Transporte de datos.

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. 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.

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” (figura 1). 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 (figura 2). 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 (figura 3). 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.

Multiplexión de conversaciones.

El envío de algunos tipos de datos (por ejemplo, una transmisión de vídeo) a través de una red, como un flujo completo de comunicación, puede consumir todo el ancho de banda disponible. Esto impedirá que se produzcan otras comunicaciones al mismo tiempo. También podría dificultar la recuperación de errores y la retransmisión de datos dañados.

En la figura, se muestra que la segmentación de los datos en partes más pequeñas permite que se entrelacen (multiplexen) varias comunicaciones de distintos usuarios en la misma red.

Para identificar cada segmento de datos, la capa de transporte agrega un encabezado que contiene datos binarios organizados en varios campos. Los valores de estos campos permiten que los distintos protocolos de la capa de transporte lleven a cabo variadas funciones de administración de la comunicación de datos.

Confiabilidad de la capa de transporte.

La capa de transporte también es responsable de administrar los requisitos de confiabilidad de las conversaciones. Las diferentes aplicaciones tienen diferentes requisitos de confiabilidad 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 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), como se muestra en la figura. IP utiliza estos protocolos de transporte para habilitar la comunicación y la transferencia de datos entre los hosts.

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.

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

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.

El proceso de UDP es similar al envío por correo de una carta simple sin registrar. 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 seguimiento de la carta ni de informar al emisor si esta no llega a destino.

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

Para algunas aplicaciones, los segmentos deben llegar en una secuencia muy específica para que se puedan procesar correctamente. Con otras aplicaciones, los datos se consideran útiles una vez que todos se reciben en forma completa. En ambos casos, se utiliza TCP como protocolo de transporte. Los desarrolladores de aplicaciones deben elegir qué tipo de protocolo de transporte es adecuado según los requisitos de las aplicaciones.

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 pérdida de datos puede dañar una comunicación y dejarla incompleta o ilegible. Estas aplicaciones están diseñadas para utilizar TCP.

En otros casos, una aplicación puede tolerar cierta pérdida de datos durante la transmisión a través de la red, pero no se admiten retrasos en la transmisión. UDP es la mejor opción para estas aplicaciones, ya que se requiere menos sobrecarga de red. Con aplicaciones como la transmisión de audio y vídeo en vivo o de voz sobre IP (VoIP), es preferible utilizar UDP. Los reconocimientos y la retransmisión reducirían la velocidad de entrega.

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.

Nota: Las aplicaciones que transmiten audio y vídeo almacenado utilizan 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. Una vez que todos los segmentos estén en orden y se restaure un nivel mínimo de ancho de banda, la sesión TCP se reanuda y la película comienza a reproducirse.

Descripción general de TCP y UDP. 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.

Para obtener más información sobre TCP, lea RFC.

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.

Características de UCP.

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.

En la figura, se describen las características de UDP.

Para obtener más información sobre UCP, lea RFC.

Encabezado UDP.

UDP es un protocolo sin información estado, lo cual significa que ni el cliente ni el servidor están obligados a hacer un seguimiento del 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 vídeo en vivo y voz a través de la red es que los datos fluyan rápidamente. Las aplicaciones de vídeo y de voz

en vivo pueden tolerar cierta pérdida de datos con un efecto mínimo o imperceptible, y se adaptan perfectamente a 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.

Conversaciones múltiples e independientes.

La capa de transporte debe poder separar y administrar varias comunicaciones con diferentes necesidades de requisitos de transporte. Los usuarios tienen la expectativa de poder recibir y enviar correo electrónico y mensajes instantáneos, explorar sitios web y realizar una llamada telefónica de VoIP de manera simultánea. Cada una de estas aplicaciones envía y recibe datos a través de la red al mismo tiempo, a pesar de los diferentes requisitos de confiabilidad. Además, los datos de la llamada telefónica no están dirigidos al navegador web, y el texto de un mensaje instantáneo no aparece en un correo electrónico.

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.

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. Resulta habitual para un dispositivo enviar varias solicitudes de servicio HTTP a un servidor web al mismo tiempo. 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. Un servidor puede ofrecer más de un servicio de manera simultánea, por ejemplo, servicios web en el puerto 80 al mismo tiempo que ofrece el establecimiento de una conexión FTP en el puerto 21.

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, 192.168.1.7:80

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.

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:

- **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

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.

dinámicos para asignar los puertos de origen.

