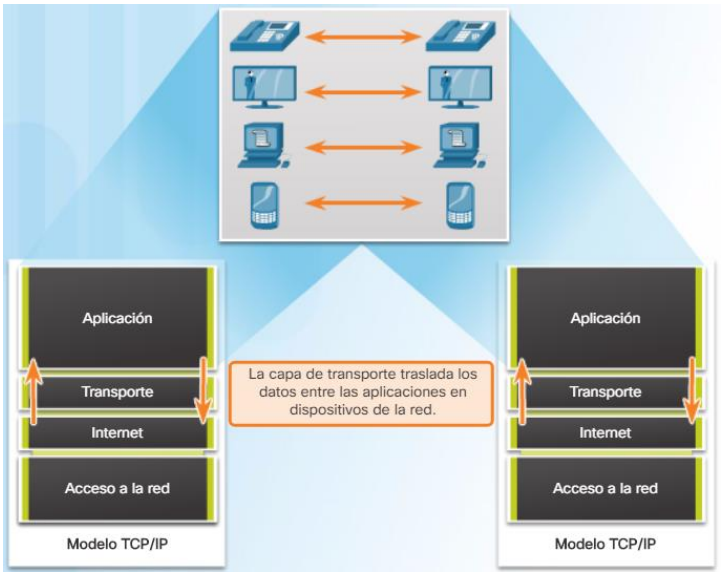
CAPÍTULO 9. CAPA DE TRANSPORTE

9.1.1.1 Función de la capa de transporte

La capa de transporte es responsable de establecer una sesión de comunicación temporal entre dos aplicaciones y de transmitir datos entre ellas. Una aplicación genera datos que se envían desde una aplicación en un host de origen a una aplicación en un host de destino. Este es, independientemente del tipo de host destino, el tipo de medios a través de los cuales deben viajar los datos, la ruta seguida por los datos, la congestión en un enlace o el tamaño de la red.



9.1.1.2 Responsabilidades de la capa de transporte

Seguimiento de conversaciones individuales

En la capa de transporte, cada conjunto de datos que fluye entre una aplicación de origen y una de destino se conoce como “conversación”. Un host puede tener varias aplicaciones que se comunican a través de la red de forma simultánea. Cada una de estas aplicaciones se comunica con una o más aplicaciones en uno o más hosts remotos. Es responsabilidad de la capa de transporte mantener y hacer un seguimiento de todas estas conversaciones.

Segmentación de datos y rearmado de segmentos

Se deben preparar los datos para el envío a través de los medios en partes manejables. La mayoría de las redes tienen un límite de la cantidad de datos que se puede incluir en un solo paquete. Los protocolos de la capa de transporte tienen servicios que segmentan los datos de aplicación en bloques de un tamaño apropiado. Estos servicios incluyen el encapsulamiento necesario en cada porción de datos. Se agrega un encabezado a cada bloque de datos para el rearmado. Este encabezado se utiliza para hacer un seguimiento del flujo de datos.

En el destino, la capa de transporte debe poder reconstruir las porciones de datos en un flujo de datos completo que sea útil para la capa de aplicación. Los protocolos en la capa de transporte describen cómo se utiliza la información del encabezado de dicha capa para rearmar las porciones de datos en flujos y transmitirlos a la capa de aplicación.

Identificación de las aplicaciones

Para pasar flujos de datos a las aplicaciones adecuadas, la capa de transporte debe identificar la aplicación objetivo. Para lograrlo, la capa de transporte asigna un identificador a cada aplicación, llamado número de puerto. A todos los procesos de software que requieran acceso a la red se les asigna un número de puerto exclusivo para ese host.

9.1.1.4 Confiabilidad de la capa de transporte

La capa de transporte también es responsable de administrar los requisitos de confiabilidad de las conversaciones.

IP se ocupa solo de la estructura, el direccionamiento y el routing de paquetes. IP no especifica la manera en que se lleva a cabo la entrega o el transporte de los paquetes. Los protocolos de transporte especifican la manera en que se transfieren los mensajes entre los hosts. TCP/IP proporciona dos protocolos de la capa de transporte: el protocolo de control de transmisión (TCP) y el protocolo de datagramas de usuario (UDP).

TCP se considera un protocolo de la capa de transporte confiable y completo, ya que garantiza que todos los datos lleguen al destino. Sin embargo, esto requiere campos adicionales en el encabezado TCP que aumentan el tamaño del paquete y también la demora. En cambio, UDP es un protocolo de capa de transporte más simple, aunque no proporciona confiabilidad. Por lo tanto, tiene menos campos y es más rápido que TCP.

9.1.1.5 TCP

La función del protocolo de transporte TCP es similar al envío de paquetes de los que se hace un seguimiento de origen a destino. Si se divide un pedido de envío en varios paquetes, el cliente puede revisar en línea el orden de la entrega. Con TCP, hay tres operaciones básicas de confiabilidad:

- Numeración y seguimiento de los segmentos de datos transmitidos a un host específico desde una aplicación específica
- Reconocimiento de los datos recibidos
- Retransmisión de los datos sin reconocimiento después de un tiempo determinado

9.1.1.6 UDP

Si bien las funciones de confiabilidad de TCP proporcionan una comunicación más sólida entre aplicaciones, también representan una sobrecarga adicional y pueden provocar demoras en la transmisión. Existe una compensación entre el valor de la confiabilidad y la carga que implica para los recursos de la red. Agregar sobrecarga para garantizar la confiabilidad para algunas aplicaciones podría reducir la utilidad a la aplicación e incluso ser perjudicial. En estos casos, UDP es un protocolo de transporte mejor.

