

---

# TCP/IP e Internet

Eytan Modiano  
MIT

# 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>Drivers, tarjetas de interfaz</b>

**TCP - Protocolo de control de transmisión**

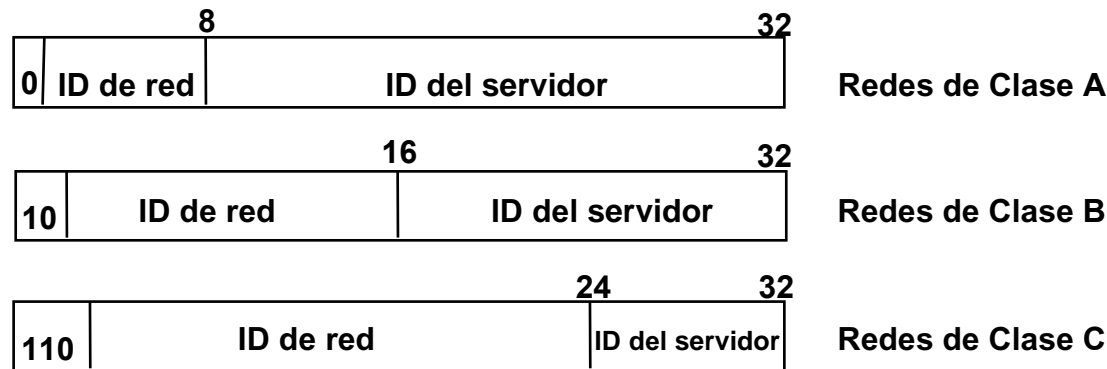
**UDP - Protocolo de datagramas de usuario**

**IP - Protocolo de Internet**

# 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*

# 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, 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)

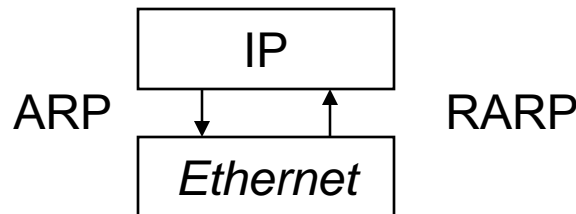
---

- **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

---

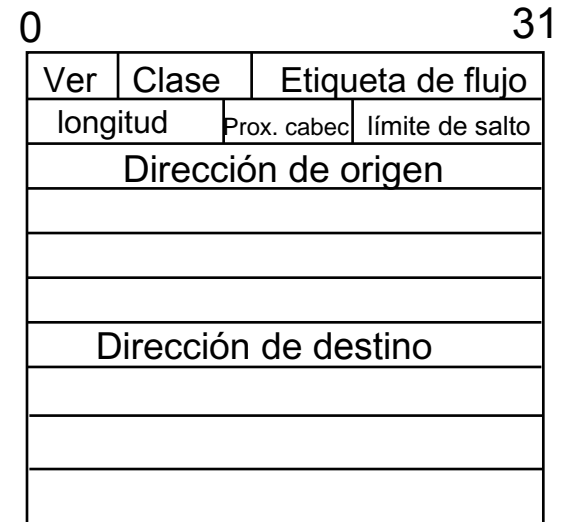
- **Internet está dividida en subredes, cada una de ellas bajo el control de una sola autoridad, conocida como sistema autónomo (AS)**
- **Los algoritmos de enrutamiento se dividen en dos categorías:**
  - **Protocolos interiores (operan dentro de un AS)**
  - **Protocolos exteriores (operan entre sistemas 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)**

# IPv6

- Se inició en 1991 con el IPng
- Motivos:
  - Necesidad de incrementar el espacio de direcciones IP
  - Soporte para aplicaciones en tiempo real: “QoS”
  - Seguridad, movilidad y autoconfiguración

- Principales cambios:

- Aumento del espacio de direcciones (*6 bytes*)
    - 1500 direcciones IP por cada pie cuadrado de la tierra
    - Partición de direcciones similar a la del CIDR
  - Soporte para QoS por medio del campo Etiqueta de flujo
  - Cabecera simplificada
- La mayoría de los motivos para desarrollar el IPv6 se habían tenido en cuenta en el IPv4
    - ¿Es realmente necesario el IPv6?
    - La transición del V4 al V6 es compleja