Haga clic aquí para ver la lista completa de los números de puerto y las aplicaciones asociadas en el sitio web de la IANA.

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 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.

Proceso de comunicación TCP.

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.

Por ejemplo, un host que ejecuta una aplicación de servidor web y una de transferencia de archivos no puede configurar ambas para utilizar el mismo puerto (por ejemplo, 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.

Consulte las figuras 1 a 5 para ver la asignación típica de puertos de origen y de destino en las operaciones TCP de cliente y servidor.

Establecimiento de conexiones TCP.

En algunas culturas, cuando dos personas se conocen, generalmente se saludan dándose la mano. Ambas culturas entienden el acto de darse la mano como señal de un saludo amigable. Las conexiones en la red son similares. En las conexiones TCP, el cliente del host establece la conexión con el servidor.

Una conexión TCP se establece en tres pasos:

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.

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,

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

pero el proceso de finalización puede ser iniciado por dos hosts cualesquiera que tengan una sesión abierta:

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.

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. Haga clic en el campo de bits de control en la figura para ver los seis marcadores. Ya hemos hablado de SYN, ACK y FIN. El marcador RST se utiliza para restablecer una conexión cuando ocurre un error o se agota el tiempo de espera. Haga clic aquí para obtener más información sobre los marcadores PSH y URG.

- **URG** - importante campo de puntero urgente
- **ACK** - importante campo de confirmación
- **PSH** - función de empujar
- **RST** - restablecer la conexión
- **SYN** - sincronizar los números de secuencia
- **FIN** - no hay más datos del emisor

- **TCP y UDP.**

Confiabilidad y control de flujo.

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.

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”.

En la figura, se muestra un ejemplo de tamaño de ventana y reconocimientos. 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 para una sesión TCP de la PC B es 10 000 bytes. A partir del primer byte, el byte 1, el último byte que la PC A puede enviar sin recibir un reconocimiento es el byte 10 000. Esto se conoce como la “ventana de envío” de la PC A. El tamaño de ventana se incluye en cada segmento TCP para que el destino pueda modificar el tamaño de ventana en cualquier momento, según la disponibilidad del búfer.

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 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 lo general, la PC B no esperará hasta que los 10 000 bytes se hayan recibido antes de enviar un reconocimiento. Esto significa que la PC A puede ajustar su ventana de envío a medida que recibe reconocimientos de la PC B. Como se muestra en la figura, cuando la PC A recibe un reconocimiento con el número 2921, la ventana de envío de PC A se incrementará otros 10 000 bytes (el tamaño de la ventana actual de la PC B) a 12 920. La PC A puede ahora seguir enviando hasta otros 10 000 bytes a la PC B, siempre y cuando no supere la nueva ventana de envío establecida en 12 920.

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.

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: Por lo general, los dispositivos utilizan el protocolo de ventanas deslizantes. Con las ventanas deslizantes, el receptor no espera que se alcance la cantidad de bytes en el tamaño de ventana antes de enviar un acuse de recibo. 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. 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.

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.

Comunicación UDP.

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.

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.

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.

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 auditoría para administrar el acceso de usuario. El funcionamiento de RADIUS excede el ámbito de este curso.

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.

Haga clic en las figuras 1 a 5 para ver los detalles de los procesos de cliente UDP.

TCP o UDP.

Aplicaciones que utilizan TCP.

TCP es un excelente ejemplo de cómo las diferentes capas de la suite de protocolos TCP/IP tienen funciones específicas. TCP se encarga de todas las tareas asociadas con la división del flujo de datos en segmentos, lo que proporciona confiabilidad, control del flujo de datos y reordenamiento de 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.

Aplicaciones que utilizan UDP.

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

- **Aplicaciones multimedia y vídeo en vivo:** pueden tolerar cierta pérdida de datos, pero requieren retrasos cortos o que no haya retrasos. Los ejemplos incluyen VoIP y la transmisión de vídeo en vivo.
- **Aplicaciones con solicitudes y respuestas simples:** aplicaciones con transacciones simples en las que un host envía una solicitud y existe la posibilidad de que reciba una respuesta o no. Por ejemplo, DNS y DHCP.
- **Aplicaciones que manejan la confiabilidad por su cuenta:** comunicaciones unidireccionales que no requieran control de flujo, detección de errores, reconocimientos ni recuperación de errores o que puedan ser manejados por la aplicación. 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 tiene 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.