UDP proporciona las funciones básicas para entregar segmentos de datos entre las aplicaciones adecuadas, con muy poca sobrecarga y revisión de datos. UDP se conoce como un protocolo de entrega de máximo esfuerzo. En el contexto de redes, la entrega de máximo esfuerzo se denomina “poco confiable” porque no hay reconocimiento que indique que los datos se recibieron en el destino. Con UDP, no existen procesos de capa de transporte que informen al emisor si la entrega se realizó correctamente.

9.1.2.1 Características de TCP

Para entender las diferencias entre TCP y UDP, es importante comprender la manera en que cada protocolo implementa las características específicas de confiabilidad y la forma en que realizan el seguimiento de las conversaciones. Además de admitir funciones básicas de segmentación y rearmado de datos, TCP, como se muestra en la figura, también proporciona otros servicios.

Establecimiento de una sesión

TCP es un protocolo orientado a la conexión. Un protocolo orientado a la conexión es uno que negocia y establece una conexión (o sesión) permanente entre los dispositivos de origen y de destino antes de reenviar tráfico. Mediante el establecimiento de sesión, los dispositivos negocian la cantidad de tráfico que se puede reenviar en un momento determinado, y los datos que se comunican entre ambos se pueden administrar detenidamente.

Entrega confiable

En términos de redes, la confiabilidad significa asegurar que cada segmento que envía el origen llegue al destino. Por varias razones, es posible que un segmento se dañe o se pierda por completo a medida que se transmite en la red.

Entrega en el mismo orden

Los datos pueden llegar en el orden equivocado, debido a que las redes pueden proporcionar varias rutas que pueden tener diferentes velocidades de transmisión. Al numerar y secuenciar los segmentos, TCP puede asegurar que estos se rearmen en el orden correcto.

Control del flujo

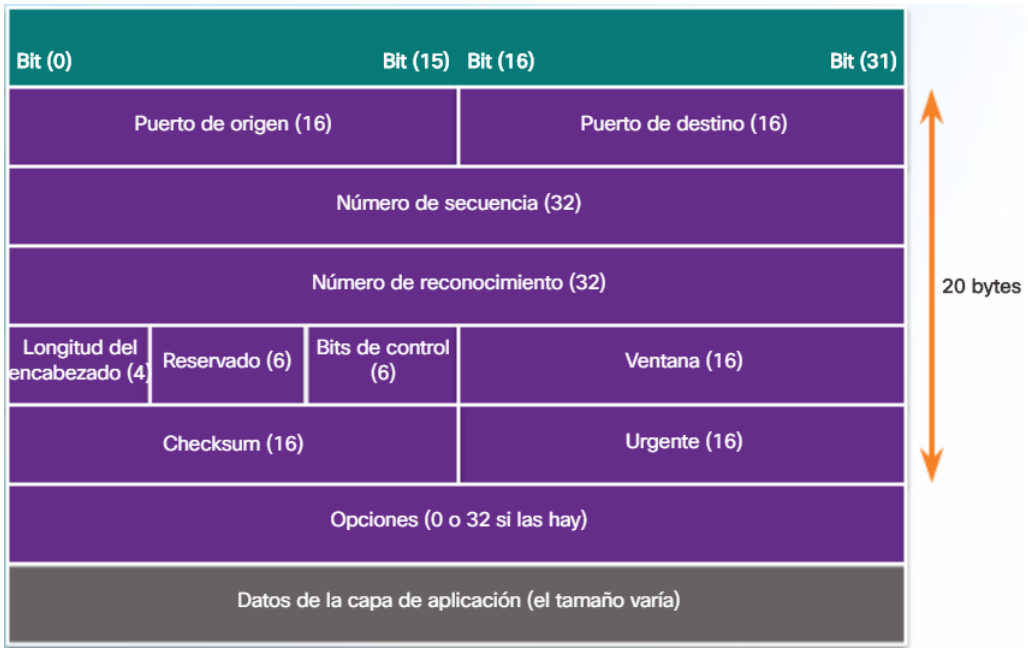
Los hosts de red tienen recursos limitados, como la memoria o la capacidad de procesamiento. Cuando TCP advierte que estos recursos están sobrecargados, puede solicitar que la aplicación emisora reduzca la velocidad del flujo de datos. Esto lo lleva a cabo TCP, que regula la cantidad de datos que transmite el origen. El control de flujo puede evitar la necesitad de retransmitir los datos cuando los recursos del host receptor están desbordados.

9.1.2.2 Encabezado TCP

TCP es un protocolo con información de estado. Un protocolo con información de estado es un protocolo que realiza el seguimiento del estado de la sesión de comunicación. Para hacer un seguimiento del estado de una sesión, TCP registra qué información se envió y qué información se reconoció. La sesión con estado comienza con el establecimiento de sesión y finaliza cuando se cierra en la terminación de sesión.

Como se muestra en la figura, cada segmento TCP tiene 20 bytes de sobrecarga en el encabezado que encapsula los datos de la capa de aplicación:

- **Puerto de origen (16 bits) y puerto de destino (16 bits)** : se utilizan para identificar la aplicación.
- **Número de secuencia (32 bits)**: se utiliza para rearmar los datos.
- **Número de reconocimiento (32 bits)**: indica los datos que se recibieron.
- **Longitud del encabezado (4 bits)**: conocido como “desplazamiento de datos”. Indica la longitud del encabezado del segmento TCP.
- **Reservado (6 bits)**: este campo está reservado para el futuro.
- **Bits de control (6 bits)**: incluye códigos de bit, o marcadores, que indican el propósito y la función del segmento TCP.
- **Tamaño de la ventana (16 bits)**: indica la cantidad de bytes que se puedan aceptar por vez.
- **Checksum (16 bits)**: se utiliza para la verificación de errores en el encabezado y los datos del segmento.
- **Urgente (16 bits)**: indica si la información es urgente.



