Redes de Computadores

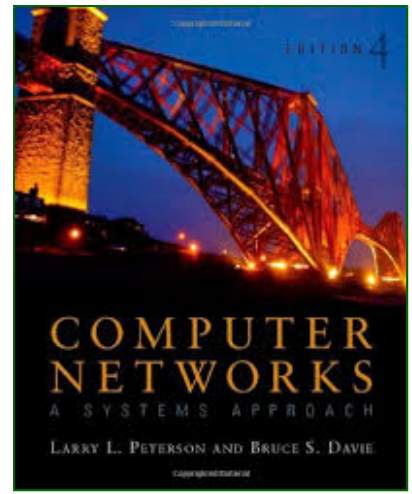
Tema 4 – Capa de red

Natalia Ayuso, Juan Segarra y Jesús Alastruey

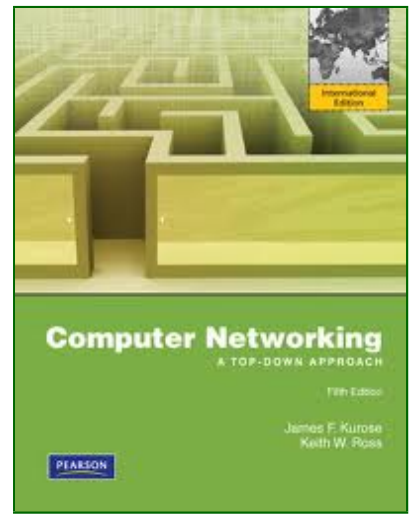
Departamento de
Informática e Ingeniería
de Sistemas
Universidad Zaragoza

Índice

1. Introducción
2. Modelos en conmut. paquetes
3. Encaminadores
4. Protocolo IP
5. Direcciones IPv4
6. IP versión 6
7. Túneles
8. Protocolos de encaminamiento
9. Estructura de Internet



Capítulos 3.1, 4



Capítulo 4

1 Introducción

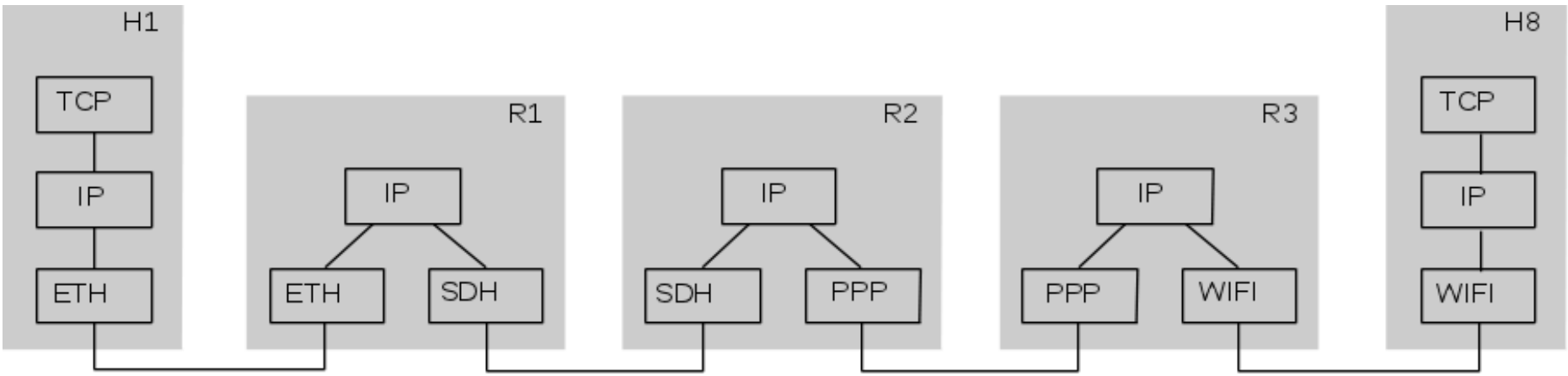
¿Por qué la capa 2 no sirve al aumentar el tamaño de la red?

- No puede interconectar redes físicas distintas
- Excesivo tamaño de tablas de encaminamiento en conmutadores aprendices (requiere una entrada por cada nodo en la red)
- Crecimiento desordenado:
  spanning tree $\rightarrow$ rutas no óptimas
- Problemas con mensajes de difusión total (broadcast)

1 Introducción (II)

Solución: pasar de red «física» (capa 2) a redes lógicas interconectadas → capa de red (capa 3)

- Abstrae diferencias de funcionamiento de redes físicas
- Todas las redes deben tener un protocolo común para interconectarse: IP



Internet es un conjunto mundial de redes interconectadas con protocolos comunes (TCP/IP) y direccionamiento universal (IP)

2 Índice

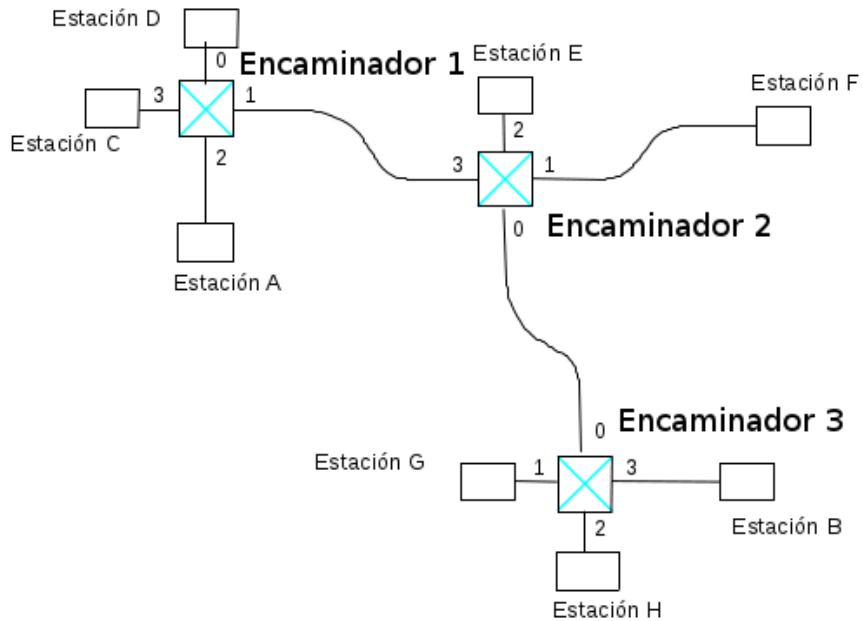
2 Modelos en conmutación de paquetes

Dependiendo de la información usada para decidir el camino, existen varios modelos dentro de la conmutación de paquetes:

- *Conmutación por datagrama*
  - En cada esquina preguntar a alguien por qué calle nos acercamos más al destino
- *Encaminamiento fuente*
  - Buscar la ruta y anotarla antes de iniciar el viaje para después seguirla
- *Conmutación por circuito virtual*
  - Especificar un destino a una agencia e iniciar el viaje hasta destino con los trasbordos ya organizados
- Combinaciones de los modelos anteriores

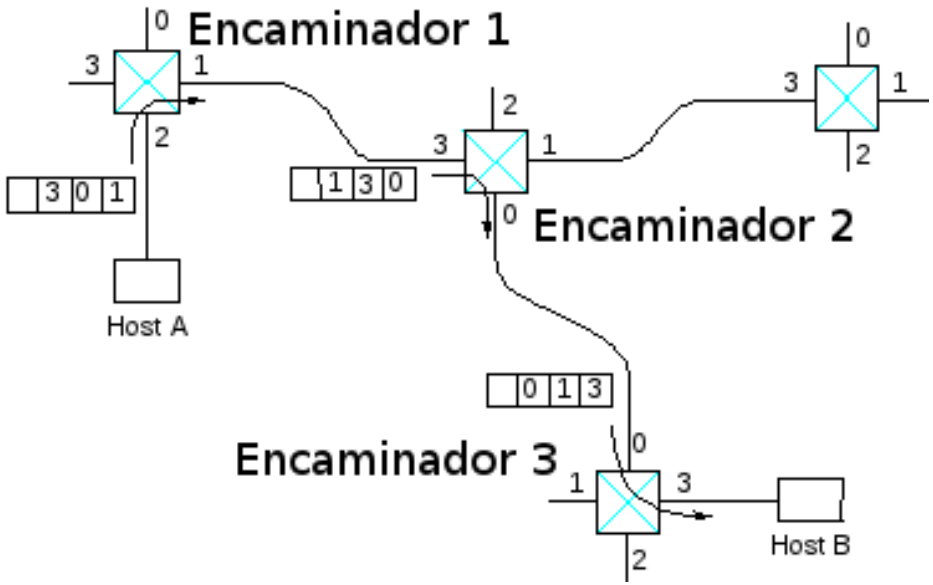
2.1 Conmutación por datagrama

- *Usado en Internet* en el protocolo IP
- Cada paquete es encaminado de forma independiente
  - No hay establecimiento del camino
  - Paquetes con mismo origen-destino pueden ir por distintos caminos
  - Alta tolerancia a fallos
- Cada *encaminador* enruta en función de la dirección destino de cada paquete
  - Debe conocer el camino a ¡cualquier destino! [p. 66]
  - Mantiene una *tabla de encaminamiento* con esa info
  - Cada tabla de encaminamiento es distinta



2.2 Encaminamiento fuente

- El propio paquete lleva la ruta a seguir
  - El emisor debe conocer la topología de la red
  - La información de ruta (campo «siguiente puerto») se modifica en cada encaminador (rotación, punteros, etc.)
  - La cabecera tiene un tamaño variable sin límite

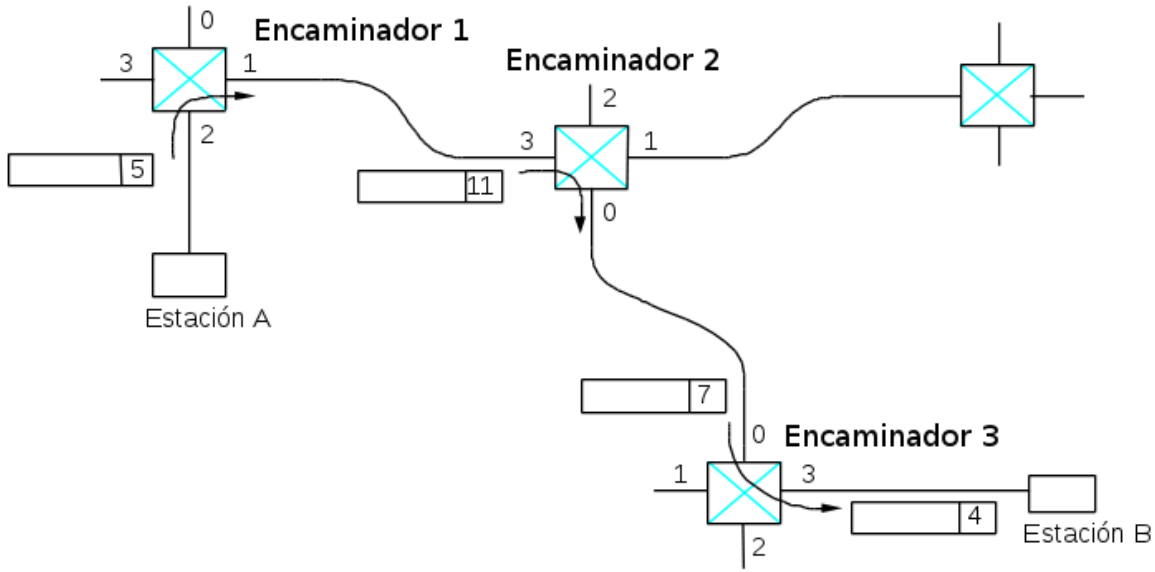


- El protocolo IP permite usar encaminamiento fuente añadiendo ciertas opciones en la cabecera de los paquetes

2.3 Conmutación por circuito virtual

- Conmutación de circuitos vía software
- Fase de creación (y destrucción) explícita de circuitos (paquetes especiales de ida y vuelta)
- Paquetes de datos se enrutan por circuito creado
- Cada encaminador mantiene *tabla de circuitos virtuales*
- Petición de conexión contiene dirección completa destino, pero cada paquete de datos solo un pequeño identificador
- Si un encaminador o un enlace de una conexión falla, la conexión se deshace y se necesita establecer una nueva
- Establecimiento de conexión proporciona oportunidad para reserva de recursos (QoS) [Tema 6]
- E.g. X.25, Frame relay, ATM

2.3 Conmutación por circuito virtual (II)



Después de establecer el circuito A-B:

1. A envía paquete con etiqueta de entrada a circuito A-B (5)
2. Encaminador 1 ve etiqueta 5 y enruta por puerto 1 con etiqueta 11 (tabla circuitos virtuales)
3. E2 ve etiqueta 11 y enruta por puerto 0 con etiqueta 7
4. E3 ve etiqueta 7 y enruta por puerto 3 con etiqueta 4

2.4 Multi Protocol Label Switching

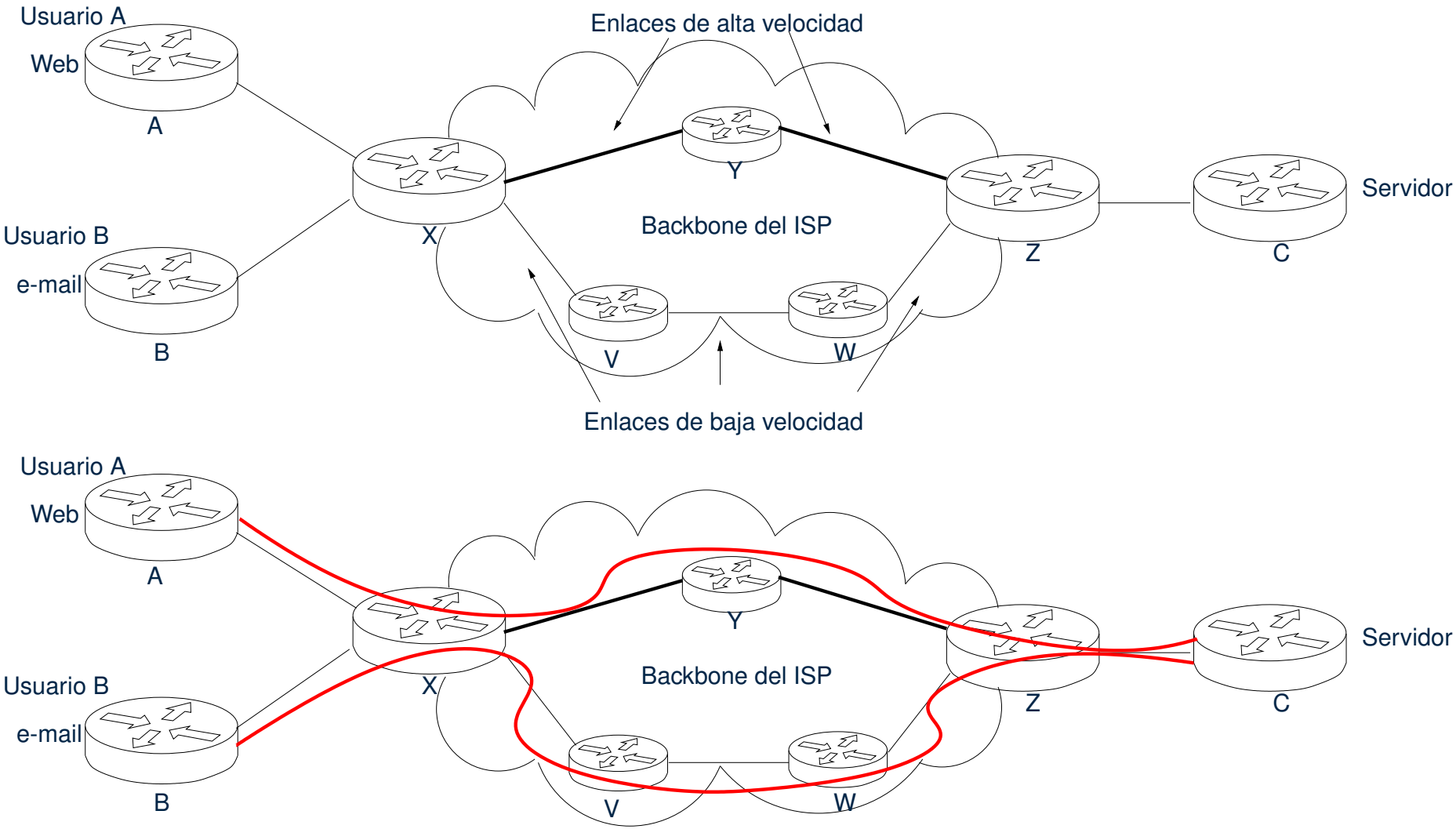
- Sobre conmutación por datagrama, los encaminadores solicitan que para ciertos destinos los paquetes incorporen ciertas etiquetas
- Conmutación de etiquetas multiprotocolo
- Combinación de:
  - Circuitos virtuales (etiquetas cortas de longitud fija y ámbito local)
  - con flexibilidad y robustez de datagramas
- Características:
  - Necesita de protocolos encaminamiento IP para crear rutas
  - Es capaz de llevar cualquier protocolo de la capa de red
  - Situado conceptualmente entre capas 2 y 3

