Tema 5 – Nivel de Transporte

Redes de Computadores

Grado en Ingeniería Informática en Tecnologías de la Información

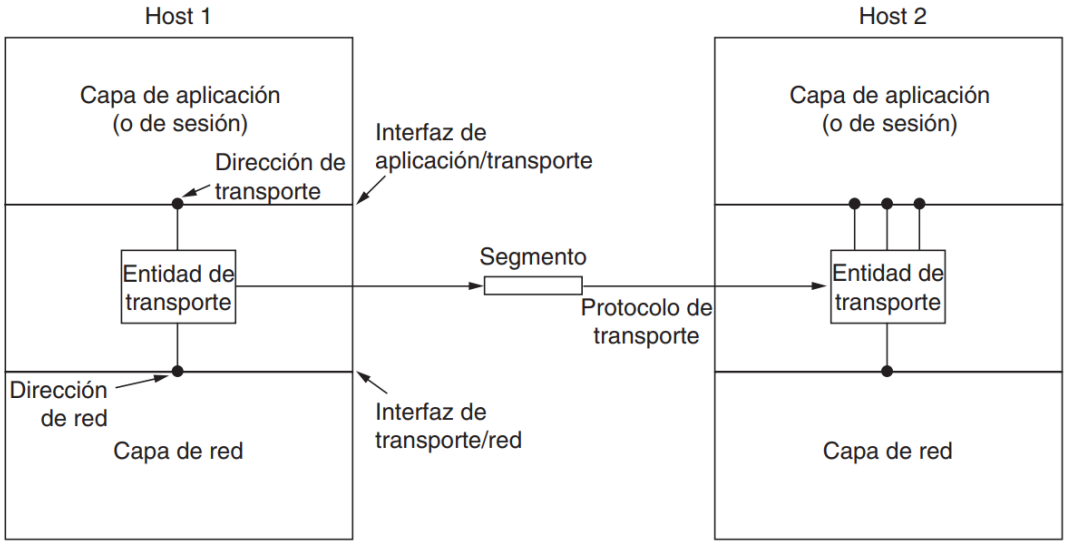
Ingeniería Telemática

ÍNDICE

1. Introducción
2. Elementos de protocolos de transporte
3. El protocolo UDP
4. El protocolo TCP

1.- INTRODUCCIÓN

- Proporciona una comunicación lógica entre procesos que se ejecutan en hosts diferentes (comunicación extremo a extremo)
  - Aísla a la capa de aplicación de los detalles de la red o redes intermedias
  - Host origen: divide el mensaje en segmentos y se los pasa al nivel de red
  - Host destino: junta los segmentos en mensajes y se los pasa a la capa de aplicación



[1]

Introducción

- Abstracción mediante sockets: Utilización de primitivas para facilitar el diseño y la programación a través de interfaces
- Se permite el intercambio de datos en ambos sentidos de forma simultánea: full-dúplex
- Existen dos tipos de protocolos:
  - Orientados a conexión: Segmentos
  - No orientados a conexión: Datagramas
- Comparación con la capa de red – Hosts vs Procesos
- Redundancia de tareas de la capa de enlace:
  - Control de flujo
  - Control de errores
  - Secuenciación

Introducción

- Se emplean dos protocolos principalmente:
  - TCP (*Transmission Control Protocol*)
    - Fiable
    - Entrega de información ordenada
    - Establecimiento de conexión
    - Control de flujo mediante ventana deslizante
    - Control de congestión explícita e implícita
  - UDP (*User Datagram Protocol*)
    - No fiable
    - Entrega de información no ordenada

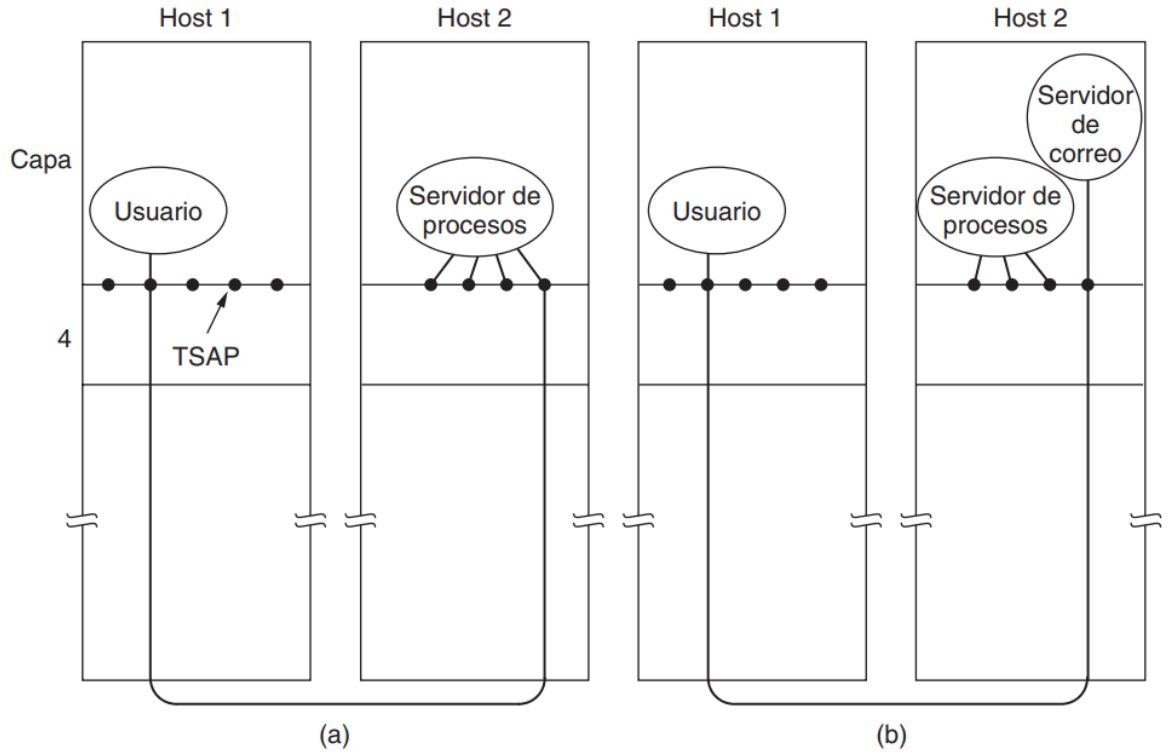
2.- ELEMENTOS DE PROTOCOLOS DE TRANSPORTE

- **Direccionamiento**
  - Conocer el *punto de acceso al servicio de transporte* (TSAP), que suele ser un número de puerto

Protocolo	Nº de Puerto
20 - 21	FTP
22	SSH
23	Telnet
25	SMTP
53	DNS
80	HTTP
443	HTTPS

Elementos de protocolos de transporte

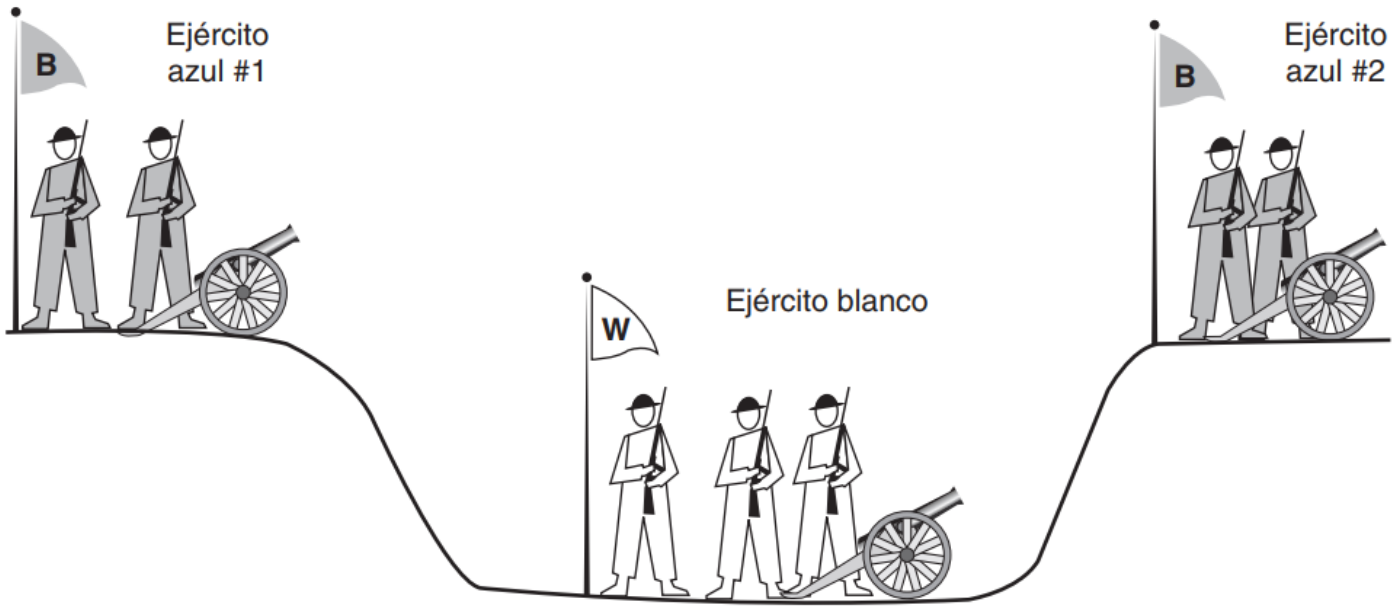
- **Direccionamiento**
  - Si el puerto no pertenece a un protocolo conocido, es necesario "negociar" con el host el puerto de acceso: *portmapper*



[1]

Elementos de protocolos de transporte

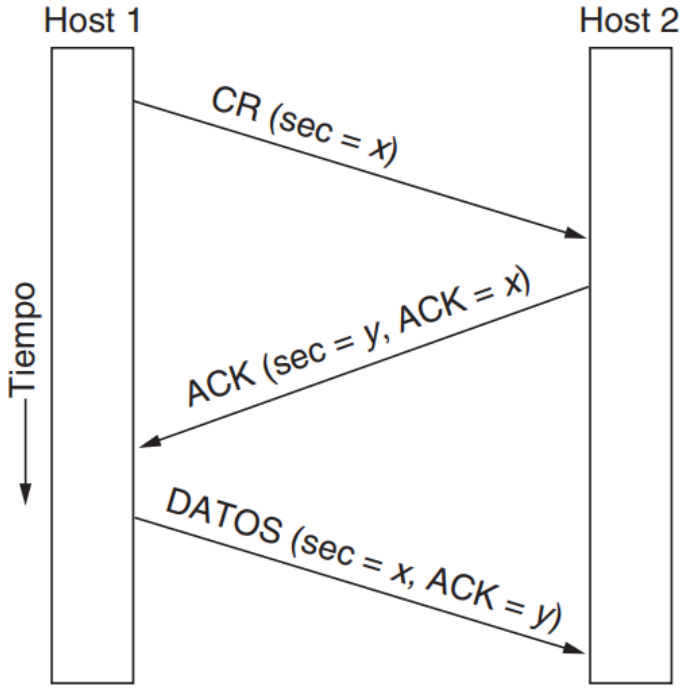
- **Gestión de la conexión**
  - Problema de los dos ejércitos - ¿Podemos asegurar que se nuestras comunicaciones llegan?



