Bloque III: El nivel de transporte

Tema 5: UDP y TCP

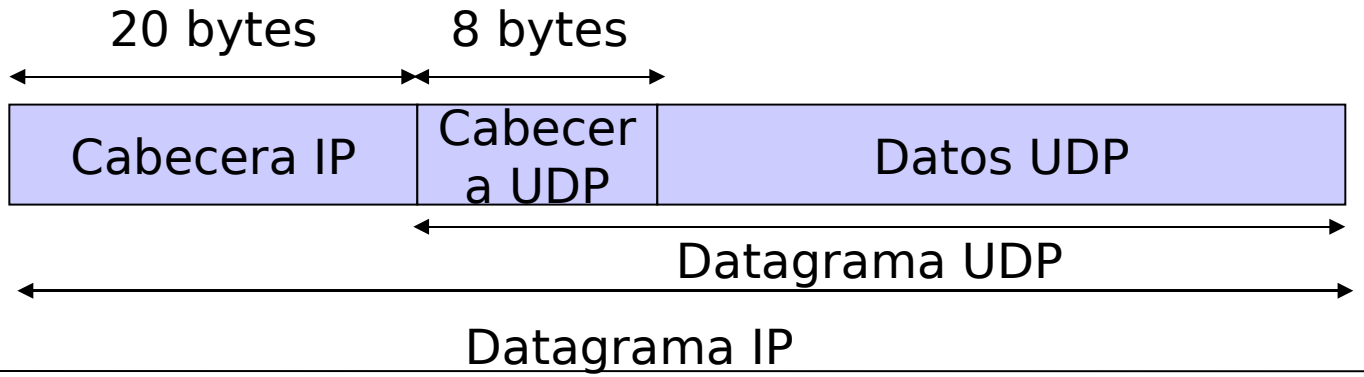
Índice

- Bloque III: El nivel de transporte
  - Tema 5: UDP y TCP
    - UDP
      - Cabecera UDP
    - TCP
      - Cabecera TCP

- **Referencias**
  - Capítulo 3 de “Redes de Computadores: Un enfoque descendente basdado en Internet”. James F. Kurose, Keith W. Ross. Addison Wesley, 2ª edición. 2003.
  - Capítulos 11 y 17 de “TCP/IP Illustrated, Volume 1: The Protocols”, W. Richard Stevens, Addison Wesley, 1994.

UDP

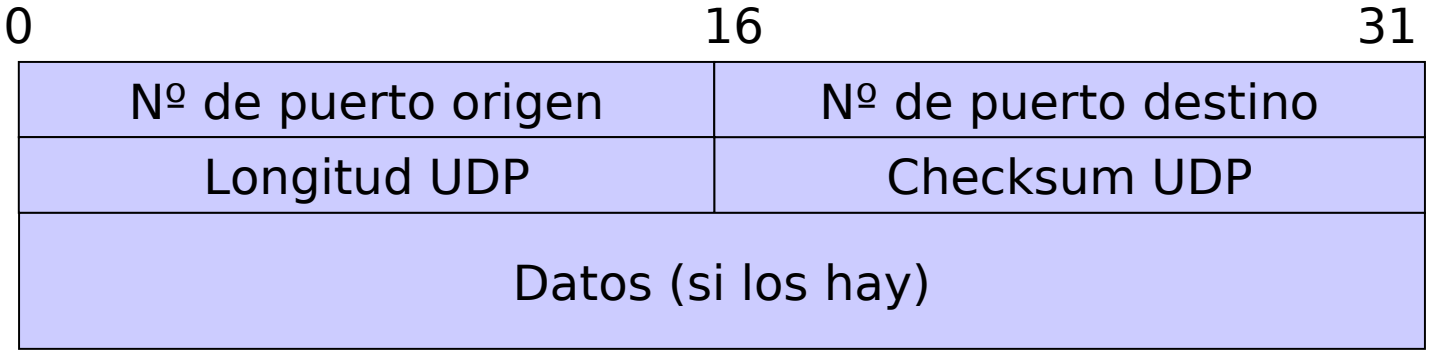
- User Datagram Protocol – Especificado en el RFC 768.
- UDP es un protocolo de nivel de transporte, orientado a datagramas, y simple ➔ “cada operación de salida generada por un proceso produce exactamente un único datagrama UDP”.
- UDP no garantiza que el datagrama alcance su destino.
- UDP multiplexa los datos de las aplicaciones y efectúa opcionalmente una comprobación de errores, pero no realiza:
  - Control de flujo
  - Control de congestión
  - Retransmisión de datos perdidos
  - Conexión/desconexión