9.1.2.3 Características de UDP

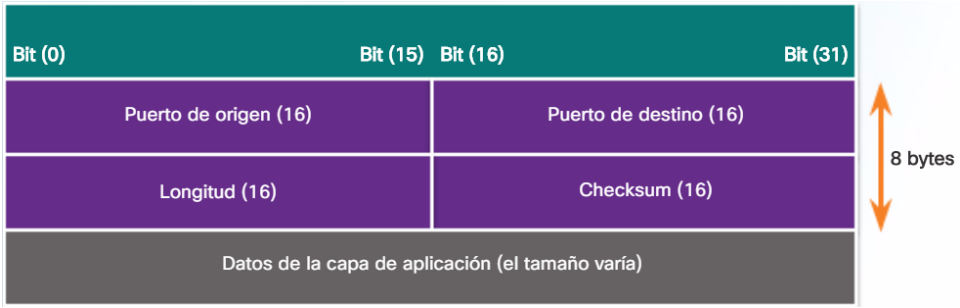
El protocolo UDP se considera un protocolo de transporte de máximo esfuerzo. UDP es un protocolo de transporte liviano que ofrece la misma segmentación y rearmado de datos que TCP, pero sin la confiabilidad y el control del flujo de TCP. UDP es un protocolo tan simple que, por lo general, se lo describe en términos de lo que no hace en comparación con TCP.

Se caracteriza por lo siguiente:

- Los datos se reconstruyen en el orden en que se recibieron.
- Los segmentos perdidos no se vuelven a enviar.
- No hay establecimiento de sesión.
- No le informa al emisor sobre la disponibilidad de recursos.

9.1.2.4 Encabezado UDP

Los fragmentos de comunicación en UDP se llaman datagramas, como se muestra en la figura. El protocolo de la capa de transporte envía estos datagramas como máximo esfuerzo. UDP tiene una sobrecarga baja de 8 bytes.



9.1.2.5 Conversaciones múltiples e independientes

La capa de transporte debe poder separar y administrar varias comunicaciones con diferentes necesidades de requisitos de transporte. TCP y UDP administran estas diferentes conversaciones simultáneas por medio de campos de encabezado que pueden identificar de manera exclusiva estas aplicaciones. Estos identificadores únicos son números de puertos.

Diferentes aplicaciones	Correo electrónico	Página HTML	Chat por Internet
Puerto	Puerto 110	Puerto 80	Puerto 531

9.1.2.6 Números de puerto

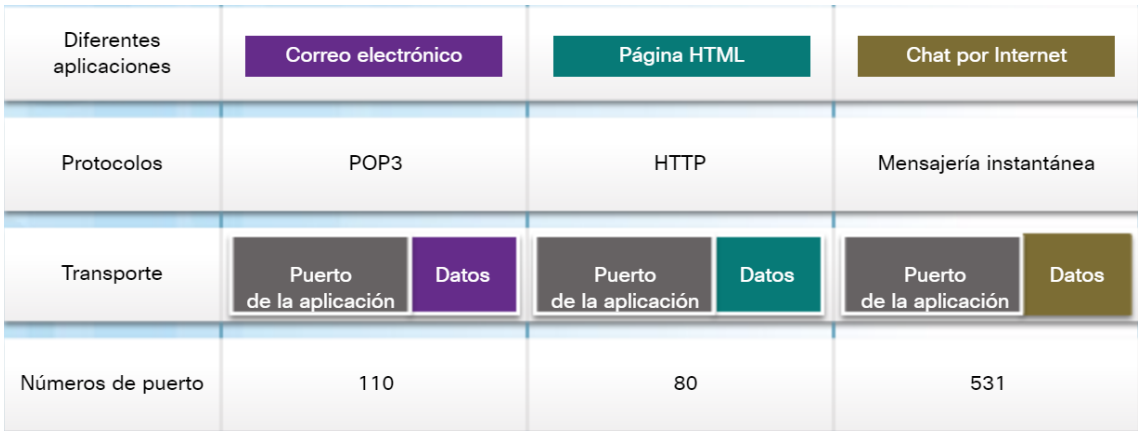
El número de puerto de origen está asociado con la aplicación que origina la comunicación en el host local. El número de puerto de destino está asociado con la aplicación de destino en el host remoto.

Puerto de origen

El número de puerto de origen es generado de manera dinámica por el dispositivo emisor para identificar una conversación entre dos dispositivos. Este proceso permite establecer varias conversaciones simultáneamente. El seguimiento de cada conversación HTTP por separado se basa en los puertos de origen.

Puerto de destino

El cliente coloca un número de puerto de destino en el segmento para informar al servidor de destino el servicio solicitado, como se muestra en la figura. Por ejemplo, cuando un cliente especifica el puerto 80 en el puerto de destino, el servidor que recibe el mensaje sabe que se solicitan servicios web.

