## 2.6. User Datagram Protocol (UDP)

El *User Datagram Protocol* (UDP) [RFC768] es probablemente el protocolo de transporte más sencillo posible. Tal como muestra la **Figura 2.11**, su cabecera solo tiene 4 campos de 16 bits: puerto origen, puerto destino, longitud del datagrama UDP (incluyendo cabecera y los datos que transporta), y un *checksum* para comprobar que no se ha corrompido durante su transmisión.

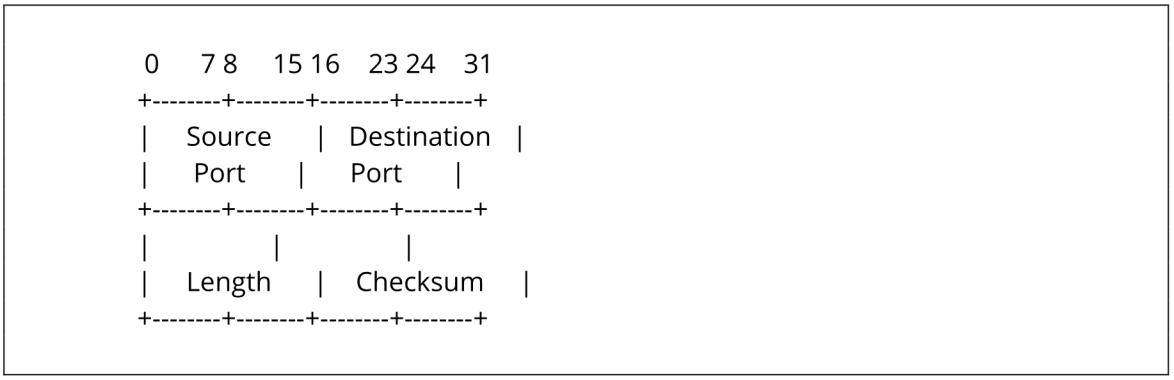


**Figura 2.11** – Formato de la cabecera del protocolo UDP [RFC768]

Los campos más importantes de la cabecera UDP son los puertos origen y destino que, junto a las direcciones IP origen y destino, definen las **direcciones de nivel de transporte** del origen (<dirección IP origen, protocolo UDP, puerto UDP origen>) y del destino (<dirección IP destino, protocolo UDP, puerto UDP destino>), lo que permite identificar "unívocamente" (obviando la presencia de NATs) los procesos extremos de la comunicación. La dirección IP identifica el ordenador, y el puerto un proceso que corre en el mismo. La combinación de las direcciones de transporte origen y destino da lugar a la quíntupla: <dirección IP origen, puerto origen, protocolo de transporte, IP destino, puerto destino>, que permite identificar un **flujo** de paquetes en Internet (tanto en UDP como en TCP, o cualquier otro

protocolo de transporte). Al ser campos de 16 bits, cada equipo solo puede tener $2^{16}$ = 65.536 puertos diferentes para cada protocolo de transporte, aunque el puerto 0 está reservado, por lo que solo se pueden utilizar los puertos entre el 1 y el 65.535. Los puertos UDP y TCP están organizados en diferentes grupos:

