14.0 Introducción

Desplázate para empezar

14.0.1 ¿Por qué debería tomar este módulo?

¡Bienvenido a Capa de transporte!

La capa de transporte es donde, como su nombre indica, los datos se transportan de un host a otro. ¡Aquí es donde su red realmente se mueve! La capa de transporte utiliza dos protocolos: TCP y UDP. Piense en TCP como obtener una carta registrada en el correo. Tienes que firmarlo antes de que el transportista de correo te lo permita. Esto ralentiza un poco el proceso, pero el remitente sabe con certeza que recibió la carta y cuándo la recibió. UDP es más como una carta regular y sellada. Llega a tu buzón de correo y, si lo hace, probablemente esté destinado a ti, pero en realidad podría ser para alguien más que no vive allí. Además, es posible que no llegue a su buzón de correo en absoluto. El remitente no puede estar seguro de que lo recibió. Sin embargo, hay momentos en que UDP, como una carta estampada, es el protocolo que se necesita. En este tema se profundiza en cómo funcionan TCP y UDP en la capa de transporte. Más adelante en este módulo hay varios videos para ayudarle a entender estos procesos.

14.0.2 ¿Qué aprenderé en este módulo?

Título del módulo: Capa de transporte

Objetivos del módulo: Compare el funcionamiento de los protocolos de capa de transporte en la admisión de la comunicación de extremo a extremo.

14.1 Transporte de datos

Desplázate para empezar

14.1.1 Función de la Capa de Transporte

Los programas de capa de aplicación generan datos que deben intercambiarse entre los hosts de origen y de destino. La capa de transporte es responsable de las comunicaciones lógicas entre aplicaciones que se ejecutan en diferentes hosts. Esto puede incluir servicios como el establecimiento de una sesión temporal entre dos hosts y la transmisión fiable de información para una aplicación.

Como se muestra en la ilustración, la capa de transporte es el enlace entre la capa de aplicación y las capas inferiores que son responsables de la transmisión a través de la red.

muestra un diagrama de cómo los dispositivos utilizan la capa de transporte para mover datos entre aplicaciones en el modelo TCP/IP

La capa de transporte traslada los datos entre las aplicaciones en dispositivos de la red.

La capa de transporte no tiene conocimiento del tipo de host de destino, el tipo de medio por el que deben viajar los datos, la ruta tomada por los datos, la congestión en un enlace o el tamaño de la red.

La capa de transporte incluye dos protocolos:

- Protocolo de Control de Transmisión (TCP)
- Protocolo de Datagramas de Usuario (UDP)

14.1.2 Responsabilidades de la Capa de Transporte

La capa de transporte tiene muchas responsabilidades.

Haga clic en cada botón para obtener más información.
Seguimiento de conversaciones individuales
Segmentación de Datos y Rearmado de Segmentos
Agregar Información de Encabezado
Identificación de las Aplicaciones

Multiplexión de Conversaciones

Seguimiento de conversaciones individuales

En la capa de transporte, cada conjunto de datos que fluye entre una aplicación de origen y una aplicación de destino se conoce como una conversación y se rastrea por separado. Es responsabilidad de la capa de transporte mantener y hacer un seguimiento de todas estas conversaciones.

Como se ilustra en la figura, un host puede tener múltiples aplicaciones que se comunican a través de la red simultáneamente.

La mayoría de las redes tienen un límite de la cantidad de datos que se puede incluir en un solo paquete. Por lo tanto, los datos deben dividirse en piezas manejables.

El PC ejecuta simultáneamente varias aplicaciones de red, incluyendo un cliente de correo electrónico, cliente de mensajería instantánea, páginas web del navegador web, streaming de vídeo y un cliente de videoconferencia.

Para: usted@ejemplo.com
De: yo@ejemplo.com
Asunto: Vacaciones
Correo electrónico
Online Video Chat

Transmisión de Vídeo
Varias Páginas Web
Mensajería Instantánea
Red

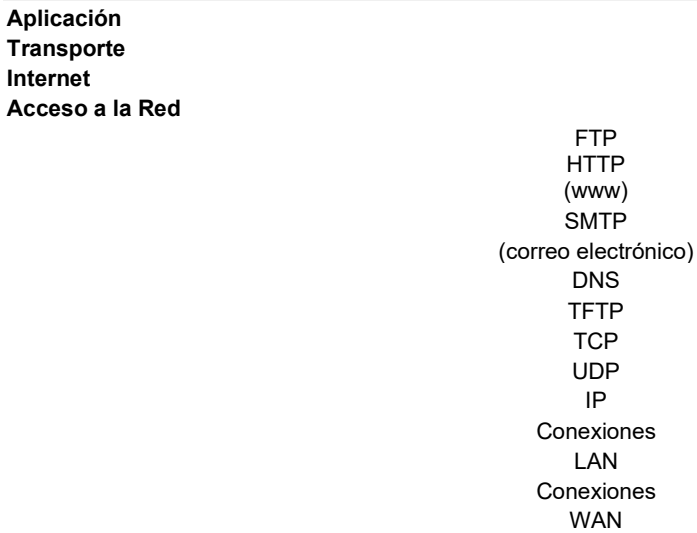
14.1.3 Protocolos de Capa de Transporte

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 capa de transporte especifican cómo transferir mensajes entre hosts y son responsables de administrar los requisitos de fiabilidad de una conversación. La capa de transporte incluye los protocolos TCP y UDP.

Las diferentes aplicaciones tienen diferentes requisitos de confiabilidad de transporte. Por lo tanto, TCP/IP proporciona dos protocolos de capa de transporte, como se muestra en la figura.

muestra cómo los protocolos de capa de aplicación como FTP, HTTP, SMTP utilizan TCP en la capa de transporte y DNS y TFTP utilizan UDP. Cómo todos usan IP en la capa de Internet independientemente de si se conectan a una LAN o a una WAN en la capa de acceso a la red



14.1.4 Protocolo de Control de Transmisión (TCP)

IP solo se refiere a la estructura, direccionamiento y enrutamiento de paquetes, desde el remitente original hasta el destino final. IP no es responsable de garantizar la entrega o determinar si es necesario establecer una conexión entre el remitente y el receptor.

El TCP se considera un protocolo de la capa de transporte confiable y completo, que garantiza que todos los datos lleguen al destino. TCP incluye campos que garantizan la entrega de los datos de la aplicación. Estos campos requieren un procesamiento adicional por parte de los hosts de envío y recepción.

Nota: TCP divide los datos en segmentos.

La función del protocolo de transporte TCP es similar al envío de paquetes de los que se hace un rastreo de origen a destino. Si se divide un pedido de envío en varios paquetes, un cliente puede verificar en línea para ver el orden de la entrega.

TCP proporciona confiabilidad y control de flujo mediante estas operaciones básicas:

- Enumerar y rastrear segmentos de datos transmitidos a un host específico desde una aplicación específica
- Confirmar datos recibidos
- Retransmitir cualquier información no reconocida después de un cierto período de tiempo
- Secuenciar datos que pueden llegar en un orden incorrecto

- Enviar datos a una velocidad eficiente que sea aceptable por el receptor

Para mantener el estado de una conversación y realizar un seguimiento de la información, TCP debe establecer primero una conexión entre el remitente y el receptor. Es por eso que TCP se conoce como un protocolo orientado a la conexión.

