

---

## **Temas 24 y 25**

# **Protocolos de capas superiores: TCP/IP y ATM**

**Eytan Modiano**

**Instituto Tecnológico de Massachusetts**

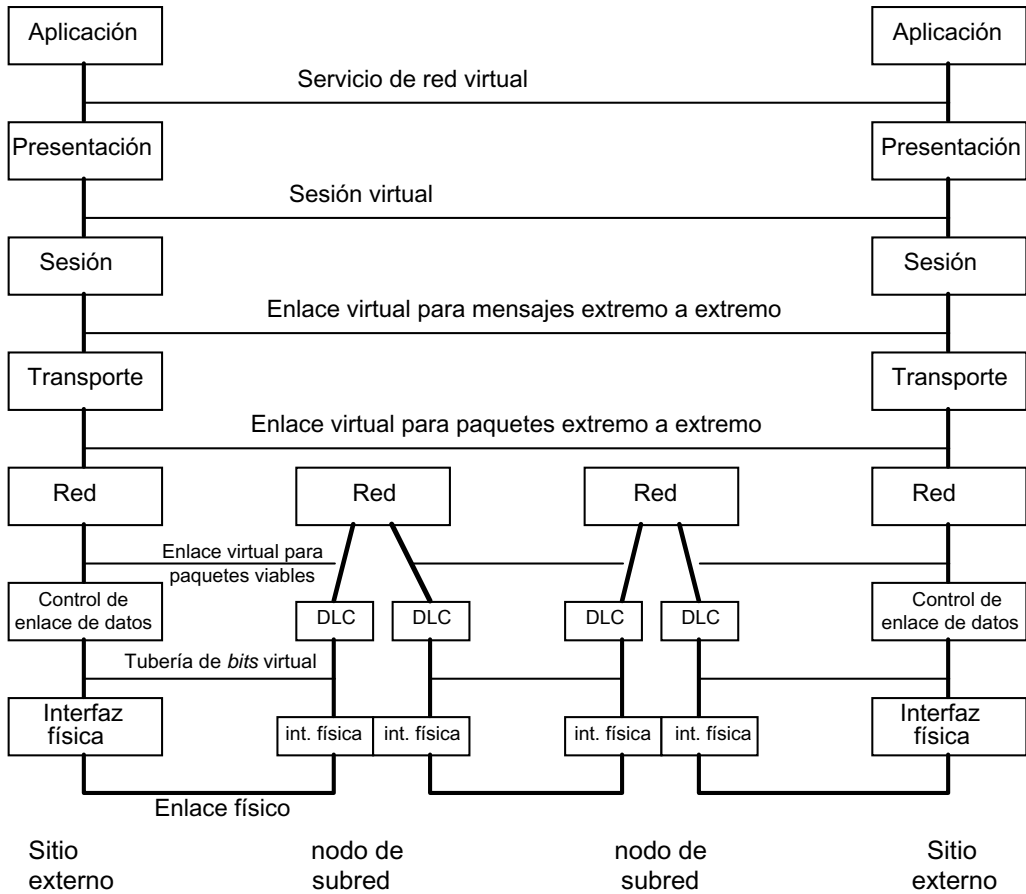
**Laboratorio de sistemas de decisión e información (LIDS)**

# Esquema

---

- **Interconexión de redes y la capa de red**
- **El protocolo TCP/IP**
- **ATM**
- **MPLS**

# Capas superiores



TCP, UDP

IP, ATM

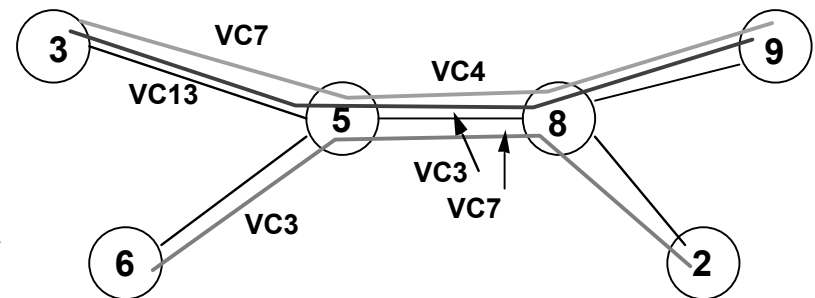
# Conmutación de paquetes

---

- **Conmutación de paquetes por datagramas:**
  - La ruta se elige según el método paquete por paquete
  - Los distintos paquetes pueden seguir rutas diferentes
  - Los paquetes pueden no llegar en orden a su destino
  - Ej.: IP (protocolo de Internet)
- **Conmutación de paquetes con establecimiento de circuito virtual (VC):**
  - Todos los paquetes asociados a una sesión siguen el mismo camino
  - La ruta se elige al inicio de la sesión
  - Los paquetes contienen un identificador de VC que indica la ruta
  - El identificador de VC debe ser único para un enlace dado, pero puede variar de un enlace a otro:
    - Supongamos que tenemos que establecer conexiones entre 1000 nodos de una red
    - El que los identificadores sean únicos supone que 1 millón de números VC han de ser representados y almacenados en cada nodo
  - Ej.: ATM (*Asynchronous transfer mode* o modo de transferencia asíncrona)

# Conmutación de paquetes con establecimiento de VC

- En los datagramas, la información de destino debe distinguir de forma inequívoca cada sesión y nodo de la red:
  - Son necesarias unas direcciones de origen y de destino únicas
- En los circuitos virtuales, sólo es necesario distinguir por medio de la dirección entre los circuitos virtuales de un enlace:
  - Es necesaria una dirección global para establecer un circuito virtual
  - Una vez establecido, se pueden utilizar los números VC locales para representar los circuitos virtuales de un enlace dado: el identificador de VC varía de un enlace a otro



- Méritos de los circuitos virtuales:
  - Ahorro en el cálculo de la ruta:  
Sólo es necesario calcularla una vez, al inicio de la sesión
  - Ahorro en el tamaño de la cabecera
  - Más complejo
  - Menos flexible

Tabla del nodo 5

(3,5) VC13	->	(5,8) VC3
(3,5) VC7	->	(5,8) VC4
(6,5) VC3	->	(5,8) VC7

# El protocolo TCP/IP

---

- **Protocolo de control de transmisión / Protocolo de Internet**
- **Desarrollado por DARPA con el fin de conectar entre sí las universidades y los laboratorios de investigación:**

## Modelo de cuatro capas

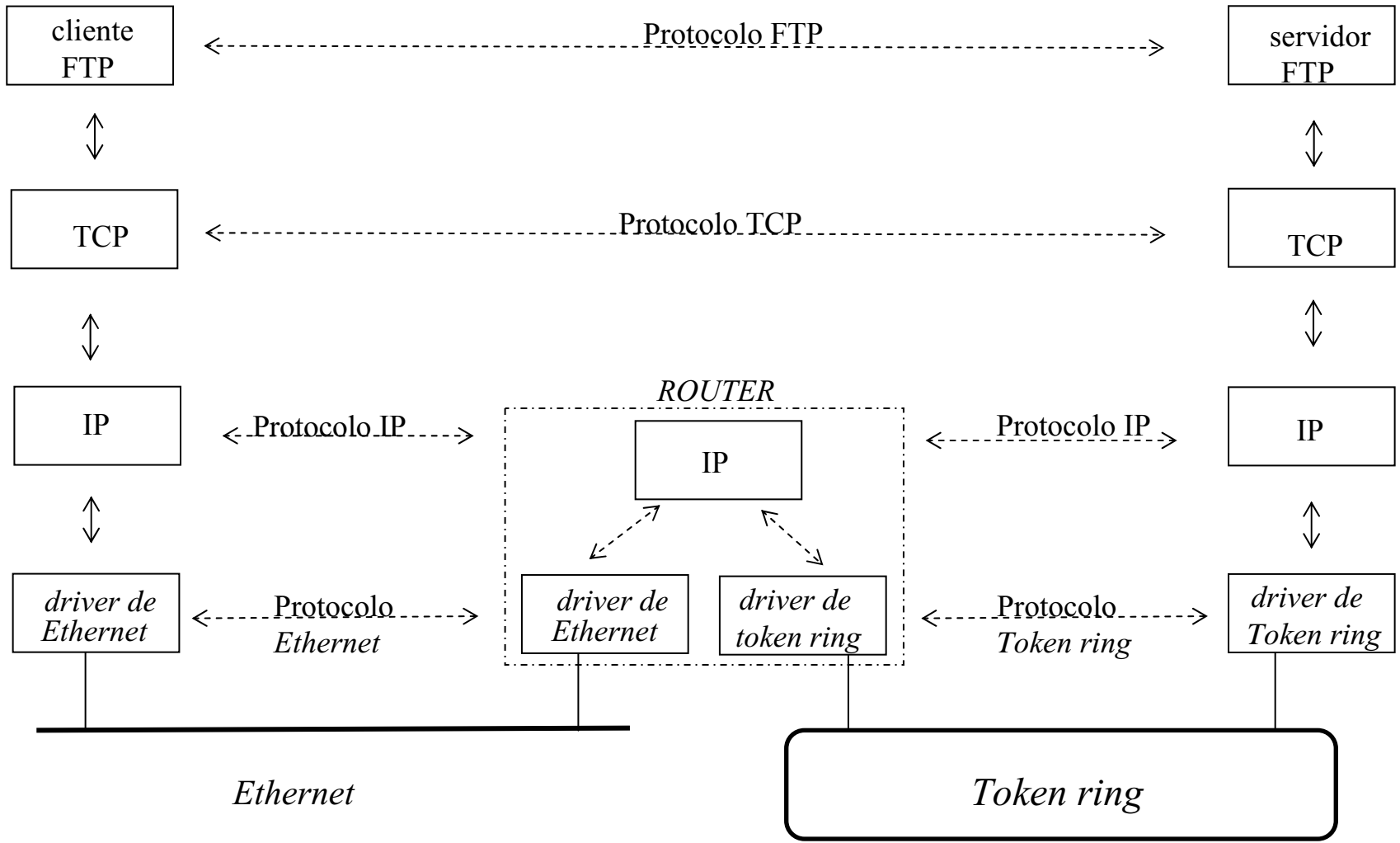
<b>Aplicación</b>	<b>Telnet, FTP, correo, etc.</b>
<b>Transporte</b>	<b>TCP, UDP</b>
<b>Red</b>	<b>IP, ICMP, IGMP</b>
<b>Enlace</b>	<b><i>Drivers</i>, tarjetas de interfaz</b>