UDP

- Se utiliza principalmente en los siguientes casos:
  - Cuando el intercambio de mensajes es muy escaso, por ejemplo: consultas al DNS (servidor de nombres).
  - Cuando la aplicación es en tiempo real y no se pueden esperar los ACKs. Por ejemplo, videoconferencia, voz sobre IP.
  - Cuando los mensajes se producen regularmente y no importa si se pierde alguno. Por ejemplo: NTP, SNMP.
  - Cuando el medio de transmisión es altamente fiable y sin congestion (LANs). Por ejemplo: NFS.
  - Si se envía tráfico broadcast o multicast.

Cabecera UDP



- Los números de puerto identifican los procesos emisor y receptor.
  - Los números de puerto UDP son independientes de los de TCP.
- Longitud UDP = longitud de la cabecera UDP + longitud de datos.
  - El valor mínimo es de 8 bytes.
  - Es redundante con la información de la cabecera IP.
- Checksum: se calcula sobre la cabecera UDP y los datos UDP.
  - Se calcula de manera similar al checksum en IP (complemento a 1 de la suma de las palabras de 16 bits).
  - Aunque hay algunas diferencias:
    - El datagrama UDP puede contener un número impar de bytes ➔ Se añade un byte de relleno (todo ceros).
    - Se considera una pseudo-cabecera de 12 bytes para el cálculo del checksum, que contiene algunos campos de la cabecera IP ➔ Doble comprobación de estos campos (igual en TCP).

UDP: Modelo cliente-servidor

- Servidor:
  - Más complicado que los clientes.
  - Se comunica con múltiples clientes simultáneamente.
  - Arranca, pasa a estado “espera”, y espera a que llegue alguna demanda de cliente. Despierta cuando llega algún datagrama de un cliente (una solicitud, normalmente).
- ¿Cómo se conoce la dirección IP y número de puerto del cliente?
  - Cliente envía al servidor un datagrama UDP:
    - Datagrama IP: contiene dirección IP de origen y destino.
    - Datagrama UDP: contiene número de puerto de origen y destino.
- Cola de entrada UDP
  - Normalmente, un servidor UDP gestiona múltiples peticiones de cliente en un único puerto (nº de puerto identifica la aplicación servidora).
  - Hay una cola de entrada de datagramas asociada a cada puerto UDP que utiliza una aplicación:
    - Capacidad limitada.
    - Cola FIFO: los datagramas se pasan a las aplicaciones en el orden de llegada.
    - Si se produce desbordamiento ➔ se descartan los datagramas.
- Varios recipientes por puerto:
  - En sistemas que soportan multicast son posibles varios recipientes por puerto.

