

Redes de Computadores

Nivel de Red:

Reenvío IP + ICMP

Área de Ingeniería Telemática
Dpto. Automática y Computación
<http://www.tlm.unavarra.es/>

En clases anteriores...

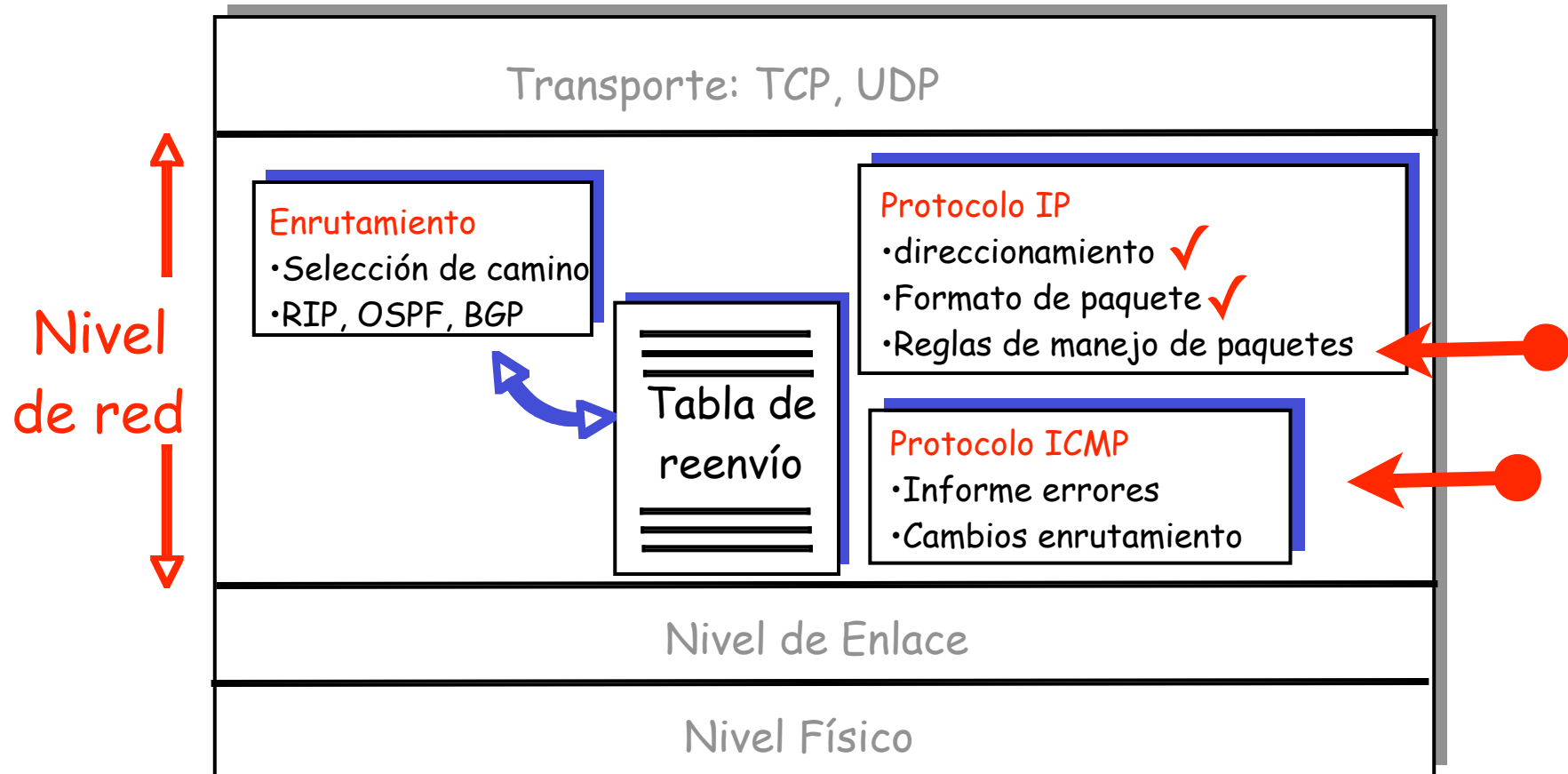
- ▶ El nivel de red IP
- ▶ Tabla de reenvío prefijos y direccionamiento IP

En esta clase...

