Fragmentación y Reensamblado en IP ICMP

Area de Ingeniería Telemática
http://www.tlm.unavarra.es

Arquitectura de Redes, Sistemas y Servicios
3º Ingeniería de Telecomunicación

Temario

1. Introducción
2. Arquitecturas, protocolos y estándares
3. Conmutación de paquetes
4. Conmutación de circuitos
5. Tecnologías
6. Control de acceso al medio en redes de área local
7. Servicios de Internet

Temario

1. Introducción
2. Arquitecturas, protocolos y estándares
3. **Conmutación de paquetes**
   - Arquitectura de protocolos para LANs
   - Ethernet
   - Protocolos de Internet
     - Internetworking
     - Direccionamiento
     - **Fragmentación e ICMP.** IP en LAN
4. Conmutación de circuitos
5. Tecnologías
6. Control de acceso al medio en redes de área local
7. Servicios de Internet

Objetivo

- Completar los conceptos básicos sobre el nivel de red en Internet

Contenido

- Fragmentación y reensamblado
  - Necesidad
  - Implementación
  - Problemas
- ICMP
  - Características generales
  - Condiciones generales de envío
  - Mensajes
- Traceroute

Contenido

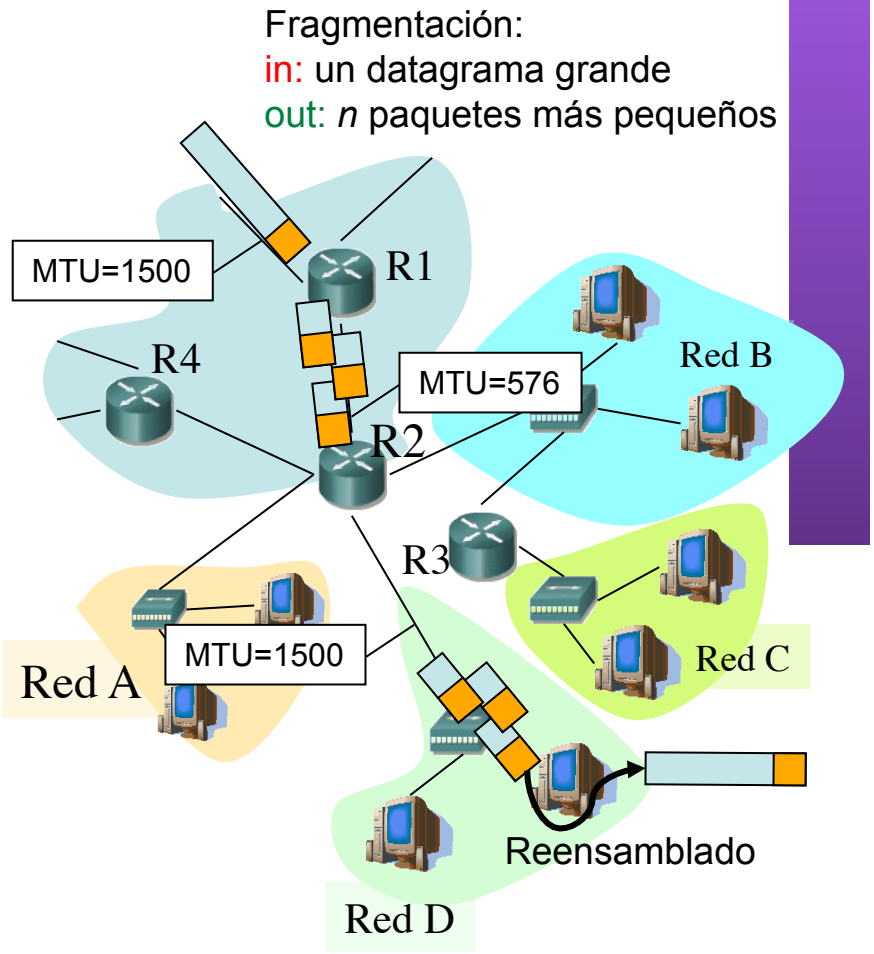
- **Fragmentación y reensamblado**
  - **Necesidad**
  - **Implementación**
  - **Problemas**
- ICMP
  - Características generales
  - Condiciones generales de envío
  - Mensajes
- Traceroute

Fragmentación y Reensamblado

Necesidad

- El nivel de enlace impone unos límites al tamaño
- MTU = Maximum Transfer Unit
- Un datagrama IP es dividido dentro de la red (…)
- Un datagrama se convierte en varios paquetes
- Hosts y routers fragmentan
- *Los routers NO reensamblan (…)*
- Solo el host receptor final reensambla (…)

