

Redes



Manual

Vol. 3 (Capa 4 Modelo OSI)



ALHUBO

Alejandro Huerta Bolaños

Primera Edición

2023

Índice

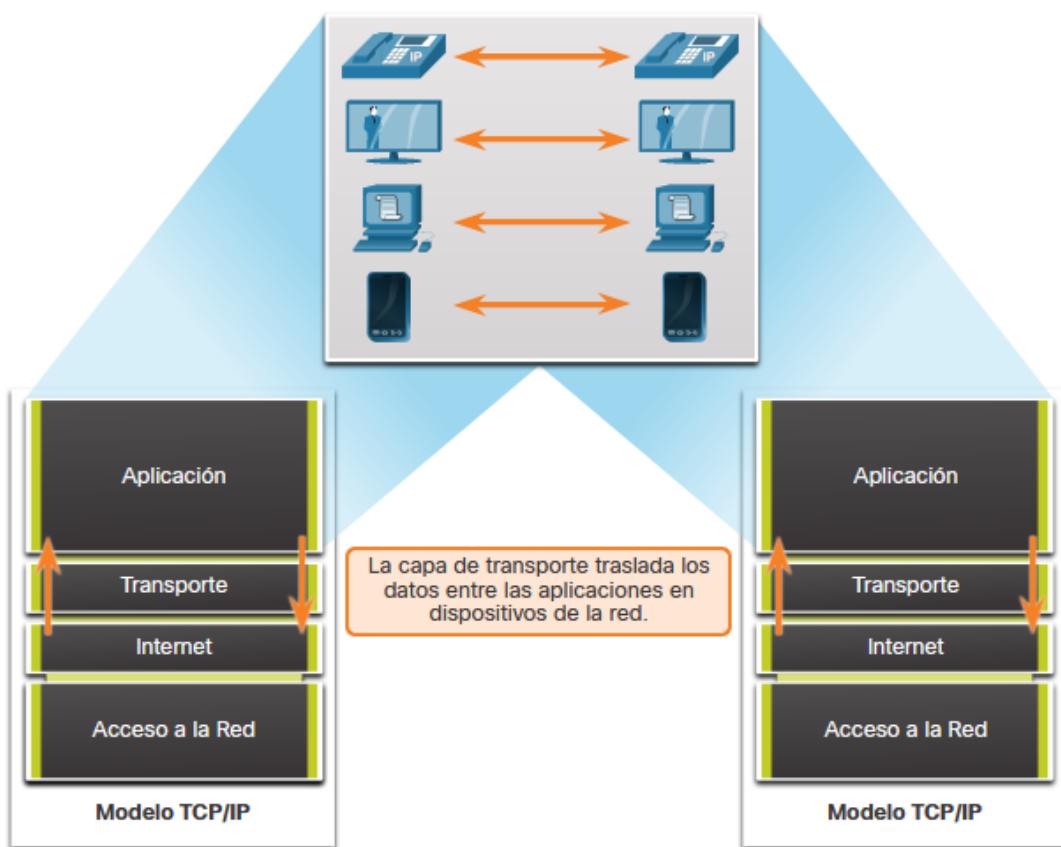
Índice	2
Función de la Capa de Transporte	4
Responsabilidades de la Capa de Transporte	4
Seguimiento de conversaciones individuales	5
Segmentación de Datos y Rearmado de Segmentos	5
Aregar Información de Encabezado	6
Identificación de las Aplicaciones	7
Multiplexión de Conversaciones	8
Protocolos de Capa de Transporte	9
Protocolo de Control de Transmisión (TCP)	10
Protocolo de Datagramas de Usuario (UDP)	17
Protocolo de la capa de transporte correcto para la aplicación adecuada	19
Características de TCP	21
Encabezado TCP	22
Campos de Encabezado TCP	23
Aplicaciones que utilizan TCP	24
Características de UDP	24
Encabezado UDP	25
Campos de Encabezado UDP	26
Aplicaciones que utilizan UDP	26
Números de puerto	27
Comunicaciones Múltiples Separadas	27
Pares de Sockets	28
Grupos de números de puerto	30
El comando netstat	32
Proceso de comunicación TCP	33
Procesos del Servidor TCP	33
Clientes Envían Solicitudes TCP	33
Solicitud de Puertos de Destino	34
Establecimiento de Conexiones TCP	36
Paso 1. SYN	36
Paso 2. ACK y SYN	37
Paso 3. ACK	38
Terminación de Sesión	39
Paso 1. FIN	39
Paso 2. ACK	40
Paso 3. FIN	40
Paso 4. ACK	41
Análisis del enlace de tres vías de TCP	42
Campo de Bits de Control	43
Confiabilidad y control de flujo	44

Fiabilidad de TCP: Entrega Garantizada y Ordenada	44
Fiabilidad de TCP: Pérdida y Retransmisión de Datos	46
Control de Flujo de TCP: Tamaño de la Ventana y Reconocimientos	57
Ejemplo de tamaño de ventana de TCP	58
Control de Flujo TCP - Tamaño Máximo de Segmento (MSS Maximum Segment Size)	60
Control de flujo de TCP: Prevención de Congestiones	61
Control de Congestión de TCP	62
Comunicación UDP	62
Comparación de baja sobrecarga y confiabilidad de UDP	62
Reensamblaje de datagramas de UDP	63
UDP: sin conexión y poco confiable	64
Procesos y solicitudes del servidor UDP	64
Servidor UDP a la escucha de solicitudes	65
Procesos de cliente UDP	65
Clientes Mandando Solicitudes UDP	65
Solicitud UDP de Puertos de Origen	66
UDP Solicitud de Puertos de Origen	66
Destino de respuesta UDP	67
UDP Respuesta de Puertos de Origen	68

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.



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 Transmission Control Protocol)
- Protocolo de Datagramas de Usuario (UDP User Datagram Protocol)

Responsabilidades de la Capa de Transporte

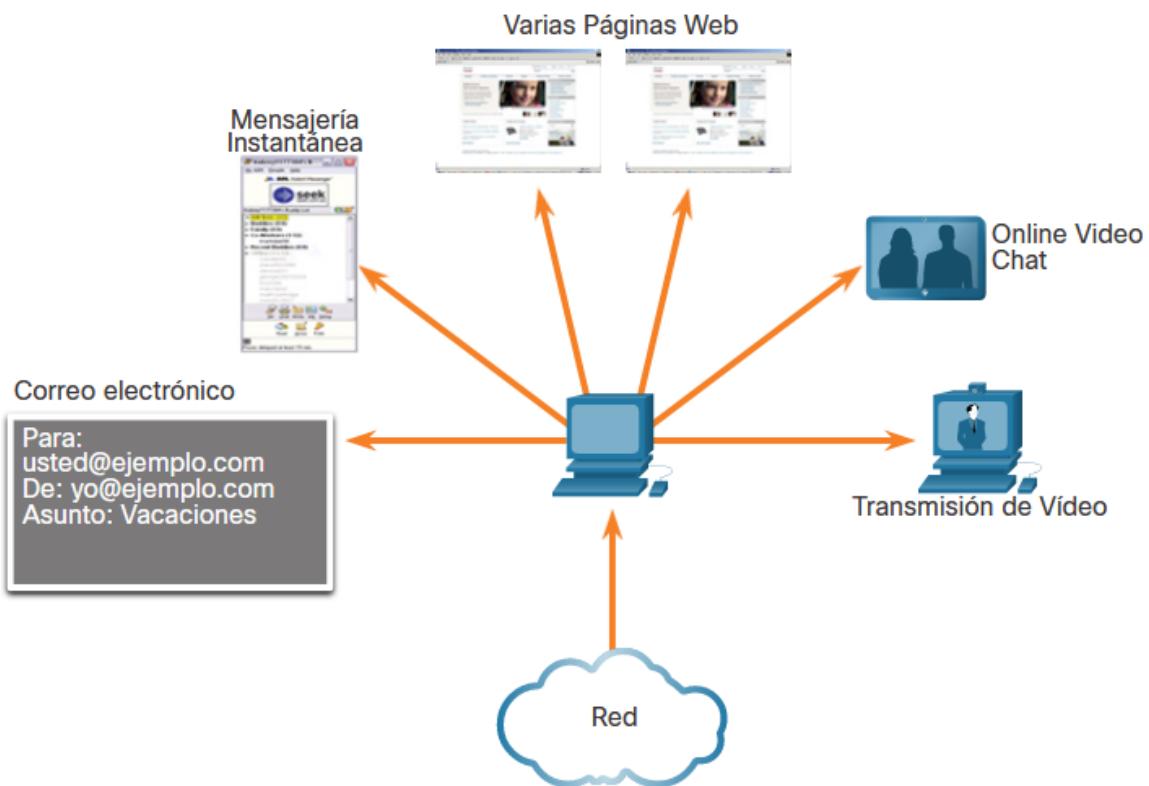
La capa de transporte tiene muchas responsabilidades.

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.

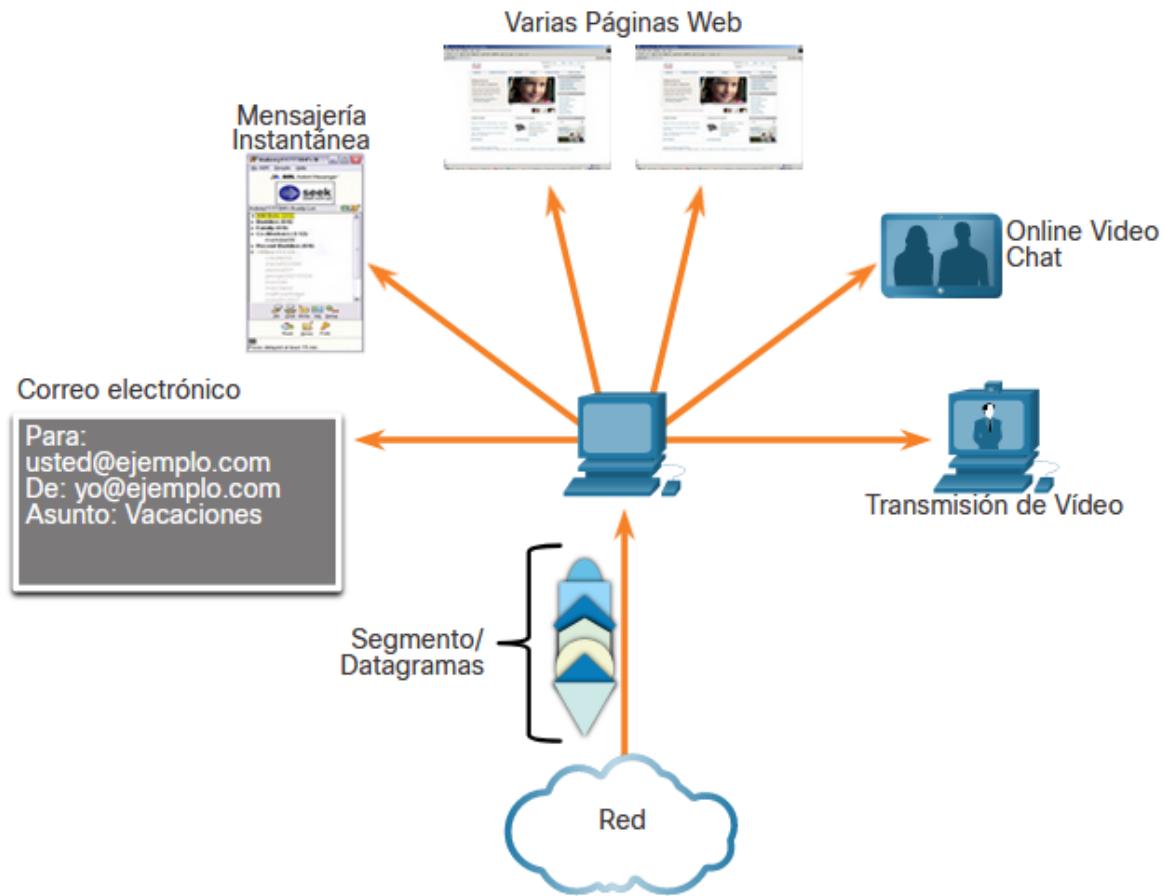


Segmentación de Datos y Rearmado de Segmentos

Es responsabilidad de la capa de transporte dividir los datos de la aplicación en bloques de tamaño adecuado. Dependiendo del protocolo de capa de transporte utilizado, los bloques de capa de transporte se denominan segmentos o

datagramas. La figura ilustra la capa de transporte utilizando diferentes bloques para cada conversación.

La capa de transporte divide los datos en bloques más pequeños (es decir, segmentos o datagramas) que son más fáciles de administrar y transportar.

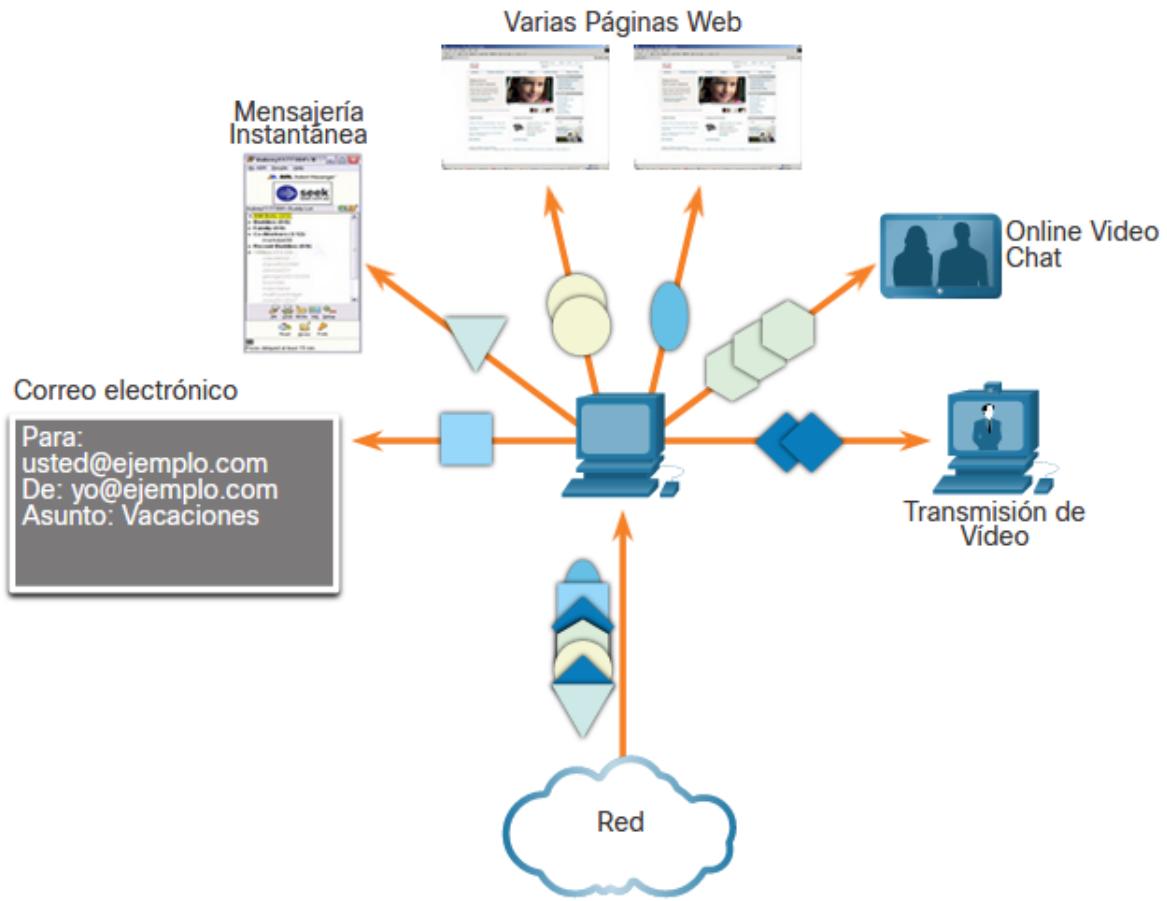


Agregar Información de Encabezado

El protocolo de capa de transporte también agrega información de encabezado que contiene datos binarios organizados en varios campos a cada bloque de datos. 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.

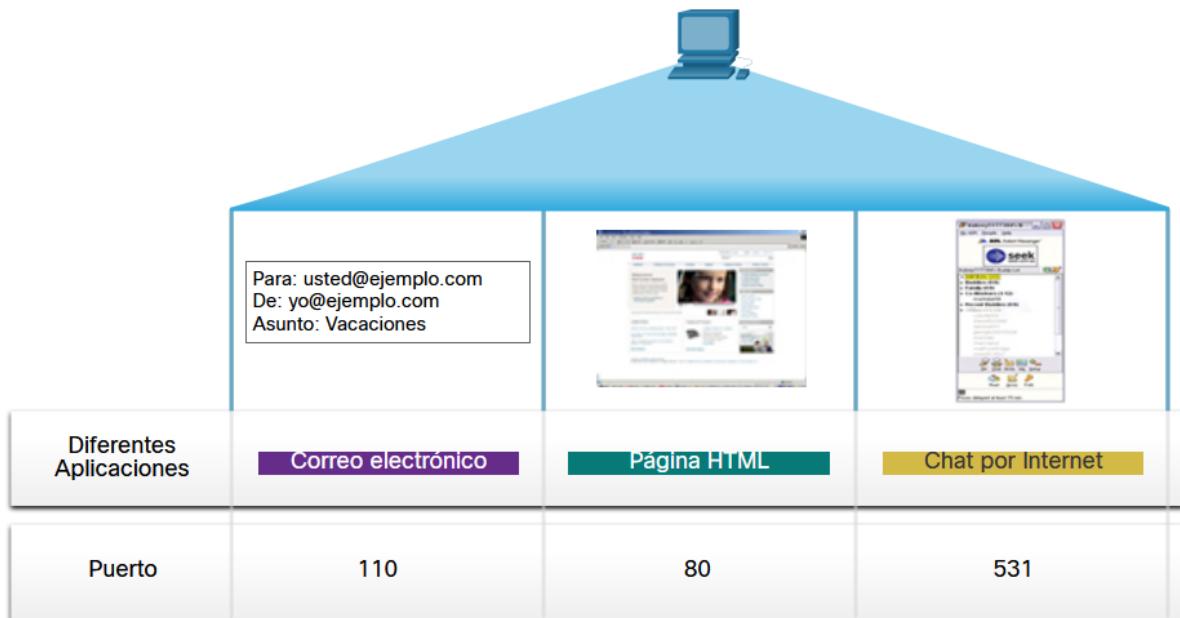
Por ejemplo, el host receptor utiliza la información de encabezado para volver a ensamblar los bloques de datos en un flujo de datos completo para el programa de capa de aplicación de recepción.

La capa de transporte garantiza que incluso con múltiples aplicaciones que se ejecutan en un dispositivo, todas las aplicaciones reciben los datos correctos.



Identificación de las Aplicaciones

La capa de transporte debe poder separar y administrar varias comunicaciones con diferentes necesidades de requisitos de transporte. Para pasar flujos de datos a las aplicaciones adecuadas, la capa de transporte identifica la aplicación de destino utilizando un identificador llamado número de puerto. Como se ilustra en la figura, a cada proceso de software que necesita acceder a la red se le asigna un número de puerto único para ese host.

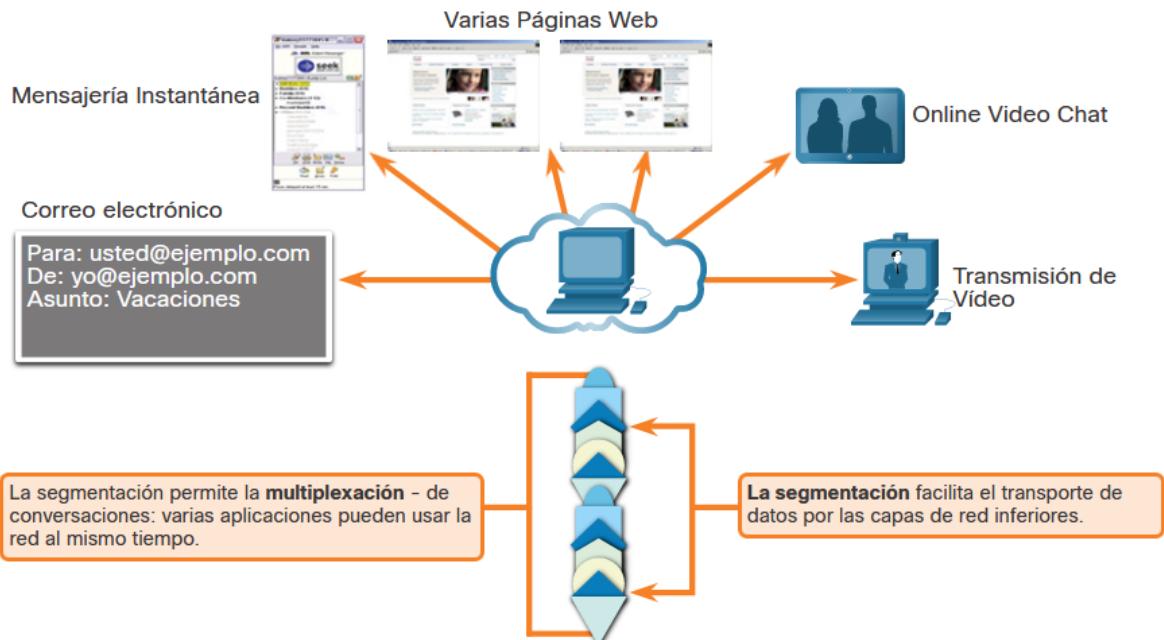


Multiplexión de Conversaciones

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

Como se muestra en la figura, la capa de transporte utiliza segmentación y multiplexación para permitir que diferentes conversaciones de comunicación se intercalen en la misma red.

La verificación de errores se puede realizar en los datos del segmento, para determinar si el segmento se modificó durante la transmisión.

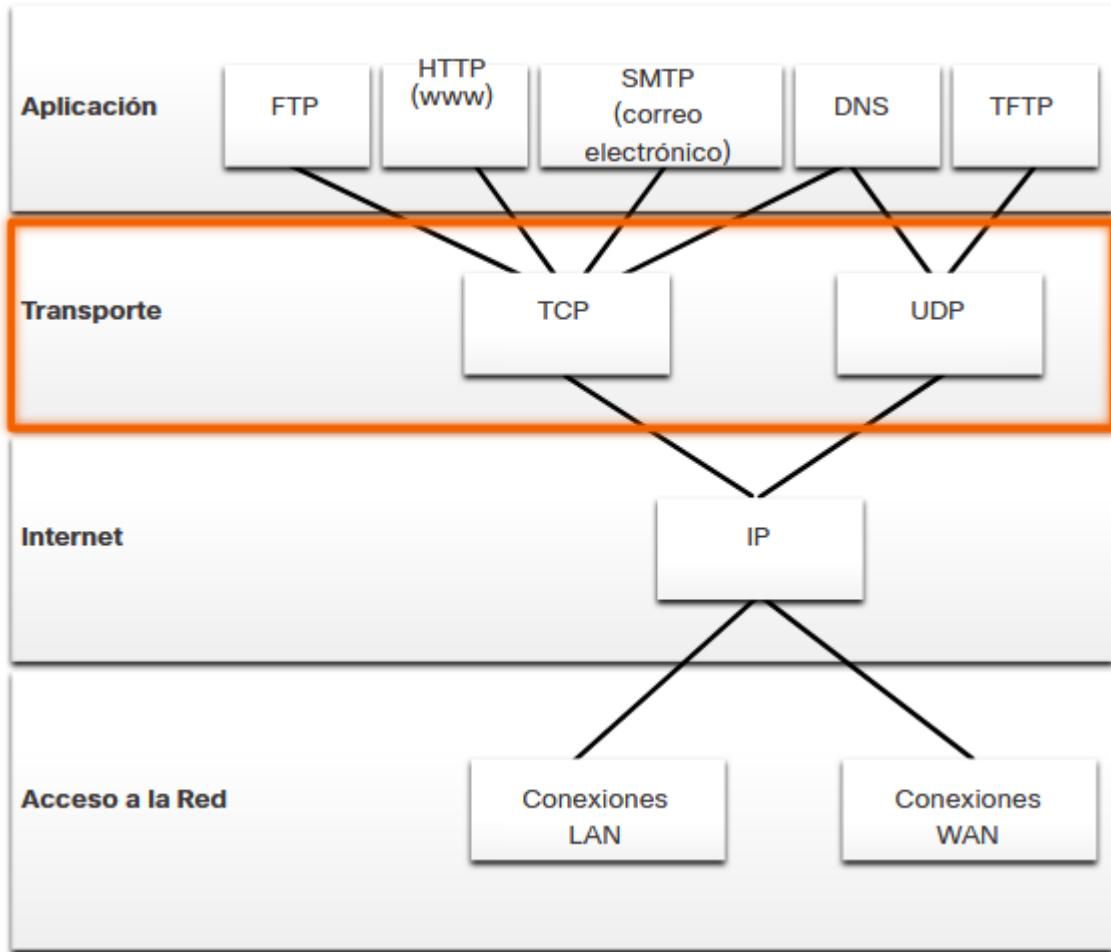


Protocolos de Capa de Transporte

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.



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.