Haga clic en Reproducir en la figura para ver cómo los segmentos de TDP y los reconocimientos se transmiten entre el emisor y el receptor.

la animación muestra una conexión a un servidor FTP iniciada con un protocolo de enlace TCP de 3 vías y los segmentos de datos contabilizados mediante números de secuencia y confirmaciones

Se envía un archivo a un servidor con la aplicación del Protocolo de Transferencia de Archivos (FTP). TCP rastrea la conversación y divide los datos que se enviarán en 6 segmentos.
De seis segmentos, se reenvían al servidor los primeros tres.
El servidor de archivos reconoce los tres primeros segmentos recibidos.
El cliente reenvía los siguientes tres segmentos.
No se recibe ningún segmento; no se envía ningún acuse de recibo.
El cliente reenvía los últimos tres segmentos.
Se reciben los últimos tres segmentos y se los reconoce
ISP 1
FTP
Granja de Servidores
Internet
ISP 2

play_circle_filled

14.1.5 Protocolo de Datagramas de Usuario (UDP)

UDP es un protocolo de capa de transporte más simple que TCP. No proporciona confiabilidad y control de flujo, lo que significa que requiere menos campos de encabezado. Debido a que los procesos UDP remitente y receptor no tienen que administrar la confiabilidad y el control de flujo, esto significa que los datagramas UDP se pueden procesar más rápido que los segmentos TCP. El UDP proporciona las funciones básicas para entregar segmentos de datos entre las aplicaciones adecuadas, con muy poca sobrecarga y revisión de datos.

Nota: UDP divide los datos en datagramas que también se conocen como segmentos.

UDP es un protocolo sin conexión. Debido a que UDP no proporciona fiabilidad ni control de flujo, no requiere una conexión establecida. Debido a que UDP no realiza un seguimiento de la información enviada o recibida entre el cliente y el servidor, UDP también se conoce como protocolo sin estado.

UDP también se conoce como un protocolo de entrega de mejor esfuerzo porque no hay reconocimiento de que los datos se reciben en el destino. Con UDP, no existen procesos de capa de transporte que informen al emisor si la entrega se realizó correctamente.

UDP es como colocar una carta regular, no registrada, en el correo. El emisor de la carta no conoce la disponibilidad del receptor para recibir la carta. Además, la oficina de correos tampoco es responsable de hacer un rastreo de la carta ni de informar al emisor si esta no llega a destino.

Haga clic en Reproducir en la figura para ver una animación de los datagramas UDP que se transmiten del remitente al receptor.

muestra una conexión a un servidor TFTP utilizando datagramas UDP que se envían sin números de secuencia o confirmaciones

Se envía un archivo a un servidor con la aplicación del Protocolo trivial de Transferencia de Archivos (TFTP). UDP divide los datos en datagramas y los envía utilizando la entrega del mejor esfuerzo.
El servidor de archivos recibe los seis segmentos; no se envía acuse de recibo.

ISP 1
TFTP
Granja de Servidores
Internet
ISP 2

play_circle_filled

14.1.6 Protocolo de la capa de transporte correcto para la aplicación adecuada

Algunas aplicaciones pueden tolerar cierta pérdida de datos durante la transmisión a través de la red, pero los retrasos en la transmisión son inaceptables. Para estas aplicaciones, UDP es la mejor opción porque requiere menos sobrecarga de red. UDP es preferible para aplicaciones como Voz sobre IP (VoIP). Los reconocimientos y la retransmisión retrasarían la entrega y harían inaceptable la conversación de voz.

UDP también es utilizado por las aplicaciones de solicitud y respuesta donde los datos son mínimos, y la retransmisión se puede hacer rápidamente. Por ejemplo, el servicio de nombres de dominio (DNS) utiliza UDP para este tipo de transacción. El cliente solicita direcciones IPv4 e IPv6 para obtener un nombre de dominio conocido desde un servidor DNS. Si el cliente no recibe una respuesta en un período de tiempo predeterminado, simplemente envía la solicitud de nuevo.

Por ejemplo, si uno o dos segmentos de una transmisión de vídeo en vivo no llegan al destino, se interrumpe momentáneamente la transmisión. Esto puede manifestarse como distorsión en la imagen o el sonido, pero puede no ser perceptible para el usuario. Si el dispositivo de destino tuviera que dar cuenta de los datos perdidos, la transmisión se podría demorar mientras espera las retransmisiones, lo que ocasionaría que la imagen o el sonido se degraden considerablemente. En este caso, es mejor producir el mejor vídeo o audio posible con los segmentos recibidos y prescindir de la confiabilidad.

Para otras aplicaciones es importante que todos los datos lleguen y que puedan ser procesados en su secuencia adecuada. Para estos tipos de aplicaciones, TCP se utiliza como protocolo de transporte. Por ejemplo, las aplicaciones como las bases de datos, los navegadores web y los clientes de correo electrónico, requieren que todos los datos que se envían lleguen a destino en su formato original. Cualquier dato faltante podría corromper una comunicación, haciéndola incompleta o ilegible. Por ejemplo, cuando se accede a la información bancaria a través de la web, es importante asegurarse de que toda la información se envía y recibe correctamente.

Los desarrolladores de aplicaciones deben elegir qué tipo de protocolo de transporte es adecuado según los requisitos de las aplicaciones. El vídeo puede enviarse a través de TCP o UDP. Las aplicaciones que transmiten audio y video almacenado generalmente usan TCP. La aplicación utiliza TCP para realizar buffering, sondeo de ancho de banda y control de congestión, con el fin de controlar mejor la experiencia del usuario.

El vídeo y la voz en tiempo real generalmente usan UDP, pero también pueden usar TCP, o tanto UDP como TCP. Una aplicación de videoconferencia puede usar UDP de forma predeterminada, pero debido a que muchos firewalls bloquean UDP, la aplicación también se puede enviar a través de TCP.

Las aplicaciones que transmiten audio y video almacenado usan TCP. Por ejemplo, si de repente la red no puede admitir el ancho de banda necesario para ver una película a pedido, la aplicación detiene la reproducción. Durante la pausa, es posible que vea un mensaje de "almacenando en búfer..." mientras TCP intenta restablecer la transmisión. Cuando todos los segmentos están en orden y se restaura un nivel mínimo de ancho de banda, se reanuda la sesión TCP y se reanuda la reproducción de la película.

La figura resume las diferencias entre UDP y TCP.

enumera las diferencias entre UDP: sobrecarga rápida, baja, sin confirmaciones, sin reenvío y TCP: confiable, reconoce datos, reenvía datos perdidos y entrega datos con números de secuencia
TCP
UDP

VoIP
(telefonía IP)
DNS
(Domain Nameresolución de nombre de dominio Resolution)
SMTP/IMAP
(Correo electrónico)
HTTP/HTTPS
(World Wide Web)

Propiedades de protocolo requeridas:

- Rápido
- Baja sobrecarga
- No requiere reconocimiento
- No reenvía los datos perdidos
- Entrega los datos a medida que llegan

Propiedades de protocolo requeridas:

- Confiable
- Reconoce los datos
- Reenvía los datos perdidos
- Entrega los datos en orden secuencial

14.2 Descripción general de TCP

Desplázate para empezar

14.2.1 Características de TCP

En el tema anterior, aprendió que TCP y UDP son los dos protocolos de capa de transporte. Este tema proporciona más detalles sobre lo que hace TCP y cuándo es una buena idea usarlo en lugar de UDP.