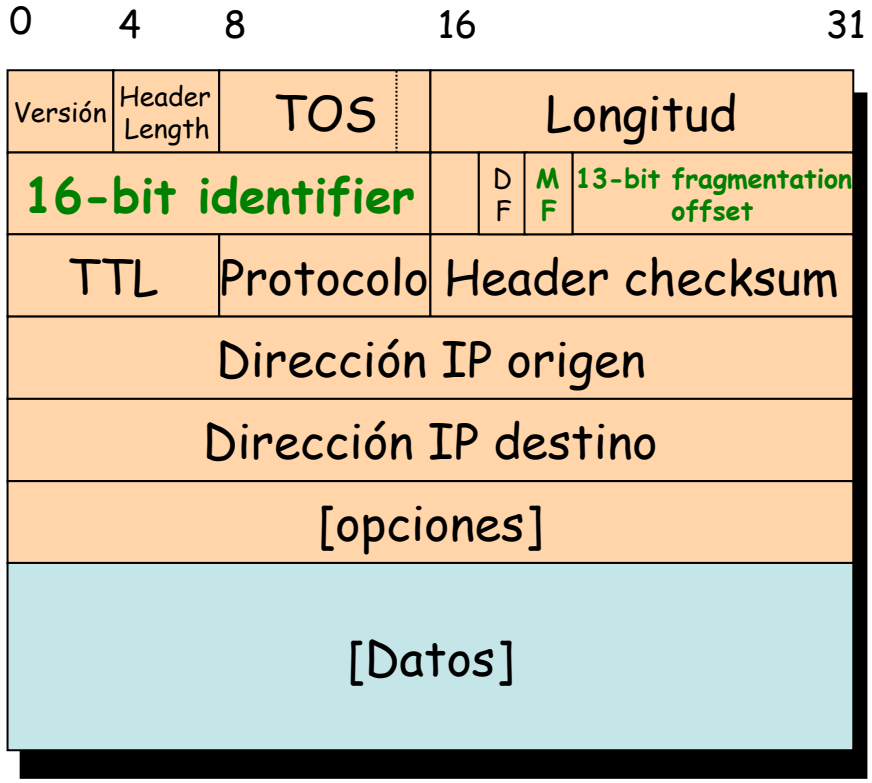
Red (RFC 1191)	**MTU**
16Mbps Token Ring	17914
IEEE 802.4	8166
FDDI	4352
Ethernet	1500
IEEE 802.3	1492
X.25	576



Fragmentación y Reensamblado

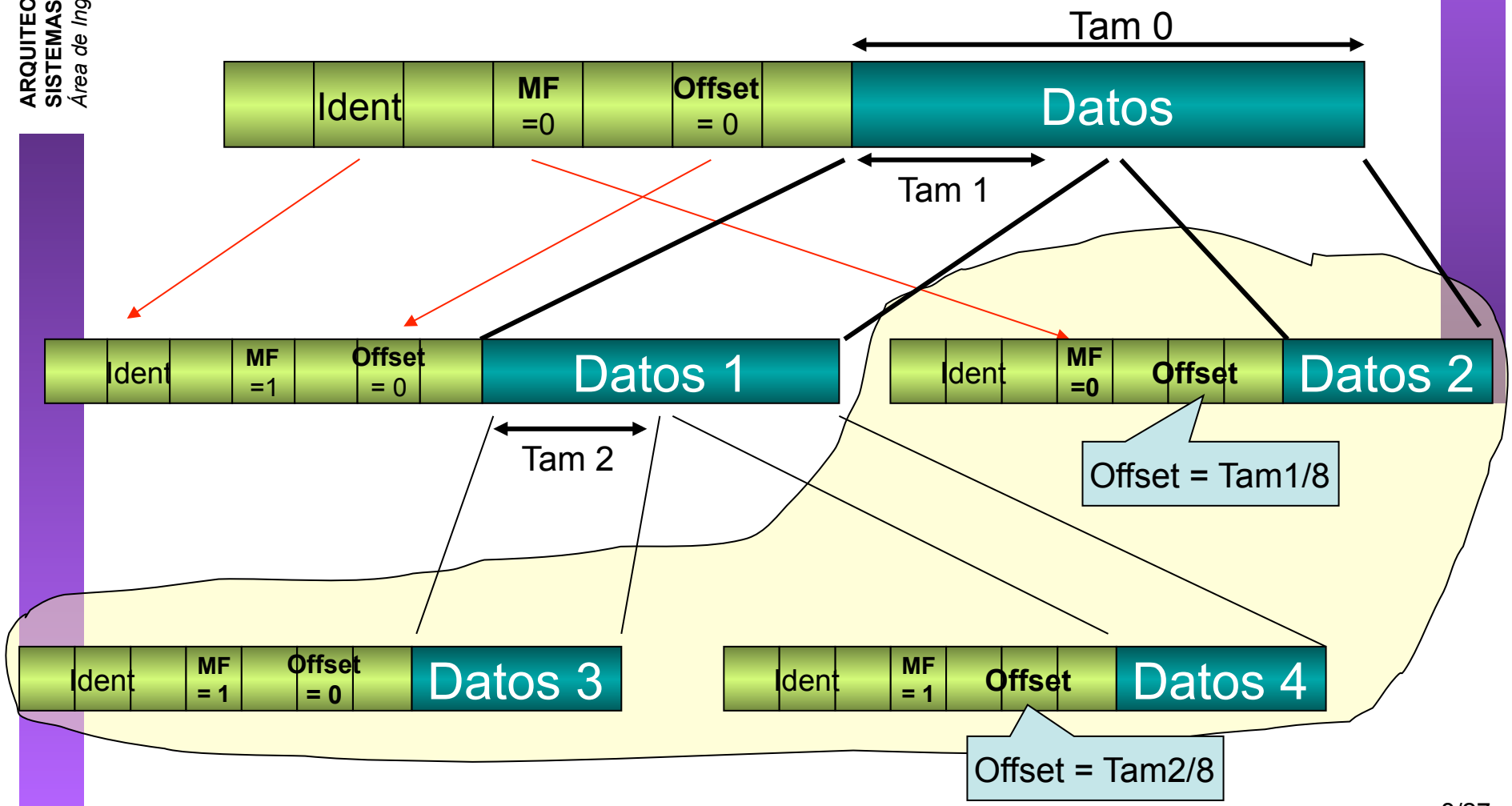
Codificación de la información

- Campos empleados:
  - Identificación
  - Bit MF
  - Fragment offset
- Fragmentos del datagrama:
  - Igual identificación, IP origen, IP destino y protocolo
- "*Longitud*" es la del paquete, no del datagrama
- Ante un primer fragmento ⇒ reservar zona de memoria donde reensamblar
- Debe reservar suficiente para reensamblar al menos datagramas de 576 Bytes



