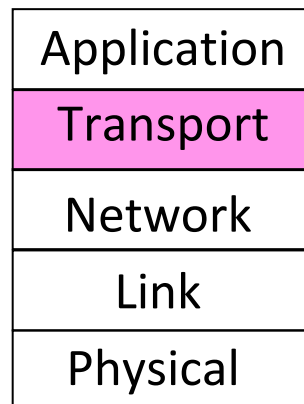


Capítulo 3

Capa de Transporte Generalidades e introducción a TCP



Capa de transporte

- **Propósito de la capa de transporte**

- La **capa de transporte (CT)** provee comunicación lógica **entre procesos** de aplicación que ejecutan en diferentes sistemas finales.
 - esto no lo puede hacer la capa de red – CR.
 - La CT se implementa (salvo alguna excepción que veremos más adelante) solo en los sistemas finales.
- **Comunicación lógica**: como si los hosts ejecutando los procesos estuvieran directamente conectados.
- Para **mejorar la calidad** los servicios de la CR.
 - P.ej: retransmisiones de paquetes perdidos en redes no orientadas a la conexión.
 - P.ej: cuando hay congestión en la red, regulando de manera fina la variación de la tasa de transmisión de paquetes de los hosts.

Capa de transporte

- La CT se ejecuta por completo en los hosts/sistemas finales.
- La CT **confía** en los servicios de la CR.
- **Entidad de transporte (ET)** = software/hardware de la CT.

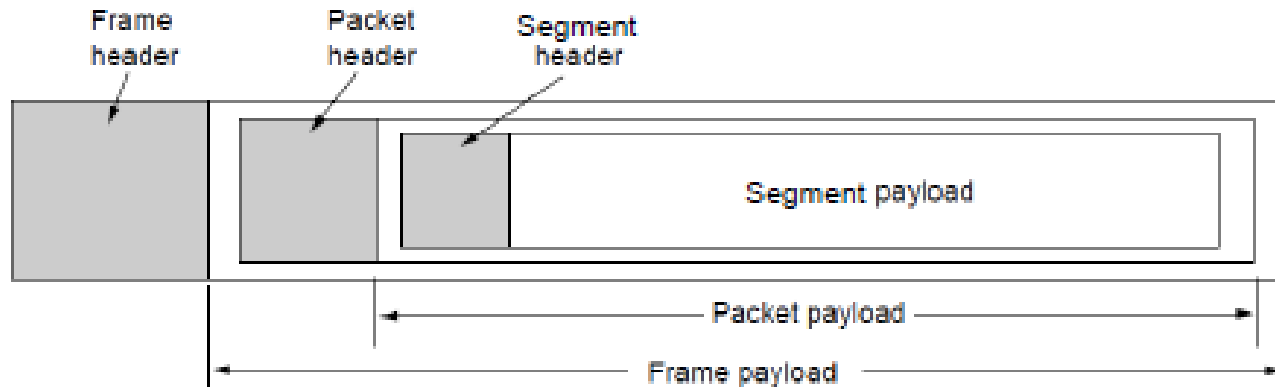
Capa de transporte

- **¿Por qué conviene estudiar la capa de transporte?**
 - Al desarrollar una aplicación de red, hay que pensar en qué requisitos ella tiene referentes a la capa de transporte.
 - Ayuda a hacer aplicaciones más eficientes y de mejor calidad el conocer cómo funciona la capa de transporte.
 - Para usar la API de los sockets hace falta entender cómo funcionan algunos protocolos de capa de transporte.

Capa de transporte

- Problemas que soluciona la capa de transporte
 - Uso de temporizadores y las retransmisiones de paquetes.
 - El direccionamiento explícito de los destinos.
 - ¿Cómo hacer para que un proceso adecuado atienda a las necesidades de una máquina cliente?
 - El proceso podría no estar activo, el cliente podría no saber cuál proceso usar, etc.
 - Uso de búferes y control de flujo.
 - Evitar congestionar la red poniendo demasiados paquetes en ella.
 - Cuando la CR pierde paquetes, la CT puede solucionarlo.

Capa de transporte



- **Segmento** = unidad de datos del protocolo de transporte
- **Confirmaciones de recepción** de paquetes enviados.
- **Tipos de paquetes que deben ser confirmados.**
 - paquete de datos
 - paquetes con información de control.
 - La misma es transparente para los usuarios de CT.