Para comprender las diferencias entre TCP y UDP, es importante comprender cómo cada protocolo implementa funciones de confiabilidad específicas y cómo cada protocolo rastrea las conversaciones.

Además de admitir las funciones básicas de segmentación y reensamblado de datos, TCP también proporciona los siguientes servicios:

- **Establece una sesión** - TCP es un protocolo orientado a la conexión que negocia y establece una conexión permanente (o sesión) entre los dispositivos de origen y destino antes de reenviar cualquier 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.
- **Garantiza una entrega confiable** - por muchas razones, es posible que un segmento se corrompa o se pierda por completo, ya que se transmite a través de la red. TCP asegura que cada segmento que envía la fuente llega al destino.
- **Proporciona entrega en el mismo pedido** - Debido a que las redes pueden proporcionar múltiples rutas que pueden tener diferentes velocidades de transmisión, los datos pueden llegar en el orden incorrecto. Al numerar y secuenciar los segmentos, TCP garantiza que los segmentos se vuelvan a ensamblar en el orden correcto.
- **Admite control de flujo** - los hosts de red tienen recursos limitados (es decir, memoria y potencia 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 un cabo TCP, que regula la cantidad de datos que transmite el origen. El control de flujo puede evitar la necesidad de retransmitir los datos cuando los recursos del host receptor se ven desbordados.

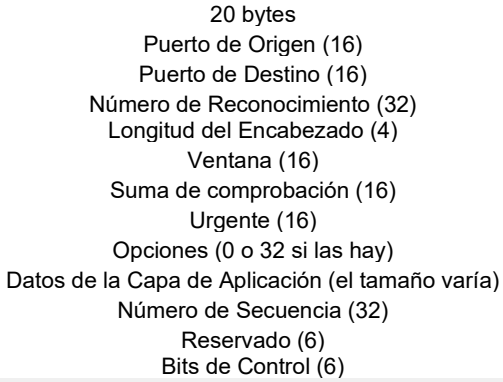
Para obtener más información sobre TCP, busque en Internet el RFC 793.

14.2.2 Encabezado TCP

TCP es un protocolo con estado, lo que significa que realiza un 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 la sesión y termina con la finalización de la sesión.

Un segmento TCP agrega 20 bytes (es decir, 160 bits) de sobrecarga al encapsular los datos de la capa de aplicación. La figura muestra los campos en un encabezado TCP.

muestra los campos en el encabezado TCP



14.2.3 Campos de Encabezado TCP

La tabla identifica y describe los diez campos de un encabezado TCP.

14.2.4 Aplicaciones que utilizan TCP

TCP es un buen ejemplo de cómo las diferentes capas del conjunto de protocolos TCP / IP tienen roles específicos. TCP maneja todas las tareas asociadas con la división del flujo de datos en segmentos, proporcionando confiabilidad, controlando el flujo de datos y reordenando segmentos. TCP libera la aplicación de tener que administrar estas tareas. Las aplicaciones, como las que se muestran en la figura, simplemente puede enviar el flujo de datos a la capa de transporte y utilizar los servicios de TCP.

muestra flechas que señalan ambas direcciones desde HTTP, FTP, SMTP y SSH a TCP y, a continuación, desde TCP a IP

TCP
IP

14.3 Visión generalde UDP

Desplázate para empezar

14.3.1 Características de UDP

Este tema abarcará UDP, lo que hace y cuándo es una buena idea usarlo en lugar de TCP. UDP es un protocolo de transporte del mejor 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.

Las características UDP incluyen 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.
- El envío no está informado sobre la disponibilidad de recursos.

Para obtener más información sobre UDP, busque en Internet el RFC.

14.3.2 Encabezado UDP

UDP es un protocolo sin estado, lo que significa que ni el cliente ni el servidor rastrean el estado de la sesión de comunicación. Si se requiere confiabilidad al utilizar UDP como protocolo de transporte, a esta la debe administrar la aplicación.

Uno de los requisitos más importantes para transmitir video en vivo y voz a través de la red es que los datos fluyan rápidamente. Las aplicaciones de video y de voz en vivo pueden tolerar cierta pérdida de datos con un efecto mínimo o imperceptible, y se adaptan perfectamente a UDP.

Los bloques de comunicación en UDP se denominan datagramas o segmentos. Estos datagramas se envían como el mejor esfuerzo por el protocolo de la capa de transporte.

El encabezado UDP es mucho más simple que el encabezado TCP porque solo tiene cuatro campos y requiere 8 bytes (es decir, 64 bits). La figura muestra los campos en un encabezado TCP.

El diagrama de datagramas UDP muestra 4 campos de encabezado: puerto de origen, puerto de destino, longitud y suma de comprobación, así como los datos de capa de aplicación sin encabezado
8 bytes

Puerto de Origen (16)
Puerto de Destino (16)
Longitud (16)
Suma de Comprobación (16)
Datos de la Capa de Aplicación (el tamaño varía)

14.3.3 Campos de Encabezado UDP

La tabla identifica y describe los cuatro campos de un encabezado UDP.

14.3.4 Aplicaciones que utilizan UDP

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

- **Aplicaciones de video y multimedia en vivo:** - estas aplicaciones pueden tolerar cierta pérdida de datos, pero requieren poco o ningún retraso. Los ejemplos incluyen VoIP y la transmisión de video en vivo.
- **Solicitudes simples de solicitud y respuesta:** - aplicaciones con transacciones simples en las que un host envía una solicitud y puede o no recibir una respuesta. Los ejemplos incluyen DNS y DHCP.
- **Aplicaciones que manejan la confiabilidad por sí mismas:** - comunicaciones unidireccionales donde el control de flujo, la detección de errores, los reconocimientos y la recuperación de errores no son necesarios o la aplicación puede manejarlos. Los ejemplos incluyen SNMP y TFTP.

La figura identifica las aplicaciones que requieren UDP.

muestra flechas que señalan ambas direcciones desde DHCP, DNS, SNMP, TFTP, VoIP e IPTV a UDP y, a continuación, desde UDP a IP

UDP
IP
SNMP
TFTP
Envenenamiento
VoIP
DHCP
Videoconferencia

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

14.4 Números de puerto

Desplázate para empezar

14.4.1 Comunicaciones Múltiples Separadas

Como ha aprendido, hay algunas situaciones en las que TCP es el protocolo correcto para el trabajo, y otras situaciones en las que se debe usar UDP. Independientemente del tipo de datos que se transporten, tanto TCP como UDP utilizan números de puerto.

Los protocolos de capa de transporte TCP y UDP utilizan números de puerto para administrar múltiples conversaciones simultáneas. Como se muestra en la figura, los campos de encabezado TCP y UDP identifican un número de puerto de aplicación de origen y destino.

muestra los campos de encabezado del puerto de origen y del puerto de destino que son de 2 bytes cada uno

Puerto de origen (16)
Puerto de destino (16)

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

Por ejemplo, supongamos que un host está iniciando una solicitud de página web desde un servidor web. Cuando el host inicia la solicitud de página web, el host genera dinámicamente el número de puerto de origen para identificar de forma exclusiva la conversación. Cada solicitud generada por un host utilizará un número de puerto de origen creado dinámicamente diferente. Este proceso permite establecer varias conversaciones simultáneamente.