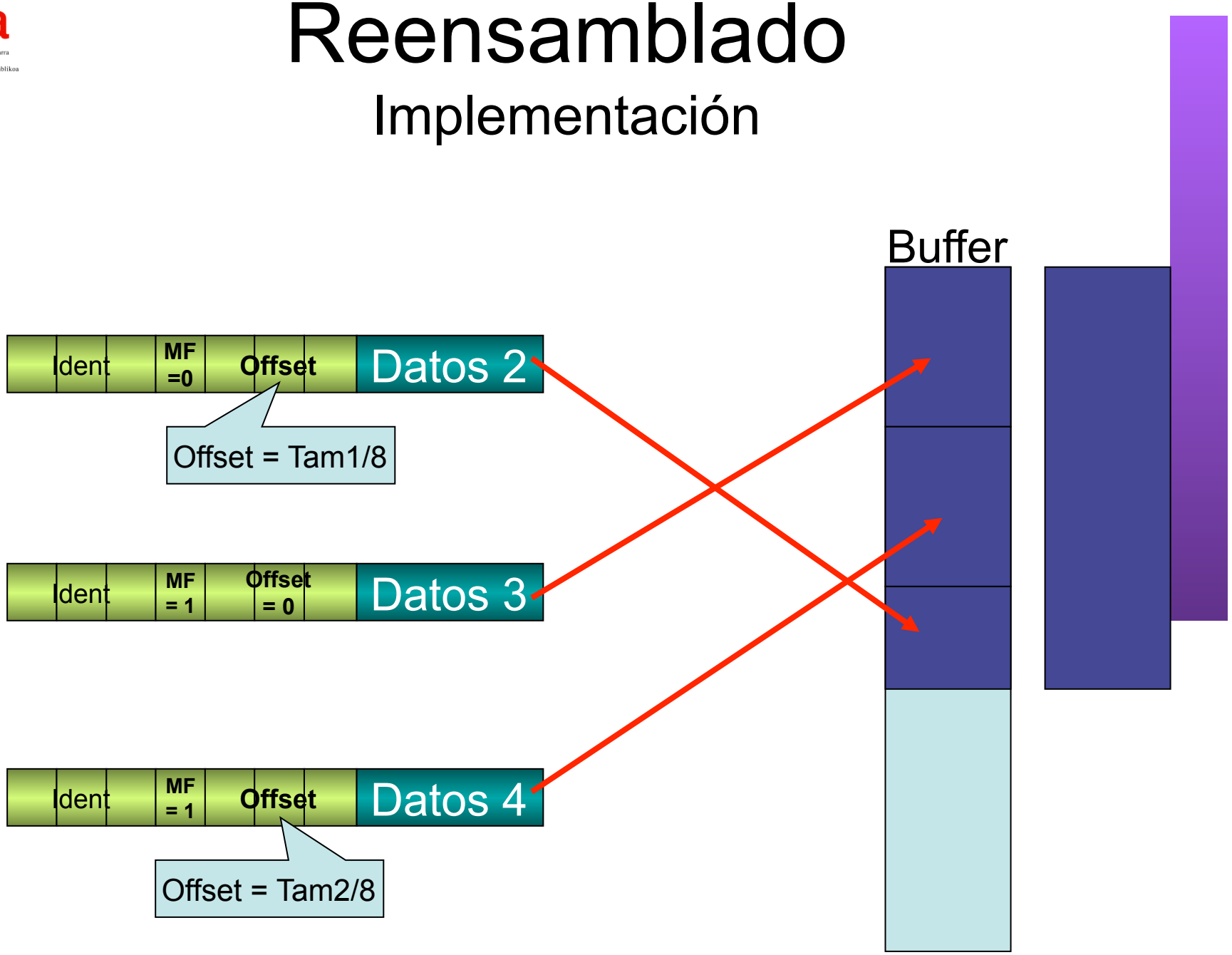
Fragmentación

Implementación



Reensamblado

Implementación



Situaciones de "error"

- Bit DF:
  - En la cabecera IP
  - DF==1 ⇒ routers no pueden fragmentar el paquete
  - (Tam>MTU)&&(DF==1) ⇒ lo descarta y devuelve al host origen un paquete indicando el error (ICMP)
- Reensamblado:
  - Inicia un *timer* con el primer fragmento que recibe
  - Si caduca el *timer* sin tener todos los fragmentos descarta todo lo recibido y devuelve al origen un paquete indicando el error (ICMP)