**TCP - Protocolo de control de transmisión**

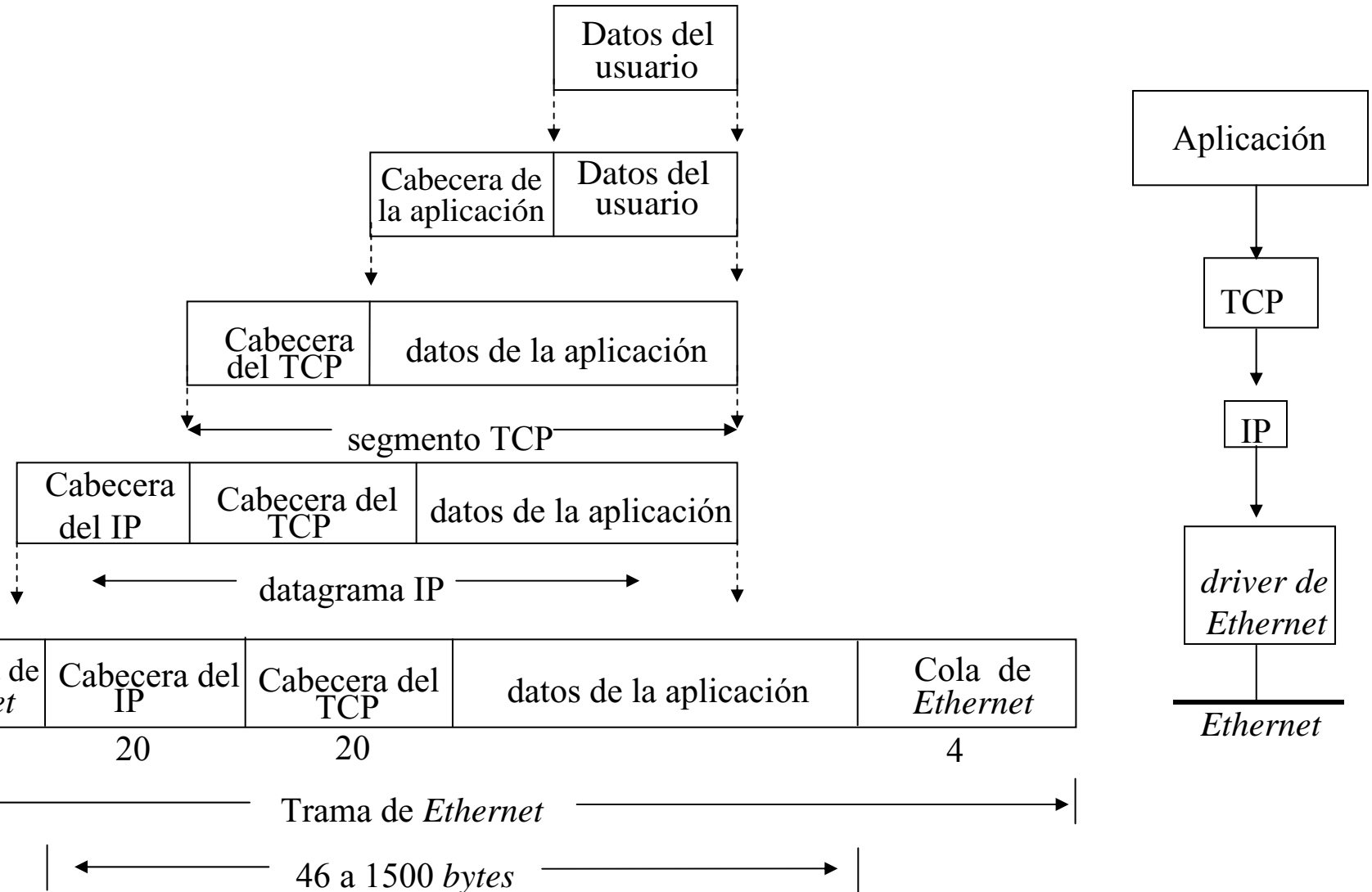
**UDP - Protocolo de datagramas de usuario**

**IP - Protocolo de Internet**

# Interconexión de redes con TCP/IP



# Encapsulación

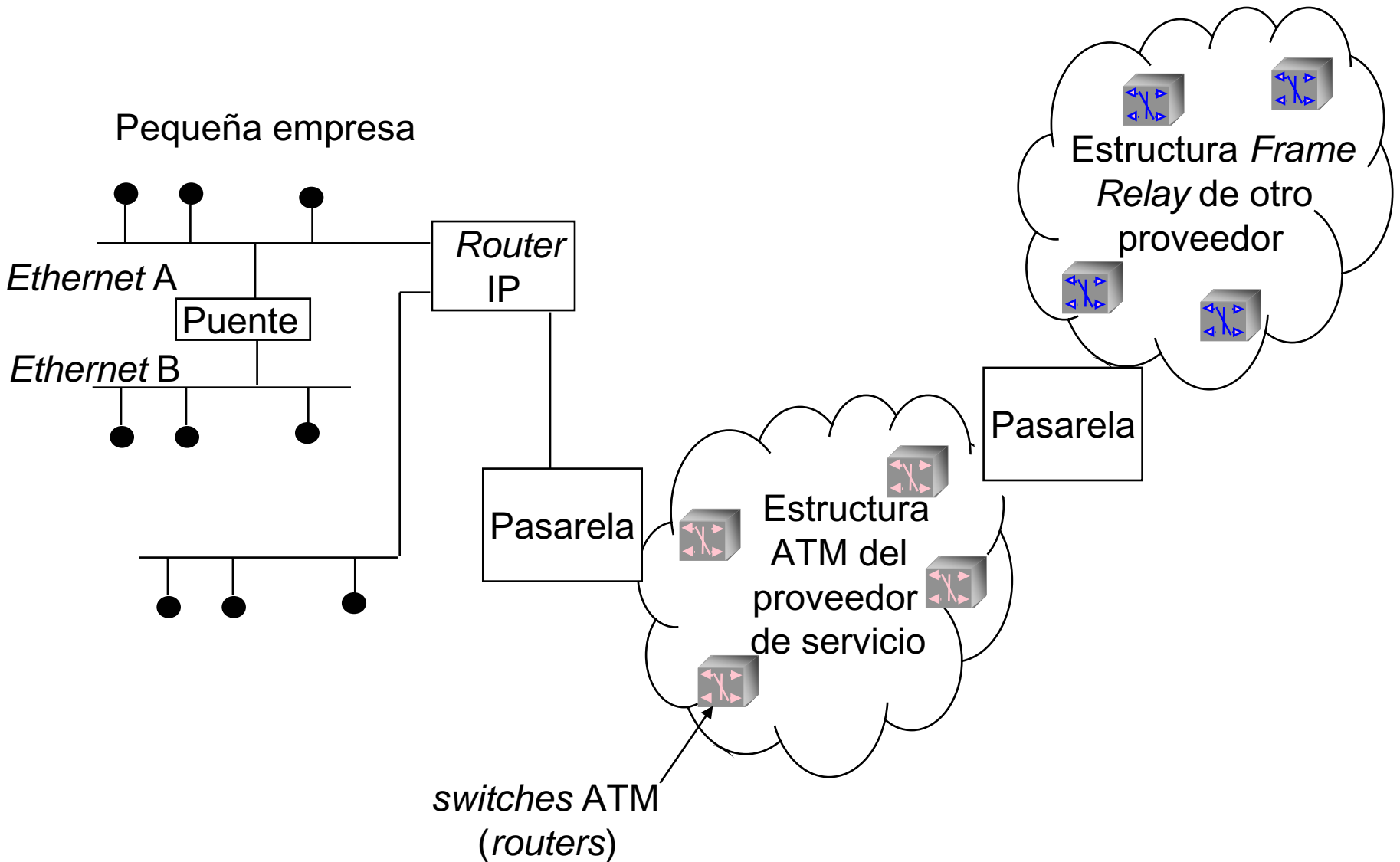


# Puentes, *routers* y pasarelas

---

- **Un puente (*bridge*) se utiliza para conectar múltiples segmentos de redes LAN:**
  - Enrutamiento de la capa 2 (*Ethernet*)
  - No conoce la dirección IP
  - Distintos niveles de sofisticación:
    - Los puentes sencillos simplemente reenvían los paquetes
    - Los puentes inteligentes empiezan a parecerse a los *routers*
- **Un *router* se utiliza para establecer la conexión entre diferentes redes mediante la dirección de la capa de red:**
  - Dentro de sistemas autónomos o entre ellos
  - Con el mismo protocolo (ej.: IP o ATM)
- **Una pasarela es una conexión entre redes que utilizan distintos protocolos:**
  - Realiza la conversión de protocolos
  - Resuelve la dirección
- **Estas definiciones a menudo son mixtas y parecen evolucionar**

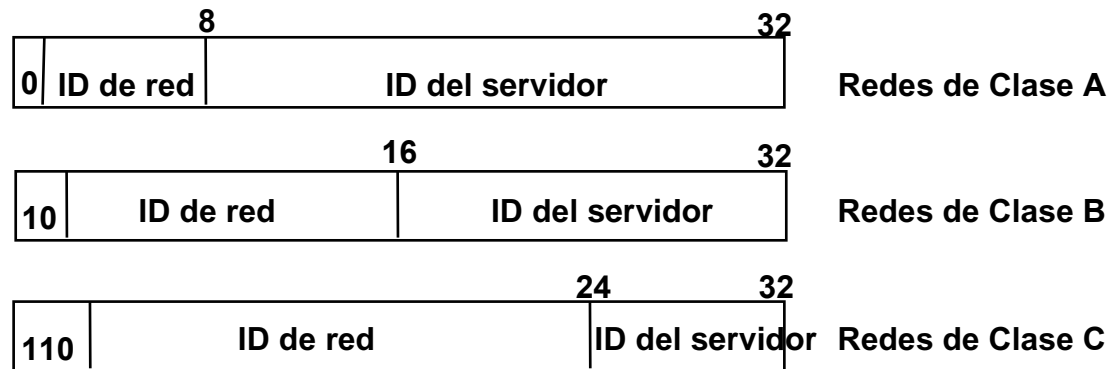
# Puentes, *routers* y pasarelas



# Direcciones IP

---

- Las direcciones de 32 *bits* se escriben con cuatro números decimales
  - Uno por cada *byte* de la dirección (ej.: 155.34.60.112)
- Estructura de la dirección jerárquica:
  - ID de red / ID del servidor (*host*) / ID del puerto
  - La dirección completa se denomina zócalo (*socket*)
  - El ID de red y el del servidor van incorporados en la cabecera IP
  - El ID del puerto (proceso de envío) va incorporado en la cabecera TCP
- Clases de direcciones IP:



La clase D es para el tráfico multidifusión (*multicast*)

# Nombres de servidores

---

- Cada máquina tiene también un nombre único
- Sistema de los nombres de dominio: base de datos distribuida que proporciona un servicio de mapeo entre las direcciones IP y los nombres de los servidores
- Ej.: 155.34.50.112 => plymouth.ll.mit.edu

# Estándares de Internet

---

- **La IETF (*Internet Engineering Task Force*):**
  - Desarrolla los estándares de Internet a corto plazo
  - Es un grupo abierto
  - Se reúne 3 veces al año
- **Los RFC (*Request for Comments*):**
  - Estándares oficiales de Internet
  - Disponibles en la página web de la IETF: <http://www.ietf.org>