[1]

Elementos de protocolos de transporte

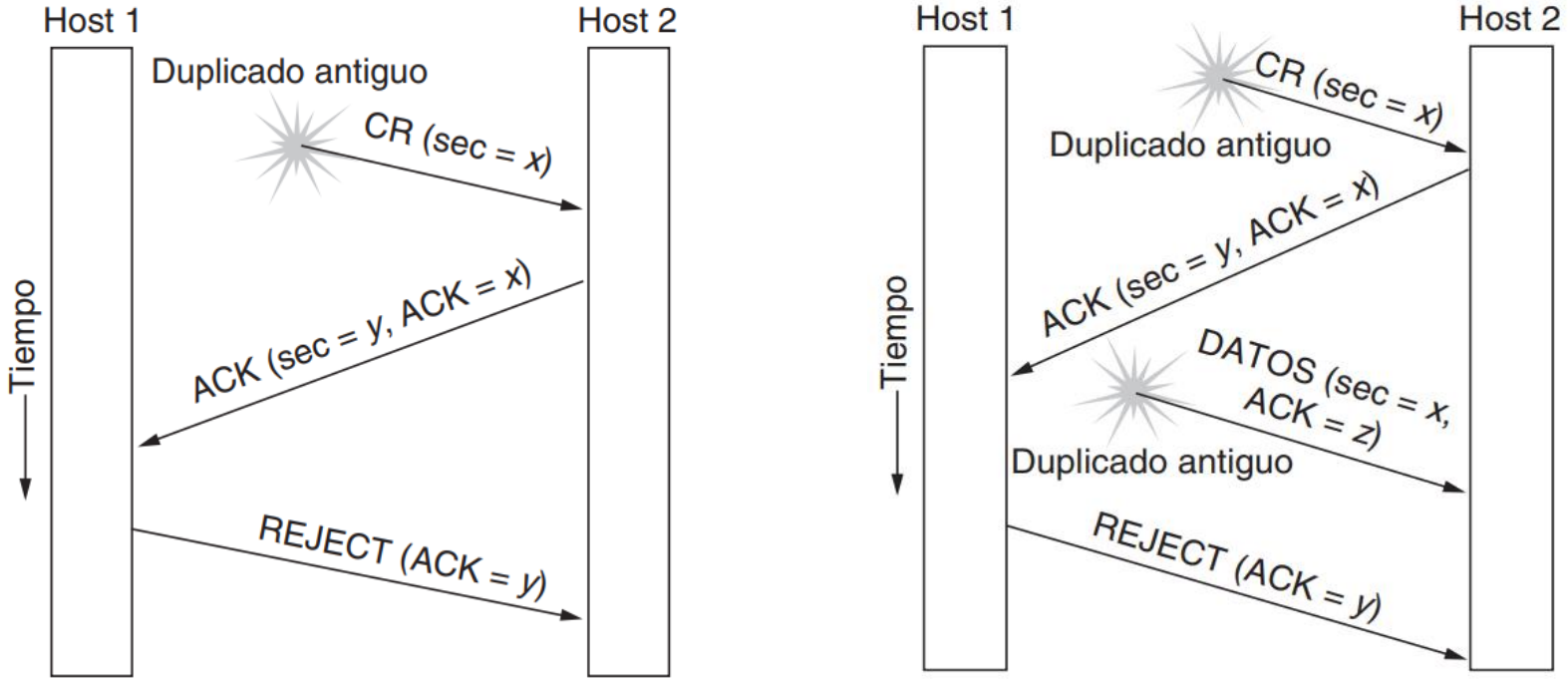
- **Gestión de la conexión**
  - Establecimiento: Gestión de paquetes perdidos o que llegan con retardo
  - Acuerdo de tres vías: *Three-way handshake*
  - Emisor y receptor acuerdan y confirman los números de inicio de secuencia



[1]

Elementos de protocolos de transporte

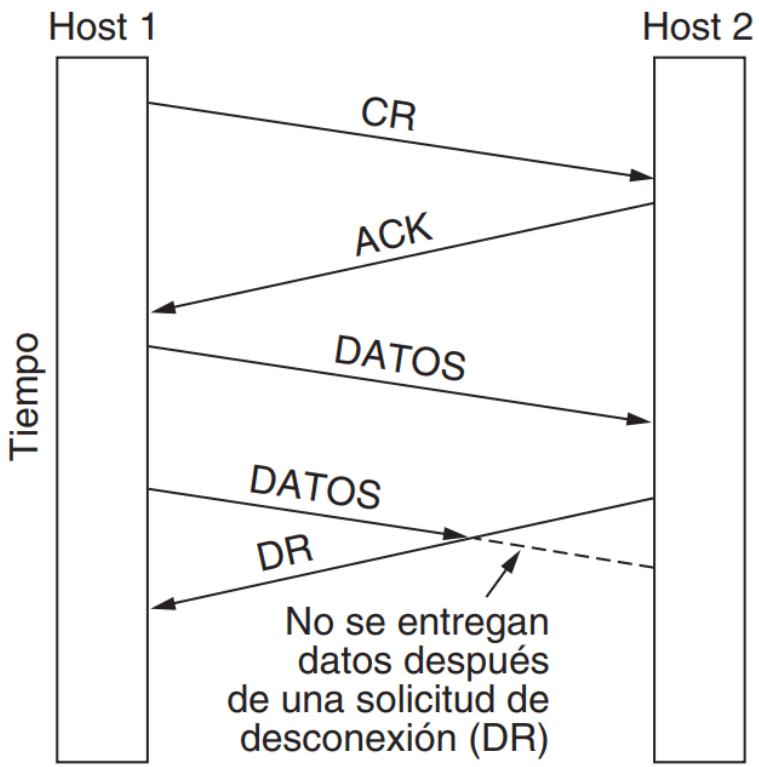
- **Gestión de la conexión**
  - Pérdida o retraso de los paquetes en el establecimiento



[1]

Elementos de protocolos de transporte

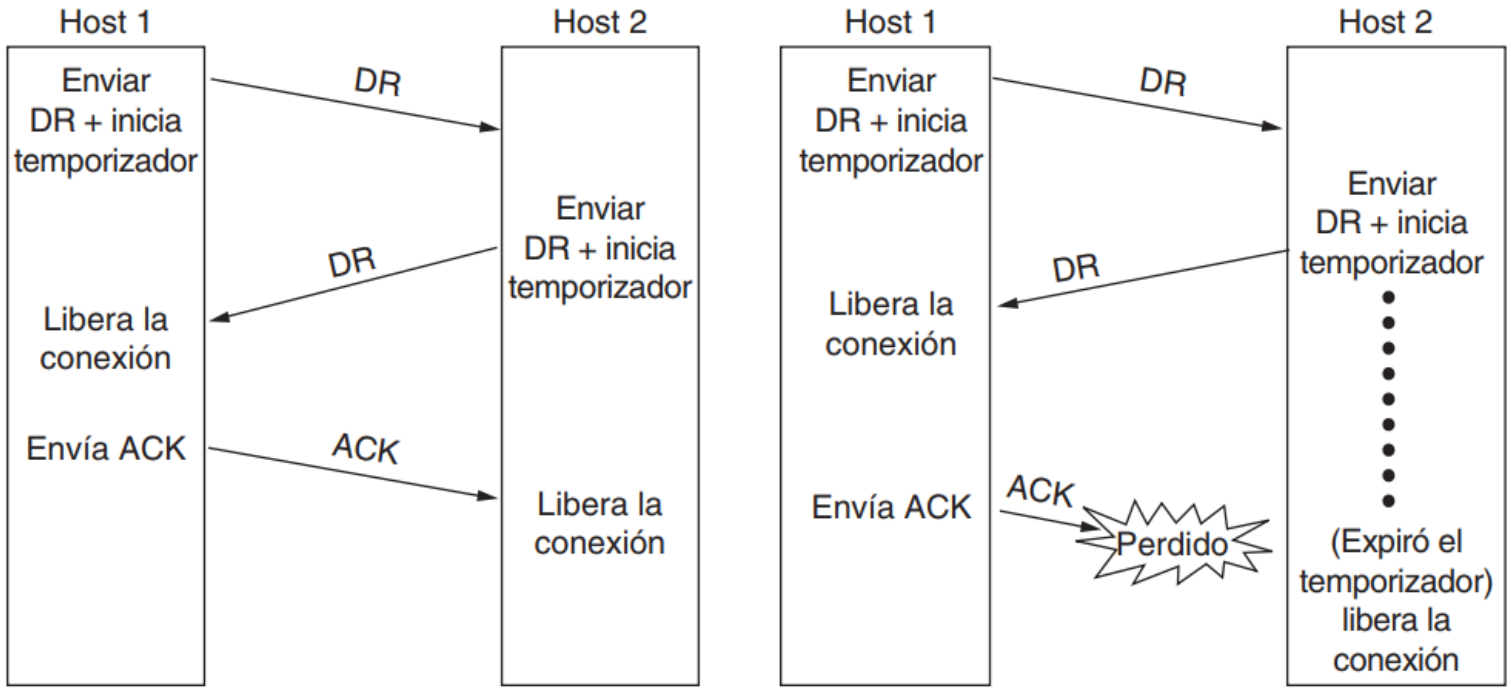
- **Gestión de la conexión**
  - Liberación de la conexión:
    - Asimétrica - Línea telefónica tradicional
    - Simétrica – Evitar pérdidas abruptas de datos



[1]