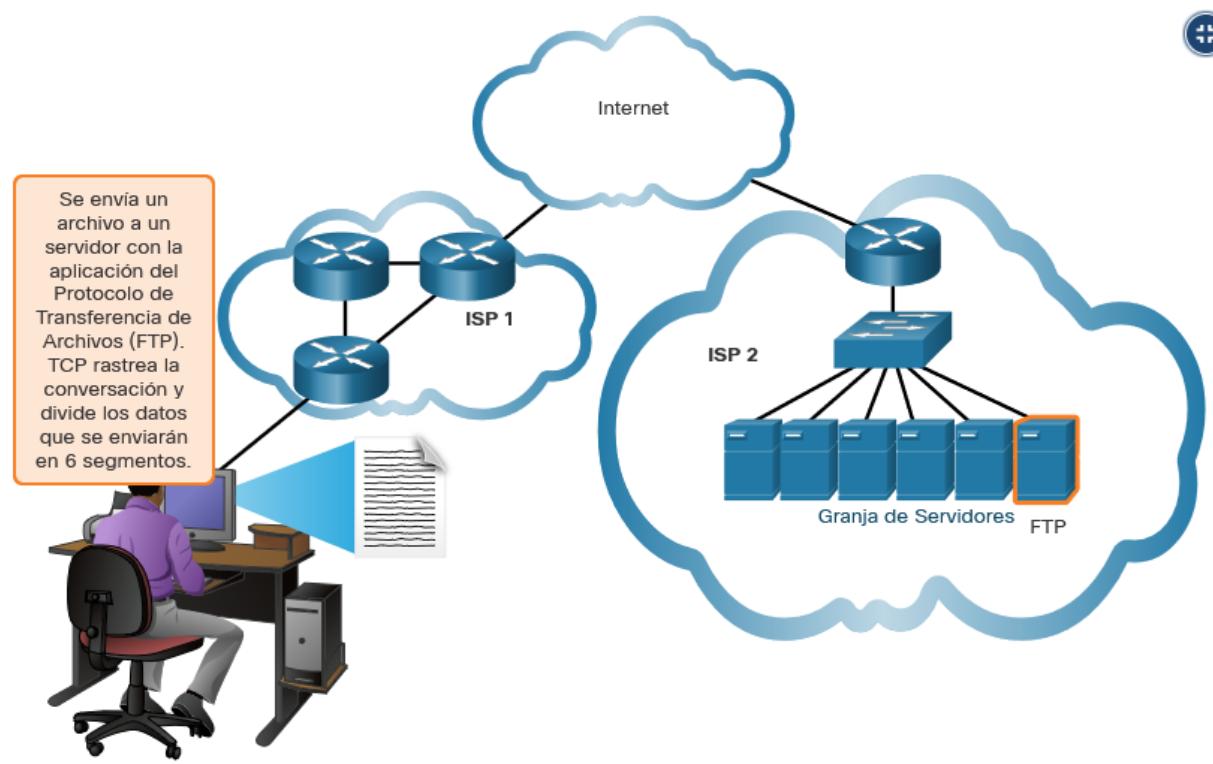
Nota: TCP divide los datos en segmentos.

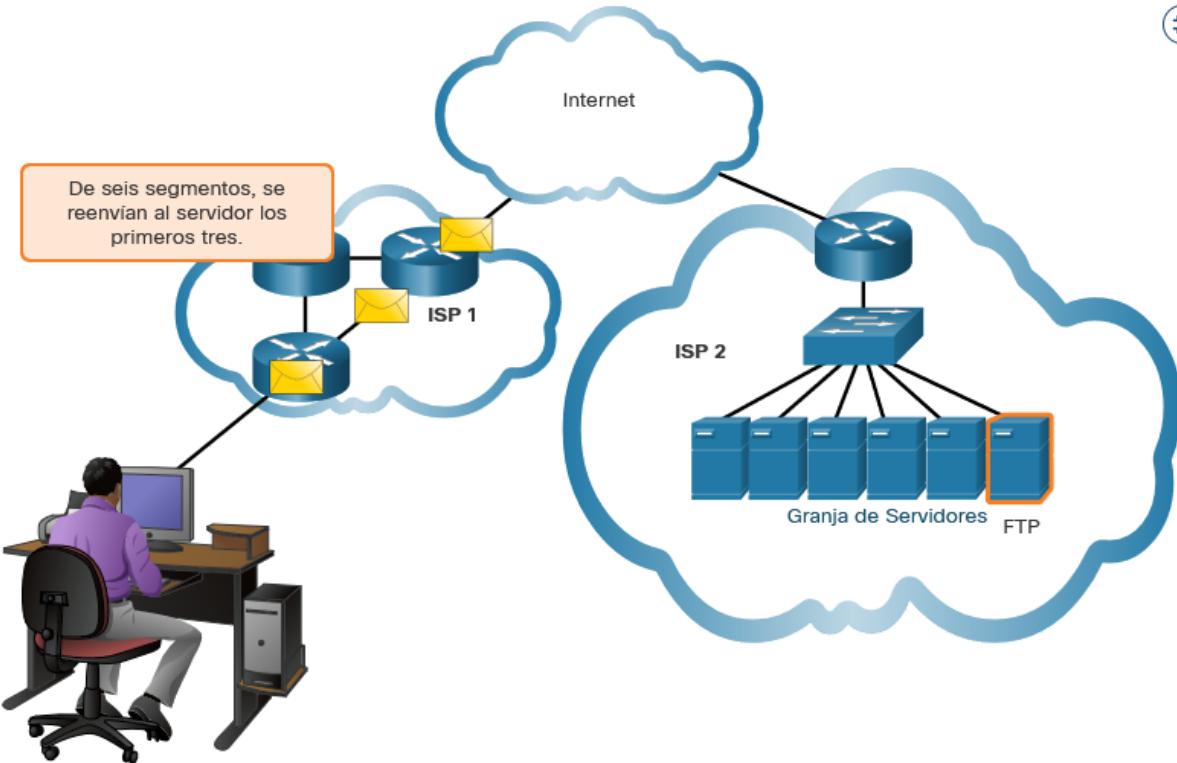
La función del protocolo de transporte TCP es similar al envío de paquetes de los que se hace un rastreo de origen a destino. Si se divide un pedido de envío en varios paquetes, un cliente puede verificar en línea para ver el orden de la entrega.

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.





De seis segmentos, se reenvían al servidor los primeros tres.

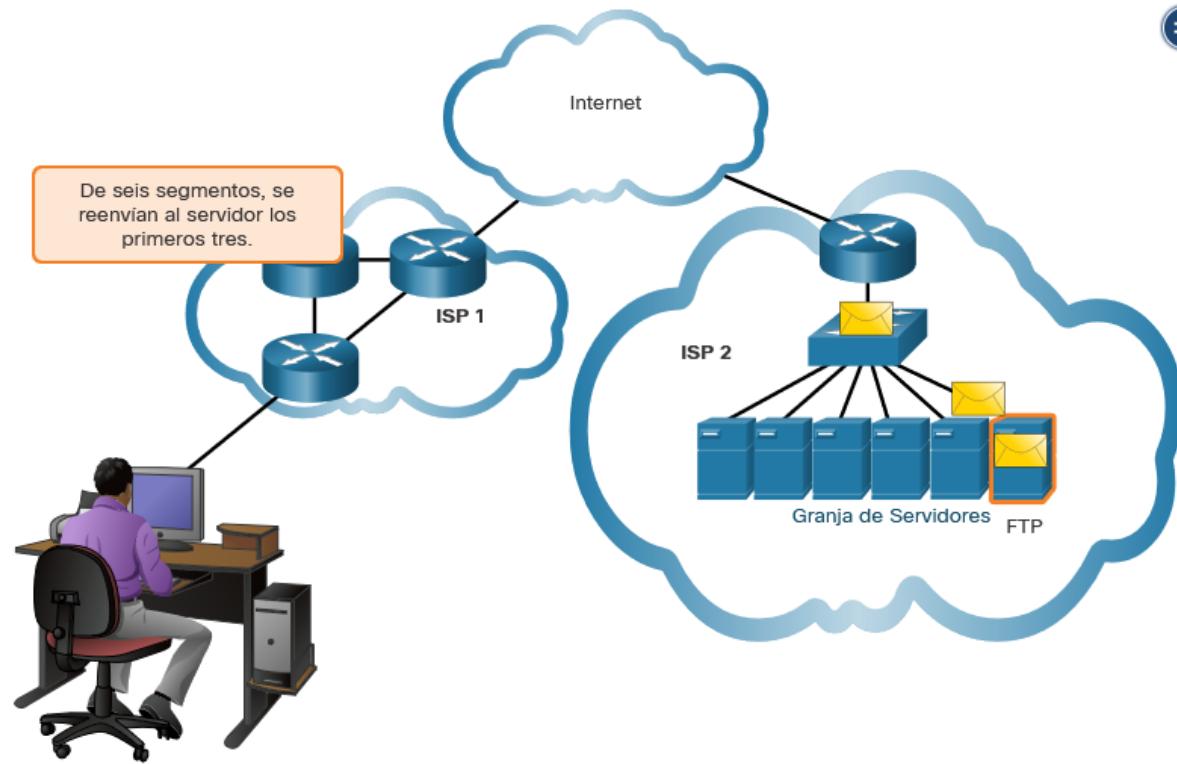


Internet

ISP 1

ISP 2

Granja de Servidores FTP



De seis segmentos, se reenvían al servidor los primeros tres.

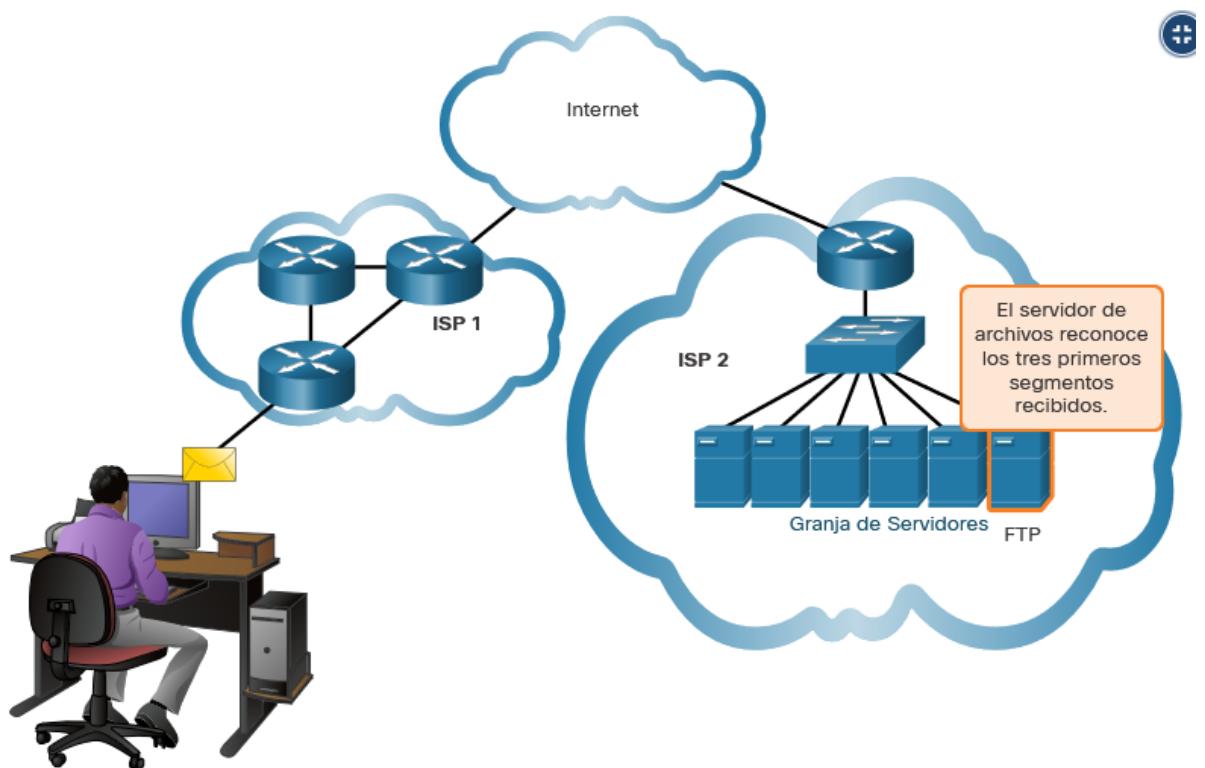
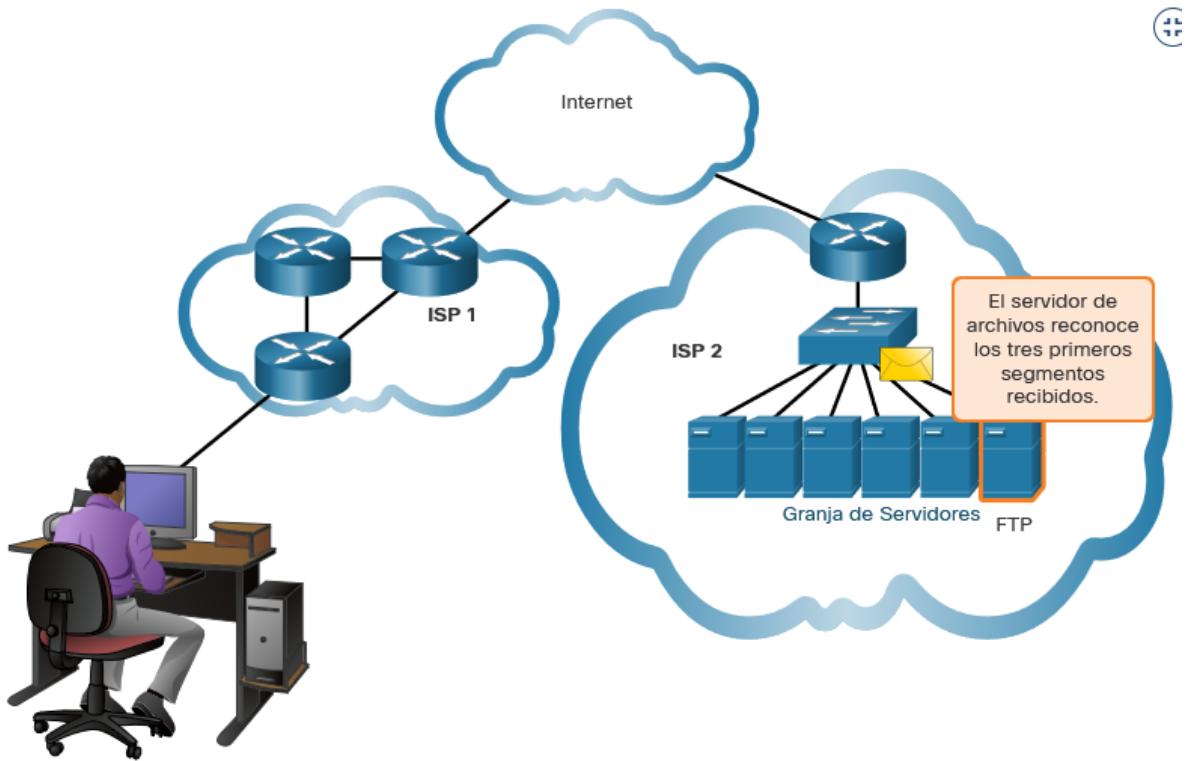


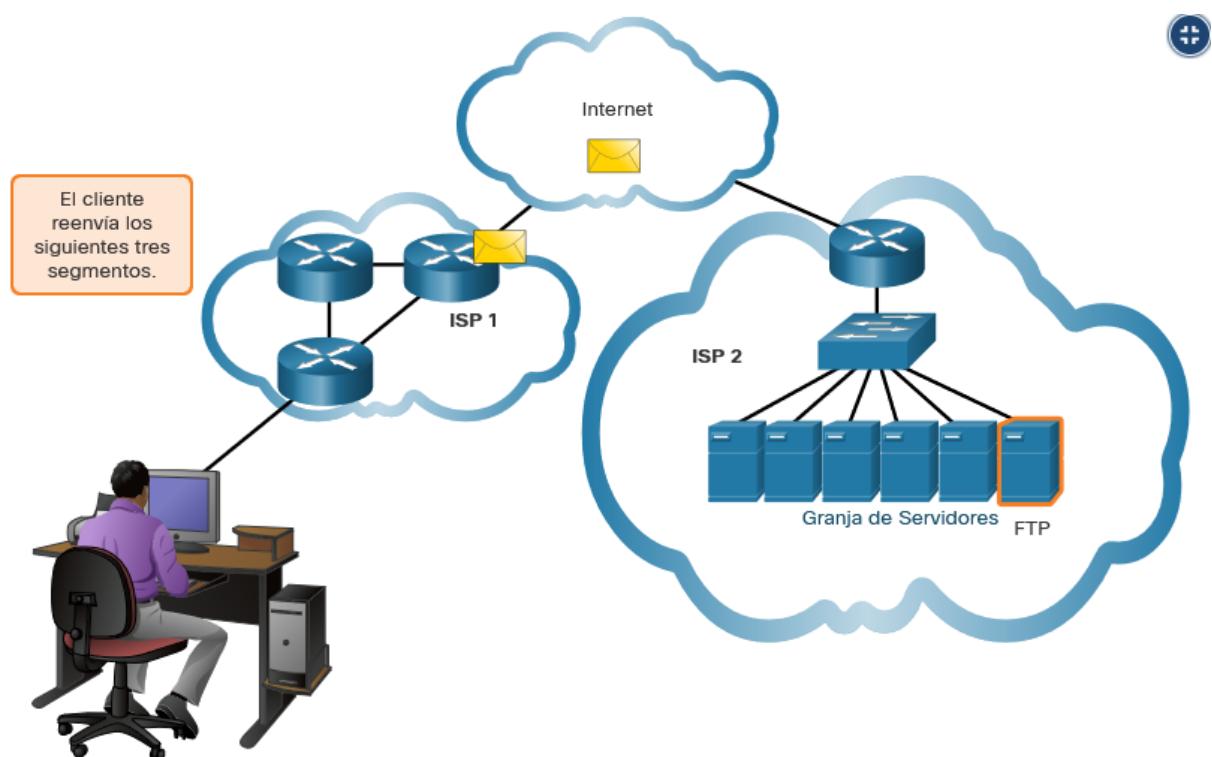
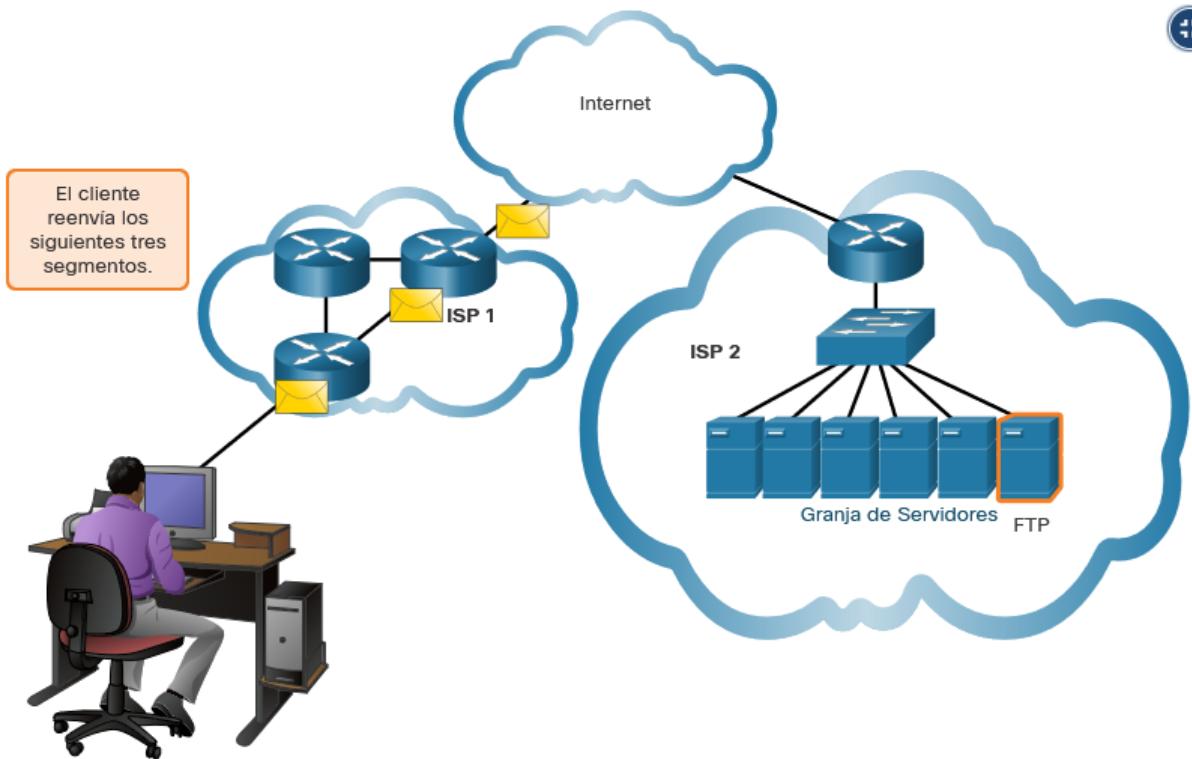
Internet

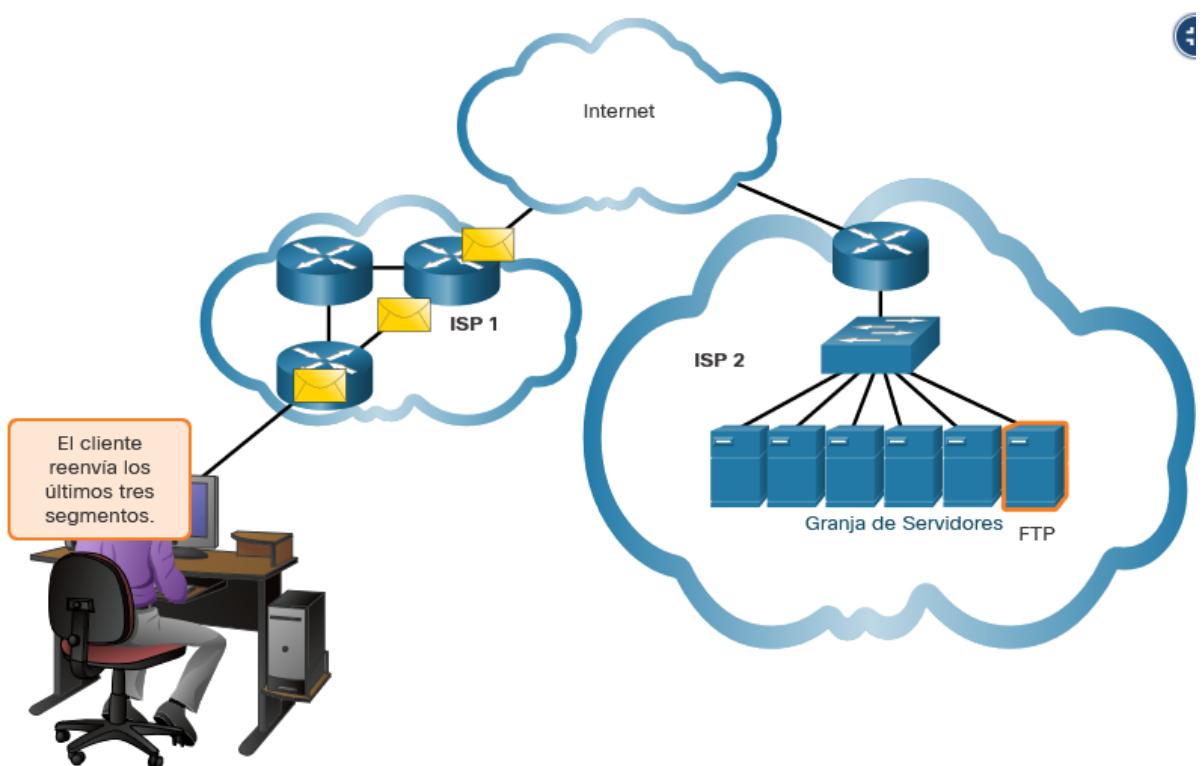
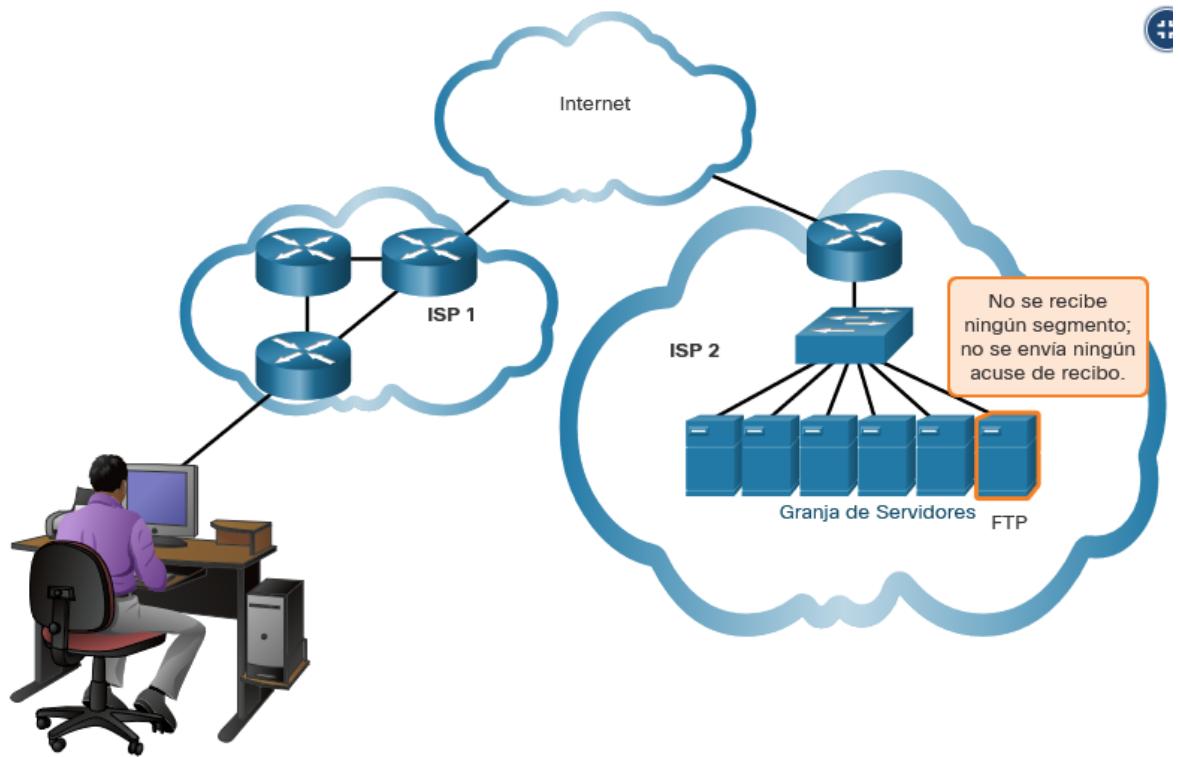
ISP 1