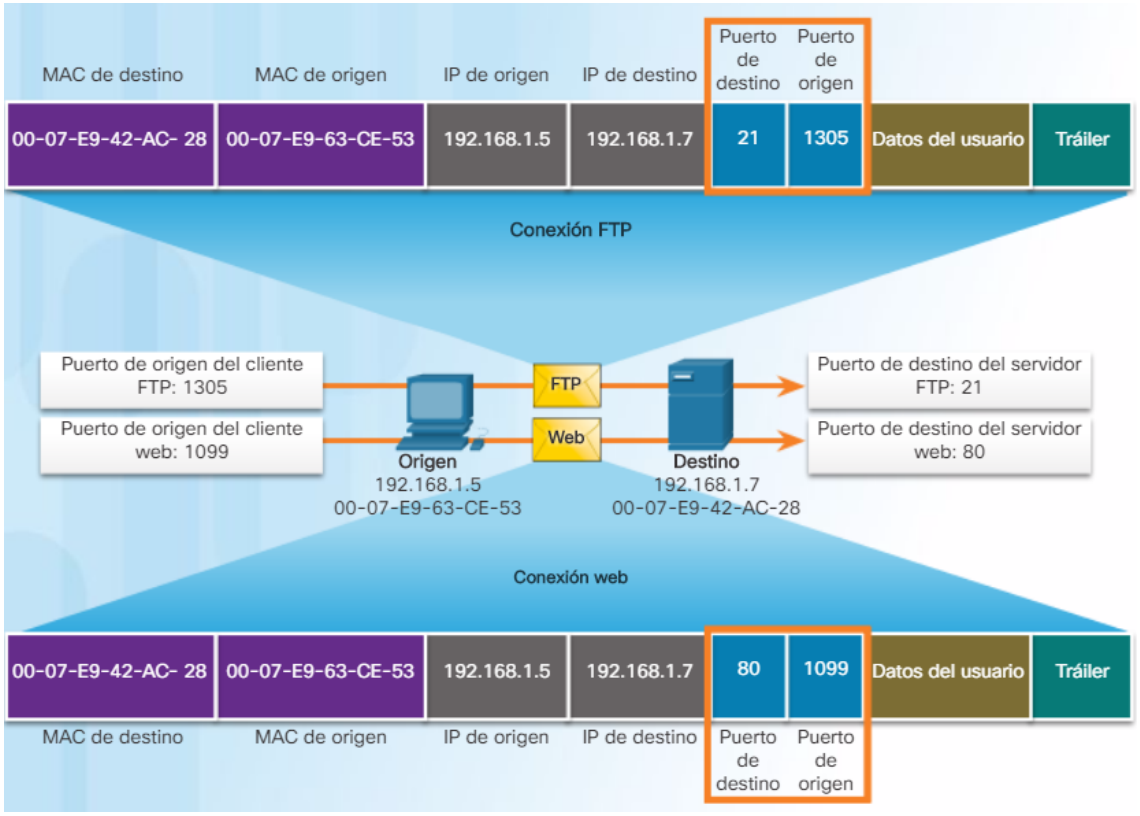


9.1.2.7 Pares de sockets

Los puertos de origen y de destino se colocan dentro del segmento. Los segmentos se encapsulan dentro de un paquete IP. El paquete IP contiene la dirección IP de origen y de destino. Se conoce como socket a la combinación de la dirección IP de origen y el número de puerto de origen, o de la dirección IP de destino y el número de puerto de destino. El socket se utiliza para identificar el servidor y el servicio que solicita el cliente.

- Un socket de cliente puede ser parecido a esto, donde 1099 representa el número de puerto de origen: 192.168.1.5:1099
- El socket en un servidor web podría ser el siguiente: 192.168.1.7:80

Juntos, estos dos sockets se combinan para formar un par de sockets: 192.168.1.5:1099, 192.168.1.7:8



9.1.2.8 Grupos de números de puerto

La Autoridad de Números Asignados de Internet (IANA) es el organismo normativo responsable de asignar los diferentes estándares de direccionamiento, incluidos los números de puerto. Existen diferentes tipos de números de puerto, como se muestra en la figura 1:

Rango de números de puerto	Grupo de puertos
Entre 0 y 1023	Puertos bien conocidos
de 1024 a 49151	Puertos registrados
de 49152 a 65535	Puertos privados y/o dinámicos

- **Puertos conocidos (números del 0 al 1023)** : estos números se reservan para servicios y aplicaciones. Por lo general, se utilizan para aplicaciones como navegadores web, clientes de correo electrónico y clientes de acceso remoto. Al definir estos puertos bien conocidos para las aplicaciones de los servidores, las aplicaciones cliente se pueden programar para solicitar una conexión a ese puerto en particular y a su servicio relacionado.

- **Puertos registrados (números del 1024 al 49151)**: IANA asigna estos números de puerto a una entidad que los solicite para utilizar con procesos o aplicaciones específicos. Principalmente, estos procesos son aplicaciones individuales que el usuario elige instalar en lugar de aplicaciones comunes que recibiría un número de puerto bien conocido. Por ejemplo, Cisco ha registrado el puerto 1985 para su proceso Hot Standby Routing Protocol (HSRP).

- **Puertos dinámicos o privados (números 49152 a 65535)**: también conocidos como puertos efímeros, usualmente el SO del cliente los asigna de forma dinámica cuando se inicia una conexión a un servicio. El puerto dinámico se utiliza para identificar la aplicación cliente durante la comunicación.

Nota: Algunos sistemas operativos cliente pueden utilizar números de puerto registrados en lugar de números de puerto dinámicos para asignar los puertos de origen.

En la figura 2 se muestran algunos números de puerto conocidos y sus aplicaciones asociadas. Algunas aplicaciones pueden utilizar TCP y UDP. Por ejemplo, DNS utiliza UDP cuando los clientes envían solicitudes a un servidor DNS. Sin embargo, la comunicación entre dos servidores DNS siempre utiliza TCP.

Número de puerto	Protocolo	Aplicación	Acrónimo
20	TCP	Protocolo de transferencia de archivos (datos)	FTP
21	TCP	Protocolo de transferencia de archivos (control)	FTP
22	TCP	Shell Seguro	SSH
23	TCP	Telnet	–
25	TCP	otocolo simple de transferencia de correo (Simple Mail Transfer Protocol)	SMTP
53	UDP, TCP	Servicio de nombres de dominios	DNS
67	UDP	Protocolo de configuración dinámica de host (servidor)	DHCP
68	UDP	Protocolo de configuración dinámica de host (cliente)	DHCP
69	UDP	Protocolo de transferencia de archivos trivial	TFTP
80	TCP	Protocolo de transferencia de hipertexto	HTTP
110	TCP	Protocolo de oficina de correos versión 3 (Post Office Protocol version 3)	POP3
143	TCP	Protocolo de acceso a mensajes de Internet (Internet Message Access Protocol)	IMAP
161	UDP	Simple Network Management Protocol	SNMP
443	TCP	Protocolo seguro de transferencia de hipertexto	HTTPS

9.1.2.9 El comando netsat

Las conexiones TCP no descritas pueden representar una importante amenaza a la seguridad. Pueden indicar que algo o alguien está conectado al host local. A veces es necesario conocer las conexiones TCP activas que están abiertas y en ejecución en el host de red. Netstat es una utilidad de red importante que puede usarse para verificar esas conexiones. Como se muestra en la figura, introduzca el comando **netstat** para obtener un listado de los protocolos que se están usando, las direcciones y los números de puerto locales, las direcciones y los números de puerto externos y el estado de la conexión.