# El protocolo de Internet (IP)

---

- **Enrutamiento de paquetes a través de la red**
- **Servicio poco fiable:**
  - Entrega "Best effort"
  - Los paquetes perdidos se deben recuperar en las capas superiores
- **Sin conexión:**
  - Los paquetes se distribuyen (enrutan) de forma independiente
  - Se pueden entregar desordenados
  - La secuencia se reestablece en las capas superiores
- **La versión actual es la V4**
- **Futura versión V6:**
  - Añade más direcciones (¡cabecera de 40 bytes!)
  - Capacidad para ofrecer QoS

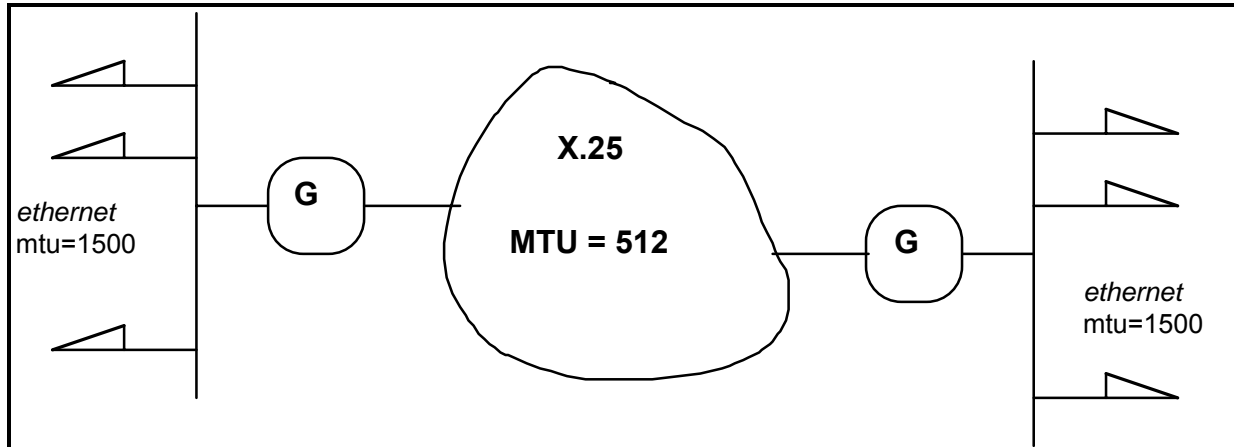


# CAMPOS DE LA CABECERA IP

---

- **Versión:** Número de la versión IP (la versión actual es la 4)
- **IHL:** Longitud de la cabecera en palabras de 32 *bits*
- **Tipo de servicio:** Ignorado en la mayoría de los casos
- **Longitud total** Longitud del datagrama IP
- **ID** ID de datagrama único
- **Banderas (*Flags*):** No fragmentar, Más fragmentos
- **Desplazamiento:** *Offset* del fragmento en unidades de 8 octetos
- **TTL:** Tiempo de vida en "segundos" o saltos (*hops*)
- **Protocolo:** N° ID del protocolo de capa superior
- ***Checksum:*** Comprobación de la suma de los complementos a 1 de 16 *bits* (sólo en la cabecera)
- **SA y DA:** Direcciones de red
- **Opciones:** *Record Route, Source Route y TimeStamp*

# FRAGMENTACIÓN



- Una pasarela fragmenta un datagrama si la longitud del mismo es demasiada para la siguiente red (la fragmentación es necesaria debido a las rutas desconocidas)
- Cada fragmento necesita un identificador único del datagrama más un identificador que indique su posición dentro de él.
- En IP, el ID del datagrama es un campo de 16 *bits* que contabiliza el datagrama desde el servidor dado

# POSICIÓN DEL FRAGMENTO

---

- El campo desplazamiento indica la posición de inicio del fragmento dentro del datagrama en incrementos de 8 *bytes* (campo de 13 *bits*)
- El campo longitud de la cabecera indica la longitud total en *bytes* (campo de 16 *bits*)
  - El tamaño máximo del paquete IP es de 64 Kb
- Un *bit* marcador (*flag*) señala el último fragmento del datagrama
- IP reorganiza los fragmentos una vez que llegan a su destino y los elimina si uno o más de ellos llega demasiado tarde

# Enrutamiento IP

---

- La tabla de enrutamiento de cada nodo contiene para cada destino el siguiente *router* al que se debe enviar el paquete:
  - No todas las direcciones de destino están en la tabla de enrutamiento:
    - Busca el ID de red para la coincidencia del prefijo de destino (“Prefix match”)
    - Utiliza el *router* establecido por defecto
- Los *routers* no conocen la ruta completa hasta el destino, sino tan sólo el salto al siguiente *router*
- IP utiliza algoritmos de enrutamiento distribuidos: RIP o OSPF
- En una LAN, el servidor envía el paquete al *router* establecido por defecto, que a su vez, proporciona una pasarela al mundo exterior

# Asignación de direcciones de subred

---

- Las direcciones de clase A y B asignan demasiados servidores a una red dada
- La asignación de direcciones de subred nos permite dividir el espacio de ID del servidor en “subredes” de menor tamaño:
  - Simplifica el enrutamiento dentro de una organización
  - Tablas de enrutamiento más pequeñas
  - Potencialmente, permite la asignación de la misma dirección de clase B a más de una organización
- La máscara de subred de 32 *bits* se utiliza para dividir el campo ID del servidor en subredes:
  - “1” indica el campo de la dirección de red
  - “0” indica el campo ID de un servidor

	ID de red de 16 <i>bits</i>	ID del servidor de 16 <i>bits</i>	
Dirección de clase B	140.252	ID de subred	ID del servidor
Máscara	111111 111 11111111	11111111	00000000

# Enrutamiento entre dominios sin clase (CIDR)

---

- Las direcciones de clase A y B asignan demasiados servidores a una organización, mientras que las direcciones de clase C no le asignan los suficientes:
  - Esto conlleva una asignación ineficaz del espacio de direcciones
- El enrutamiento sin clase permite la asignación de direcciones sin los límites de las clases (dentro de la gama de direcciones de clase C):
  - Asignar un bloque de direcciones contiguas:
    - Ej.: 192.4.16.1 - 192.4.32.155
    - Poner desordenadamente 16 direcciones de clase C
    - Los primeros 20 *bits* del campo dirección son los mismos y constituyen, básicamente, el ID de red
  - A continuación, hay que describir los números de red con su longitud y valor (es decir, la longitud del prefijo de red)
  - Consultar la tabla de enrutamiento utilizando la coincidencia del prefijo más largo
- Obsérvese la similitud entre la asignación de subred y la de “superred”

# Configuración dinámica del servidor (DHCP)

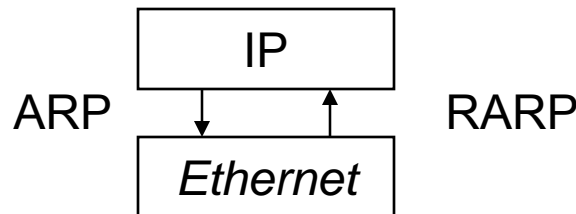
---

- **Método automatizado de asignación de números de red:**
  - Direcciones IP o *routers* por defecto
- **Los ordenadores se ponen en contacto con el servidor DHCP al iniciar**
- **El servidor les asigna una dirección IP**
- **Permite compartir el espacio de dirección:**
  - Uso más eficiente del espacio de dirección
  - Añade escalabilidad
- **El número de direcciones es “menor” durante un tiempo**
  - No se asignan permanentemente

# Protocolo de resolución de direcciones

---

- Las direcciones IP sólo tienen sentido en el entorno IP
- Las redes de área local, como *Ethernet*, tienen su propio esquema de direccionamiento:
  - Para hablar con un nodo de una LAN es necesario tener su dirección física (las tarjetas de interfaz física no reconocen sus direcciones IP)
- ARP proporciona un mapeo entre las direcciones IP y las direcciones LAN
- RARP proporciona un mapeo de direcciones LAN a direcciones IP
- Para ello se envía un paquete “broadcast” (de amplia difusión) pidiendo al propietario de la dirección IP que responda con su dirección física:
  - Todos los nodos de la LAN reconocen el mensaje *broadcast*
  - El propietario de la dirección IP responde con su dirección física
- En cada nodo se mantiene una caché ARP con los últimos mapeos



# Enrutamiento en Internet

---

- **Internet está dividida en subredes, cada una de ellas bajo el control de una sola autoridad, conocida como sistema autónomo (AS)**
- **Los algoritmos de enrutamiento se dividen en dos categorías:**
  - **Protocolos interiores (operan dentro de un AS)**
  - **Protocolos exteriores (operan entre sistemas autónomos)**
- **Los protocolos interiores utilizan algoritmos del camino más corto:**
  - **Protocolos del vector distancia basados en el algoritmo de Bellman-Ford:**  
Los nodos se intercambian las tablas de enrutamiento entre ellos  
Ej.: Protocolo de información de enrutamiento (RIP)
  - **Protocolos de estado de los enlaces basados en el algoritmo de Dijkstra:**  
Los nodos monitorizan el estado de sus enlaces (ej.: retardo)  
Los nodos difunden ampliamente esta información por toda la red  
Ej.: OSPF (*Open Shortest Path First*)
- **Los protocolos exteriores enrutan los paquetes a través de los AS**
  - **Problemas: no hay una única métrica de coste, política de enrutamiento, etc.**
  - **Las rutas se suelen calcular por adelantado**
  - **Protocolos de ejemplo: Protocolo de pasarela exterior (EGP) y Protocolo de pasarela de borde (BGP)**



# Reserva de recursos (RSVP)

---

- **Clases de servicio (definidas por la IETF):**
  - *Best effort*
  - **Servicio garantizado**
    - Retardo máximo de paquete
  - **Carga controlada**
    - Emula una red ligeramente cargada con un mecanismo de colas de prioridad
- **Es necesario reservar recursos en los *routers* distribuidos a lo largo de la ruta**
- **Mecanismo RSVP:**
  - **Clasificación de paquetes:**
    - Se asocian los paquetes con sesiones (utiliza el campo flujo del IPv6)
  - **Las reservas las inicia el receptor para el soporte de multidifusión**
  - **“Soft state”: reserva temporal que finaliza después de 30 segundos:**
    - Simplifica la gestión de conexiones
    - Requiere mensajes de actualización
  - **Gestión de paquetes para garantizar el servicio:**
    - Mecanismos propietarios (ej.: el WFQ o *Weighted fair queueing*)
- **Problemas de escalabilidad:**
  - **Cada *router* necesita hacer un seguimiento de un gran número de flujos que aumenta con el tamaño (capacidad) del *router***

# Servicios diferenciados (*Diffserv*)

---

- Aunque el *Diffserv* RSVP no necesita hacer un seguimiento de cada uno de los flujos:
  - Asigna recursos a un pequeño número de clases de tráfico  
Reúne en cola los paquetes de la misma clase
  - Ej.: dos clases de tráfico: especial (*premium*) y regular  
Utiliza un *bit* para diferenciar los paquetes especiales de los regulares
  - Problemas:
    - ¿Quién establece el *bit* especial?
    - ¿En qué se diferencia el servicio especial del regular?
- La IETF propone utilizar el campo TOS de la cabecera IP para identificar las clases de tráfico:
  - Potencialmente, más de dos clases