Capa de transporte

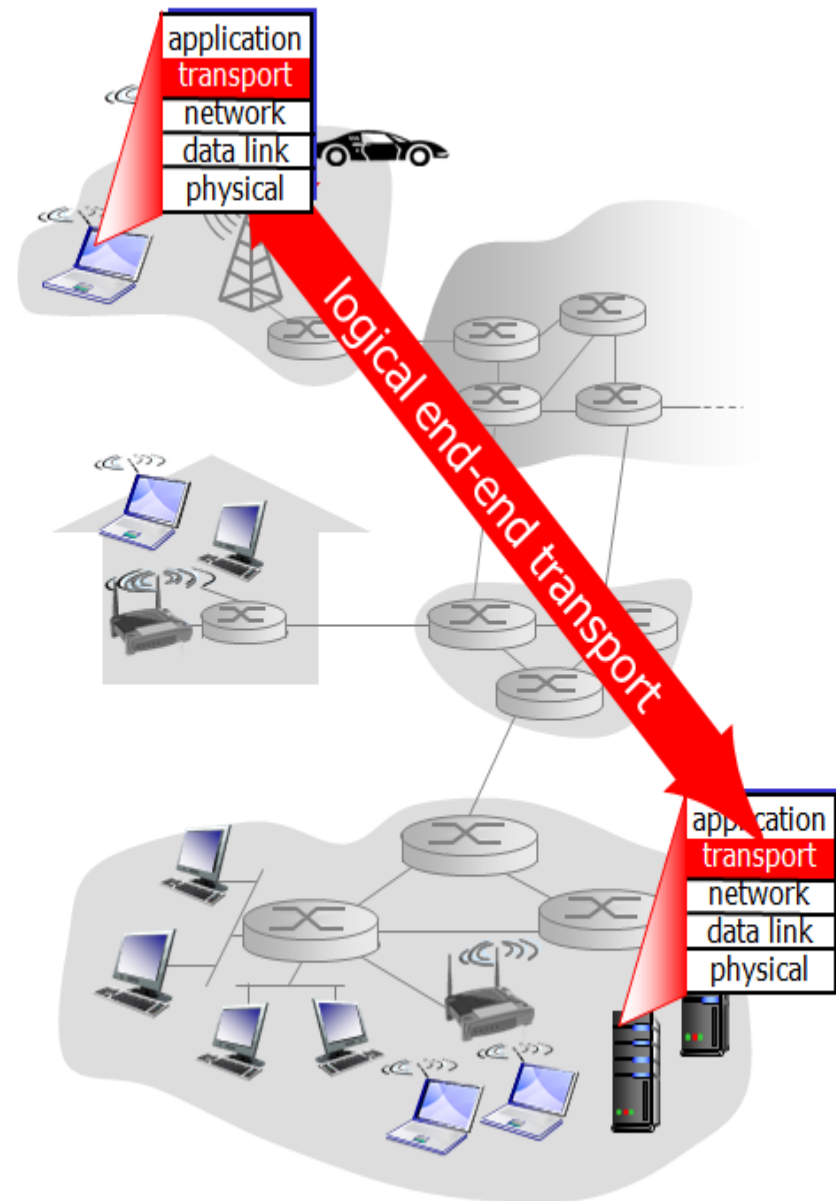
- **Problema:** La capa de transporte debería permitir:
 - la entrega de segmentos al host de destino;
 - que la entrega de segmentos sea ordenada (respetando el orden del flujo de datos a enviar recibido de la capa de aplicación).

Capa de transporte

- **Solución 1:** Para la entrega ordenada de segmentos al host de destino se puede:
 - Numerar los segmentos a enviar (usando **números de secuencia**) – respetando el orden del flujo de datos recibido de la capa de aplicación.
 - Usar para cada número de segmento enviado un **temporizador de retransmisiones**.
 - Mandar **confirmaciones de recepción (ACK)** para segmentos recibidos correctamente.
 - Si expira el temporizador de un segmento sin recibir el ACK, retransmitir el segmento correspondiente.
 - Los segmentos recibidos son **re-ensamblados en orden** y entregados a la capa de aplicación del receptor.

TCP

- **TCP (protocolo de control de transmisión)**
 - **Meta:** proporcionar un flujo de bytes confiable de extremo a extremo a través de una interred no confiable.
- TCP se adapta dinámicamente a las propiedades de la interred y se superpone a muchos tipos de fallas.
- **ET TCP (ETCP).**
- Usaremos la palabra TCP para referirnos: a veces a la ETCP y a veces al protocolo TCP.



TCP

- **Problemas que resuelve TCP:**
 - Retransmisión de paquetes:
 - uso de números de secuencia, confirmaciones de recepción y temporizadores.
 - Fijar la duración de temporizadores de retransmisiones (algoritmo complejo)
 - Manejo de conexiones entre pares de procesos
 - Direccionamiento
 - Control de congestión
 - Control de flujo

TCP

- Una ETCP acepta **flujos de datos** a **transmitir** de procesos locales,
 - Cada flujo de datos se **divide en fragmentos** que no excedan los 64 KB llamados segmentos,
 - y se envía cada segmento dentro de un datagrama IP.
- **¿Qué pasa cuando los datagramas con datos TCP llegan a una máquina?**

TCP

- El servicio TCP se obtiene al hacer que tanto el servidor como el cliente creen **sockets**.
 - **Dirección de un socket** = IP + Puerto
 - Para obtener el servicio TCP se debe **establecer una conexión** explícitamente entre el socket en la máquina emisora y uno en la máquina receptora.
- Un socket puede usarse para **múltiples conexiones** al mismo tiempo:
 - dos o más conexiones pueden terminar en el mismo socket.
 - Las **conexiones se identifican** mediante los identificadores de sockets de los dos extremos: (socket1, socket2).