Problemas de la fragmentación

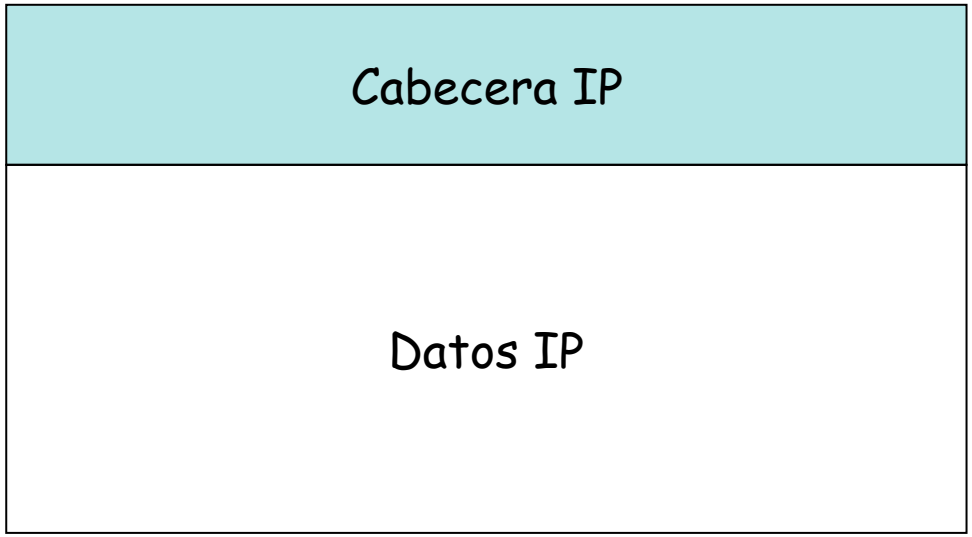
- Menor cociente Datos/Cabeceras
- Añade más carga a los routers (IPv6 la elimina)
- Si se pierde un fragmento:
  - El receptor no puede recomponer el datagrama
  - Tira todos los fragmentos recibidos
- Hasta que no se reciba todo el datagrama no se pueden pasar los datos al nivel de transporte (mayor retardo)

Contenido

- Fragmentación y reensamblado
  - Necesidad
  - Implementación
  - Problemas
- **ICMP**
  - **Características generales**
  - **Condiciones generales de envío**
  - **Mensajes**
- Traceroute

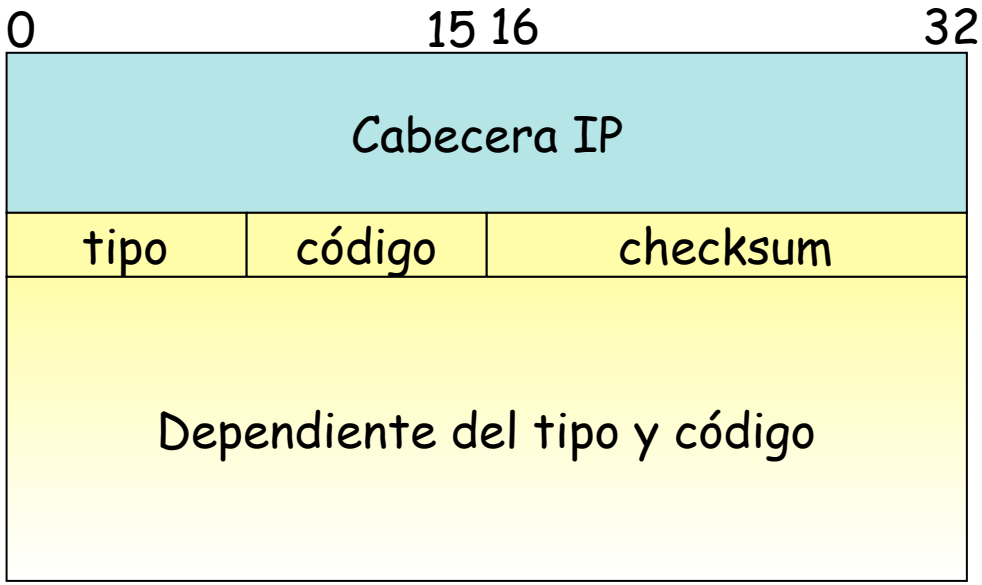
Características generales

- *Internet Control Message Protocol* (RFC 792)
- Para comunicar mensajes de error y otra información del nivel de red
- Mensajes transportados dentro de datagramas IP
- El destino es la dirección del paquete IP que generó el error
- Parte del nivel IP
- Estructura general del mensaje (…):