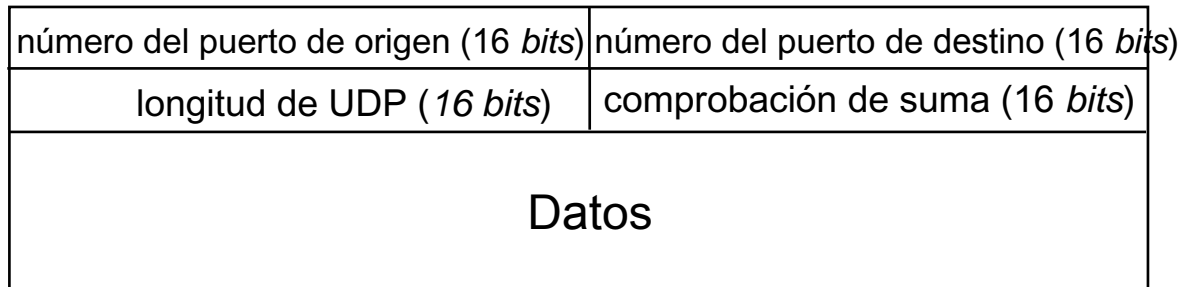
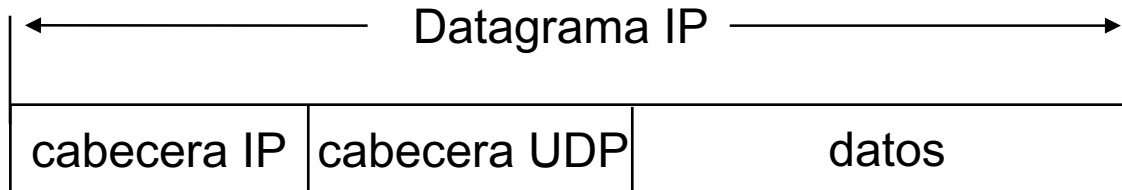
# Protocolo de datagramas de usuario (UDP)

---

- **Protocolo de la capa de transporte:**
  - Difusión de mensajes a través de la red
- **Orientado a datagramas:**
  - **No es fiable:**
    - No posee un mecanismo de control de errores
  - Sin conexión
  - No es un protocolo de flujo (*“stream” protocol*)
- **La longitud máxima de los paquetes es de 65 Kb**
- **Comprobación de suma del UDP:**
  - Abarca la cabecera y los datos
  - **Opcional:**
    - La pueden utilizar las aplicaciones
- **El UDP permite que las aplicaciones se entiendan directamente con el IP con un mínimo proceso adicional o protocolo superior**

# Formato de la cabecera del UDP

---



- **Los números de puerto identifican los procesos de envío y recepción:**
  - Es decir, FTP, email, etc.
  - Permiten que el UDP multiplexe los datos en una sola trama
- **Longitud del UDP = longitud del paquete en bytes:**
  - Un mínimo de 8 y un máximo de  $2^{16} - 1 = 65.535$  bytes
- **La comprobación de suma abarca la cabecera y los datos:**
  - Opcional, el UDP no hace nada con la comprobación de suma

# Protocolo de control de transmisión (TCP)

---

- **Protocolo de la capa de transporte:**
  - Transmisión fiable de mensajes
- **Orientado a conexión:**
  - Tráfico de tramas
  - Debe reordenar los paquetes IP desordenados
- **Fiable:**
  - Mecanismo ARQ
  - Obsérvese que los paquetes tienen un número de orden y un número *ACK*
  - Obsérvese que la cabecera de los paquetes tiene el tamaño de una ventana (para el Retroceso N)
- **Mecanismo de control de flujo:**
  - Inicio lento:
    - Limita el tamaño de la ventana en función de la congestión

# Funcionamiento básico del TCP

---

- **En el emisor:**
  - Se dividen los datos de aplicación en segmentos TCP
  - El TCP utiliza un temporizador, mientras espera la confirmación de cada paquete
  - Los paquetes sin confirmar se retransmiten
- **En el receptor:**
  - Se detectan los errores mediante una comprobación de suma
  - Se admiten los datos recibidos correctamente
  - Se reúnen los segmentos en el orden adecuado
  - Se eliminan los segmentos duplicados
- **Control de flujo y retransmisión basada en ventanas**

# Campos de la cabecera TCP

---

		16	32
Puerto de origen		Puerto de destino	
Número de secuencia (SN)			
Número de reconocimiento (RN)			
<i>Offset</i> de datos	Reservado	Control	Ventana
Comprobación de suma		Puntero urgente	
Opciones (si las hay)			
Datos			

# Campos de la cabecera TCP

---

- Los números de los puertos son los mismos que para el UDP
- El SN de 32 *bits* identifica de forma inequívoca los datos de aplicación que contiene el segmento TCP:
  - El SN está en *bytes*
  - Identifica el primer *byte* de datos
- El RN de 32 *bits* se utiliza para la superposición de confirmaciones o reconocimiento (*piggybacking*):
  - El RN indica el siguiente *byte* que espera el receptor
  - Implica la confirmación (ACK) de todos los *bytes* recibidos hasta ese momento
- El *offset* de datos es una longitud de cabecera en 32 palabras de *bits* (mínimo de 20 *bytes*)
- El tamaño de la ventana:
  - Se utiliza para la recuperación de errores (ARQ) y como mecanismo de control de flujo:
    - El emisor no puede tener más de una ventana de paquetes en la red simultáneamente
  - Especificado en *bytes*:
    - El ajuste de ventana se utiliza para aumentar el tamaño de la misma en redes de alta velocidad
- La comprobación de suma abarca la cabecera y los datos

- 
- **La recuperación de errores se realiza en múltiples capas**
    - Enlace, transporte y aplicación
  - **La recuperación de errores de la capa de transporte es necesaria porque:**
    - Se pueden producir pérdidas de paquetes en la capa de red
      - Ej.: desbordamiento de *buffer*
    - Algunas capas de enlace pueden no ser fiables
  - **El SN y el RN se utilizan para la recuperación de errores de forma similar al Retroceso N de la capa de enlace:**
    - Para reordenar los paquetes es necesario un SN grande
  - **El TCP utiliza un mecanismo de tiempo de espera (*Timeout*) para la retransmisión de paquetes:**
    - Cálculo del tiempo de espera
    - Retransmisión rápida

# Cálculo del tiempo de espera en el TCP

---

- **Se basa en el cálculo del tiempo correspondiente a una vuelta de recorrido (RTT):**

- Promedio de peso

$$\text{RTT\_PROM} = a * (\text{RTT\_calculado}) + (1-a) * \text{RTT\_PROM}$$

- **El tiempo de espera es un múltiplo de RTT\_PROM (generalmente dos):**
  - Un tiempo de espera corto conlleva demasiadas retransmisiones
  - Un tiempo de espera largo conlleva largos retardos e ineficacia
- **Para lograr que el tiempo de espera sea más tolerante a las variaciones de retardo se ha propuesto (Jacobson) establecer el valor del tiempo de espera en función de la desviación estándar del RTT:**

$$\text{Tiempo de espera} = \text{RTT\_PROM} + 4 * \text{RTT\_SD}$$

- **En muchas implementaciones del TCP el valor mínimo del tiempo de espera es de 500 ms debido a la granularidad del reloj**

# Retransmisión rápida

---

- **Cuando el TCP recibe un paquete con un SN mayor de lo esperado, envía un paquete de confirmación con un número de reconocimiento del SN del paquete esperado:**
  - Esto se puede deber a una entrega fuera de orden o a la pérdida de un paquete
- **Si se pierde un paquete, el TCP enviará un duplicado de los RN hasta que el paquete llegue correctamente:**
  - Pero no se retransmitirá el paquete hasta que haya un tiempo de espera
  - Esto conlleva un retardo añadido e ineficacia
- **La retransmisión rápida supone que si el módulo emisor recibe 3 RN duplicados, el paquete se ha perdido:**
  - Tras la recepción de 3 RN duplicados se retransmite el paquete
  - Tras la retransmisión, se continúa con el envío de nuevos datos
- **La retransmisión rápida permite que la retransmisión TCP actúe a modo de repetición selectiva de ARQ:**
  - Futura opción para los ACK selectivos (SACK)

# Control de congestión en el TCP

---

- El TCP utiliza el tamaño de su ventana para llevar a cabo un control de la congestión extremo a extremo:
  - Hablaremos más sobre el control de flujo por ventana más tarde
- Idea principal:
  - Con un ARQ basado en ventanas, el número de paquetes de la red no puede exceder el tamaño de la ventana (CW)

$$\text{Último\_byte\_enviado (SN)} - \text{Último\_byte\_confirmado (RN)} \leq \text{CW}$$

- Al utilizar el control de flujo por ventana la tasa de transmisión es igual a una ventana de paquetes cada RTT:

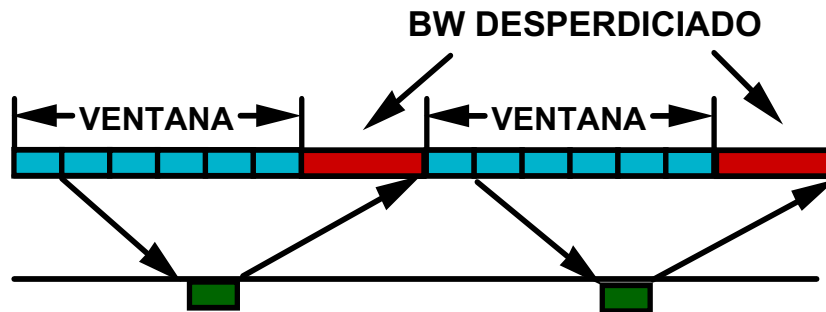
$$R = \text{CW}/\text{RTT}$$

- Controlando el tamaño de la ventana TCP se controla la tasa

# Efecto del tamaño de la ventana

---

- El tamaño de la ventana es el número de *bytes* que se pueden transportar simultáneamente



- Una ventana demasiado pequeña evita la transmisión continua
- Para permitir la transmisión continua, el tamaño de la ventana debe exceder el RTT

## Longitud de un *bit* (viajando a 2/3C)

---

<b>A 300 bps</b>	<b>1 bit = 415 millas</b>	<b>3000 millas = 7 bits</b>
<b>A 3.3 kbps</b>	<b>1 bit = 38 millas</b>	<b>3000 millas = 79 bits</b>
<b>A 56 kbps</b>	<b>1 bit = 2 millas</b>	<b>3000 millas = 1.5 kbits</b>
<b>A 1.5 Mbps</b>	<b>1 bit = 438 pies</b>	<b>3000 millas = 36 kbits</b>
<b>A 150 Mbps</b>	<b>1 bit = 4.4 pies</b>	<b>3000 millas = 3.6 Mbits</b>
<b>A 1 Gbps</b>	<b>1 bit = 8 pulgadas</b>	<b>3000 millas = 240 Mbits</b>