Elementos de protocolos de transporte

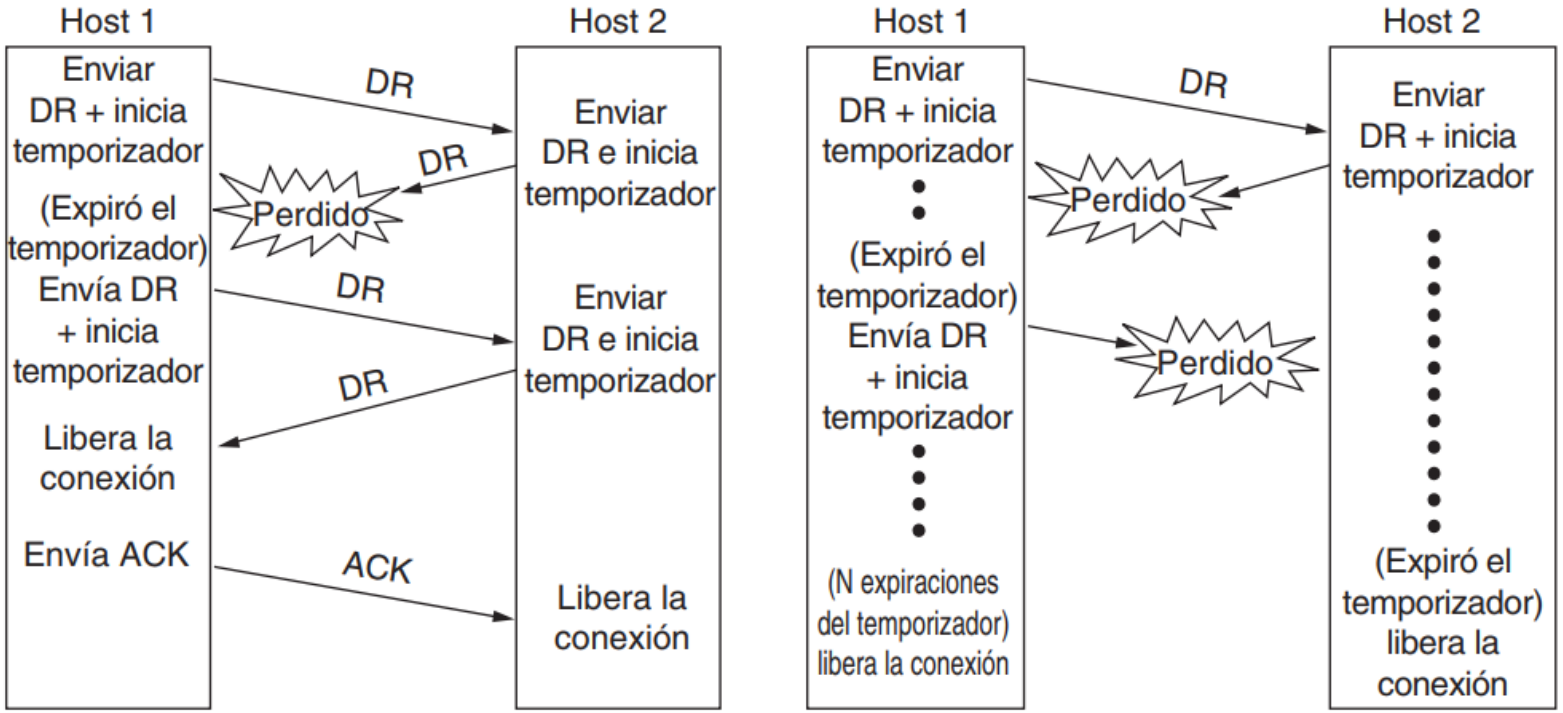
- **Gestión de la conexión**
  - Pérdida o retraso de los paquetes en la liberación



[1]

Elementos de protocolos de transporte

- **Gestión de la conexión**
  - Pérdida o retraso de los paquetes en la liberación



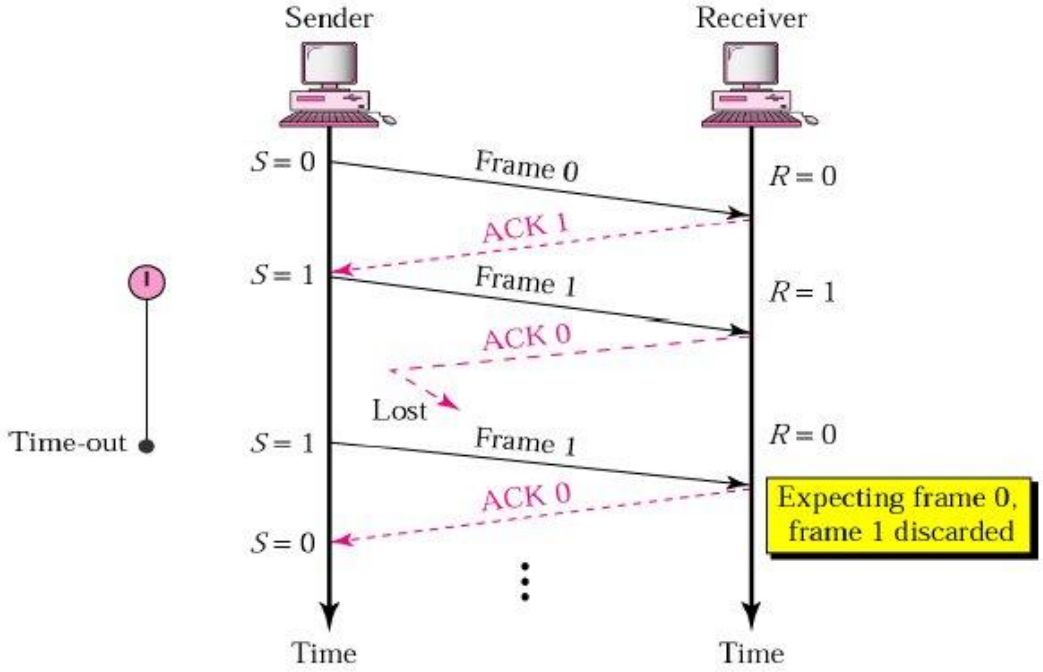
[1]

Elementos de protocolos de transporte

- **Control de congestión**
  - Evitar la saturación del sistema por enviar una cantidad de paquetes mayor de la que admite
  - Tarea compartida por las capas de red y de transporte
  - Principales causas:
    - Ancho de banda y fiabilidad de la red
    - Capacidad del receptor

Elementos de protocolos de transporte

- **Parada y espera**
  - El emisor envía un paquete y espera la confirmación del receptor para enviar el siguiente
  - No son necesarios buffers y únicamente se almacena el último paquete enviado



Elementos de protocolos de transporte

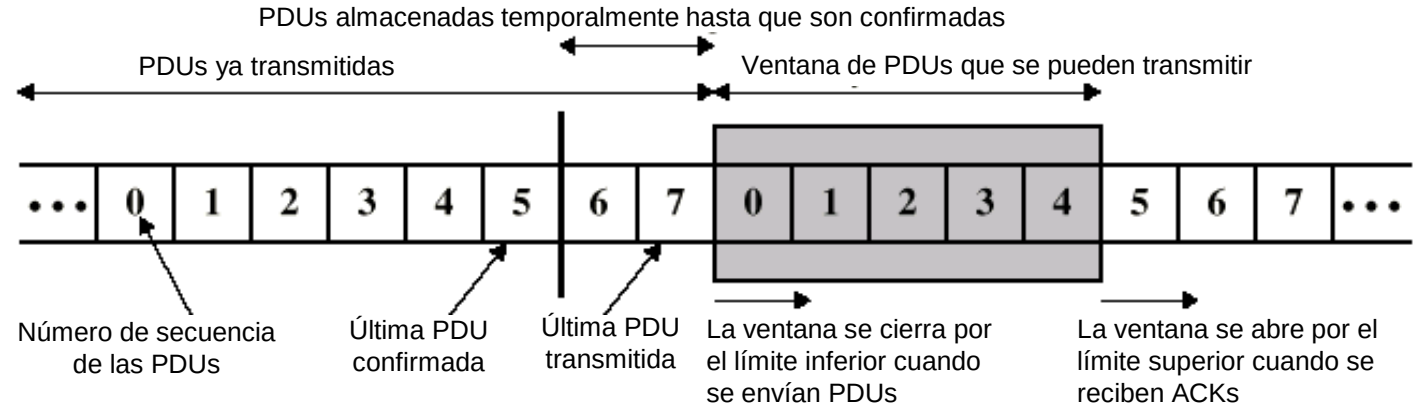
- **Ventana deslizante**
  - El emisor mantiene una lista con los W números de secuencia de los paquetes que puede transmitir → Ventana emisora de tamaño W
  - El receptor mantiene una lista con los W números de secuencia de los paquetes que está autorizado a recibir → Ventana receptora de tamaño W
  - Como los paquetes pueden perderse, el emisor guarda una copia de todos los paquetes que están enviados pero no asentidos por si hay que reenviarlos

Elementos de protocolos de transporte

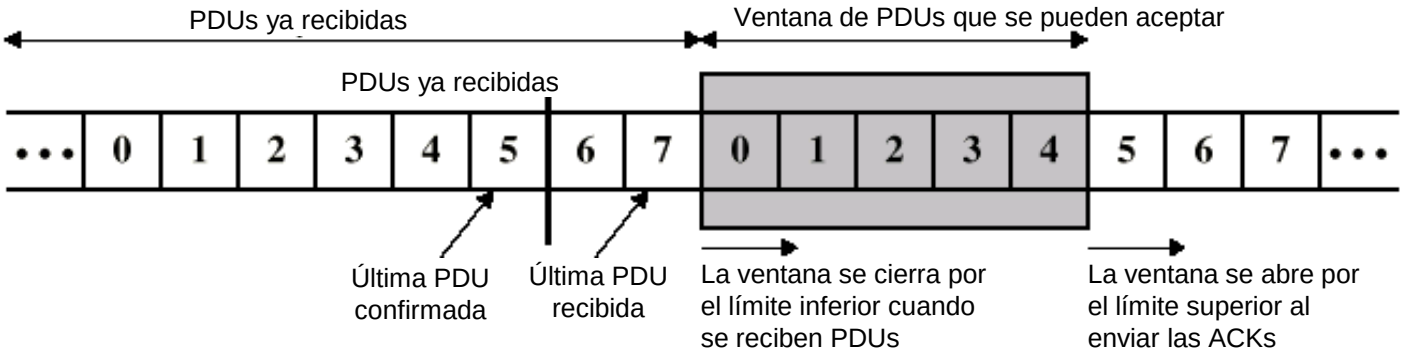
- **Ventana deslizante**
  - Asentimientos:
    - Cada asentimiento puede asentir a un grupo de paquetes o hacerlo de forma individual
    - Controlan el flujo y notifican el resultado de la transmisión de un paquete
    - Indican el número de paquete que se espera en la siguiente transmisión