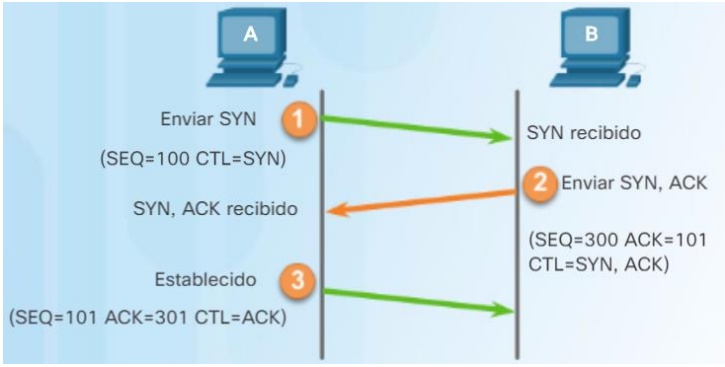
De manera predeterminada, el comando **netstat** intentará resolver direcciones IP en nombres de dominio y números de puerto en aplicaciones conocidas. La opción **-n** se puede utilizar para mostrar direcciones IP y números de puerto en su formato numérico.

```
C:\> netstat

Active Connections

Proto  Local Address  Foreign Address          State
TCP    kenpc:3126     192.168.0.2:netbios-ssn  ESTABLISHED
TCP    kenpc:3158     207.138.126.152:http     ESTABLISHED
TCP    kenpc:3159     207.138.126.169:http     ESTABLISHED
TCP    kenpc:3160     207.138.126.169:http     ESTABLISHED
TCP    kenpc:3161     sc.msn.com:http          ESTABLISHED
TCP    kenpc:3166     www.cisco.com:http       ESTABLISHED

C:\>
```

9.2.1.1 Procesos del servidor TCP

Cada proceso de aplicación que se ejecuta en el servidor está configurado para utilizar un número de puerto, ya sea predeterminado o de forma manual, por el administrador del sistema. Un servidor individual no puede tener dos servicios asignados al mismo número de puerto dentro de los mismos servicios de la capa de transporte.

9.2.1.2 Establecimiento de conexiones TCP



Paso 1: el cliente de origen solicita una sesión de comunicación de cliente a servidor con el servidor.

Paso 2: el servidor reconoce la sesión de comunicación de cliente a servidor y solicita una sesión de comunicación de servidor a cliente.

Paso 3: el cliente de origen reconoce la sesión de comunicación de servidor a cliente.

Cuando se inicia una conversación entre un host y un servidor, el host genera un número aleatorio para la secuencia. Si capturamos ese mensaje con Wireshark, ese valor nos aparecerá como 0, pero se trata de un 0 relativo; no significa que el campo de secuencia tenga verdaderamente el valor 0.

Cuando el servidor recibe la petición de inicio de sesión, genera otro número aleatorio que, según Wireshark, también será un 0 relativo. A partir de ese instante, los números de secuencia aumentarán de uno en uno por cada byte que se mande. Nótese que los números de secuencia son independientes entre el servidor y el host.

Si el host manda 10 bytes, su número de secuencia será 9 (teniendo en cuenta que se empieza a contar desde 0), pero puede ser que el servidor no haya mandado ninguno y su número de secuencia sea 0.

Si recibe los 10 bytes, el servidor mandará un ACK 10. El valor de ACK es el número de secuencia del próximo mensaje que espera recibir. Cuando se inicia la sesión, en el servidor el campo ACK vale 1 porque ha mandado un ACK al host para confirmar el inicio de sesión.

Luego el host envía otro ACK para confirmar el inicio de sesión y su campo ACK también se pone a 1.

9.2.1.3 Finalización de la sesión TCP

Para cerrar una conexión, se debe establecer el marcador de control de finalización (FIN) en el encabezado del segmento. Para finalizar todas las sesiones TCP de una vía, se utiliza un enlace de dos vías, que consta de un segmento FIN y un segmento de reconocimiento (ACK). Por lo tanto, para terminar una conversación simple admitida por TCP, se requieren cuatro intercambios para finalizar ambas sesiones.

Nota: En esta explicación, los términos “cliente” y “servidor” se utilizan como referencia con fines de simplificación, pero el proceso de finalización puede ser iniciado por dos hosts cualesquiera que tengan una sesión abierta:

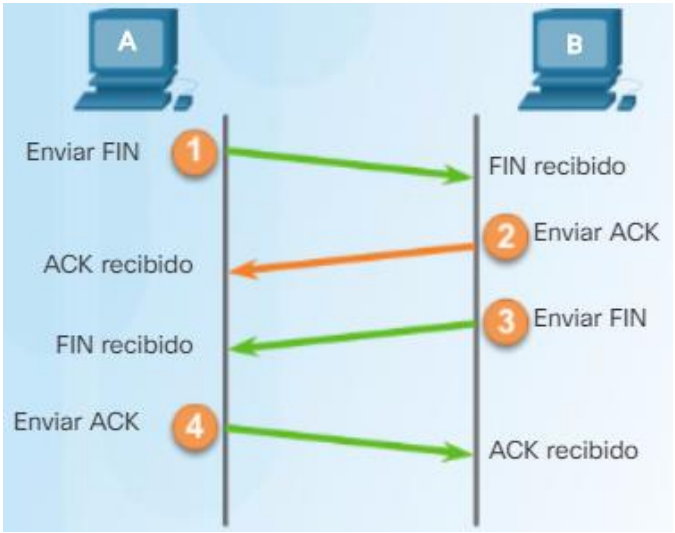
Paso 1: cuando el cliente no tiene más datos para enviar en la transmisión, envía un segmento con el marcador FIN establecido.