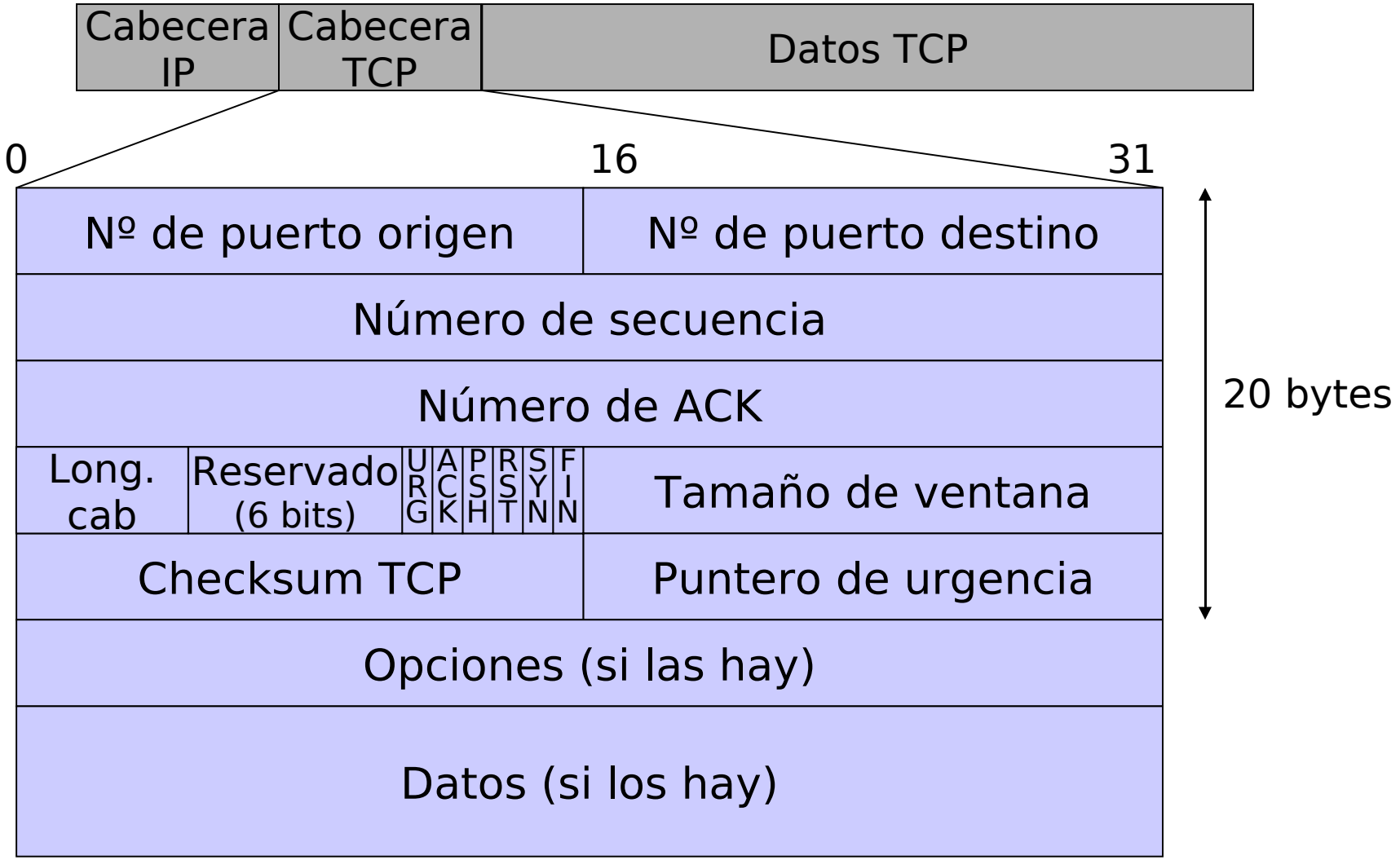
TCP

- Transmission Control Protocol – Especificado en RFC 793.
- TCP proporciona un servicio de datagramas fiable, orientado a conexión y de flujo de bytes ("byte stream").
  - Orientado a conexión: dos aplicaciones (cliente-servidor) deben establecer una conexión TCP entre ellos antes de comenzar el intercambio de datos.
  - Broadcasting y multicasting no son aplicables.
  - De flujo de bytes: los dos extremos de una conexión TCP intercambian secuencias de bytes. Los bytes los interpreta el nivel de aplicación.
- Para implementar la fiabilidad TCP implementa lo siguiente:
  - Divide datos de la aplicación en segmentos con la longitud más adecuada para la aplicación.
  - Asocia un temporizador con cada segmento que envía. Si no recibe el ACK del destino a tiempo ➔ retransmite el segmento.
  - Mantiene un checksum en la cabecera TCP para comprobar el segmento recibido. No se envía ACK si el segmento es incorrecto.
  - El receptor TCP reordena los segmentos, si es necesario, para pasarlos ordenados a la aplicación (los segmentos se pueden desordenar en la transmisión).
  - Descarta segmentos que se hayan podido duplicar.
  - Proporciona control de flujo: un receptor TCP sólo deja transmitir al otro extremo segmentos que pueden almacenarse en su buffer de entrada sin producirse desbordamientos.

TCP

- Funciones de TCP:
  - Establecer y terminar conexiones
  - Gestionar los buffers y ejercer control de flujo de forma eficiente
  - Multiplexar el nivel de aplicación (puertos) e intercambiar datos con las aplicaciones
  - Controlar errores, retransmitir segmentos perdidos o erróneos y eliminar duplicados
  - Efectuar control de congestión