Elementos de protocolos de transporte

- **Ventana deslizante**



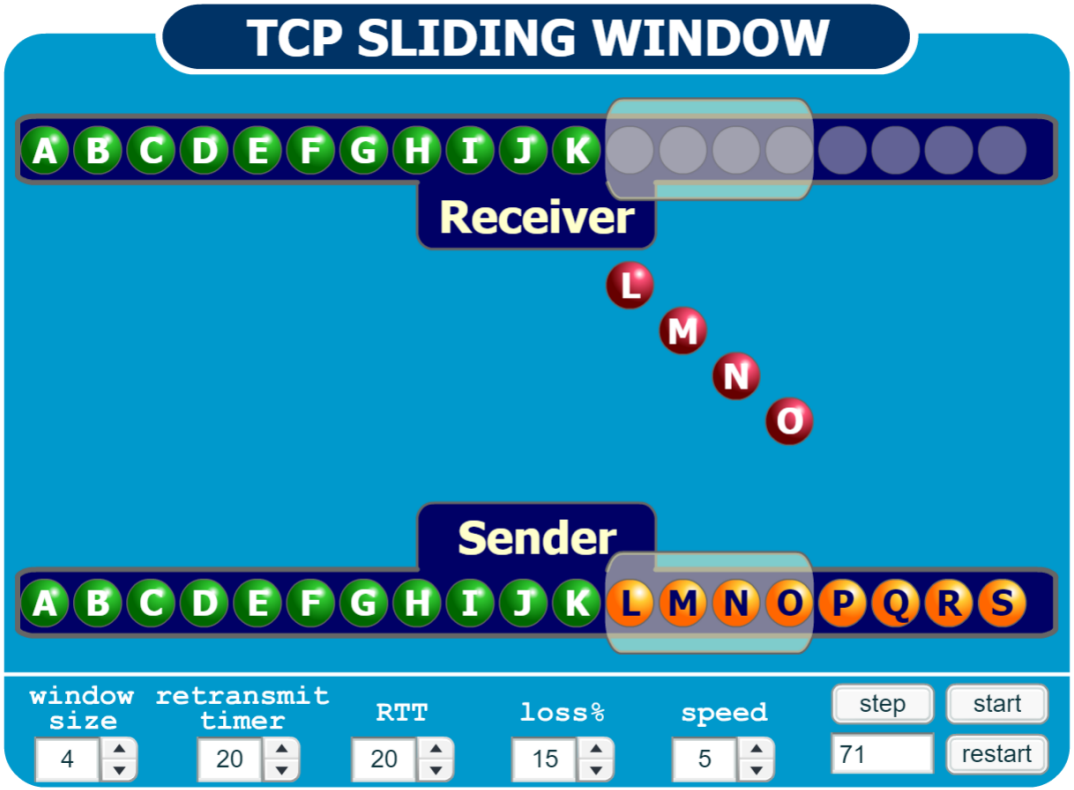
(a) Desde el punto de vista del transmisor



(b) Punto de vista del receptor

Elementos de protocolos de transporte

- **Ventana deslizante**
  - **http://www2.rad.com/networks/2004/sliding_window/**



3.- EL PROTOCOLO UDP

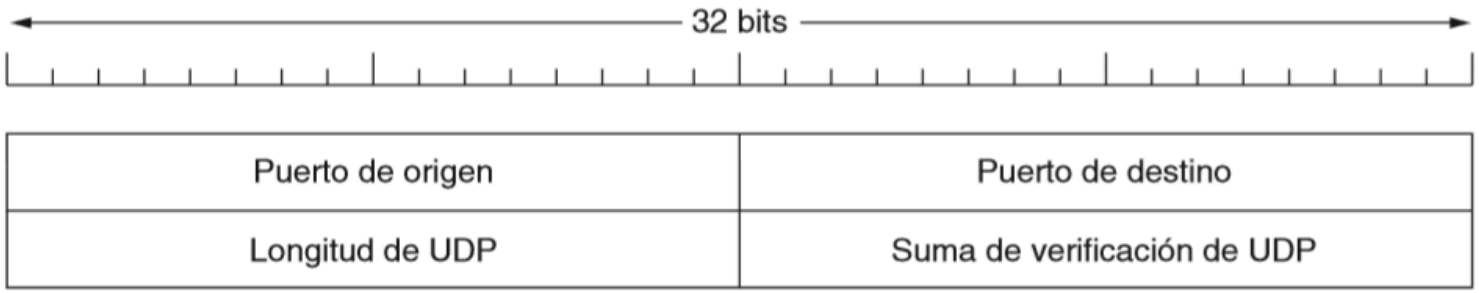
- *User Datagram Protocol*
- Protocolo no orientado a conexión
  - Cada segmento se trata de forma independiente de los demás
- Es un protocolo no fiable → ofrece un servicio "best effort"
  - Sus mensajes pueden llegar fuera de secuencia o perderse
- No se envían asentimientos: se reduce el tráfico de la red
- No controla la congestión
- Reduce la información suplementaria a enviar

El protocolo UDP

- Proporciona interfaz intermedia entre la capa de aplicación y la de red
  - Gestión del uso de los puertos
  - Puede proporcionar control de errores
- Adecuado para situaciones con requisitos de conexión bajos
  - Servicio DNS
  - Vídeo bajo demanda
  - Radio en Internet
  - Telefonía en Internet
  - Algunos modelos cliente-servidor

El protocolo UDP

- **Cabecera UDP**

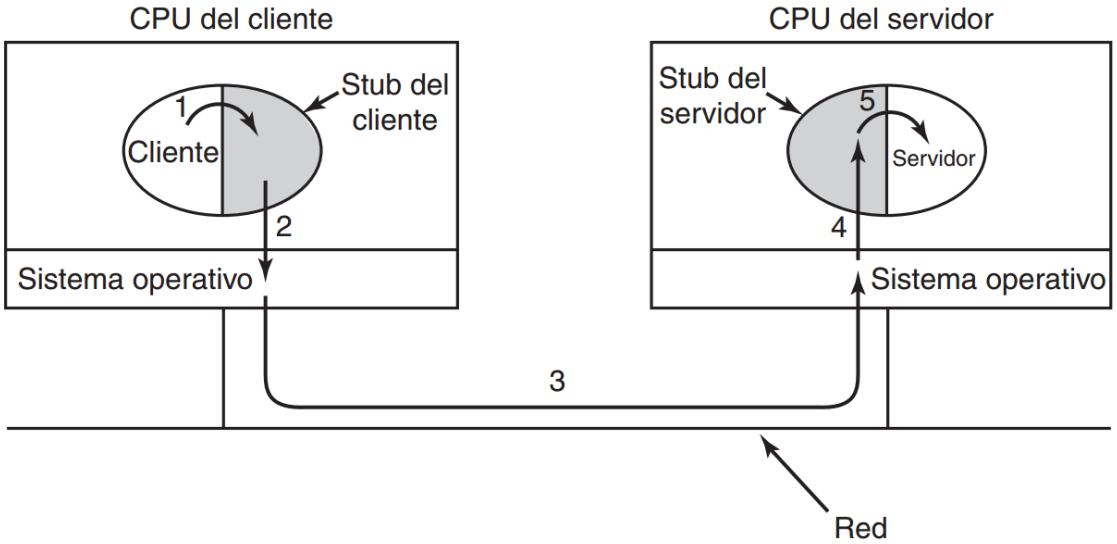


[1]

  - Puerto de origen: contiene el número de puerto por si es necesario responder al origen
  - Puerto de destino: contiene el número de puerto del destino
  - Longitud: longitud de los datos del datagrama IP
  - Suma de comprobación: asegura la integridad del datagrama. Se calcula utilizando la cabecera UDP y el campo de datos

El protocolo UDP

- ***Remote Procedure Call* (RPC)**
  - Hacer que una llamada a un procedimiento remoto sea parecida a una a un procedimiento local



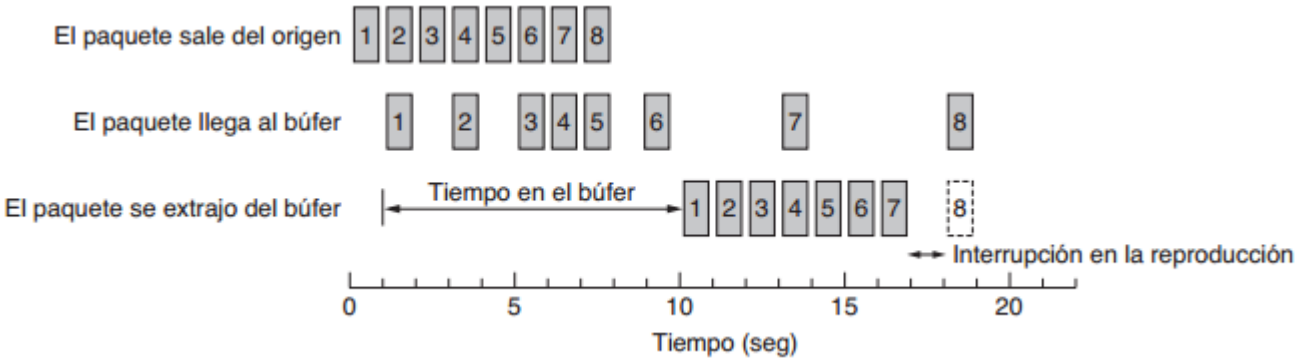
[1]

El protocolo UDP

- ***Real-time Transport Protocol*** **(RTP)**
  - Está ubicado justo por encima de UDP en la capa de transporte
  - Se suele utilizar en la transmisión de paquetes de audio y vídeo en tiempo real
  - Puede ser unidifusión o multidifusión
  - Los números de paquetes son incrementales y consecutivos

El protocolo UDP

- ***Real-time Transport Protocol* (RTP)**
  - Puede transmitir información relacionada a través de varios flujos
  - Utiliza las estampas de tiempo (*timestamping*) para sincronizar los diferentes flujos y reducir la variación de retardo o *jitter*
  - Empleo de *buffers* para el control del tráfico



[1]