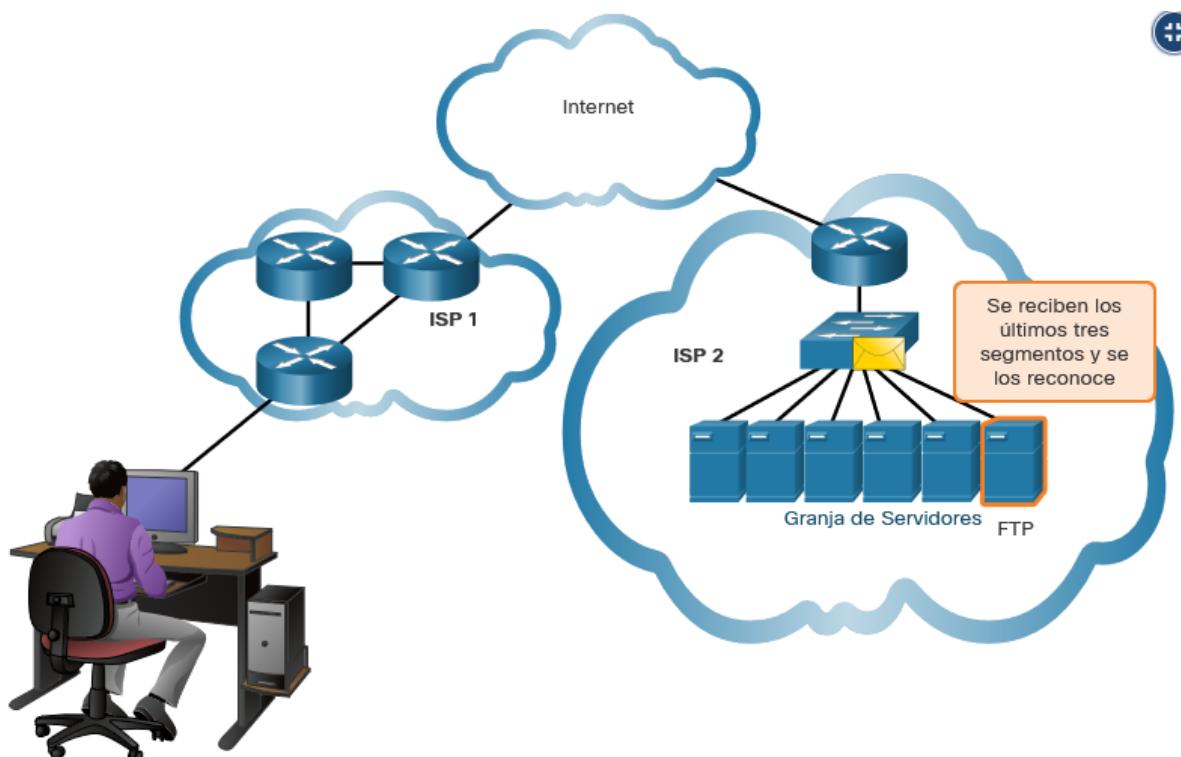
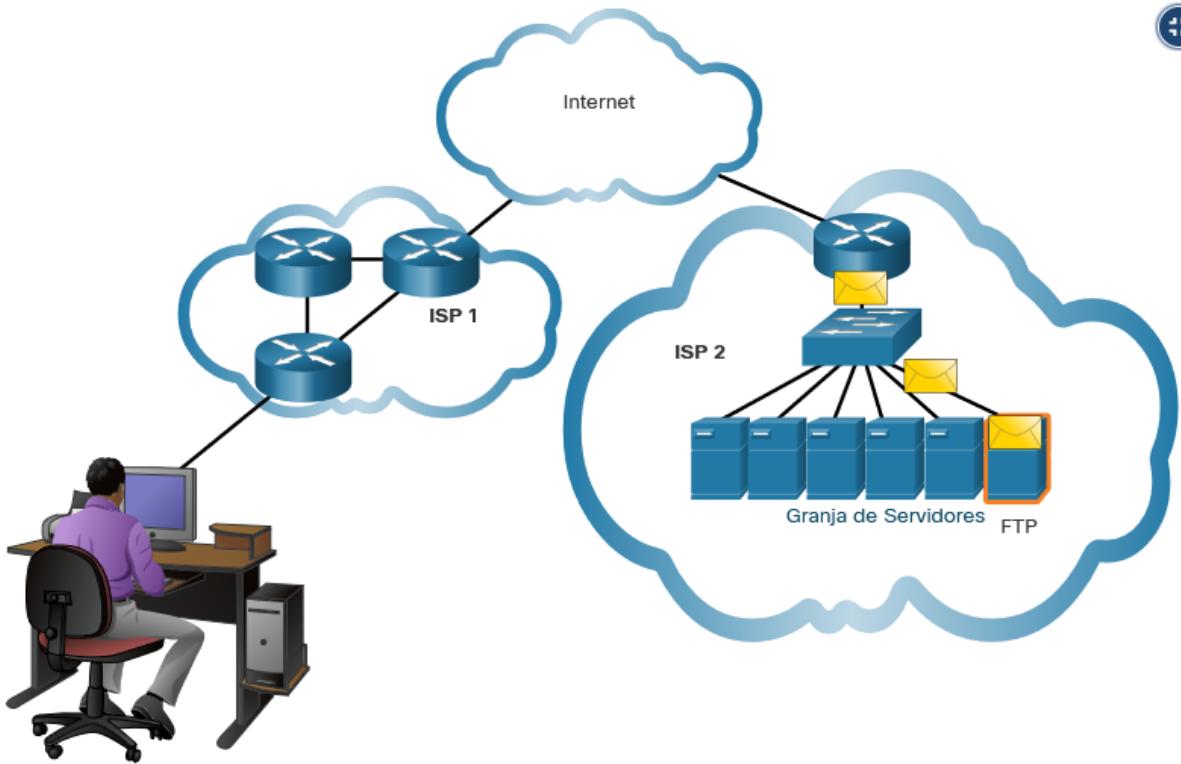
ISP 2

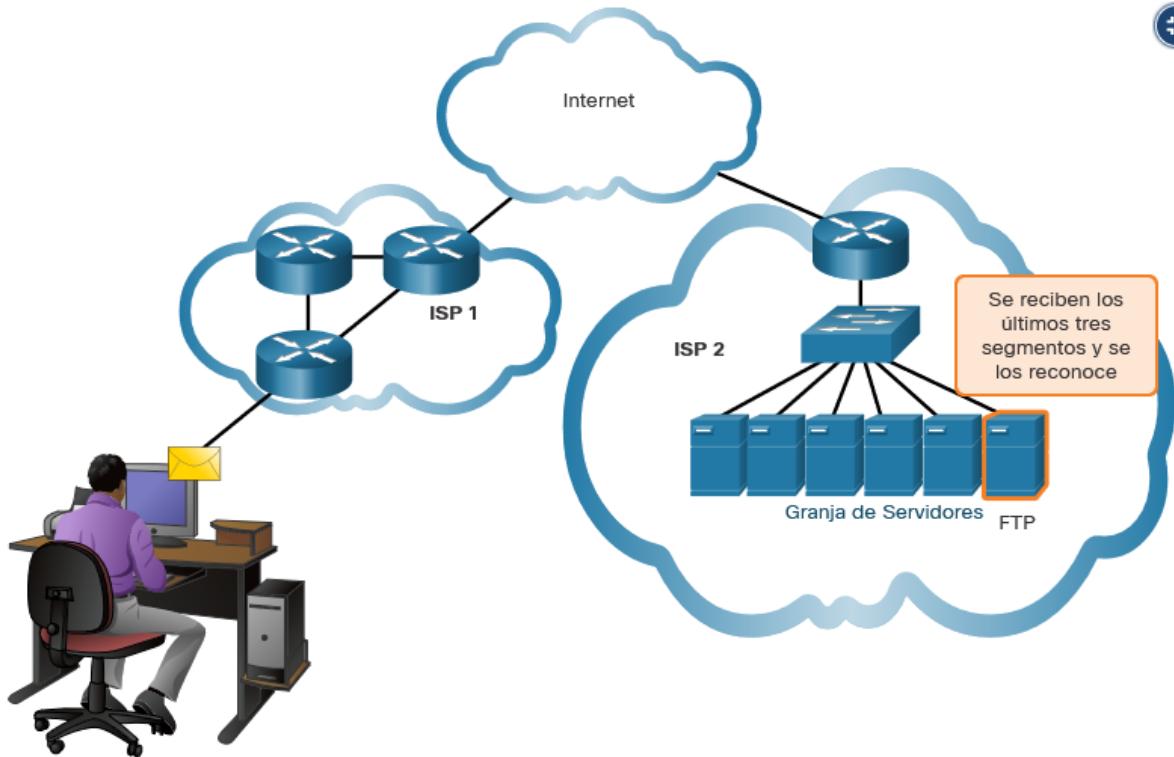
Granja de Servidores FTP











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.

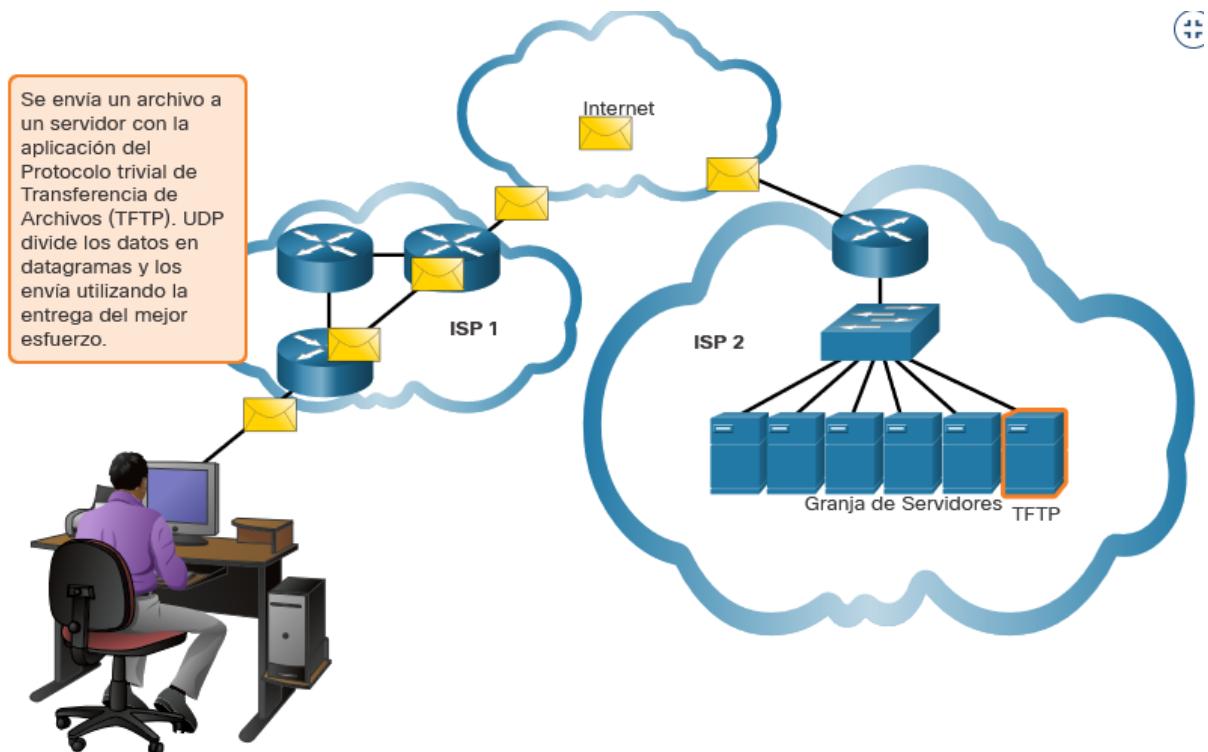
Nota: UDP divide los datos en datagramas que también se conocen como segmentos.

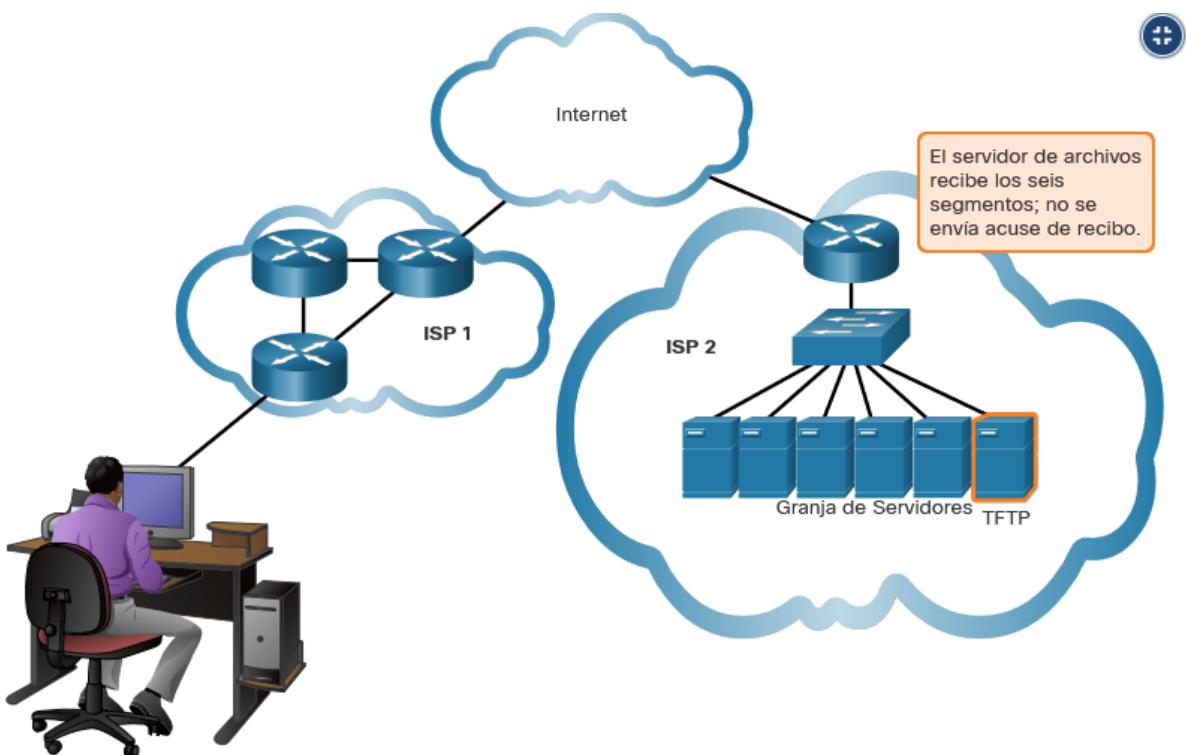
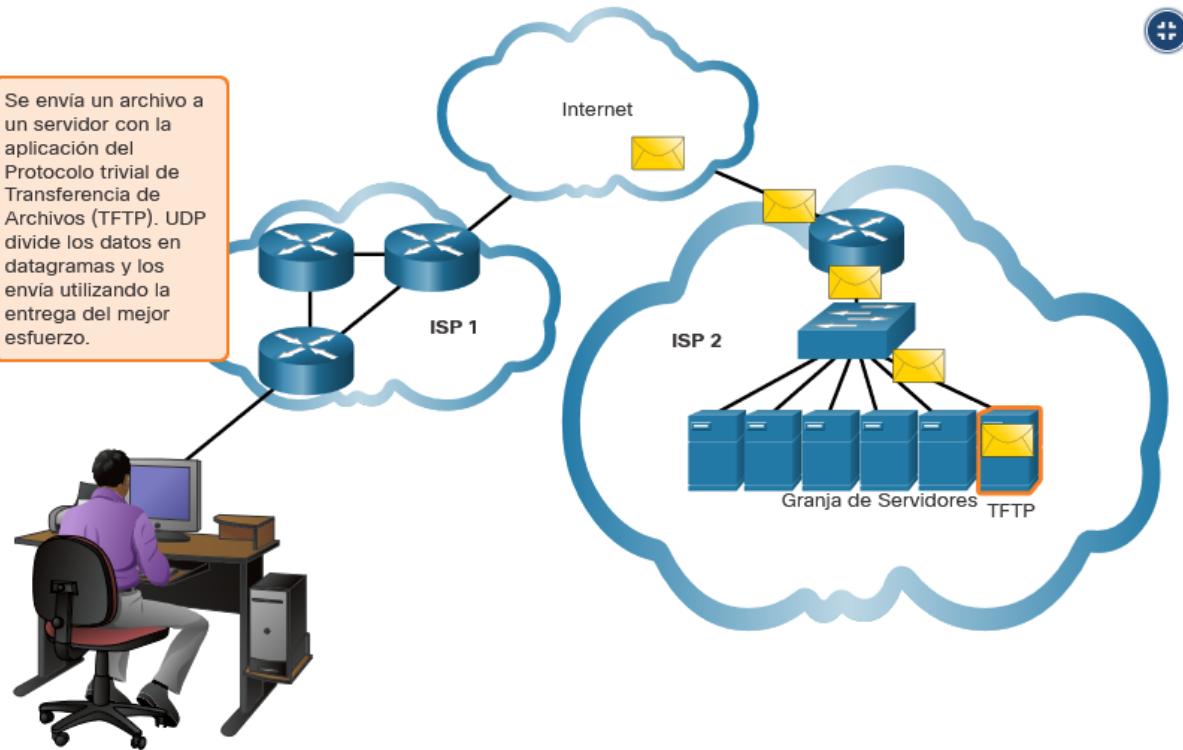
UDP es un protocolo sin conexión. Debido a que UDP no proporciona fiabilidad ni control de flujo, no requiere una conexión establecida. Debido a que UDP no realiza un seguimiento de la información enviada o recibida entre el cliente y el servidor, UDP también se conoce como protocolo sin estado.

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.





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.

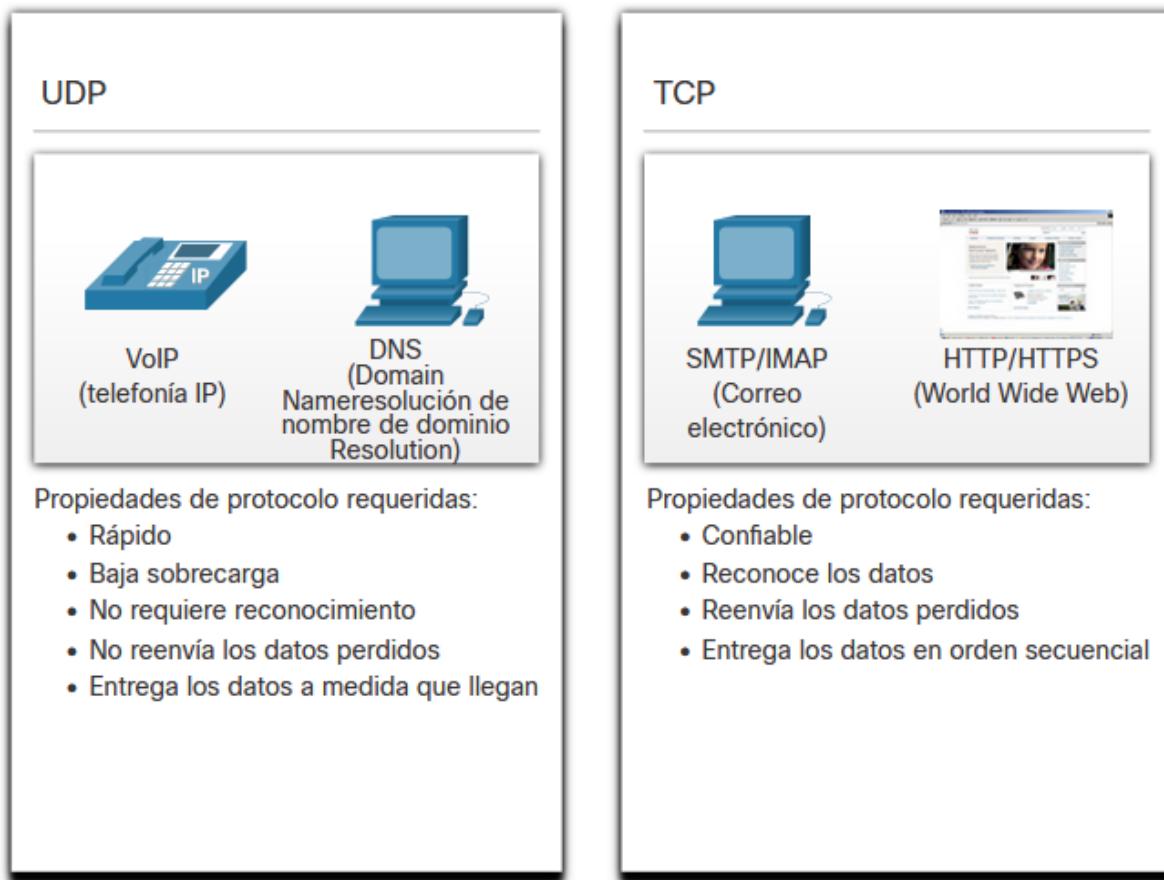
Para otras aplicaciones es importante que todos los datos lleguen y que puedan ser procesados en su secuencia adecuada. Para estos tipos de aplicaciones, TCP se utiliza como protocolo de transporte. Por ejemplo, las aplicaciones como las bases de datos, los navegadores web y los clientes de correo electrónico, 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.



Características de TCP

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 cabecera 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.

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.



Campos de Encabezado TCP

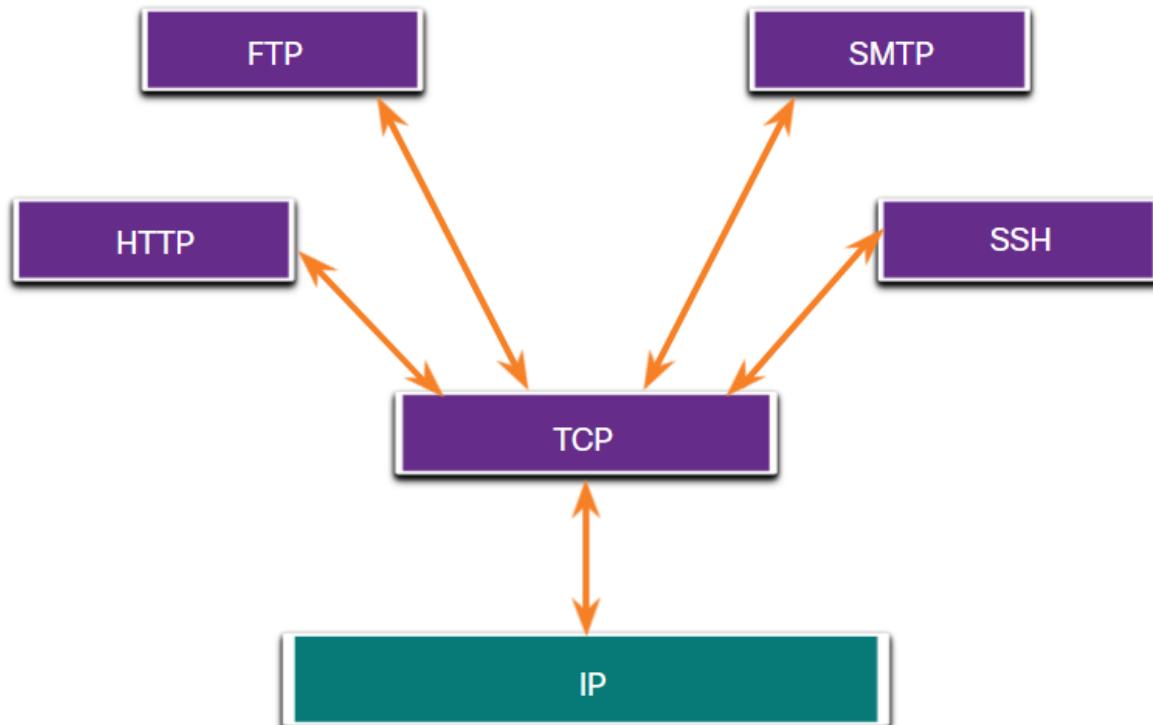
La tabla identifica y describe los diez campos de un encabezado TCP.

Campo de Encabezado TCP	Descripción
Puerto de Origen	Campo de 16 bits utilizado para identificar la aplicación de origen por número de puerto.
Puerto de Destino	Un campo de 16 bits utilizado para identificar la aplicación de destino por puerto número.
Secuencia de Números	Campo de 32 bits utilizado para reensamblar datos.
Número de Acuse de Recibo	Un campo de 32 bits utilizado para indicar que se han recibido datos y el siguiente byte esperado de la fuente.
Longitud del Encabezado	Un campo de 4 bits conocido como «desplazamiento de datos» que indica la propiedad longitud del encabezado del segmento TCP.
Reservado	Un campo de 6 bits que está reservado para uso futuro.
Bits de Control	Un campo de 6 bits utilizado que incluye códigos de bits, o indicadores, que indican el propósito y función del

	segmento TCP.
Tamaño de la ventana	Un campo de 16 bits utilizado para indicar el número de bytes que se pueden aceptar a la vez.
Suma de Comprobación	A 16-bit field used for error checking of the segment header and data.
Urgente	Campo de 16 bits utilizado para indicar si los datos contenidos son urgentes.

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.



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.