- ▶ IP
 - > reglas de reenvío
- ▶ ICMP
 - > ping
 - > traceroute

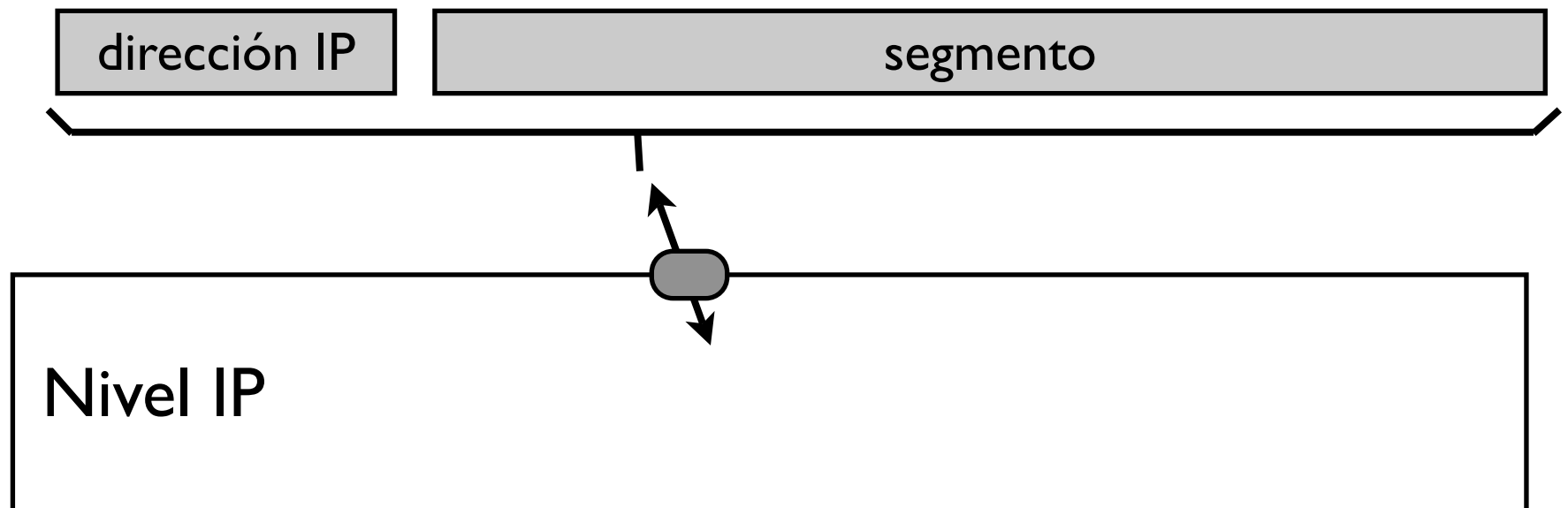
El nivel de Red de Internet

► Componentes del Nivel de Red



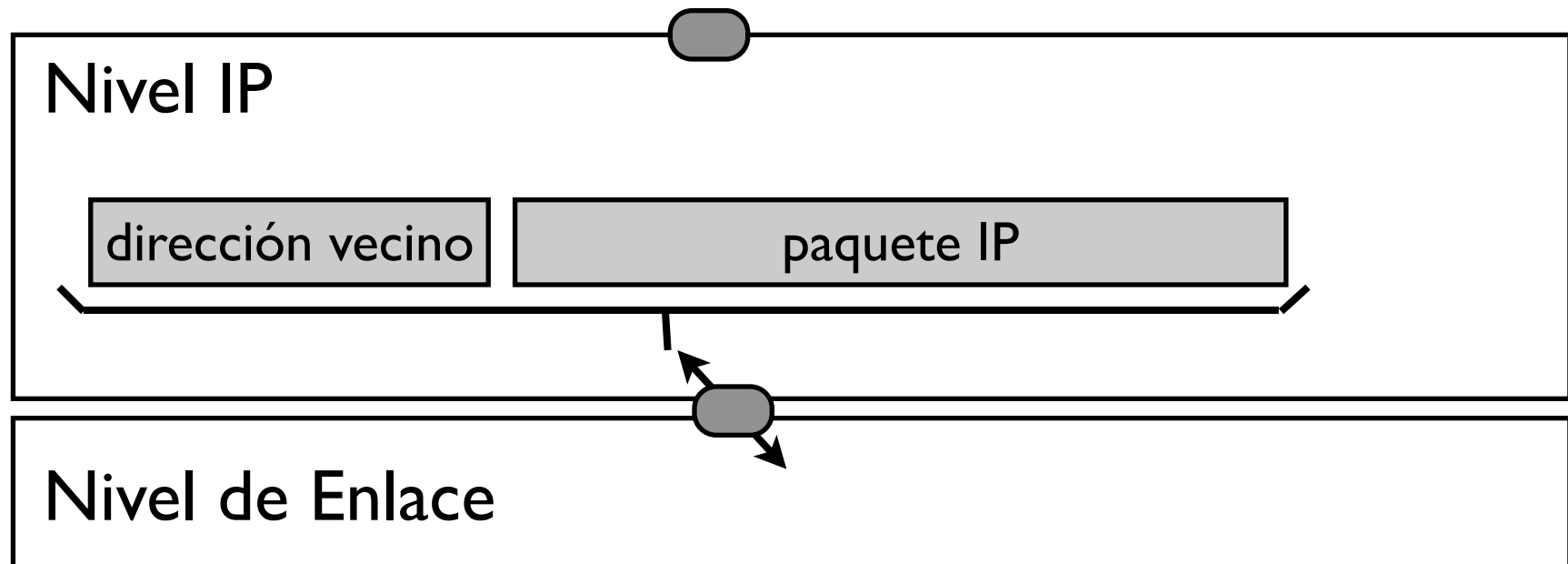
Interfaz con el nivel de transporte

- ▶ Envía este segmento a esta dirección IP
 - > El puerto no existe a nivel IP
 - > El tamaño máximo de segmento es: 65535-20
- ▶ Llega este segmento desde esta IP



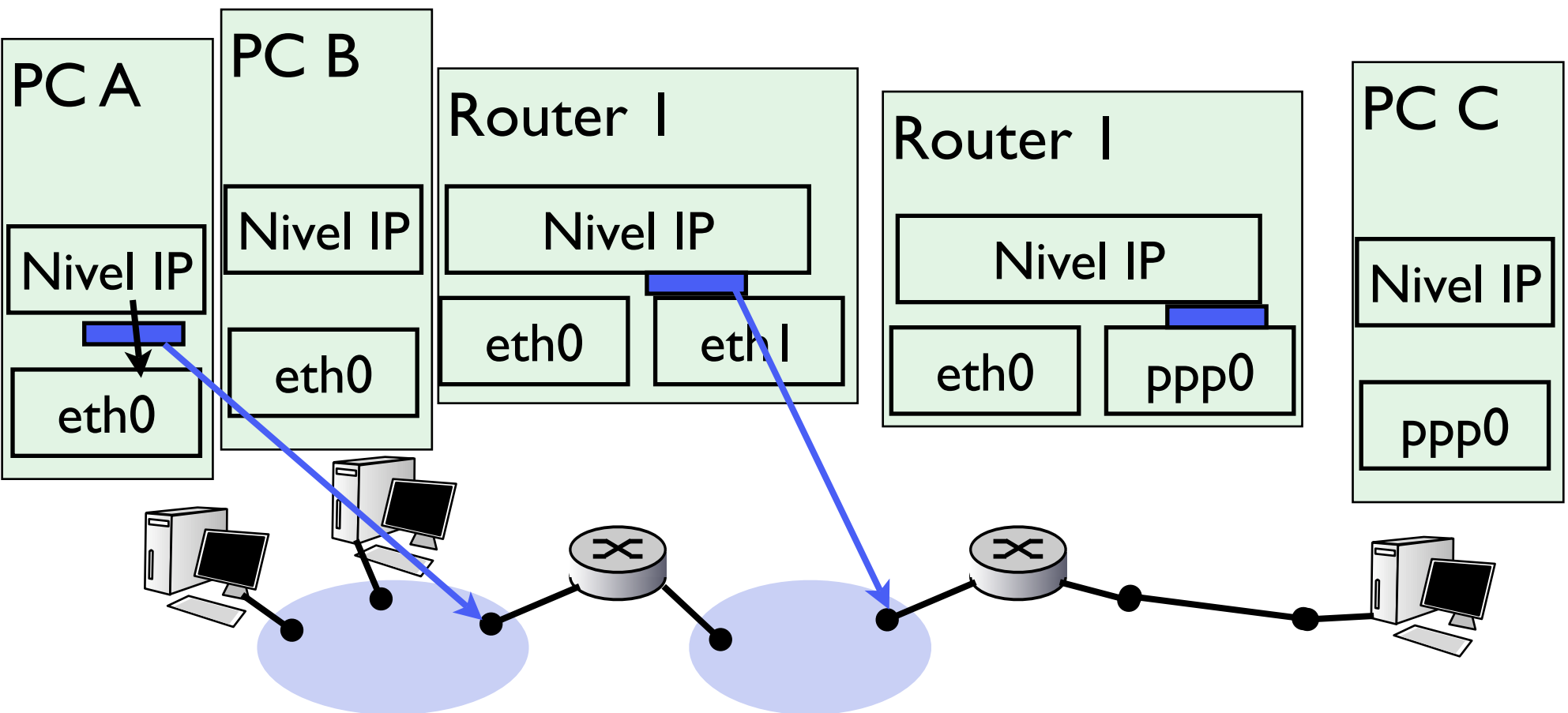
Interfaz con el nivel de enlace

- ▶ Envía estos datos (paquete IP) a este vecino
- ▶ Llega este paquete IP de este vecino
- ▶ Como identificamos a los vecinos?
- ▶ Que pasa si el ordenador tiene varios interfaces?