# Ajuste dinámico del tamaño de la ventana

---

- **TCP empieza con un paquete con  $CW = 1$  y aumenta el tamaño de la ventana lentamente a medida que se reciben las confirmaciones:**
  - Fase de inicio lenta
  - Fase de evitar la congestión
- **Fase de inicio lenta:**
  - Durante el inicio lento, el TCP aumenta la ventana en una proporción de un paquete por cada confirmación recibida
  - Cuando  $CW = \text{umbral}$  (*Threshold*) el TCP pasa a la fase de evitar la congestión
  - Importante: durante el inicio lento,  $CW$  dobla cada RTT  
¡Aumento exponencial!
- **Fase de evitar la congestión:**
  - Durante esta fase, el TCP aumenta la ventana en una proporción de un paquete por cada ventana de confirmaciones recibida
  - Obsérvese que durante esta fase,  $CW$  aumenta en 1 por cada RTT  
¡Aumento lineal!
- **El TCP continúa aumentando el  $CW$  hasta que se produzca la congestión**

# Reacción ante la congestión

---

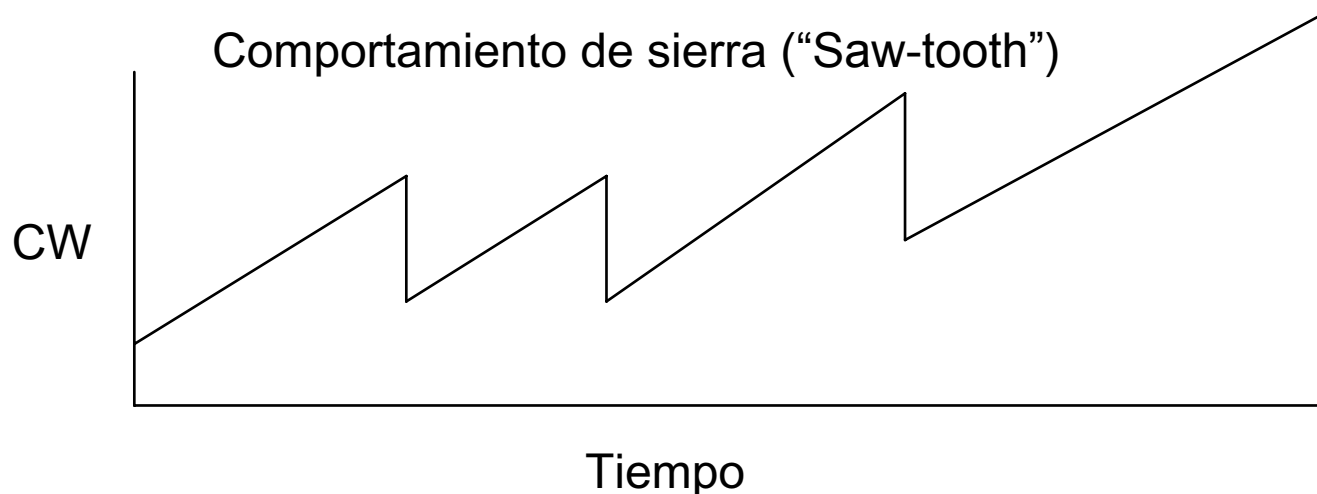
- **Hay muchas variaciones: Tahoe, Reno, Vegas, etc.**
- **Idea principal: cuando se produce la congestión disminuye el tamaño de la ventana**
- **Hay dos mecanismos que indican la congestión:**
  - **Confirmaciones duplicadas: se podría deber a una congestión temporal**
  - **Tiempo de espera: es más probable que se deba a una congestión importante**
- **TCP Reno: implementación más habitual**
  - **Si se alcanza el tiempo de espera,  $CW = 1$  y vuelve a la fase de inicio lento**
  - **Si hay confirmaciones duplicadas,  $CW = CW/2$  y permanece en la fase de evitar la congestión**

# Explicación de la dinámica del TCP

---

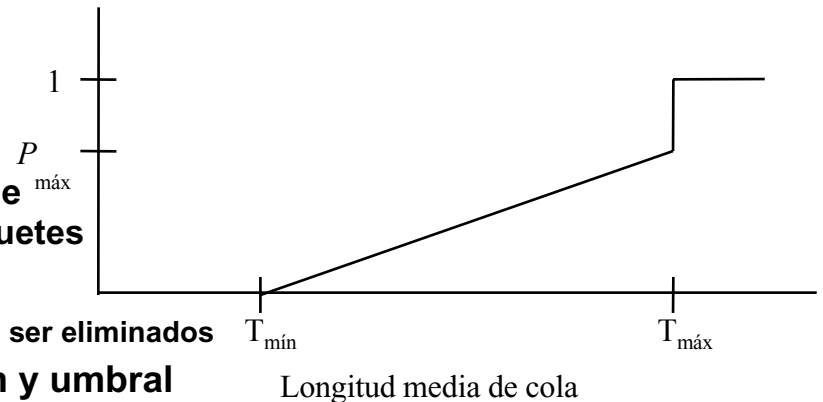
- La fase de inicio lento es, en realidad, rápida
- El TCP se pasa la mayor parte del tiempo en fase de evitar la congestión
- Mientras está en la fase de evitar la congestión:
  - CW aumenta en 1 por cada RTT
  - CW disminuye en una proporción de 2 con cada pérdida

“Aumento añadido / Disminución multiplicativa”

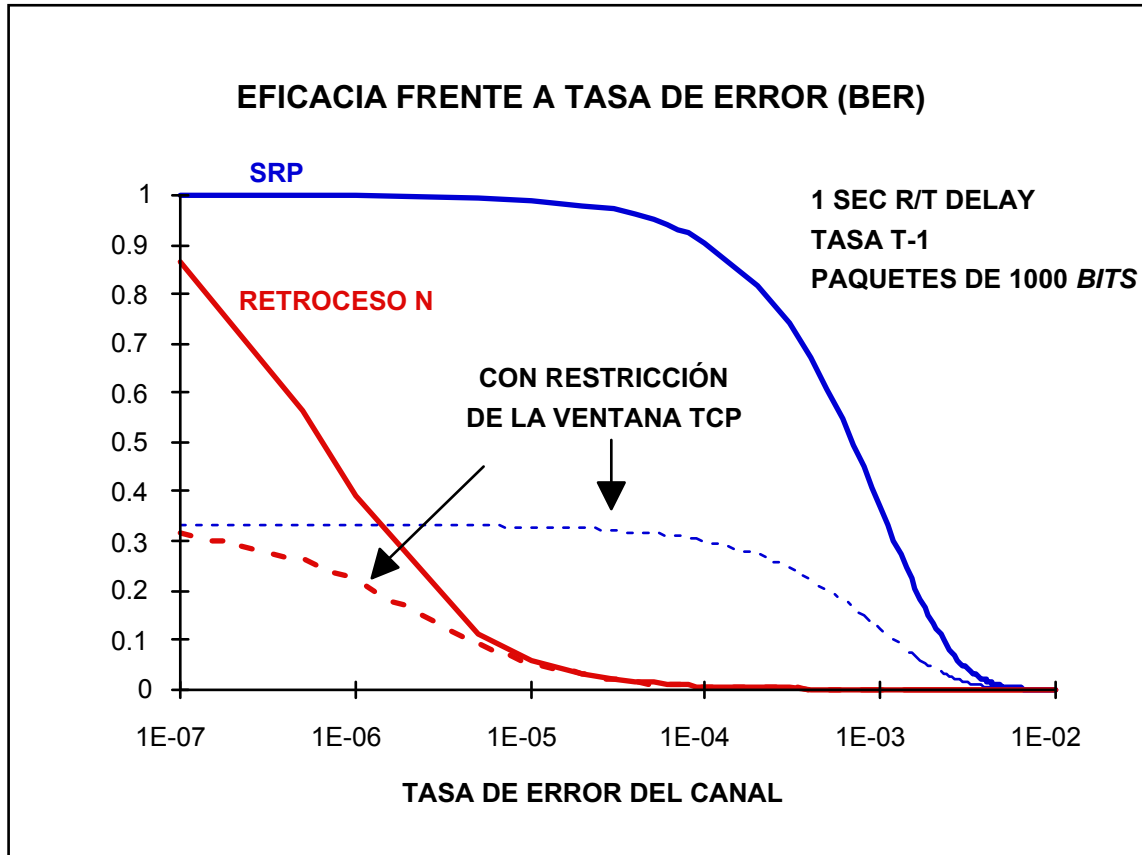


# RED (*Random Early Detection*)

- En lugar de eliminar un paquete por desbordamiento de la cola, se elimina antes de forma probabilística
- **Motivos:**
  - Los paquetes eliminados se utilizan como mecanismo para obligar al origen a transmitir más despacio:
    - Si esperamos a que se produzca el desbordamiento de la cola será demasiado tarde y puede que tengamos que eliminar muchos paquetes
    - Nos lleva al problema de sincronización del TCP en el que todos los orígenes se frenan a la vez
  - RED avisa pronto de la congestión:
    - El uso del azar reduce el problema de sincronización del TCP
- **Mecanismo:**
  - Utiliza el tamaño medio de cola de peso:
    - Si  $Q\_MED > T_{\min}$  se elimina con prob.  $P$
    - Si  $Q\_MED > T_{\max}$  se elimina con prob. 1
  - RED se puede utilizar con un aviso explícito de congestión, en lugar de la eliminación de paquetes
  - RED tiene una propiedad de equidad:
    - Los flujos grandes tienen más probabilidades de ser eliminados
  - Los valores de probabilidad de eliminación y umbral (threshold) son un campo de investigación activo

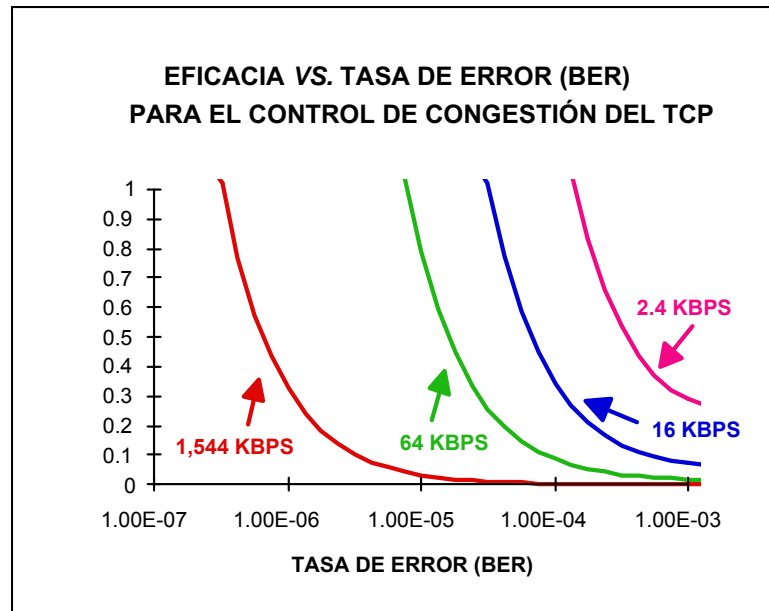


# Control de errores en el TCP



- TCP original, diseñado para una BER baja y enlaces de retardo bajo
- Futuras versiones (RFC 1323) permitirán ventanas de mayor tamaño y retransmisiones selectivas

