

## Resumen Modulo 14

### Capa de transporte

#### 14.1 Transporte de datos

• Función de la capa de transporte → Es responsable de las comunicaciones lógicas entre aplicaciones que se ejecutan en diferentes hosts. Puede incluir servicios como el establecimiento de una sesión temporal entre 2 hosts. Es el enlace entre la capa de aplicación y capas inferiores que son responsables de la transmisión a través de la red. No tiene conocimiento del tipo de host de destino, el tipo de medio por el que viajan los datos. Incluye 2 protocolos: **Protocolo de control de Transmisión (TCP)** y **Protocolo de Datagramas de usuario (UDP)**.

• Responsabilidades de la capa de transporte → Se divide en 6:

**Seguimiento de conversaciones individuales** → Cada conjunto de datos que fluye entre una aplicación de origen y una de destino. A esto se le llama conversación y es responsabilidad de la capa de transporte mantener y dar seguimiento de todas las conversaciones.

**Segmentación de datos y rearmado de segmentos** → Es responsabilidad dividir los datos de la aplicación en bloques de tamaño adecuado. Dependiendo del protocolo, los bloques son Segmentos o datagramas. Divide los datos en bloques y son fáciles de administrar y transportar.

**Agregar información de encabezado** → Agrega información de encabezado que contiene datos binarios organizados en campos a cada bloque de datos. Los valores permiten que los protocolos lleven a cabo funciones de comunicación de datos.

**Identificación de las aplicaciones** → Debe poder separar y administrar varias comunicaciones con diferentes requisitos de transporte. Para identificar y pasar el flujo de datos utiliza un identificador: número de puerto.

**Multiplex de conversaciones** → Debido a que el envío de algunos tipos de datos en la red gastan el ancho de banda disponible, la capa de transporte utiliza segmentación y multiplexación para permitir comunicaciones y conversación que se intercalen en la misma red.

• **Protocolos de capa de transporte** → IP se ocupa de la estructura, direccionamiento y routing. Los protocolos de capa de transporte especifican cómo transferir mensajes entre hosts y son responsables de administrar los requisitos de fiabilidad de una conversación. Los cuales son **TCP** y **UDP**. Los cuales tienen diferentes requisitos de confiabilidad. TCP/IP proporciona 2 protocolos para esto.

● Protocolo de control de Transmisión → Se considera un protocolo confiable y completo, que garantiza que todos los datos lleguen al destino. Incluye campos que garantizan la entrega de datos. TCP divide los datos en segmentos. Operaciones básicas:  
Enumerar y rastrear segmentos de datos transmitidos a un host específico desde una aplicación específica. Confirmar datos recibidos. Retransmitir información no reconocida después de un tiempo. Secuenciar datos para buen orden. Envío de datos a buena velocidad.

● Protocolo de Datagramas de Usuario (UDP) → Protocolo más simple que TCP. No proporciona confiabilidad ni control de flujo, requiere menos encabezados. Significa que los datagramas se puede procesar más rápidos que segmentos TCP. Es un protocolo sin conexión, no requiere conexión establecida. Conocido como protocolo sin estado. UDP no informa al emisor si la entrega si se realizó.

● Protocolo de la capa de transporte correcto para la aplicación adecuada → Esto va a depender de los requisitos de la aplicación algunas admiten cierta pérdida de datos durante una transmisión, los retrasos son **inaceptables**. En este caso UDP es la mejor opción por menos sobrecarga en la red. UDP también utilizado por aplicaciones de solicitud y respuesta donde los datos son mínimos. **DNS** utiliza UDP.

Para otras aplicaciones es importante que todos los datos lleguen y puedan ser procesados en su secuencia adecuada. TCP es el ejemplo correcto. Aplicaciones como bases de datos, navegadores web y correo. Los desarrolladores de aplicaciones deben elegir qué tipo de protocolo de transporte es adecuado a los requisitos de aplicaciones. La aplicación utiliza TCP realiza buffering, sondeo de ancho de banda, control de gestión. Por ejemplo el video y voz en tiempo real usan UDP. Una videoconferencia TCP.

UDP → (Rápido, baja sobrecarga, NO requiere reconocimiento, NO reenvía datos perdidos)  
TCP → (Confiable, reconoce datos, reenvía datos perdidos, entrega los datos en secuencia).

## 14.2 Descripción general de TCP

● Características de TCP → Establece una sesión → Protocolo orientado a conexión que negocia y establece una conexión permanente (sesión) entre origen y destino. Garantiza una entrega confiable → Asegura que cada segmento llegue al destino. Proporciona entrega en el mismo pedido → TCP garantiza que los segmentos se vuelvan a ensamblar en el orden correcto.