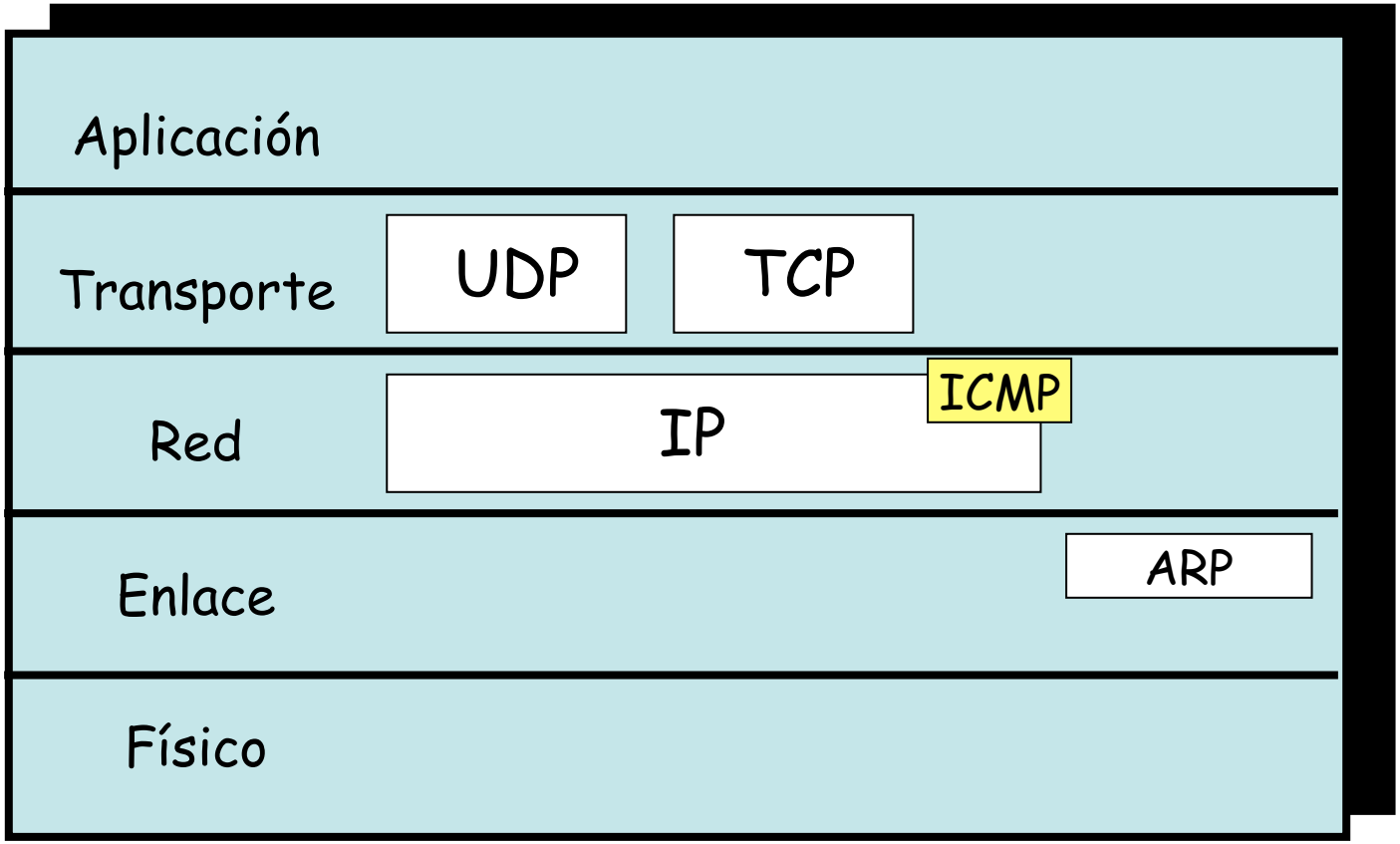


Características generales

- *Internet Control Message Protocol* (RFC 792)
- Para comunicar mensajes de error y otra información del nivel de red
- Mensajes transportados dentro de datagramas IP
- El destino es la dirección del paquete IP que generó el error
- Parte del nivel IP
- Estructura general del mensaje (…):



¿Dónde encaja ICMP en la pila TCP/IP?



Clases de mensajes ICMP

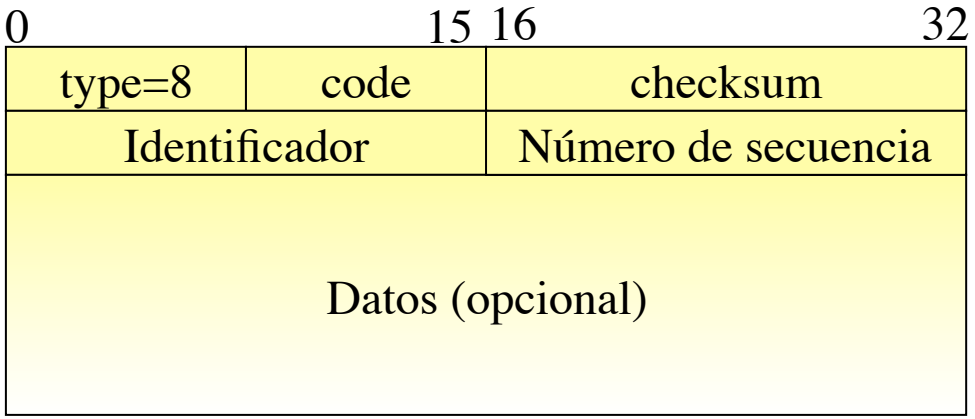
- Mensajes de Error:
  - Destino inalcanzable
  - *Redirect*
  - Tiempo excedido
  - *Source Quench*
  - Problema de parámetros
- Mensajes de pregunta (*query*):
  - Echo
  - *Router Advertisement*
  - *Timestamp*
  - Información
  - *Address Mask*

Condiciones generales de envío

- Para evitar tormentas de errores
- Nunca se generan ICMPs *de error* en respuesta a:
  - Un ICMP de error
  - Un datagrama destinado a una IP de broadcast o multicast
  - Un broadcast (o multicast) a nivel de enlace
  - Un fragmento que no sea el primero
  - Un datagrama cuya IP origen no sea *single-host*: *loopback, broadcast, multicast*

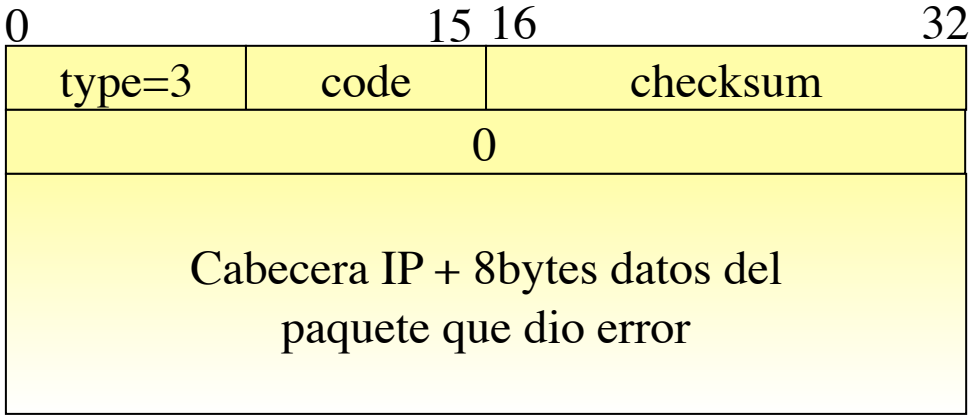
Mensajes ICMP

- Echo request/reply (*query*) (PING)
  - tipo = 8 (request) o 0 (reply), código = 0
  - Servidor debe hacer *echo* del paquete (incluidos los datos)
  - Obligatorio de implementar (generalmente en el kernel)