2.4 Multi Protocol Label Switching (II)

- Originalmente para mejorar prestaciones de Internet
- Utilización actual: encaminamiento explícito
  - Reexpedición basada en destino: por construcción, es el destino el que solicita etiquetas
  - Encaminamiento explícito
  - Creación de túneles [p. 63]

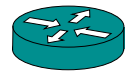
2.4 Multi Protocol Label Switching (III)

Ejemplo encaminamiento explícito:



3 Índice

3 Encaminadores

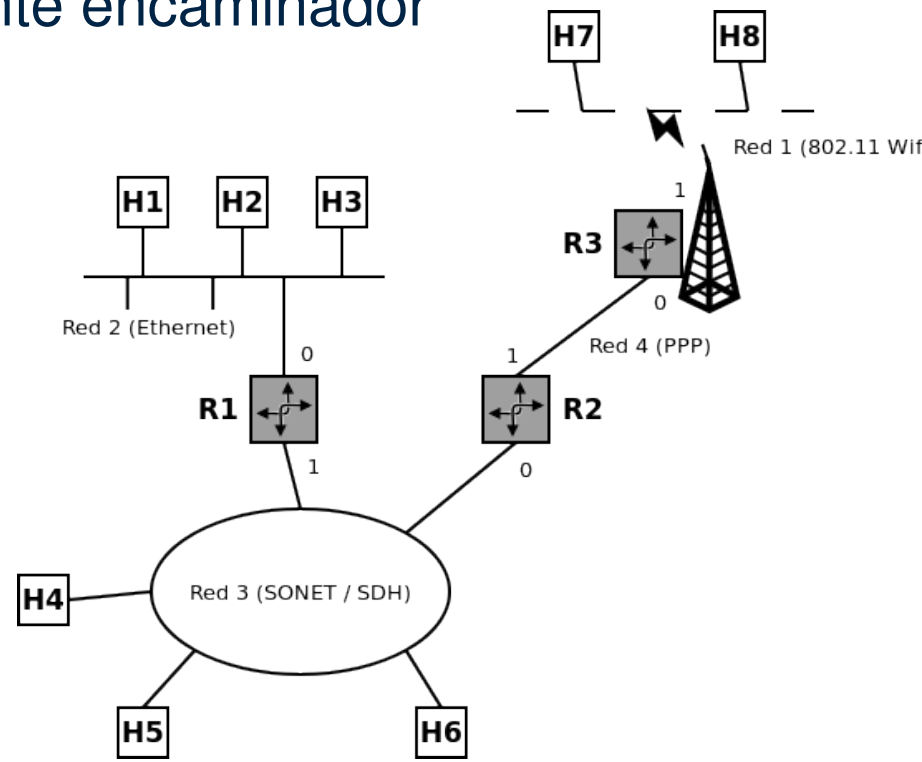
- Encaminador/*router*: conmutador de capa de red (capa 3)
  - Interconecta redes: inter-net 
  - Trabaja con direcciones lógicas (dir. IP)
  - Varios interfaces o puertos, *distintos tipos de redes*
  - Conoce ruta óptima hacia cada red destino
  - Sólo enruta hacia donde corresponde
  - Nunca enruta tramas broadcast (MAC: ff:ff:ff:ff:ff:ff)
- Cada puerto: dir. lógica (IP) y física (MAC)
- Ejemplo router ADSL: Ethernet + PPPoE (ADSL) + WiFi

3.1 Enrutamiento desde encaminador

- Cada encaminador tiene una tabla con una *lista de destinos* y cómo acercarse/llegar a ellos
- Ante un paquete con una dir. destino, la busca en su tabla
  - Si está directamente conectado a la red destino → enruta hacia el nodo destino por interfaz correspondiente
  - Si no → enruta hacia siguiente encaminador según la tabla (sig. salto)

Tabla de encaminamiento de *R2*

Destino	Sig. salto	Interfaz
Red 1	R3	1
Red 2	R1	0
Red 3	—	0
Red 4	—	1

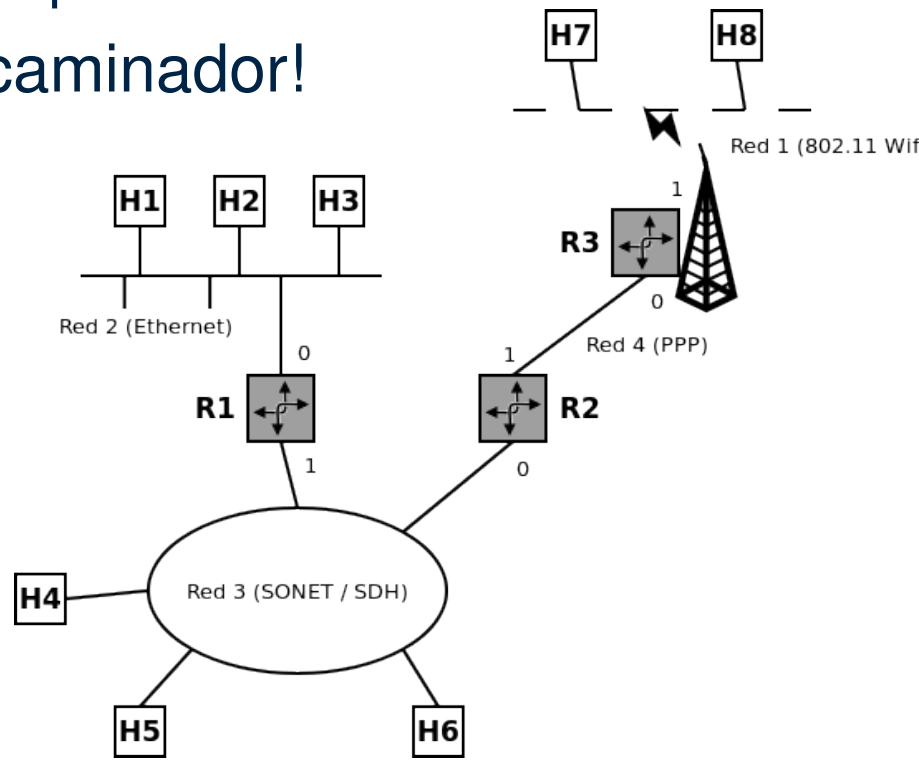


3.2 Enrutamiento desde nodo

- Cada nodo tiene una tabla con su red y su *encaminador (gateway) por defecto*
- Para enviar un paquete hacia dir. destino:
  - Si está en la propia red → envía a nodo destino
  - Si no → envía a encaminador por defecto
- ¡Mismo procedimiento que encaminador!

Tabla de encaminamiento de *H1*

Destino	Sig. salto	Interfaz
Red 2	—	0
default	R1	0



3.3 Ejemplo tablas

```
lab000:~$ /sbin/route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         155.210.152.254 0.0.0.0         UG    425    0        0 br0
155.210.152.0   0.0.0.0         255.255.255.0   U     425    0        0 br0
```

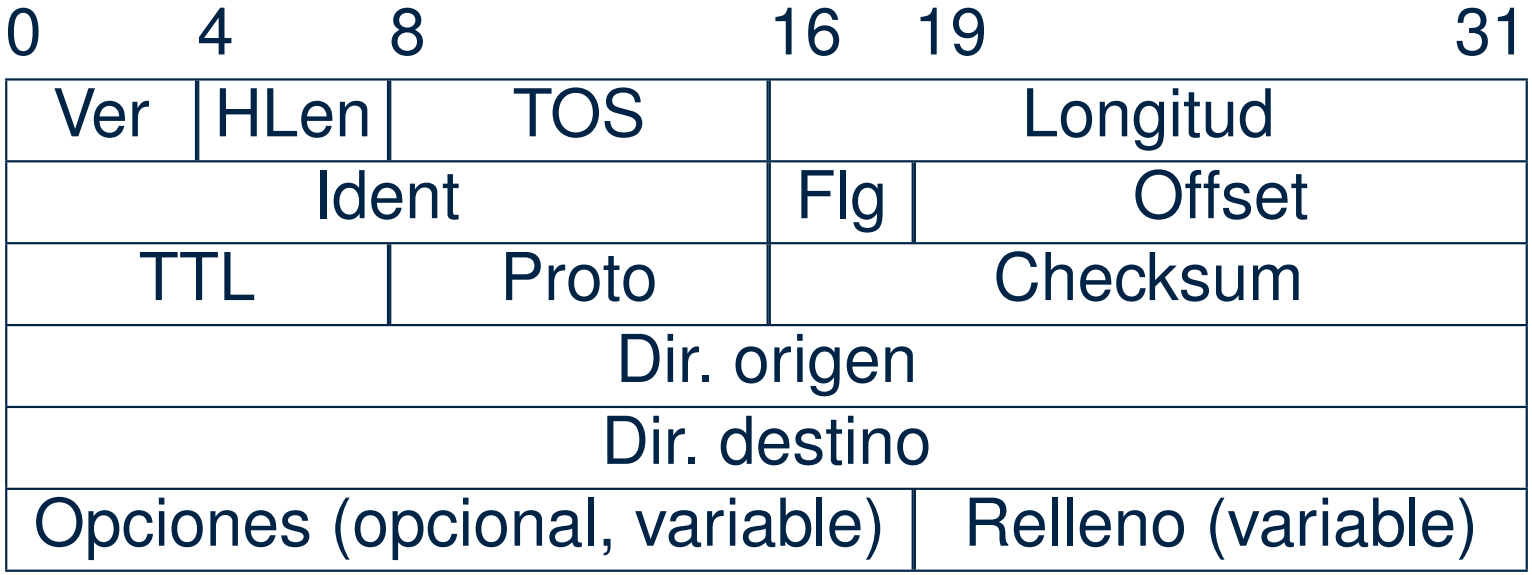
```
lab000:~$ ip route
default via 155.210.152.254 dev br0 proto dhcp src 155.210.152.177 metric 425
155.210.152.0/24 dev br0 proto kernel scope link src 155.210.152.177 metric 425
```

4 Índice

4 Protocolo IP

- Internet Protocol (IP)
- Conmutación por datagrama [p. 7]
- Servicio *no fiable* (*unreliable*)
  - La red hace lo posible para la entrega de los paquetes, pero sin garantizarlo (*best effort*)
  - Pueden perderse paquetes
  - Pueden llegar desordenados respecto al envío
  - Se pueden entregar paquetes duplicados
  - El tiempo de entrega puede ser muy variable
- Actualmente funciona la versión 4 del protocolo IP, pero la transición a IPv6 está en marcha [p. 56]

4.1 Cabecera IPv4



Versión: 4, desde comienzo de los años 80

Hlen: longitud de la cabecera, en palabras de 32 bits.
Usualmente 5 $\rightarrow$ 20 bytes, máximo 15 $\rightarrow$ 60 bytes

Type Of Service (TOS): calidad de servicio (QoS) [Tema 6]

Longitud: longitud total del paquete, en bytes

- 16 bits $\rightarrow$ máximo: 65535 bytes $(2^{16} - 1)$

Identificador, Flags, Offset: campos para fragmentación [p. 23]

4.1 Cabecera IPv4 (II)

Time-To-Live: valor fijado por emisor y decrementado por cada encaminador atravesado. Paquete descartado por encaminador que decrementa el valor a 0.

Protocolo: tipo de datos que contiene el paquete (ICMP, IGMP, TCP, UDP). Usado para demultiplexación en la capa superior

Checksum: verificación de la cabecera, no de los datos

Direcciones: IP origen y destino, 32 bits cada una

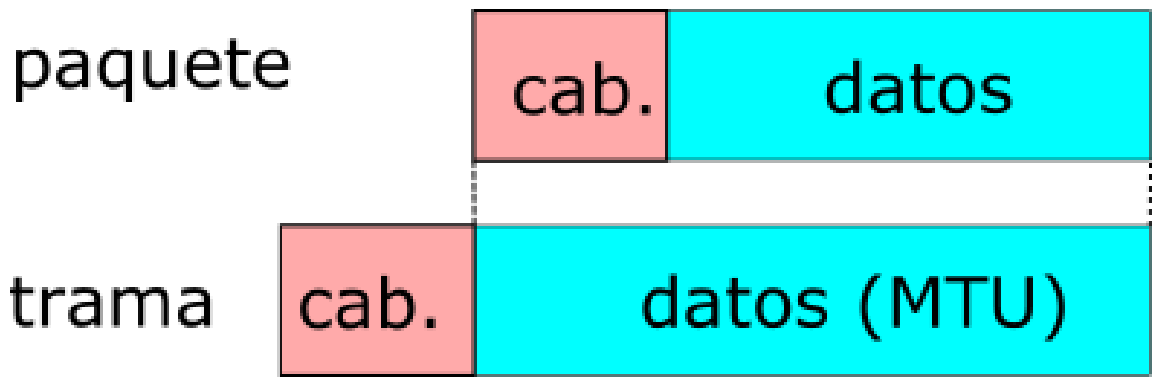
Opciones:

- Pedir que los encaminadores atravesados añadan información al paquete (su IP, la hora, etc.)
- Encaminamiento fuente: lista de encaminadores que pueden/deben ser atravesados
- Y otras

Relleno: para que los datos del paquete comiencen en posición múltiplo de 32 bits

4.2 Fragmentación y reensamblado

- *MTU*, *maximum transmission unit*: tamaño del mayor paquete que puede enviarse en los datos de una trama
- Depende del protocolo de la capa de enlace.
  - Ej. $1\,500$ B en 802.3, $4\,464$ B en 802.5, $7\,981$ B en 802.11