**Admite control de flujo** → TCP regula la cantidad de datos que se transmiten en el origen. El control de flujo puede evitar la necesidad de retransmitir los datos cuando los recursos del host receptor se ven desbordados.

● **Encabezado TCP** → Es un protocolo con estado, significa que realiza un seguimiento del estado de la sesión de comunicación. TCP registra qué información se envió y cuál se reconoció. La sesión comienza con el establecimiento de la sesión y termina con la finalización de la sesión. Un segmento TCP agrega 20 bytes de sobrecarga al encapsular los datos de la capa de aplicación.

● **Campos de encabezado TCP** → **Puerto de origen** → Campo 16 bits para identificar la aplicación de origen por número de puerto. **Puerto destino** → Campo 16 bits para identificar la aplicación de destino por número de puerto. **Siguencia de números** → Campo de 32 bits utilizado para reensamblar datos. **Número de Acuse de recibo** → Campo 32 bits utilizado para indicar qué se han recibido datos y el siguiente byte esperado de la fuente.

**Longitud del encabezado** → Campo de 4 bits indica la longitud del encabezado del segmento TCP. **Reservado** → Campo de 6 bits reservado para uso futuro. **Bits de control** → Campo de 6 bits incluye códigos de bits, indicadores, función del segmento TCP. **Tamaño de la Ventana** → Campo 16 bits indica el número de bytes que se aceptan a la vez. **Suma de Comprobación** → Campo de 16 bits usados por errores o chequeo de errores. **Urgentes** → Campo de 16 bits para indicar los datos contenidos son urgentes.

● **Aplicaciones que utilizan TCP** → TCP es un buen ejemplo de como las diferentes capas del conjunto TCP/IP tienen roles específicos. TCP maneja todas las tareas asociadas con la división del flujo de datos en segmentos. Libera la aplicación de tener que administrar estas tareas. Algunas aplicaciones de TCP son HTTP, FTP, SMTP, SSH.

### 14.3 Visión general de UDP.

● **Características de UDP** → Es un protocolo de transporte del mejor esfuerzo, transporte liviano que ofrece la misma segmentación y rearmado de datos que TCP, pero sin la confiabilidad y control del flujo TCP. Características más específicas:

- 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.

● Encabezado UDP → UDP es un protocolo sin estado, o sea ni el cliente ni el servidor rastrean el estado de la sesión de comunicación. Las aplicaciones de video y de voz en vivo pueden tolerar cierta pérdida de datos mínimo. Los bloques de comunicación UDP se denominan datagramas o segmentos. Datagramas se envían como el mejor esfuerzo por el protocolo de la capa de transporte.

Puerto origen (16) | Puerto de destino (16) | Son 8 bytes.  
Longitud (16) | Suma comprobación (16)

● Campos de encabezado UDP → Puerto de origen → Campo 16 bits para identificar la aplicación de origen de número de puerto. Puerto destino → Campo 16 bits para identificar la aplicación destino con número de puerto. Longitud → Campo 16 bits que indica la longitud del encabezado del datagrama UDP. Suma de comprobación → Campo 16 bits para comprobación de errores del encabezado y datos del datagrama.

● Aplicaciones que utilizan UDP → Aplicaciones de video y multimedia en vivo: Toleran cierta pérdida de datos, requieren poco o ningún retraso. 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. Aplicaciones que manejan la confiabilidad por sí mismas: Comunicaciones unidireccionales donde el control de flujo, detección de errores, reconocimientos y recuperación de errores no son necesarios o la aplicación puede manejarlos. Ejemplos: DHCP, envenenamiento, SNMP, TFTP, VoIP, videoconferencia.

#### 14.4 Números de puerto

● Comunicaciones múltiples separadas → UDP y TCP usan números de puerto. Estos utilizan números de puerto para administrar múltiples conversaciones simultáneas. TCP y UDP identifican un número de puerto de aplicación origen y destino. El número de puerto de origen está asociado con la aplicación de origen y así con el puerto destino. Cuando se hace una solicitud a un servidor web, el host genera dinámicamente el número de puerto para identificar la conversación. El número de puerto destino es lo que identifica el tipo de servicio que se solicita del servidor web destino. Por ejemplo puerto 80.

