etc.



# Protocolo ICMP: Formato del mensaje

- Los mensajes ICMP se transmiten dentro de **paquetes IP**
  - El protocolo ICMP se corresponde con el identificador 1



- La **cabecera ICMP** contiene la siguiente información:
  - **Tipo** (8 bits): Indica el tipo del mensaje ICMP
  - **Código** (8 bits): Ofrece información adicional sobre el contenido del mensaje cuyo significado depende del tipo del mensaje
  - **Checksum** (16 bits): Es un campo para detectar errores en el mensaje ICMP

# Protocolo ICMP: Tipos de mensajes

Mensajes Error	
Tipo	Significado
3	Destination Unreachable
4	Source Quench
11	Time Exceeded
12	Parameter Problem

Mensajes Informativos	
Tipo	Significado
0	Echo Reply
5	Redirect
8	Echo Request
9	Router Solicitation
10	Router Advertisement

# Protocolo ICMP: Echo Request/Reply

- **Se utilizan para ver si un computador es alcanzable**
- **La orden ping:** Envía mensajes ICMP Echo Request y espera la recepción de mensajes ICMP Echo Reply
- **Formato de los mensajes Echo Request/Reply**
  - **Tipos:** 8 (Echo Request) y 0 (Echo Reply)
  - **Código:** 0 (no hay subtipos)
  - **Identificador:** Permite establecer la correspondencia entre solicitud y respuesta, ya que ambas tienen el mismo identificador
  - **Secuencia:** También se utiliza para establecer la correspondencia entre solicitud y respuesta, cuando se envían varias solicitudes consecutivas con el mismo identificador
  - **Datos:** Contiene un número determinado de bytes, generados aleatoriamente por la herramienta de diagnóstico (el tamaño se puede especificar como un parámetro de la orden `ping`)

Tipo (0/8)	Código (0)	Checksum
Identificador		Nº de secuencia
Datos		

# Protocolo ICMP: Destination Unreachable

- Estos mensajes **los envía el router cuando el destino de un paquete es inalcanzable, para informar al host emisor del paquete de esta situación**
- **Formato del mensaje**
  - Tipo: 3
  - Código: Especifica la razón por la cual el destino es inalcanzable
    - Véase a continuación la lista de códigos

Tipo (3)	Código	Checksum
No usado (cero)		
Cabecera IP del datagrama original + 64 primeros bits de datos		

# Protocolo ICMP: Destination Unreachable

## Valores del campo Código

- 0: Network unreachable:** Routing incorrecto
- 1: Host unreachable:** IP incorrecta/host desconectado
- 2: Protocol unreachable:** N° de protocolo incorrecto / No disponible
- 3: Port unreachable:** Puerto UDP cerrado
- 4: Fragmentation needed but the Do Not Fragment bit was set**
- 5: Source route failed**
- 6: Destination network unknown:**
- 7: Destination host unknown:**
- 9: Destination network administratively prohibited:** Red protegida con un firewall
- 10: Destination host administratively prohibited:** Host protegido con un firewall

# Protocolo ICMP: Destination Unreachable

## Mecanismo “Path MTU Discovery” (PMTUD)

- Se utiliza para evitar la fragmentación de los paquetes IP
  - Para ello, el host origen debe ajustar el tamaño de los paquetes a la “MTU del camino” (MTU mínima de todas las redes que debe atravesar)
- El mecanismo para descubrir la MTU del camino (PMTUD) es el siguiente
  - El host origen envía el paquete ajustándose a la MTU de su red local y con el bit DF activado.
  - Si el paquete debe atravesar una red con una MTU menor, el encaminador no puede fragmentarlo, al estar el bit DF activado por lo que descarta el datagrama y devuelve al emisor un mensaje ICMP de tipo 3 (destino inalcanzable), código 4 (fragmentación necesaria)
  - El host origen envía un nuevo paquete, que se ajusta a la MTU de dicha red.
  - El proceso se repite hasta que el paquete llega al destino