En la solicitud, el número de puerto de destino es lo que identifica el tipo de servicio que se solicita del servidor web de destino. 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.

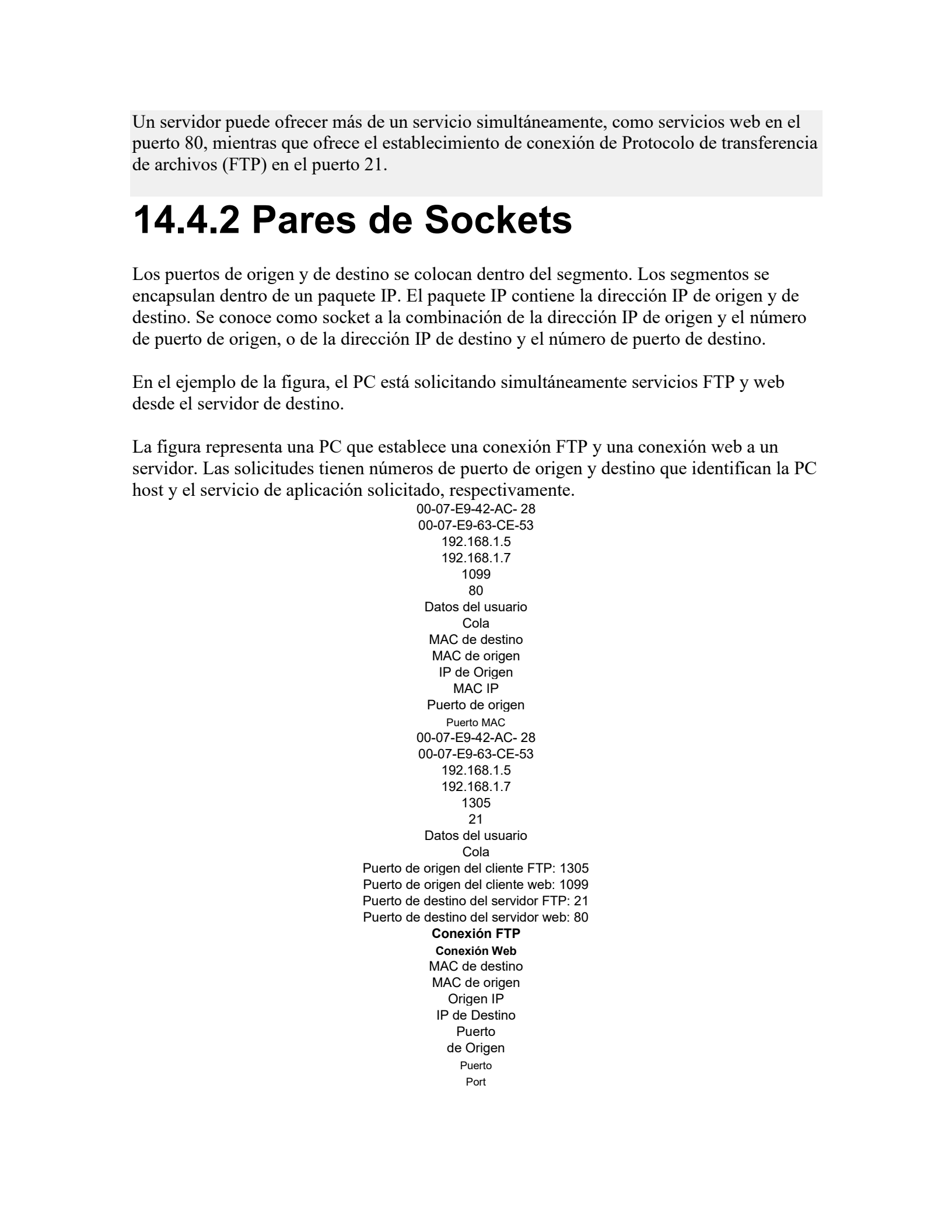
Un servidor puede ofrecer más de un servicio simultáneamente, como servicios web en el puerto 80, mientras que ofrece el establecimiento de conexión de Protocolo de transferencia de archivos (FTP) en el puerto 21.

14.4.2 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.

En el ejemplo de la figura, el PC está solicitando simultáneamente servicios FTP y web desde el servidor de destino.

La figura representa una PC que establece una conexión FTP y una conexión web a un servidor. Las solicitudes tienen números de puerto de origen y destino que identifican la PC host y el servicio de aplicación solicitado, respectivamente.



Origen
192.168.1.5
00-07-E9-63-CE-53
Destino
192.168.1.7
00-07-E9-42-AC-28
FTP
Web

En el ejemplo, la solicitud FTP generada por el PC incluye las direcciones MAC de Capa 2 y las direcciones IP de Capa 3. La solicitud también identifica el puerto de origen 1305 (es decir, generado dinámicamente por el host) y el puerto de destino, identificando los servicios FTP en el puerto 21. El host también ha solicitado una página web del servidor utilizando las mismas direcciones de Capa 2 y Capa 3. Sin embargo, está utilizando el número de puerto de origen 1099 (es decir, generado dinámicamente por el host) y el puerto de destino que identifica el servicio web en el puerto 80.

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 puede ser 192.168.1.7:80

Juntos, estos dos sockets se combinan para formar un par de zócalos: 192.168.1.5:1099, 192.168.1.7:80

Los sockets permiten que los diversos procesos que se ejecutan en un cliente se distingan entre sí. También permiten la diferenciación de diferentes conexiones a un proceso de servidor.

El número de puerto de origen actúa como dirección de retorno para la aplicación que realiza la solicitud. La capa de transporte hace un seguimiento de este puerto y de la aplicación que generó la solicitud de manera que cuando se devuelva una respuesta, esta se envíe a la aplicación correcta.

14.4.3 Grupos de números de puerto

La Autoridad de Números Asignados de Internet (IANA) es la organización de estándares responsable de asignar varios estándares de direccionamiento, incluidos los números de puerto de 16 bits. Los 16 bits utilizados para identificar los números de puerto de origen y destino proporcionan un rango de puertos entre 0 y 65535.

La IANA ha dividido el rango de números en los siguientes tres grupos de puertos.

Nota: Algunos sistemas operativos de clientes pueden usar números de puerto registrados en lugar de números de puerto dinámicos para asignar puertos de origen.

La tabla muestra algunos números de puerto conocidos y sus aplicaciones asociadas.

Well-Known Port Numbers

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 usa TCP.

Busque en el sitio web de IANA el registro de puertos para ver la lista completa de números de puerto y aplicaciones asociadas.