Cabecera TCP



Cabecera TCP

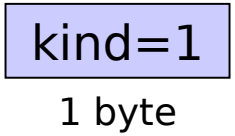
- **Nº de puerto origen y destino** + dir. IP origen y destino de cabecera IP identifican unívocamente la conexión TCP.
  - nº puerto + dir. IP = socket
  - Conexión TCP = par de sockets
- **Número de secuencia**: identifica el nº de byte en el flujo de bytes TCP entre el emisor y el receptor que supone el primer byte de la sección de datos:
  - Cuando se llega a 2^{32} -1 se comienza de nuevo por 0.
  - Cuando se establece una conexión, se pone a 1 el flag SYN, y la máquina selecciona un ISN (Initial Sequence Number) para esa conexión.
- **Número de ACK** (acknowledgment) indica el siguiente número de secuencia que el emisor del ACK espera recibir.
  - Es el nº de secuencia + 1 del último byte recibido satisfactoriamente.
  - TCP proporciona una comunicación "full-duplex" al nivel de aplicación ➔ Cada extremo mantiene su nº de secuencia.
  - No existen ACK's selectivos o negativos.
- **Longitud de cabecera** (4 bits): tamaño de la cabecera incluyendo opciones.
  - Especifica el número de palabras de 32 bits
  - Valor máximo 60 bytes
- **Flags**:
  - URG: puntero de urgencia válido.
  - ACK: número de ACK válido.
  - PSH: el receptor debe pasar estos datos a la aplicación lo antes posible.

Cabecera TCP

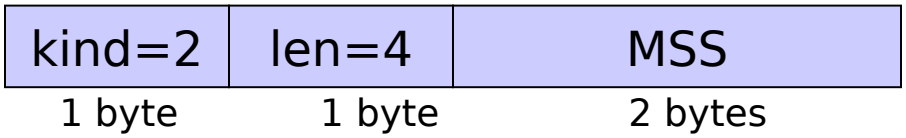
- **Flags**:
  - RST: reinicializar la conexión (reset).
  - SYN: sincronizar números de secuencia para iniciar una conexión.
  - FIN: el emisor finaliza el envío de datos.
- **Tamaño de ventana**: indica el nº de bytes, comenzando por el valor del campo de nº de acknowledge, que el receptor puede aceptar.
  - Utilizado para establecer control de flujo.
  - Máximo 65.535, pero existe una opción de factor de escala para incrementar este valor.
- **Checksum**: sobre todo el segmento TCP (cabecera + datos).
  - Es obligatorio: debe calcularlo el emisor y comprobarlo el receptor.
  - El cálculo es similar al checksum de UDP.
- **Puntero de urgencia**: Válido si el flag URG es 1.
  - Indica un offset a añadir al nº de secuencia.
  - Se utiliza para transmitir datos urgentes.
- **Opciones**: La más común es la opción de máximo tamaño de segmento (Maximum Segment Size).
  - MSS: Indica el máximo tamaño que quiere recibir el emisor. Se anuncia en el primer segmento intercambiado.
- **Datos**: Incluye la información a intercambiar (opcional)

Cabecera TCP: Opciones

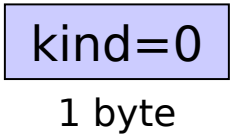
- Las opciones contempladas en el RFC 793 son:
  - No operation: para rellenar a múltiplos de 4 bytes el campo opciones.



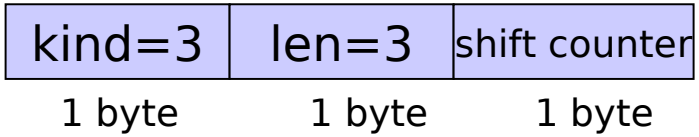
  - Maximum Segment Size



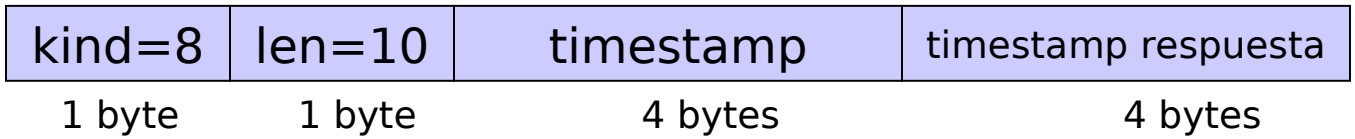
- En el RFC 1323 se definen otras opciones (no contempladas en todas las implementaciones):
  - End of Option List



  - Window Scale Factor



  - Timestamp



Multiplexación

- La multiplexación se realiza mediante el puerto (origen o destino) que está entre 0 y 65535.
- Los puertos 0 a 1023 están reservados para servicios “bien conocidos” (“well known ports”)
- La combinación de dirección IP y puerto identifica el “socket”
- Una conexión TCP queda especificada por los dos sockets que se comunican:
  - Dos conexiones TCP deben tener al menos uno de los cuatro componentes distinto (dirección IP origen y destino + puerto origen y destino)