4.- EL PROTOCOLO TCP

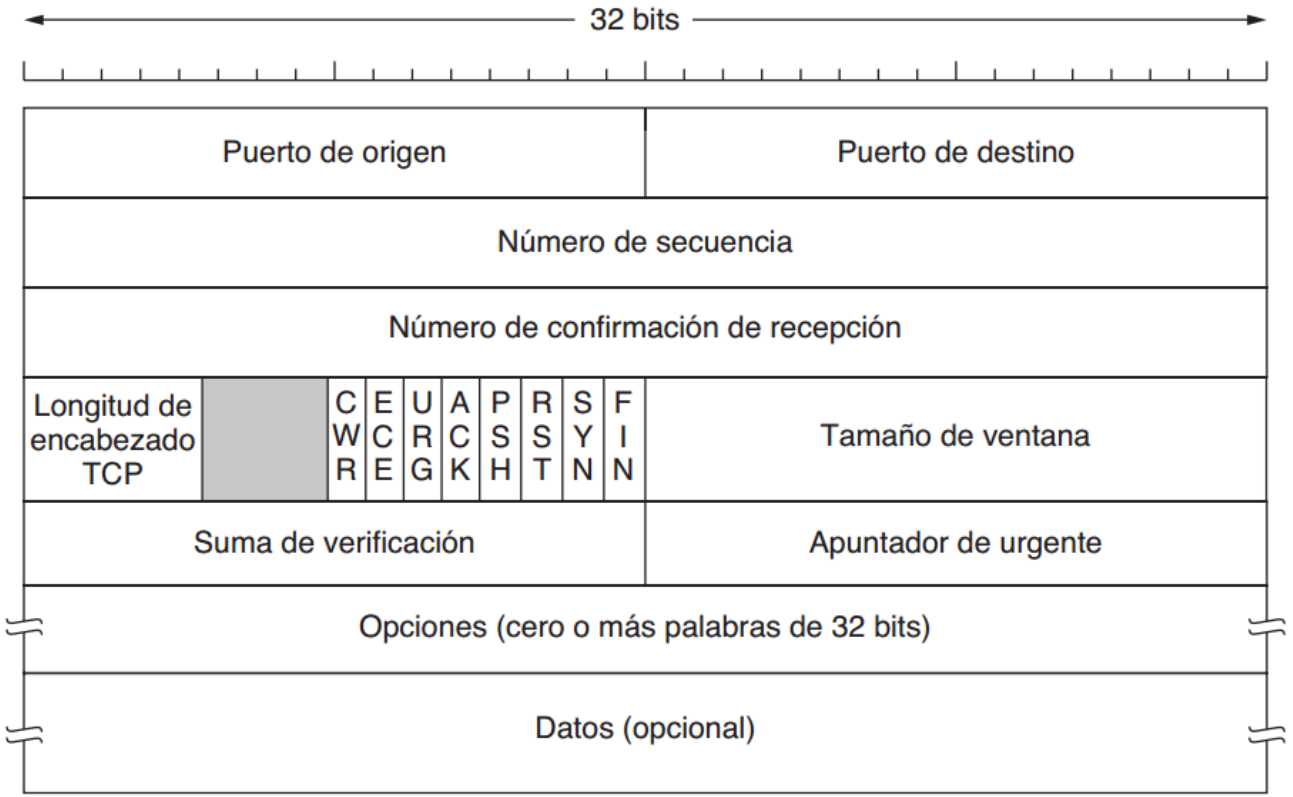
- *Transmission Control Protocol*
- Protocolo orientado a conexión:
  - 3 fases: establecimiento de conexión, transferencia, cierre de conexión
- Proporciona una capa fiable por encima del protocolo IP:
  - Se utilizan asentimientos *(ACK)*
  - Solicita reenvíos
- Se encarga de fragmentar la información que recibe del nivel superior
  - Tamaños máximos de 64 KB
  - Habitualmente 1460 bytes

El protocolo TCP

- Emplea puertos que son llamados a través de *sockets*
- Utiliza un sistema de ventana deslizante para el control de flujo a este nivel
- Utiliza *buffers* para la transferencia haciéndola más eficiente
  - Acumula datos hasta que tiene suficientes para llenar un datagrama
  - También se puede forzar el envío
- Se intercambian flujos de bytes, divididos en segmentos
- Realiza control de la congestión a nivel de transporte. Se necesitan algoritmos diferentes a los utilizados en niveles más bajos

El protocolo TCP

- **Cabecera TCP**



[1]

El protocolo TCP

- **Cabecera TCP**
  - Puerto de origen y destino: contiene los números de los puertos de envío y recepción
  - Número de secuencia: identifica el número de secuencia del primer byte de datos del segmento. Si es un segmento SYN, es el número de secuencia inicial
  - Nº confirmación de recepción: indica el número del siguiente *byte* que se desea recibir, no el último *byte* recibido
  - Longitud encabezado: cantidad de palabras de 32 bits incluidas en el encabezado
  - Campo reservado para posibles usos

El protocolo TCP

- **Cabecera TCP**
  - *CWR (Congestion Window Reduced)*: bit para indicar reducción del tamaño de la ventana
  - *ECN (Explicit Congestion Notification)*: identificador que se utiliza para indicar que se está congestionando la red
  - *URG (Urgent)*: utilizado para indicar que el valor del campo "apuntador de urgente" es válido
  - *ACK (Acknowledgment)*: se utiliza para indicar que la respuesta también confirma datos recibidos
  - *PSH (Pushed Data)*: indica la entrega inmediata de los datos al nivel superior. No se espera a que se llene el buffer

El protocolo TCP

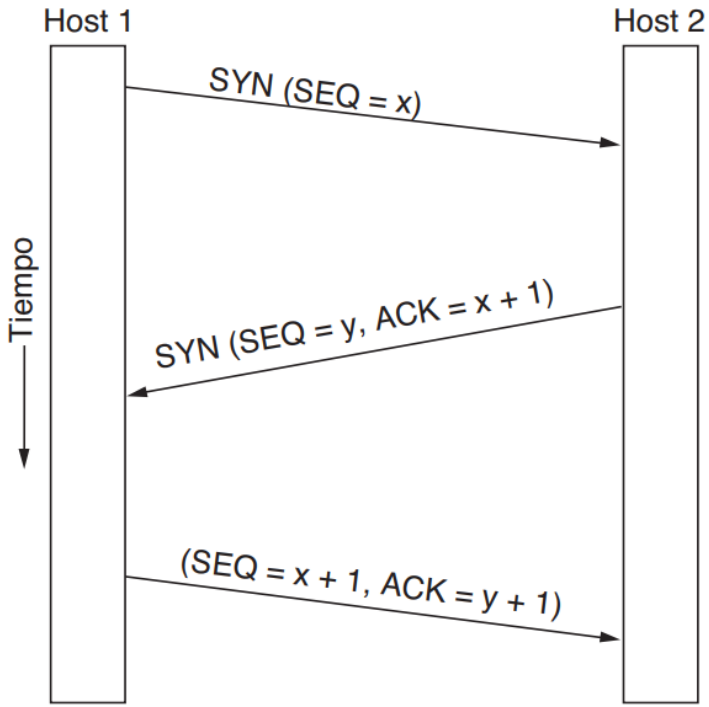
- **Cabecera TCP**
  - *RST (Reset)*: empleado para reiniciar la conexión
  - *SYN (Synchronize)*: se utiliza para establecer la conexión. Únicamente los primeros mensajes tendrían este bit a 1
  - FIN: corta la conexión y es el último mensaje enviado por cada transmisor
  - Tamaño de ventana: indica el tamaño de la ventana. Puede ser igual a 0
  - Suma de verificación: sirve para comprobar que el mensaje se ha transmitido sin errores
  - Apuntador de urgente: es un *offset* que permite conocer el último valor de *byte* de los datos urgentes

El protocolo TCP

- **Cabecera TCP**
  - *Options*: permite definir nuevas opciones que no estén entre las incluidas por defecto en la cabecera
    - Tamaño máximo del segmento
    - Escala de ventana
    - Estampa de tiempo
    - Selective ACK

El protocolo TCP

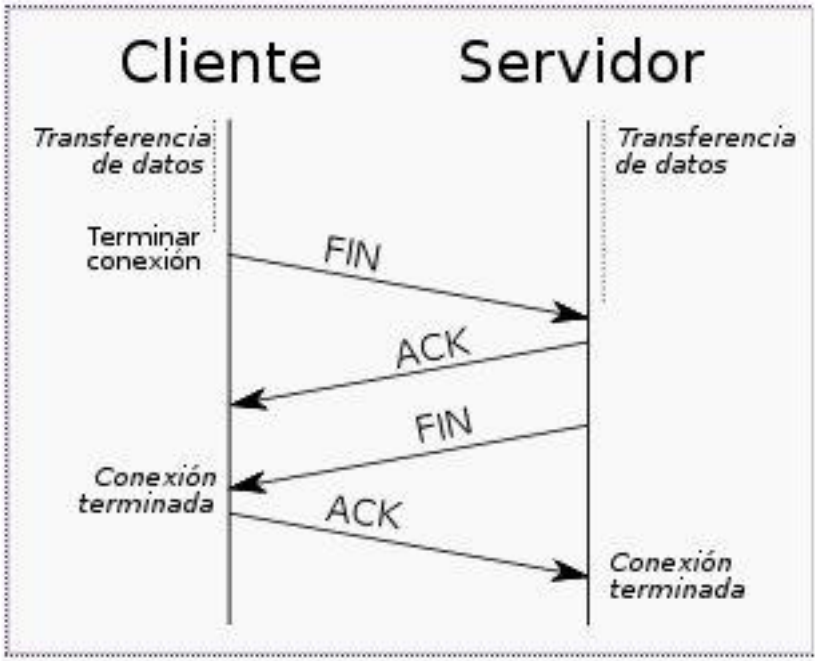
- **Establecimiento de conexión**
  - *Handshake* de triple vía



[1]

El protocolo TCP

- **Cierre de conexión**
  - *Handshake* de cuatro vías



El protocolo TCP

- **Fiabilidad en el protocolo TCP**
  - Pérdida de segmentos:
    - Los segmentos tienen número de secuencia
    - Se responderá a la llegada de segmentos correctos mediante asentimientos (ACK)
    - Los asentimientos hacen referencia al flujo de bytes recibidos, no a segmentos individuales
    - Se utilizarán temporizadores para controlar la pérdida de tramas: retransmisión
  - Duplicados:
    - Cuando TCP considera que se ha perdido un segmento enviará un duplicado
    - El receptor detectará el doble envío gracias al número de secuencia y descartará la trama