14.4.4 El comando netstat

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 a continuación, ingrese el comando **netstat** Como se muestra a continuación, ingrese el comando para enumerar los protocolos en uso, la dirección local y los números de puerto, la dirección extranjera y los números de puerto y el estado de la conexión.

```
C:\> netstat
Active Connections
  Proto  Local Address          Foreign
Address            State
  TCP    192.168.1.124:3126     192.168.0.2:netbios-
ssn    ESTABLISHED
  TCP    192.168.1.124:3158     207.138.126.152:http       ES
TABLISHED
  TCP    192.168.1.124:3159     207.138.126.169:http       ES
TABLISHED
  TCP    192.168.1.124:3160     207.138.126.169:http       ES
TABLISHED
  TCP    192.168.1.124:3161     sc.msn.com:http            ES
TABLISHED
  TCP    192.168.1.124:3166     www.cisco.com:http         ES
TABLISHED
(output omitted)
C:\>
```

De manera predeterminada, el **netstat** comando intentará resolver las direcciones IP para nombres de dominio y números de puerto para aplicaciones conocidas. La **-n** opción se puede usar para mostrar las direcciones IP y los números de puerto en su forma numérica.

14.5 Proceso de comunicación TCP

Desplázate para empezar

14.5.1 Procesos del Servidor TCP

Ya conoce los fundamentos de TCP. Comprender el papel de los números de puerto le ayudará a comprender los detalles del proceso de comunicación TCP. En este tema, también aprenderá acerca de los procesos de enlace de tres vías TCP y terminación de sesión.

Cada proceso de aplicación que se ejecuta en el servidor para utilizar un número de puerto. El número de puerto es asignado automáticamente o configurado manualmente por un 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. Por ejemplo, un host que ejecuta una aplicación de servidor web y una aplicación de transferencia de archivos no puede tener ambos configurados para usar el mismo puerto, como el puerto TCP 80.

Una aplicación de servidor activa asignada a un puerto específico se considera abierta, lo que significa que la capa de transporte acepta y procesa los segmentos dirigidos a ese puerto. Toda solicitud entrante de un cliente direccionada al socket correcto es aceptada y los datos se envían a la aplicación del servidor. Pueden existir varios puertos abiertos simultáneamente en un servidor, uno para cada aplicación de servidor activa.

Haga clic en cada botón para obtener más información sobre los procesos del servidor TCP.
Clientes Envían Solicitudes TCP
Solicitar Puertos de Destino
Solicitar Puertos de Origen
Respuesta de Puertos de Destino
Respuesta de puertos de Origen

Clientes Envían Solicitudes TCP

El Cliente 1 está solicitando servicios web y el Cliente 2 está solicitando servicio de correo electrónico del mismo servidor.

Solicitud HTTP:
Puerto de Origen: 49152
Puerto de Destino: 80
Solicitud SMTP:
Puerto de Origen: 51152
Puerto de Destino: 25
Servidor
Cliente 1
Cliente 2
HTTP: Puerto 80
SMTP: Puerto 25
Solicitudes del cliente al servidor TCP

14.5.2 Establecimiento de Conexiones TCP

En algunas culturas, cuando dos personas se conocen, generalmente se saludan dándose la mano. Ambas partes entienden el acto de estrechar la mano como una señal de saludo amistoso. Las conexiones en la red son similares. En las conexiones TCP, el cliente host establece la conexión con el servidor mediante el proceso de enlace de tres vías.

Haga clic en cada botón para obtener más información sobre cada paso de establecimiento de conexión TCP.
Paso 1. SYN
Paso 2. ACK y SYN
Paso 3. ACK

Paso 1. SYN

El cliente de origen solicita una sesión de comunicación con el servidor.

PCA inicia un apretón de manos de tres vías mediante el envío de un segmento syn a PCB.
1AB

Enviar SYN

(SEQ=100 CTL=SYN)

SYN recibido

A envía solicitud de SYN a B

El protocolo de enlace de tres vías valida que el host de destino esté disponible para comunicarse. En este ejemplo, el host A ha validado que el host B está disponible.

14.5.3 Terminación de Sesión

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. El cliente o el servidor pueden iniciar la terminación.

En el ejemplo, los términos cliente y servidor se utilizan como referencia por simplicidad, pero dos hosts que tengan una sesión abierta pueden iniciar el proceso de finalización.

Haga clic en cada botón para obtener más información sobre los pasos de finalización de la sesión.
Paso 1. FIN
Paso 2. ACK
Paso 3. FIN
Paso 4. ACK

Paso 1. FIN

Cuando el cliente no tiene más datos para enviar en la transmisión, envía un segmento con el indicador FIN establecido.

PCA envía un segmento de alerta a PCB para finalizar la sesión cuando no hay más datos para enviar
1AB

Enviar FIN

FIN recibido

A envía la solicitud de FIN a B.

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

14.5.4 Análisis del enlace de tres vías de TCP

Los hosts mantienen el estado, rastrean cada segmento de datos dentro de una sesión e intercambian información sobre qué datos se reciben utilizando la información en el encabezado TCP. TCP es un protocolo full-duplex, donde cada conexión representa dos sesiones de comunicación unidireccionales. Para establecer la conexión, los hosts realizan un enlace de tres vías. Como se muestra en la figura, los bits de control en el encabezado TCP indican el progreso y el estado de la conexión.

Estas son las funciones del apretón de manos de tres vías:

- Establece que el dispositivo de destino está presente en la red.
- Verifica que el dispositivo de destino tenga un servicio activo y acepte solicitudes en el número de puerto de destino que el cliente de origen desea 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.

Campo de Bits de Control

muestra los campos de encabezado del segmento tcp con el campo de bits de control de 6 bits resaltado

20 bytes

Puerto de Origen (16)
Puerto de Destino (16)
Número de Reconocimiento (32)
Longitud del Encabezado (4)
Ventana (16)
Suma de comprobación (16)
Urgente (16)
Opciones (0 o 32 si las hay)
Datos de la Capa de Aplicación (el tamaño varía)
Número de Secuencia (32)
Reservado (6)
Bits de Control (6)

Los seis bits del campo de bits de control del encabezado del segmento TCP también se conocen como marcadores. Una bandera es un bit que está activado o desactivado.

Los seis indicadores de bits de control son los siguientes:

- **URG** - campo de puntero urgente significativo
- **ACK** - Indicador de acuse de recibo utilizado en el establecimiento de la conexión y la terminación de la sesión
- **PSH** - Función de empuje
- **RST** - Restablece la conexión cuando se produce un error o un tiempo de espera
- **SYN** - Sincroniza números de secuencia utilizados en el establecimiento de conexión
- **FIN** - No más datos del remitente y se utilizan en la terminación de la sesión

Busque en Internet para obtener más información sobre las banderas PSH y URG.

14.6 Confiabilidad y control de flujo

Desplázate para empezar

14.6.1 Fiabilidad de TCP - Entrega Garantizada y Ordenada

La razón por la que TCP es el mejor protocolo para algunas aplicaciones es porque, a diferencia de UDP, reenvía paquetes descartados y paquetes numerados para indicar su orden correcto antes de la entrega. TCP también puede ayudar a mantener el flujo de paquetes para que los dispositivos no se sobrecarguen. En este tema se tratan detalladamente estas características de TCP.

Puede haber momentos en que los segmentos TCP no llegan a su destino. Otras veces, los segmentos TCP podrían llegar fuera de servicio. 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 que se transmiten 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.

El ISN no comienza en uno, pero es efectivamente 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.

Los segmentos TCP se Reordenan en el Destino

muestra que aunque los segmentos pueden tomar diferentes rutas y llegar fuera de servicio en el destino, TCP tiene la capacidad de reordenar los segmentos

Segmento 1
Segmento 2
Segmento 3
Segmento 4
Segmento 5
Segmento 6
Segmento 1
Segmento 2
Segmento 6
Segmento 5
Segmento 4
Segmento 3
Segmento 1
Segmento 2
Segmento 3
Segmento 4
Segmento 5
Segmento 6

Al tomar diferentes rutas al destino, los segmentos llegan desordenados.