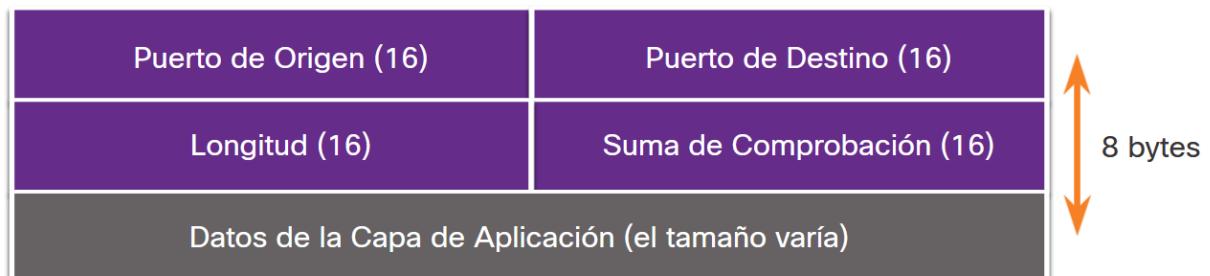
Encabezado UDP

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.



Campos de Encabezado UDP

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

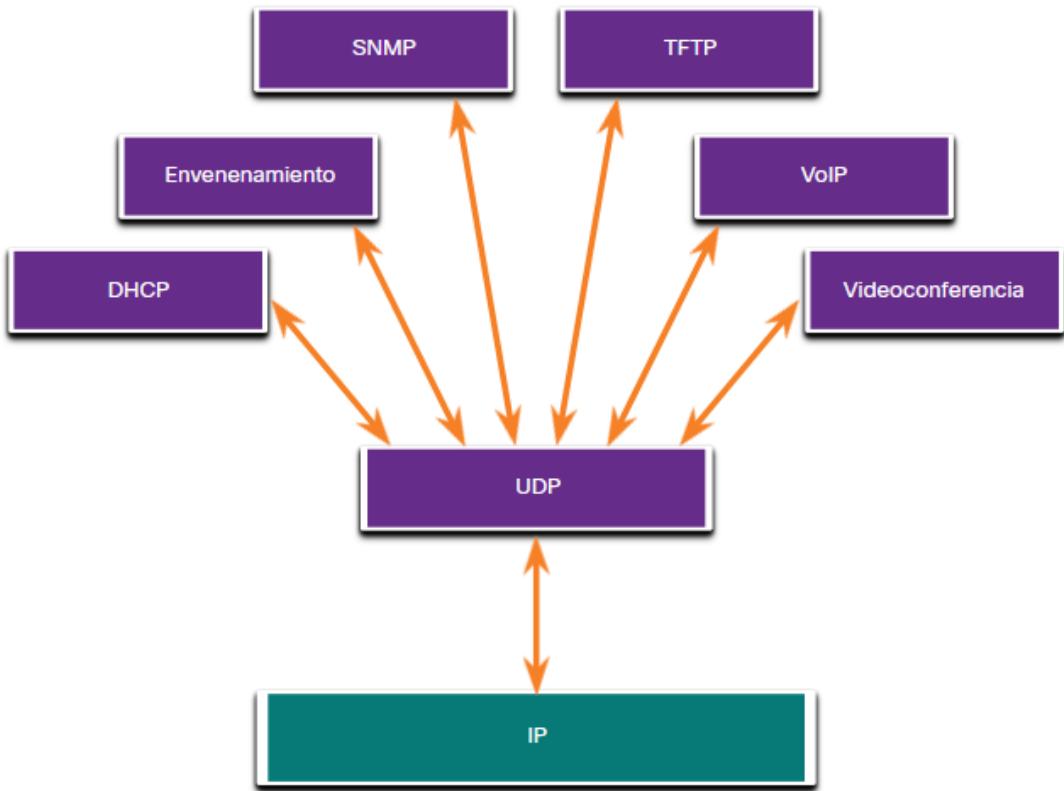
Campo de Encabezado UDP	Descripción
Puerto de Origen	Campo de 16 bits utilizado para identificar la aplicación de origen por número de puerto.
Puerto de Destino	Un campo de 16 bits utilizado para identificar la aplicación de destino por puerto número.
Longitud	Campo de 16 bits que indica la longitud del encabezado del datagrama UDP.
Suma de comprobación	Campo de 16 bits utilizado para la comprobación de errores del encabezado y los datos del datagrama.

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.



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.

Números de puerto

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.



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.

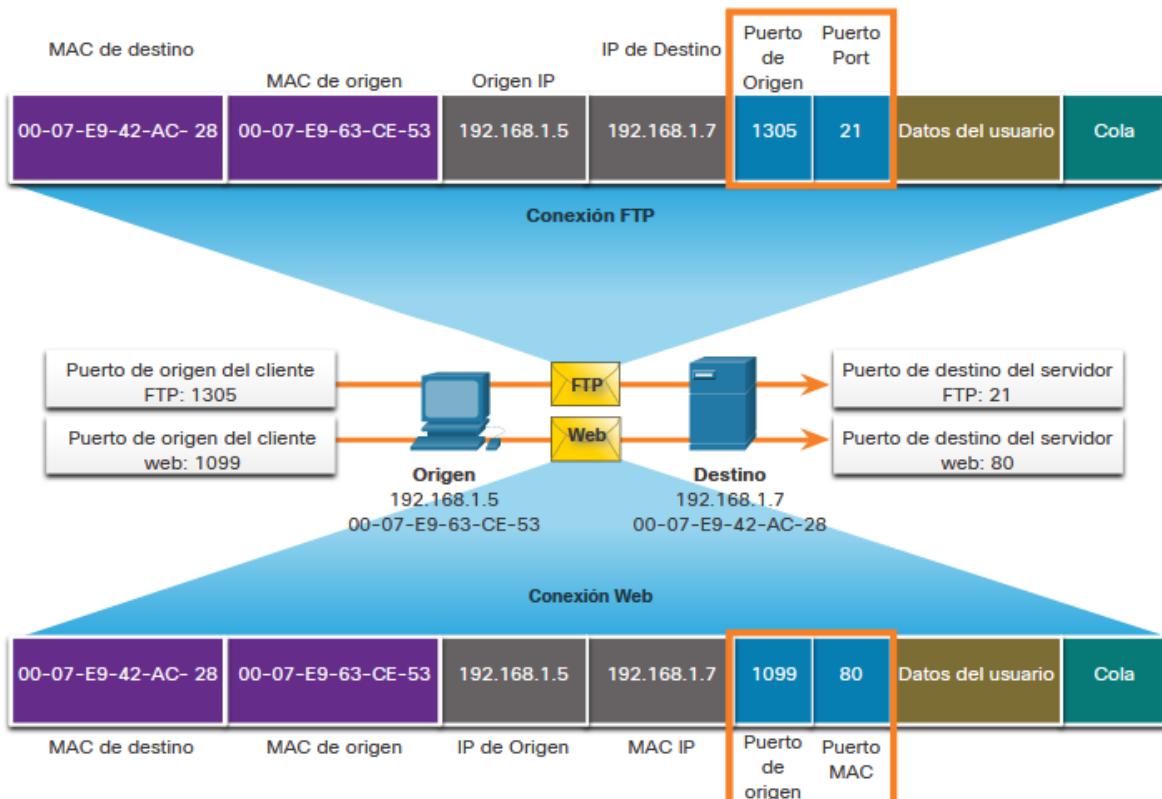
En la solicitud, el número de puerto de destino es lo que identifica el tipo de servicio que se solicita del servidor web de destino. Por ejemplo, cuando un cliente especifica el puerto 80 en el puerto de destino, el servidor que recibe el mensaje sabe que se solicitan servicios web.

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.

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.



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.

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.

Grupo de puertos	Rango de números	Descripción
Puertos bien conocidos	0 - 1,023	<ul style="list-style-type: none">• Estos números de puerto están reservados para servicios comunes o populares y aplicaciones como navegadores web, clientes de correo electrónico y acceso remoto clientes.• Los puertos conocidos definidos para aplicaciones de servidor comunes permiten para identificar fácilmente el servicio asociado requerido.
Puertos registrados	1,024 - 49,151	<ul style="list-style-type: none">• Estos números de puerto son asignados por IANA a una entidad solicitante a utilizar con procesos o aplicaciones específicos.• Estos procesos son principalmente aplicaciones individuales que un usuario ha elegido instalar, en lugar de aplicaciones comunes que recibir un número de puerto conocido.• Por ejemplo, Cisco ha registrado el puerto 1812 para su servidor RADIUS proceso de autenticación

Privada y/o puertos dinámicos	49,152 - 65,535	<ul style="list-style-type: none"> Estos puertos también se conocen como puertos efímeros. El sistema operativo del cliente generalmente asigna números de puerto dinámicamente cuando se inicia una conexión a un servicio. El puerto dinámico se utiliza para identificar la aplicación del cliente durante la comunicación
--------------------------------------	------------------------	---

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.

Número de puerto	Protocolo	Aplicación
20	TCP	Protocolo de transferencia de archivos (FTP) - Datos
21	TCP	Protocolo de transferencia de archivos (FTP) - Control
22	TCP	Secure Shell (SSH)
23	TCP	Telnet
25	TCP	Protocolo simple de transferencia de correo (SMTP)
53	UDP, TCP	Servicio de nombres de dominio (DNS, Domain Name System)
67	UDP	Protocolo de configuración dinámica de host (DHCP): servidor
68	UDP	Protocolo de configuración dinámica de host: cliente
69	UDP	Protocolo trivial de transferencia de archivos (TFTP)
80	TCP	Protocolo de transferencia de hipertexto (HTTP)

110	TCP	Protocolo de oficina de correos, versión 3 (POP3)
143	TCP	Protocolo de acceso a mensajes de Internet (IMAP)
161	UDP	Protocolo simple de administración de redes (SNMP)
443	TCP	Protocolo seguro de transferencia de hipertexto (HTTPS)

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.

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   ESTABLISHED
  TCP    192.168.1.124:3159    207.138.126.169:http   ESTABLISHED
  TCP    192.168.1.124:3160    207.138.126.169:http   ESTABLISHED
  TCP    192.168.1.124:3161    sc.msn.com:http       ESTABLISHED
  TCP    192.168.1.124:3166    www.cisco.com:http     ESTABLISHED
  (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.

Proceso de comunicación TCP

Procesos del Servidor TCP

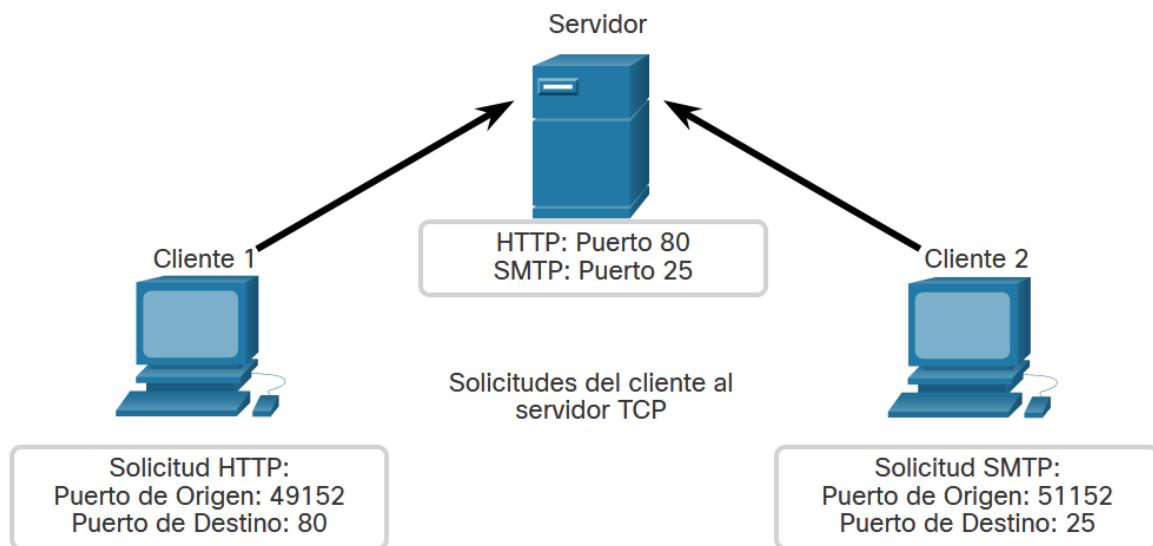
Cada proceso de aplicación que se ejecuta en el servidor para utilizar un número de puerto. El número de puerto es asignado automáticamente o configurado manualmente por un administrador del sistema.

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 direccionalada 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.

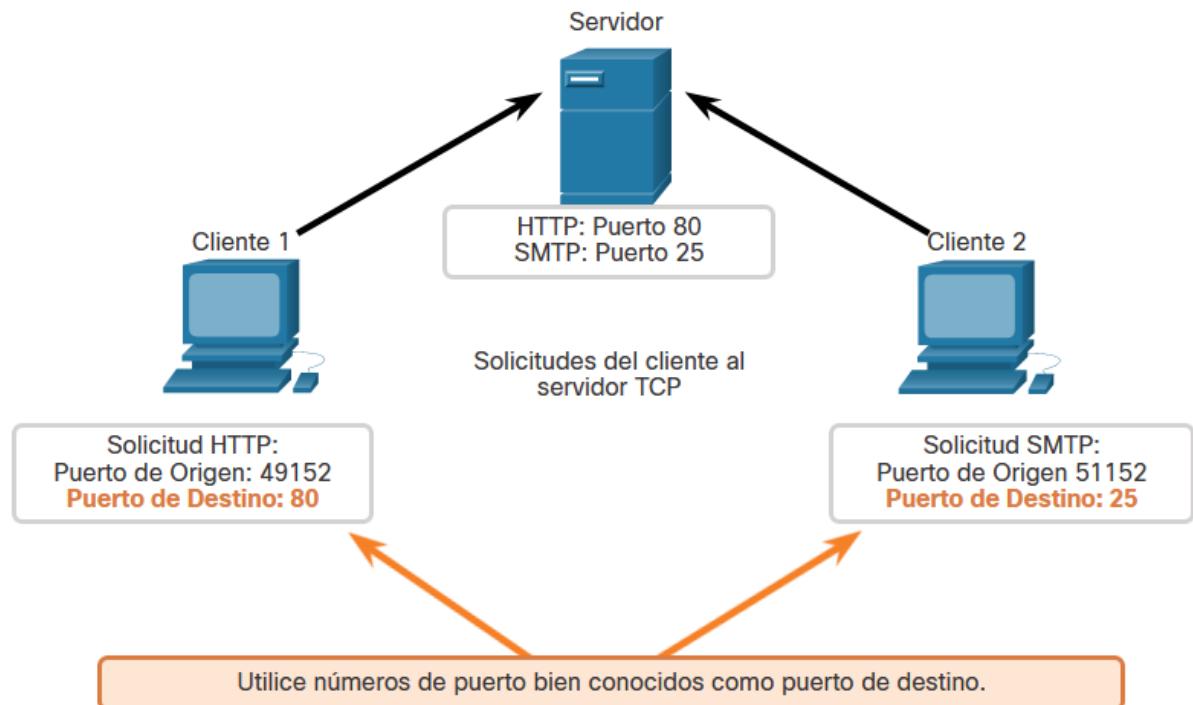
Clientes Envían Solicitudes TCP

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



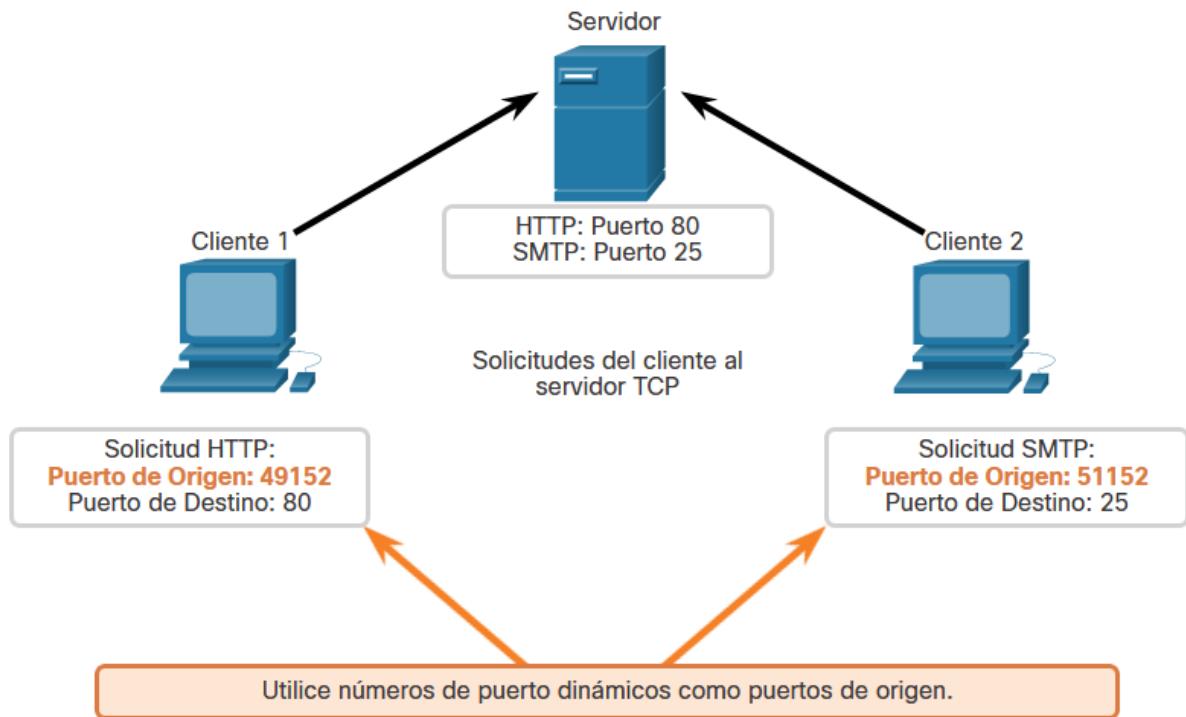
Solicitud de Puertos de Destino

Las solicitudes generan dinámicamente un número de puerto de origen. En este caso, el cliente 1 está utilizando el puerto de origen 49152 y el cliente 2 está utilizando el puerto de origen 51152.



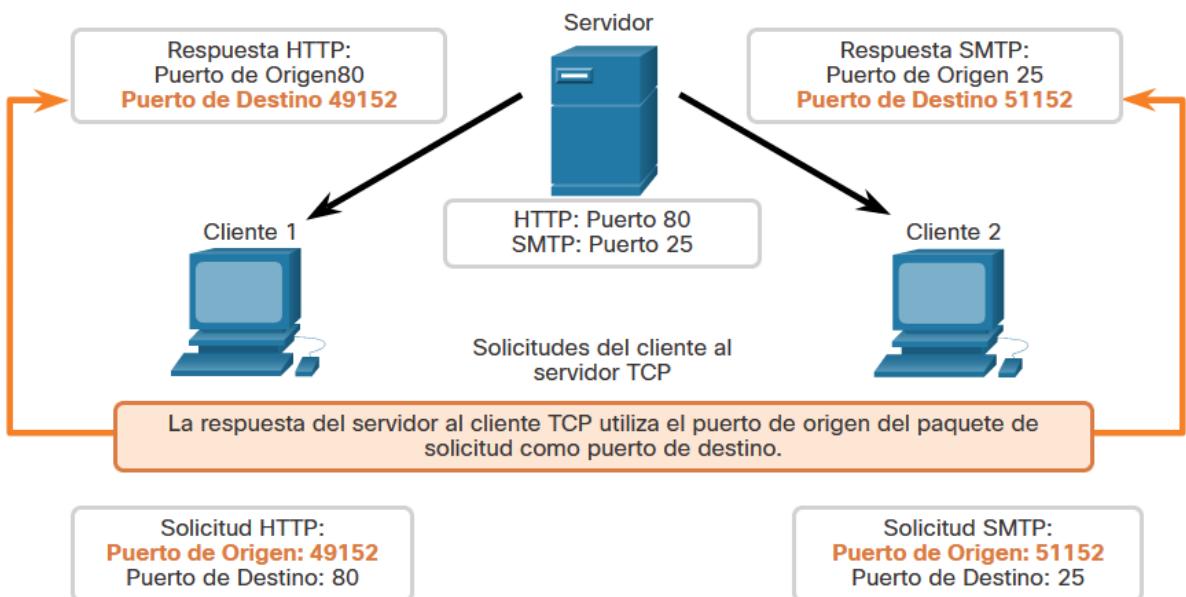
Solicitar Puertos de Origen

Las solicitudes de cliente generan dinámicamente un número de puerto de origen. En este caso, el Cliente 1 está utilizando el puerto de origen 49152 y el Cliente 2 está utilizando el puerto de origen 51152.



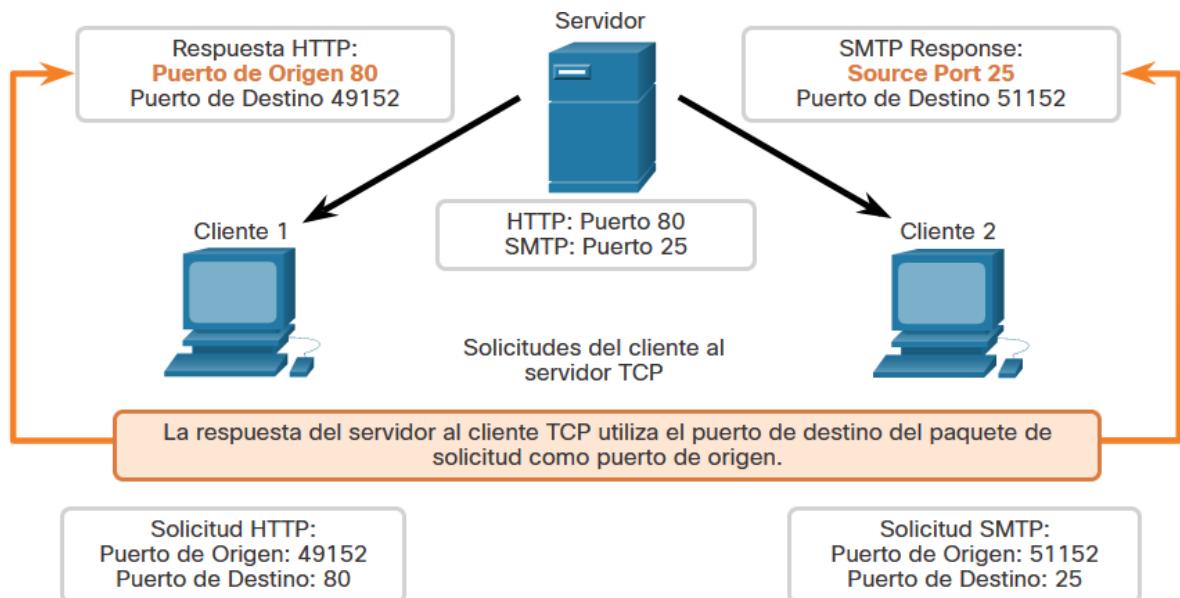
Respuesta de Puertos de Destino

Cuando el servidor responde a las solicitudes del cliente, invierte los puertos de destino y origen de la solicitud inicial. Observe que la respuesta del servidor a la solicitud web ahora tiene el puerto de destino 49152 y la respuesta de correo electrónico ahora tiene el puerto de destino 51152.



Respuesta de Puertos de Origen

El puerto de origen en la respuesta del servidor es el puerto de destino original en las solicitudes iniciales.

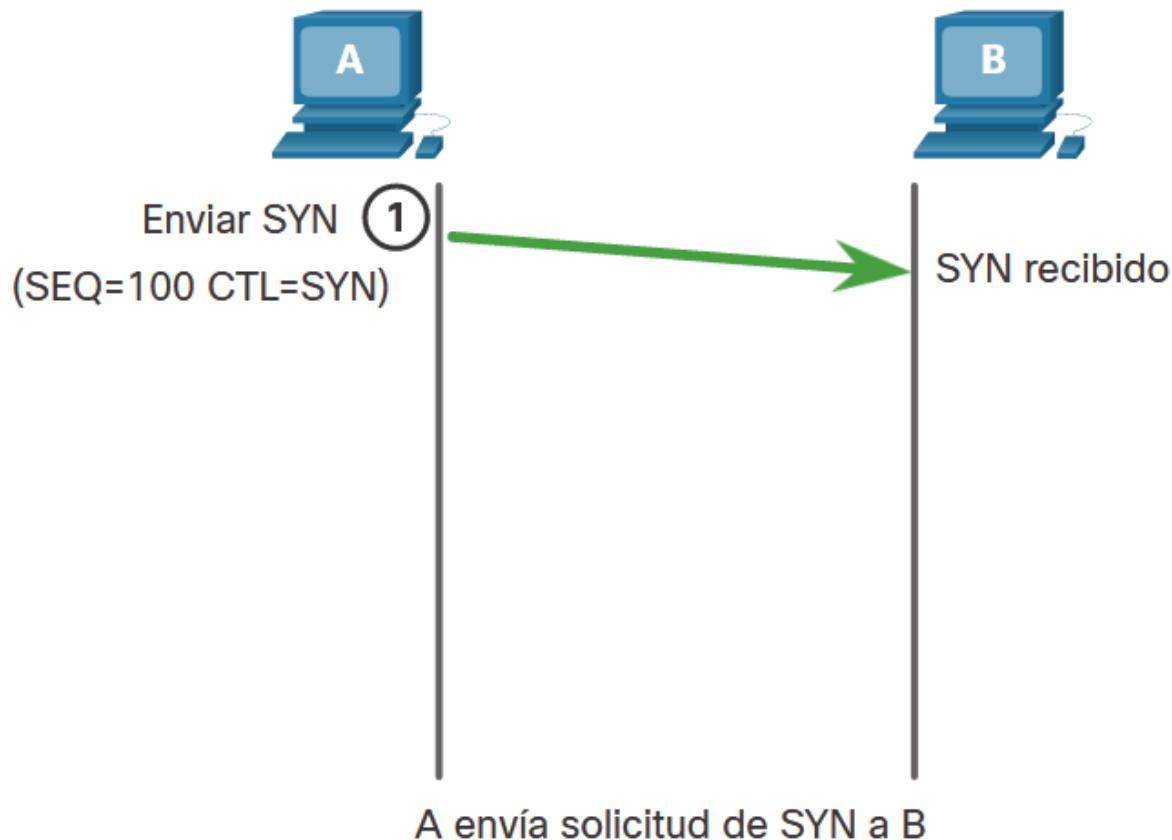


Establecimiento de Conexiones TCP

En algunas culturas, cuando dos personas se conocen, generalmente se saludan dándose la mano (shaking hands). 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.