4.2 Fragmentación y reensamblado (II)

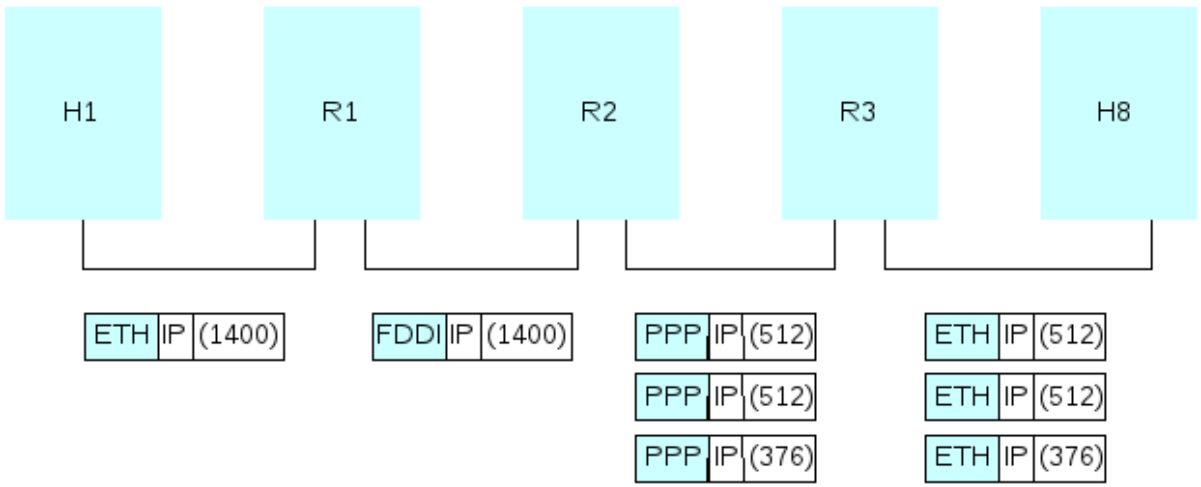
- Un paquete generado en una red puede no caber en otra
- Estrategia IPv4:
  - Fragmentar cuando MTU $<$ longitud paquete
  - El encargado de fragmentar es el encaminador directamente conectado a la red que requiere tramas más pequeñas
  - Los fragmentos son paquetes completos en sentido estricto: es posible fragmentar fragmentos
  - Reensamblado en nodo destino

Fuente: Cloudfare: Broken packets: IP fragmentation is flawed

4.2.1 Ejemplo fragmentación

Ejemplo fragmentación:

- Ident (16 bits): identificador de paquete (igual en todos los fragmentos)
- *Flags* (3 b): reservado (0), Prohibido fragmentar, Siguen más fragmentos
- *Offset (13 b)*: posición de los datos en el paquete original (unidades de 8 B)



Inicio cabecera				
Ident x	0	0	0	Offset 0
Resto cabecera				
1400 bytes datos				

$\Downarrow$

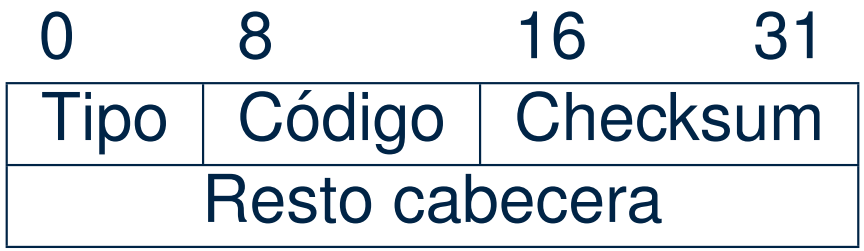
Inicio cabecera				
Ident x	0	0	1	Offset 0
Resto cabecera				
512 B datos ($/8 = 64$)				

Inicio cabecera				
Ident x	0	0	1	Offset 64
Resto cabecera				
512 bytes datos				

Inicio cabecera				
Ident x	0	0	0	Offset 128
Resto cabecera				
376 bytes datos				

4.3 ICMP

- Internet Control Message Protocol, RFC 792
- Encaminador o nodo destino informan al nodo emisor de un error, por ejemplo, paquete que no ha llegado a destino, paquete descartado . . .
- No se envían mensajes ICMP sobre mensajes ICMP
- Encapsulado dentro de un paquete IP
- Cabecera ICMP: 8 bytes



Tipo/Código: tipo/subtipo de mensaje ICMP

Checksum: del mensaje ICMP (cabecera + datos)

Resto cabecera: depende del tipo de mensaje

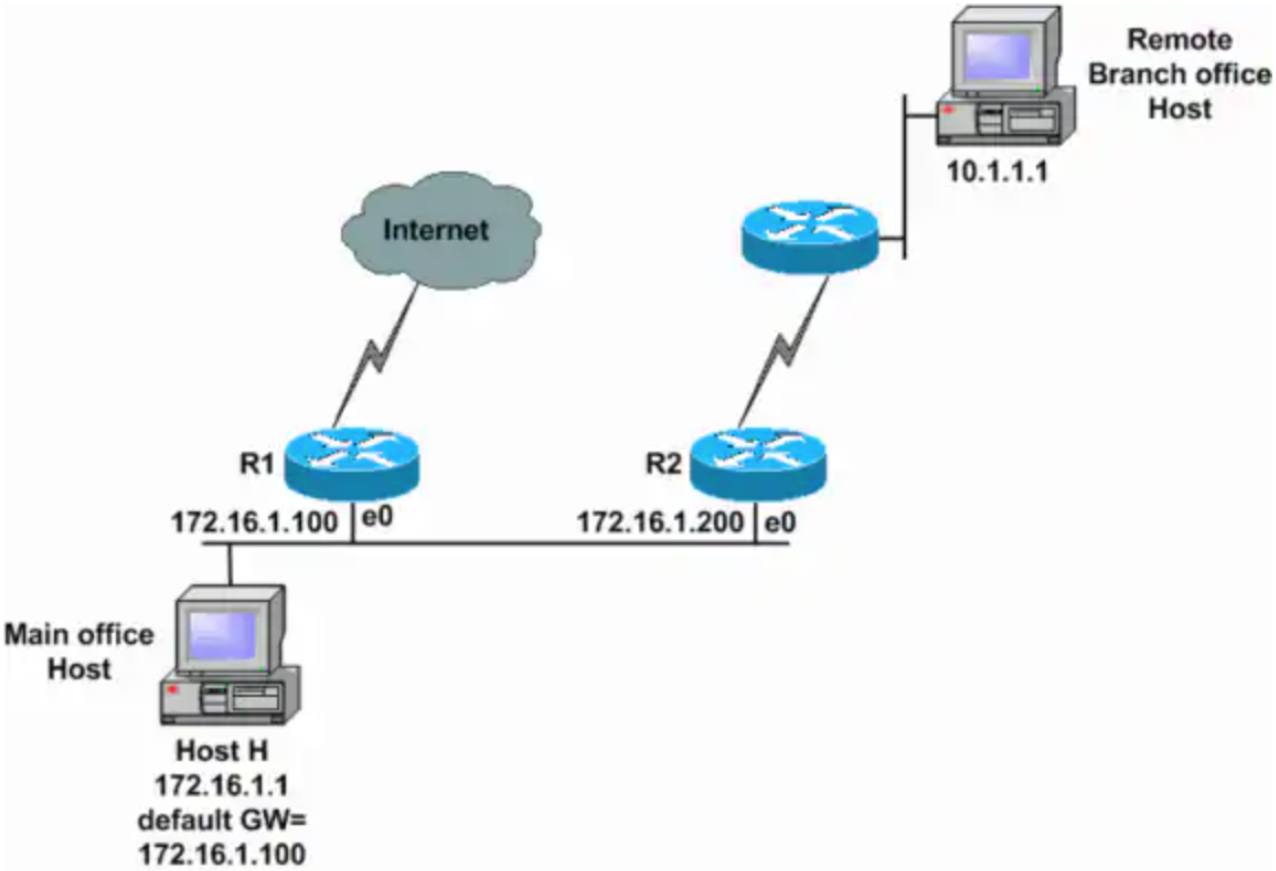
4.3.1 Ejemplos ICMP

- Diagnóstico de red
  - ping: disponibilidad y latencia de un enlace
  - traceroute: ruta entre origen y destino
- Se requiere fragmentar pero DF+

Fuente: Cloudfare: Broken packets: IP fragmentation is flawed

4.3.1 Ejemplos ICMP (II)

- Redirect: router notifica a nodo que hay una ruta mejor hacia un destino



Fuente: Cisco: When Are ICMP Redirects Sent?

4.3.2 Ejercicio ping + fragmentación

✍ En una red 802.3, el MTU es de 1500 bytes. ¿Cuántos datos se pueden enviar en un paquete ICMP sin que haya fragmentación?

4.4 Procesado de un datagrama IP

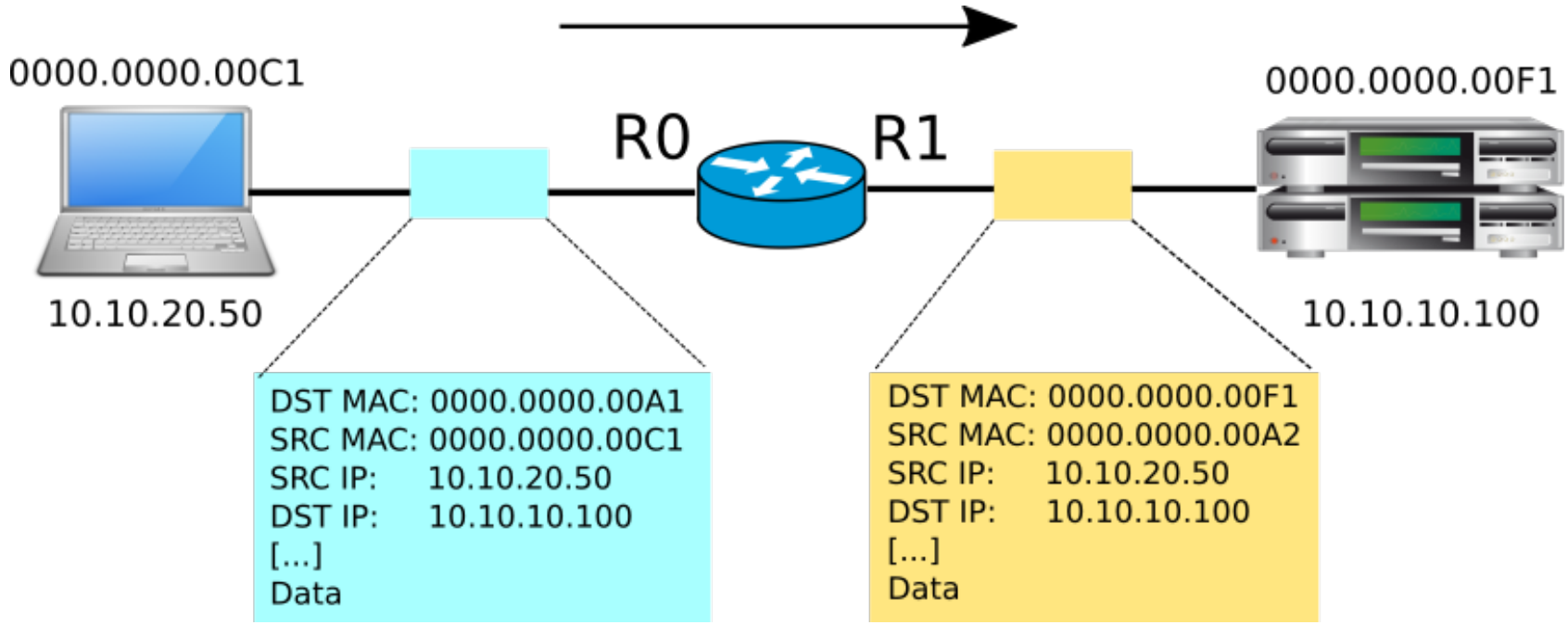
Al recibir una trama, el encaminador debe:

1. Validar trama (checksum/crc/etc.)
2. Validar cabecera IP (comprobar checksum)
3. Procesar opciones de la cabecera (si las hay)
4. Buscar dirección destino en tabla de encaminamiento
5. Decrementar TTL
6. Realizar fragmentación (si es necesario)
7. Calcular checksum (por cada fragmento)
8. Construir trama de capa inferior (cabecera, crc, etc.)
9. Transmitir a siguiente salto por interfaz correspondiente
10. Generar y enviar paquete ICMP (si es necesario)

El tiempo de procesado en cada salto *no es despreciable*

4.4 Procesado de un datagrama IP (II)

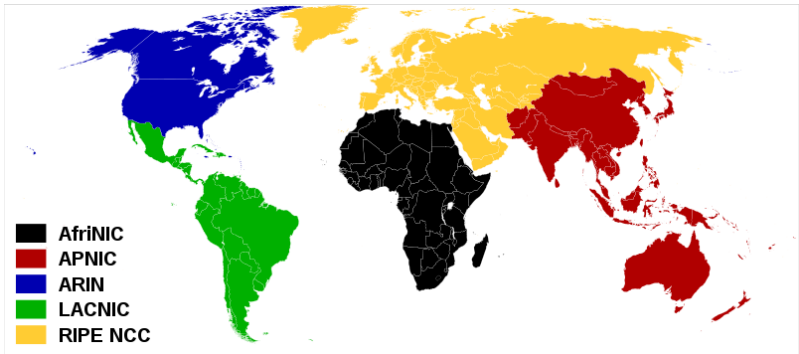
- Ejemplo de procesado de paquete en router:



5 Índice

5 Direcciones IPv4

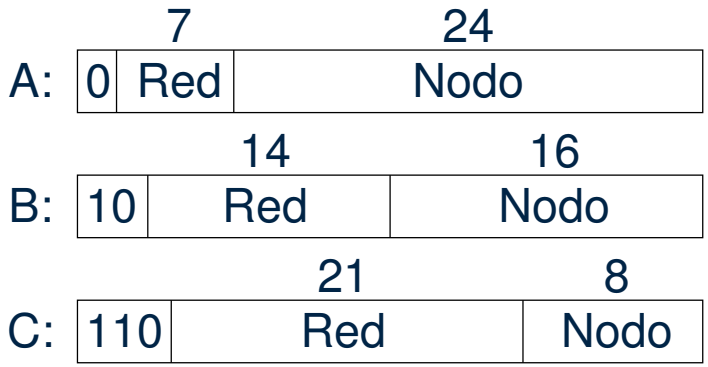
- Direcciones de 32 bits
- Globalmente únicas, excepto las privadas
- Jerárquicas: red + nodo
- Asociadas a interfaces de red más que a nodos
- Internet Assigned Numbers Authority (IANA) reparte bloques de direcciones IPv4 a Registros Regionales de Internet (RIR) bajo demanda → *todos ya repartidos*
  - AfriNIC: Africa
  - APNIC: Asia-Pacífico
  - ARIN: EE.UU.-Canadá
  - LACNIC: Lat.América-Caribe
  - RIPE: Europa-Asia occidental



- DNS traduce nombres a direcciones IP [Tema 7]

5.1 Clases de direcciones IPv4

- Notación de puntos:
  - 10.3.2.4
  - 128.96.33.81
  - 192.12.69.77



- 5 formatos de dirección:

Clase	Prefijo	1er byte	Nº redes	Nº dir./red	Uso
Clase A	0	0-127	128	16777216	Unicast
Clase B	10	128-191	16348	65536	Unicast
Clase C	110	192-223	2097152	256	Unicast
Clase D	1110	224-239			Multicast
Clase E	1111	240-255			Experimental

- Históricamente, a una organización se le asignaba un id. red de una clase, por ejemplo, 155.210.0.0
- Actualmente, las direcciones se consideran sin clase (CIDR), por ej., puede asignarse el id. red 193.10.20.192

5.1.1 Direcciones sin clase

- Ignora la distinción entre clases A, B y C y las fronteras entre identificadores de red y nodo
- Notación dirección red: dirección red/número bits red
  - 155.210.0.0/16
  - 206.62.226.0/24
- El uso de direcciones sin clase requiere encaminamiento sin clase: *Classless Inter-Domain Routing*, RFC 1519
- Objetivos CIDR
  - Reducir tamaño tablas encaminamiento routers
  - Reducir agotamiento direcciones IPv4