Interfaz con el nivel de enlace

- ▶ La dirección de un vecino es:
{interfaz local, dirección del vecino en la red de area local}
- ▶ En algunos tipos de enlaces el interfaz es suficiente



Interfaz con el nivel de enlace

- ▶ Los niveles de enlace de redes de área local (por ejemplo Ethernet) utilizan diferentes tipos de direcciones para identificar a los ordenadores en una red de área local
- ▶ Estas direcciones las llamaremos direcciones de nivel de enlace o direcciones Ethernet (también direcciones físicas o direcciones MAC)

Las veremos en el tema de nivel de enlace

- ▶ El sistema operativo proporciona a IP mecanismos para obtener la dirección de nivel de enlace de los vecinos a partir de la dirección IP.
 - > Se mantiene una tabla de direcciones de los vecinos conocidos en cada interfaz
 - > Cuando un vecino que necesitamos no está en la tabla se envían mensajes para localizarlo (Vease ARP en el siguiente tema)

Nivel IP

▶ Entradas

- > Llega un paquete procedente del nivel de enlace (independientemente de cual)
- > Llega un paquete a enviar procedente del nivel de transporte
- > En ambos casos IP hace el mismo proceso

▶ Resultados

- > Entregar un paquete al nivel superior
- > Enviar un paquete a través de uno de los interfaces

Recibiendo un paquete IP

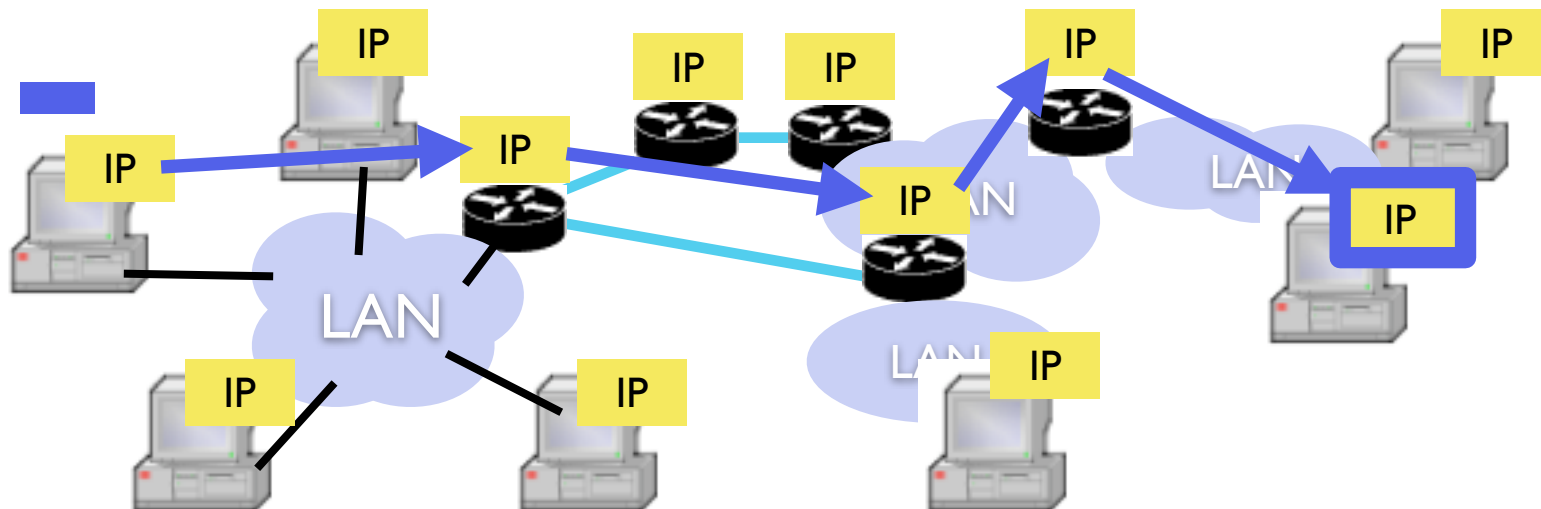
- ▶ El nivel IP recibe un paquete proveniente del nivel inferior
 - > **Comprobar que la cabecera es correcta con el checksum**
 - + Error: descartar el paquete
 - > **Extraer la dirección IP destino**
 - > **Es un paquete dirigido a este host? (la dirección IP es la mía? o una de las mías?)**
 - + SI: entregar al nivel superior
 - + NO: pasa a la siguiente pregunta...
 - > **Soy un router? (IP_FORWARDING activado?)**
 - + SI: pasamos al algoritmo de envío
 - + NO: descartar el paquete

Enviando un paquete IP

- ▶ El nivel IP recibe un paquete proveniente del nivel de transporte
 - > **Extraer la dirección IP destino**
 - > **Es un paquete dirigido a este host? (la dirección IP es la mía? o una de las mías?)**
 - + SI: entregar al nivel superior
 - + NO: pasamos al algoritmo de envío

Envío de un paquete IP

- ▶ 2 envíos diferentes
 - > Saber que hacer con los destinos que no están en mi LAN
 - + Saber cual es el siguiente salto
 - + Enviar al siguiente salto
 - > Enviar a los destinos **que estén en mi LAN**



Enviando un paquete IP

- ▶ El nivel IP tiene un paquete a enviar
 - > Proveniente del nivel superior
 - > Es un paquete que me ha llegado y no es para mi (y soy un router)

- ▶ **Extraer dirección de destino**

Esta en alguna de las subredes a las que estoy conectado?

- > SI = el destino es un vecino
enviar paquete utilizando el nivel de enlace

- > NO = el destino no es un vecino

Determinar el **siguiente salto**

- + Tabla de encaminamiento con los siguientes saltos según prefijos

La tabla de rutas

Problemas

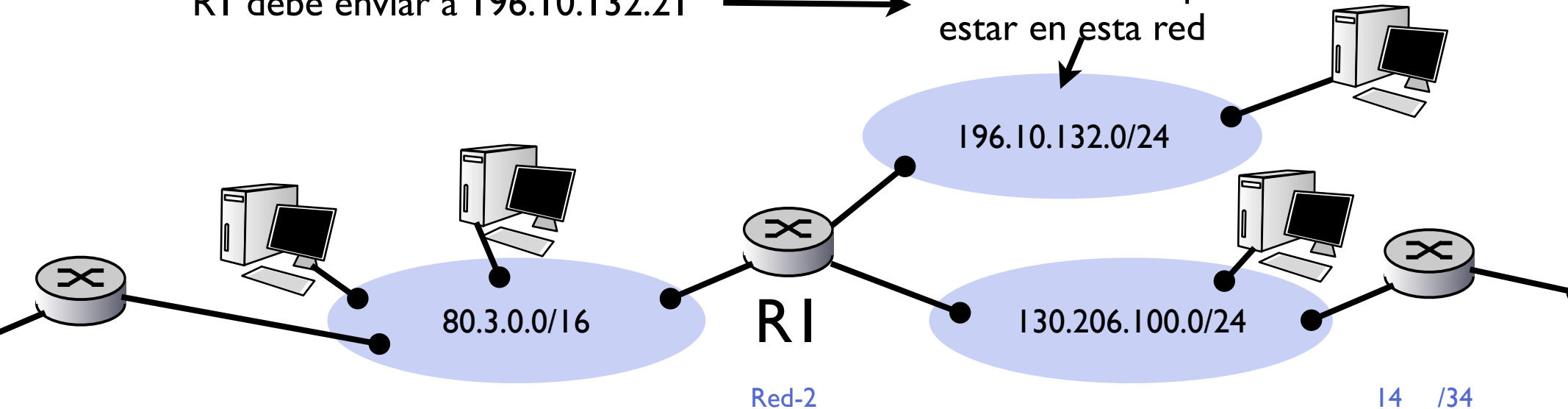
- ▶ **Cómo saber si el destino es un vecino o no de este router?**
 - > Para eso se organizan las redes en grupos de direcciones contiguas para decidir rápidamente si una dirección de destino está en una red
- ▶ **Cómo saber cual es el siguiente salto?**
 - > Para eso tenemos la tabla de rutas, nos dice como ir a cada posible dirección (o a cada posible prefijo)
- ▶ **Qué hacemos cuando sabemos el siguiente salto?**
 - > El siguiente salto si tiene que ser un vecino
 - > Se le envía al vecino el paquete IP, ese paquete IP va al vecino pero no tiene como destino la dirección IP del vecino así que el vecino lo reenviará