# Protocolo ICMP: Time Exceeded

- Este paquete lo puede enviar un router intermedio o el host destinatario:
  - Lo envía un router al host origen **cuando descarta el paquete por haber agotado su tiempo de vida (TTL de tránsito)**
  - Lo envía el host destinatario en el caso de un paquete fragmentado, **cuando no puede reensamblar por falta de algún fragmento y se agota el tiempo de espera para reensamblado (TTL de reensamblado)**
- **Formato del mensaje**
  - Tipo: 11
  - Código: 0 (agotado TTL de tránsito) o 1 (agotado TTL de reensamblado)

Tipo (11)	Código (0/1)	Checksum
No usado		
Cabecera IP del datagrama original + 64 primeros bits de datos		

# Protocolo ICMP: Otros mensajes

## ICMP Redirect

- Los mensajes de redirección los envía un router cuando un host no está eligiendo la ruta adecuada hacia un determinado destino
- El mensaje de redirección le indica al host cuál es el camino más adecuado para alcanzar dicho destino

## ICMP Source Quench

- Se utilizan para notificar al emisor que debe reducir el ritmo de envío de paquetes
- Los puede enviar un router intermedio o el host destinatario, cuando no tienen capacidad para procesar los paquetes procedentes de un determinado emisor

## ICMP Parameter Problem

- Indica que se ha encontrado algún tipo de problema durante el procesamiento de los parámetros de la cabecera IP
  - Valor inválido en la cabecera o checksum erróneo
  - Campo opción necesario, pero no presente

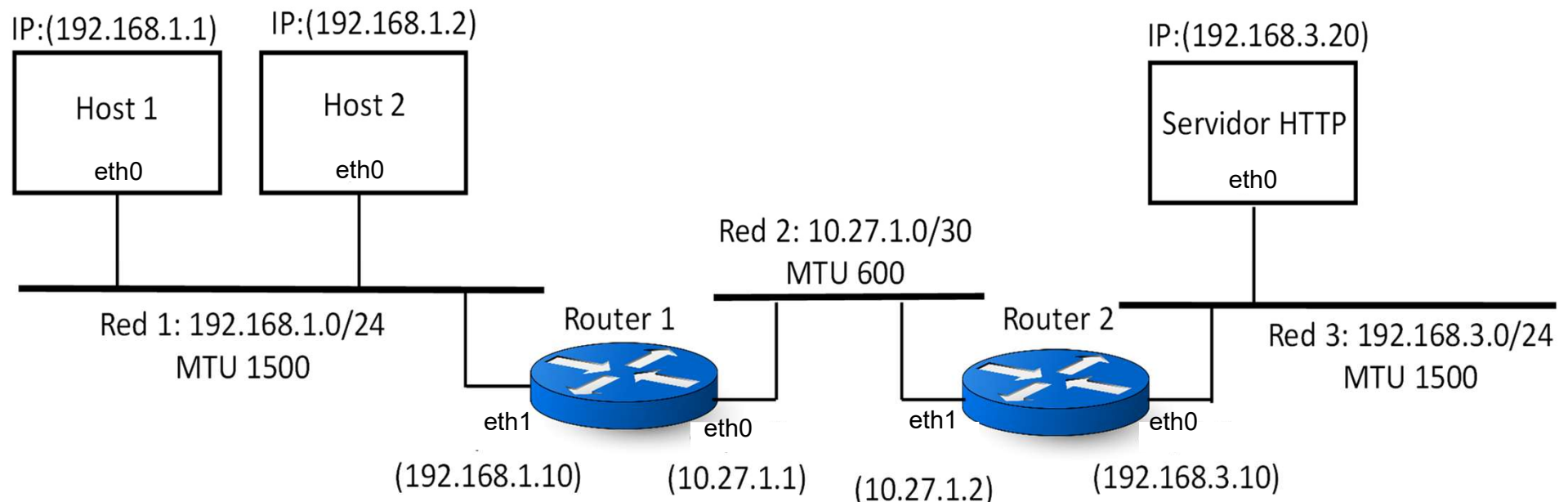


# Ejercicio 2 ampliado

Usando la red del ejercicio 2 (transparencia 44) y sabiendo que las *tablas ARP* de todos los nodos ESTÁN VACIAS, **indicar las tramas que circularán por la red** en los siguientes casos

- a) El Host 2 envía un ping a Host 1
- b) A continuación, desde el Host 2 se envía un ping al Servidor HTTP