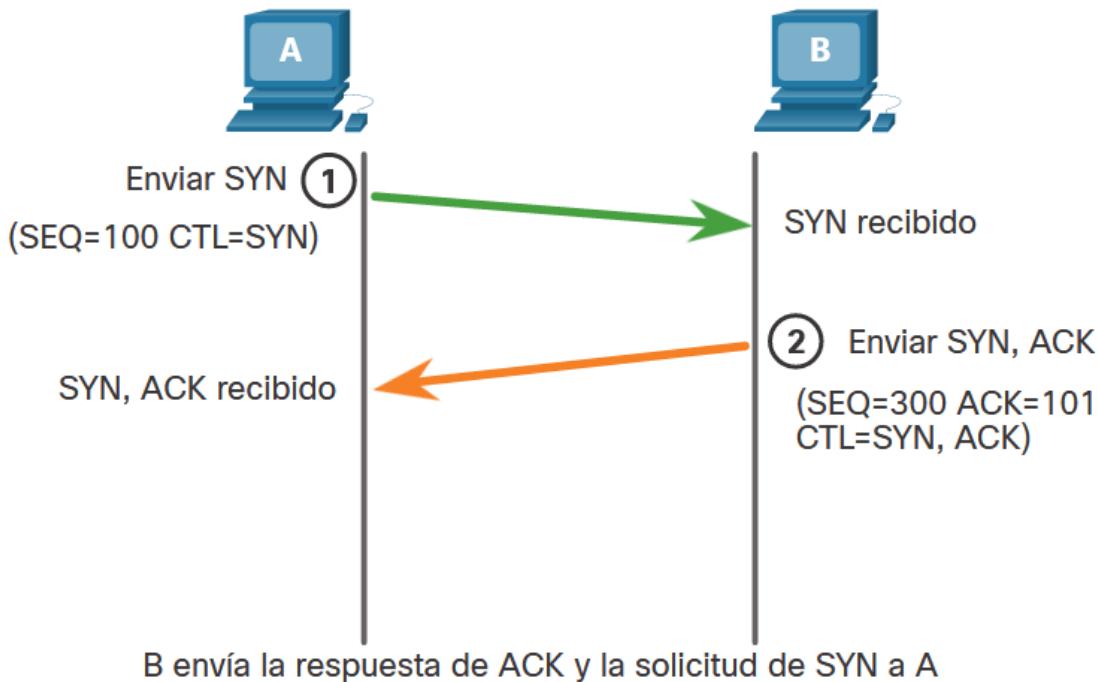
Paso 1. SYN

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



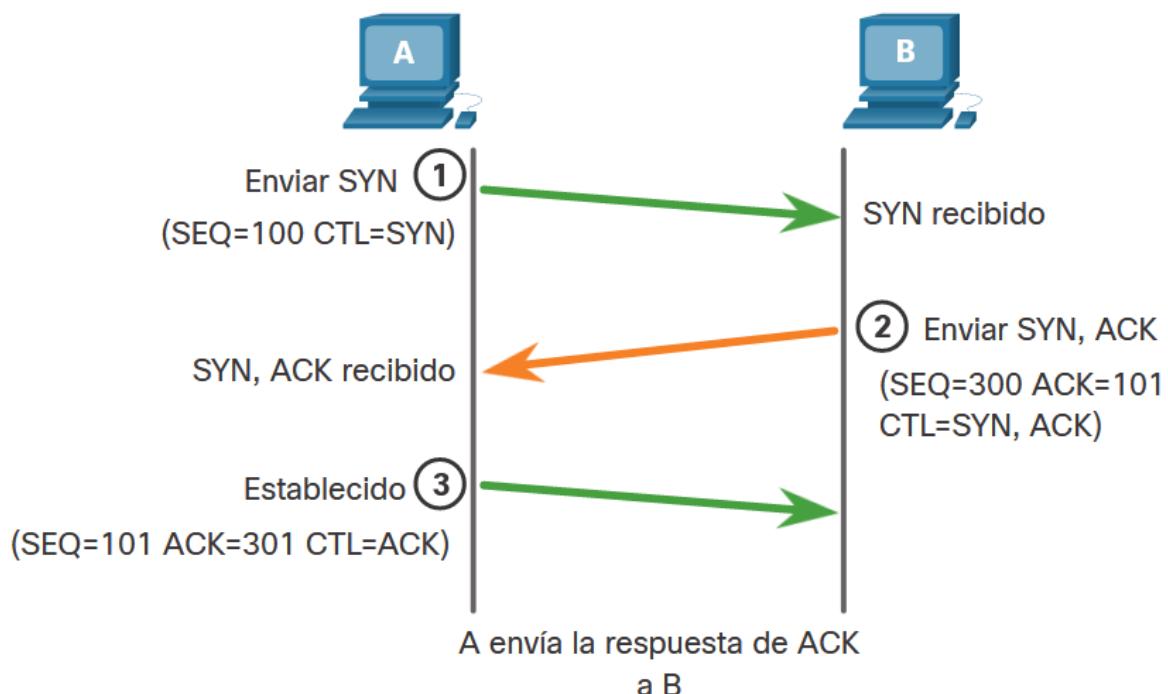
Paso 2. ACK y SYN

El servidor acusa recibo de la sesión de comunicación de cliente a servidor y solicita una sesión de comunicación de servidor a cliente.



Paso 3. ACK

El cliente de origen acusa recibo de la sesión de comunicación de servidor a cliente.



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.

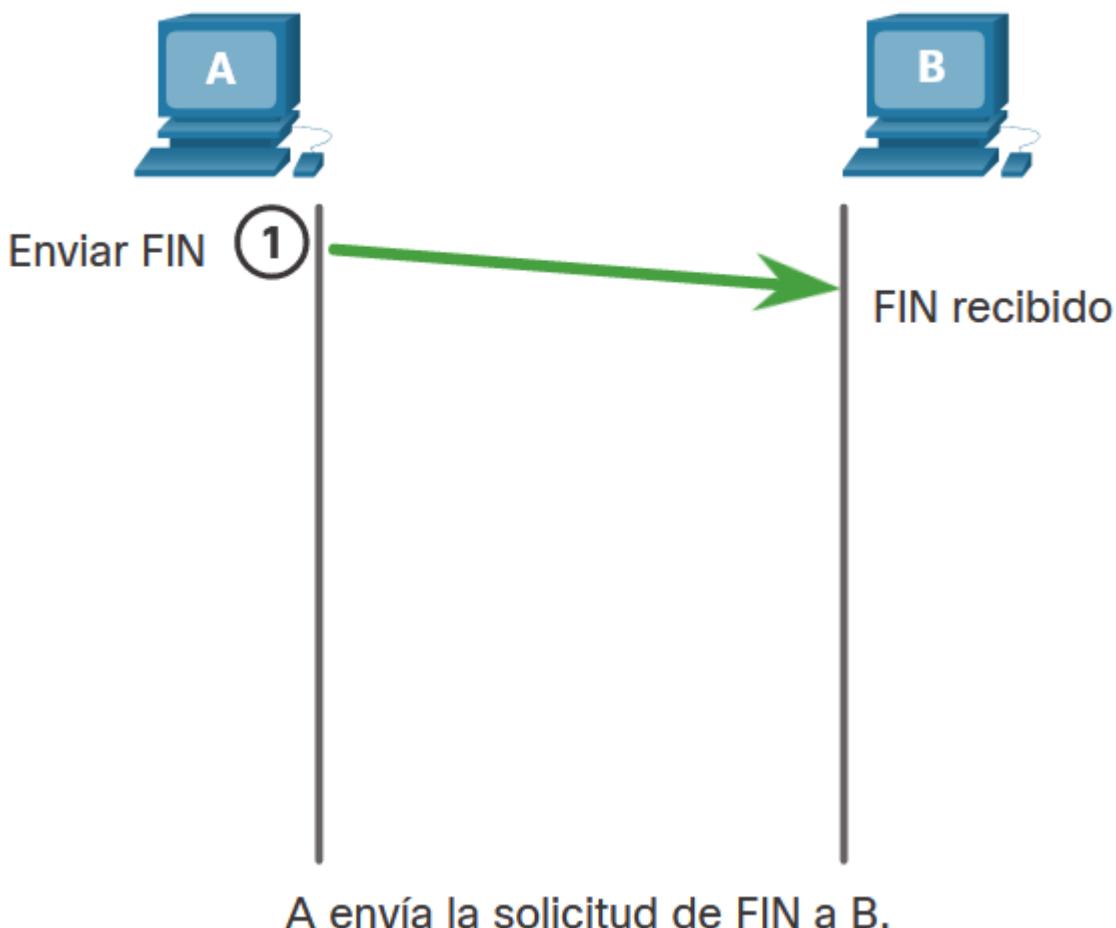
Terminación de Sesión

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.

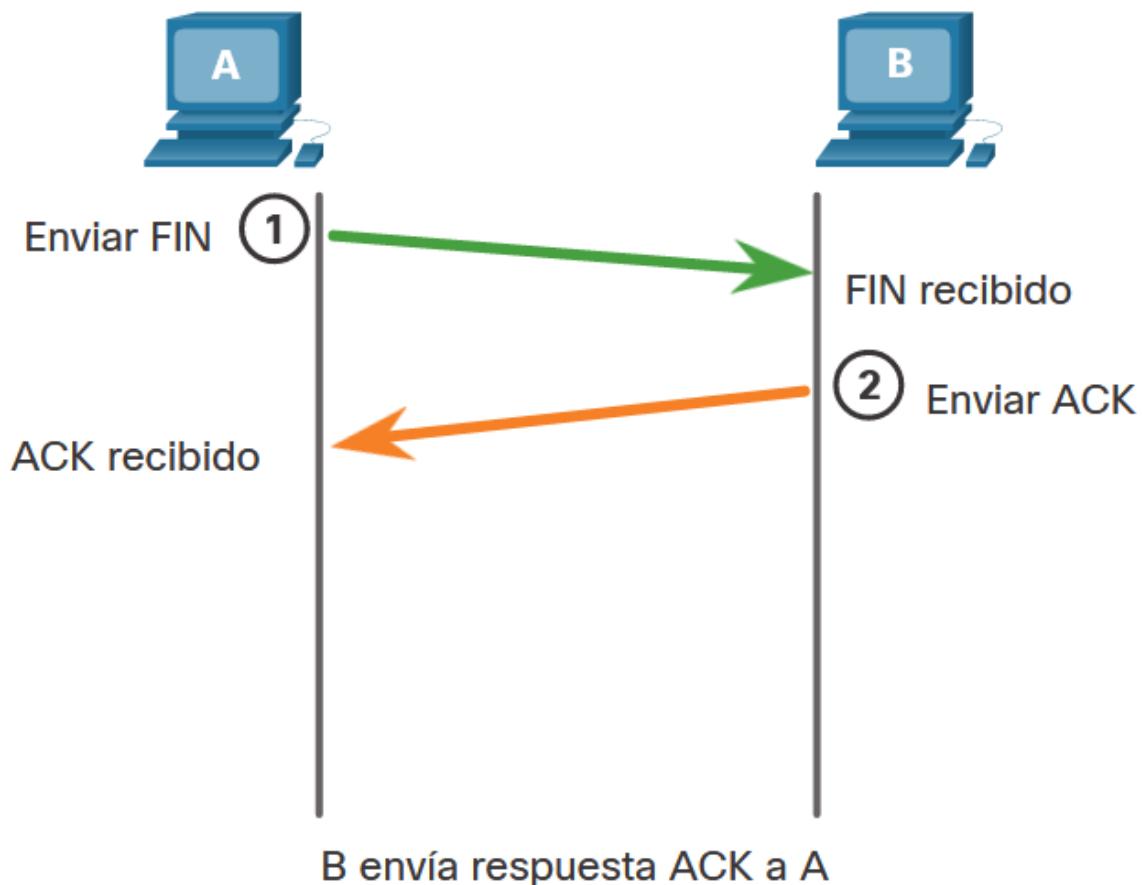
Paso 1. FIN

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



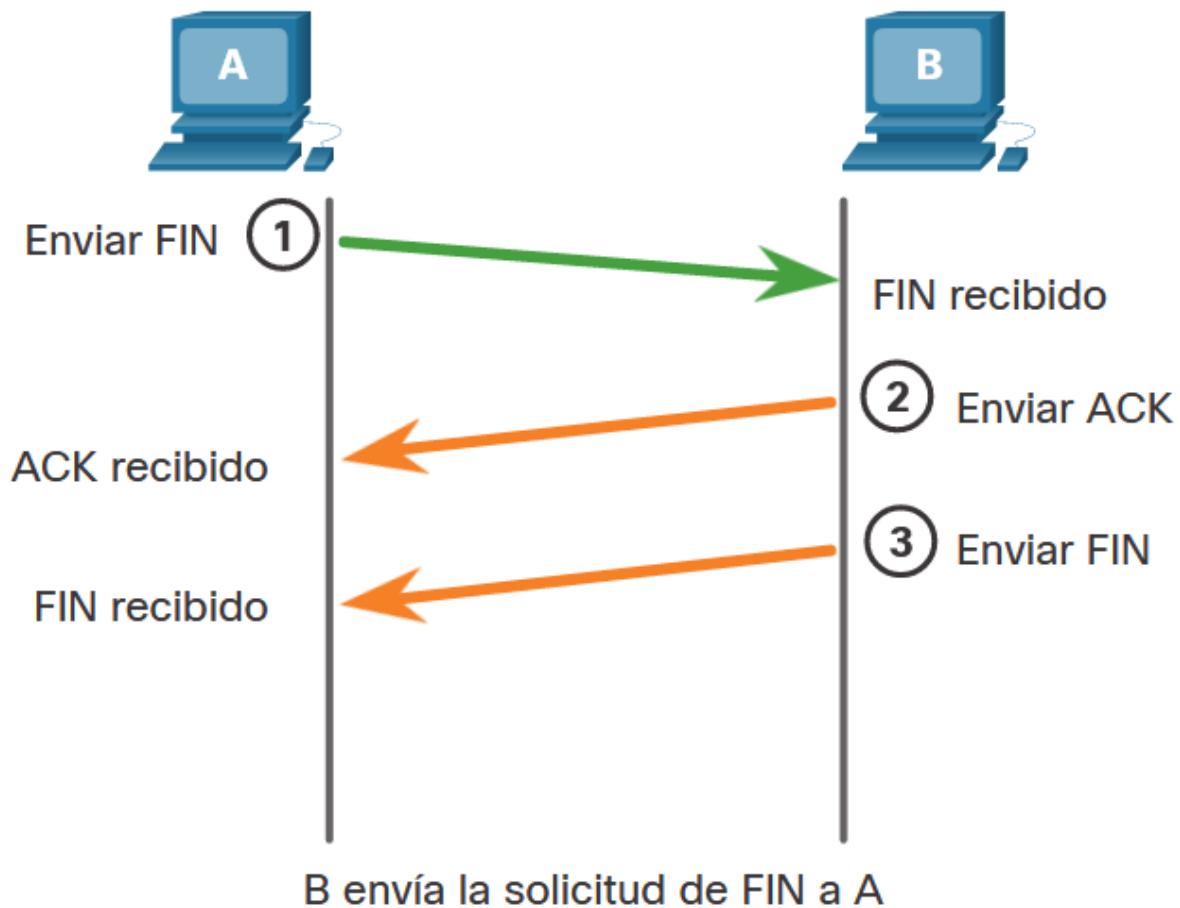
Paso 2. ACK

El servidor envía un ACK para acusar recibo del FIN para terminar la sesión de cliente a servidor.



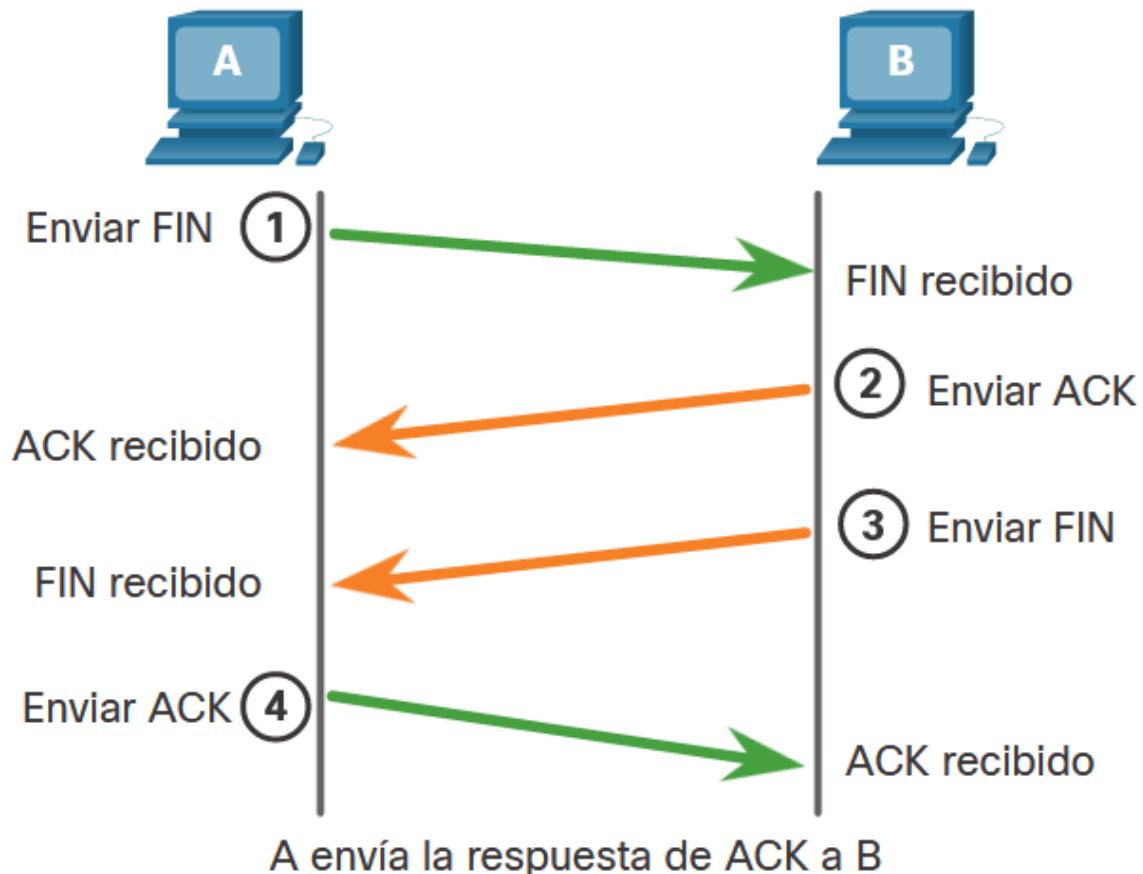
Paso 3. FIN

El servidor envía un FIN al cliente para terminar la sesión de servidor a cliente.



Paso 4. ACK

El cliente responde con un ACK para dar acuse de recibo del FIN desde el servidor.



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 (handshake) 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.

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

Campo de Bits de Control



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.

Confiabilidad y control de flujo

Fiabilidad de TCP: Entrega Garantizada y Ordenada

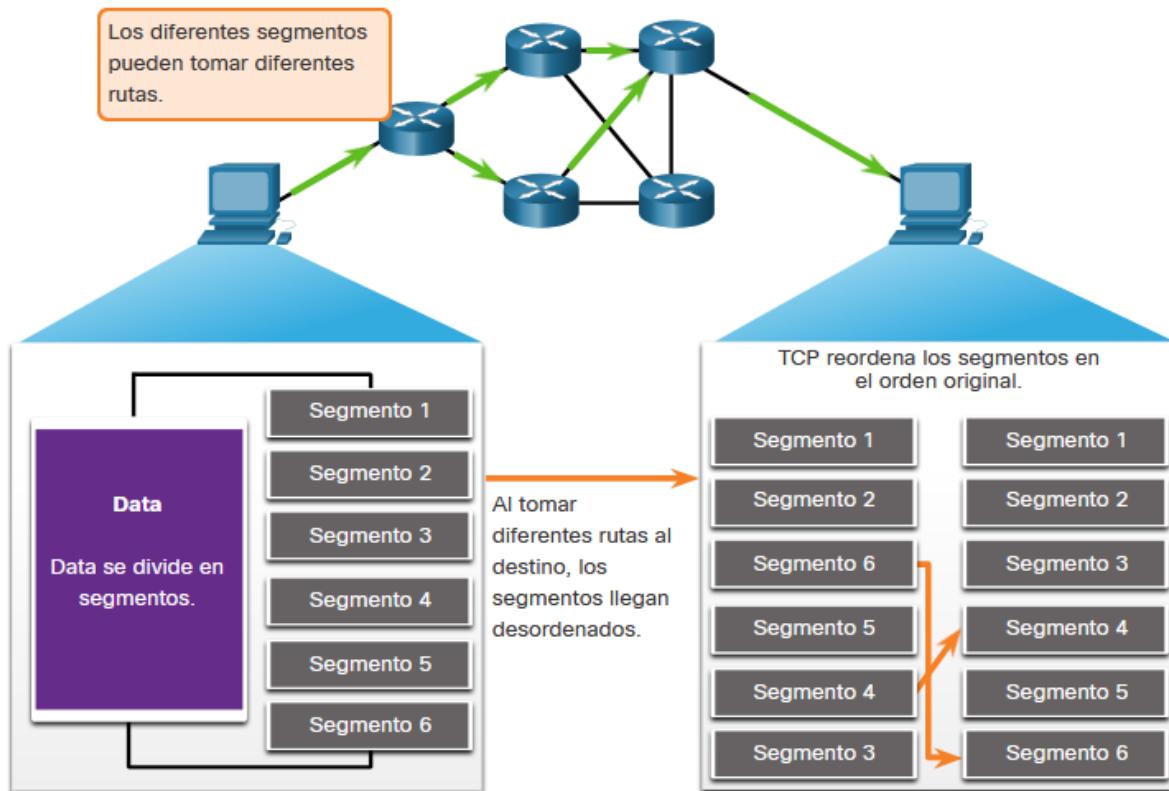
La razón por la que TCP es el mejor protocolo para algunas aplicaciones es porque, a diferencia de UDP, reenvía paquetes descartados y paquetes numerados para indicar su orden correcto antes de la entrega. TCP también puede ayudar a mantener el flujo de paquetes para que los dispositivos no se sobrecarguen.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

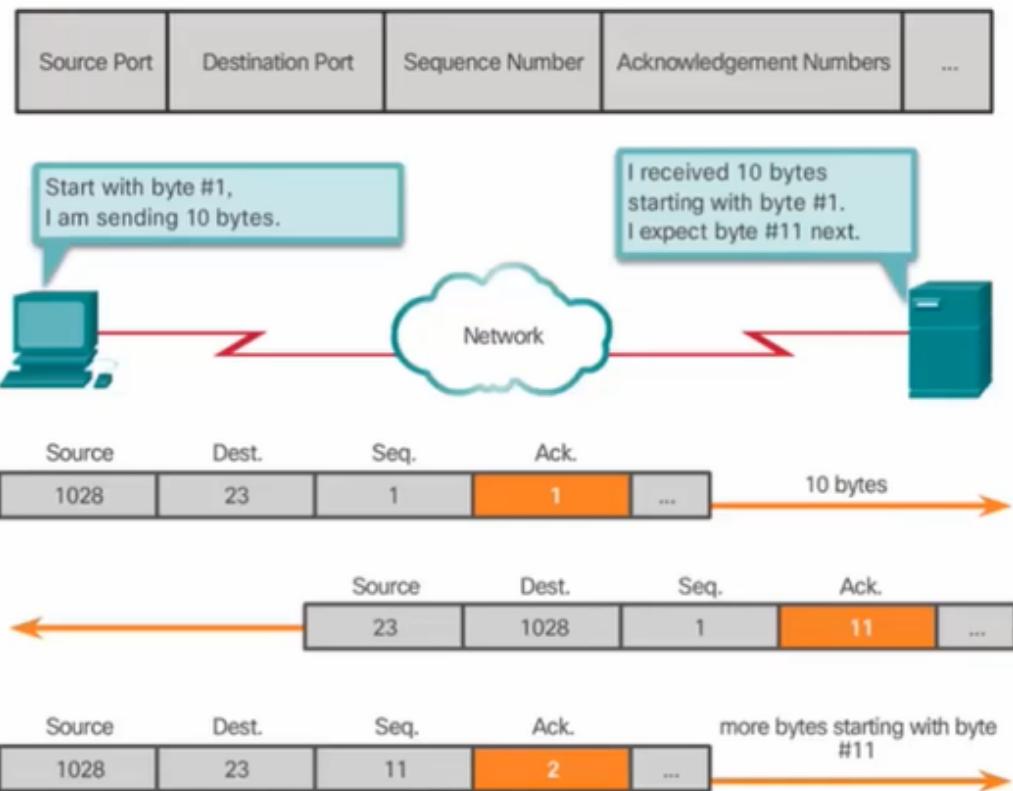
Durante la configuración de la sesión, se establece un número de secuencia inicial (ISN initial sequence number). Este ISN representa el valor inicial de los bytes que se transmiten a la aplicación receptora. A medida que se transmiten los datos durante la sesión, el número de secuencia se incrementa 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.



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.