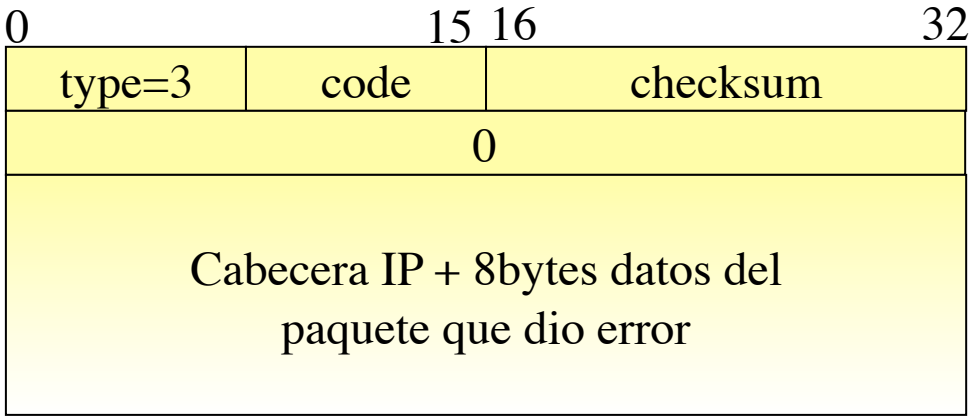
Mensajes ICMP

- Destino inalcanzable (*error*)
  - tipo = 3
  - Si según la tabla de rutas no se puede llegar al destino, host/router debe enviarlo (…)



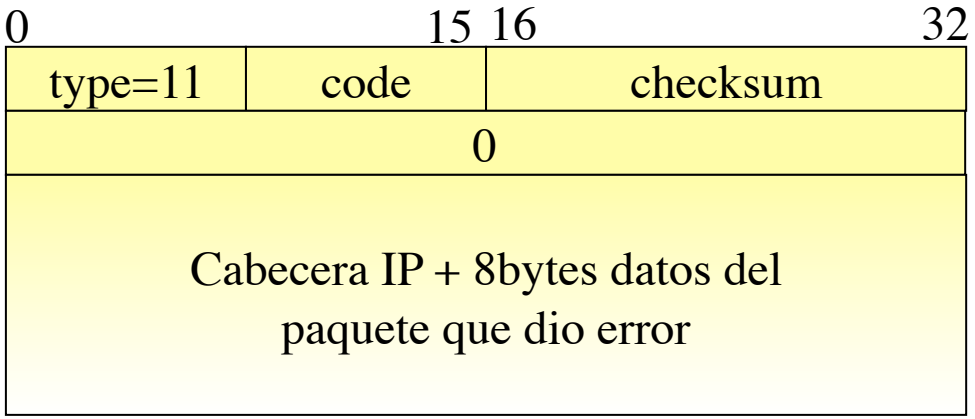
Mensajes ICMP
(Destino inalcanzable)

- Código:
  - 0 = Red destino inalcanzable
  - 1 = Host destino inalcanzable
  - 2 = Protocolo destino inalcanzable
  - 3 = Puerto destino inalcanzable
  - 4 = Fragmentación necesaria y DF activo
  - 5 = Source route failed



Mensajes ICMP

- Tiempo excedido (*error*)
  - tipo = 11
  - código = 0 (TTL=0 en tránsito), 1 (timeout durante reensamblado, necesita primer paquete)

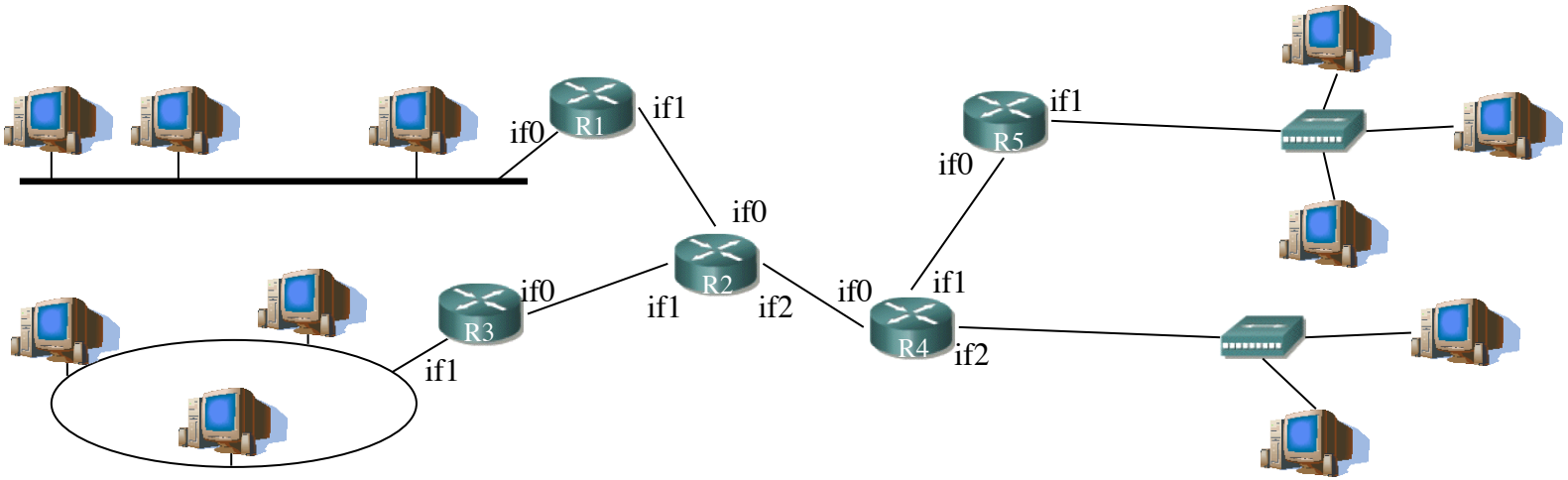


Contenido

- Fragmentación y reensamblado
  - Necesidad
  - Implementación
  - Problemas
- ICMP
  - Características generales
  - Condiciones generales de envío
  - Mensajes
- **Traceroute**