NOTA: Indicar cómo quedan las tablas ARP al final



# Subredes

# Subredes

## Ventajas de las subredes

- Permite **aislar el tráfico** entre las distintas subredes
- Se **reduce el tráfico global**
- Permite limitar y proteger el acceso a las distintas subredes
- La comunicación entre éstas se realiza mediante un router
- Permite organizar la red en áreas o departamentos
- Se asigna a cada departamento un subconjunto de direcciones IP
- La gestión de las direcciones IP se puede delegar en el propio área o departamento
  - Se descentraliza la tarea de asignación de direcciones
  - Se facilita la tarea del administrador de la red

# Subredes

## Ejemplo: Supongamos la red de la clase B: 150.23.0.0

- Tenemos 16 bits para identificar al host ( $2^{16}$  hosts)

IP de red: 150. 23.0.0 =  $\overbrace{10010110.00010111}^{\text{Red}}.\overbrace{00000000.00000000}^{\text{Host}}$

Máscara: 255.255.0.0 =  $11111111.11111111.00000000.00000000$

- Dividir en 256 subredes con 256 hosts cada una
  - Usamos 8 bits para identificar a la subred ( $2^8 = 256$  subredes)
  - Usamos 8 bits para identificar al host ( $2^8 = 256$  hosts)

IP de red: 150. 23. 0.0 =  $\overbrace{10010110.00010111}^{\text{Red}}.\overbrace{ssssssss}^{\text{Subred}}.\overbrace{hhhhhhhh}^{\text{Host}}$

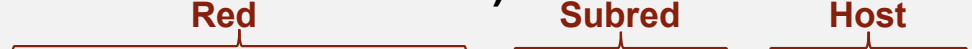
Máscara: 255.255.255.0 =  $11111111.11111111.11111111.00000000$

# Subredes

## Ejemplo (cont): Supongamos la red de la clase B: 150.23.0.0

- Nos queda la siguiente organización:

- Subred 0: 150.23.0.0** (Dpto. de administración)

Rango Dir.: 150.23.0.1 =  10010110.00010111.00000000.00000001  
150.23.0.254 = 10010110.00010111.00000000.11111110  
Dir. Broad. 150.23.0.255 = 10010110.00010111.00000000.11111111

- Subred 1: 150.23.1.0** (Dpto. de RRHH)

Rango Dir.: 150.23.1.1 = 10010110.00010111.00000001.00000001  
150.23.1.254 = 10010110.00010111.00000001.11111110  
Dir. Broad. 150.23.1.255 = 10010110.00010111.00000001.11111111

- ...


- Subred 255: 150.23.255.0** (Dpto. comercial)

Rango Dir.: 150.23.255.1 = 10010110.00010111.11111111.00000001  
150.23.255.254 = 10010110.00010111. 11111111.11111110  
Dir. Broad. 150.23.255.255 = 10010110.00010111. 11111111.11111111

# Subredes

## Ejemplo: Supongamos la red de la clase C: 192.168.44.0

- Queremos dividir la red en 8 subredes
  - 3 bits para identificar la subred ( $2^3 = 8$  subredes)
  - 5 bits para identificar el host ( $2^5 = 32$  dir. por subred)

IP: 192.168. 44. x =  11000000.10101000.00101100.ssshhhhh  
Máscara: 255.255.255.224 = 11111111.11111111.11111111.11100000