Problemas

- ▶ Cómo saber si el destino es un vecino o no de este router?
- ▶ Las direcciones en la red se usan en bloques con prefijo común
 - > Cada router sabe que bloques tienen las redes de area local a las que esta conectado
 - > Si un paquete va a uno de estos bloques hay que enviarlo en esas redes de area local
 - > Si no hay que enviarlo a otro router

RI debe enviar a 80.6.1.2 → Hay que pasarlo a otro router

RI debe enviar a 196.10.132.21 → El destino tiene que estar en esta red



Redes y máscaras

- ▶ Para almacenar las redes vecinas (y para las tablas de rutas) el nivel IP utiliza la máscara para extraer el prefijo de una IP destino
- > Almaceno por cada interfaz mi dirección IP y una máscara (al aplicar la mascara a la dirección IP obtengo el prefijo)

Una IP dentro de la red

11000100.00001010.10000100.00010101

196.10.132.21

16 bits de máscara

11111111.11111111.00000000.00000000

255.255.0.0

Obtenemos la red

11000100.00001010.00000000.00000000

196.10.0.0

Redes y máscaras

- ▶ En este caso el router RI tendrá las siguientes configuraciones en sus interfaces

eth0:

IP: 80.3.3.23
mascara: 255.255.0.0

eth1:

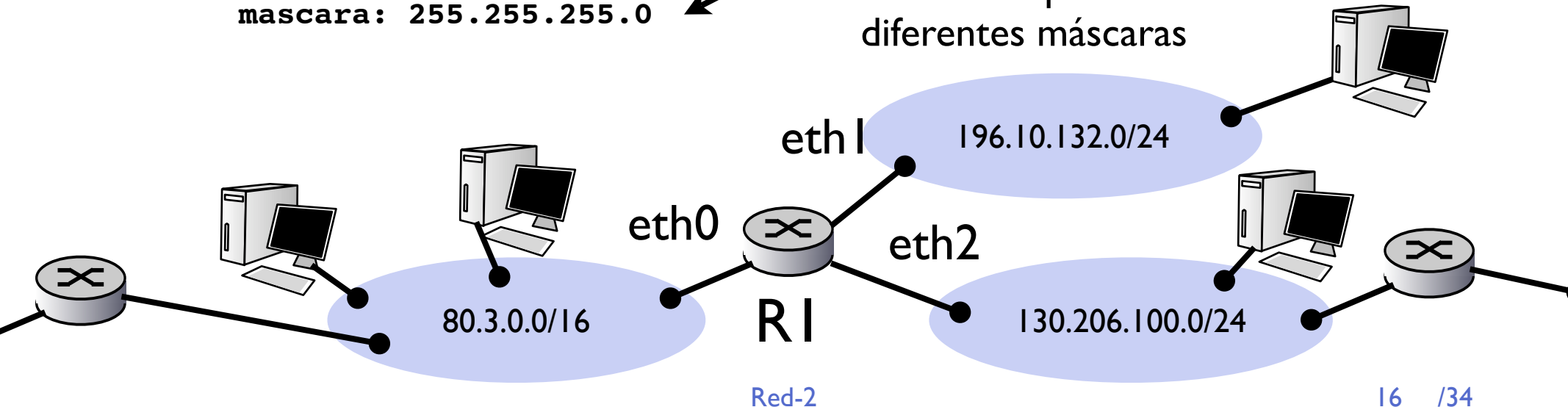
IP: 196.10.132.1
mascara: 255.255.255.0

eth2:

IP: 130.206.100.254
mascara: 255.255.255.0

No tiene por que ser la .1
No tiene por que ser la .254
Podría ser la .255 ?

**La máscara es del
interfaz no del router**
un router puede tener
diferentes máscaras



Configuración de IP

- ▶ Cada interfaz del nivel IP se configura con
 - > Una **dirección IP**
Indica que paquetes debo recibir y que IP origen debo enviar
 - > Una **máscara de red**
Indica quienes son mis vecinos
Que IPs hay en la misma red de área local
- ▶ Con esto es suficiente para comunicarse con la subred

Configurando IP

- ▶ Ejemplos configuración de IP
 - > En Linux

```
$ ifconfig eth0 10.1.1.14 netmask 255.255.0.0
$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0D:56:84:6D:A1
          inet addr:10.1.1.14  Bcast:10.1.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:16424 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12100 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5772581 (5.5 Mb)  TX bytes:3466780 (3.3 Mb)
          Base address:0xdf40 Memory:fcfe0000-fd000000
```

- > En CiscoIOS

```
# sh interface ethernet 0
Ethernet0 is up, line protocol is up
  Hardware is QUICC Ethernet, address is 0004.2721.e196 (bia 0004.2721.e196)
  Internet address is 10.1.1.247/24
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec, rely 255/255, load 1/255
  [...]
```

La tabla de rutas

- ▶ Para enviar a otras subredes el nivel IP mantiene una tabla con los caminos a otras redes organizadas por prefijos
 - > Los prefijos se indican/configuran con prefijo y mascara
 - > o con prefijo y numero de bits de la máscara
- ▶ Para cada prefijo {dirección y máscara} se almacena el *siguiente salto*, la dirección IP de un router que reenviará el paquete
- ▶ Ejemplo:

```
----- prefijo -----      --- sig salto ---
red          mascara
130.206.20.0 255.255.255.0 -> router1
80.10.0.0    255.255.0.0   --> router2
10.0.0.0     255.0.0.0    ----> router1
los demas    -----> router2
```

Más ejemplo

- ▶ La tabla almacena una serie de prefijos que se almacenen o se configuren con mascararas o con bits (es indiferente y depende solo de interfaz de usuario)
- ▶ Las dos tablas de rutas dicen lo mismo:

```
----- prefijo -----      --- sig salto ---  
red          mascara  
130.206.20.0 255.255.255.0 -> router1  
80.10.0.0    255.255.0.0   --> router2  
10.0.0.0     255.0.0.0    ----> router1  
default      -----> router2
```

```
---- prefijo ----      --- sig salto ---  
130.206.20.0 / 24 -> router1  
80.10.0.0 / 16  ----> router2  
10.0.0.0 / 8   ----> router1  
default        -----> router2
```

Y ¿qué son los siguientes saltos?

El prefijo por defecto es el prefijo que todas las IPs cumplen o sea 0.0.0.0 máscara 0.0.0.0 o bien 0.0.0.0/0

Tabla de rutas

- ▶ Pongamos que la tabla de rutas de R1 sea esta

```
----- prefijo -----      --- sig salto ---  
red          mascara  
130.206.20.0 255.255.255.0 --> 130.206.100.8  
80.10.0.0    255.255.0.0   --> 80.3.9.21  
10.0.0.0     255.0.0.0     -----> 130.206.100.8  
default      -----> 80.3.9.21
```

- ▶ Los siguientes saltos tienen que ser direcciones IP vecinas !!
Para que podamos entregarles un paquete a nivel de enlace