Paso 2: el servidor envía un ACK para reconocer el marcador FIN y terminar la sesión de cliente a servidor.

Paso 3: el servidor envía un FIN al cliente para terminar la sesión de servidor a cliente.

Paso 4: el cliente responde con un ACK para reconocer el recibo del FIN desde el servidor.

Una vez reconocidos todos los segmentos, la sesión se cierra.



9.2.1.4 Análisis del enlace de tres vías de TCP

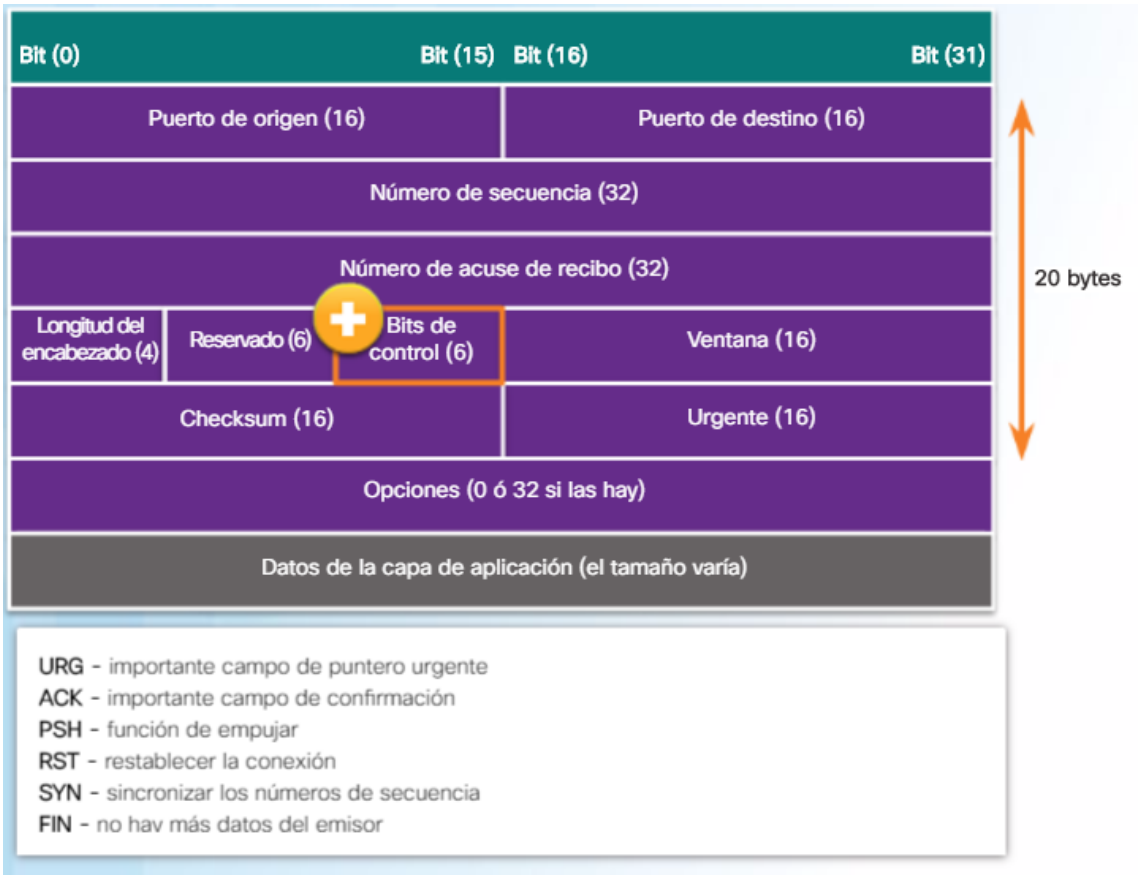
Los hosts hacen un seguimiento de cada segmento de datos dentro de una sesión e intercambian información sobre qué datos se reciben mediante la información del encabezado TCP. TCP es un protocolo dúplex completo, en el que cada conexión representa dos transmisiones de comunicación unidireccionales o sesiones. Para establecer la conexión, los hosts realizan un enlace de tres vías. Los bits de control en el encabezado TCP indican el progreso y estado de la conexión.

Enlace de tres vías:

- Establece que el dispositivo de destino se presente en la red
- Verifica que el dispositivo de destino tenga un servicio activo y que acepte solicitudes en el número de puerto de destino que el cliente de origen intenta utilizar.
- Informa al dispositivo de destino que el cliente de origen intenta establecer una sesión de comunicación en dicho número de puerto.

Una vez que se completa la comunicación, se cierran las sesiones y se finaliza la conexión. Los mecanismos de conexión y sesión habilitan la función de confiabilidad de TCP.

Los seis bits del campo de bits de control del encabezado del segmento TCP también se conocen como marcadores. Un marcador es un bit que se establece como activado o desactivado. El marcador RST se utiliza para restablecer una conexión cuando ocurre un error o se agota el tiempo de espera.



9.2.2.1 Confiabilidad de TCP: entrega ordenada

Los segmentos TCP pueden llegar a su destino desordenados. Para que el receptor comprenda el mensaje original, los datos en estos segmentos se vuelven a ensamblar en el orden original. Para lograr esto, se asignan números de secuencia en el encabezado de cada paquete. El número de secuencia representa el primer byte de datos del segmento TCP.