- **Puertos bien conocidos (*well-known*, en inglés): 1-1023**. Estos puertos solo los deberían usar procesos del sistema, por lo que, en muchos sistemas operativos como los UNIX, es necesario tener permisos de superusuario para utilizar uno de estos puertos. La mayoría de los primeros protocolos que se definieron en Internet usan puertos en este rango, como por ejemplo: SSH (TCP/22), SMTP (TCP/25), DNS (UDP/53, TCP/53), HTTP (TCP/80), HTTPS (TCP/443). Normalmente estos puertos siempre se usan en el lado servidor (que debe tener una dirección de transporte conocida para el cliente), aunque hay excepciones como BOOTP/DHCP, donde el servidor utiliza el puerto UDP/67, y el cliente el puerto UDP/68.
- **Puertos registrados: 1024-49151**. La popularidad de Internet y la explosión de protocolos hizo que el rango de los puertos bien conocidos se agotase rápidamente, por lo que el rango de puertos reservados se amplió hasta este grupo. Estos puertos también suelen estar reservados para la parte servidora de los protocolos, aunque normalmente no es necesario tener permisos de superusuario para utilizar uno de estos puertos.
- **Puertos dinámicos o efímeros: 49152–65535**. Estos puertos no pueden reservarse para ningún protocolo, porque los usan los clientes de los diferentes protocolos para comunicarse con los servidores. Normalmente es el sistema operativo el que selecciona un puerto al azar o el primero que se encuentre disponible, por lo que cada vez que se ejecuta un cliente puede utilizar un puerto origen diferente. Esto no es mayor problema en las aplicaciones cliente-servidor porque, como las comunicaciones las inicia el cliente y la dirección de transporte del servidor es bien conocida (i.e. utiliza un puerto fijo en el rango 1-1023 o 1024-49151), el servidor únicamente tiene que responder a la dirección de transporte origen del cliente. Así que, en general, es posible identificar si un datagrama UDP lo ha enviado un cliente o un servidor simplemente comprobando si el puerto origen o el destino pertenecen a este rango.

La lista oficial de puertos asignados [PortList] la mantiene el IANA (*Internet Assigned Numbers Authority*), aunque hay muchos protocolos que no registran su uso (especialmente en el caso de *malware* o protocolos propietarios), por lo que también es recomendable consultar páginas como [SpeedGuide], que van registrando los usos no oficiales de cada puerto.

El campo longitud de UDP indica el número de octetos que ocupa tanto la cabecera UDP como los datos que transporta. Por lo tanto, el mensaje UDP más largo solo

puede tener 65.536 – 8 = 65.628 octetos, aunque depende de IP para fragmentar mensajes que no quepan en un datagrama IP. Los mensajes UDP fragmentados suelen dar muchos problemas, ya que la pérdida de un único fragmento impide la recepción del mensaje completo. Además, dado que la cabecera UDP solo aparece en el primer fragmento, hay *firewalls* que filtran el resto de los fragmentos. Por esta razón hay muchas aplicaciones UDP que limitan el tamaño de sus mensajes. Por ejemplo, los mensajes DNS sobre UDP por defecto solo pueden tener una longitud de 512 octetos. En general UDP se utiliza para el envío de mensajes pequeños (vs. TCP que se emplea para la transmisión de grandes cantidades de datos).

Por último, el campo *checksum* de UDP es una suma de comprobación de la cabecera UDP, de los datos que transporta, y de algunos campos de la cabecera IP (dado que las direcciones de transporte también están formadas por las direcciones IP y el identificador de protocolo de IP), lo que sirve para comprobar si algunos de estos datos se han corrompido durante el reenvío del datagrama UDP hasta el destino. Sin embargo, el algoritmo de *checksum* definido para UDP (que es el mismo para IPv4 y TCP), es muy débil y es posible que llegue un datagrama UDP con datos corruptos que no sean detectados por el *checksum*. De hecho, UDP incluso permite enviar datagramas con el campo *checksum* a 0, que no se comprueba en la recepción. Esto puede ser útil para aplicaciones, como las multimedia, que tienen cierta tolerancia a errores.

Como puede observarse, en la cabecera UDP no hay ningún campo de control o de número de secuencia (como sí tiene TCP), así que UDP no permite establecer una conexión con estado entre los equipos origen y destino, ni dispone de la capacidad de detectar si se pierde algún datagrama, por lo que obviamente no puede solicitar su retransmisión. Por lo tanto, UDP es un protocolo no orientado a conexión, no fiable y sin control de flujo ni de congestión (en contraposición a TCP que es orientado a conexión, fiable y con control de congestión y de flujo). Así que es responsabilidad de la aplicación por encima de UDP de detectar si se pierden datos y retransmitirlos en caso necesario. Por otro lado, UDP permite enviar datos inmediatamente al destino (incluso si este es una dirección IP *broadcast* o *multicast*), y permite a la aplicación controlar exactamente la tasa y el momento en la que se envía su información, por lo que históricamente ha sido el protocolo más utilizado por las aplicaciones multimedia, así como por los ataques de denegación de servicio (DoS).