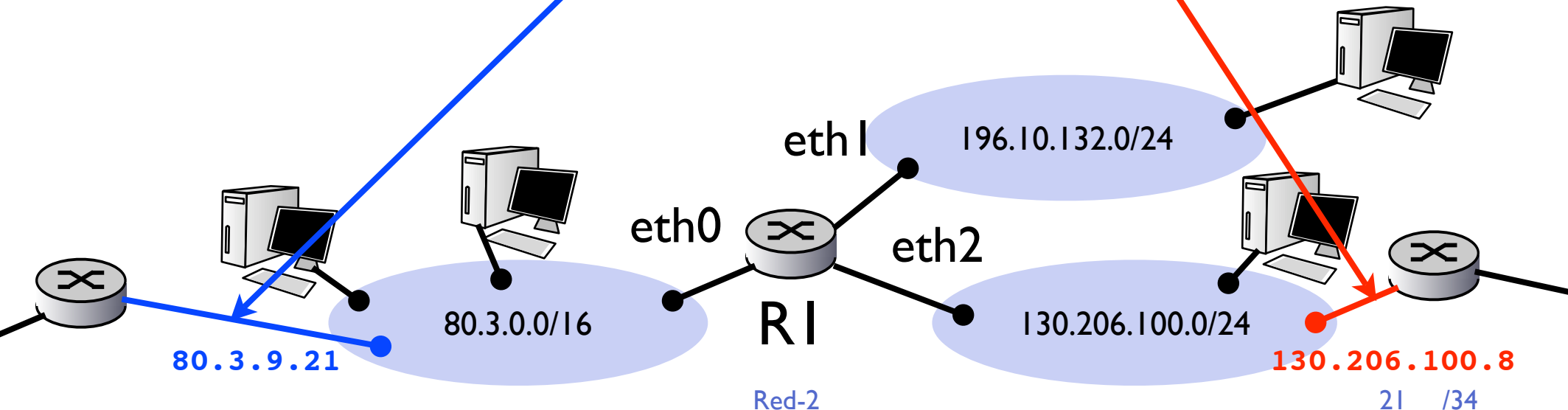


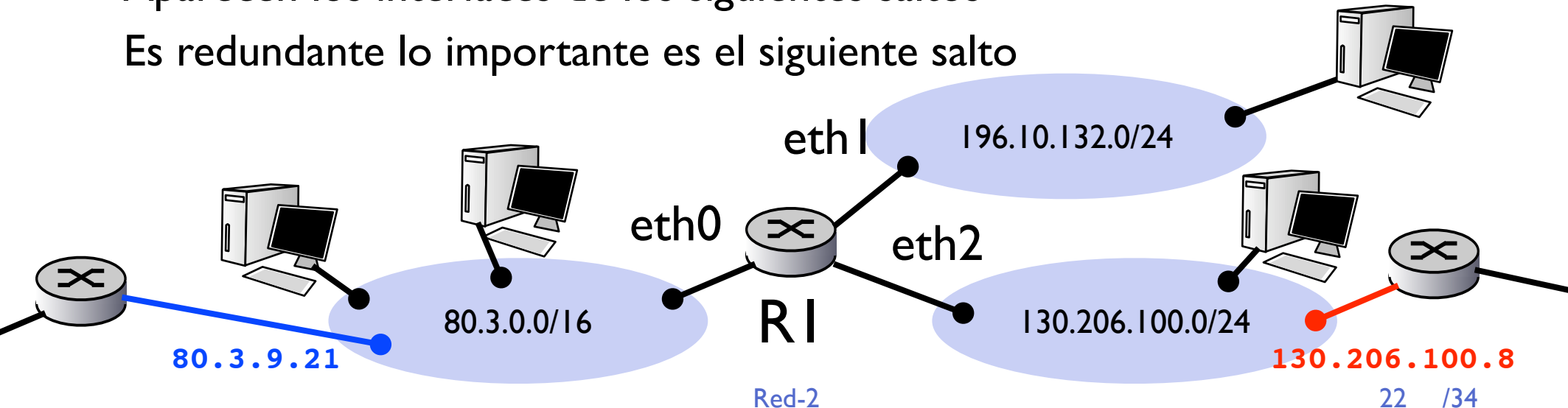
Tabla de rutas

- ▶ En implementaciones reales sale más información

----- prefijo -----		--- sig salto ---	-- interfaz --
red	mascara		
130.206.100.0	255.255.255.0	conectado	eth2
196.10.132.0	255.255.255.0	conectado	eth1
80.3.0.0	255.255.0.0	conectado	eth0
130.206.20.0	255.255.255.0	-> 130.206.100.8	eth2
80.10.0.0	255.255.0.0	--> 80.3.9.21	eth0
10.0.0.0	255.0.0.0	----> 130.206.100.8	eth2
0.0.0.0	0.0.0.0	-----> 80.3.9.21	eth0

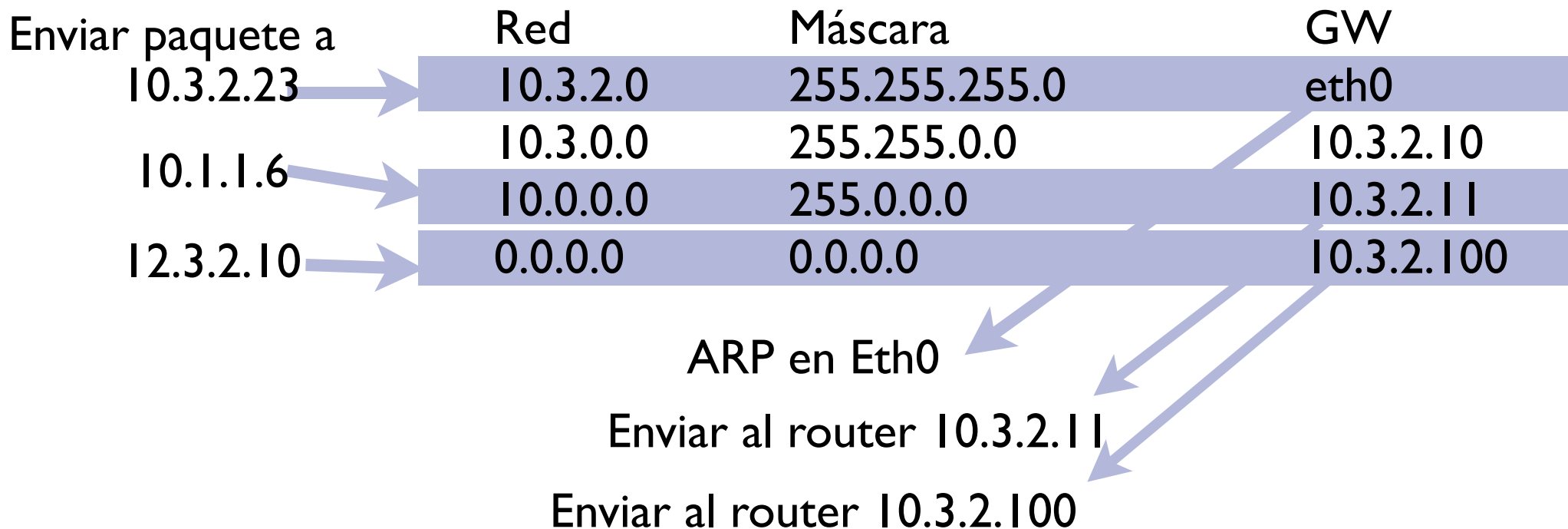
- ▶ Aparecen las IPs conectadas a los interfaces
- ▶ Aparecen los interfaces de los siguientes saltos

Es redundante lo importante es el siguiente salto



Enviando un paquete IP

- ▶ Se busca la primera entrada de la tabla que coincida.
Empezando por las máscaras más restrictiva (según la implementación la tabla está almacenada en orden más restrictivo o bien se busca en ese orden)
- ▶ Si coincide obtenemos el interfaz o el siguiente router