TCP

- **Importante:** Cada byte de un flujo de datos a enviar en una conexión TCP tiene su propio **número de secuencia** de 32 bits.
 - Esto impone un límite en el tamaño de un flujo de datos.
- **¿Por qué se necesitan los números de secuencia?**
 - para confirmaciones de recepción y para otros asuntos según veremos.
- La ETCP emisora y la receptora intercambian datos en forma de **segmentos**.
 - Segmento = **encabezado TCP** ++ (0 o más bytes) de datos.

TCP

- **Límites que restringen el tamaño de un segmento**
 - Cada segmento, debe caber en la carga útil de 65.515 bytes del IP.
 - Cada red tiene una **unidad máxima de transferencia (MTU)** y cada segmento debe caber en la MTU.
 - En la práctica la MTU es usualmente de 1500 bytes (el tamaño de la carga útil de Ethernet).

TCP

- **Problema:** La capa de red (que incluye IP)
 - no proporciona ninguna garantía de que los datagramas se entregarán de manera apropiada,
 - tampoco garantiza que se entregarán.
- **Solución de TCP:**
 - Si un datagrama se recibe correctamente se confirma su recepción.
 - Si no se confirma la recepción de un datagrama luego de un intervalo de tiempo entonces se debe retransmitir.
 - Corresponde a TCP terminar los temporizadores y retransmitir los datagramas conforme sea necesario.

TCP

- **Problema:** Los datagramas que llegan podrán hacerlo en el orden incorrecto.
 - Esto sucede cuando se trabaja con redes de datagramas.
- **Esto es un problema porque:**
 - Usualmente la capa de aplicación del receptor necesita procesar los mensajes en el orden en que fueron enviados.
- **Solución:** Corresponde a TCP **reensamblar** los mensajes en la *secuencia apropiada*.

TCP

- Cuando un transmisor envía un segmento, también inicia un temporizador.
 - Cuando llega el segmento a destino, la ETCP receptora devuelve un segmento (con datos si existen, sino sin ellos) que contiene un **número de confirmación de recepción** igual al *siguiente número de secuencia que espera recibir*.
 - Si el temporizador expira antes de llegar el ack, el emisor envía de nuevo el segmento.