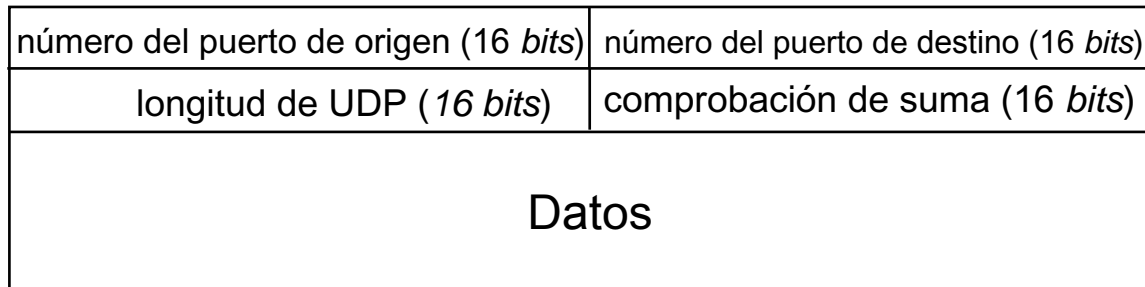
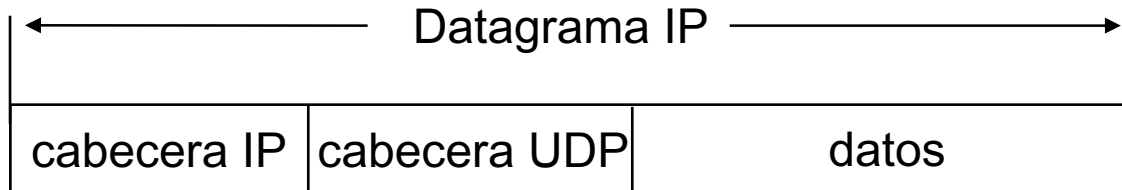
# Protocolo de datagramas de usuario (UDP)