Durante la configuración de la sesión, se establece un número de secuencia inicial (ISN). Este ISN representa el valor inicial de los bytes para esta sesión que se transmite a la aplicación receptora. A medida que se transmiten los datos durante la sesión, el número de secuencia se incrementa según el número de bytes que se han transmitido. Este seguimiento de bytes de datos permite identificar y reconocer cada segmento de manera exclusiva. A partir de esto, se pueden identificar segmentos perdidos.

Nota: El ISN no comienza en uno, sino que se trata de un número aleatorio. Esto permite evitar ciertos tipos de ataques maliciosos. Para mayor simplicidad, usaremos un ISN de 1 para los ejemplos de este capítulo.

Los números de secuencia de segmento indican cómo reensamblar y reordenar los segmentos recibidos, como se muestra en la figura.

El proceso TCP receptor coloca los datos del segmento en un búfer de recepción. Los segmentos se colocan en el orden de secuencia correcto y se pasan a la capa de aplicación cuando se vuelven a ensamblar. Todos los segmentos que lleguen con números de secuencia desordenados se retienen para su posterior procesamiento. A continuación, cuando llegan los segmentos con bytes faltantes, tales segmentos se procesan en orden.

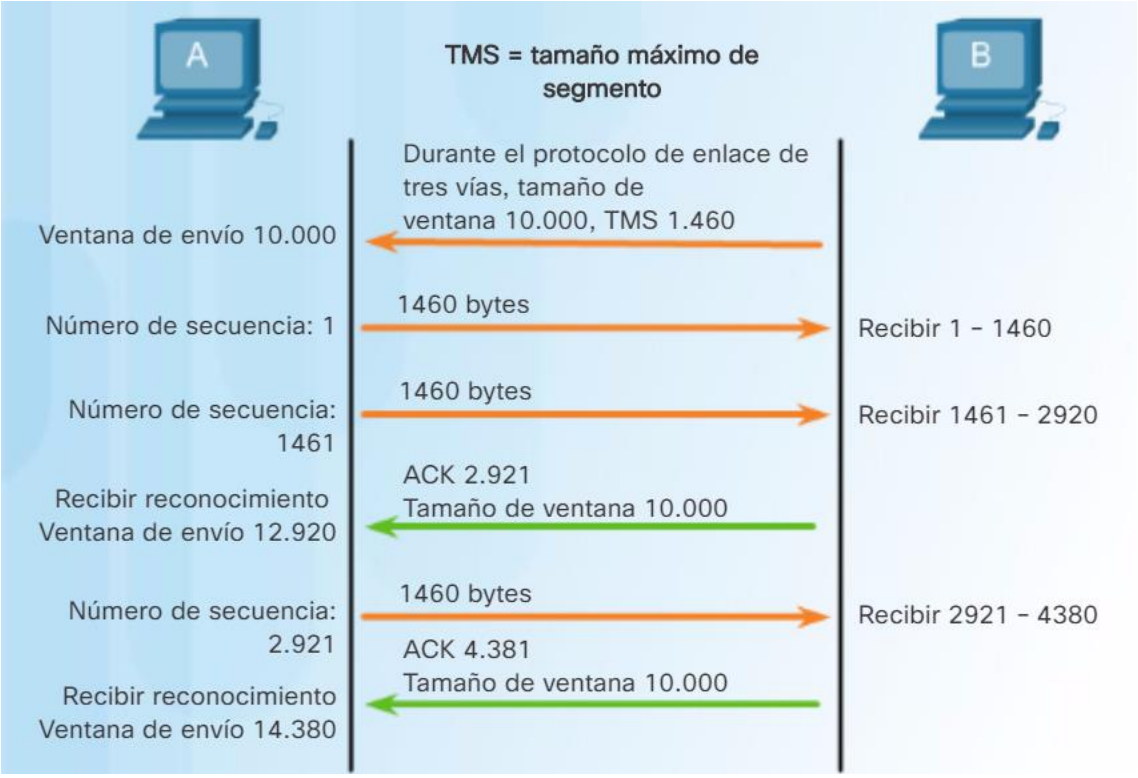
9.2.2.4 Control de flujo de TCP: tamaño de la ventana y reconocimientos

TCP también ofrece mecanismos de control de flujo, la cantidad de datos que el destino puede recibir y procesar con fiabilidad. El control de flujo permite mantener la confiabilidad de la transmisión de TCP mediante el ajuste de la velocidad del flujo de datos entre el origen y el destino para una sesión dada. Para lograr esto, el encabezado TCP incluye un campo de 16 bits llamado “tamaño de la ventana”.

Nota: En la figura, el origen está transmitiendo 1460 bytes de datos dentro de cada segmento TCP. Esto se conoce como el MMS (tamaño de segmento máximo).

El tamaño inicial de la ventana se acuerda cuando se establece la sesión TCP durante la realización del enlace de tres vías. El dispositivo de origen debe limitar la cantidad de bytes enviados al dispositivo de destino en función del tamaño de la ventana de destino. El dispositivo de origen puede continuar enviando más datos para la sesión solo cuando obtiene un reconocimiento de los bytes recibidos

Por lo general, el receptor envía un acuse de recibo cada dos segmentos recibidos. El número de segmentos recibidos antes de que se acuse recibo puede variar.



9.2.2.5 Control de flujo de TCP: prevención de congestiones

Cuando se produce congestión en una red, el router sobrecargado comienza a descartar paquetes. Cuando los paquetes que contienen los segmentos TCP no llegan a su destino, se quedan sin reconocimiento. Mediante la determinación de la tasa a la que se envían pero no se reconocen los segmentos TCP, el origen puede asumir un cierto nivel de congestión de la red.

Siempre que haya congestión, se producirá la retransmisión de los segmentos TCP perdidos del origen. Si la retransmisión no se controla adecuadamente, la retransmisión adicional de los segmentos TCP puede empeorar aún más la congestión. No sólo se introducen en la red los nuevos paquetes con segmentos TCP, sino que el efecto de retroalimentación de los segmentos TCP retransmitidos que se perdieron también se sumará a la congestión. Para evitar y controlar la congestión, TCP emplea varios mecanismos, temporizadores y algoritmos de manejo de la congestión.