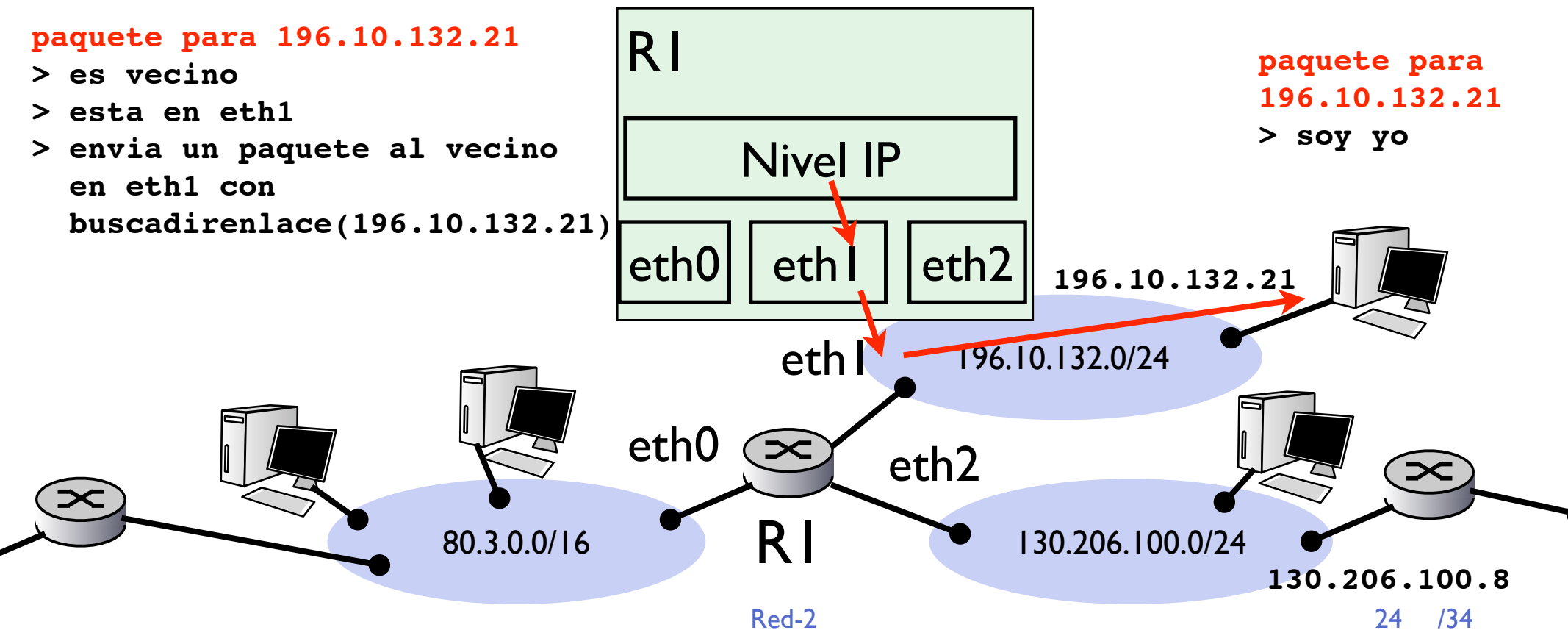
Enviando al siguiente salto

- ▶ Si el destino es vecino
 - > Se le da al nivel de enlace el paquete IP pidiéndole que lo transmita a ese vecino en el interfaz correspondiente
 - > El vecino recibe un paquete que va para su IP y lo entrega al nivel superior

paquete para 196.10.132.21

- > es vecino
- > esta en eth1
- > envia un paquete al vecino en eth1 con `buscadirenlace(196.10.132.21)`

paquete para 196.10.132.21
> soy yo

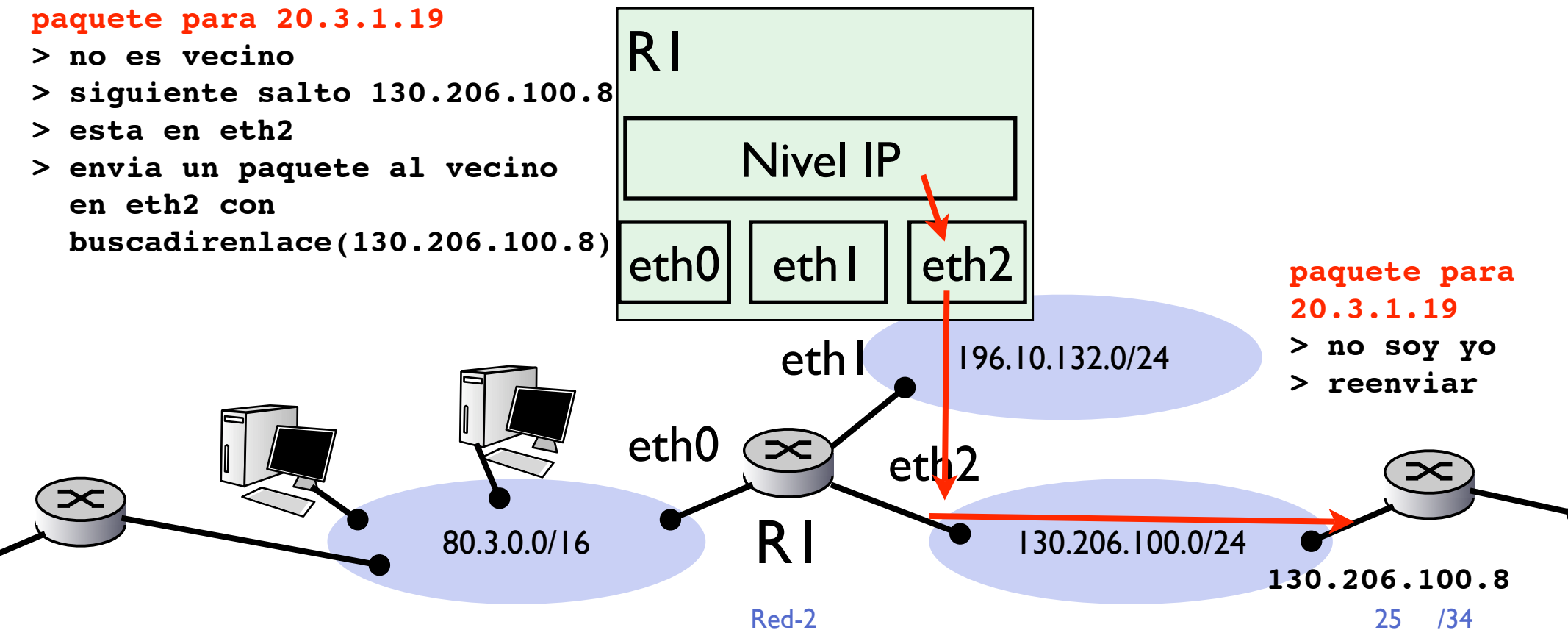


Enviando al siguiente salto

- ▶ Si el destino no es vecino
 - > Se busca el siguiente salto en la tabla de rutas
 - > Se le da al nivel de enlace el paquete IP pidiéndole que lo transmita al siguiente salto ese vecino en el interfaz correspondiente
 - > El vecino recibe un paquete que no va para su IP y lo reenvía

paquete para 20.3.1.19

- > no es vecino
- > siguiente salto 130.206.100.8
- > esta en eth2
- > envia un paquete al vecino en eth2 con `buscadirenlace(130.206.100.8)`



paquete para 20.3.1.19

- > no soy yo
- > reenviar

La tabla de rutas

- ▶ Ejemplos consultando la tabla de rutas
 - > En Linux

```
$ route -n
10.1.0.0      0.0.0.0      255.255.0.0   U    0    0    0 eth0
169.254.0.0  0.0.0.0      255.255.0.0   U    0    0    0 eth0
127.0.0.0    0.0.0.0      255.0.0.0     U    0    0    0 lo
0.0.0.0      10.1.1.1     0.0.0.0       UG   0    0    0 eth0
```

