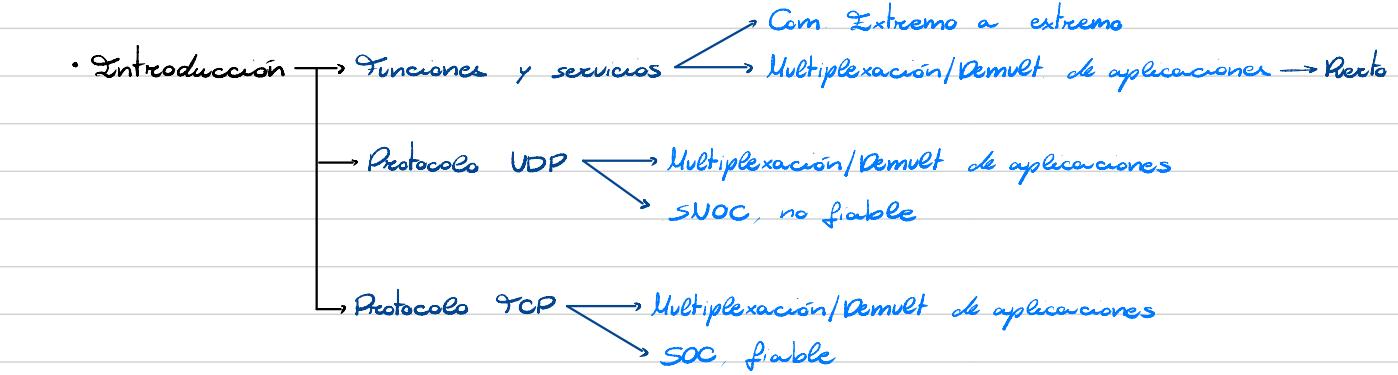


ASRRR TR

Tema 3. Capa de transporte



• User Datagram Protocol (UDP)

Funcionalidad: best-effort

- SNOC no fiable
- No entrega ordenada
- No control congestión

Mux / Demux. transportar TDDU al proceso correcto ⇒ Puerto → Libre disposición

Frecuentemente en aplicaciones multimedia

Cada segmento UDP ~~se encapsula~~ → Datoagrama TCP

Presignados

• Transmission Control Protocol (TCP)

- Servicio punto a punto
- SOC → Hand-shaking y fiable
- Entrega ordenada
- Transmisión full-duplex
- Mecanismos de detección y recuperación de errores, y ACKs
- Funcionalidades de TCP:

- Mux / Demux Aplicaciones ⇒ Puerto → 3 fases → Establecer → Trans. de datos → Cierre
- Control conexión (start/stop) Se ident por puerto e IP origen / puerto e IP destino
- Control errores flaco y congestión → Se manif en pérdidas o retrasos ACKs ⇒ limitar tráfico generado

- Cada seg. TCP ~~se encapsula~~ → Datoagrama TCP

→ esquema ARQ ⇒ Timeouts cálculo

Control de errores: cómo estimar los timeouts?

- Mayor que el tiempo de ida y vuelta (RTT)
- Si es demasiado pequeño se pierden paquetes
- Si es demasiado grande: reacción lenta o pérdida de segmentos.
- Para situaciones cambiantes la mejor solución es la adaptable.

RTTmedido: tiempo desde la emisión de un segmento hasta la recepción del ACK.

$$RTT_{nuevo} = \alpha RTT_{viejo} + (1-\alpha) RTT_{medido}, \alpha \in [0,1]$$

$$\text{Desviación}_{nueva} = (1-x) * \text{Desviación}_{vieja} + x * | RTT_{medido} - RTT_{nuevo} |$$

$$\text{Timeout} = RTT_{nuevo} + 4 * \text{Desviación}$$

Generación de ACKs:

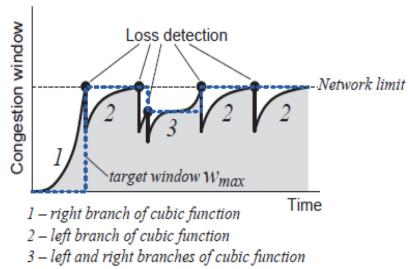
Evento	Acción del TCP receptor
Llegada ordenada de segmento, sin discontinuidad, todo lo anterior ya confirmado.	Retrasar ACK. Esperar recibir al siguiente segmento hasta 500 mseg. Si no llega, enviar ACK.
Llegada ordenada de segmento, sin discontinuidad, hay pendiente un ACK retrasado.	Inmediatamente enviar un único ACK acumulativo.
Llegada desordenada de segmento con # de sec. mayor que el esperado, discontinuidad detectada.	Enviar un ACK duplicado, indicando el # de sec. del siguiente byte esperado.
Llegada de un segmento que completa una discontinuidad parcial o totalmente.	Confirmar ACK inmediatamente si el segmento comienza en el extremo inferior de la discontinuidad.

- Problema con ACKs repetidos: ambigüedad en la interpretación.
- Solución: Algoritmo de Korn, actualizar el RTT sólo para los no ambiguos, pero si hay que repetir un segmento incrementar el timeout:

$$tout_{nuevo} = \gamma * tout_{viejo}, \gamma = 2.$$

• Extensiones TCP

TCP se define con múltiples sabores? no afectan a la interoperabilidad
 Linux reciente \Rightarrow TCP Cubic



Adaptación de TCP a redes actuales \rightarrow Ventana escalada opción en segmentos SYN hasta 1GB autorizado

\rightarrow Estimación Rtt: sello de tiempo en todos los seq.

\rightarrow PAWS ("Protect Against Wrapped Sequence numbers")
 sello de tiempo + rechazo seq duplicados