# Impacto de los errores de transmisión sobre el control de congestión del TCP



- El TCP supone que los paquetes eliminados se deben a la congestión y responde reduciendo la tasa de transmisión
- En un enlace con una tasa de error elevada es más probable que los paquetes eliminados se deban a errores que a la congestión
- Extensiones del TCP (RFC 1323):
  - Mecanismo de retransmisión rápida, recuperación rápida y ajuste de ventana

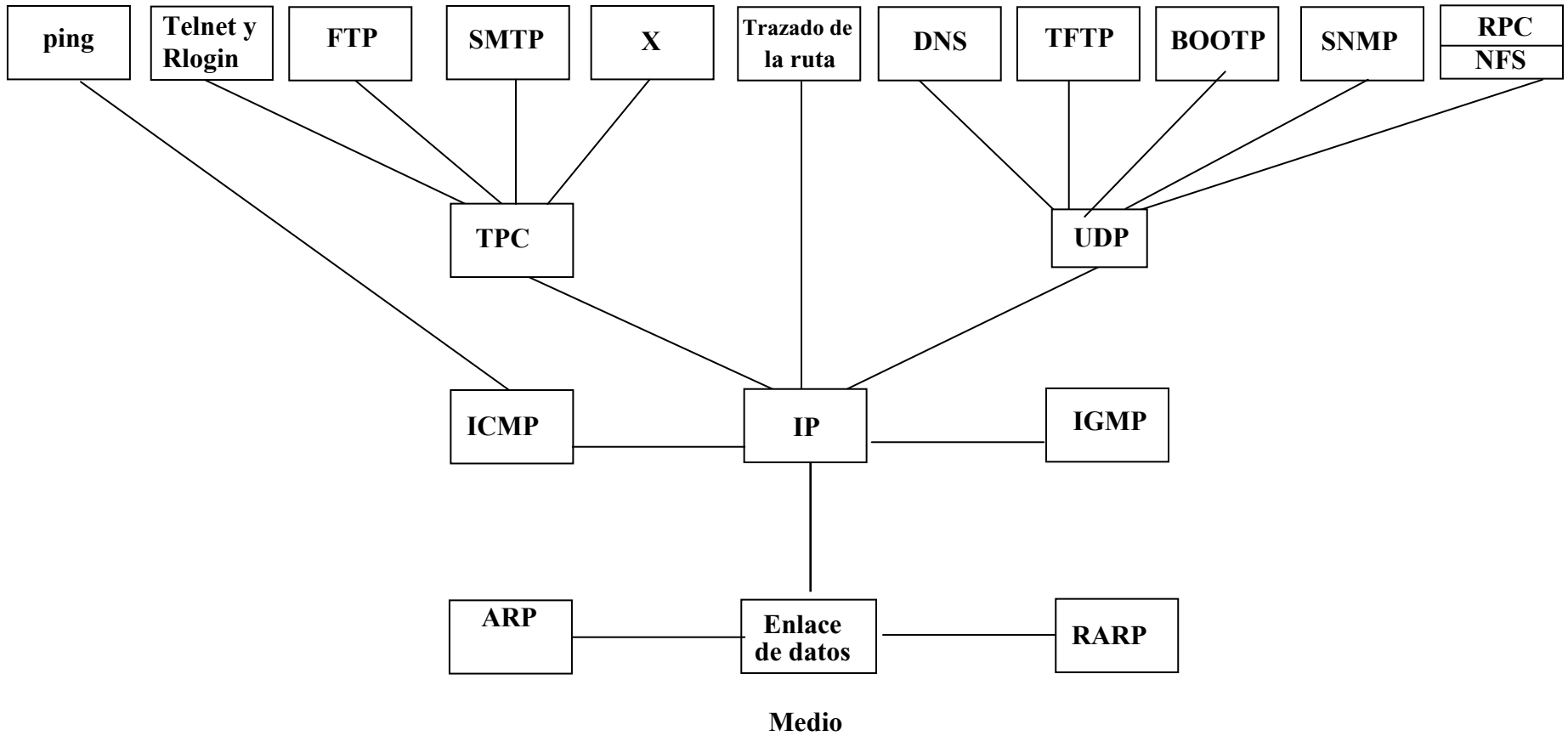
# Versiones del TCP

---

- **Los estándares del TCP se publican como RFCs**
- **Las implementaciones del TCP en ocasiones difieren unas de otras:**
  - Pueden no implementar las últimas extensiones, *bugs*, etc.
- **La implementación estándar es la BSD:**
  - Grupo de investigación de sistemas informáticos de UC-Berkeley
  - La mayoría de las implementaciones del TCP se basan en la BSD:  
SUN, MS, etc.
- **Versiones BSD:**
  - **4.2BSD - 1983**  
Primera versión de gran difusión
  - **4.3BSD Tahoe - 1988**  
Fase de inicio lento y fase para evitar la congestión
  - **4.3BSD Reno - 1990**  
Compresión de la cabecera
  - **4.4BSD - 1993**  
Soporte para multidifusión, RFC 1323 para un alto rendimiento

# El conjunto TCP/IP

---



# Modo de transferencia asíncrona (ATM)

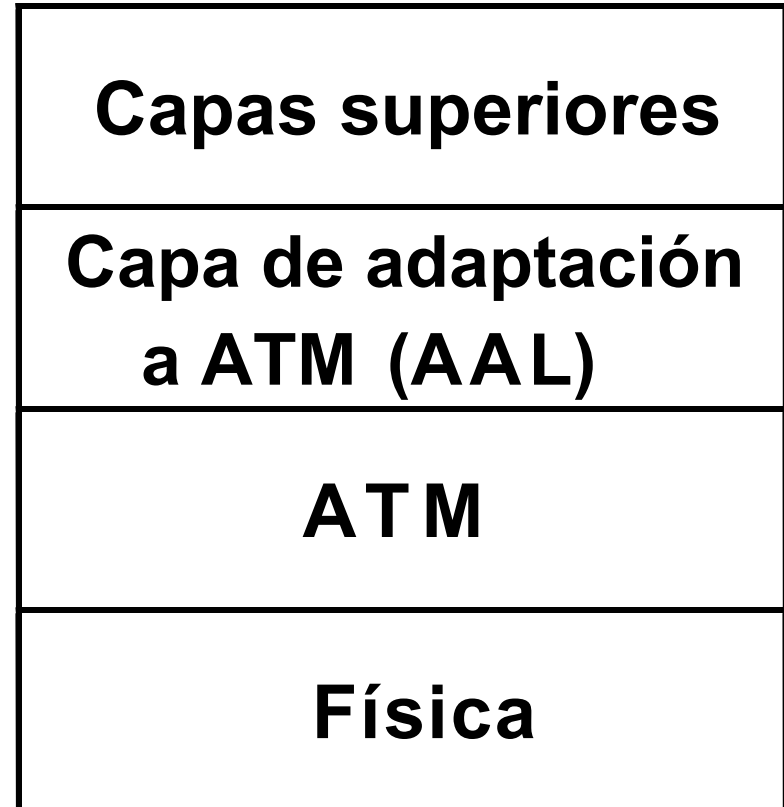
---

- **Intento de las compañías telefónicas en los años 80 para desarrollar un estándar de red integrada (BISDN) que pudiera dar soporte a voz, datos, vídeo, etc.**
- **ATM utiliza pequeños paquetes de tamaño fijo (53 *bytes*) llamados “celdas”:**
  - **¿Por qué celdas?**
    - La conmutación de celdas presenta propiedades tanto de la conmutación de paquetes como de la de circuitos
    - Es más fácil implementar conmutadores (*switches*) de alta velocidad
  - **¿Por qué 53 *bytes*?**
  - **Las celdas pequeñas son buenas para el tráfico de voz (limitan los retardos de muestreo):**
    - En el caso de voz a 64Kbps tarda 6 ms en llenar una celda con datos
- **Las redes ATM son orientadas a conexión:**
  - **Circuitos virtuales**

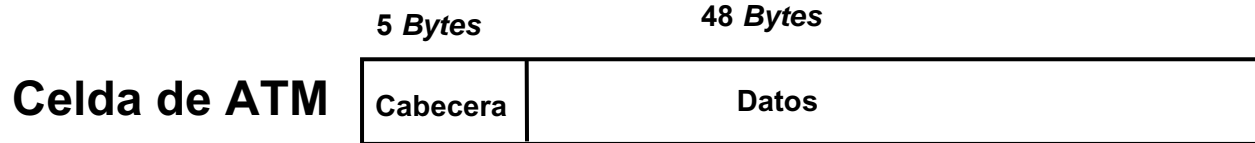
# Arquitectura de referencia para ATM

---

- **Capas superiores:**
  - Aplicación
  - TCP/IP
- **Capa de adaptación a ATM:**
  - Similar a la capa de transporte
  - Proporciona una interfaz entre ATM y las capas superiores
    - Divide los mensajes en celdas y los vuelve a unir
- **Capa ATM:**
  - Conmutación por celdas
  - Control de la congestión
- **Capa física:**
  - ATM diseñado para SONET:
    - Red óptica síncrona
    - Esquema de transmisión TDMA con tramas de 125  $\mu$ s