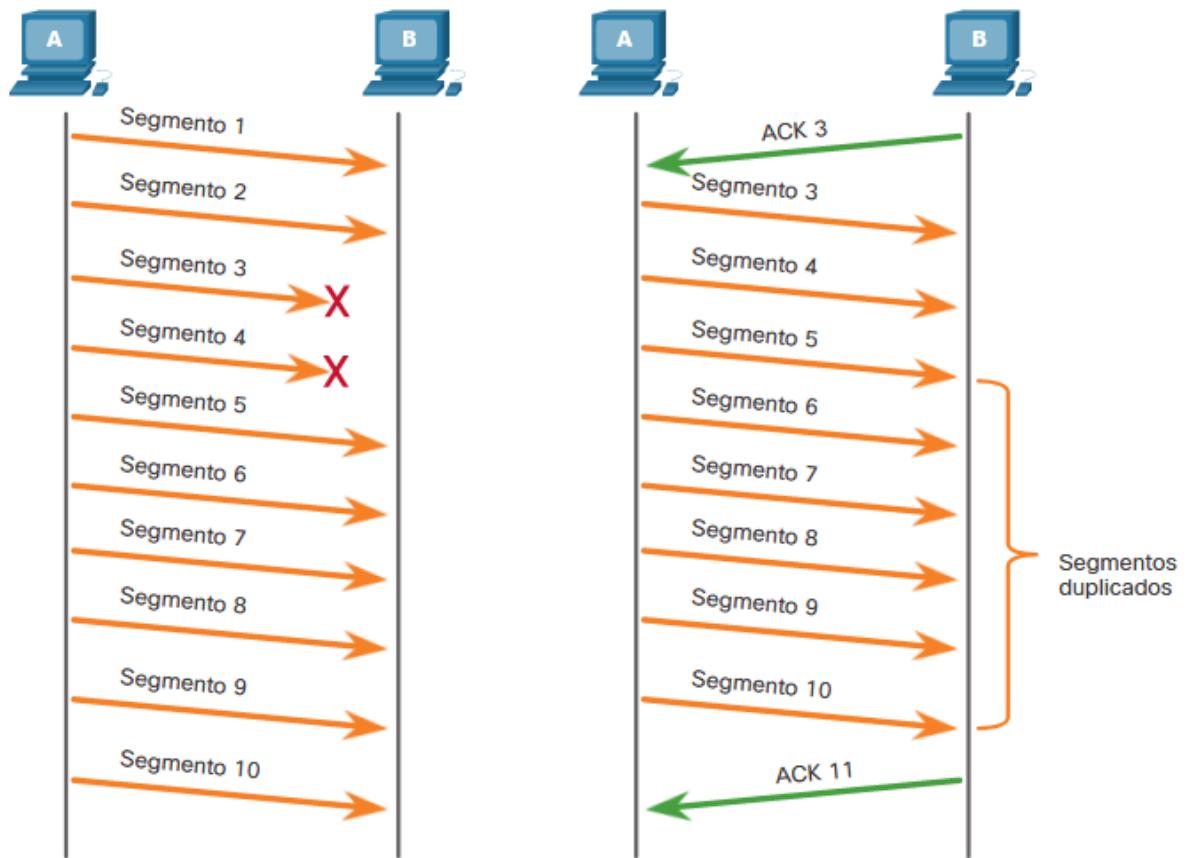


Fiabilidad de TCP: Pérdida y Retransmisión de Datos

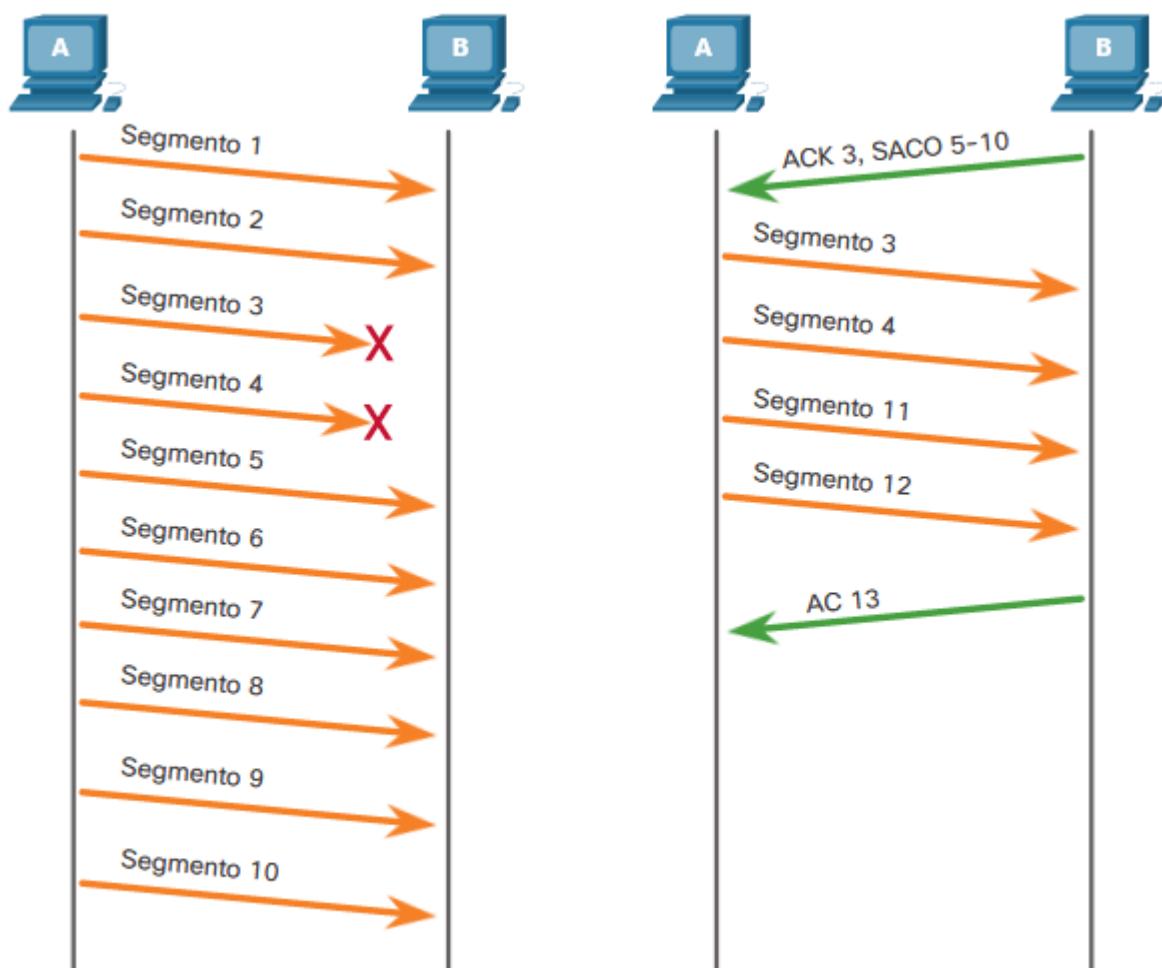
No importa cuán bien diseñada esté una red, ocasionalmente se produce la pérdida de datos. TCP proporciona métodos para administrar la pérdida de segmentos. Entre estos está un mecanismo para retransmitir segmentos para los datos sin reconocimiento.

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.

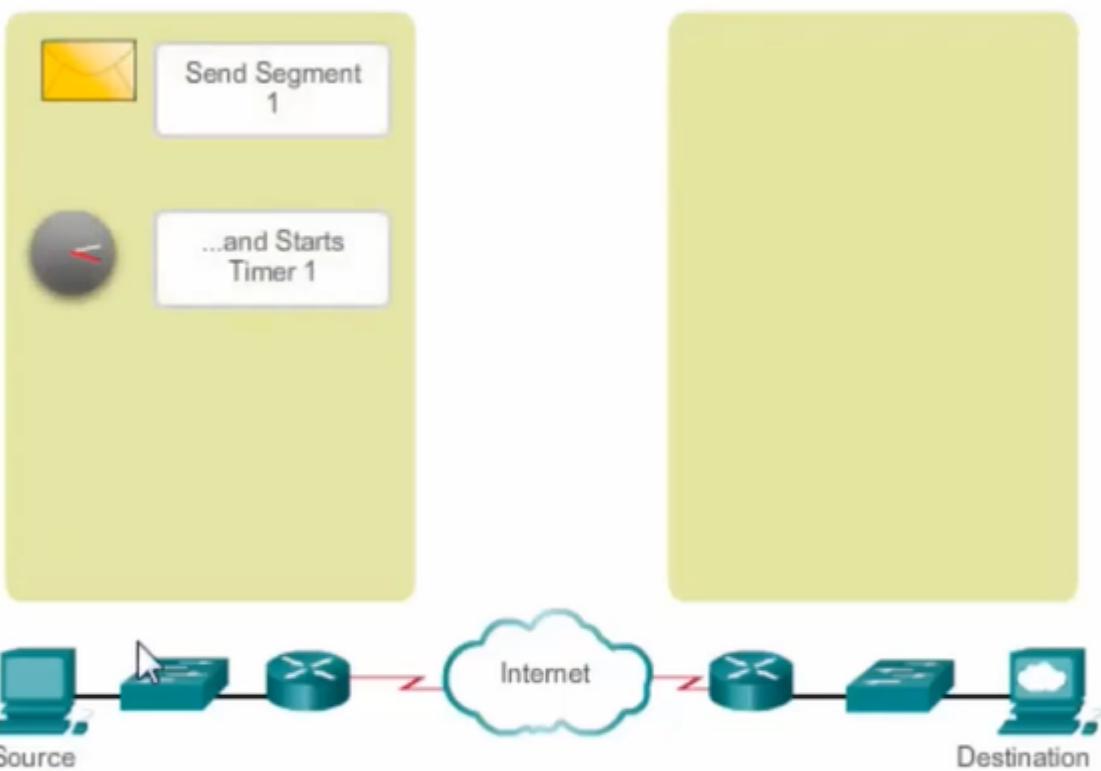
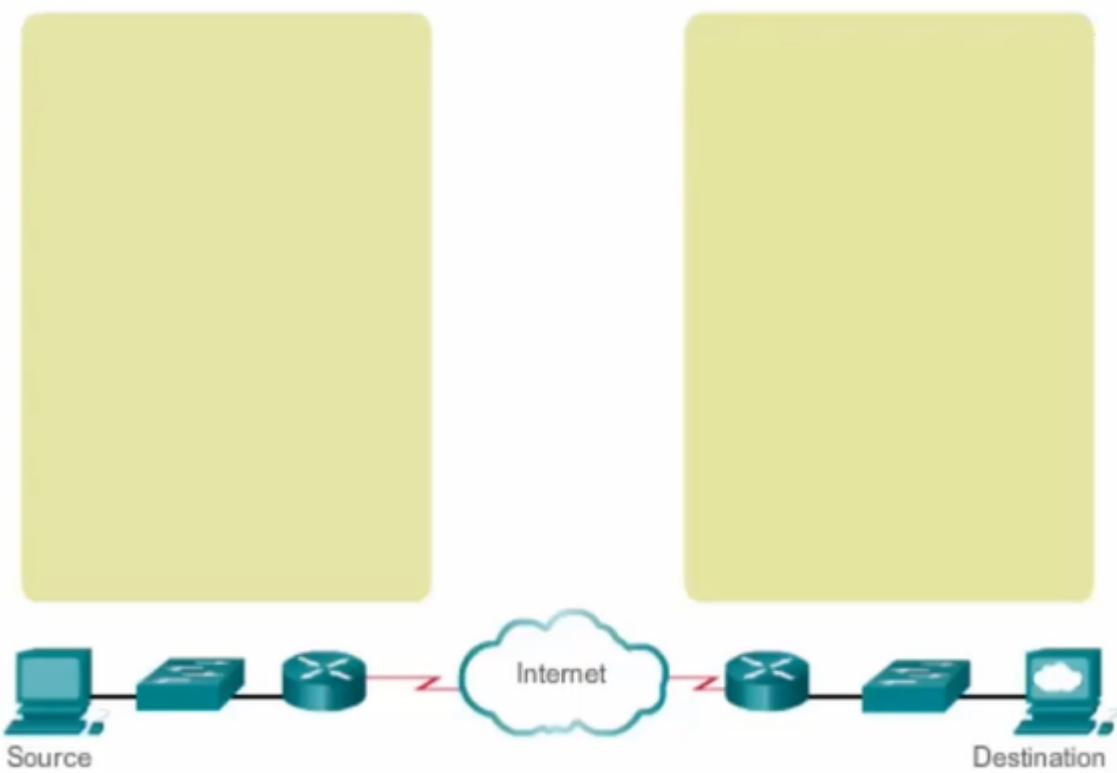
Antes de mejoras posteriores, TCP solo podía reconocer el siguiente byte esperado. Por ejemplo, en la figura, utilizando números de segmento para simplificar, el host A envía los segmentos del 1 al 10 al host B. Si llegan todos los segmentos excepto los segmentos 3 y 4, el host B respondería con acuse 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.

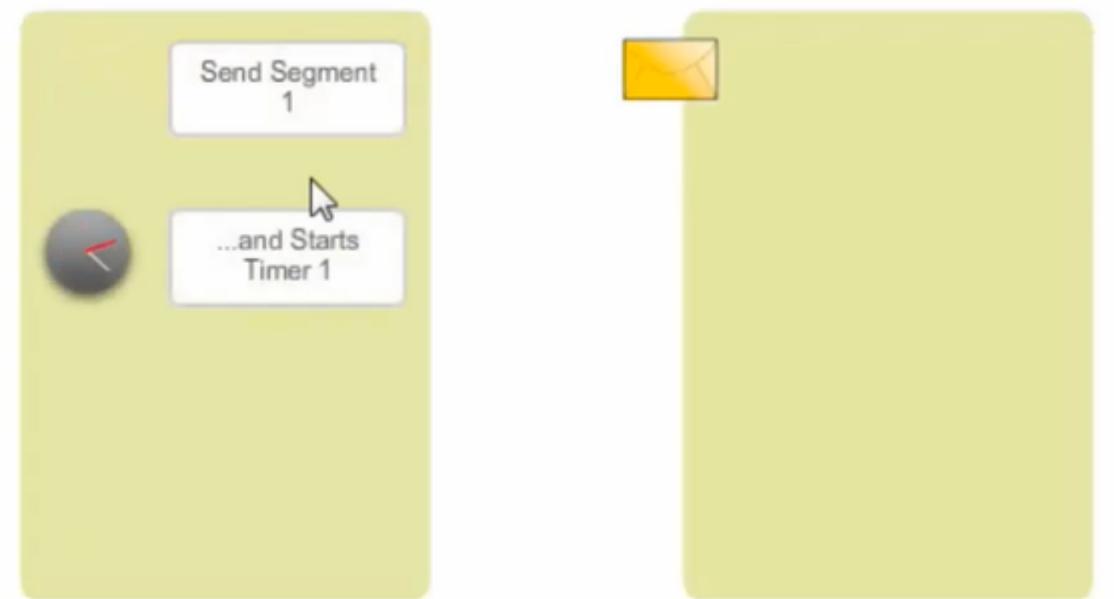


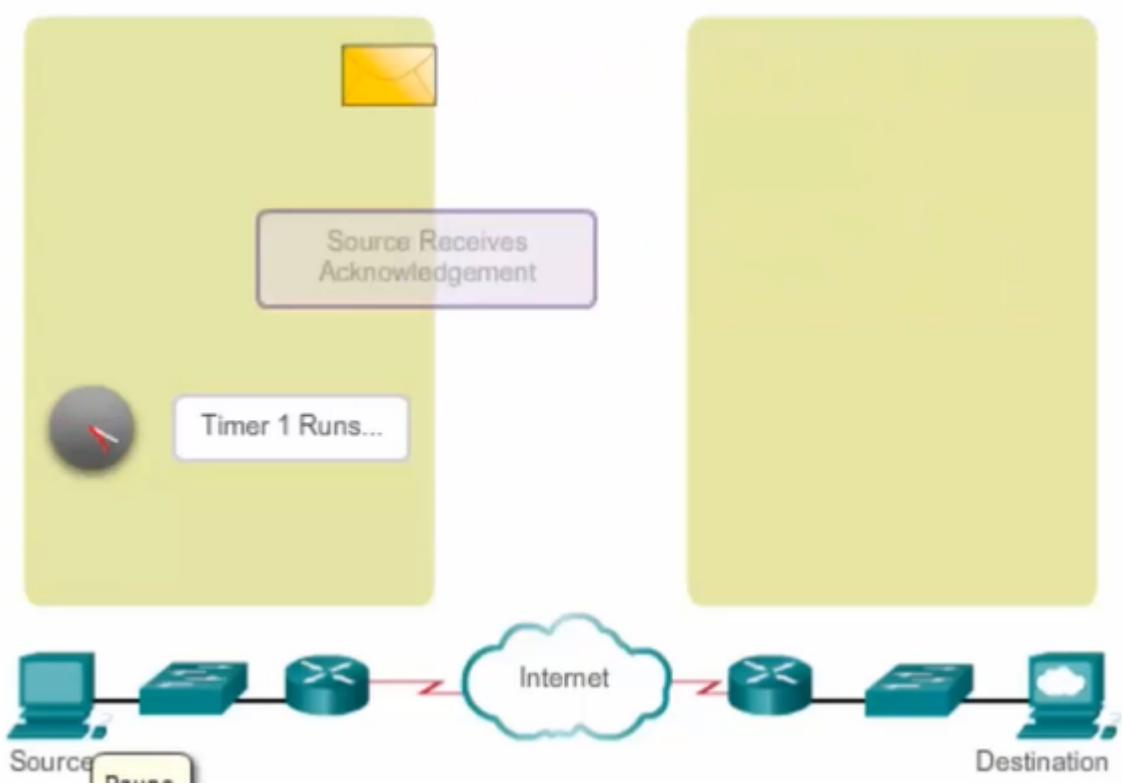
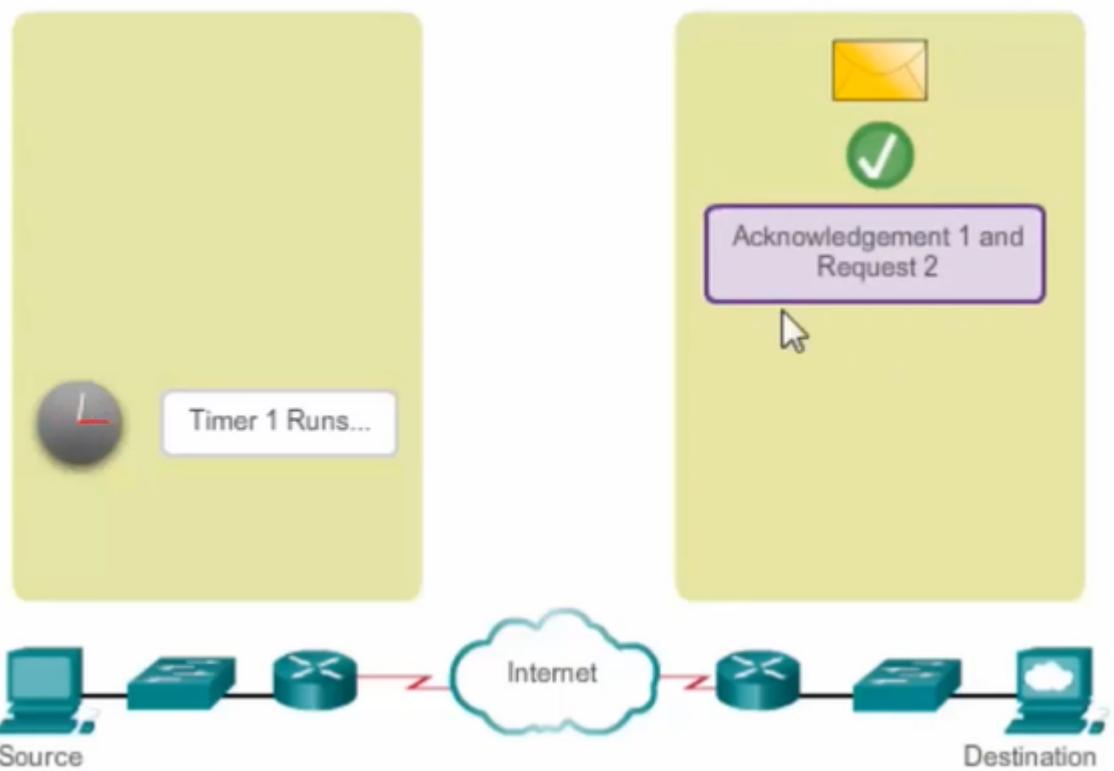
Los sistemas operativos actualmente suelen emplear una característica TCP opcional llamada reconocimiento selectivo (SACK selective acknowledgment), negociada durante el protocolo de enlace de tres vías. Si ambos hosts admiten SACK, el receptor puede reconocer explícitamente qué segmentos (bytes) se 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.

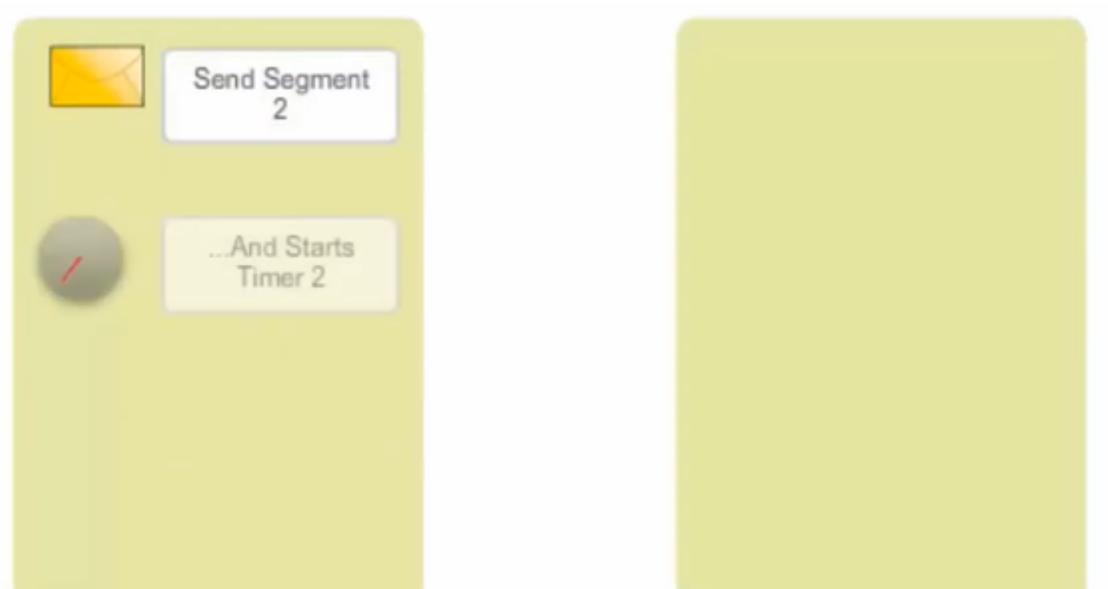
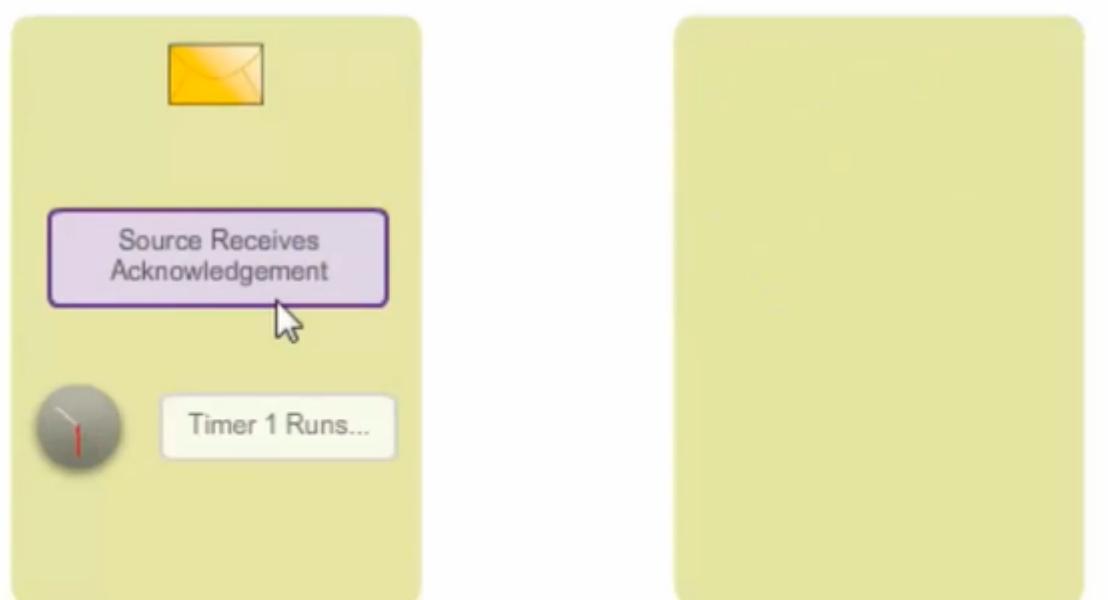