TCP reordena los segmentos en el orden original.

Los diferentes segmentos pueden tomar diferentes rutas.

Data

Data se divide en segmentos.

El proceso TCP receptor coloca los datos del segmento en un búfer de recepción. Los segmentos se colocan en el orden de secuencia adecuado y se pasan a la capa de aplicación cuando se vuelven a montar. 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.

14.6.2 Video - Confiabilidad de TCP: números de secuencia y acuses de recibo

Una de las funciones de TCP es garantizar que cada segmento llegue a su destino. Los servicios TCP en el host de destino reconocen los datos que ha recibido la aplicación de origen.

Haga clic en Reproducir en la figura para ver una lección acerca de los reconocimientos y los números de secuencia TCP.

14.6.3 Fiabilidad de TCP - Pérdida y Retransmisión de Datos

No importa cuán bien diseñada esté una red, ocasionalmente se produce la pérdida de datos. TCP proporciona métodos para administrar la pérdida de segmentos. Entre estos está un mecanismo para retransmitir segmentos para los datos sin reconocimiento.

El número de secuencia (SEQ) y el número de acuse de recibo (ACK) se utilizan juntos para confirmar la recepción de los bytes de datos contenidos en los segmentos transmitidos. El número SEQ identifica el primer byte de datos en el segmento que se transmite. TCP utiliza el número de ACK reenviado al origen para indicar el próximo byte que el receptor espera recibir. Esto se llama acuse de recibo de expectativa.

Antes de mejoras posteriores, TCP solo podía reconocer el siguiente byte esperado. Por ejemplo, en la figura, utilizando números de segmento para simplificar, el host A envía los segmentos del 1 al 10 al host B. Si llegan todos los segmentos excepto los segmentos 3 y 4, el host B respondería con acuse de recibo especificando que el siguiente segmento esperado es el segmento 3. El host A no tiene idea de si algún otro segmento llegó o no. Por lo tanto, el host A reenviaría los segmentos 3 a 10. Si todos los segmentos de reenvío llegan correctamente, los segmentos 5 a 10 serían duplicados. Esto puede provocar retrasos, congestión e ineficiencias.

muestra PCA enviando 10 segmentos a PCB, pero los segmentos 3 y 4 no llegan. Así que a partir del segmento 3, PCA vuelve a enviar los segmentos 3 a 10, aunque la PCB solo necesitaba los segmentos 3 y 4.

ABAB
Segmento 1
Segmento 2
Segmento 3
Segmento 5
Segmento 6
Segmento 7
Segmento 8
Segmento 9
Segmento 10
Segmento 4
Segmento 3
Segmento 4
Segmento 5
Segmento 6
Segmento 7
Segmento 8
Segmento 9
Segmento 10
ACK 3
ACK 11

Segmentos duplicados

Los sistemas operativos actualmente suelen emplear una característica TCP opcional llamada reconocimiento selectivo (SACK), negociada durante el protocolo de enlace de tres vías. Si ambos hosts admiten SACK, el receptor puede reconocer explícitamente qué segmentos (bytes) se recibieron, incluidos los segmentos discontinuos. Por lo tanto, el host emisor solo necesitaría retransmitir los datos faltantes. Por ejemplo, en la siguiente figura, utilizando de nuevo números de segmento para simplificar, el host A envía los segmentos 1 a 10 al host B. Si llegan todos los segmentos excepto los segmentos 3 y 4, el host B puede reconocer que ha recibido los segmentos 1 y 2 (ACK 3) y reconocer selectivamente los segmentos 5 a 10 (SACK 5-10). El host A solo necesitaría reenviar los segmentos 3 y 4.

muestran PCA enviando 10 segmentos a PCB, pero los segmentos 3 y 4 no llegan. Esta vez PCB envía un ack 3 y un saco 5-10 para que PCA sepa reenviar los segmentos faltantes 3 y 4 y continuar con el segmento 11

ABAB
Segmento 1
Segmento 2
Segmento 3
Segmento 5
Segmento 6
Segmento 7
Segmento 8
Segmento 9
Segmento 10
Segmento 4
Segmento 3
Segmento 4
Segmento 11
Segmento 12
ACK 3, SACO 5-10
AC 13

Nota: TCP normalmente envía ACK para cada otro paquete, pero otros factores más allá del alcance de este tema pueden alterar este comportamiento.

TCP utiliza temporizadores para saber cuánto tiempo debe esperar antes de reenviar un segmento. En la figura, reproduzca el vídeo y haga clic en el enlace para descargar el archivo PDF. El vídeo y el archivo PDF examinan la pérdida y la retransmisión de datos.

14.6.4 Video - Confiabilidad de TCP: Pérdida y retransmisión de datos

Haga clic en Reproducir en la figura para ver una lección acerca de la retransmisión TCP.

14.6.5 Control de Flujo de TCP - Tamaño de la Ventana y Reconocimientos

TCP también proporciona mecanismos para el control de flujo. El control de flujo es la cantidad de datos que el destino puede recibir y procesar de manera confiable. 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”.

En la figura, se muestra un ejemplo de tamaño de ventana y reconocimientos.

Ejemplo de tamaño de ventana de TCP

muestra el PCB que envía PCA un tamaño de ventana negociado de 10.000 bytes y un tamaño máximo de segmento de 1.460 bytes. PCA comienza a enviar segmentos comenzando con la secuencia número 1. Se puede enviar una confirmación de PCB sin esperar hasta que se alcance el tamaño de la ventana y el tamaño de la ventana se puede ajustar mediante PCA creando una ventana deslizante

AB

TMS = Tamaño Máximo de Segmento

Durante el protocolo de enlace de tres vías, tamaño de Ventana 10.000, TMS 1.460

Número de secuencia: 1.461

1.460 bytes
ACK 2.921
Tamaño de ventana 10.000
1.460 bytes
ACK 4.381
Tamaño de ventana 10.000

Ventana de envío 10.000
Número de secuencia: 1

1.460 bytes

Recibir reconocimiento
Ventana de envío 12.920
Número de secuencia: 2.921
Recibir reconocimiento
Ventana de envío 14.380

Recibir 1 - 1.460
Recibir 1461 - 2.920
Recibir 2921 - 4.380

El tamaño de la ventana determina la cantidad de bytes que se pueden enviar para recibir un reconocimiento. El número de reconocimiento es el número del siguiente byte esperado.

El tamaño de ventana es la cantidad de bytes que el dispositivo de destino de una sesión TCP puede aceptar y procesar al mismo tiempo. En este ejemplo, el tamaño de la ventana inicial de la PC B para la sesión TCP es de 10,000 bytes. A partir del primer byte, el byte1, el último byte que la PC A puede enviar sin recibir un reconocimiento es el byte 10000. Esto se conoce como la ventana de envío de la PC A. El tamaño de la ventana se incluye en cada segmento TCP para que el destino pueda modificar el tamaño de la ventana en cualquier momento dependiendo de la disponibilidad del búfer.