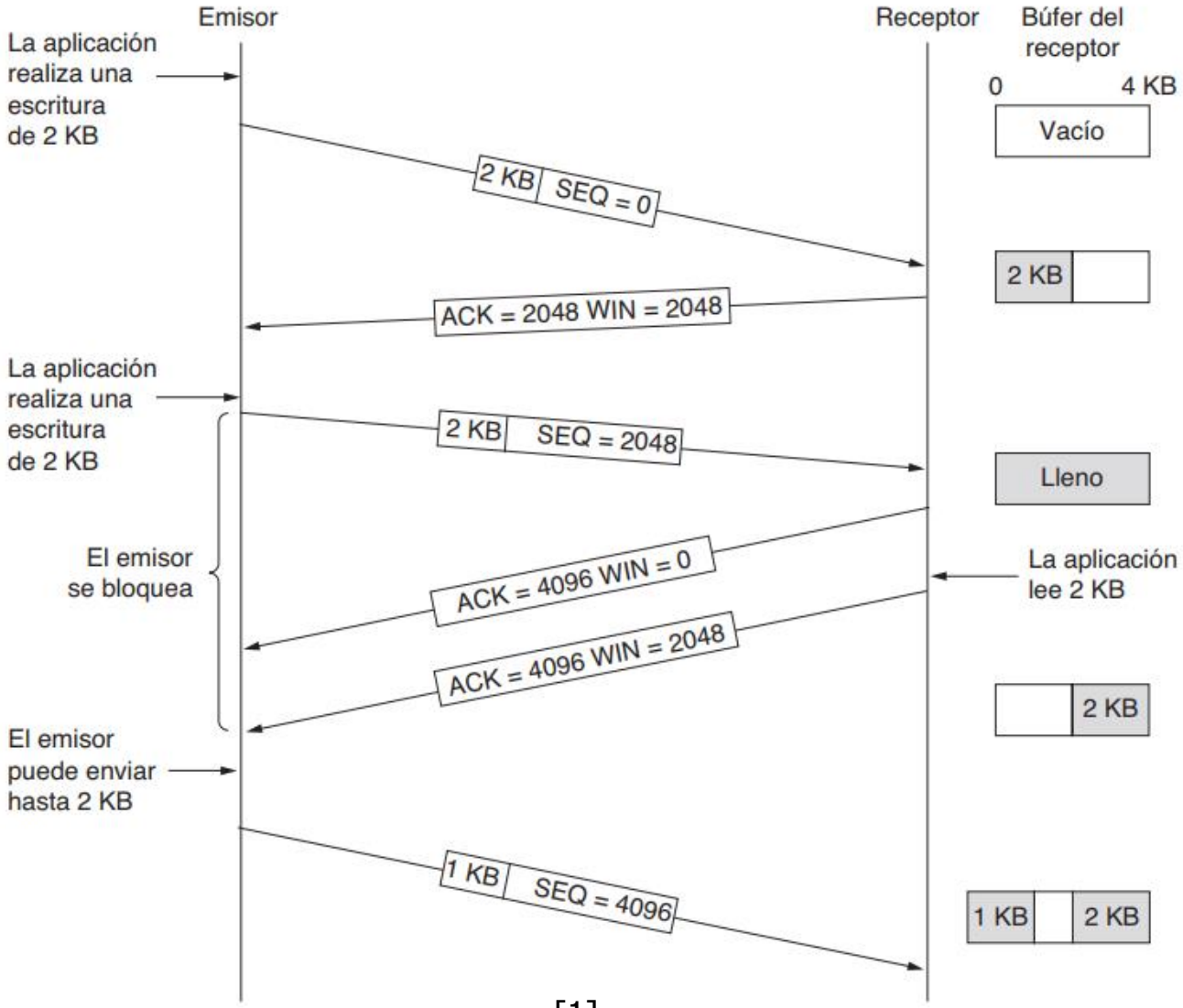
El protocolo TCP

- **Fiabilidad en el protocolo TCP**
  - Eficiencia y control de flujo:
    - Se utiliza un sistema de ventana deslizante para gestionar el flujo
    - Se utiliza un tamaño de ventana variable controlado por el receptor
    - Se utiliza el sistema de superposición para el ahorro de ancho de banda consumido por los ACKs
  - Control de errores:
    - Entrega los datos sin errores
    - Suma de comprobación

El protocolo TCP

- **Control del flujo mediante ventana deslizante**
  - La ventana es de tamaño variable y está controlada por el receptor
  - No controla el número de segmentos recibidos, sino el número de bytes
  - Ventana del emisor: número de bytes que puede enviar sin recibir asentimiento
  - Ventana del receptor: número de bytes que puede aceptar
  - Las respuestas transportan el número de bytes recibidos correctamente y el tamaño de la ventana receptora, que puede aumentar o disminuir
  - Se pueden realizar asentimientos acumulativos con el objetivo de reducir el ancho de banda utilizado

El protocolo TCP



[1]

El protocolo TCP

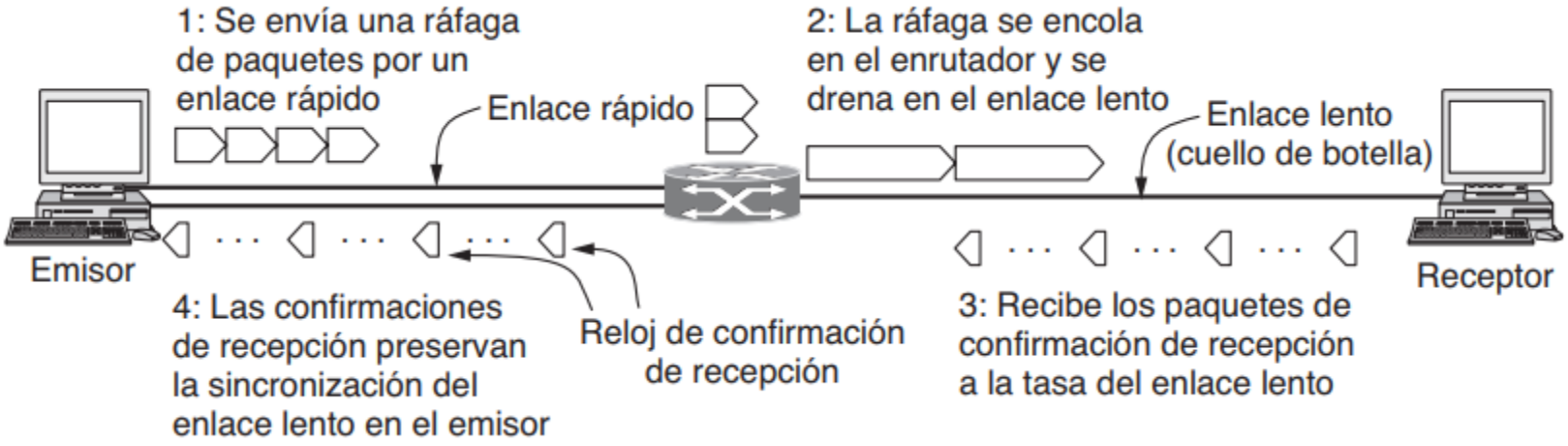
- **Control del flujo mediante ventana deslizante**
  - Los datos con el flag *URG* siempre pueden enviarse
  - Si la ventana está llena, puede enviarse un segmento de tamaño 1 *byte*
  - Algoritmo de Naggle: Adecuado para situaciones de envío con paquetes pequeños
    - Se envía el primer segmento de información que llegue
    - La nueva información se almacena en un *buffer* hasta que llegue la confirmación del anterior segmento
    - Reducir el gasto de ancho de banda por culpa de las cabeceras

El protocolo TCP

- **Control de la congestión**
  - El reloj de confirmación de recepción *(ack clock)*
  - Utilización de temporizadores para evitar sobrecargar la red
  - Ventana de congestión
  - Algoritmos de control:
    - Inicio lento
    - Retransmisión rápida
    - Recuperación rápida
    - Asentimientos selectivos

El protocolo TCP

- **Control de la congestión – *Ack clock***
  - La velocidad de la red por la que se emite, está limitada por su enlace más lento
  - El emisor necesita adaptar su velocidad a la máxima permitida por dicho enlace
  - Se utiliza el llamado reloj de confirmación de recepción o *ack clock*

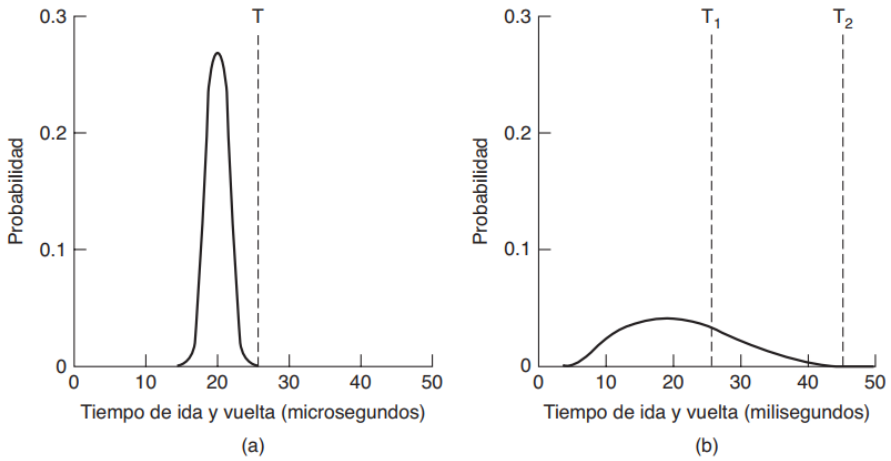


[1]

El protocolo TCP

- **Control de congestión - Temporizadores**
  - *Retransmission TimeOut (RTO)*
    - Tiempo que se espera antes de reenviar un segmento

RTO = Tiempo medio ida y vuelta + 4 · desviación media



[1]

El protocolo TCP

- **Control de congestión - Temporizadores**
  - Temporizador de Persistencia
    - El receptor envía un ACK con tamaño de ventana 0
    - Cuando actualiza el tamaño de ventana, el paquete se pierde
    - El emisor envía un mensaje de sondeo para forzar que el receptor le confirme el tamaño de la ventana
  - Temporizador *Keep Alive*
    - Después de tiempo sin mensajes, una de las partes envía un mensaje vacío para confirmar que el otro extremo sigue activo

El protocolo TCP

- **Control de congestión – Ventana congestión**
  - Es el máximo número de bytes que el emisor puede poner en la red
  - Funciona en paralelo con la ventana deslizante del control de flujo – El valor más pequeño de ambas se corresponde con el valor de la ventana que se vaya a utilizar

El protocolo TCP

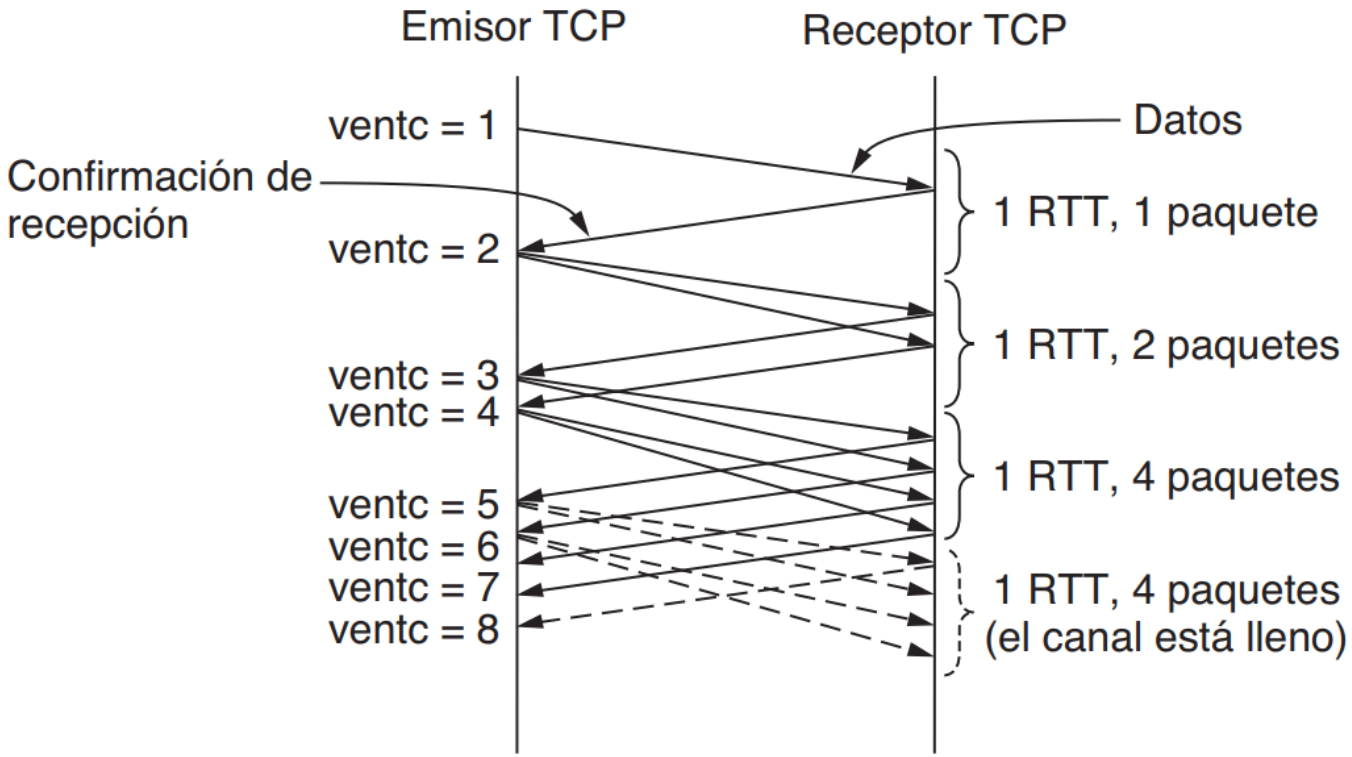
- **Control de congestión – Ventana congestión**
  - Hay que obtener su valor óptimo para evitar saturar la red
  - El valor ideal puede variar y es necesario que la ventana se adapte a dicho tamaño
  - Se intentan utilizar reglas AIMD *(Additive Increase Multiplicative Decrease)*