Traceroute

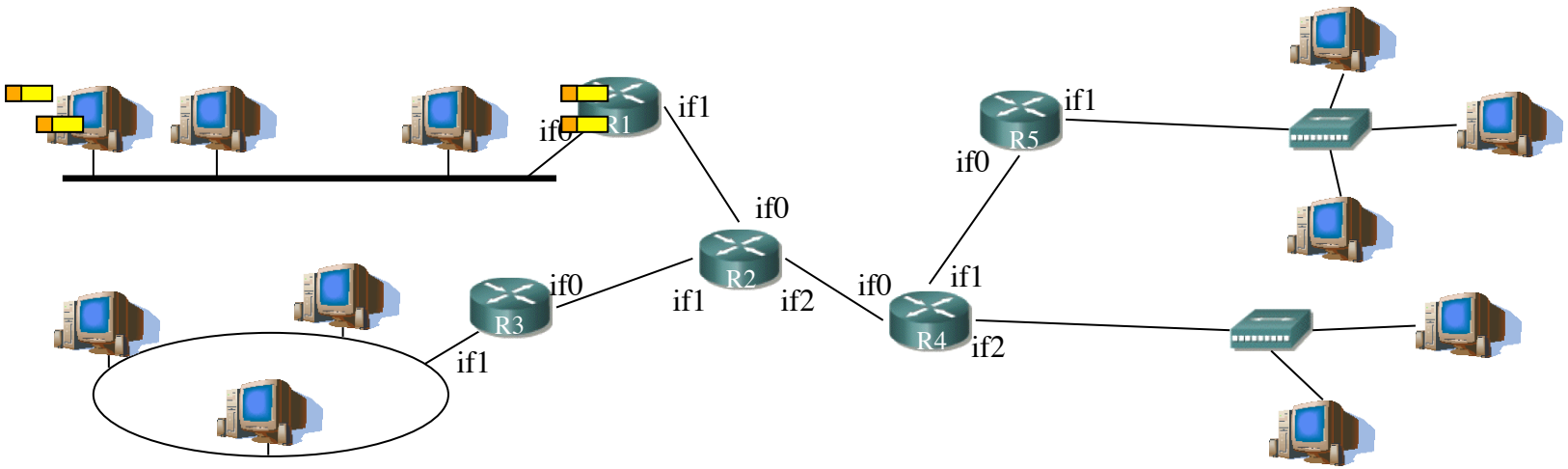
- Permite averiguar *el camino* entre dos hosts
- Suponiendo que el camino se mantiene entre diferentes paquetes
- Requiere que el destino final soporte UDP
- Requiere que se generen ciertos mensajes ICMP
- *Implemented by **Van Jacobson** from a suggestion by Steve Deering. Debugged by a cast of thousands with particularly cogent suggestions or fixes from C. Philip Wood, Tim Seaver, and Ken Adelman.*



Traceroute

- El host inicial envía un datagrama UDP (…)
  - Dirigido al host final
  - Con TTL = 1
- El primer router decrementa el TTL a 0 (…)
  - Tira el paquete
  - Devuelve al origen un ICMP de Error *Tiempo excedido en tránsito*
  - Este es un paquete IP con dirección origen la del interfaz de R1 en la red del host (… …)