● Pares de sockets → Los puertos origen y destino se colocan dentro del segmento. Un socket es la combinación de la IP de origen y número de puerto de origen, o la IP destino con 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. Permiten que los procesos que se ejecutan en un cliente se distingan entre sí. También permiten la diferenciación de diferentes conexiones a un proceso de servidor.

● **grupos de números de puerto** → La IANA es la organización de estándares responsables de asignar estándares de direccionamiento, incluidos los números de puerto de 16 bits. Los 16 bits proporcionan un rango entre 0 y 65535. Los 3 grupos son:

**Puertos bien conocidos** → 0 a 1,023. Reservados para servicios comunes o populares y aplicaciones como navegadores, correo, acceso remoto de clientes.

**Puertos registrados** → 1,024 a 49,151. Asignados por IANA a entidad solicitante a utilizar con procesos o apps específicos. Aplicaciones individuales.

**Privada and/or Puertos dinámicos** → 49,152 a 65,535. Puertos efímeros; el sistema operativo del cliente asigna números de puerto dinámicamente cuando se inicia una conexión a un servicio. Número de puertos:

20 - TCP - FTP datos	67 - UDP - DHCP	161 - UDP - SNMP
21 - TCP - FTP control	68 - UDP - DHCP cliente	443 - TCP - HTTPS
22 - TCP - SSH	69 - UDP - TFTP	
23 - TCP - Telnet	80 - TCP - HTTP	
25 - TCP - SMTP	110 - TCP - POP3	
53 - UDP, TCP - DNS	143 - TCP - IMAP	

● El comando **netstat** → Netstat es una utilidad de red importante que puede usar para verificar esas conexiones. El comando netstat para enumerar los protocolos en uso, la dirección local y números de puerto, dirección extranjera y números de puertos y estado de la conexión. -n puede usar para las ip y número de puerto numérico.

#### 14.5 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. Un host que ejecuta una aplicación del servidor web y una app de transferencia de archivos no puede tener ambos en el mismo puerto como el TCP 80. Una aplicación de servidor activa asignada a un puerto específico se considera **abierta**.

Algunos procesos de TCP son:

Clients envían Solicituds TCP → se solicitan servicios web, correo, etc. al mismo servidor desde hosts.

Solicitar puertos de destino → Las solicitudes generan dinámicamente un número de puerto de origen.

Solicitar puertos de origen → Las solicitudes de cliente generan dinámicamente un número de puerto de origen.

Respuesta de Puertos de dominio → El servidor responde a las solicitudes del cliente, invirtiendo los puertos de destino y origen de la solicitud inicial.

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

El protocolo de enlace de 3 vías valida que el host de destino esté disponible para comunicarse.

● Terminación de sesión → Para cerrar sesión, se debe establecer el marcador de control de finalización (FIN) en el encabezado del segmento. Se usa un enlace de dos vías con un segmento FIN y un ACK para finalizar todas las sesiones TCP de una vía.

Paso 1. FIN → Cuando el cliente no tiene más datos para enviar 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.

Paso 3. FIN → El servidor envía un FIN al cliente para terminar la sesión servidor-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 → TCP es un protocolo full-duplex, donde cada conexión representa 2 sesiones de comunicación unidireccional. Para establecer conexión los hosts usan un enlace de 3 vías. Funciones del enlace 3 vías:

● Establece que el dispositivo destino está presente en la red.

● Verifica que el dispositivo destino tenga servicio activo y acepte solicitudes en el

el puerto destino que el cliente origen desea usar.

- Informa al dispositivo destino que el cliente origen intenta establecer una sesión en ese puerto.

Los 6 bits indicadores de control son:

**URG, ACK, PSH, RST, SYN, FIN.**

#### 14.6 Confidabilidad y control del flujo

- **Fiabilidad de TCP:** Entrega garantizada y ordenada → Durante la configuración de la sesión, se establece un número de secuencia inicial (**ISN**). Representa el valor inicial de los bytes que se transmiten a la aplicación receptora. Un seguimiento de bytes de datos permite identificar cada segmento de manera exclusiva. Gracias a esto se identifican segmentos perdidos. Los números de secuencia indican como reensamblar y reordenar segmentos recibidos.

