# **Tema 3: Parte A (UDP) Capa de transporte**

**Conceptos:**

**UDP**: protocolo de transporte sin conexión. Puertos 0 al 1023 están reservados para protocolos estándares.

Trocea los mensajes de Aplicación en segmentos que pasa a la capa de red y el receptor reensambla los segmentos en mensajes que pasa a la capa de aplicación.

Imagen de la pantalla de un celular de un mensaje en letras blancas

Descripción generada automáticamente con confianza media

**Multiplexación:** Varios procesos de capa “N+1” pasan datos a un proceso de capa “N”.Varias entradas (n) pasen por una determinada capa.

**Demultiplexación**: Un proceso de capa “N” pasa datos a varios procesos de capa “N+1”. Un proceso tiene que dar lugar a varias salidas.

**Demultiplexación sin conexión (UDP):** Se requiere puerto origen y puerto destino. 1 sockets.

**Demultiplexación orientada a la conexión (TCP):** Se requiere puerto e IP origen y puerto e IP destino. Cada socket es identificado por su 4-tupla propia. Un servidor Web tiene un socket diferente por cada cliente conectado. HTTP no-persistente tendrá diferentes sockets por cada petición de datos, todos ellos con puerto 80 en servidor.

Diagrama

Descripción generada automáticamente

**Checksum UDP**: Forma de controlar los errores en UDP. Con esto se descartan los los errores pero no se avisan.

Imagen que contiene Texto

Descripción generada automáticamente

# **Tema 3: Parte B (TCP) Estructura de segmento**

**Conceptos:**

**TCP**:

* Estructura de un segmento
* Transferencia confiable de datos
* Control de flujo
* Gestión de la conexión
* Control de congestión

**Número de secuencia:** flujo de ida. Para saber el primer bit de datos que estoy mandando, dentro del flujo de datos que estoy mandando.

**Número de Acknowledgement**: número de bits que estoy esperando de ti. Flujo de vuelta.

Estos dos números sirven para fijarnos si se ha perdido un paquete o no. Permiten controlar el flujo.

Escala de tiempo

Descripción generada automáticamente

# **Tema 3: Parte C (TCP) Transferencia confiable de datos**

**Conceptos:**

* Usa envío de segmentos en cadena “pipeline”.
* ACKs acumulativos.
* Usa un timer único de retransmisión.

Texto

Descripción generada automáticamenteDiagrama, Esquemático

Descripción generada automáticamente

Cuando llega el tercer Ack duplicado es cuando se renvía el paquete.

**Resumen**:

Tabla

Descripción generada automáticamente

**ACK retardado**: sirve para no sobre cargar el canal.

**Time-out**: Retardo largo antes de reenvío de paquetes perdidos. Suelen ser largos.

# **Tema 3: Parte D (TCP) Control de flujo**

**Conceptos:**

**Diagrama

Descripción generada automáticamente**

**Pregunta examen**

**Cuando el transmisor de una conexión TCP está a punto de enviar un segmento con número de secuencia 773, recibe un acuse de recibo con numeración 123 y ventana de recepción 1300. ¿Cuántos bytes como máximo puede transportar el segmento que está a punto de enviar?**

Los bytes 123 hasta 772 inclusive (650 bytes) están en tránsito para el valor de ventana de recepción 1300. Es así como podemos asegurar que el receptor podrá almacenar 1300-650 = 650 bytes. Éste es el número máximos de bytes a transportar en el próximo segmento.

# **Tema 3: Parte E (TCP) Gestión de la conexión**

**Conceptos:**

**SYNbit**: cuando se pasa a 1, significa que quiere establecer conexión contigo.

**Seq**: x, Es un número aleatorio elegido por el cliente y el servidor. Se utiliza para que no se pueda inferir en el flujo de una conexión.

**ACKnum**: x+1

**Cierre de conexión**: Cliente y servidor cada uno cierra su lado de la conexión. Envía segmento TCP con bit FIN=1. Responde a FIN recibido con ACK.

El cliente envia el FINbit a 1 y el número de secuencia (seq = x) que corresponda. El servidor responde con ACKbit = 1 y ACKbit = x+1. El servidor aún puede enviar datos, una vez que cierra el socket envía el FINbit=1 y el seq = y, en este momento ya no pude enviar más datos. Por último, el cliente responde con un ACKbit =1 y el ACKmun y +1.

Diagrama

Descripción generada automáticamente

**Pregunta examen**

**¿Cuántos byte se han intercambiado en una conexión?**

Se tiene que mirar el número de seq final y el inicial y se restan. Algo así, no exactamente.

# **Tema 3: Parte F (TCP) Control de congestión**

**Conceptos:**

**Enfoque**: el emisor aumenta la velocidad de transmisión (tamaño de la ventana), buscando ancho de banda utilizable, hasta que se produce una pérdida

**Crecimiento aditivo**: aumentar cwnd en 1 MSS cada RTT hasta que se detecte una pérdida.

**Decrecimiento multiplicativo**: baja cwnd a la mitad después de la pérdida.

Imagen que contiene Flecha

Descripción generada automáticamente

Cuando empieza la conexión, aumentamos la tasa de envío exponencialmente hasta el primer evento de pérdida: inicialmente cwnd = 1 MSS, se dobla el cwnd cada RTT e incrementamos en 1 cwnd por cada ACK recibido. Resumiendo: La tasa inicial es lenta, pero aumenta exponencialmente rápido.

**¿Cuándo se debe cambiar de incremento exponencial a incremento lineal?** Cuando cwnd alcanza 1/2 de su valor antes del TimeOut.

Gráfico, Gráfico de líneas

Descripción generada automáticamente

**La pérdida**: se puede dar por recibir 3 ACKs duplicados o un timeout (este último es peor).