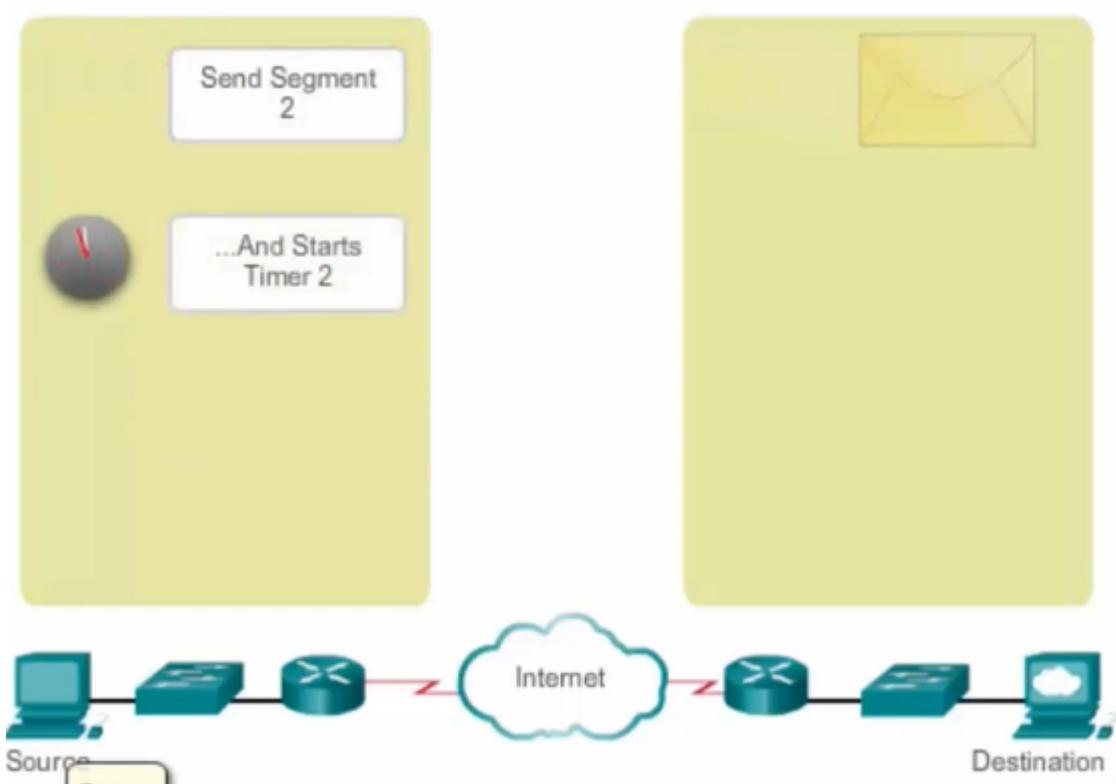
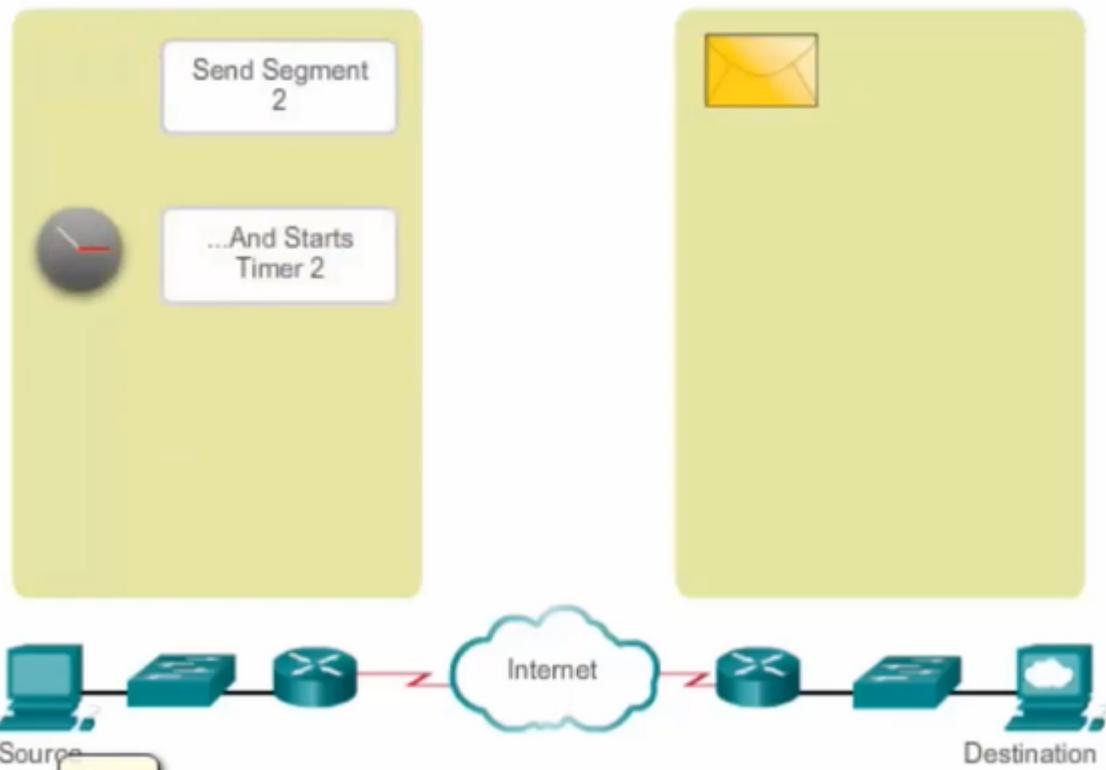
Nota: TCP normalmente envía ACK para cada otro paquete, pero otros factores más allá del alcance de este tema pueden alterar este comportamiento.

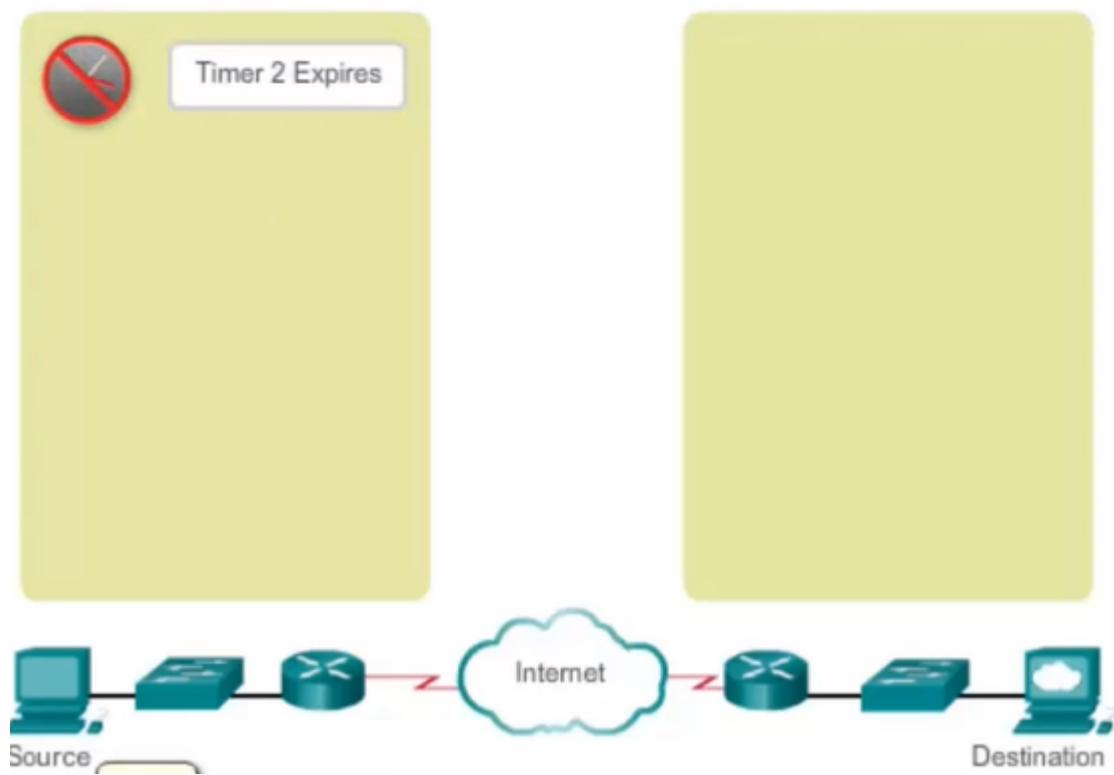
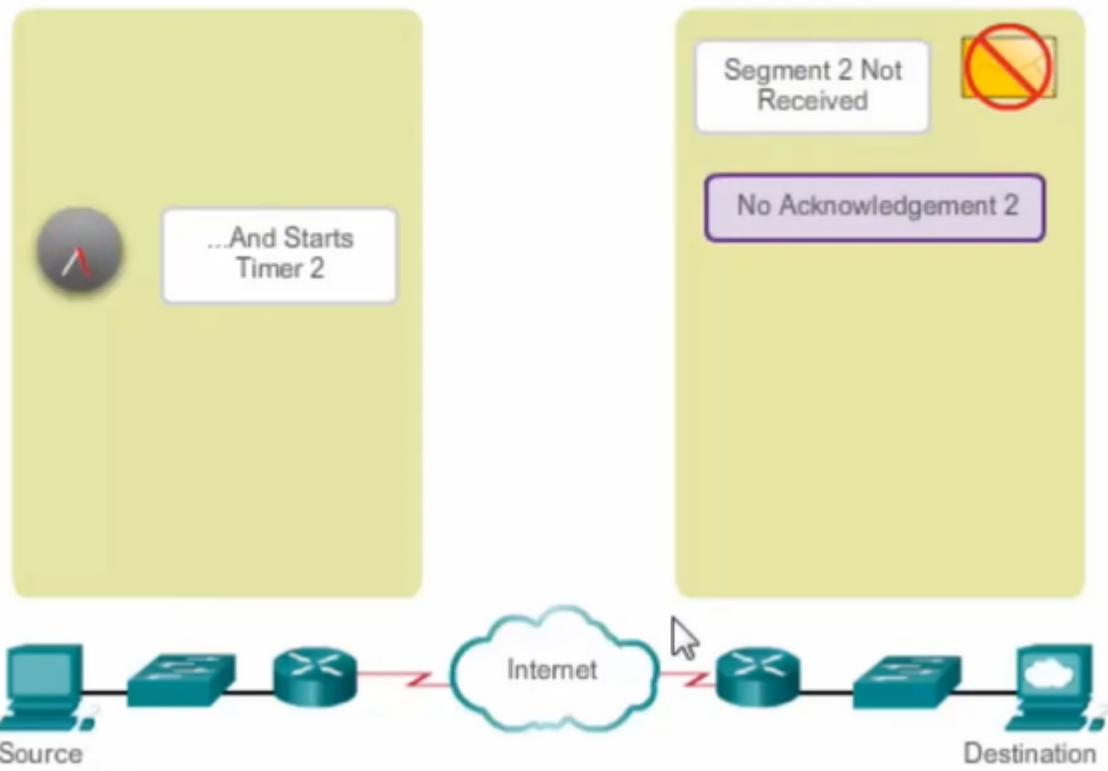


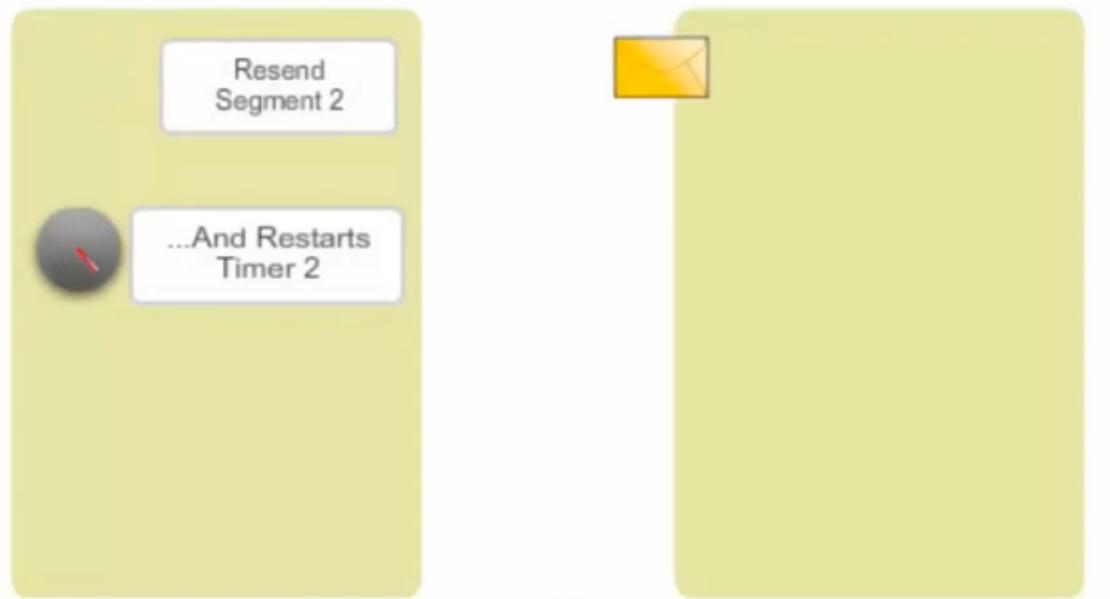


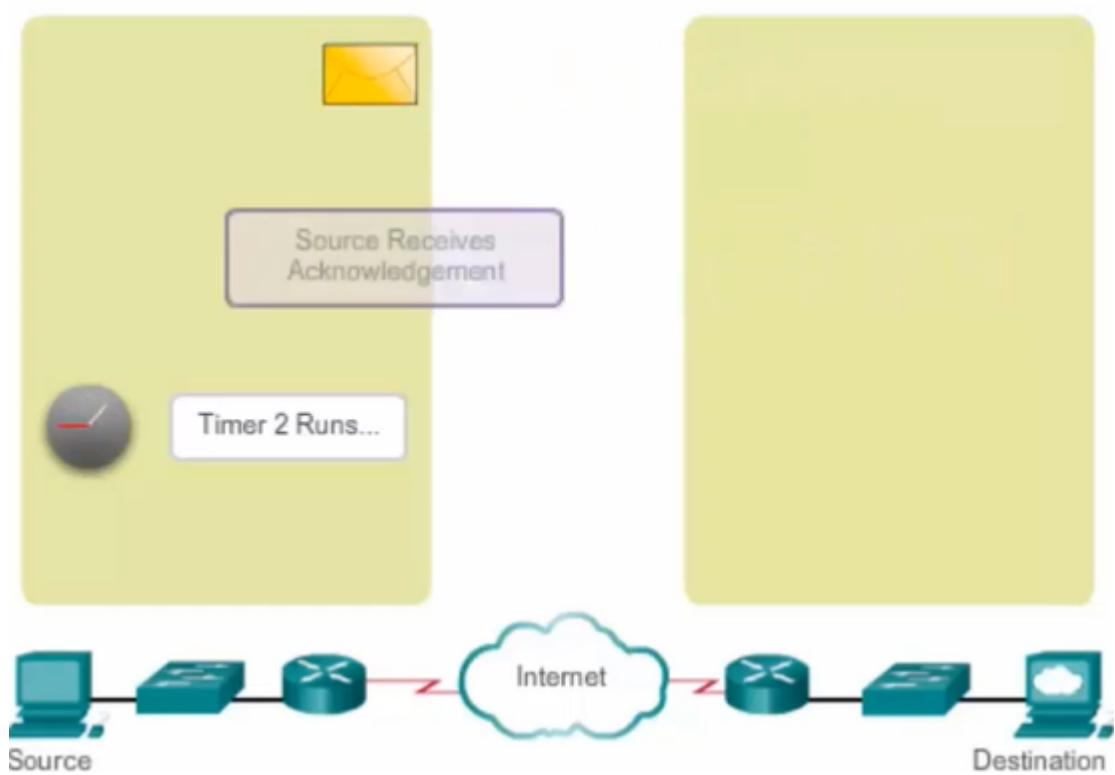
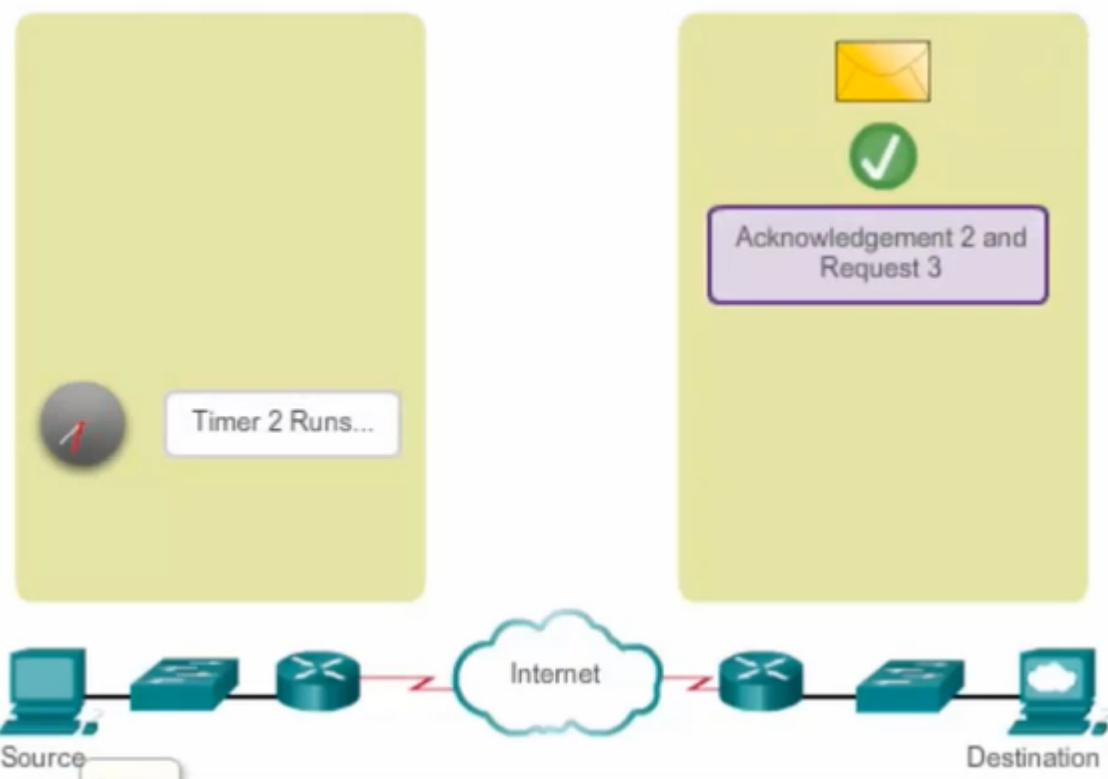


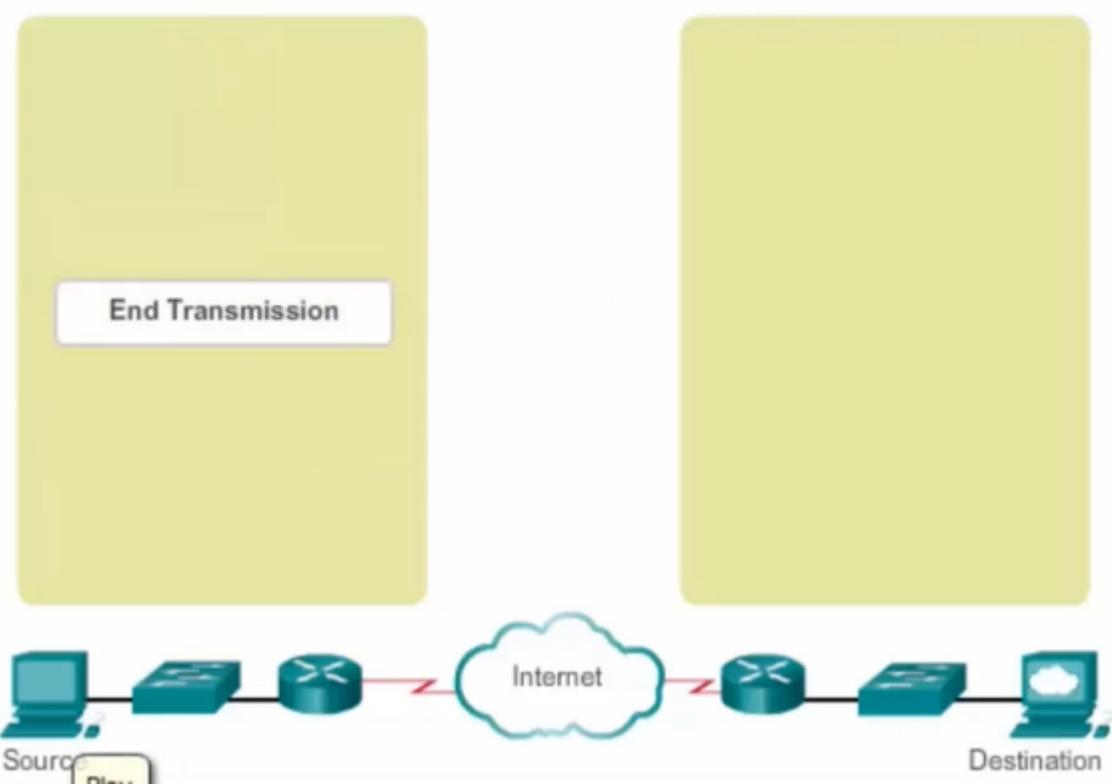
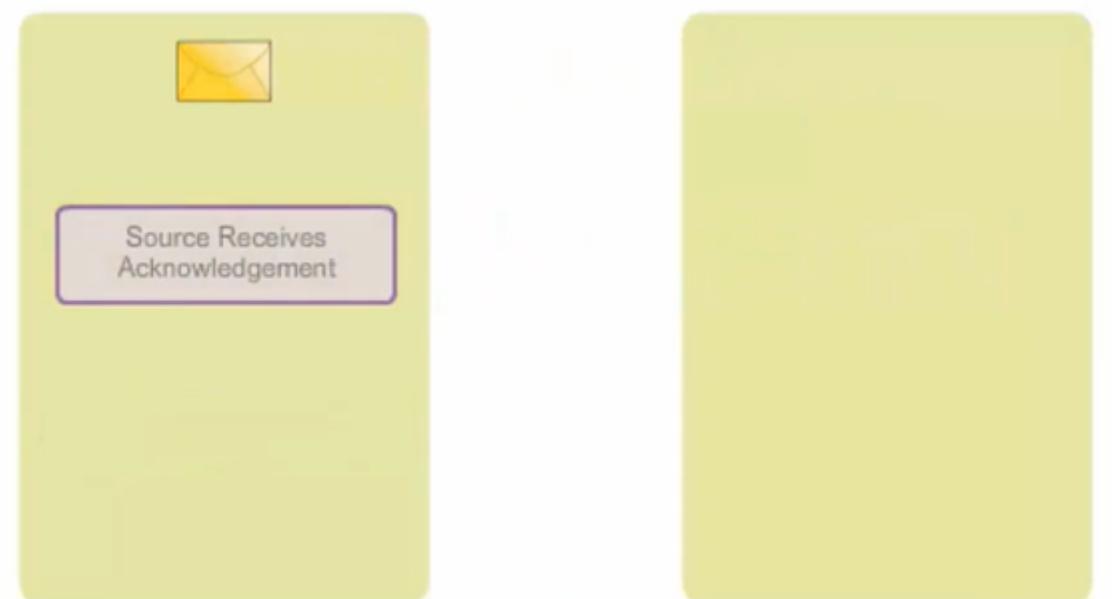










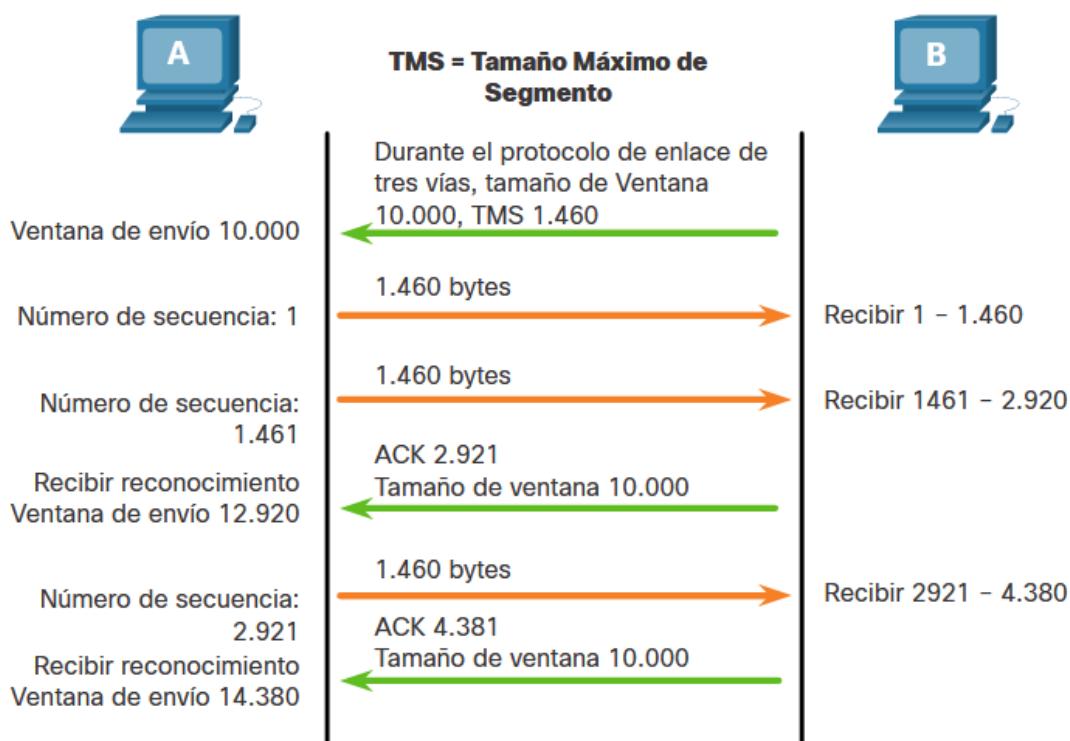


Control de Flujo de TCP: Tamaño de la Ventana y Reconocimientos

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



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.