5.2 Direcciones especiales

- Dirección de red: todo 0s en bits de nodo. Ej: 128.96.0.0
- Difusión/broadcast: todo 1s en bits nodo. Ej: 192.12.69.255
- 0.0.0.0: dirección no encaminable.
  - Significado dependiente del contexto, ej. ruta por defecto
- 127.0.0.0/8: bucle (*loopback*), el propio nodo
  - Ejemplo: 127.0.0.1
  - RFC 1122, Section 3.2.1.3
- 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16: privadas
  - No únicas globalmente $\rightarrow$ no sirven como dirección destino a nivel global (Internet)
  - Válidas a nivel local $\rightarrow$ redes privadas
  - Para conectarse a Internet se necesita traducción de direcciones privadas (NAT) [p. 50]
  - RFC 1918

5.2 Direcciones especiales (II)

- 169.254.0.0/16: link-local
  - Dirección que se autoasigna un nodo cuando no ha podido obtener otra mediante configuración manual (fichero) o automática (DHCP)
  - Válida únicamente para comunicación local
  - Bits de nodo: valores aleatorios para mitigar conflictos
  - RFC 3927
- Paquetes con destino difusión/privadas/link-local *no salen de la red local*

5.3 Subredes

- Direcciones unicast posibles en una red: $2^{\text{nº bits nodo}} - 2$
- Problema: redes muy grandes
  - Broadcast excesivo, colisiones, retardos, pérdida de tramas
- Solución: añadir nivel de jerarquía red-*subred*-nodo
  - Se usan *bits de nodo* para identificar subredes
  - Cada subred necesita @ red, broadcast y encaminador
  - Fuera de una red no hay conciencia de sus subredes (los encaminadores externos propagan hasta la red)
- Gestionadas mediante *máscaras de subred*

5.3.1 Máscara de subred

- Información total & Máscara de bits = Información útil

- Dirección IP & Máscara de subred = Dirección de subred
  - Red 155.210.0.0/16

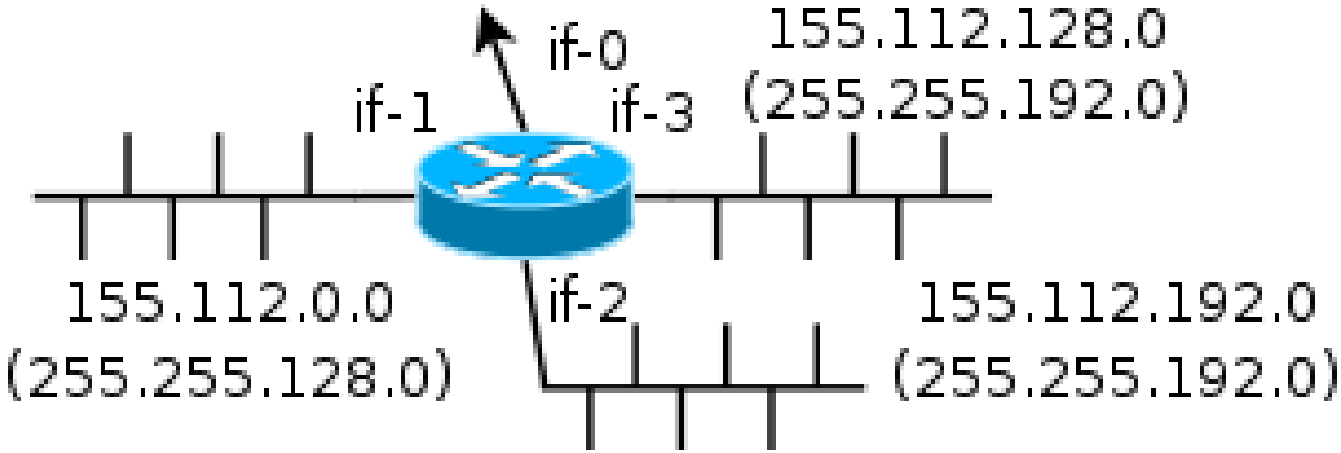
bytes id. red | bits id. red | bits id. nodo

IP 155.210.153.238 - 10011011.11010010.10011001.11101110
Mask 255.255.248.0 - 11111111.11111111.11111000.00000000
Sub 155.210.152.0 - 10011011.11010010.10011000.00000000

bits subred | bits nodo

- Subred 155.210.152.0/255.255.248.0 = 155.210.152.0/21

5.3.2 Ejemplo subredes



Destino	Máscara subred	Interfaz	(Destinos)
155.112.0.0	255.255.128.0	If-1	155.112.0xxxxxxx.X
155.112.128.0	255.255.192.0	If-3	155.112.10xxxxxx.X
155.112.192.0	255.255.192.0	If-2	155.112.11xxxxxx.X
default	-	if-0	

✍ Resolver con notación CIDR

5.3.3 Ejercicio subredes

✍ Para la red 155.210.0.0/16, ¿son válidas estas máscaras?

a) 255.224.0.0 (`11111111.111`**`0`**` 0000.00000000.00000000`)

b) 255.255.216.0 (`11111111.11111111.11`**`0`**`11000.00000000`)

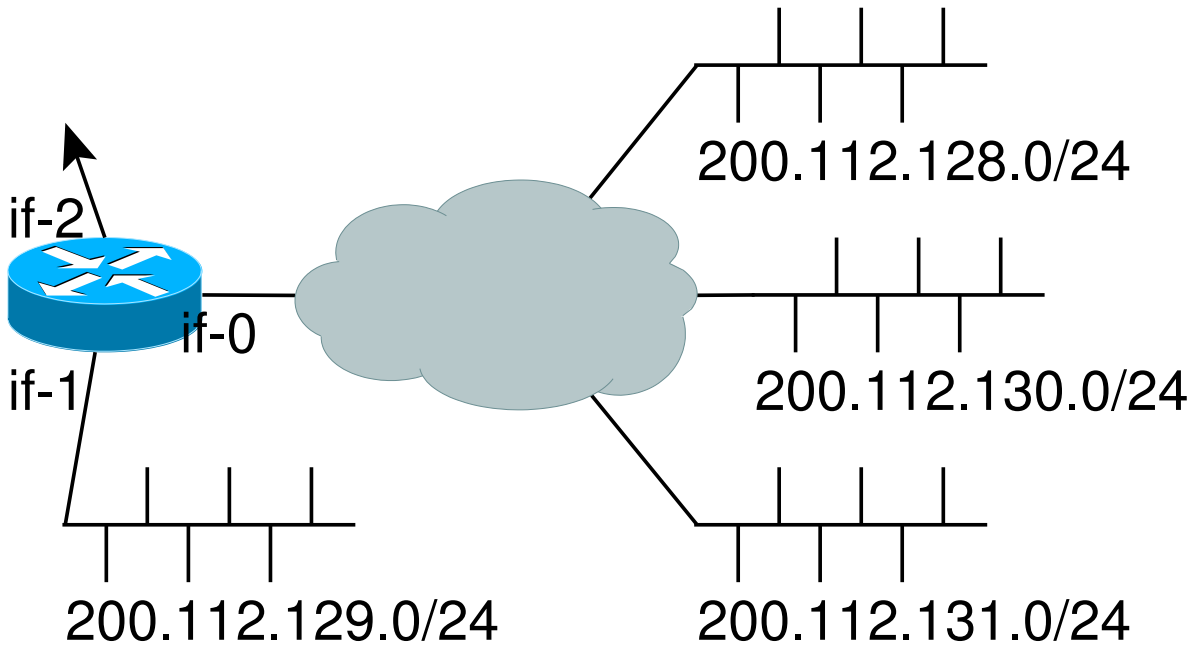
c) 255.255.255.0 (`11111111.11111111.11111111.00000000`)

5.4 Superredes

- RFC 4632
- Los encaminadores «necesitan» una entrada en la tabla por cada red destino, y hay más de 2 millones de redes
- Problema: con tantas entradas las tablas no son eficientes
- Objetivo: juntar varias redes contiguas en una única entrada en tabla
  - Redes 192.4.8.0/24 - 192.4.15.0/24 $\rightarrow$ 192.4.8.0/21

$$\left.\begin{array}{r} 192.4.00001000.0 \\ 192.4.00001001.0 \\ 192.4.00001010.0 \\ \ldots \\ 192.4.00001111.0 \end{array}\right\} 192.4.8.0/21$$

5.4.1 Ejemplo CIDR



Destino	Interfaz
200.112.128.0	If-0
200.112.129.0	If-1
200.112.130.0	If-0
200.112.131.0	If-0
default	If-2

Destino	Interfaz
200.112.129.0/24	If-1
200.112.128.0/22	If-0
default	If-2

- If-0: 200.112.128.0, 130.0, 131.0 ($100000\frac{0}{1}\frac{0}{1}$)
  - 22 bits de red comunes: 200.112.100000xx.xxxxxxxx
  - Prefijo común: 200.112.*128.0/22*
  - Redes 128, 130 y 131 se agrupan en una única entrada
- If-1: 200.112.129.0 (10000001) más prioritaria en tabla por estar incluida en 200.112.128.0/22

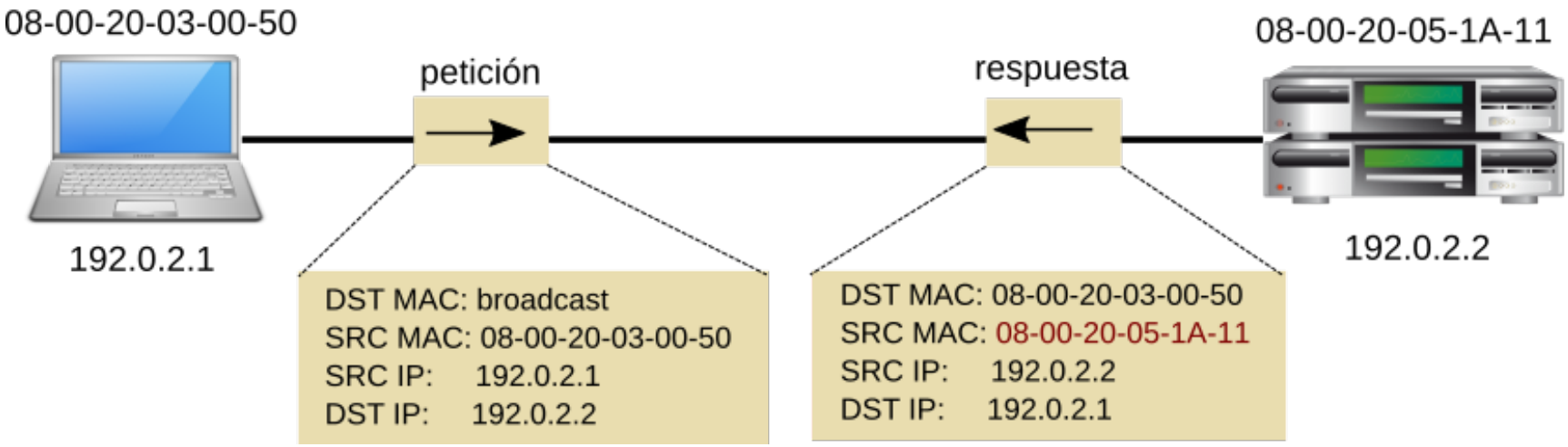
5.5 ¿Cómo usar menos direcciones IP?

- Problema: agotamiento de direcciones IPv4 unicast
- Servidores necesitan estar localizables en todo momento para recibir peticiones
  - IP *estática* o fija: dirección IP que no cambia con el tiempo
- Clientes no necesitan estar localizables
  - IP *estática*, si hay para todos
  - IP *dinámica*: @IP que puede cambiar con el tiempo
- Asignación dinámica desde un servidor (DHCP, PPP)
  - Al apagar el equipo, otro puede usar su @IP
  - Al reiniciar el equipo, podemos usar una @IP distinta
  - Una @IP nunca es usada por varios equipos a la vez
- Compartir direcciones IP unicast (NAT)
  - Una dirección IP es usada por varios equipos a la vez

5.6 Correspondencia IP-MAC

- La capa de red funciona con direcciones IP
- La capa de enlace funciona con identificadores MAC
- ¿Cuándo se necesita asociar ambas?
  - Nodo necesita el id. MAC de su encaminador por defecto y de los nodos de la misma red
  - Encaminador necesita el id. MAC del siguiente salto
  - Encaminador final necesita el id. MAC del nodo destino
- ¿Cómo se asocian ambas direcciones?
  - Codificar id. MAC dentro de dir. red (e.g. EUI-64)
    - Usado en IPv6
    - No sirve en IPv4 porque los identificadores MAC tienen un tamaño mayor que las direcciones IPv4
  - Mediante tablas, e.g. Address Resolution Protocol - ARP
    - Usado en IPv4

5.7 Address Resolution Prot. (ARP)



- Mensajes de petición y respuesta sobre capa de enlace
  1. Nodo origen consulta @IP destino en tabla ARP.
     Si no está, petición ARP por difusión
  2. Nodos que reciben petición actualizan su tabla con la correspondencia IP-MAC origen
  3. Nodo destino responde con su id. MAC
  4. Nodo origen actualiza su tabla con IP-MAC destino

5.7 Address Resolution Prot. (ARP) (II)

- Entradas de tabla ARP:
  - No se añaden por ningún otro procedimiento
  - Son eliminadas tras un tiempo sin usarse
- Ejemplo consulta tablas ARP mediante orden `arp`:

```
lab000:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
hendrix02.cps.unizar.es  ether   00:14:4f:ec:4d:54   C                     br0
155.210.152.254          ether   f0:f7:55:f3:c7:c1   C                     br0
camposancos.cps.unizar.  ether   00:1c:c0:ef:8d:8d   C                     br0
```