---

- **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

# Recuperación de errores en el TCP

---

- **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

# 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:

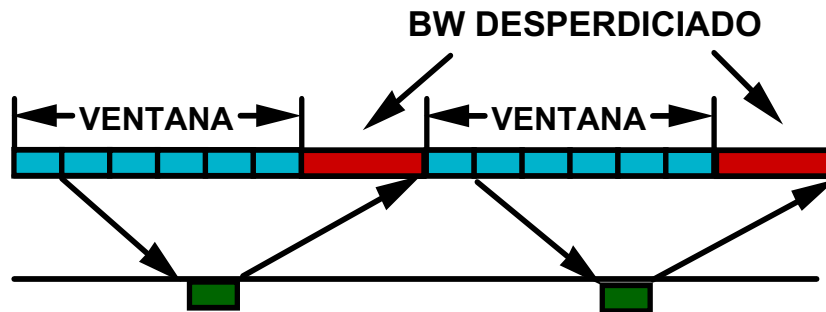
$$R = \text{CW}/\text{RTT}$$

- 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

# Ajuste dinámico del tamaño de la ventana

---

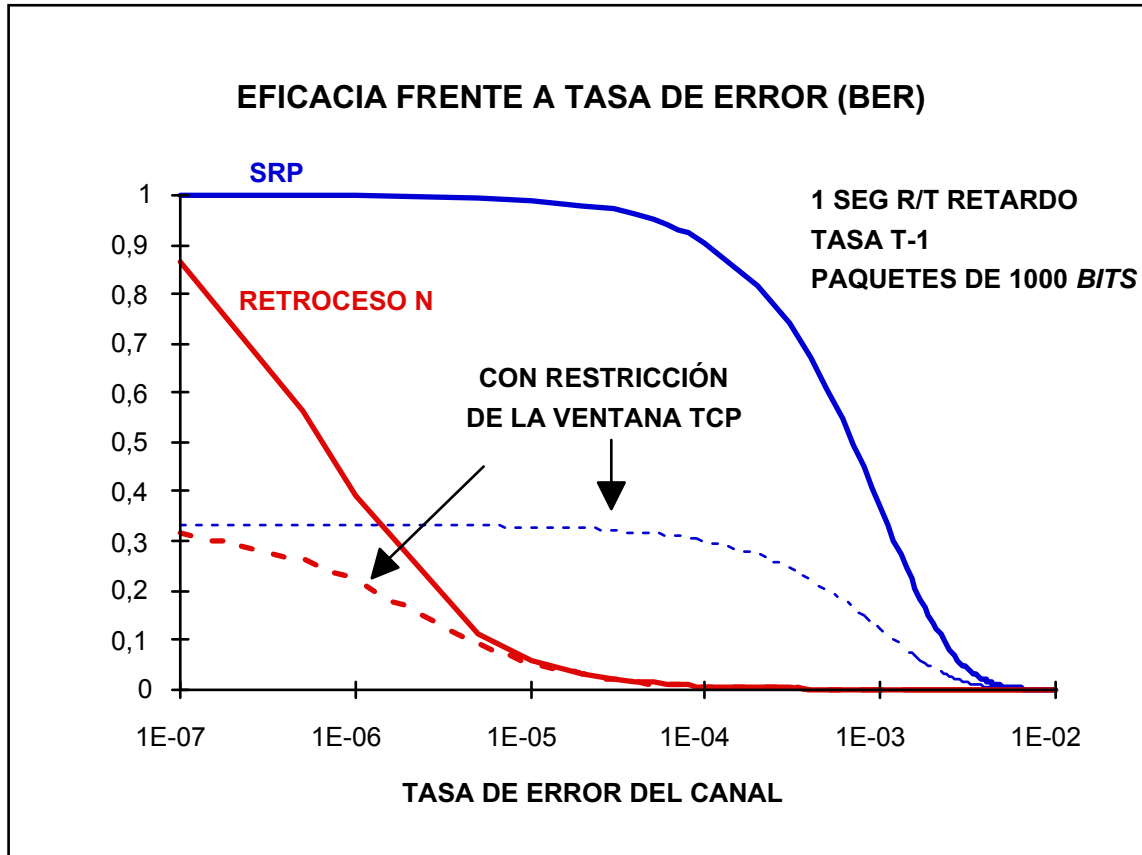
- **TCP empieza con un paquete con  $CW = 1$  y aumenta el tamaño de la ventana lentamente a medida que se reciben las confirmaciones:**
  - Fase de inicio lenta
  - Fase de evitar la congestión
- **Fase de inicio lenta:**
  - Durante el inicio lento, el TCP aumenta la ventana en una proporción de un 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

---

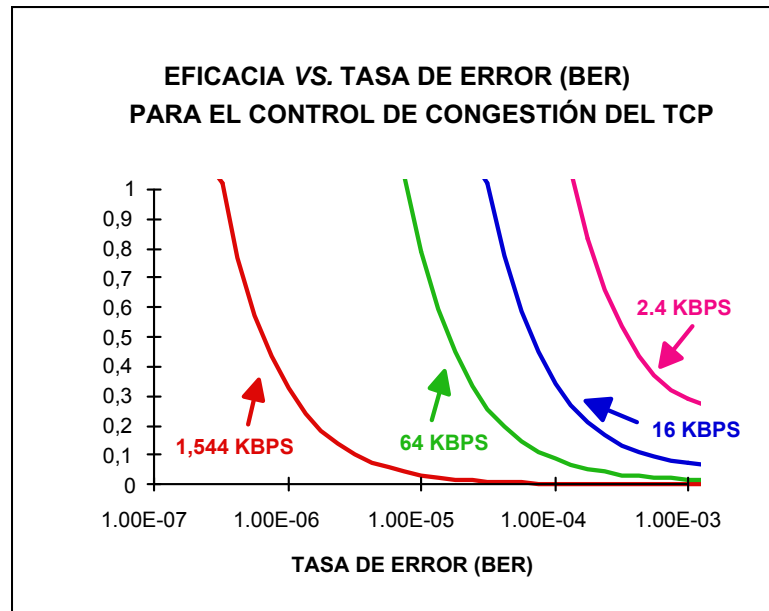
- **Hay muchas variaciones: Tahoe, Reno, Vegas, etc.**
- **Idea principal: cuando se produce la congestión disminuye el tamaño de la ventana**
- **Hay dos mecanismos que indican la congestión:**
  - **Confirmaciones duplicadas: se podría deber a una congestión temporal**
  - **Tiempo de espera: es más 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**

# 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

---