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 el número de bytes enviados al dispositivo de destino en función del tamaño de la ventana del 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 destino no esperará que se reciban todos los bytes de su tamaño de ventana antes de contestar con un acuse de recibo. A medida que se reciben y procesan los bytes, el destino envía reconocimientos para informar al origen que puede continuar enviando bytes adicionales.

Por ejemplo, es típico que la PC B no espere hasta que se hayan recibido los 10,000 bytes antes de enviar un acuse de recibo. Esto significa que la PC A puede ajustar su ventana de envío a medida que recibe confirmaciones de la PC B. Como se muestra en la figura, cuando la PC A recibe una confirmación con el número de confirmación 2,921, que es el siguiente byte esperado. La ventana de envío de PC A incrementará 2.920 bytes. Esto cambia la ventana de envío de 10.000 bytes a 12.920. Esto significa que la PC A puede ajustar su ventana de envío a medida que recibe confirmaciones de la PC B. Como se muestra en la figura, cuando la PC A recibe una confirmación con el número de confirmación 2,921, que es el siguiente byte esperado.

Un destino que envía confirmaciones a medida que procesa los bytes recibidos, y el ajuste continuo de la ventana de envío de origen, se conoce como ventanas deslizantes. En el ejemplo anterior, la ventana de envío del PC A aumenta o se desliza sobre otros 2.921 bytes de 10.000 a 12.920.

Si disminuye la disponibilidad de espacio de búfer del destino, puede reducir su tamaño de ventana para informar al origen que reduzca el número de bytes que debe enviar sin recibir un reconocimiento.

Nota: Hoy en día, los dispositivos utilizan el protocolo de ventanas deslizantes. El receptor generalmente envía un acuse de recibo después de cada dos segmentos que recibe. El número de segmentos recibidos antes de que se acuse recibo puede variar. La ventaja de las ventanas deslizantes es que permiten que el emisor transmita continuamente segmentos mientras el receptor está acusando recibo de los segmentos anteriores. Los detalles de las ventanas deslizantes exceden el ámbito de este curso.

14.6.6 Control de Flujo TCP - Tamaño Máximo de Segmento (MSS)

En la figura, la fuente está transmitiendo 1.460 bytes de datos dentro de cada segmento TCP. Normalmente, es el Tamaño Máximo de Segmento (MSS) que puede recibir el dispositivo de destino. El MSS forma parte del campo de opciones del encabezado TCP que especifica la mayor cantidad de datos, en bytes, que un dispositivo puede recibir en un único segmento TCP. El tamaño MSS no incluye el encabezado TCP. El MSS se incluye normalmente durante el apretón de manos de tres vías.

muestra el mismo diagrama que antes, pero el énfasis está en el MSS de tamaño máximo de segmento de 1460
AB

TMS = Tamaño Máximo de Segmento

Durante el protocolo de enlace de tres vías, tamaño de ventana 10.000, TMS 1.460

Número de secuencia: 1.461

1.460 bytes
ACK 2.921
Tamaño de ventana 10.000
1.460 bytes
ACK 4.381
Tamaño de ventana 10.000

Ventana de envío 10.000
Número de secuencia: 1

1.460 bytes

Recibir reconocimiento
Ventana de envío 12.920
Número de secuencia: 2.921
Recibir reconocimiento
Ventana de envío 14.380

Recibir 1 - 1.460
Recibir 1461 - 2.920
Recibir 2921 - 4.380

Un MSS común es de 1.460 bytes cuando se usa IPv4. Un host determina el valor de su campo de MSS restando los encabezados IP y TCP de unidad Máxima de transmisión (MTU) de Ethernet. En una interfaz de Ethernet, el MTU predeterminado es de 1500 bytes. Restando el encabezado IPv4 de 20 bytes y el encabezado TCP de 20 bytes, el tamaño predeterminado de MSS será 1460 bytes, como se muestra en la figura.

muestra un diagrama de un marco Ethernet completo del cual la MTU es 1500 bytes, con 20 bytes siendo el encabezado IP, y 20 bytes siendo el encabezado TCP, esto deja 1460 bytes que es el tamaño máximo del segmento TCP MSS

Ethernet
IPv4
TCP
Carga útil
FCS
MTU de Ethernet
MTU de IP
20 bytes
20 bytes
1460 bytes
1500 bytes
TCP MSS

14.6.7 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 segmentos TCP no llegan a su destino, se dejan sin confirmar. 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. Como se ilustra en la figura, PC A detecta que hay congestión y, por lo tanto, reduce el número de bytes que envía antes de recibir un acuse de recibo de PC B.

Control de Congestión de TCP

muestra PCA enviando segmentos a PCB donde los segmentos perdidos y la retransmisión pueden causar congestión

AB
Segmento TCP 1
Segmento TCP 2
Segmento TCP 3
Segmento TCP 4
Segmento de confirmación 2
Segmento TCP 2

Segmento TCP 3

No recibo la confirmación que espero de la PC B. Entonces reduzco la cantidad de bytes que envío antes de una confirmación.

Los números de acuse de recibo corresponden al siguiente byte esperado y no a un segmento. Los números de segmentos utilizados se simplifican con fines ilustrativos.

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.

14.7 Comunicación UDP

Desplázate para empezar

14.7.1 Comparación de baja sobrecarga y confiabilidad de UDP

Como se explicó anteriormente, UDP es perfecto para comunicaciones que necesitan ser rápidas, como VoIP. Este tema explica en detalle por qué UDP es perfecto para algunos tipos de transmisiones. Como se muestra en la figura, UDP no establece una conexión. UDP suministra transporte de datos con baja sobrecarga debido a que posee un encabezado de datagrama pequeño sin tráfico de administración de red.

muestra un remitente de host que necesita enviar datos de voz y vídeo que se envían con UDP que no requiere una conexión negociada previa

Red
Emisor
Receptor
Datos

UDP no establece ninguna conexión antes de enviar datos.

14.7.2 Reensamblaje de datagramas de UDP

Tal como los segmentos con TCP, 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.

UDP: sin conexión y poco confiable

muestra los datagramas UDP enviados en orden pero que llegan fuera de servicio debido a la posibilidad de diferentes rutas para llegar al destino

Datagrama 1
Datagrama 2
Datagrama 3
Datagrama 4
Datagrama 5
Datagrama 6
Datagrama 1
Datagrama 2
Datagrama 6
Datagrama 5
Datagrama 4

Al tomar diferentes rutas para llegar al destino, los datagramas llegan desordenados.
Los datagramas desordenados no se vuelven a ordenar.
Los diferentes datagramas pueden tomar diferentes rutas.
Los datagramas perdidos no se reenvían.

Data

Datos Los datos se dividen en datagramas.

14.7.3 Procesos y 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.

Servidor UDP a la escucha de solicitudes

muestra que una aplicación de servidor RADIUS utiliza UDP para escuchar solicitudes en el puerto 53

Servidor
Cliente 1
Cliente 2

Aplicaciones del servidor Las solicitudes
DNS del cliente se recibirán en el puerto 53.

Las solicitudes de RADIUS de clientes se recibirán en el puerto 1812.

Solicitud de DNS
Solicitud de RADIUS

Nota: El servidor del Servicio de usuario de acceso telefónico de autenticación remota (RADIUS) que se muestra en la figura proporciona servicios de autenticación, autorización y contabilidad para administrar el acceso de los usuarios. El funcionamiento de RADIUS está más allá del alcance de este curso.