# Subredes

Ejemplo: Supongamos la red de la clase C: 192.168.44.0

IP: 192.168.44.x =

Máscara: 255.255.255.224 = 11111111.11111111.11111111.11100000

## Subred 192.168.44.0

hosts: de 192.168.44.1 al 192.168.44.30  
broadcast : 192.168.44.31

## Subred 192.168.44.32

hosts: de 192.168.44.33 al 192.168.44.62  
broadcast : 192.168.44.63

## Subred 192.168.44.64

hosts: de 192.168.44.65 al 192.168.44.94  
broadcast: 192.168.44.95

## Subred 192.168.44.96

hosts: de 192.168.44.97 al 192.168.44.126  
broadcast: 192.168.44.127

## Subred 192.168.44.128

hosts: de 192.168.44.129 al 192.168.44.158  
broadcast: 192.168.44.159

## Subred 192.168.44.160

hosts: de 192.168.44.161 al 192.168.44.190  
broadcast: 192.168.44.191

## Subred 192.168.44.192

hosts: de 192.168.44.193 al 192.168.44.222  
broadcast: 192.168.44.223

## Subred 192.168.44.224

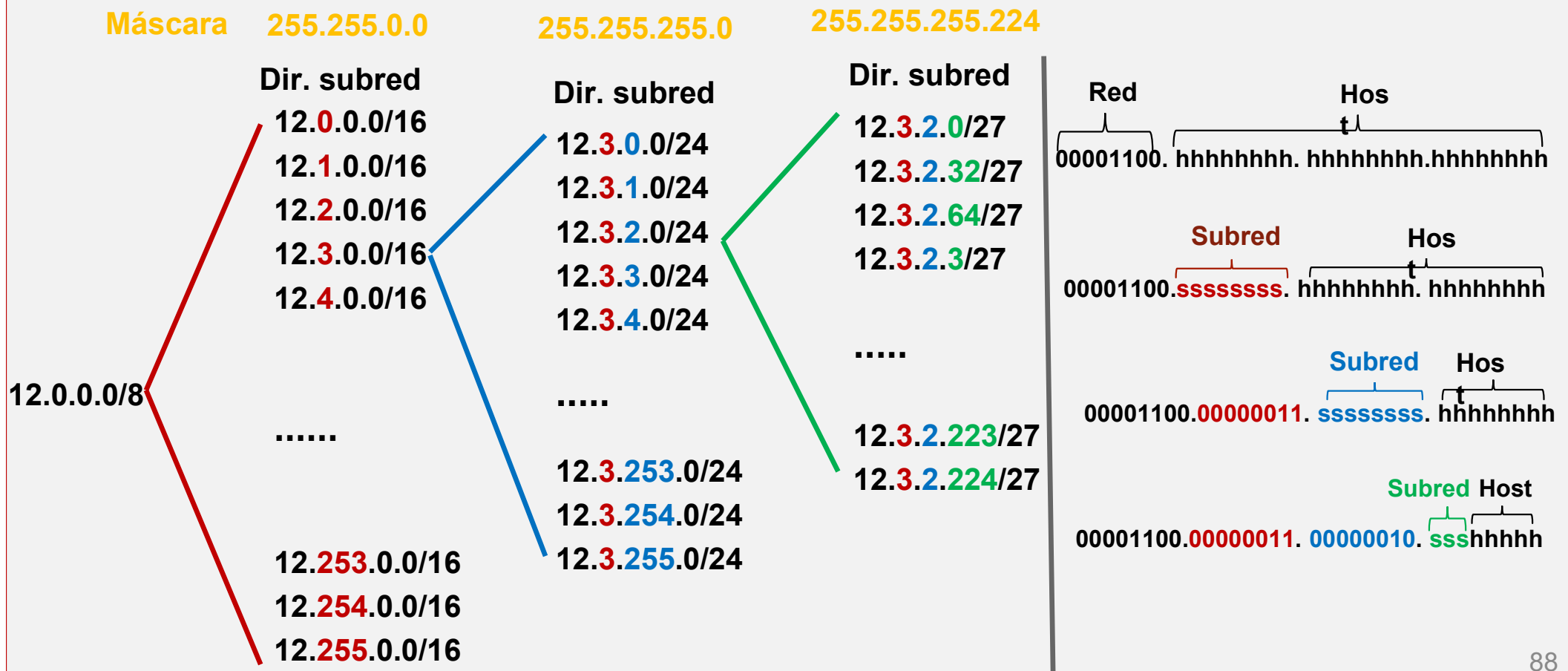
hosts: de 192.168.44.225 al 192.168.44.254  
broadcast: 192.168.44.255

# Subredes

## Máscaras de subred de longitud variable: VLSM (*Variable Length Subnet Mask*)

- Muchas organizaciones utilizan una organización jerárquica de direcciones de red
  - La red se divide en subredes, que a su vez se pueden dividir en sub-subredes
  - La longitud de la máscara de subred puede ser variable, para cada subred