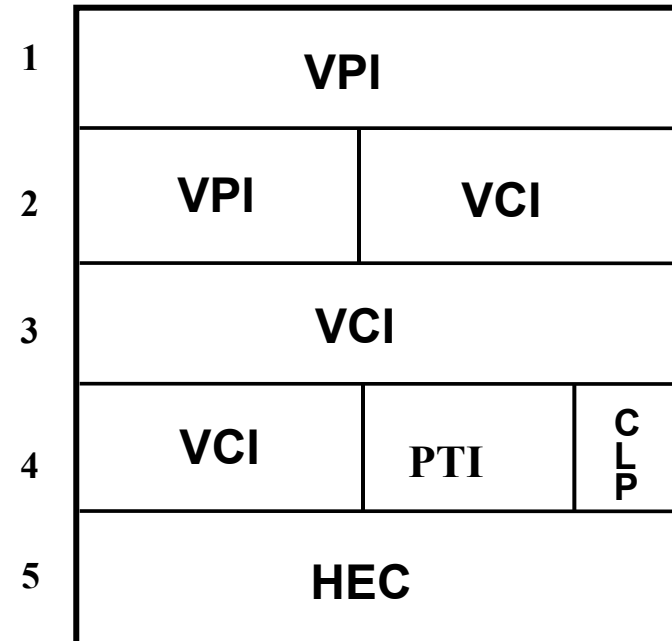


# Formato de celda en ATM



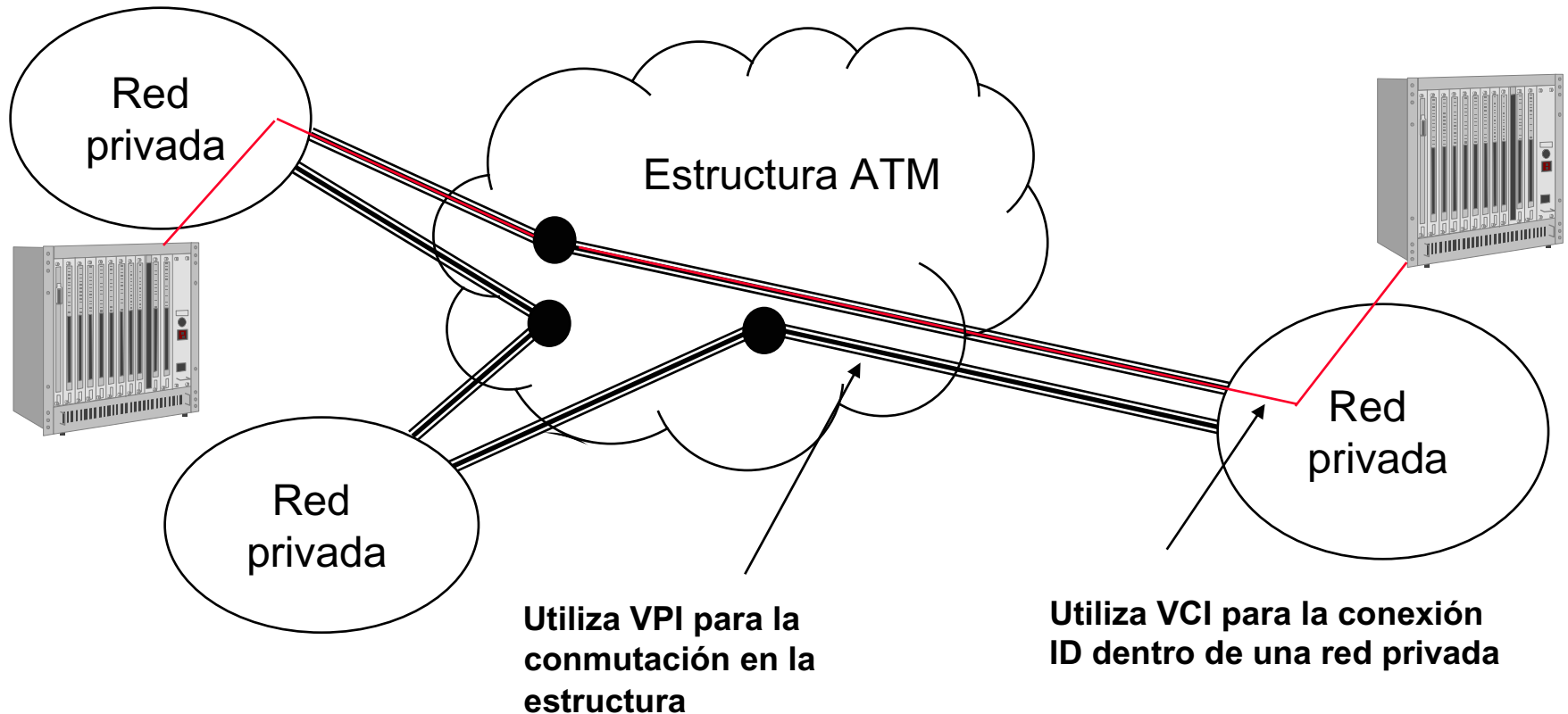
- **Números de circuitos virtuales** (obsérvese que el espacio de la dirección es relativamente pequeño):
  - ID del canal virtual
  - ID de la ruta virtual
- **PTI: tipo de carga útil**
- **CLP: la celda pierde prioridad (1 bit):**
  - Indica las celdas que se pueden eliminar
- **HEC: CRC en la cabecera**

## Cabecera de ATM (NNI)



# VPI/VCI

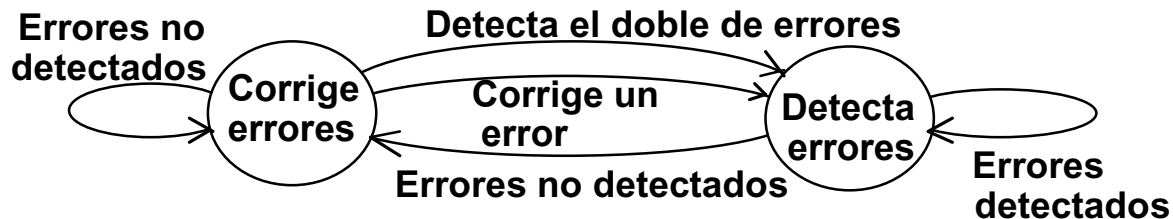
- **VPI** identifica una ruta física entre el origen y el destino
- **VCI** identifica una conexión lógica (sesión) dentro de dicha ruta:
  - Este enfoque permite tablas de enrutamiento más pequeñas y simplifica el cálculo de las rutas



# CRC EN LA CABECERA ATM

---

- **ATM utiliza un CRC de 8 *bits* capaz de corregir 1 error**
- **Comprueba sólo la cabecera de la celda y alterna entre dos modos:**
  - **En modo detección no corrige ningún error, pero es capaz de detectar más errores**
  - **En modo corrección puede corregir de forma fiable hasta un error, pero no es capaz de detectar tantos errores**
- **Cuando el canal es relativamente bueno, interesa estar en modo corrección; sin embargo, cuando el canal es malo interesa estar en modo detección para maximizar la capacidad de detección**



# Categorías del servicio ATM

---

- Tasa de *bits* constante (CBR): por ejemplo, voz sin compresión
  - Emulación de circuito
- Tasa de *bits* variable (rt-VBR): por ejemplo, vídeo comprimido
  - En tiempo real y sin ser en tiempo real
- Tasa de *bits* disponibles (ABR): por ejemplo, interconexión de redes LAN
  - Para tráfico a ráfagas con garantías BW limitadas y control de congestión
- Tasa de *bits* no especificados (UBR): por ejemplo, Internet
  - ABR sin garantías BW y sin control de congestión

# Parámetros del servicio en ATM (ejemplos)

---

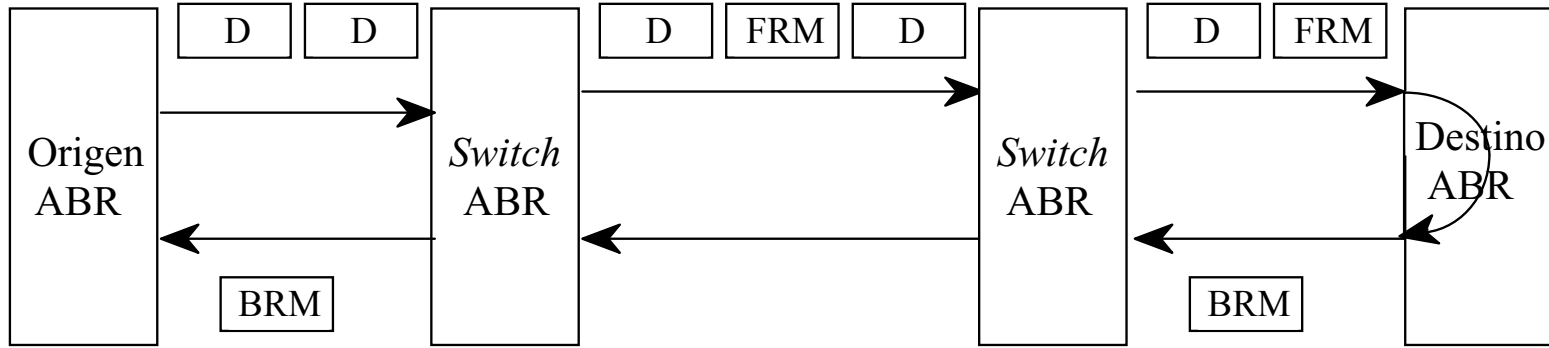
- Tasa pico de celda (PCR)
- Tasa sostenida de celda (SCR)
- Tamaño máximo de la ráfaga (MBS)
- Tasa mínima de celda (MCR)
- Tasa de celdas perdidas (CLR)
- Retardo de transmisión de la celda (CTD)
- Variación del retardo de la celda (CDV)
  
- No todos los parámetros se aplican a todas las categorías del servicio:
  - Ej.: CBR especifica PCR y CDV
  - VBR especifica MBR y SCR
  
- La red garantiza el QoS siempre que el usuario se ajuste a su contrato como especifican los parámetros anteriores:
  - Cuando los usuarios exceden su tasa, la red puede eliminar sus paquetes
  - La tasa de celda se puede controlar con el esquema de control de tasa (*leaky bucket*)

# Control de flujo en redes ATM (ABR)

---

- **ATM utiliza celdas de gestión de recursos para controlar los parámetros de la tasa:**
  - **Gestión de recursos hacia adelante (FRM)**
  - **Gestión de recursos hacia atrás (BRM)**
- **Las celdas RM contienen:**
  - **Indicador de congestión (CI)**
  - **Indicador de no aumento (NI)**
  - **Tasa explícita de celda (ER)**
  - **Tasa actual de celda (CCR)**
  - **Tasa mínima de celda (MCR)**
- **El origen genera celdas RM con regularidad:**
  - **A medida que las celdas RM pasan a través de la red se pueden marcar con  $CI=1$  para indicar congestión**
  - **Las celdas RM vuelven al origen, donde:**
    - $CI = 1 \Rightarrow$  disminuye la tasa en alguna fracción**
    - $CI = 1 \Rightarrow$  aumenta la tasa en alguna fracción**
  - **ER se puede utilizar para establecer la tasa explícita**

# Flujo de celdas RM extremo a extremo



D = celda de datos

FRM = celda RM hacia adelante

BRM = celda RM hacia atrás

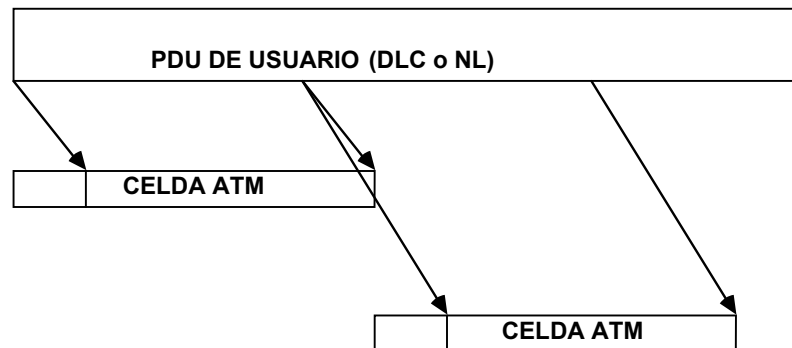
**Una vez que llega al destino, la celda RM “da media vuelta” y se dirige otra vez hacia el origen**