5.7.1 Ejemplo petición-respuesta ARP

Fuente: Sharetechnote.com: IP Network - ARP

5.8 DHCP

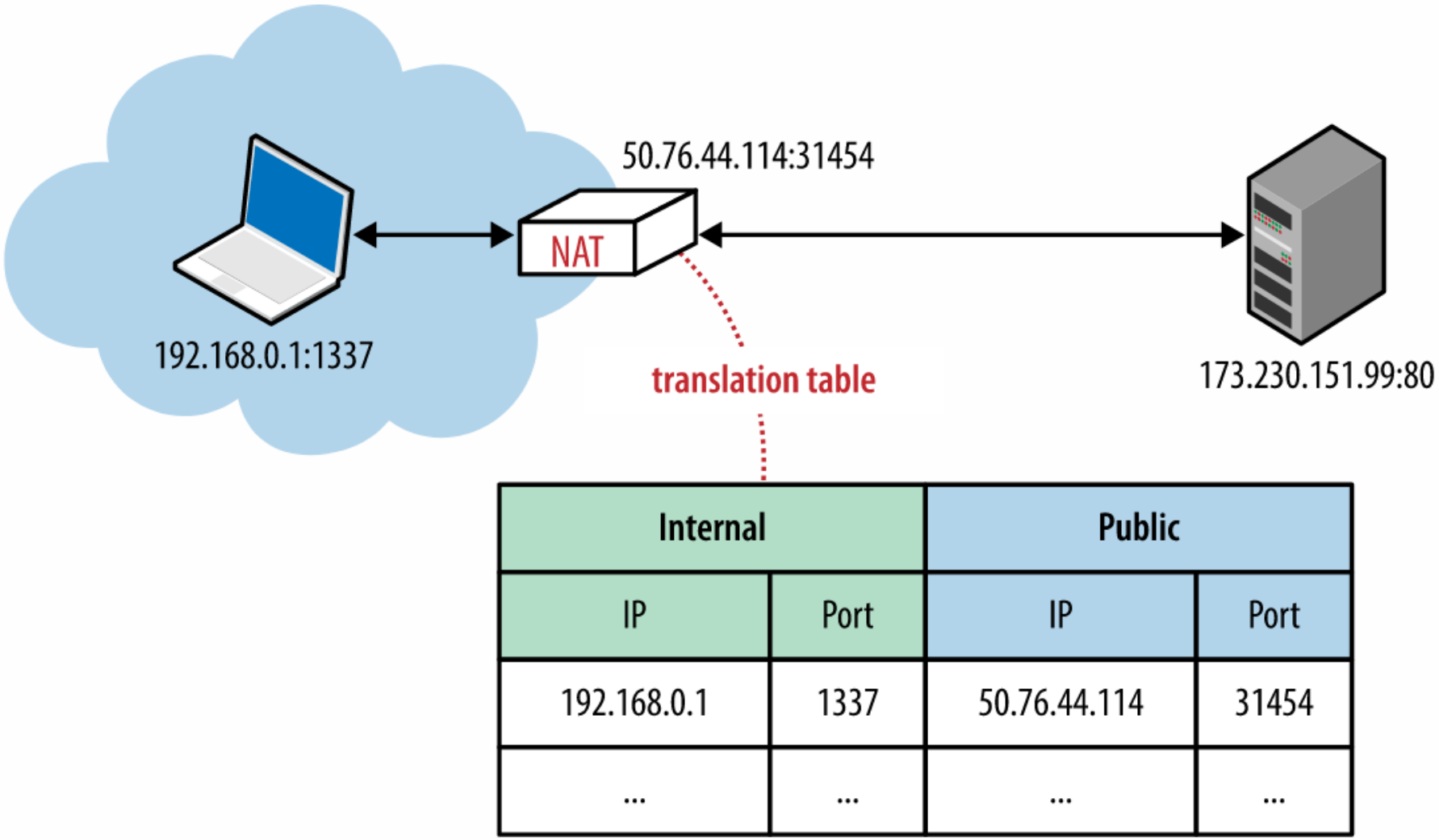
- Dynamic Host Configuration Protocol
- Un servidor DHCP proporciona información para que los equipos de su LAN configuren la red:
  - Dirección IP: no necesariamente la misma siempre
  - Máscara de subred
  - Encaminador por defecto
  - Servidor de nombres [Tema 7]
  - etc.
- Pasos de configuración:
  1. Al arrancar, el equipo busca un servidor DHCP:
     IPsrc: 0.0.0.0, IPdest: 255.255.255.255, UDP: 67
  2. El servidor responde anunciando su presencia:
     IPdest: 255.255.255.255
  3. El equipo pide datos al servidor
  4. El servidor proporciona los datos

5.9 Network Address Translation

1. Uso de @IP privadas en equipos locales
   - 10.X.X.X, 172.0001xxxx.X.X, 192.168.X.X
   - Válidas localmente pero NO globalmente
2. Enmascarar tras @IP válida con encaminador NAT
   - Network Address (and port) Translator
   - El encaminador tiene una @IP unicast válida
   - Salida: NAT sustituye @IP origen por la propia
   - Entrada: NAT deshace la sustitución: deduce a quién entregar el paquete

- *Solución arquitectónicamente mala* 
  - Modifica el funcionamiento básico de la capa de red (el NAT cambia direcciones en paquetes)
  - Implica restricciones de funcionalidad
- Carrier-grade NAT / Large-scale NAT: NAT dentro de NAT

5.9 Network Address Translation (II)

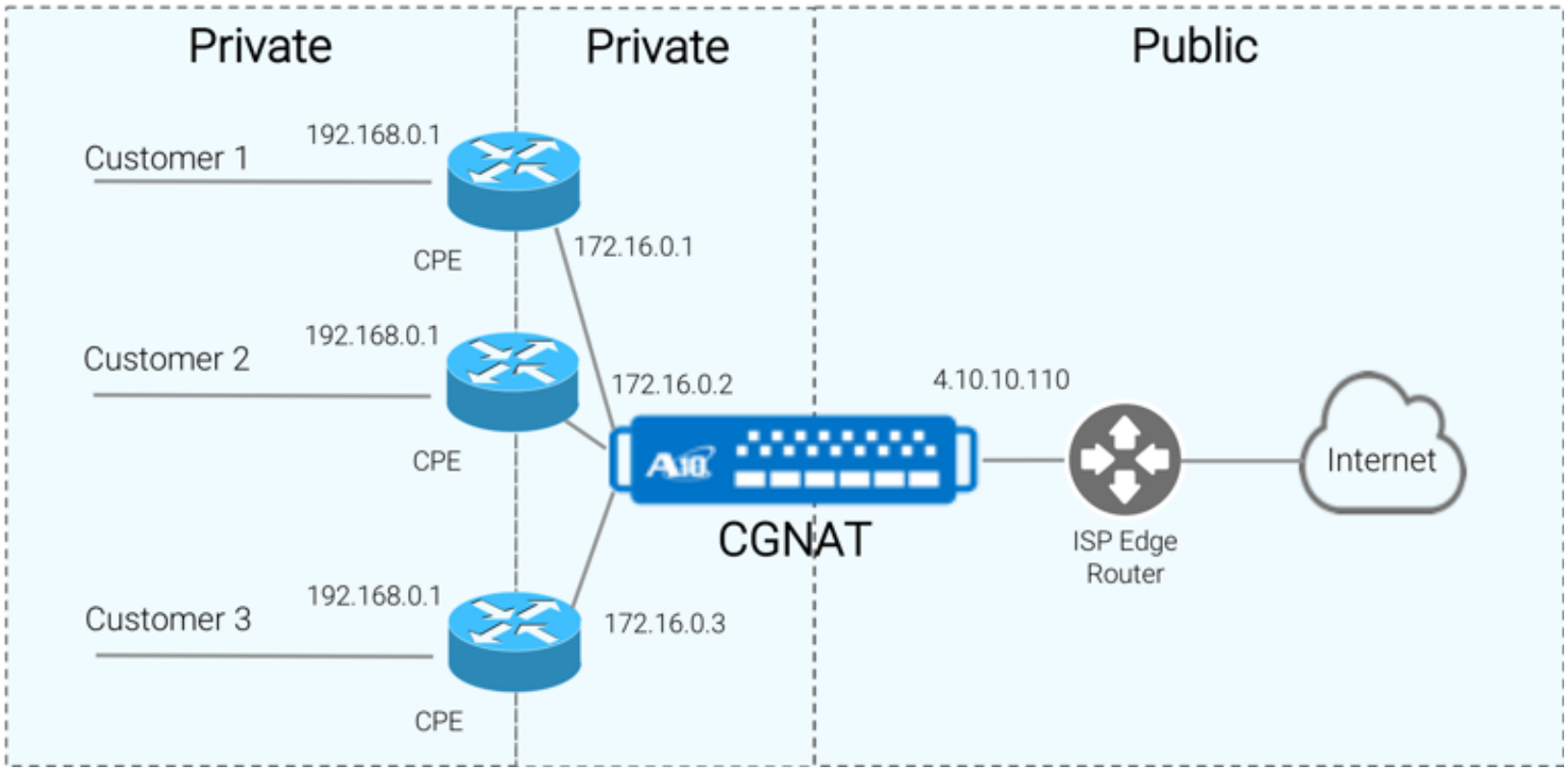


Internal		Public	
IP	Port	IP	Port
192.168.0.1	1337	50.76.44.114	31454
...	...	...	...

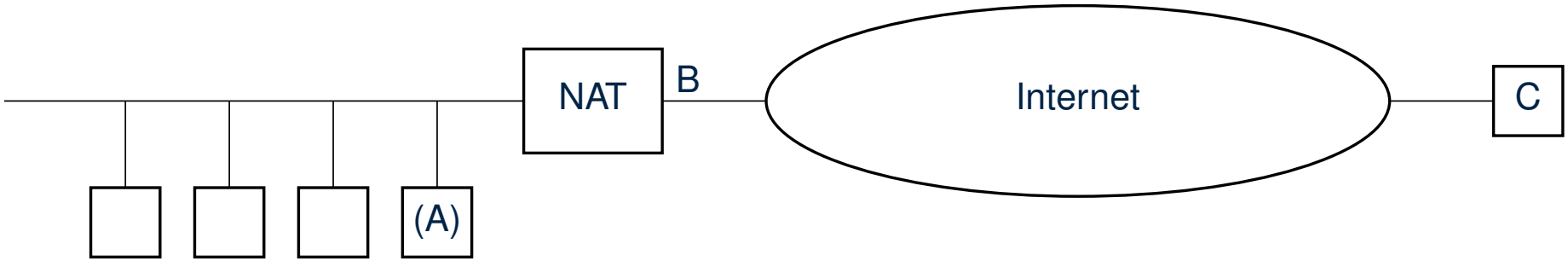
Imagen: High Performance Browser Networking: Building Blocks of UDP

5.9 Network Address Translation (III)

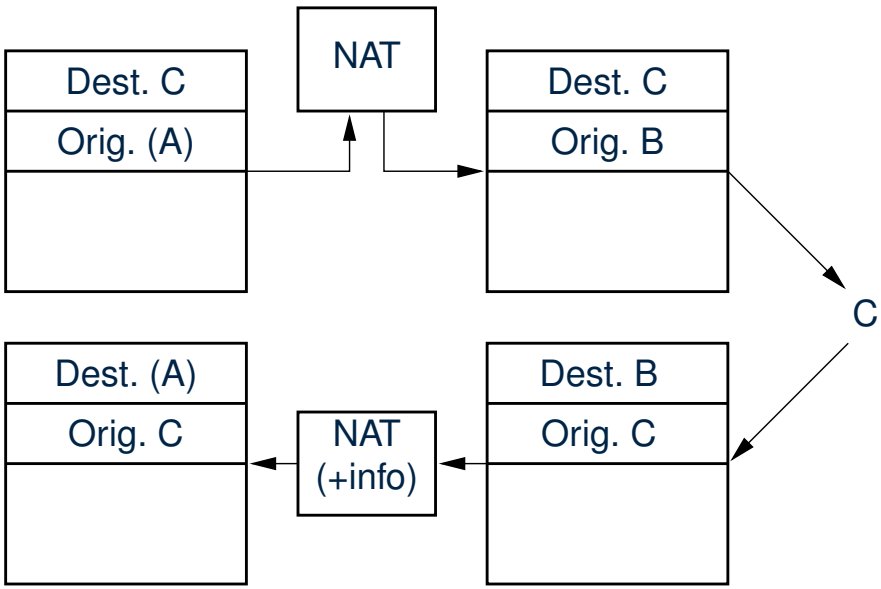
Ejemplo NAT444



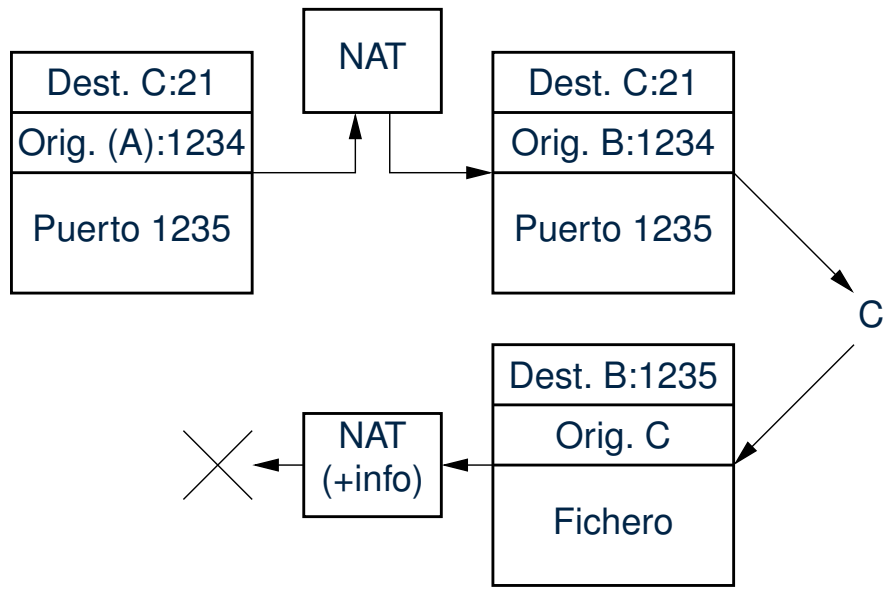
5.9 Network Address Translation (IV)



Ejemplo «funcional»

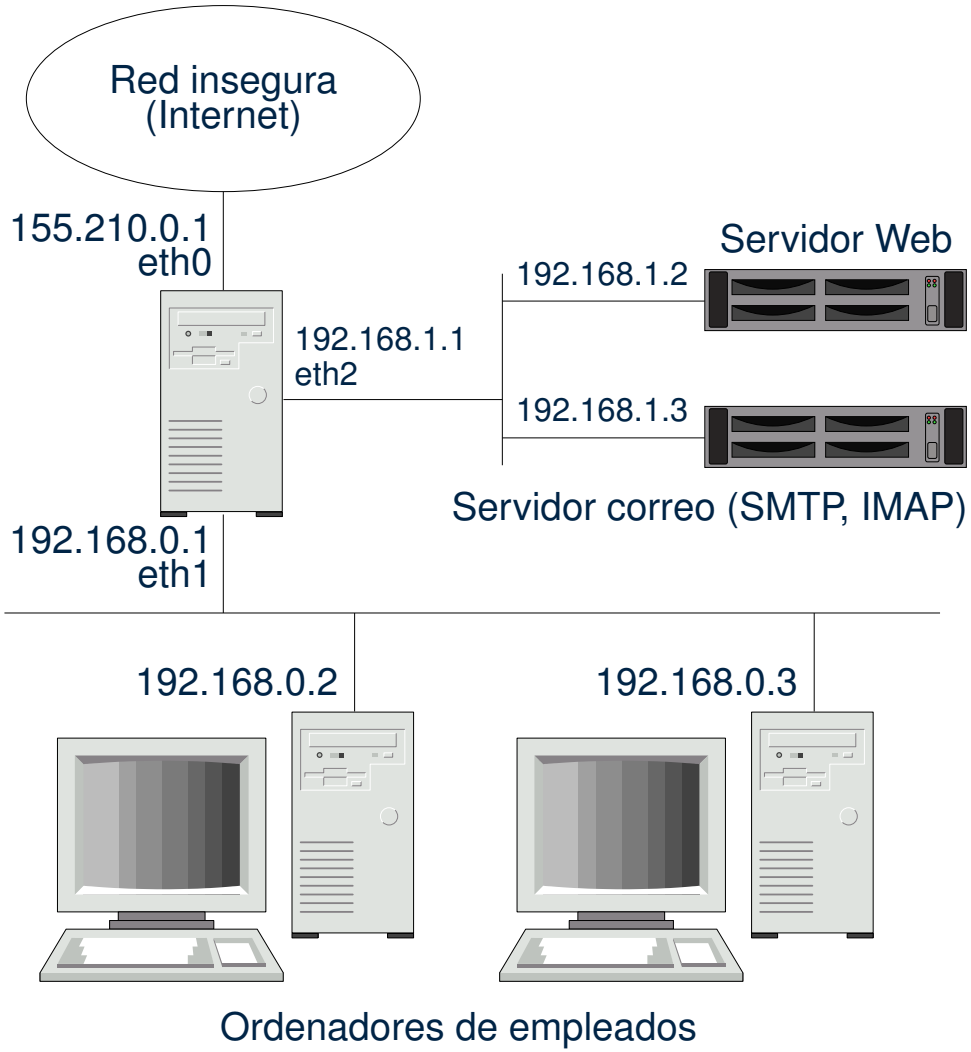


Ejemplo FTP «problemático»