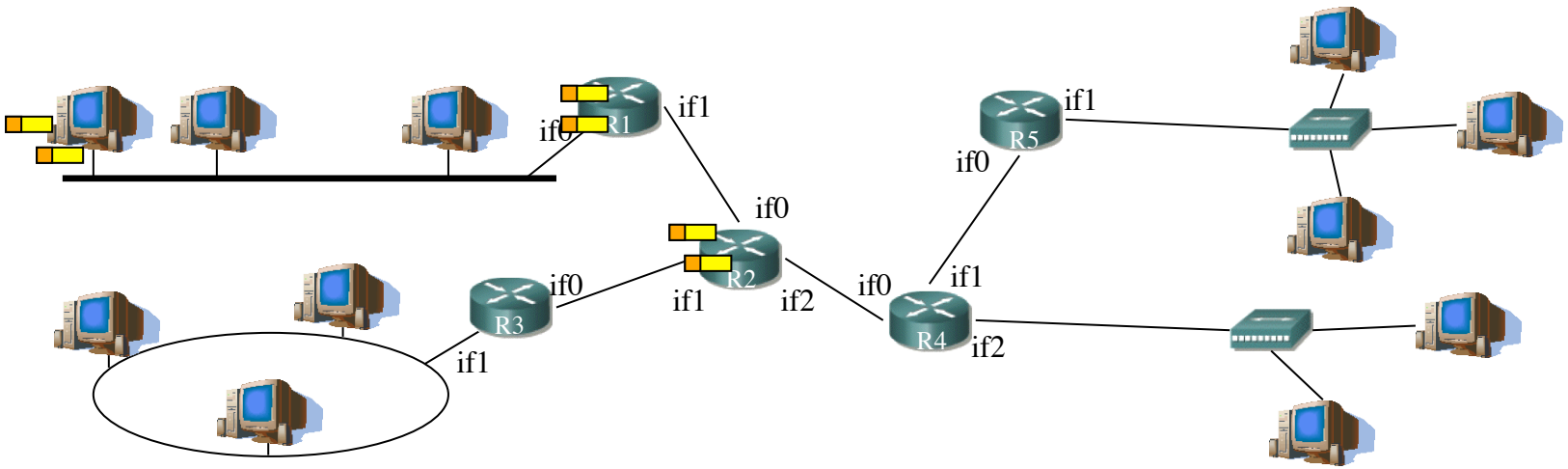
TTL	IP
1	$IPR1_{if0}$



Traceroute

- El host inicial envía un datagrama UDP (…)
  - Dirigido al host final
  - Con TTL = 2
- El primer router decrementa el TTL a 1 y lo reenvía (…)
- El segundo router decrementa el TTL a 0 (…)
  - Tira el paquete
  - Devuelve al origen un ICMP de Error *Tiempo excedido en tránsito*
  - Este es un paquete IP con dirección origen la del interfaz de R2 en dirección hacia el host origen (…)

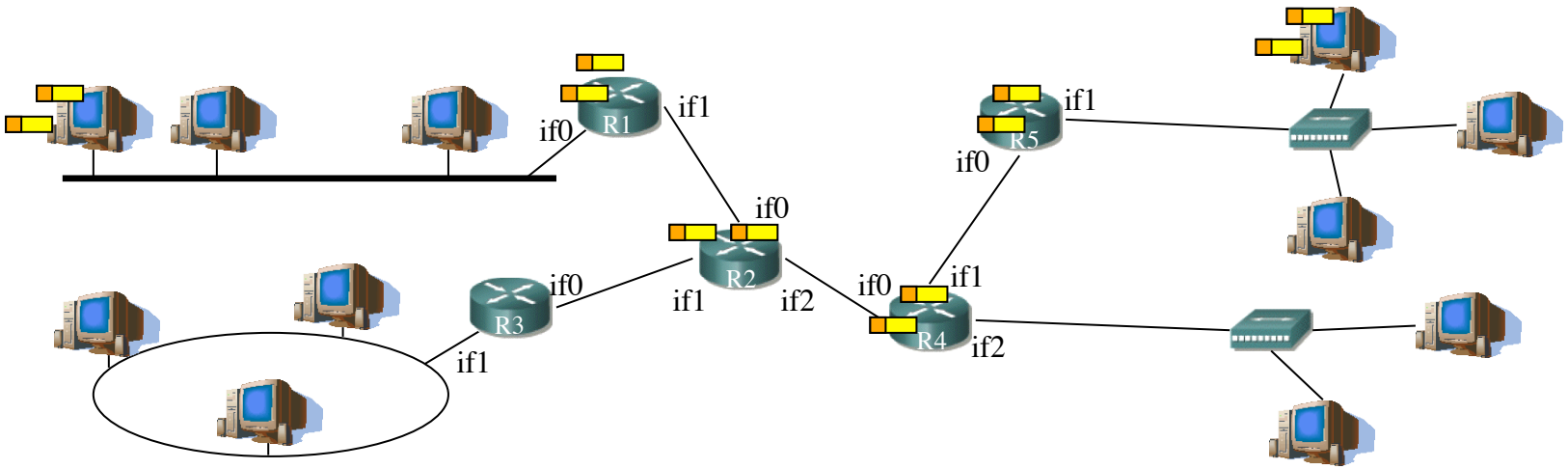
TTL	IP
1	$IPR1_{if0}$
2	$IPR2_{if0}$



Traceroute

- Idem con TTL=3 y TTL=4 (…)
- Con TTL suficientemente grande el paquete llega hasta el destino final (… …)
- En el destino no hay aplicación esperando paquetes UDP en ese puerto:
  - Lo tira
  - Devuelve al origen un ICMP de Error *Puerto destino inalcanzable (… …)*

TTL	IP
1	$IPR1_{if0}$
2	$IPR2_{if0}$
3	$IPR4_{if0}$
4	$IPR5_{if0}$
5	IPhost



Traceroute (Ejemplo)

```
daniel$ traceroute -n 64.170.98.32
traceroute to 64.170.98.32 (64.170.98.32), 64 hops max, 52 byte packets
 1  130.206.160.1  1.167 ms  1.151 ms  0.660 ms
 2  130.206.158.17  0.641 ms  0.791 ms  0.693 ms
 3  130.206.158.1  1.565 ms  1.251 ms  1.075 ms
 4  130.206.209.13  2.041 ms  1.480 ms  1.764 ms
 5  130.206.250.121  15.363 ms  8.340 ms  8.909 ms
 6  130.206.250.65  21.521 ms  20.057 ms  20.318 ms
 7  213.248.81.25  20.551 ms  20.661 ms  20.314 ms
 8  80.91.248.130  43.182 ms  42.759 ms  43.089 ms
 9  80.91.251.100  115.316 ms  114.909 ms  114.847 ms
10  80.91.250.5  115.404 ms  115.768 ms  115.413 ms
11  151.164.249.73  118.728 ms  118.529 ms  118.134 ms
12  151.164.95.192  191.823 ms  190.171 ms  188.512 ms
13  151.164.191.243  189.435 ms  189.415 ms  188.504 ms
14  75.61.192.10  191.332 ms  189.101 ms  191.219 ms
15  64.170.98.32  189.490 ms  192.163 ms  190.255 ms
```

Probadlo y ved los paquetes con tcpdump o wireshark