El protocolo TCP

- **Control de congestión – Inicio lento**
  - Al inicio de la transmisión, se envía un único segmento
  - Una vez que llega correctamente la confirmación, se envían dos segmentos
  - Cuando llegan nuevamente las confirmaciones, se duplica de nuevo el tamaño de la ventana – cuatro segmentos
  - La operación se repite hasta que ocurra algún evento que indique que hay congestión en la red
  - Incremento exponencial – La ventana de congestión puede crecer muy rápido

El protocolo TCP

- **Control de congestión – Inicio lento**



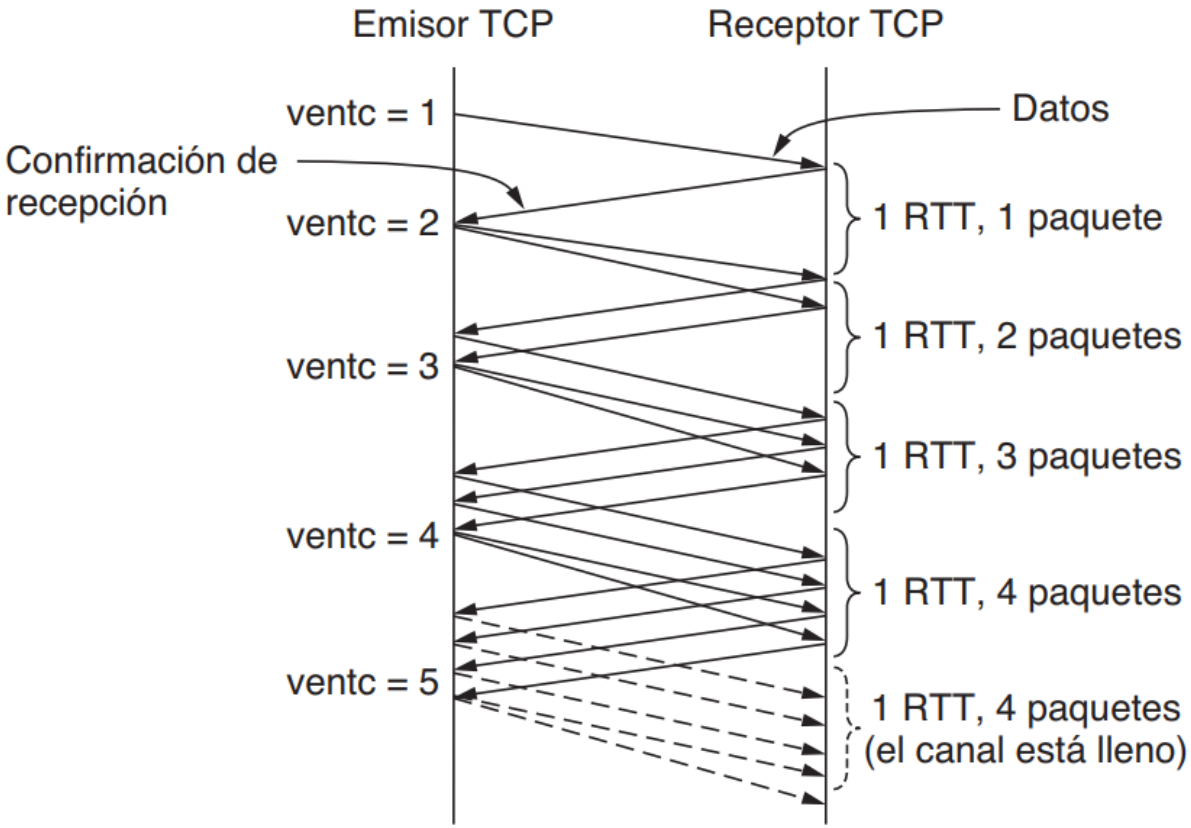
[1]

El protocolo TCP

- **Control de congestión – Inicio lento**
  - Un crecimiento excesivamente rápido, hace que sea muy difícil encontrar el tamaño de ventana ideal
  - Se puede establecer un umbral de inicio lento, a partir del cual el incremento pasa a ser lineal y no exponencial
  - Cada vez que llegan todas las confirmaciones, el tamaño de la ventana se incrementa en un solo segmento en lugar de duplicarse
  - Este umbral va aumentando cada vez que aumenta el tamaño de la ventana
  - Esto permite encontrar de una forma más precisa el tamaño ideal de la ventana

El protocolo TCP

- **Control de congestión – Inicio lento**



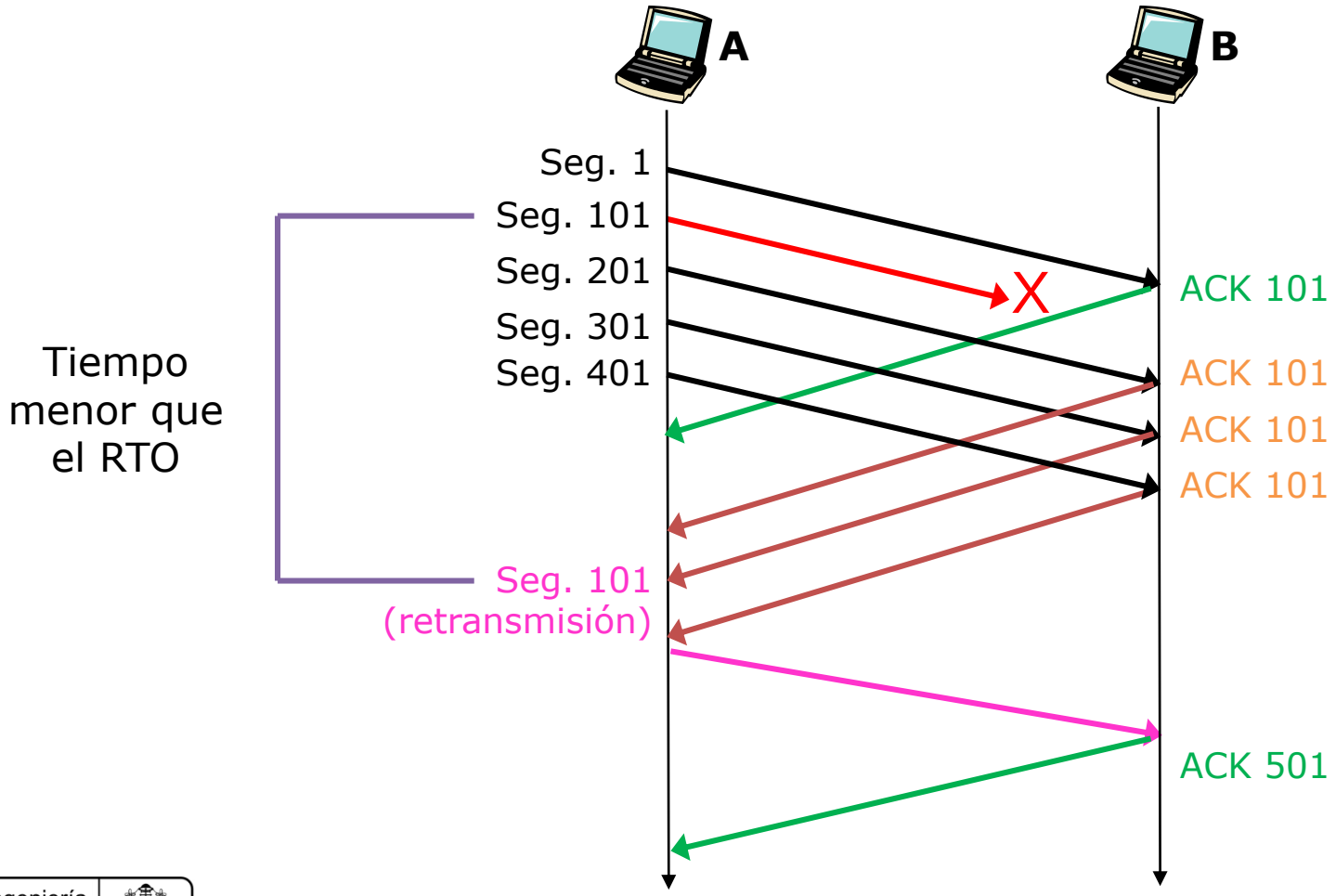
[1]

El protocolo TCP

- **Control de congestión – Pérdida de paquetes**
  - ¿Cómo detectar que se pierde un paquete?
    - Salta uno de los temporizadores RTO – Se considera que el paquete se ha perdido o que llegará demasiado tarde
    - Se reciben tres asentimientos repetidos
      - Están llegando segmentos nuevos al receptor, pero falta uno de los anteriores
      - El emisor no espera a que salte el RTO para enviar de nuevo el paquete, lo reenvía al recibir el tercer ACK repetido
      - Retransmisión rápida