5.10 Ejemplo cortafuegos + NAT

Ej. de manipulación de paquetes:



- Redirigir paquetes SMTP e IMAP al servidor de correo (NAT)
- Redirigir paquetes web, con un límite de 1 por segundo y ráfagas de 10 como máximo, al servidor web (NAT+filtro)
- Descartar todo el tráfico con protocolo distinto de TCP (filtro)
- Hacer de NAT a los ordenadores de empleados
- Inhabilitar el uso de la red en la propia máquina (filtro)
- Descartar conexiones desde 192.168.0.3 a puertos 6667 del exterior (filtro)

6 Índice

6 IP versión 6

- RFC 8200
- Propuesta inicial en 1991 (IETF IPng: IP Next Generation)
- Nuevo protocolo → requiere actualizar encaminadores
- Intención de transición progresiva, que se ha ido retrasando «forzando» IPv4
- Direcciones de 128 bits sin clases y jerárquicas (red + interfaz)
- Al no haber NAT es posible tener seguridad en capa de red
  - Implementación obligatoria de IPsec
- No hay fragmentación (ICMP notifica tamaño demasiado grande)
- Estadísticas adopción: https://www.google.com/intl/en/ipv6/statistics.html

6.1 Cabecera IPv6

Cabecera de tamaño fijo (40 bytes):

1 4	12	32	48	56	64
Ver.	TrafClas	Flow Label	Payload Length	NextHdr	HopLimit
Dirección origen					
Dirección destino					

- No hay checksum ni campos fragmentación ni opciones
- Campos ligeramente modificados: TOS $\rightarrow$ TrafficClass, Length $\rightarrow$ PayloadLength, Protocol $\rightarrow$ NextHeader, TTL $\rightarrow$ HopLimit
- Nuevo campo *FlowLabel*
- Opciones $\rightarrow$ Cabeceras de extensión (NextHeader)

6.1 Cabecera IPv6 (II)

Cabeceras de extensión (ejemplo):

IPv6 Header NextHeader= Security	*Security Header* NextHeader= Fragmentation	*Frag. Header* NextHeader= TCP	TCP Header	DATA

- Enlazadas con campos NextHeader hasta capa superior
- Procesadas por nodo destino (excepto extensión *Hop-by-Hop*)
- Permite añadir funcionalidad sin cambiar protocolo y sin modificar encaminadores
- E.g. salto a salto, fragmentación, autentificación, encapsulado de seguridad de la carga útil

6.2 Direcciones IPv6

- RFC 4291
- Direcciones de 128 bits
- 3 tipos:
  - Unicast: interfaz único
  - Anycast: conjunto de interfaces
  - Multicast: conjunto de interfaces
- Paquete enviado a dirección anycast/multicast se envía al interfaz más cercano/todos los interfaces del conjunto
- No hay direcciones broadcast
- Representación:
  - 8 números de 16 bits en formato hexadecimal
    - 2001:0DB8:0000:0000:0000:0000:A456:0024
  - 0 contiguos y a la izquierda se pueden omitir
    - 2001:DB8::A456:24
    - ”::”solamente puede aparecer una vez

6.2 Direcciones IPv6 (II)

- Tipos de direcciones:

Tipo	Prefijo	Notación IPv6
Sin especificar	00...0 (128 bits)	::/128
Bucle local (*Loopback*)	00...1 (128 bits)	::1/128
Multicast	11111111	FF00::/8
Link-Local unicast	1111111010	FE80::/10
Global Unicast	resto	

- Bloque de direcciones global unicast:

- Identificador de interfaz autoconfigurado a partir de MAC, asignado mediante DHCP, aleatorio (privacidad) o establecido manualmente

6.3 Ejercicio de direcciones

✍ Busca la configuración de los equipos de tu casa. Para cada uno de ellos (excepto el encaminador):

- ¿Cuál es su dirección IP? ¿Tiene IPv4, IPv6 o ambas?
- ¿Es pública o privada?
- ¿Está en alguna subred?
- ¿Está configurado manualmente o vía DHCP?

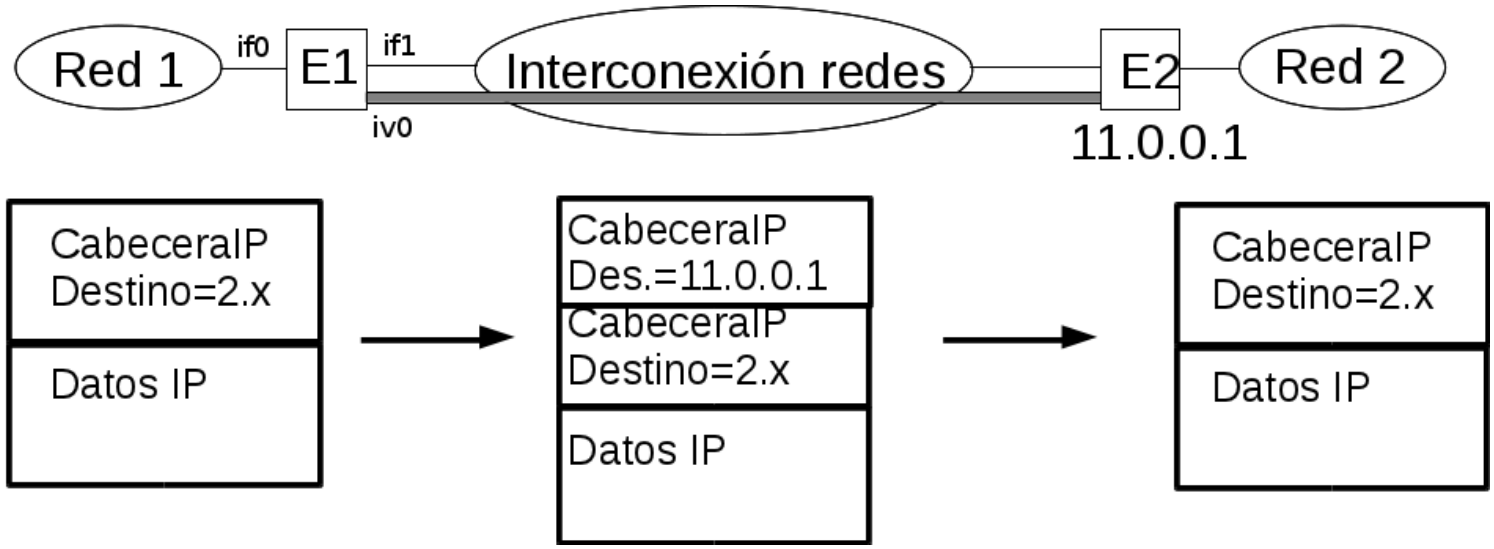
Para el encaminador:

- ¿Qué dirección IP tiene cada uno de sus interfaces?
- ¿Tiene IPv4, IPv6 o ambas?
- ¿Cuáles son públicas?
- ¿Es también un servidor DHCP?
- ¿Es también un NAT?

7 Índice

7 Túneles

- Encapsulación de paquetes de la capa de red dentro de paquetes de la capa de red



- Tabla encaminamiento E1:

Destino	Interfaz
Red 1	Interfaz 0
Red 2	Interfaz virtual 0
Por defecto	Interfaz 1

- Túnel+cifrado: VPN (Virtual Private Network)

7.1 Ejemplo VPN

https://remoto.unizar.es

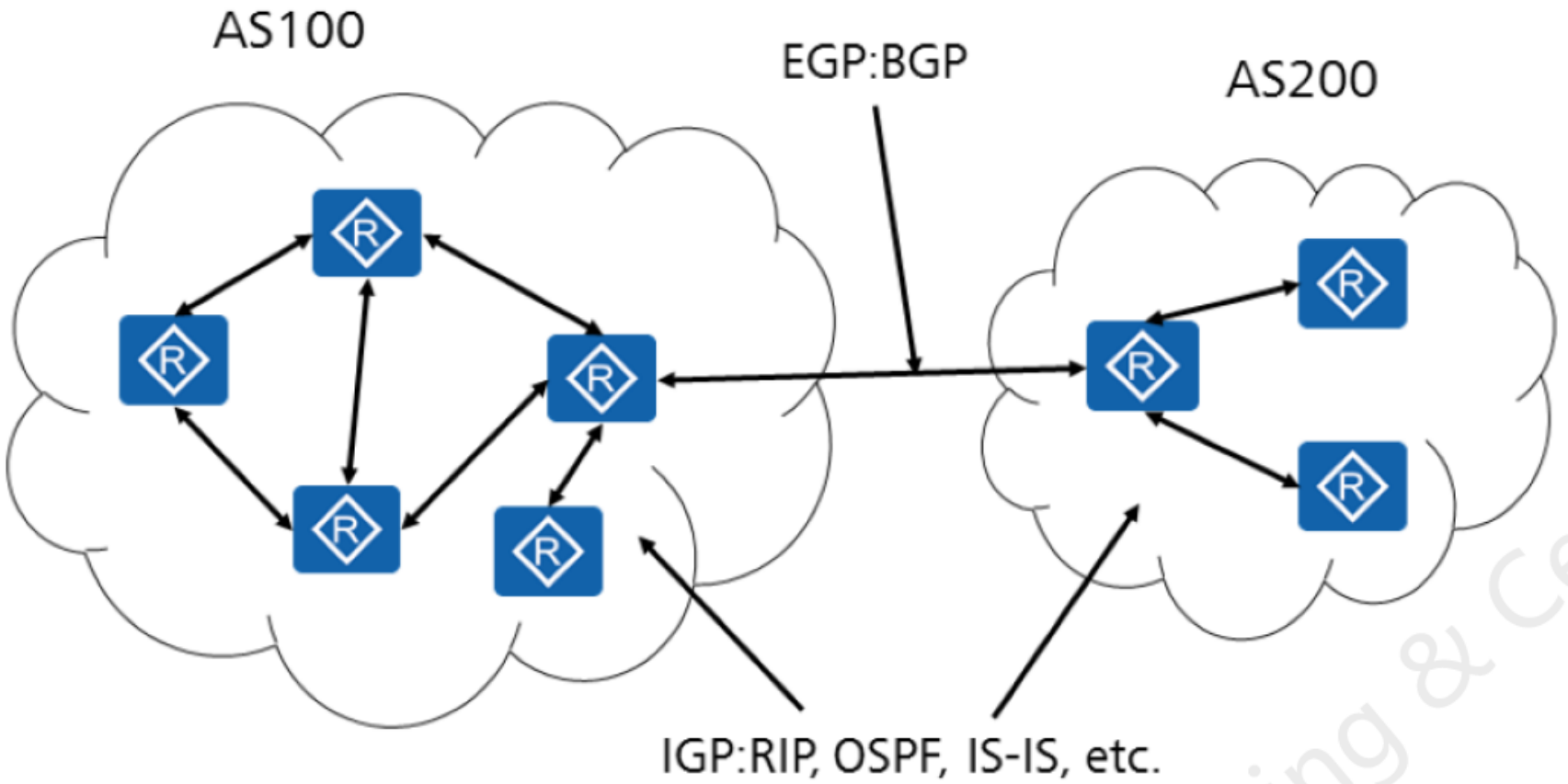
8 Índice

8 Protocolos de encaminamiento

- Encaminadores reciben paquetes y los reenvían hacia el destino
- Decisiones basadas en
  - Conocimiento topología de la red
  - Condiciones red (caídas, retardos ...)
- Encaminadores intercambian este tipo de información mediante protocolos de encaminamiento
- Objetivo: configuración automática de tablas de encaminamiento
- Ventajas: reacción y adaptación a cambios
  - Propagación rutas alternativas

8 Protocolos de encaminamiento (II)

- Dos tipos:
  - Interior: dentro de un Sistema Autónomo (SA)
  - Exterior: entre SAs



8.1 Sistemas autónomos

- Sistema Autónomo (SA) es un grupo de redes IP que poseen una política de rutas propia e independiente:
  - Un SA es un dominio administrativo independiente
  - Se asigna a cada SA un número de 32 bits (ASN)
  - Ejemplos: gran compañía, red columna vertebral
- *Gestión de caminos en dos niveles*: dentro/fuera de un SA
  - Dentro: optimizar caminos entre las redes que contiene
  - Fuera: buscar caminos que comuniquen los distintos SAs
- Ventajas por trabajar con subconjuntos de Internet:
  - *Escalabilidad*: que al crecer todo siga funcionando
  - *Eficiencia*: poco tráfico de configuración
  - *Tolerancia a fallos*: buscar rápido otras rutas en caso de fallo del enlace o de un encaminador

8.2 Protocolo encaminamiento interior

- Entre encaminadores dentro de un SA
- Todos los encaminadores del SA deben usar el mismo protocolo de búsqueda de caminos
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First

8.3 Protocolo encaminamiento exterior

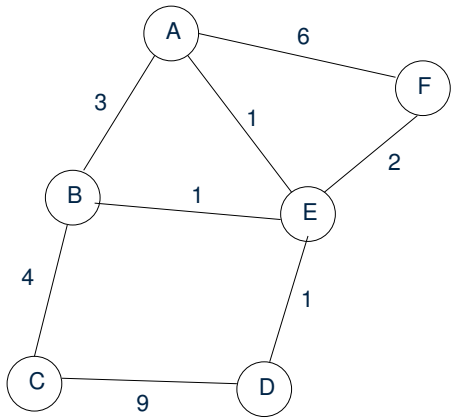
- Entre encaminadores de distintos SAs
- Cada SA tiene uno o más encaminadores frontera, que anuncian
  - Redes internas
  - Redes externas alcanzables (sólo en SA de tránsito)
  - Información de caminos
- Todos los SAs deben usar el mismo protocolo:
  - BGP-4 (*Border Gateway Protocol 4*), RFC 4271
- Elección de rutas basada en políticas explícitas (preferencias basadas en precio, etc.)
- Para poder atravesar ciertos SAs y evitar otros, BGP trabaja con rutas completas, no solo el siguiente salto

8.4 Algoritmos de encaminamiento

Objetivo: encontrar el camino con el menor coste entre dos nodos en una red dada. Se basan en:

- Topología: grafo donde cada red es un nodo
- Métricas de coste en enlaces
  - Coste constante: e.g. capacidad enlace, nº saltos, etc.
  - Factor dinámico «simple»: e.g. nº paquetes en cola
  - Factor dinámico dependiente de capacidad y carga (e.g. retardo medio de los últimos n minutos)
  - Factores económicos: e.g. precio por transmisión
  - Etc.