**Ejemplo VLSM:** Supongamos la red de clase A 12.0.0.0 con la siguiente organización:





# Subredes

**Ejemplo VLSM:** Una red de clase C (200.21.32.0) quiere dividirse en cinco subredes:

Subred 1: 50 hosts

Subred 4: 30 hosts

Subred 2: 50 hosts

Subred 5: 30 hosts

Subred 3: 50 hosts

# Subredes

**Ejemplo VLSM:** Una red de clase C (200.21.32.0) quiere dividirse en cinco subredes:

Subred 1: 50 hosts

Subred 4: 30 hosts

Subred 2: 50 hosts

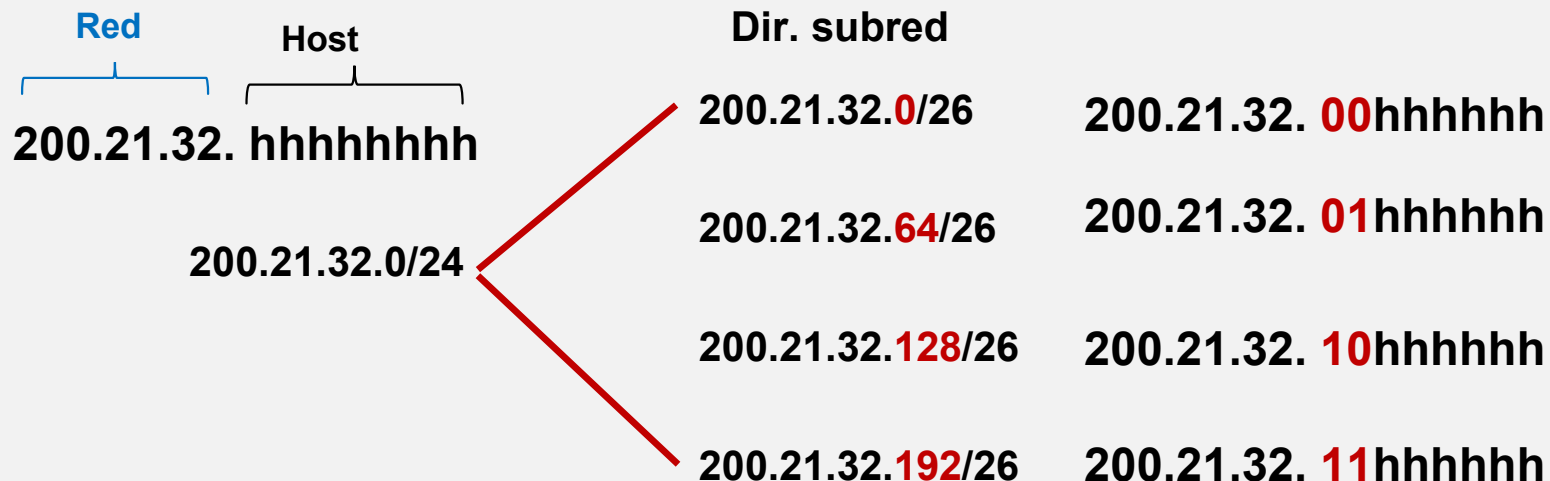
Subred 5: 30 hosts

Subred 3: 50 hosts

- Para tener 50 host se necesitan **6 bits para las direcciones de host**
  - Sobran 2 bits para las subredes:  $2^2 = 4$  subredes de 62 ( $2^6 - 2$ ) hosts cada una

Subred   Host  
200.21.32. sshhhhhh

- Máscara = 11111111 . 11111111 . 11111111 . 11000000 = **255.255.255.192**



# Subredes

**Ejemplo VLSM:** Una red de clase C (200.21.32.0) quiere dividirse en cinco subredes:

Subred 1: 50 hosts

Subred 4: 30 hosts

Subred 2: 50 hosts

Subred 5: 30 hosts

Subred 3: 50 hosts

- Subdividir una de las subredes en dos subredes de 30 hosts:
  - Se necesitan 5 bits para las direcciones de host de 30 ( $2^5-2$ ) hosts:

Subred      Host  
                  └─┬─┘  
200.21.32. **ss****s**hhhhh

- Máscara = 11111111 . 11111111 . 11111111 . 11100000 = **255.255.255.224**

Dir. subred