- > En CiscoIOS

```
> show ip route
Gateway of last resort is 10.6.1.248 to network 0.0.0.0

    10.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
C       10.6.4.0/24 is directly connected, Vlan8
S       10.3.0.0/16 [1/0] via 10.1.1.247
C       10.1.0.0/16 is directly connected, Vlan3
C       10.6.1.0/24 is directly connected, Vlan6
C       10.6.3.0/24 is directly connected, Vlan7
S*     0.0.0.0/0 [1/0] via 10.6.1.248
```

Construcción de la tabla de rutas

▶ **Enrutamiento estático**

= La configura el administrador

- > No se adapta a los cambios, no reacciona ante fallos de enlaces
- > Más fácil en redes simples. Hay pocas redes y siempre está clara la ruta por defecto hacia afuera

▶ **Enrutamiento dinámico**

= Protocolos de enrutamiento, los routers hablan con sus vecinos y se ponen de acuerdo en las tablas de rutas (Algoritmos de Dijkstra y Bellman-Ford distribuidos)

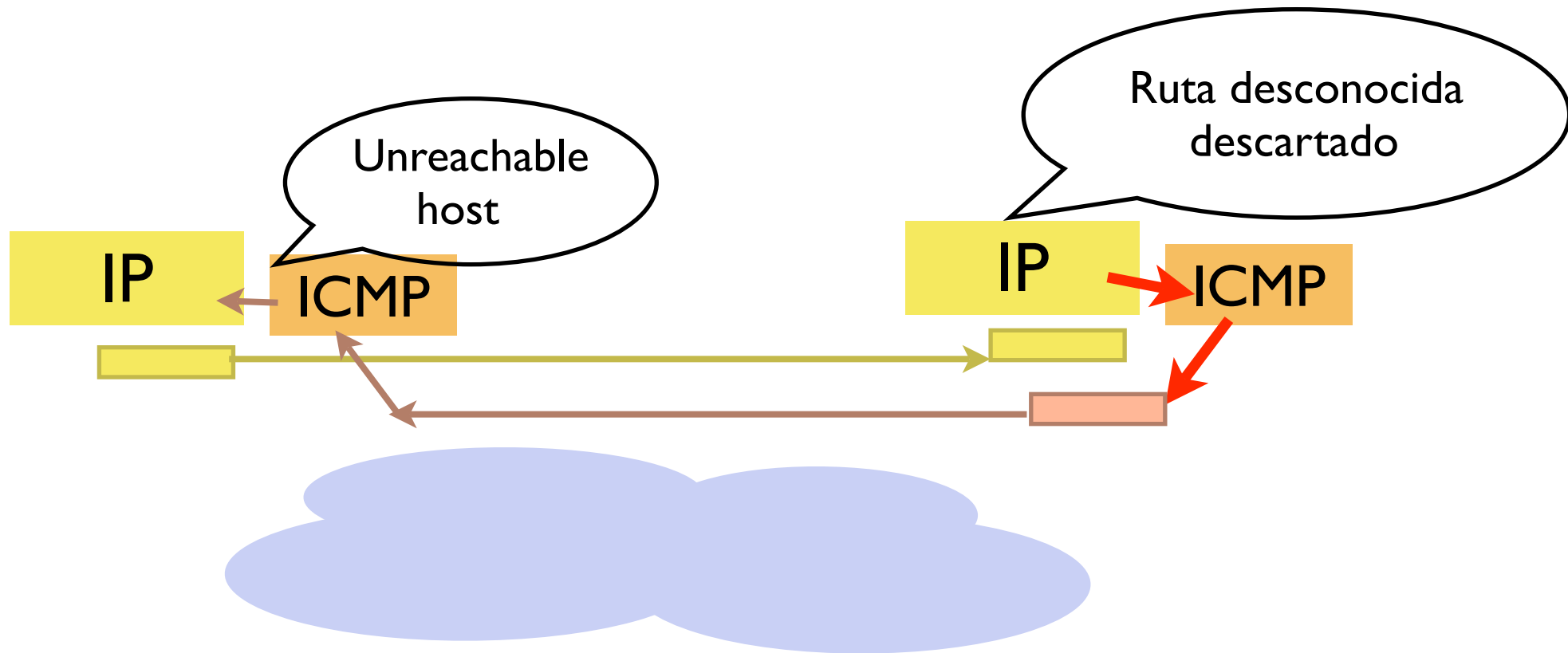
- > Se adapta a los cambios
- > Es mas cómodo cuando todo funciona
- > Problemas de algoritmos distribuidos... puede tardar en estabilizarse la solución, pueden generarse ciclos de enrutamiento...
- > Interesante pero no hay tiempo para ver esto
(si tiene interés vea Redes en segundo ciclo de Ingeniería Informática)

Protocolos de soporte a IP

- ▶ Nivel de red en Internet formado por
 - > Protocolo IP + reglas de reenvío (ok)
 - > Tabla de rutas (ok)
 - > Protocolos de encaminamiento dinámico (ok)
 - > Protocolos ICMP

ICMP

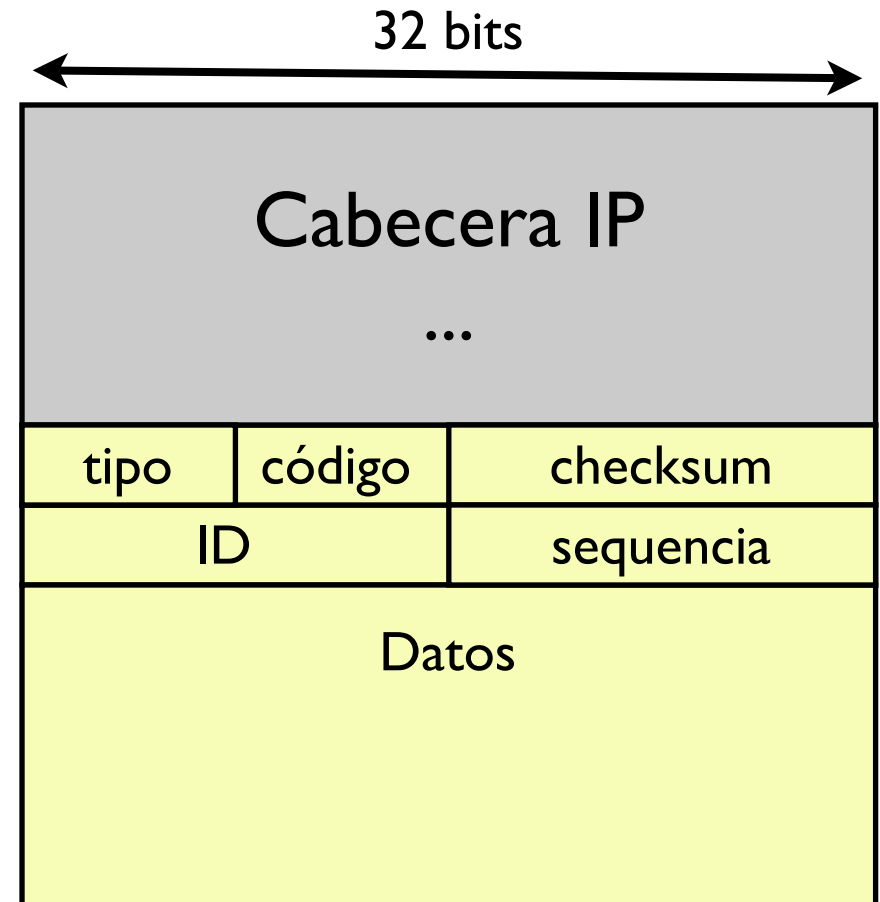
- ▶ Internet Control Message Protocol (RFC 792)
Protocolo para comunicacion de control entre niveles IP