14.7.4 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.

Después de que un cliente ha seleccionado los puertos de origen y destino, se utiliza el mismo par de puertos en el encabezado de todos los datagramas 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.

Haga clic en cada botón para obtener una ilustración de dos hosts que solicitan servicios del servidor de autenticación DNS y RADIUS.
Clientes que envían solicitudes UDP
Puertos de destino de solicitud UDP
Puertos de origen de solicitud UDP
Destino de respuesta UDP
Puertos de origen de respuesta UDP

Clientes Mandando Solicitudes UDP

El cliente 1 está enviando una solicitud DNS utilizando el conocido puerto 53, mientras que el cliente 2 solicita servicios de autenticación RADIUS mediante el puerto registrado 1812.

Dos clientes de PC diferentes necesitan realizar una solicitud a un servidor DNS

Solicitud DNS del cliente 1:
Puerto de origen 49152
Puerto de destino 53
Solicitud de autenticación de usuario RADIUS del cliente 2:
Puerto de origen 51152
Puerto de destino 1812

14.8 Práctica del Módulo y Cuestionario

Desplázate para empezar

14.8.1 Packet Tracer - Comunicaciones de TCP y UDP

En esta actividad, explorará la funcionalidad de los protocolos TCP y UDP, la multiplexación y la función de los números de puerto para determinar qué aplicación local solicitó los datos o los envía.

descriptionComunicaciones de TCP y UDPfile_downloadComunicaciones de TCP y UDP

14.8.2 ¿Qué aprenderé en este módulo?

Transportation of Data

La capa de transporte es el enlace entre la capa de aplicación y las capas inferiores que son responsables de la transmisión a través de la red. La capa de transporte es responsable de las comunicaciones lógicas entre aplicaciones que se ejecutan en diferentes hosts. La capa de transporte incluye los protocolos TCP y UDP. Los protocolos de capa de transporte especifican cómo transferir mensajes entre hosts y es responsable de administrar los requisitos de fiabilidad de una conversación. La capa de transporte es responsable del seguimiento de conversaciones (sesiones), la segmentación de datos y el reensamblaje de segmentos, la adición de información de encabezado, la identificación de aplicaciones y la multiplexación de conversaciones. TCP is stateful, reliable, acknowledges data, resends lost data, and delivers data in sequenced order. Utilice TCP para el correo electrónico y la web. UDP no tiene estado, es rápido, tiene una sobrecarga baja, no requiere acuses de recibo, no reenvía los datos perdidos y entrega los datos en el orden en que llegan. Use UDP para VoIP y DNS.

Revisar TCP

TCP establece sesiones, asegura confiabilidad, proporciona entrega del mismo pedido y admite control de flujo. Un segmento TCP agrega 20 bytes de sobrecarga como información de encabezado al encapsular los datos de capa de aplicación. Los campos de encabezado TCP son los puertos de origen y destino, número de secuencia, número de reconocimiento, longitud del encabezado, reservado, bits de control, tamaño de ventana, suma de verificación y urgente. Las aplicaciones que usan TCP son HTTP, FTP, SMTP y Telnet.

Revisar UDP

UDP reconstruye los datos en el orden en que se reciben, los segmentos perdidos no se vuelven a enviar, no se establece la sesión y UPD no informa al remitente de la disponibilidad de recursos. Los campos de encabezado UDP son puertos de origen y destino, longitud y suma de verificación. Las aplicaciones que utilizan UDP son DHCP, DNS, SNMP, TFTP, VoIP y videoconferencias.

Número de Puertos

Los protocolos de capa de transporte TCP y UDP utilizan números de puerto para administrar múltiples conversaciones simultáneas. Esta es la razón por la que los campos de encabezado TCP y UDP identifican un número de puerto de aplicación de origen y destino. 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. Hay un rango de números de puerto entre 0 y 65535. Esta gama se divide en grupos: Puertos conocidos, Puertos Registrados, Puertos Privados y/o Dinámicos. Hay algunos números de puerto conocidos que están reservados para aplicaciones comunes como FTP, SSH, DNS, HTTP y otros. 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.

Proceso de Comunicación TCP

Cada proceso de aplicación que se ejecuta en el servidor para utilizar un número de puerto. El número de puerto es asignado automáticamente o configurado manualmente por un administrador del sistema. Los procesos del servidor TCP son los siguientes: clientes que envían solicitudes TCP, solicitan puertos de destino, solicitan puertos de origen, responden a solicitudes de puerto de destino y puerto de origen. Para terminar una conversación simple admitida por TCP, se requieren cuatro intercambios para finalizar ambas sesiones. El cliente o el servidor pueden iniciar la terminación. El protocolo de enlace de tres vías establece que el dispositivo de destino está presente en la red, verifica que el dispositivo de destino tiene un servicio activo y acepta solicitudes en el número de puerto de destino que el cliente iniciador pretende utilizar, e informa al dispositivo de destino que el cliente de origen tiene la intención de establecer una sesión de comunicación sobre ese número de puerto. Los seis indicadores de bits de control son: URG, ACK, PSH, RST, SYN y FIN.

Fiabilidad y Control de Flujo

Para que el receptor comprenda el mensaje original, los datos en estos segmentos se vuelven a ensamblar en el orden original. Se asignan números de secuencia en el encabezado de cada paquete. No importa cuán bien diseñada esté una red, ocasionalmente se produce la pérdida de datos. TCP proporciona formas de administrar pérdidas de segmentos. Hay un mecanismo para retransmitir segmentos para datos no reconocidos. Los sistemas operativos host actualmente suelen emplear una característica TCP opcional llamada reconocimiento selectivo (SACK), negociada durante el protocolo de enlace de tres

vías. Si ambos hosts admiten SACK, el receptor puede reconocer explícitamente qué segmentos (bytes) se recibieron, incluidos los segmentos discontinuos. Por lo tanto, el host emisor solo necesitaría retransmitir los datos faltantes. 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 lograr esto, el encabezado TCP incluye un campo de 16 bits llamado “tamaño de la ventana”. El proceso en el que el destino envía reconocimientos a medida que procesa los bytes recibidos y el ajuste continuo de la ventana de envío del origen se conoce como ventanas deslizantes. El origen está transmitiendo 1460 bytes de datos dentro de cada segmento TCP. Este es el MSS típico que puede recibir un dispositivo de destino. Para evitar y controlar la congestión, TCP emplea varios mecanismos de manejo de congestión. Es la fuente la que está reduciendo el número de bytes no reconocidos que envía y no el tamaño de ventana determinado por el destino.

Comunicación UDP

UDP es un protocolo simple que proporciona las funciones básicas de la capa de transporte. Cuando los datagramas UDP se envían a un destino, a menudo toman caminos diferentes y llegan en el orden incorrecto. UDP no realiza un seguimiento de los números de secuencia de la manera en que lo hace TCP. UDP no puede reordenar los datagramas en el orden de la transmisión. 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. A las aplicaciones de servidor basadas en UDP se les asignan números de puerto conocidos o registrados. 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. 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. Después de que un cliente ha seleccionado los puertos de origen y destino, se utiliza el mismo par de puertos en el encabezado de todos los datagramas utilizados 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.