- Con esta información, los encaminadores construyen su tabla de encaminamiento



8.4 Algoritmos de encaminamiento (II)

Dos tipos:

- Distancia-vector
  - «Si el router X está a 5 saltos de la red Y, y yo soy adyacente a X, estoy a 6 saltos de la red Y»
  - Problemas de convergencia
  - RIP (LANs), IGRP (WANs pequeñas), EIGRP (WANs), BGP (Internet backbone)
- Estado-enlace
  - «Router X es adyacente al router Y por enlace activo»
  - Convergencia rápida
  - Requiere más memoria y CPU que distancia-vector
  - OSPF (LANs, WANs pequeñas)

8.5 Estado de enlace (link-state)

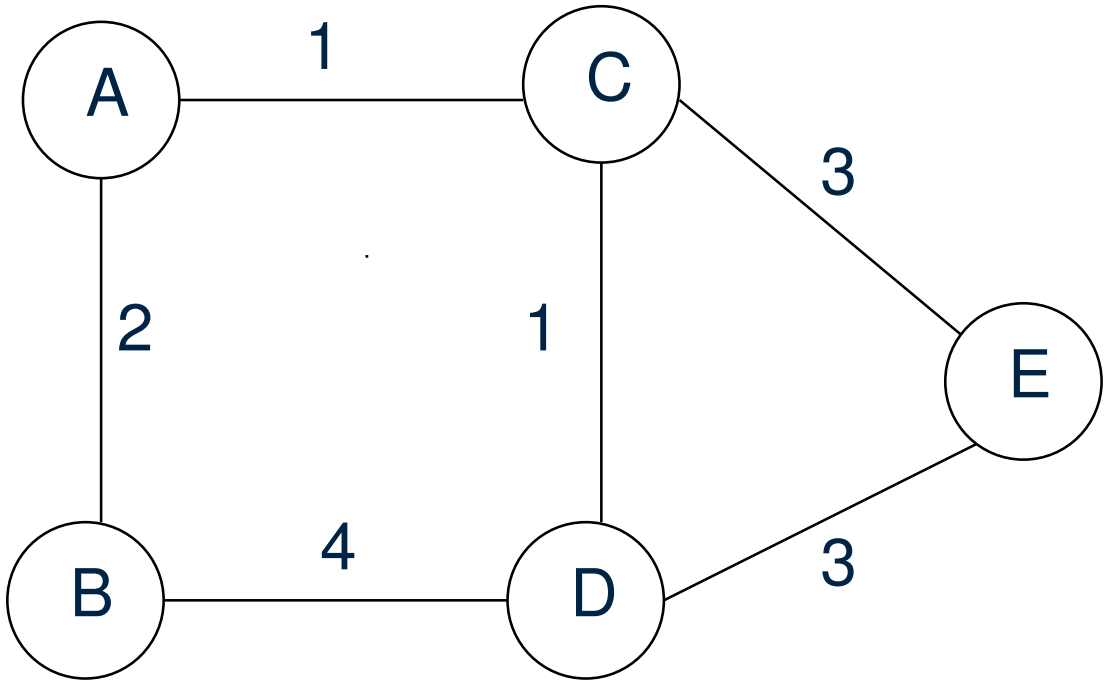
- Cada actualización contiene información sobre los *enlaces directamente conectados* (no toda la tabla de encam.)
- Envia actualización de estado a *todos los nodos* (no sólo a los vecinos inmediatos) mediante inundación fiable

Cada nodo aplica el algoritmo de *Dijkstra* (teoría de grafos) para calcular la mejor ruta hacia otros nodos:

1. Confirmar camino con distancia 0 al propio nodo
2. Crear lista vacía de posibles caminos y costes
3. Para el nuevo nodo confirmado:
   - 3.1 Añadir a la lista los nodos directamente conectados al nuevo nodo confirmado y el coste para llegar a ellos pasando por él
   - 3.2 Si hay varios caminos para llegar a un nodo, descartar los más costosos
   - 3.3 Confirmar el nodo con menor coste y quitarlo de la lista

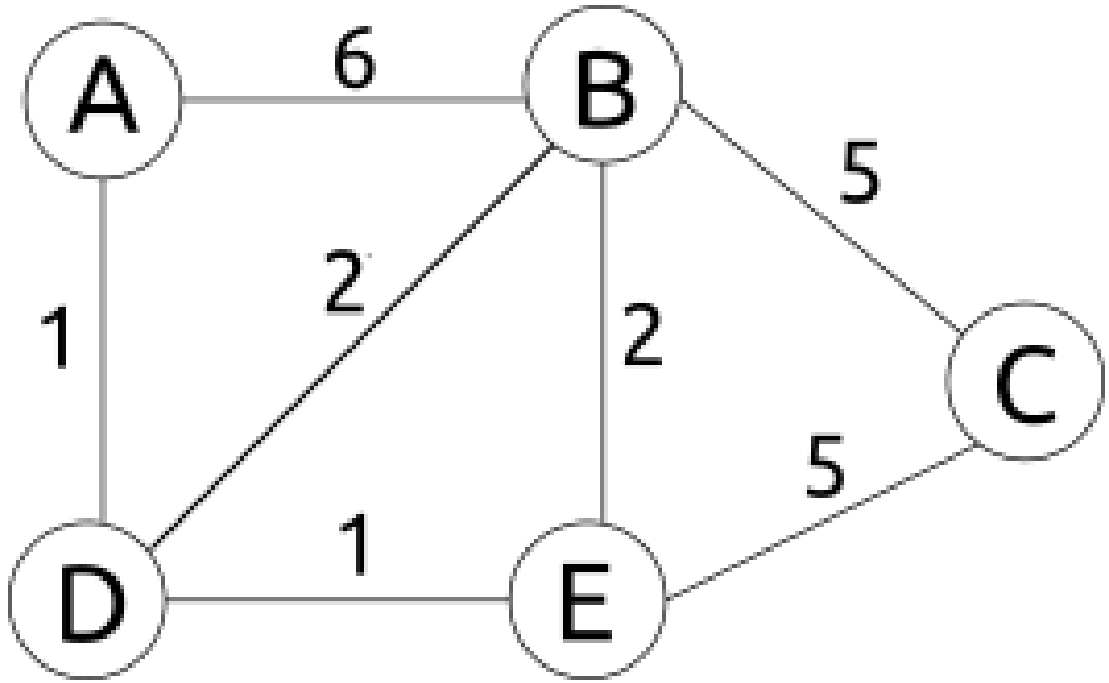
8.6 Ejemplo búsqueda de caminos

✍ Obtener la tabla de encaminamiento del nodo A mediante el algoritmo de estado de enlace

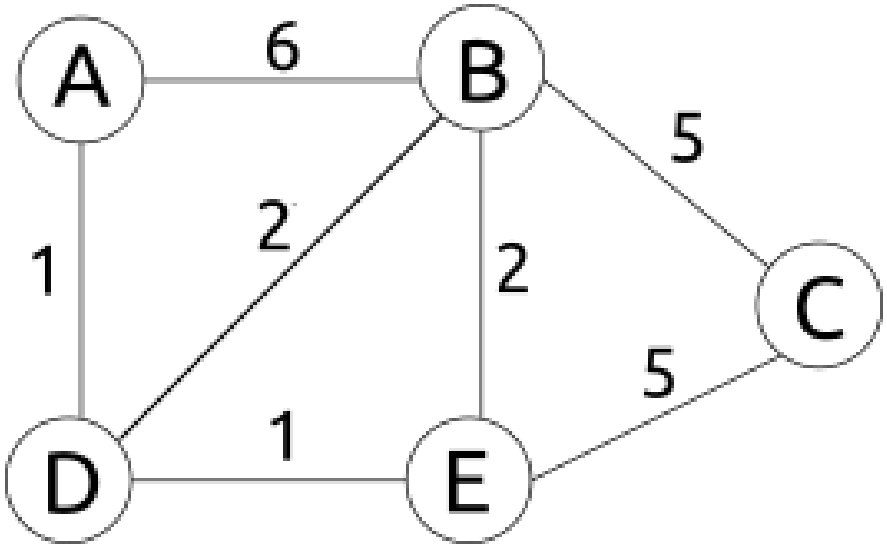


8.6 Ejemplo búsqueda de caminos

✍ Obtener la tabla de encaminamiento del nodo A mediante el algoritmo de estado de enlace

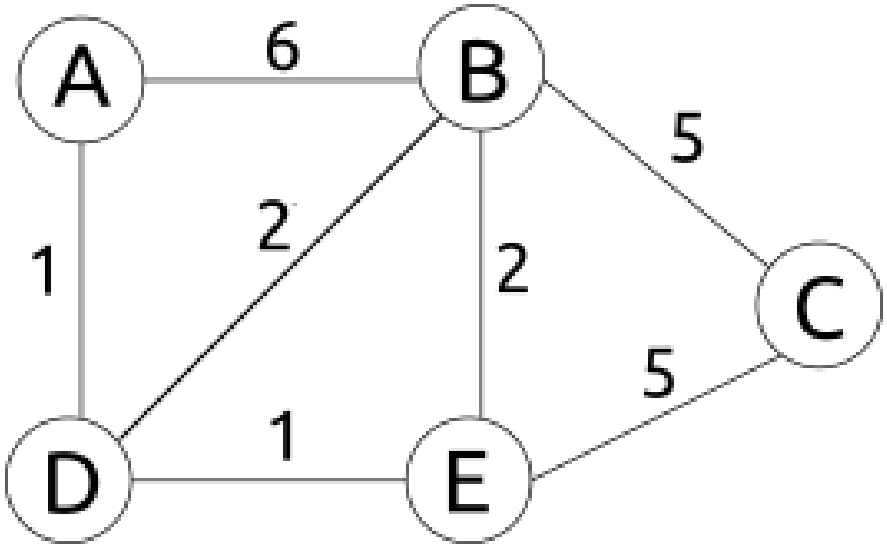


8.6 Ejemplo búsqueda de caminos



Vértice	Distancia	Último nodo
A		
B		
C		
D		
E		

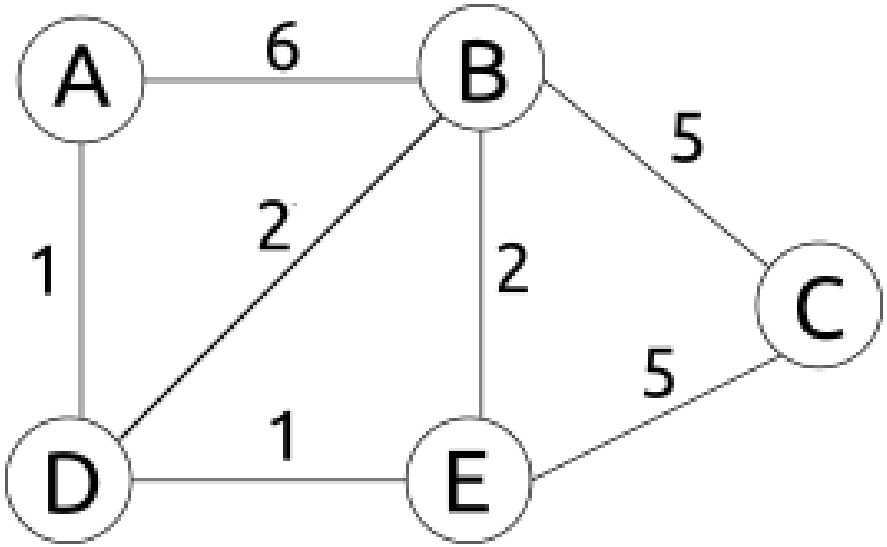
8.6 Ejemplo búsqueda de caminos



Vértice	Distancia	Último nodo
A	0	-
B	∞	
C	∞	
D	∞	
E	∞	

Visitados = [-], No visitados = [A,B,C,D,E]

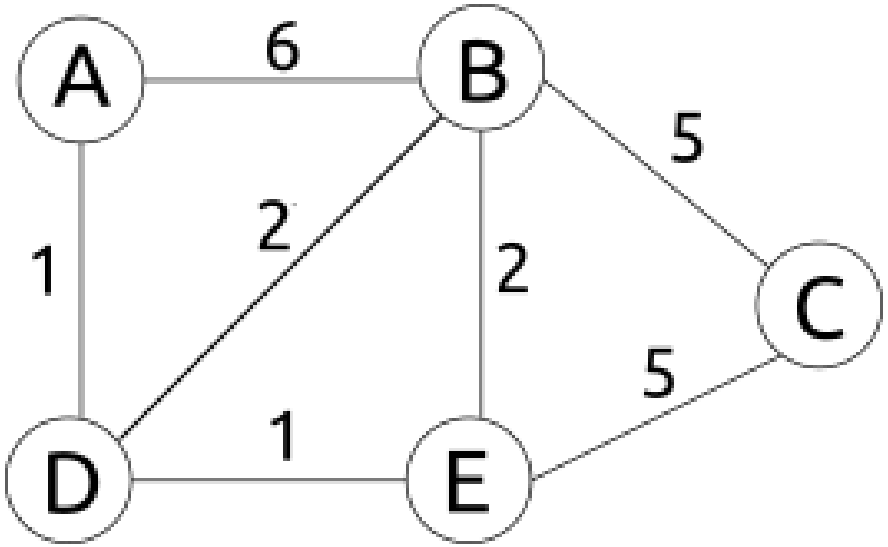
8.6 Ejemplo búsqueda de caminos



Vértice	Distancia	Último nodo
A	0	-
B	6	A
C	∞	
D	1	A
E	∞	

Visitados = [A], No visitados = [B,C,D,E]

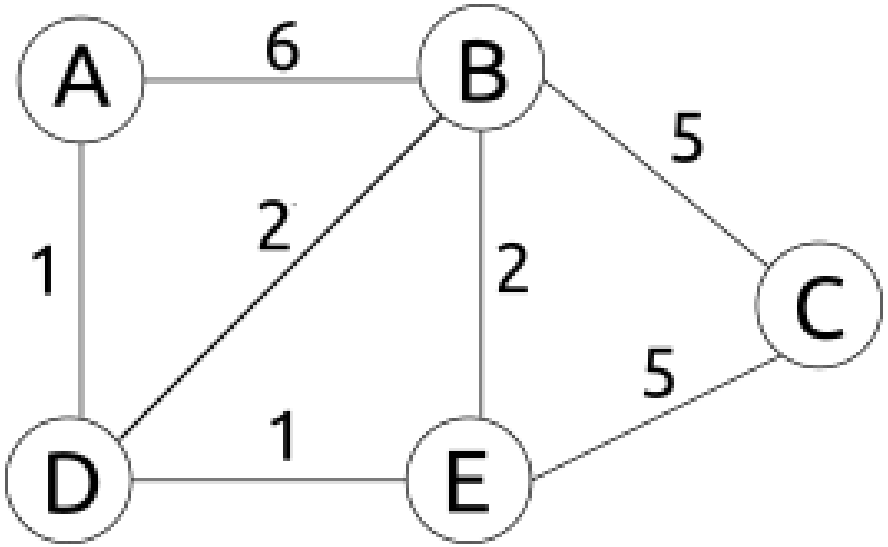
8.6 Ejemplo búsqueda de caminos



Vértice	Distancia	Último nodo
A	0	-
B	3	D
C	∞	
D	1	A
E	2	D

Visitados = [A,D], No visitados = [B,C,E]

8.6 Ejemplo búsqueda de caminos



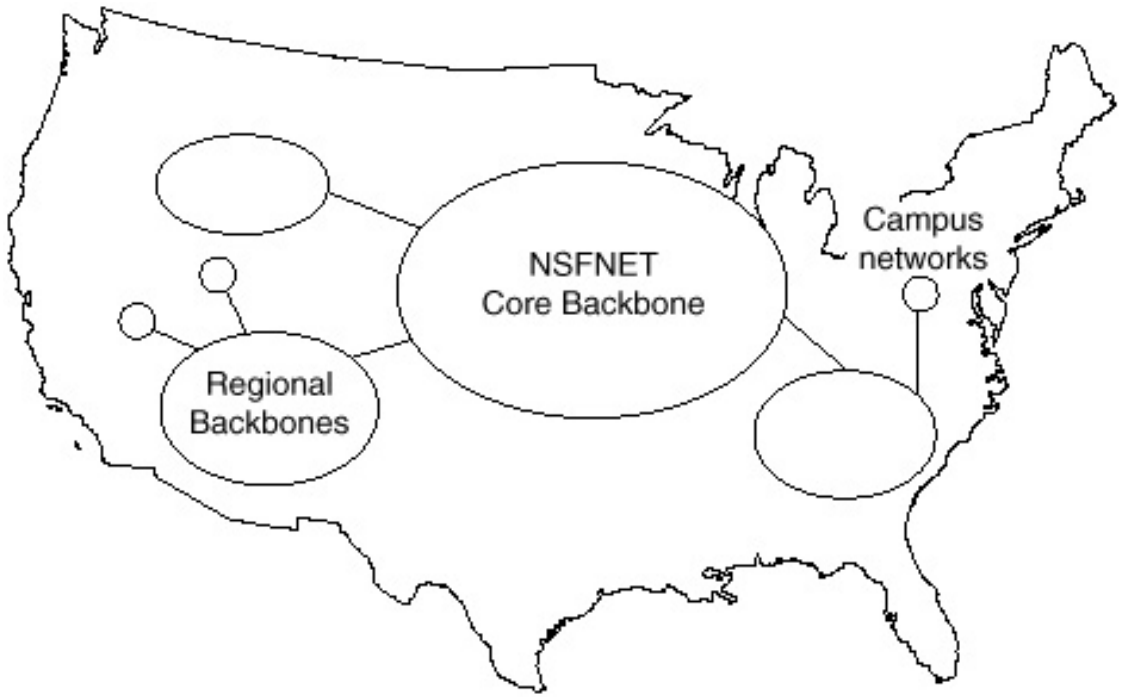
Vértice	Distancia	Último nodo
A	0	-
B	3	D
C	7	E
D	1	A
E	2	D