# Capas de adaptación a ATM

---

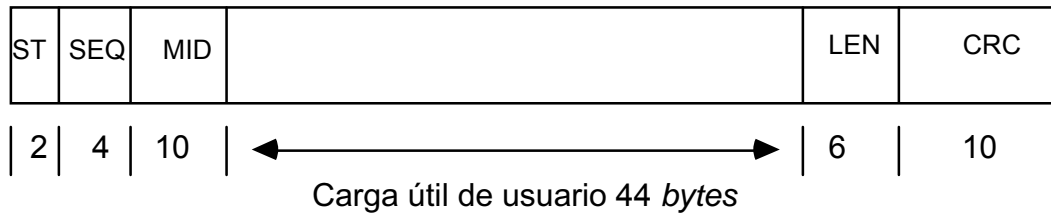
- Interfaz entre la capa ATM y los paquetes de capas superiores
- Cuatro capas de adaptación que se corresponden muy de cerca con las clases del servicio ATM:
  - AAL-1 para dar soporte al tráfico CBR
  - AAL-2 para dar soporte al tráfico VBR
  - AAL-3/4 para dar soporte al tráfico de ráfagas de datos
  - AAL-5 para dar soporte a IP con encabezado mínimo
- Las funciones y el formato de la capa de adaptación dependen de la clase de servicio:
  - Por ejemplo, el tráfico de tipo "stream" (flujo) requiere los números de secuencia para identificar qué celdas se han eliminado

Cada clase de servicio tiene un formato de cabecera diferente (además de la cabecera ATM de 5 *bytes*)



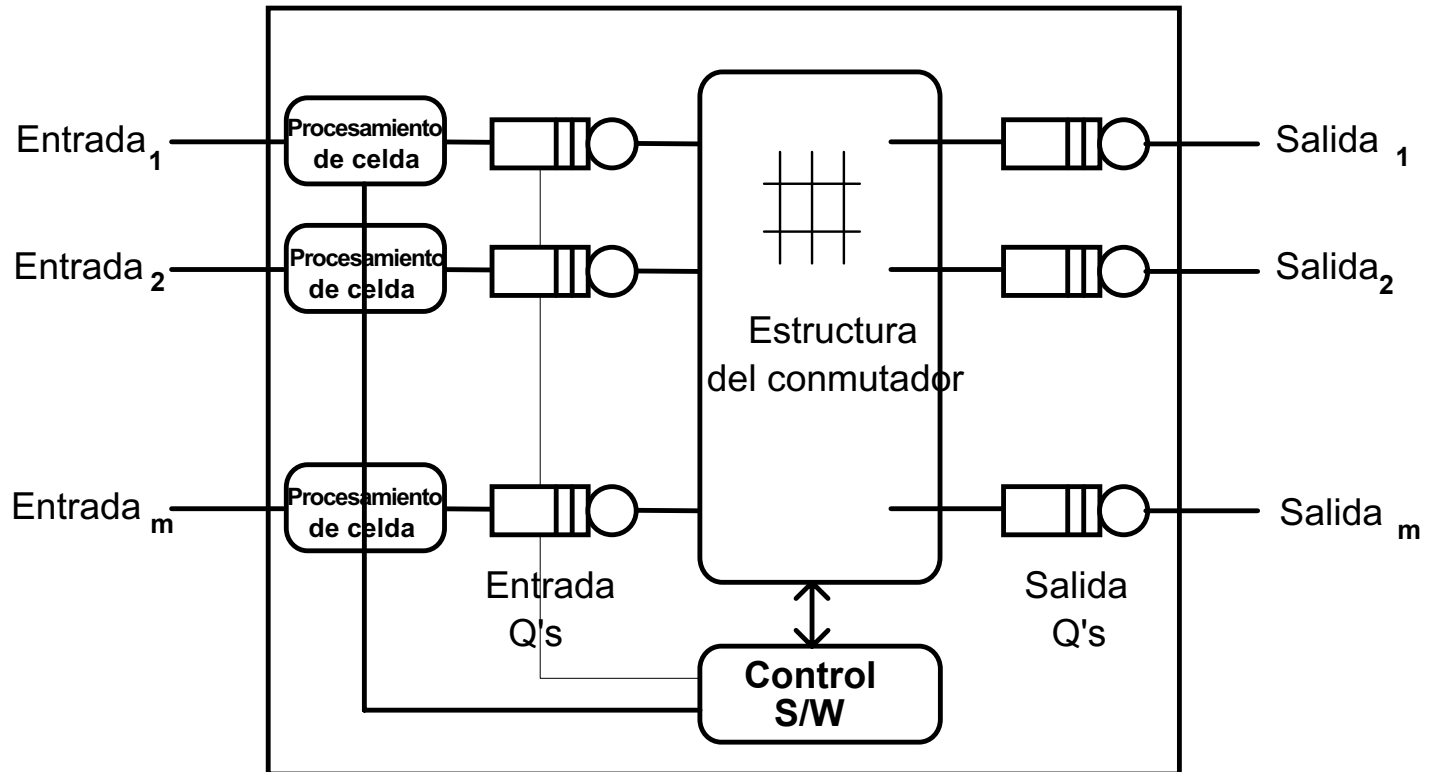
# Ejemplo: AAL 3/4

CARGA ÚTIL DE LA CELDA ATM (48 bytes)



- **ST:** tipo de segmento (primero, medio y último)
- **SEQ:** número de secuencia de 4 *bits* (detecta las celdas perdidas)
- **MID:** ID del mensaje (para reordenar y volver a unir múltiples mensajes)
- Carga útil de usuario de 44 *bytes* (~84% eficaz)
- **LEN:** longitud de los datos de este segmento
- **CRC:** segmento CRC de 10 *bits*
  
- **AAL 3/4** permite multiplexado, fiabilidad y detección de errores, pero es bastante difícil de procesar y añade mucho encabezado
- **AAL 5** se introdujo para dar soporte al tráfico IP:
  - Unas cuantas funciones menos, pero mucho menos encabezado y complejidad

# Conmutadores de celda ATM



- **Problemas de diseño:**
  - Cola de entrada frente a cola de salida
  - Bloqueo de la cabeza de línea
  - Velocidad de la estructura (*fabric*)

# Resumen del ATM

---

- **El ATM se utiliza en la mayoría de los casos como tecnología de red de núcleo (*core network*)**
- **Ventajas del ATM:**
  - **Capacidad para proporcionar QoS**
  - **Capacidad para gestionar el tráfico**
  - **Conmutación rápida de celdas utilizando números VC relativamente cortos**
- **Desventajas del ATM:**
  - **No es IP: prácticamente todo está diseñado para TCP/IP**
  - **No es un protocolo extremo a extremo convencional:**
    - No funciona bien en entornos heterogéneos**
    - No ha sido diseñado para interoperar con otros protocolos**
    - No es una buena opción para determinados medios físicos (ej.: inalámbricos)**
  - **El IP puede "tomar prestados" muchas de las ventajas del ATM:**
    - Routers de núcleo para la conmutación por celdas**
    - Mecanismos de conmutación de etiquetas**

# Conmutación de etiquetas multiprotocolo (MPLS)

---

**“A medida que se vuelvan habituales más servicios con unos requerimientos de retardo y tasa de transferencia establecidos, el IP necesitará circuitos virtuales (aunque probablemente los llame de otro modo)”**

**RG, 28 de abril de 1994**

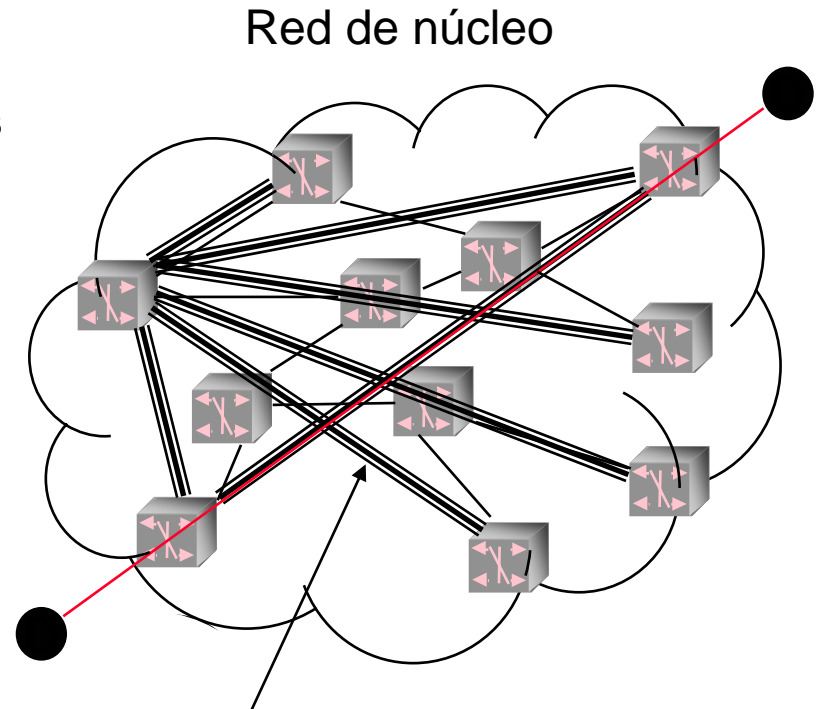
# Conmutación de etiquetas

---

- Los fabricantes de *routers* se dieron cuenta de que para aumentar la velocidad y la capacidad debían adoptar un mecanismo similar al ATM:
  - Un *switch* basado en una etiqueta simple y que no requiera consultas en complejas tablas de enrutamiento
  - Utilizar los circuitos virtuales para gestionar el tráfico (QoS)
  - Utilizar la conmutación por celdas en el núcleo del *router*
- Primer intento: conmutación IP
  - Los *routers* intentan identificar los flujos:
    - Definen un flujo tras observar un número de paquetes entre un origen y un destino dados (ej.: 5 paquetes en un segundo)
  - Mapear los pares IP de origen-destino hacia los VC de ATM
    - Algoritmo distribuido en el que cada *router* toma su propia decisión
- Conmutación de etiquetas multiprotocolo (MPLS):
  - Conocida también como Conmutación de etiquetas
  - No depende del ATM
  - Añade una etiqueta a cada paquete para que haga de número VC:
    - Las etiquetas se pueden asignar permanentemente a determinadas rutas

# La conmutación de etiquetas se puede utilizar para crear una red virtual con la red de núcleo

- Los *routers* ubicados en el borde de la red de núcleo pueden estar relacionados entre sí mediante etiquetas
- Los paquetes que llegan a un *router* del borde pueden estar etiquetados con la etiqueta que indica el *router* de borde de destino
  - “Tunneling”
  - Simplifica significativamente el enrutamiento en el núcleo
  - Los *routers* interiores no necesitan recordar todos los prefijos IP del mundo exterior
  - Permite la ingeniería de tráfico:  
Asigna capacidad a las etiquetas en función de la demanda



**Rutas conmutadas por etiquetas**

# Referencias

---

- ***TCP/IP Illustrated* (Vol. 1 y 2), Stevens**
- ***Computer Networks*, Peterson y Davie**
- ***High Performance Communication Networks*, Walrand y Varaiya**