TCP

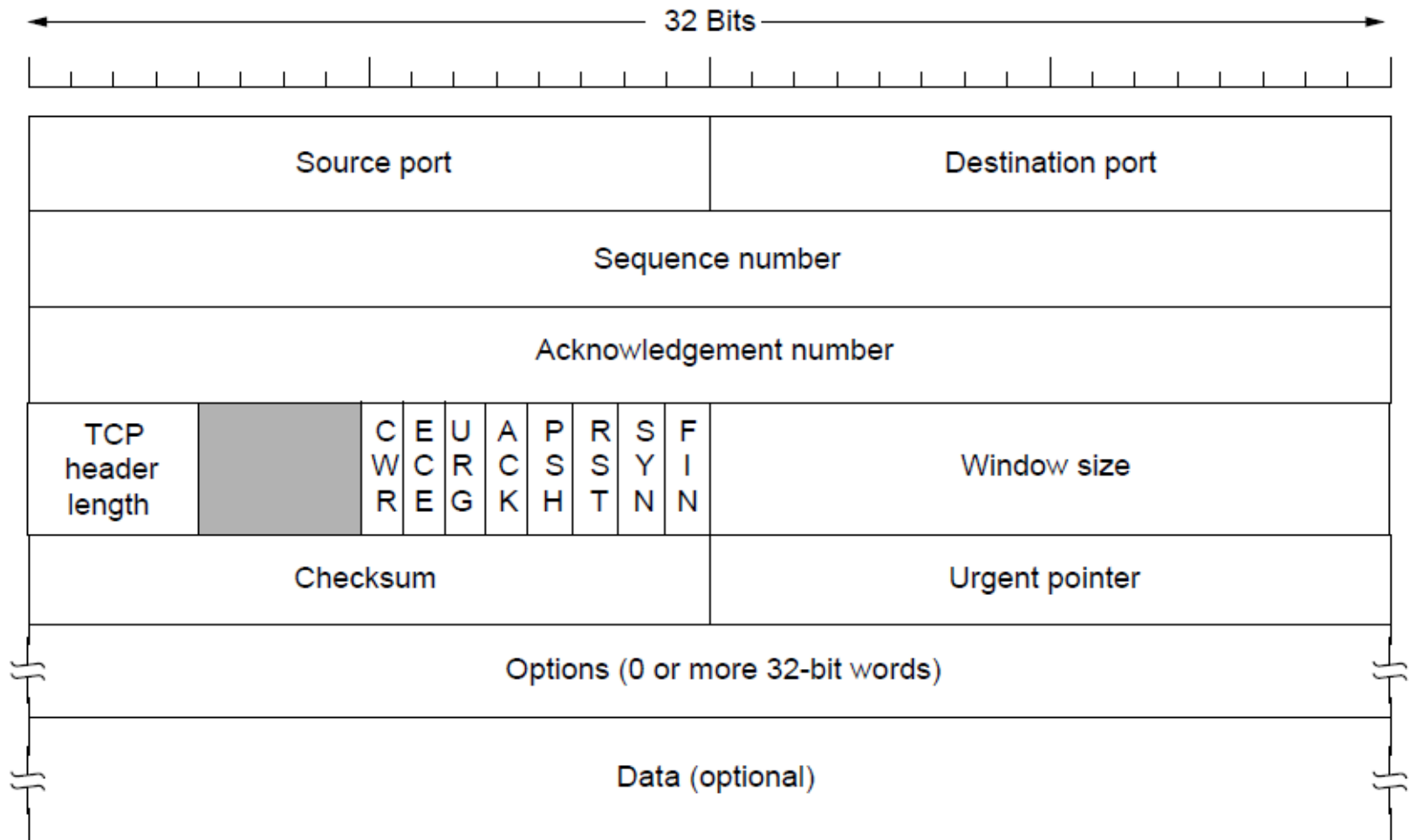
- **Problemas a manejar/resolver por TCP eficientemente:**
 - Pueden llegar segmentos fuera de orden,
 - los bytes 3072-4095 podrían llegar pero no enviarse el ack, porque los bytes 2048-3071 no han aparecido aun.
 - **Consecuencias:**
 - Habrá que esperar a veces antes de entregar segmentos a la capa de aplicación.
 - Habrá que esperar a veces antes de enviar confirmaciones de recepción.
 - También pueden **retardarse segmentos** en tránsito durante tanto tiempo que el temporizador del emisor expira y los segmentos se retransmiten.

TCP

- **Situación:** Las retransmisiones podrían incluir *rangos de bytes diferentes a los de la transmisión original*.
 - **Esto puede suceder porque:**
 - Hay nuevos datos para enviar y se los puede mandar.
- Se requiere una **administración cuidadosa** para llevar el control de los bytes que se han recibido correctamente en un momento determinado

Segmentos TCP

1. **Encabezado fijo** de 20 bytes
2. **Opciones de encabezado** en palabras de 32 bits
3. **Datos** opcionales



Segmentos TCP

- Los **segmentos sin datos** se usan para acks y **mensajes de control**.
- **Puerto de origen y puerto de destino:**
 - Son de 16 b cada uno.
 - La dirección de un puerto más la dirección IP del host forman un **punto terminal único** de 48 b.
 - Los puntos terminales de origen y de destino en conjunto **identifican** la conexión.

El encabezado del segmento TCP

- El campo **número de secuencia** de un segmento
 - es *un número de byte en el flujo de bytes transmitido* y
 - corresponde al **primer byte** en el segmento.
 - Tiene 32 b de longitud.
- El campo **número de confirmación de recepción**
 - indica el **siguiente byte esperado** del flujo de bytes a transmitir.
 - Tiene 32 b de longitud.

El encabezado del segmento TCP

- **ACK** se establece en 1 para indicar que el n° de confirmación de recepción es válido.
 - Si el ACK = 0 entonces el segmento no contiene una confirmación de recepción.

El encabezado del segmento TCP

- La **longitud del encabezado TCP**: N° de palabras de 32 bits en el encabezado TCP.
- El **campo de opciones** es de longitud variable.

El encabezado del segmento TCP

- **Ejercicio:** Responder:
 - ¿Hasta cuántas palabras de 32 b se pueden tener en un encabezado TCP?
 - ¿Hasta cuántas palabras de 32 b puede ocupar el campo de opciones?

TCP

- **Ejercicio:** Se pide:
 1. Hacer una tabla que compare la solución 1 de comunicación confiable (filmina 12) con la comunicación confiable de TCP,
 - contrastando asuntos como números de secuencia, ACKs de segmentos, retransmisiones, y cualquier otro asunto que consideren conveniente.
 2. Hacer tabla de ventajas y desventajas de la solución de TCP en relación a la solución 1.