El protocolo TCP

- **Control de congestión – Pérdida de paquetes**



El protocolo TCP

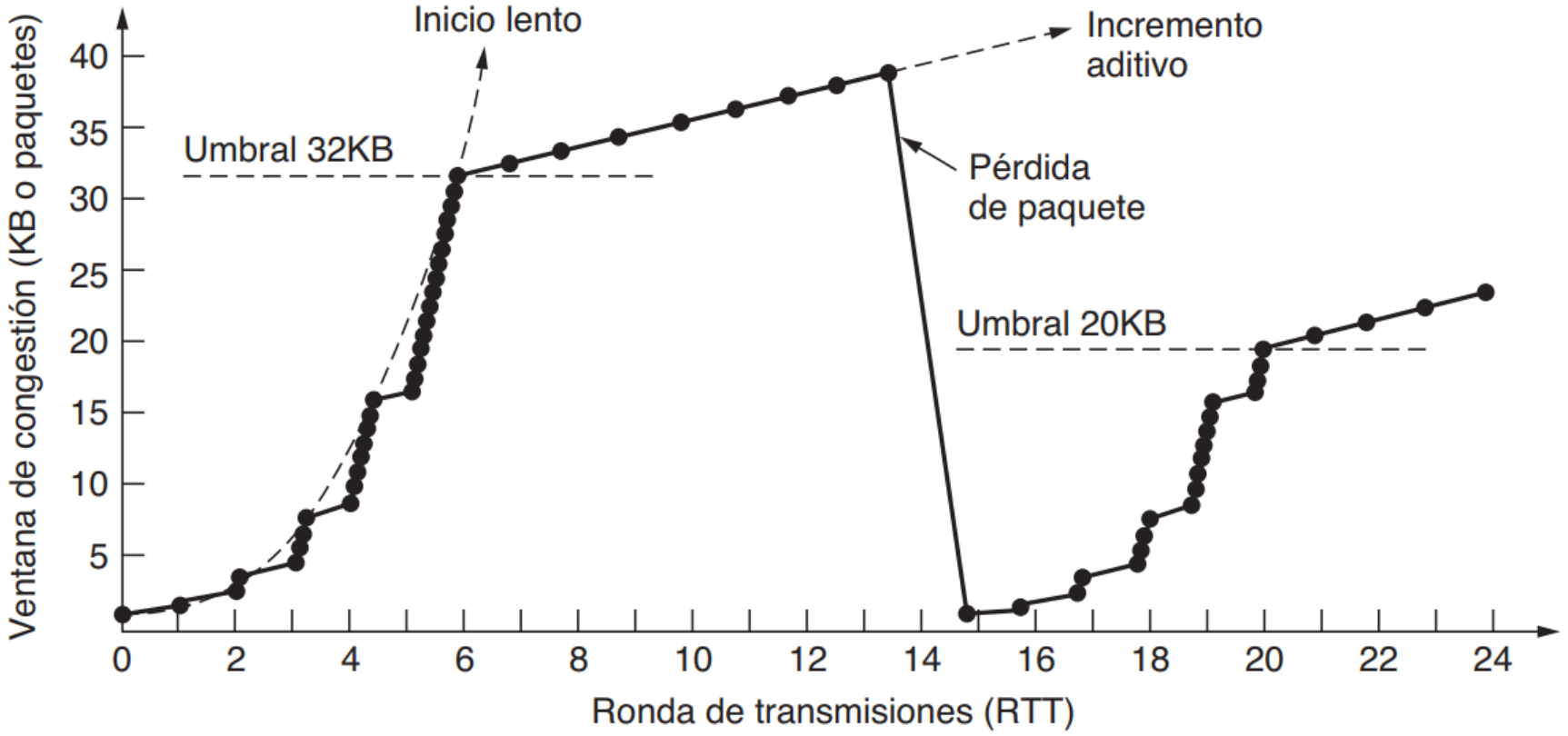
- **Control de congestión – Pérdida de paquetes**
  - ¿Cómo actuar cuando se pierde un paquete?
    - Reiniciar el valor de la ventana de congestión
    - Dividir entre dos el valor del umbral de inicio lento
    - Repetir el proceso para ir aumentando el valor de la ventana hasta que pueda volver a aparecer congestión

El protocolo TCP

- **Control de congestión – TCP Tahoe**
  - Implementa inicio lento
  - Utiliza umbral de inicio lento
  - Detecta pérdida de paquetes mediante RTO y ACKs repetidos
  - Cuando se pierde un paquete, reinicia el valor de la ventana de congestión a un segmento y el umbral de inicio lento a la mitad del valor actual

El protocolo TCP

- **Control de congestión – TCP Tahoe**



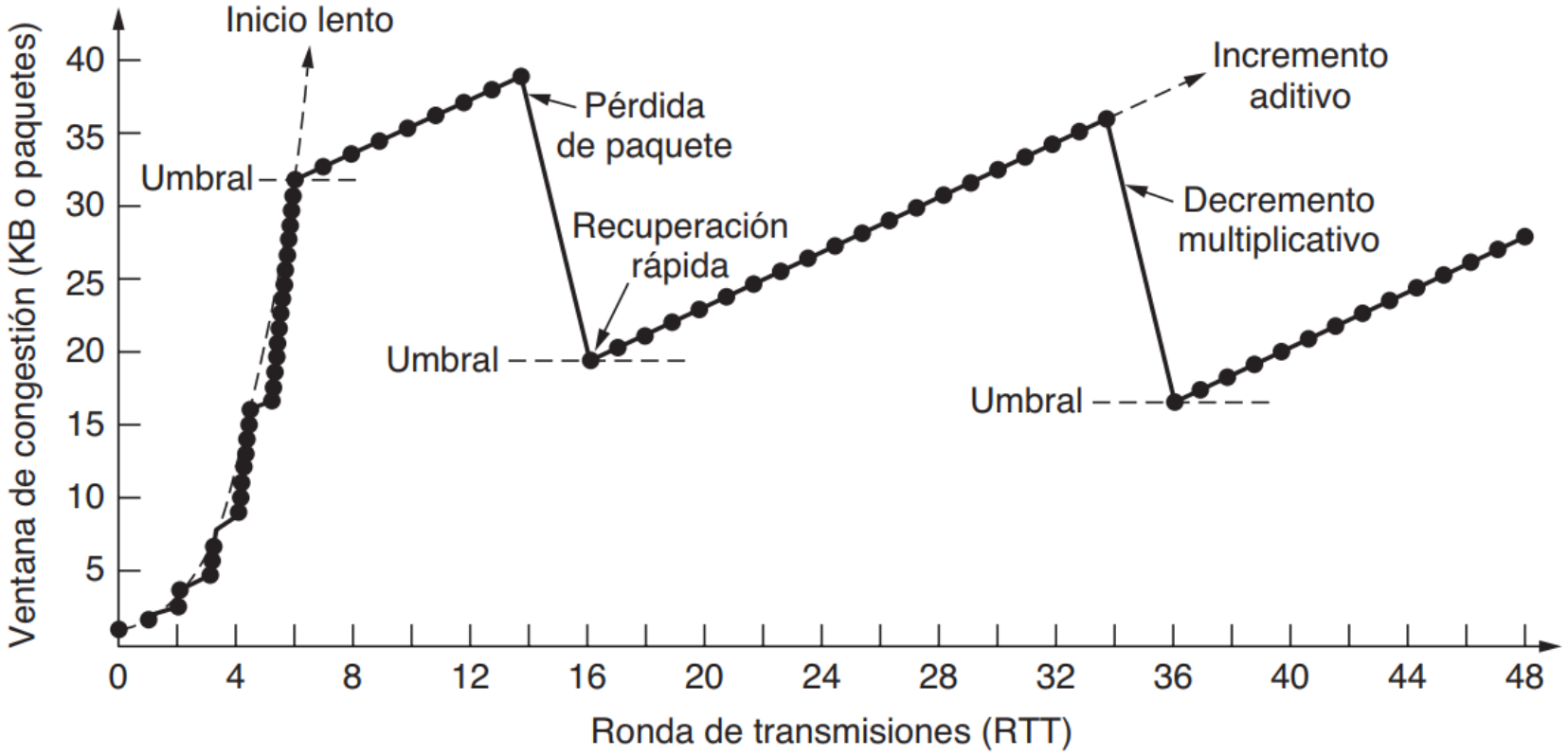
[1]

El protocolo TCP

- **Control de congestión – Recuperación rápida**
  - Se detecta que hay congestión en la red
  - El valor de la ventana de congestión se reinicia
  - No se utiliza una ventana de tamaño uno, sino una nueva ventana con la mitad del tamaño que la actual
  - Como el umbral de inicio lento tiene ese valor, los nuevos incrementos son lineales, no exponenciales
  - Algoritmo TCP Reno

El protocolo TCP

- **Control de congestión – TCP Reno**



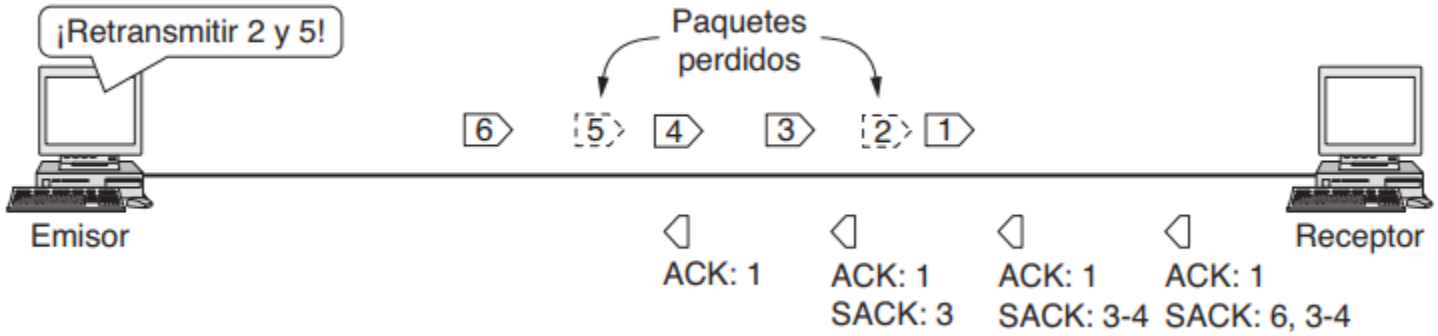
[1]

El protocolo TCP

- **Control de congestión – Asentimiento selectivo**
  - El campo ACK de la cabecera, indica el último paquete que se ha recibido en orden y correctamente
  - Mediante el campo "*options*" se pueden hacer asentimientos selectivos de tramas que llegan fuera de orden
  - Se pueden agrupar paquetes consecutivos que puedan haber llegado fuera de orden

El protocolo TCP

- **Control de congestión – Asentimiento selectivo**
  - Ayuda en la velocidad de recuperación ante pérdidas, pero es un complemento a las técnicas anteriores



[1]

El protocolo TCP

- **Problemas y futuro**
  - Desarrollado en los 80, apenas ha sufrido cambios significativos
  - El aumento de las velocidades de las redes ha supuesto un problema importante
  - Debido a su amplia implementación, es muy complicado cambiarlo por nuevos protocolos
  - El control de la congestión aún debe ser mejorado

Referencias

– [1] Redes de ordenadores, 5ª Ed., Andrew S. Tanenbaum, Prentice Hall