Formato de paquete ICMP

▶ Protocolo sobre IP

- > tipo y código del mensaje
- > id y secuencia para identificarlo
- > checksum de la cabecera



ICMP: tipos y códigos de los mensajes

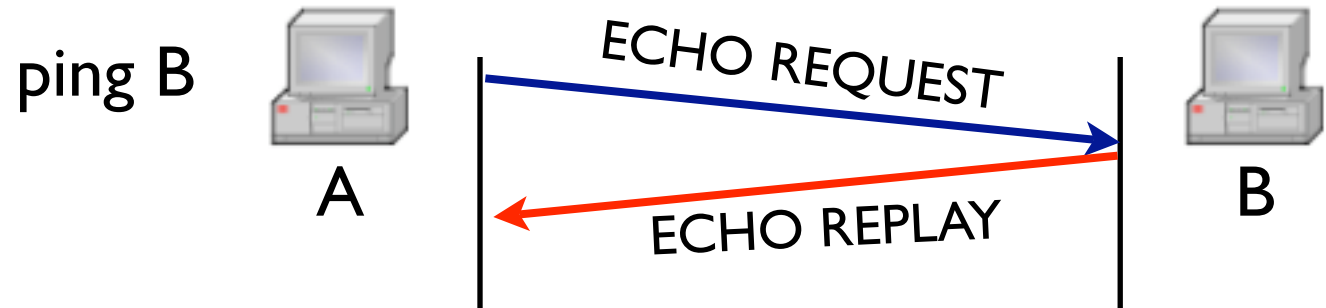
<u>tipo</u>	<u>código</u>	<u>description</u>	
0	0	echo reply (ping)	
3	0	dest. network unreachable	Informar de errores
3	1	dest host unreachable	
3	2	dest protocol unreachable	
3	3	dest port unreachable	
3	6	dest network unknown	
3	7	dest host unknown	
4	0	source quench (congestion control - not used)	
8	0	echo request (ping)	
9	0	route advertisement	control routers
10	0	router discovery	
11	0	TTL expired	Informar de errores
12	0	bad IP header	

+ ECHO REQUEST/REPLAY

Ping

▶ Mediante ICMP se ofrece el servicio de Ping

- > Si un nivel ICMP recibe un paquete ECHO REQUEST
- > Responde con un paquete ECHO REPLY al origen
- > Útil para gestión y mantenimiento de la red, permite averiguar si hay un host en una dirección IP dada (y cuanto retardo hay hasta ella)



```
$ ping www.google.com
```

```
PING www.l.google.com (216.239.59.104): 56 data bytes
```

```
64 bytes from 216.239.59.104: icmp_seq=0 ttl=242 time=196.928 ms
```

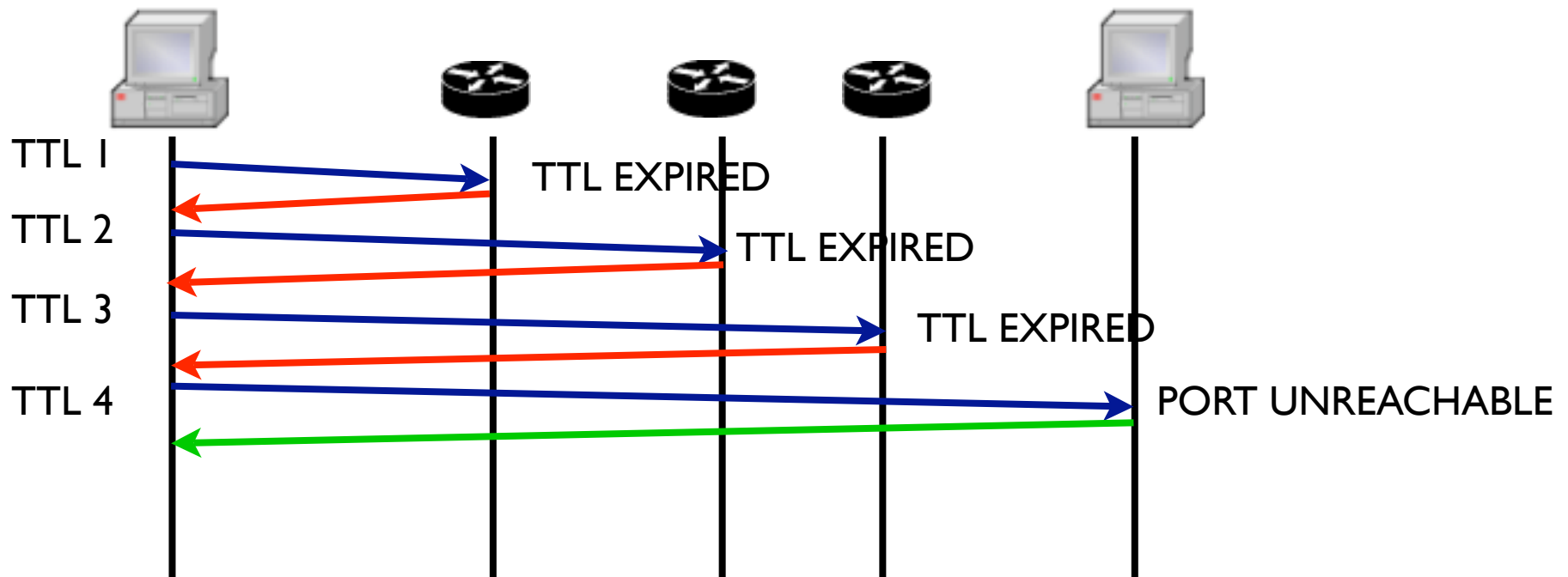
```
64 bytes from 216.239.59.104: icmp_seq=1 ttl=242 time=197.240 ms
```

```
64 1124636694.904248 IP 192.168.1.33 > 216.239.59.104: icmp 64: echo request seq 0
64 1124636695.100972 IP 216.239.59.104 > 192.168.1.33: icmp 64: echo reply seq 0
1124636695.904467 IP 192.168.1.33 > 216.239.59.104: icmp 64: echo request seq 1
1124636696.101508 IP 216.239.59.104 > 192.168.1.33: icmp 64: echo reply seq 1
1124636696.904678 IP 192.168.1.33 > 216.239.59.104: icmp 64: echo request seq 2
1124636697.101080 IP 216.239.59.104 > 192.168.1.33: icmp 64: echo reply seq 2
```

```
...
```


Traceroute

- ▶ Sistema parecido al ping pero averigua todos los saltos
 - > Usa UDP sobre IP (podría también usar ICMP ECHO)
 - > Envía paquetes al destino con TTL incrementándose
 - > El router que tira el paquete envía un ICMP TTL EXPIRED



Conclusiones

- ▶ El protocolo IP reenvía los paquetes a los routers vecinos o al destino final
 - > Para ello debe saber quien esta en su red de area local
 - > Necesita configuración una dirección, una máscara
 - > y la **tabla de rutas**
- ▶ ICMP permite enviar errores y mensajes de control entre niveles IP
 - > va sobre paquetes IP
 - > permite algunas utilidades: ping, traceroute

Próxima clase:

- ▶ Nivel de enlace