- **Vídeo:** Confidabilidad de TCP: números de secuencia y acuses de recibo → Cada segmento TCP que se envía en una conversación tiene un número de secuencia, cada byte de datos se enumera en una lista secuencial. Esto hace que el host reconstruya los datos. Toda la recepción de bytes es mediante acuses de recibo, hay un límite en cantidad de datos que el host emisor puede enviar antes de recibir un reconocimiento del receptor. Es la cantidad es el tamaño de la ventana. Hasta (1GB)

- **Fiabilidad de TCP:** Pérdida y Retransmisión de datos → TCP proporciona métodos para administrar pérdida de segmentos. El mecanismo de retransmisión da datos sin reconocimiento. El número de secuencia (**SEQ**) y acuse de recibo (**ACK**) juntos para confirmar la recepción de bytes de datos. TCP utiliza el ACK reenviado al origen para indicar el próximo byte que el receptor espera recibir. Esto se llama **acuse de recibo de expectativa**. Existe una característica TCP llamada (**SACK**) que es un reconocimiento selectivo, negociada durante el protocolo de enlace de 3 vías. Si ambos tienen SACK el receptor reconoce que segmentos se recibieron.

- **Control de Flujo de TCP:** Tamaño de la ventana y Reconocimientos → El **control de flujos** es la cantidad de datos que el destino puede recibir y procesar de manera confiable. Es manejado por TCP. Lo hace mediante el ajuste de velocidad del flujo de datos entre origen y destino. TCP incluye un campo de 16 bytes llamado **tamaño de la ventana**.

El tamaño de la ventana determina la cantidad de bytes que se pueden enviar para recibir un reconocimiento. El tamaño inicial de la ventana se acuerda cuando se establece la sesión TCP. A medida que se reciben y procesan los bytes, el destino envía reconocimientos para informar al origen que puede continuar enviando bytes.

Una **Ventana deslizante** es cuando un destino que envía confirmaciones a medida que procesa bytes, el ajuste de ventana de envío es continuo.

● **Control de Flujo TCP - Tamaño máximo de segmento (MSS)** → Normalmente el tamaño máximo del segmento MSS es de **1.460 bytes** dentro de cada segmento TCP. Este forma parte del campo de opciones del encabezado TCP. Un dispositivo puede recibir un único segmento TCP. El tamaño MSS no incluye el encabezado TCP.

Un host determina el valor de MSS restando los encabezados IP y TCP de unidad Máxima de transmisión (MTU) de Ethernet. El MTU predeterminado es de **1500 bytes**.

● **Control de Flujo de TCP: Prevención de congestiones** → Cuando se produce congestión en la red, el router sobrecargado comienza a descartar paquetes. Mediante la determinación de la tasa a la que se envían pero no se reconocen segmentos TCP, el origen puede asumir un cierto nivel de congestión de la red. Cuando hay congestión siempre hay retransmisión de segmentos TCP perdidos del origen. Para evitar congestiones, TCP emplea mecanismos, temporizadores y algoritmos de manejo de la congestión. Si un origen determina que segmentos no son reconocidos pero no de manera oportuna, puede reducir el número de bytes que envían antes de recibir un reconocimiento. Los números de acuse de recibo corresponden al siguiente byte esperado y no a un segmento. Los números de segmentos utilizados se simplifican con fines ilustrativos.

## 14.7 Comunicación UDP

● **Comparación de baja Sobre carga y Confidabilidad de UDP** → UDP es perfecto para comunicaciones rápidas como VoIP. No establece una conexión, suministra transporte de datos con baja sobre carga debido a que posee un encabezado de datagrama pequeño sin tráfico de administración de red. UDP no establece ninguna conexión antes de enviar datos.

● **Reensamblaje de datagramas de UDP** → 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. UDP reensambla los datos en el orden en que se recibieron y los envía a la aplicación. Los datagramas pueden tomar diferentes rutas. Al tomarlas, no llegan en el mismo orden. Los datagramas perdidos no se reenvían.

• Procesos y solicitudes del Servidor UDP → A las aplicaciones de servidor basadas en UDP se les asignan números de puerto conocidos o registrados. Cuando estas aplicaciones se ejecutan en un servidor, aceptan los datos que coinciden con el número de puerto asignado. Aplicaciones del servidor las solicitudes. DNS es en el puerto 53. RADIUS proporciona servicios de autenticación, autorización y contabilidad para administrar acceso de los usuarios.

• Procesos del cliente UDP → 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. El puerto destino es el número de puerto bien conocido que se asigna al proceso de servidor.

Cientes mandando solicitudes UDP → se envía una solicitud DNS utilizando puerto 53. Solicitud UDP de puertos de origen → Las solicitudes generan dinámicamente número de puerto de origen.

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

Destino de respuesta UDP → Respuesta del Servidor a la solicitud DNS.

UDP respuesta de puertos de origen → Los puertos de origen en la respuesta del server son los puertos destino originales en solicitudes iniciales.