Si el origen determina que los segmentos TCP no están siendo reconocidos o que sí son reconocidos pero no de una manera oportuna, entonces puede reducir el número de bytes que envía antes de recibir un reconocimiento. Tenga en cuenta que es el origen el que está reduciendo el número de bytes sin reconocimiento que envía y no el tamaño de ventana determinado por el destino.

Nota: La explicación de los mecanismos, temporizadores y algoritmos reales de manejo de la congestión se encuentra fuera del alcance de este curso.

9.2.3.1 Comparación de baja sobrecarga y confiabilidad de UDP

UDP es un protocolo simple que proporciona las funciones básicas de la capa de transporte. Tiene una sobrecarga mucho menor que TCP, ya que no está orientado a la conexión y no proporciona los mecanismos sofisticados de retransmisión, secuenciación y control de flujo que ofrecen confiabilidad.

Esto no significa que las aplicaciones que utilizan UDP sean siempre poco confiables ni que UDP sea un protocolo inferior. Solo quiere decir que estas funciones no las proporciona el protocolo de la capa de transporte, y se deben implementar aparte, si fuera necesario.

La baja sobrecarga del UDP es muy deseable para los protocolos que realizan transacciones simples de solicitud y respuesta. Por ejemplo, usar TCP para DHCP introduciría una cantidad innecesaria de tráfico de red. Si existe un problema con una solicitud o una respuesta, el dispositivo simplemente envía la solicitud nuevamente si no se recibe ninguna respuesta.

9.2.3.2 Reensamblaje de datagramas de UDP

Tal como los segmentos con TDP, cuando se envían datagramas UDP a un destino, a menudo toman diferentes rutas y llegan en el orden equivocado. UDP no realiza un seguimiento de los números de secuencia de la manera en que lo hace TCP. UDP no tiene forma de reordenar datagramas en el orden en que se transmiten, como se muestra en la ilustración.

Por lo tanto, UDP simplemente reensambla los datos en el orden en que se recibieron y los envía a la aplicación. Si la secuencia de datos es importante para la aplicación, esta debe identificar la secuencia adecuada y determinar cómo se deben procesar los datos.

9.2.3.3 Procesos de solicitudes del servidor UDP

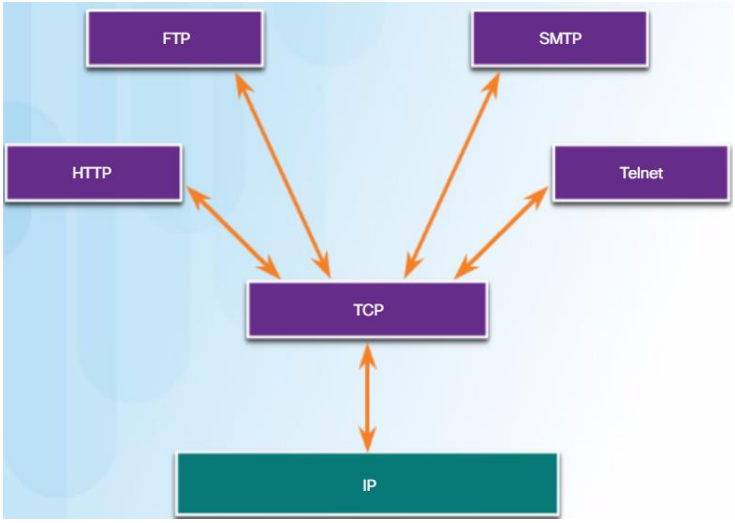
Al igual que las aplicaciones basadas en TCP, a las aplicaciones de servidor basadas en UDP se les asignan números de puerto conocidos o registrados, como se muestra en la figura. Cuando estas aplicaciones o estos procesos se ejecutan en un servidor, aceptan los datos que coinciden con el número de puerto asignado. Cuando UDP recibe un datagrama destinado a uno de esos puertos, envía los datos de aplicación a la aplicación adecuada en base a su número de puerto.

9.2.3.4 Procesos de cliente UDP

Como en TCP, la comunicación cliente-servidor es iniciada por una aplicación cliente que solicita datos de un proceso de servidor. El proceso de cliente UDP selecciona dinámicamente un número de puerto del intervalo de números de puerto y lo utiliza como puerto de origen para la conversación. Por lo general, el puerto de destino es el número de puerto bien conocido o registrado que se asigna al proceso de servidor.

Una vez que el cliente selecciona los puertos de origen y de destino, este mismo par de puertos se utiliza en el encabezado de todos los datagramas que se utilizan en la transacción. Para la devolución de datos del servidor al cliente, se invierten los números de puerto de origen y destino en el encabezado del datagrama.

9.2.4.1 Aplicaciones que utilizan TCP



9.4.2.4 Aplicaciones que utilizan UDP

Existen tres tipos de aplicaciones que son las más adecuadas para UDP:

- **Aplicaciones multimedia y vídeo en vivo:** Los ejemplos incluyen VoIP y la transmisión de vídeo en vivo.
- **Aplicaciones con solicitudes y respuestas simples**: Por ejemplo, DNS y DHCP.
- **Aplicaciones que manejan la confiabilidad por su cuenta**: Por ejemplo, SNMP y TFTP.

Aunque DNS y SNMP utilizan UDP de manera predeterminada, ambos también pueden utilizar TCP. DNS utilizará TCP si la solicitud DNS o la respuesta DNS tienen más de 512 bytes, como cuando una respuesta DNS incluye un gran número de resoluciones de nombres. Del mismo modo, en algunas situaciones, el administrador de redes puede querer configurar SNMP para utilizar TCP.