Visitados = [A,D,E], No visitados = [B,C]

9 Índice

9 Estructura de Internet

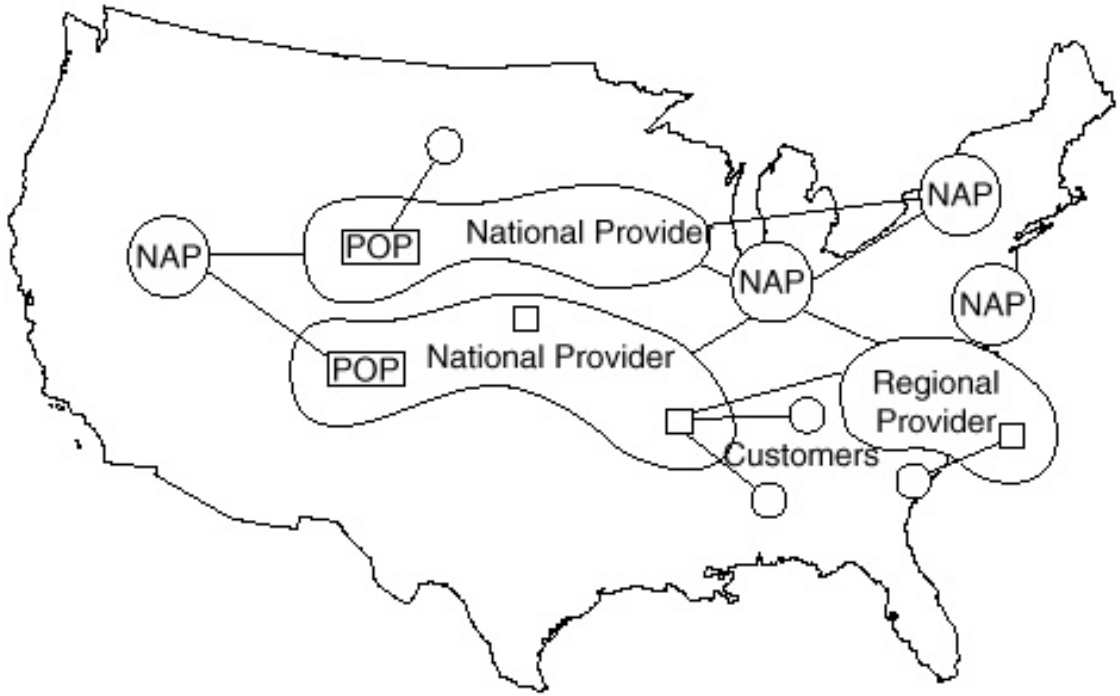
- Pasado reciente (1986-1995)



NSFNET

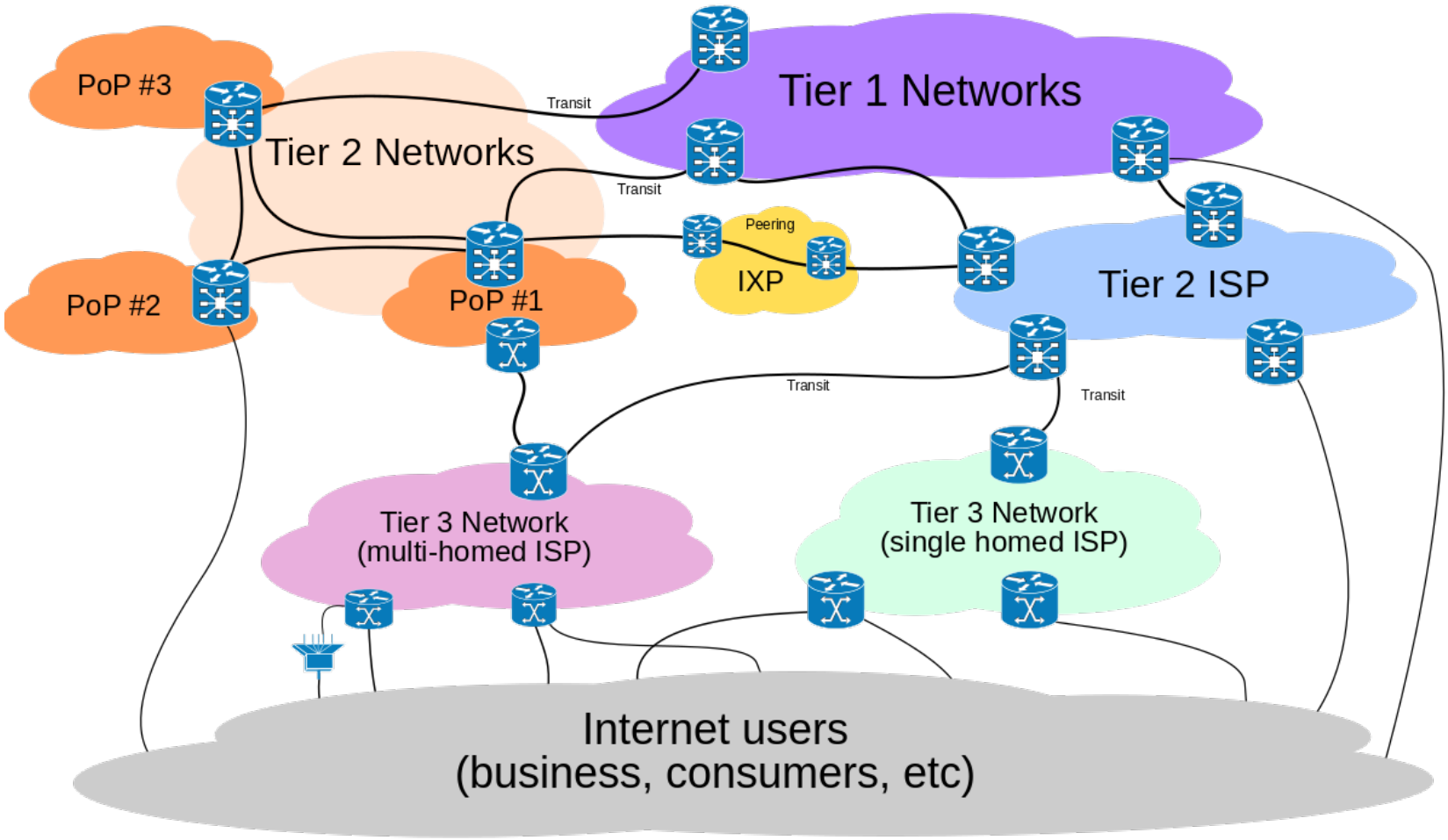
9 Estructura de Internet (II)

- Ahora es una estructura jerárquica operada por proveedores comerciales



9 Estructura de Internet (III)

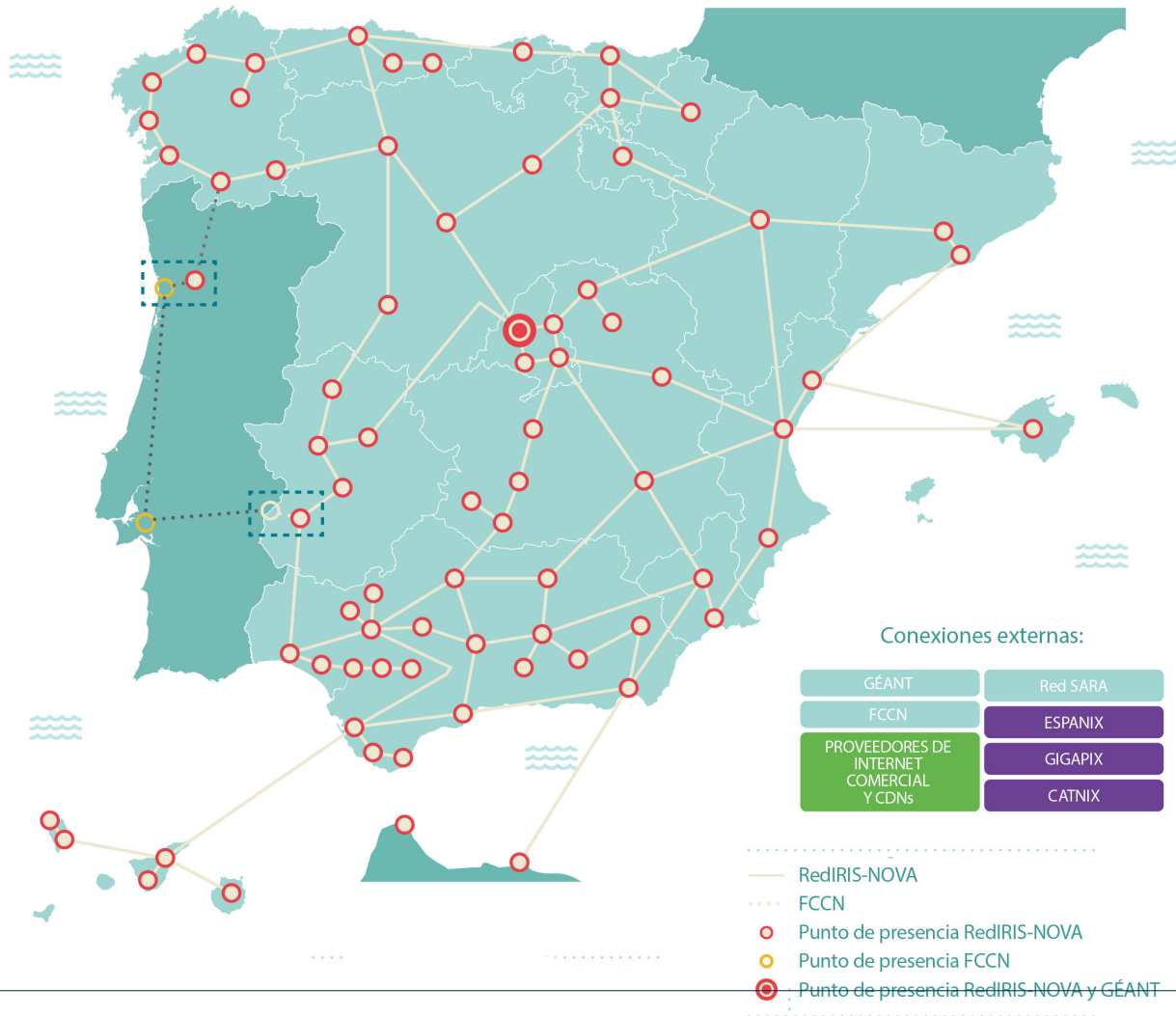
- ISP roles y relaciones



https://en.wikipedia.org/wiki/Internet_transit

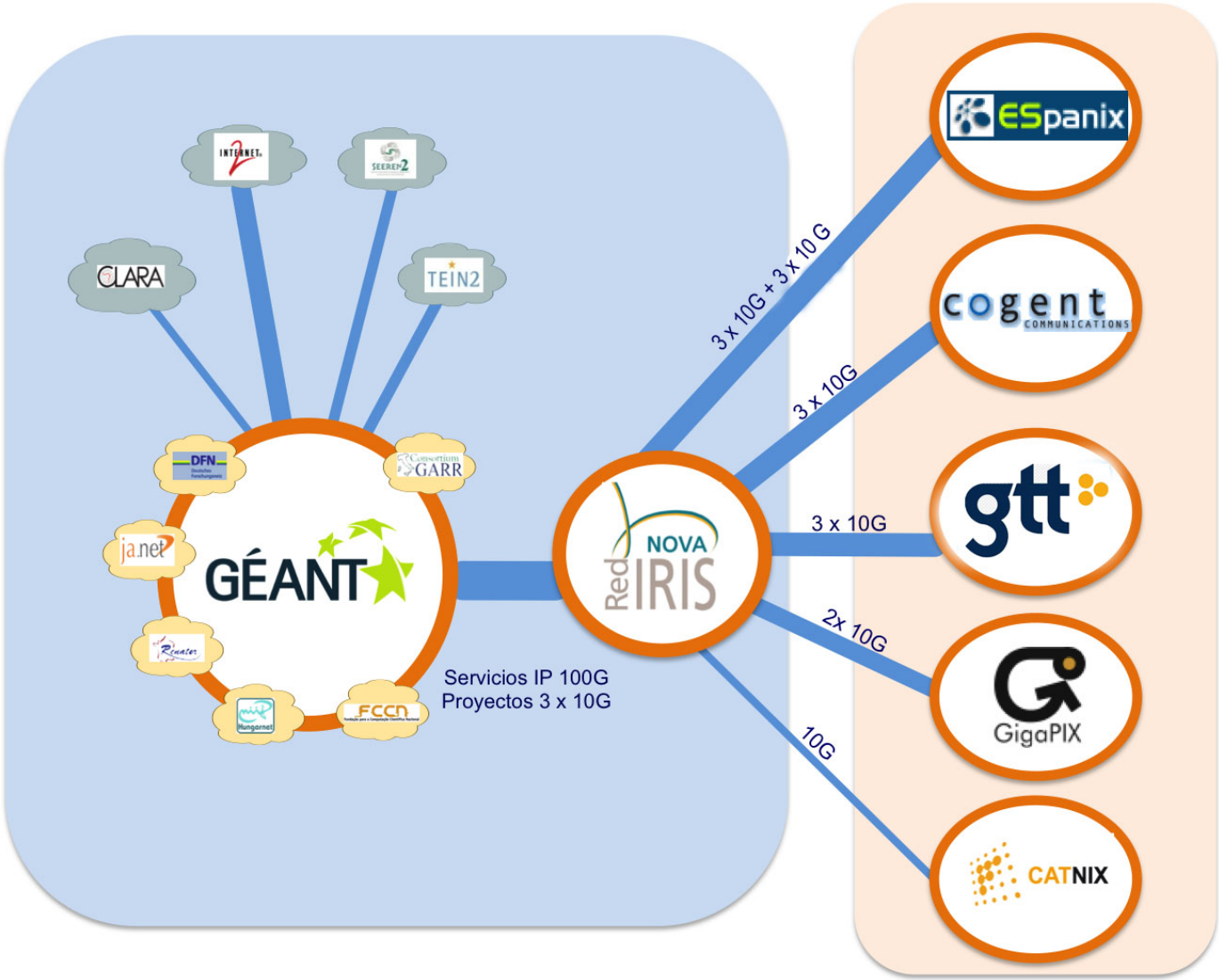
9.1 Sistemas autónomos

- E.g. RedIRIS (SA 766) agrupa las redes de las universidades y centros de investigación en España



9.1 Sistemas autónomos (II)

- Conectividad externa de RedIRIS



9.1 Sistemas autónomos (III)

- E.g. de SA troncal: GÈANT (SA 20965) comunica los SAs de investigación europeos