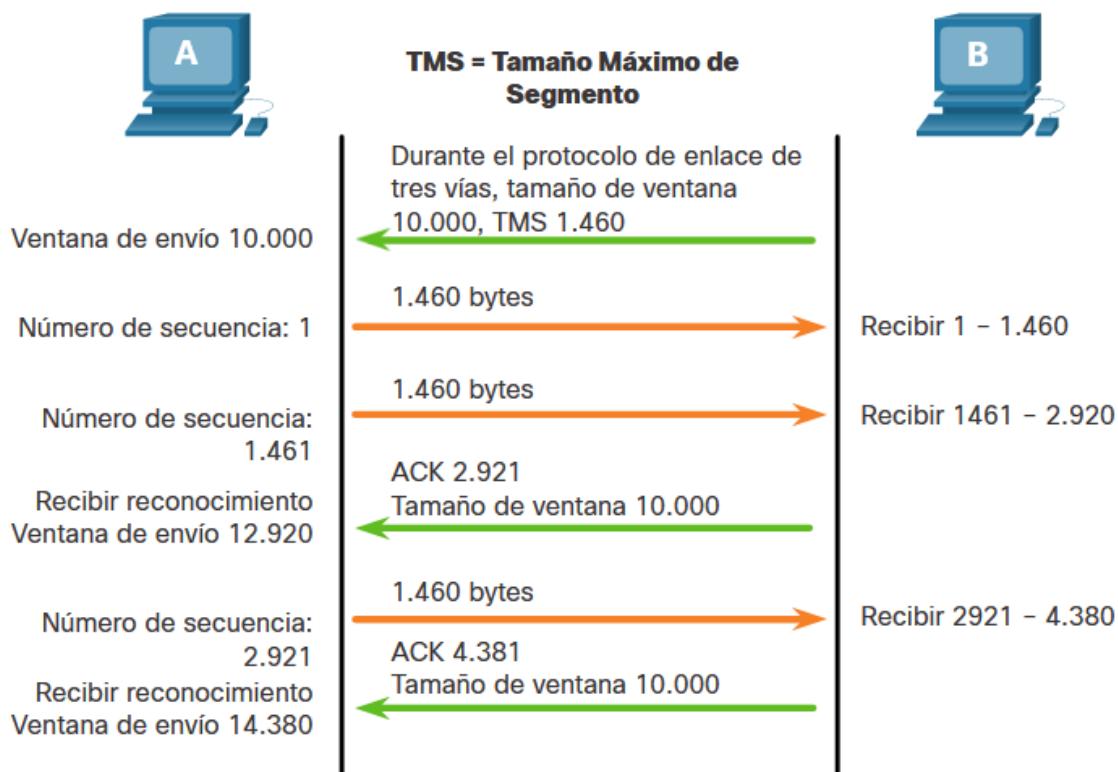
Un destino que envía confirmaciones a medida que procesa los bytes recibidos, y el ajuste continuo de la ventana de envío de origen, se conoce como ventanas deslizantes (sliding windows). En el ejemplo anterior, la ventana de envío del PC A aumenta o se desliza sobre otros 2.921 bytes de 10.000 a 12.920.

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.

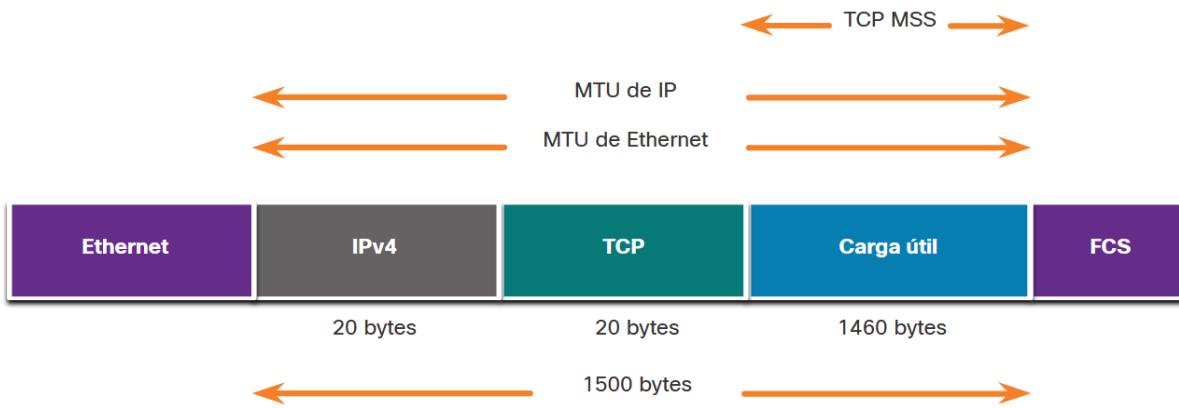
Nota: Hoy en día, los dispositivos utilizan el protocolo de ventanas deslizantes. El receptor generalmente envía un acuse de recibo después de cada dos segmentos que recibe. El número de segmentos recibidos antes de que se acuse recibo puede variar. La ventaja de las ventanas deslizantes es que permiten 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.

Control de Flujo TCP - Tamaño Máximo de Segmento (MSS Maximum Segment Size)

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.



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.



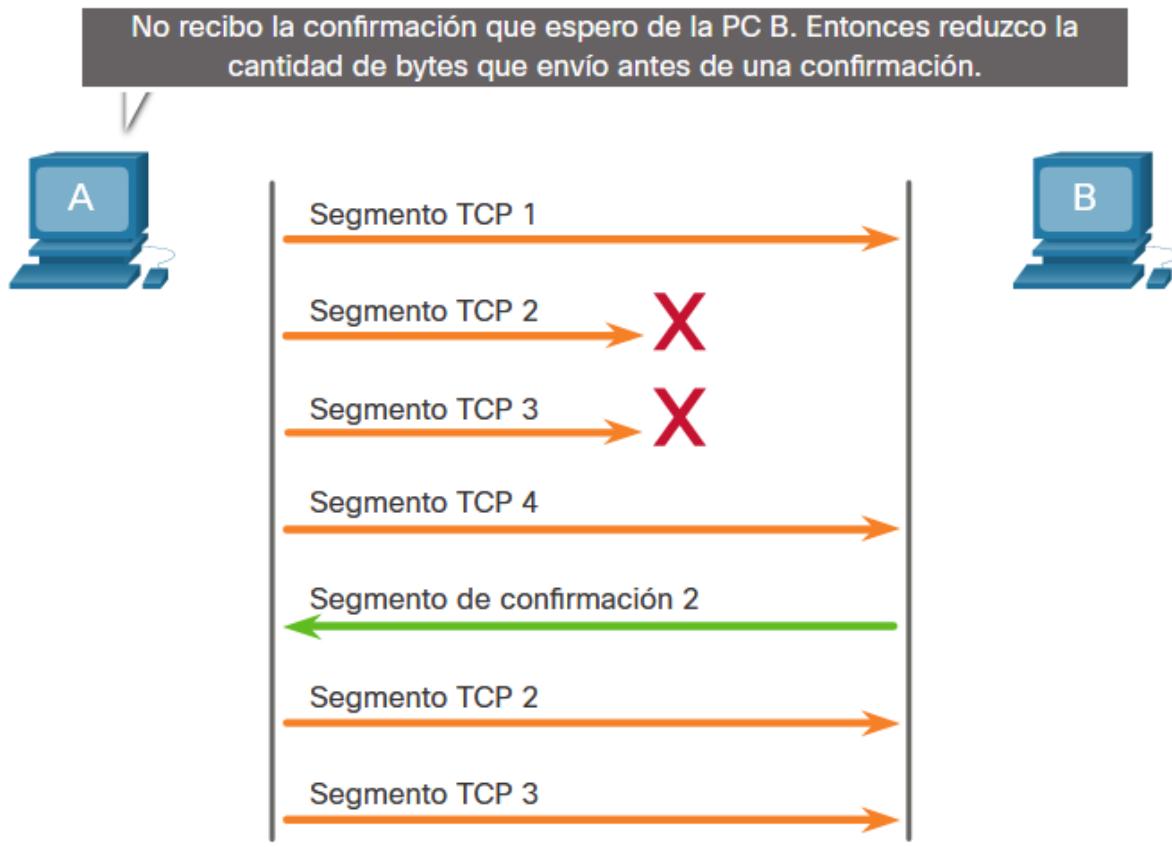
Control de flujo de TCP: Prevención de Congestiones

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



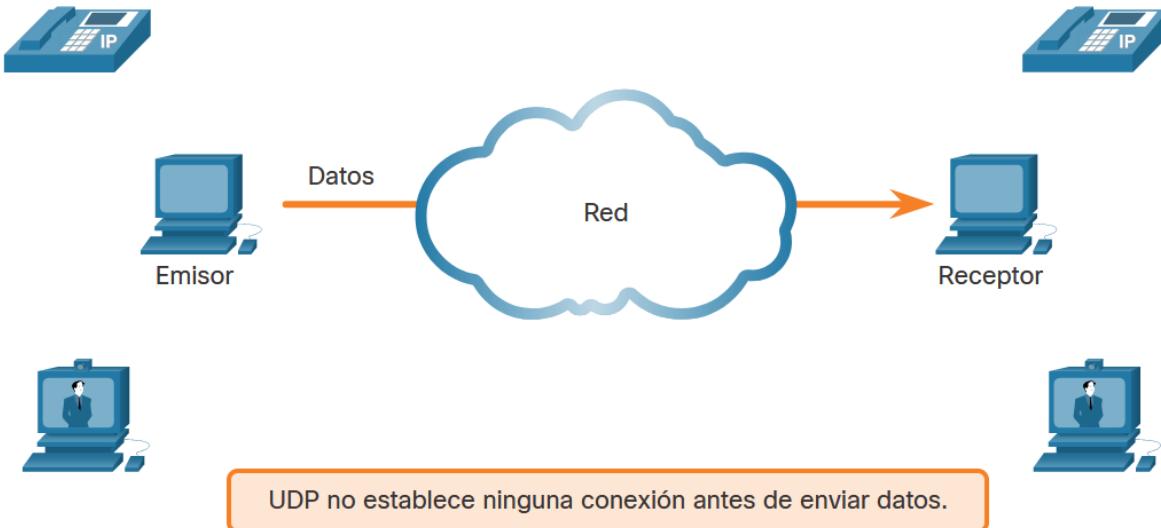
Los números de acuse de recepción corresponden al siguiente byte esperado y no a un segmento. Los números de segmentos utilizados se simplifican con fines ilustrativos.

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.

Comunicación UDP

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.



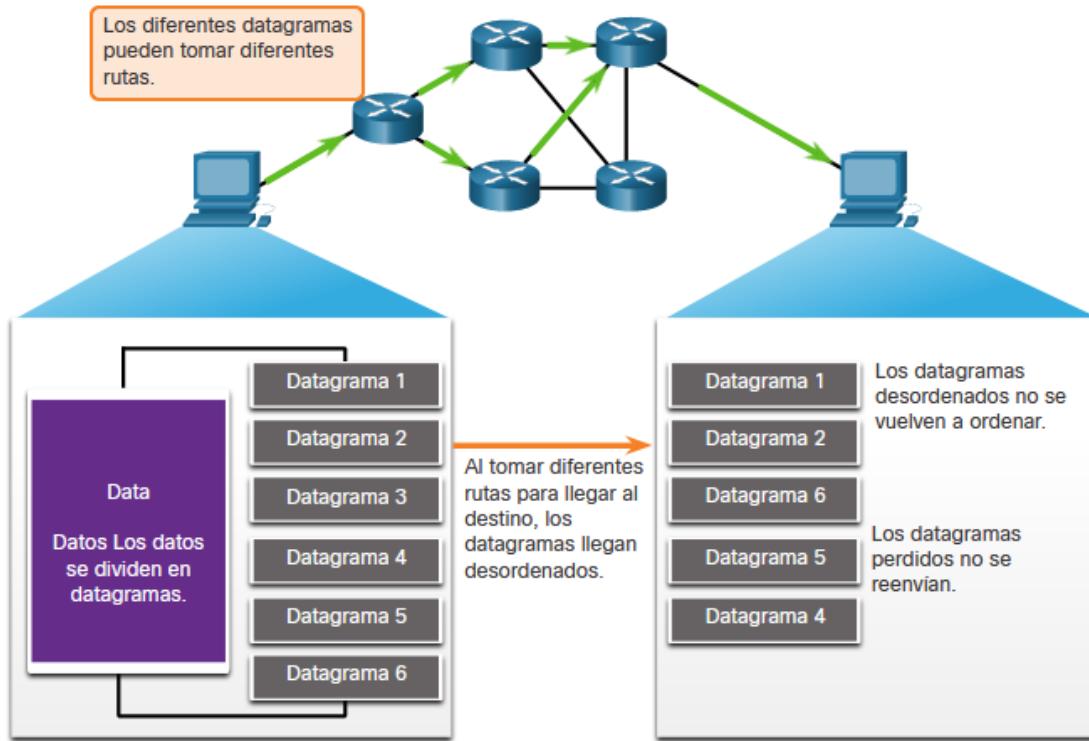
Reensamblaje de datagramas de UDP

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.

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

UDP: sin conexión y poco confiable

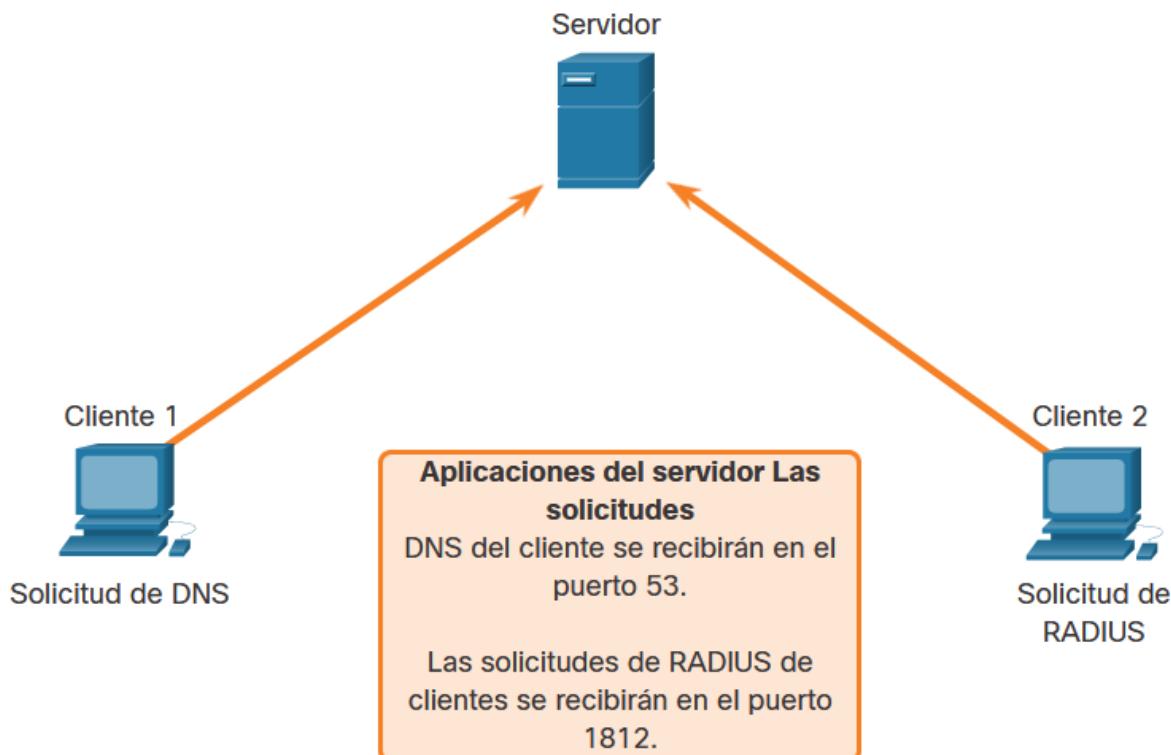


Procesos y solicitudes del servidor UDP

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.

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

Servidor UDP a la escucha de solicitudes



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.

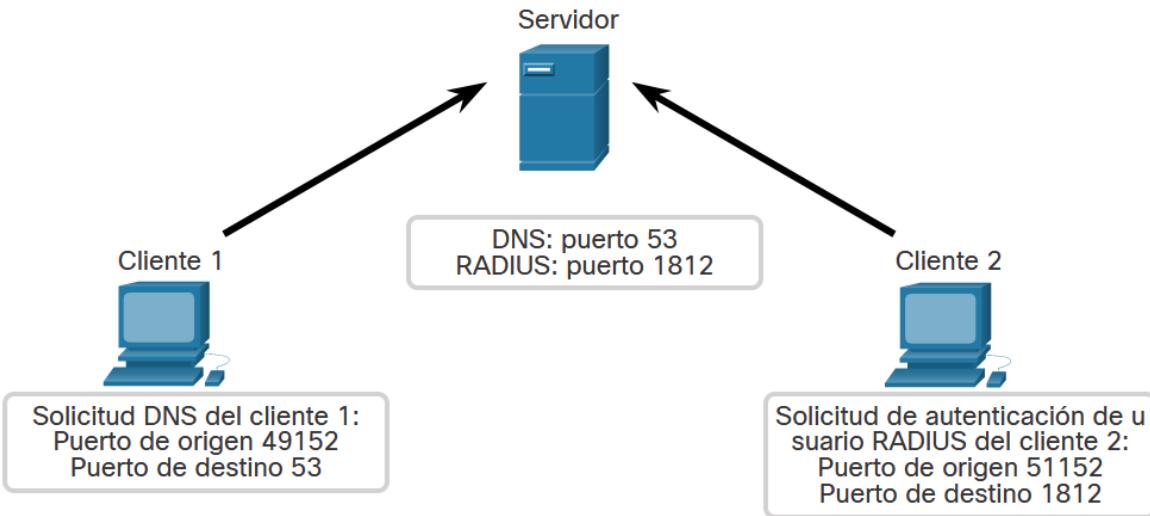
Procesos de cliente UDP

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.

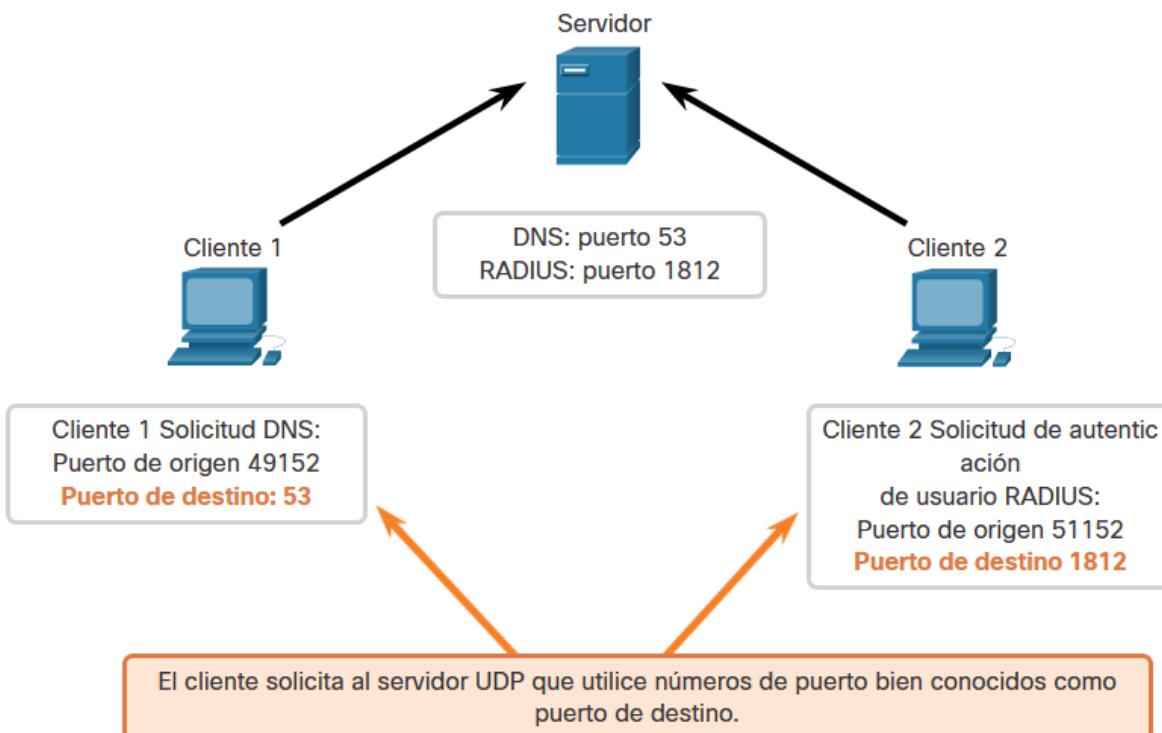
Clientes Mandando Solicitudes UDP

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 181.



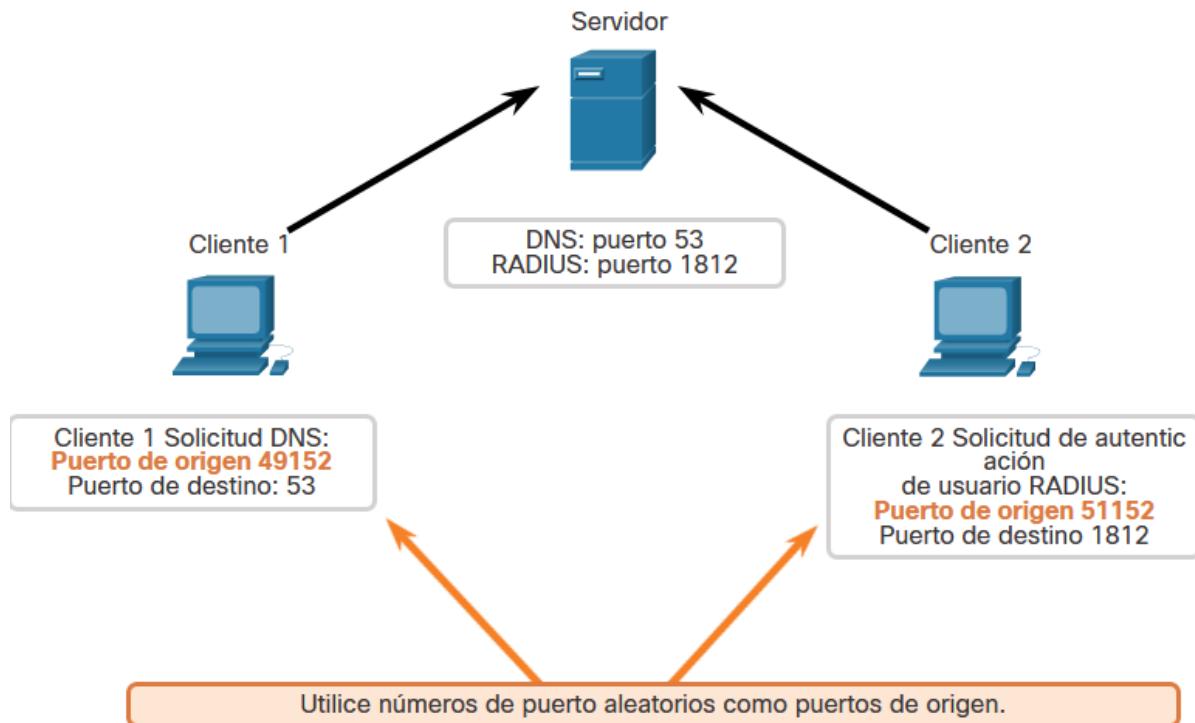
Solicitud UDP de Puertos de Origen

Las solicitudes de los clientes generan dinámicamente números de puerto de origen. En este caso, el cliente 1 está utilizando el puerto de origen 49152 y el cliente 2 está utilizando el puerto de origen 51152.



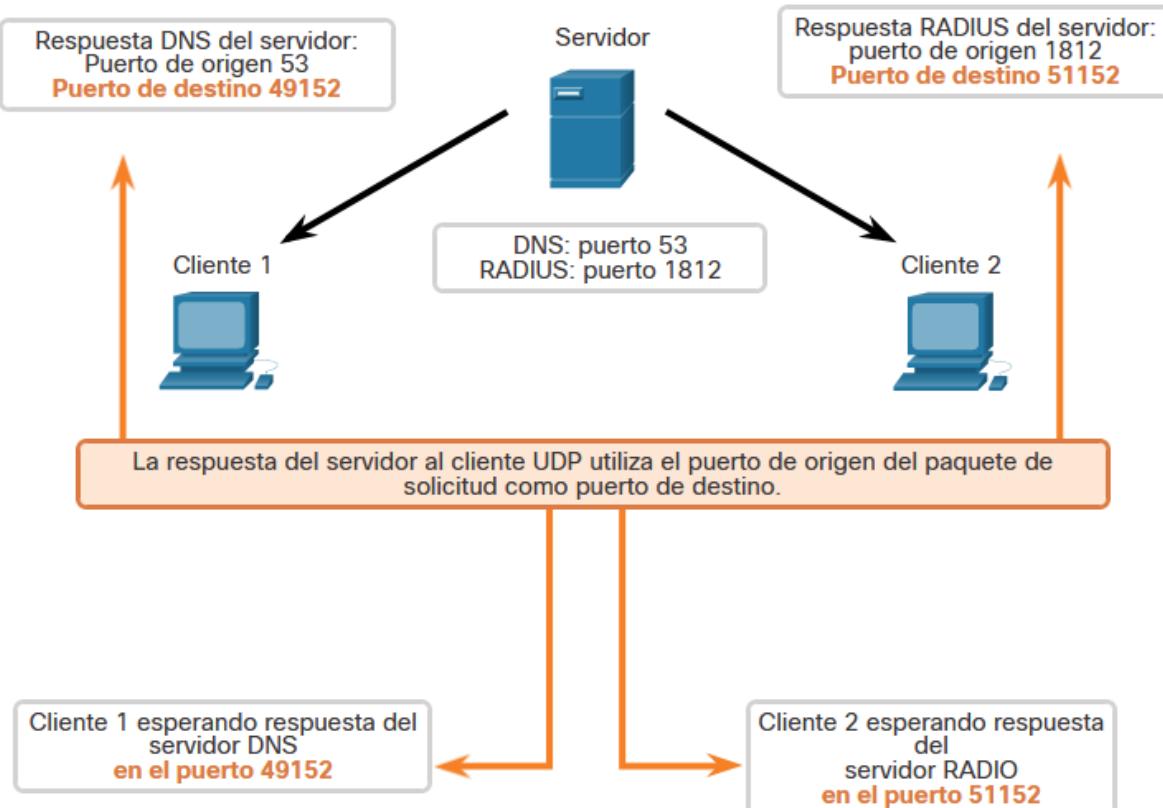
UDP Solicitud de Puertos de Origen

Cuando el servidor responde a las solicitudes del cliente, invierte los puertos de destino y origen de la solicitud inicial.



Destino de respuesta UDP

En la respuesta del servidor a la solicitud DNS ahora es el puerto de destino 49152 y la respuesta de autenticación RADIUS ahora es el puerto de destino 51152.



UDP Respuesta de Puertos de Origen

Los puertos de origen en la respuesta del servidor son los puertos de destino originales en las solicitudes iniciales.

